Universidad de las Ciencias Informáticas

Facultad 6

Título: "Componente para la captura de datos del Sistema de Información de Gobierno: Módulo Encuestas".

Trabajo de Diploma para optar por el título de

Ingeniero Informático

Autor: Yanet Rosales Morales.

Tutor: Ing. Sergio Jesús García

Co-tutor: Ing. Claudia Nuñez Sanz

JUNIO 2011

DECLARACIÓN DE AUTORÍA

Declaro ser autora 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 _____.

Yanet Rosales Morales	Ing. Sergio Jesús García	Ing. Claudia Nuñez Sanz
_____	_____	_____
Firma del Autor	Firma del Tutor	Firma de la Cotutora

DATOS DE CONTACTO

Autora:

Yanet Rosales Morales.

Universidad de las Ciencias Informáticas

e-mail: yrmorales@estudiantes.uci.cu.

Tutor:

Sergio Jesús García.

Ingeniero de las Ciencias Informáticas.

e-mail: sgarcía@uci.cu

Co-tutor:

Claudia Nuñez Sanz.

Ingeniera de las Ciencias Informáticas.

e-mail: cnunez@uci.cu

AGRADECIMIENTOS

Este es el capítulo más difícil de la tesis, donde empiezas a recordar a cuantas personas agradecerles el tiempo que han dedicado para escuchar, aconsejar y ayudar de una forma u otra. Tiempo que incluye, en mi caso, aguantar los temperamentos, los malos momentos, la terquedad y los pésimos chistes que suelen ocurrírseme. En fin, sin más rodeos:

Primero que todos a mi mami querida Raquel y a mi paciente padre César, que a sus formas han logrado que sea hoy, quien soy, brindándome su apoyo incondicional y sus rudos consejos.

A mi familia que, aunque muy, muy lejos, me han apoyado y dedicado todo el tiempo que necesitaba.

A mis amigos, que durante mi carrera han compartido conmigo los buenos y malos momentos, especialmente a mis fieles amiga Yeimy Díaz y Taimy González.

A los jueces que durante este último año me hay apoyado, guiado y sobre todo evaluado el trabajo que hoy acredita el título que se me otorga; especialmente a Yuneimy.

Al colectivo de profesores que han influido en mi formación revolucionaria y profesional.

A mis tutores que a su manera me han aconsejado y guiado.

A todos aquellos que han sido parte, de una forma u otra, en este trayecto de 5 años.

DEDICATORIA

Le dedico este momento tan importante de mi vida a las personas que le debo lo que soy:

A mi mami Raquel y a mi padre César.

A mi hermana testaruda y a mis hermosos sobrinos que han sido la alegría de mi vida.

A mi familia que siempre me apoyó.

RESUMEN

El presente trabajo está enmarcado dentro del proyecto Sistema de Información del Gobierno (SiGOB) llevado a cabo por el Centro de Tecnologías de Gestión de Datos (DATEC) en la Universidad de las Ciencias Informáticas (UCI). Es un proyecto que guiado por la arquitectura del departamento de Integración de Soluciones al cual pertenece, está concebido en varios módulos, entre los cuales se encuentra: Encuesta.

El gobierno cubano y sus respectivas oficinas en cada una de las provincias y municipios presentan dificultades a la hora de captar, consultar la información y analizar los datos, lo que genera deficiencias en la toma de decisiones. Con el propósito de solucionar las principales necesidades anteriores, el módulo Encuesta permite un diseño de encuestas heterogéneas que se adecuen a las características de las informaciones requeridas por la institución.

Con el presente trabajo de diploma se muestran los resultados del análisis, diseño, implementación y validación del componente para la captura y gestión de datos para el módulo Encuesta del proyecto SiGOB. Facilitando la toma de decisiones y el diseño de encuestas heterogéneas para todos los tipos de informaciones manejadas por el gobierno.

PALABRAS CLAVE: captura, gestión, sistemas de información.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Desde hace algunos años, las organizaciones reconocieron la importancia de administrar sus principales recursos como: la mano de obra y las materias primas. Hasta la década de los ochenta, los directivos no necesitaban saber mucho sobre cómo la información se obtenía, procesaba y distribuía en sus instituciones. La tecnología que se requería para su gestión era mínima, puesto que no se consideraba como un activo de importancia para las organizaciones. A partir de los años noventa, con el surgimiento y reforzamiento de la globalización, las economías se transformaron de industriales, a basadas en la información y el conocimiento, pasando el manejo de la información a jugar un papel fundamental dentro de las instituciones. (1)

Los entes que se encargan de las tomas de decisiones han comenzado a comprender que la información no es sólo un subproducto de la conducción empresarial, sino que también alimenta a los negocios y puede ser uno de los tantos factores críticos para la determinación del éxito o fracaso de éstos. Producto del aumento de la complejidad de las estructuras de las empresas, el cúmulo de la información generada es cada vez mayor y su administración resulta difícil. Los desafíos de hacer un uso adecuado de la información y de la tecnología, con el fin de apoyar la eficiencia y eficacia organizacional constituyen aspectos cruciales. (1)

Estos elementos, unidos a la necesidad de las organizaciones de realizar sus operaciones de manera ágil y eficiente, debido a la creciente presión competitiva a la que están sometidas, han sido los principales factores impulsores del surgimiento de los sistemas de información basados en servidores. En este contexto, es necesario establecer una infraestructura de procesamiento de información, que cuente con los componentes requeridos para proveer información adecuada, exacta y oportuna en la toma de decisiones, proporcionando un mejor servicio a los clientes. De tal manera que el sistema de información se centre en estudiar las formas para mejorar el uso de la tecnología que soporta el flujo de información dentro de la organización. (1)

El modelo Cliente/Servidor reúne las características necesarias para proveer esta infraestructura, independientemente del tamaño y complejidad de las operaciones de las organizaciones públicas o privadas y consecuentemente desempeña un papel importante en este proceso de evolución. Los servidores de aplicaciones bajo el concepto del modelo Cliente/Servidor, más conocidos por servidores Web de nueva generación, facilitan la lógica de negocio sobre la que se construyen aplicaciones y proporcionan servicios que soportan la ejecución y disponibilidad de las aplicaciones desplegadas. (2)

En Cuba, la necesidad de un enfoque integral de consolidación de información sobre el desempeño organizacional y el apoyo para la toma de decisiones, han motivado e impulsado el desarrollo de Sistemas de Información, en aras de lograr una mejor gestión institucional. Múltiples proyectos de Sistemas de Información que se llevan a cabo en Cuba son desarrollados en la Universidad de las Ciencias Informáticas (UCI), centro educacional que forma profesionales en la rama de informática y se considera una de las mayores industrias de software del país.

El Gobierno cubano y sus respectivas oficinas en cada una de las provincias y municipios presentan dificultades a la hora de captar, consultar la información y analizar los datos, lo que genera deficiencias en la toma de decisiones. Con el objetivo de apoyar este proceso, en la UCI se llevan a cabo varios proyectos, en el Centro de Tecnologías de Gestión de Datos (DATEC), el cual presenta un modelo de desarrollo basado en líneas o departamentos de producción: PostgreSQL, Almacenes, BioInformática y Soluciones Integrales. En este último se encuentran desarrollando los proyectos de Sistema Integrado de Gestión de Estadísticas (SIGE), el Generador Dinámico de Reportes y el Paquete de herramientas para la ayuda a la toma de decisiones (PATDSI).

SIGE podría considerarse como la alternativa más cercana a la solución, pero debido a que los datos manejados por el Gobierno son heterogéneos y este se encuentra diseñado para procesos estadísticos centrados en valores numéricos, no se podría aplicar a todo tipo de encuestas la misma herramienta y metodología que este posee. Debido a las deficiencias que posee SIGE, se propone el desarrollo del Sistema de Información del Gobierno (SiGOB), que permita a través de una red centralizada, un sistema bien estructurado que garantice la recopilación de la información para su gestión. Con el propósito de simplificar este proceso, SiGOB se divide en varios módulos, entre los cuales se encuentra: Encuesta. Este módulo debe permitir un diseño de encuestas heterogéneas que se adecuen a las características de las informaciones requeridas por el gobierno.

A partir de la situación descrita anteriormente, se presenta como **problema de la investigación:**

¿Cómo garantizar la captura y gestión de datos al Sistema de Información del Gobierno: Módulo Encuesta?

Por consiguiente se define como **objeto de estudio:** Sistemas de Información, enfocado en el **campo de acción:** Proceso de captura y gestión de datos en los Sistemas de Información.

Se propone como **objetivo general:** Desarrollar el componente para la captura de datos del Sistema de Información de Gobierno: Módulo Encuestas.

Se desglosa dicho objetivo general en los siguientes **objetivos específicos:**

- Analizar los preceptos teóricos y técnicos que sustentan la investigación.
- Esbozar el modelo del diseño que responderá a la arquitectura de referencia del departamento de Integración de Soluciones.
- Desarrollar el componente para la captura de datos.
- Realizar las pruebas funcionales con el fin de garantizar los indicadores de calidad en el componente desarrollado.

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

- Análisis de los conceptos básicos y técnicos implicados en el desarrollo de un componente para la captura de datos.
- Aplicación de la Ingeniería de Requisitos con el fin de evidenciar el análisis de la solución de un componente para la captura de datos.
- Descripción de la arquitectura base del sistema que responda a la arquitectura de referencia del departamento de Integración de Soluciones.
- Definición de los componentes de diseño que se ajusten a la arquitectura de referencia.
- Implementación de los componentes de diseño con el fin de lograr la obtención del componente para la captura de datos.
- Diseño de las pruebas funcionales para la validación del componente.
- Ejecución de los diseños de pruebas con el fin de garantizar los indicadores de calidad.
- Documentación y corrección de las no conformidades identificadas en la ejecución de las pruebas.

Obteniendo el **resultado esperado:**

- Componente para la captura de datos que brindará la interfaz HTTP a través de Llamadas de Procedimientos Remotos (RPC).

El presente trabajo de diploma tiene como estructura:

Capítulo 1: Fundamentación Teórica.

Este capítulo contiene los fundamentos teóricos para entender el problema a solucionar, conceptos fundamentales, tecnologías y un análisis de las herramientas utilizadas a nivel mundial para la captura de datos a través de encuestas.

Capítulo 2: Diseño de la solución.

Este capítulo describe la propuesta de solución para el problema científico de la investigación. Se traducen los requisitos a una especificación que describe cómo implementar el sistema, a través del diseño, enfocado a cómo el sistema cumple sus objetivos teniendo en cuenta los requisitos funcionales y no funcionales. Se realizan los diagramas de clases y los diagramas de interacción para los casos de uso arquitectónicamente significativos. Se explica la arquitectura utilizada y los principales patrones de diseño.

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 de la codificación mediante la realización de pruebas, garantizando la calidad del software.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Durante el presente capítulo se abordarán temas relacionados con el estado del arte de las diferentes herramientas que existen en el mundo para el diseño de encuestas, haciendo énfasis en sus características y aspectos más relevantes a través de su comparación. Se efectuará un estudio de los Sistemas de Información, mediante su clasificación y funciones que realizan, definiéndose las características del sistema a desarrollar. Se explicará brevemente la metodología seleccionada, así como el lenguaje de programación, las herramientas y tecnologías que más se adecúen para el desarrollo del tipo de aplicación que se implementará.

1.1 Sistemas de Información.

Un Sistema de Información (SI) debe servir en la empresa para captar la información que esta necesita y ponerla, con las transformaciones necesarias, en poder de aquellos miembros de la empresa que la requieran, bien sea para la toma de decisiones, para el control estratégico o también para la puesta en práctica de las decisiones adoptadas. Lo esencial de todo SI es que mediante él se va a proporcionar la información necesaria, en el momento oportuno y con la estructura adecuada, a aquellos miembros de la organización que la requieran para diversos usos. (3)

Un Sistema de información es el conjunto de elementos interrelacionados que recoge datos, los procesa y los convierte en información, para almacenarla y posteriormente distribuir entre sus usuarios. Como todo sistema, incluye también un ciclo de retroalimentación a manera de mecanismo de control mediante el cual se puede saber si se cumple con las expectativas y requerimientos de los usuarios. En toda empresa se cuenta con diversos subsistemas de información, surgidos de los diferentes departamentos, que conforman el sistema principal. (3)

Clasificación de los Sistemas de Información

Los Sistemas de Información pueden clasificarse en transaccionales, de apoyo a las decisiones y estratégicos.

Los transaccionales, se caracterizan porque a través de ellos se automatizan las tareas y procesos operativos, se puede integrar gran cantidad de información institucional para ser utilizada posteriormente por los funcionarios de nivel operativo de la organización en la toma de decisiones.

Los de apoyo a las decisiones, por su naturaleza misma apoyan la toma de decisiones repetitivas y no estructuradas, generalmente son desarrollados por el usuario final, proporcionan información de soporte para los mandos intermedios y la alta gerencia en el proceso de toma de decisiones.

Los estratégicos, su función principal no es apoyar la automatización de los procesos operativos ni proporcionar información para apoyar la toma de decisiones, son desarrollados para uso interno, para lograr ventajas competitivas a través de su implantación y uso, apoyando al nivel alto de la organización. (3)

El SI que se obtendrá es clasificado como transaccional, ya que garantizará la gestión de los procesos operativos que son llevados a cabo por el gobierno.

Funciones del Sistema de Información

1. Captación y recolección de datos.
2. Almacenamiento.
3. Tratamiento (análisis).
4. Diseminación de la información.

1. Captación y recolección de datos:

Esta función consiste en captar la información tanto externa (o relativa al entorno) como interna, y enviar dicha información a través de sistemas de comunicación a los órganos del SI encargados de reagruparla para evitar duplicidades e información no válidas. El quién o quienes deben captar esa información es algo que dependerá del tipo de empresa y de la forma de la organización de la captación de la información. El proceso de captación y recolección de datos debe realizarse de manera continuada en aquellas áreas o partes del entorno y empresas sujetas a cambios, de forma más puntual, en aquellas otras áreas más estables. (3)

2. Almacenamiento.

Una vez que la información haya sido recolectada y filtrada o eliminada la redundante, se procederá al almacenamiento de la misma. La información puede ser almacenada en un lugar único pero accesible a todos los posibles usuarios, también puede ser almacenada en diversos servicios o departamentos pero igualmente accesible a los usuarios. La idoneidad de una u otra forma de almacenamiento es algo a determinar por la propia empresa en función de diversas variables. El soporte físico puede ser así mismo diverso, desde un archivador-clasificador clásico, hasta una base de datos de tratamiento informático. Finalmente la forma de establecer el acceso o recuperación de la información por parte de los usuarios es igualmente múltiple. Es muy común para el almacenamiento de la información el uso de las bases de datos y los almacenes de datos (*datawarehouses*). (3)

3. Tratamiento y análisis de la información.

El tratamiento de la información almacenada tiene por objeto su transformación en una información útil y significativa para quien la utilice. Se efectúa esencialmente en el subsistema informático, que a través del empleo de paquetes de programas adecuados, permite un diálogo o interacción hombre-máquina u hombre-máquina-hombre, mediante el que se desarrolla dicho tratamiento. La fuerte reducción de los costes de los equipos informáticos, han hecho posible que, por un lado, el volumen de datos almacenados y procesados se incremente cada vez más. Por otro lado, al disminuir el coste de los equipos informáticos permite la generalización de este instrumento, sin embargo, la presencia del hombre en el análisis de la información es fundamental, pues aunque las máquinas han evolucionado mucho, la inteligencia humana es insustituible. (3)

4. Distribución y diseminación de la información.

El SI no sólo debe proporcionar la información que cada usuario requiera, sino que también debe difundir la información a otras personas dentro de la empresa. El por qué de ello se halla en la necesidad de que determinadas informaciones acerca de la empresa y del entorno sean conocidos por diferentes miembros de la empresa, a fin de poder hacer frente con mayor rapidez y éxito a las situaciones que cada día se les presentan y en las que se hace necesaria la resolución de problemas o adopción de decisiones. (3)

El SI que se obtendrá, de las funciones anteriormente explicadas, garantizará la captura y recolección de datos guiado por las características que describe la funcionalidad.

1.2 Análisis de herramientas existentes para el diseño de encuestas.

En la actualidad la palabra “encuesta” generalmente se usa para describir un método de obtener información de una muestra de individuos. La encuesta es quizás el instrumento más conocido y utilizado por los investigadores sociales cuando se quiere lograr precisión y representatividad. Conceptualmente se puede considerar como una estrategia para obtener información de una población mediante entrevistas a una muestra representativa. La información se recoge de forma estructurada formulando las mismas preguntas y en el mismo orden a cada uno de los encuestados. (4)

El desarrollo de las Tecnologías de Información y Comunicaciones (TIC) es la principal fuente de apoyo en el avance de las herramientas para el diseño de encuestas, con el objetivo fundamental de ayudar en el tiempo y el manejo de informaciones valiosas para la toma de decisiones de instituciones, empresas u otras entidades.

La realización de encuestas a través de Internet resulta muy apropiada en un mundo en el que el número de personas que se conectan a este es muy elevado y crece rápidamente. Utilizando el espacio de Internet, se puede realizar un mayor número de encuestas a consumidores, del que se realizaría si se utilizara el teléfono o el correo electrónico, obteniendo de este modo resultados más fiables y pertinentes. Por otra parte, estas encuestas resultan menos costosas y más rápidas. Además, múltiples herramientas para su diseño tienen como principal objetivo facilitar su cumplimiento a través de un sistema de información, pero a la vez tienen como desventaja que son desarrolladas para diseños característicos de las propias organizaciones. Estos sistemas de información para encuestas deben permitir realizar en el menor tiempo posible, el proceso de captura y registro de los datos de la encuesta, asegurando la confidencialidad necesaria de los datos. La mayoría de las organizaciones frente al desarrollo de la economía y la competencia a la que están sometidas, han desplegado sus proyectos en software privativos, dificultando la reutilización de su código. (4)

En la actualidad existen varias herramientas para el diseño de encuestas que tienen generalmente el objetivo de proporcionar una arquitectura abierta de soluciones con la flexibilidad, escalabilidad, desempeño y solidez necesarios para administrar una plataforma confiable de integración, a través de las fronteras empresariales. Entre ellas se destacan: CSPro, InfoPath, SPSS y SAS.

CSPro

El Censo y Sistema de Procesamiento de la Encuesta (CSPRO - *Census and Survey Processing System*) fue diseñado para ser usado en el diseño de encuestas para censos. La empresa desarrolladora SERPRO S.A. lanzó la última versión 4.0 en el año 2008, con una licencia de dominio público y está implementado en la plataforma Windows.

CSPRO ha sido diseñado con una estructura abierta, lo que permite su ampliación incorporando unidades de control, adquisición, análisis y procesamiento de datos que pueden ser desarrollados usando el editor de CSPRO. Permite capturar, corregir y producir tabulaciones de datos censales o de una encuesta. Desarrollado en C# bajo la plataforma del software Visual Studio. NET de Microsoft. (5)

Entre sus características se encuentran:

- ✓ Ingresa, recupera o modifica casos con la misma aplicación.
- ✓ Verifica datos con comparación y corrección inmediata de errores.
- ✓ Ilimitado números de formularios de entrada de datos.
- ✓ Los formularios pueden contener campos de distintos registros.

- ✓ Los formularios pueden contener matrices de dos dimensiones para ingreso de registros de concurrencia múltiple.
- ✓ Mensajes de error diseñados por el usuario.
- ✓ Aplicación de entrada de datos que puede ser ejecutada en batch[1] para generación de errores finales.
- ✓ Producción automática de estadística de digitadores.
- ✓ Diseñador gráfico amistoso para aplicaciones.

InfoPath

Microsoft Office InfoPath 2007 es un programa de recopilación de información incluido en la versión 2007 Microsoft Office system. Office InfoPath 2007 permite crear e implementar soluciones de formularios electrónicos para recopilar información de un modo eficiente y confiable. Se integra fácilmente en los sistemas y aplicaciones actuales, gracias al uso de estándares industriales que permiten automatizar los procesos empresariales existentes, sin tener que reinventar los mismos.

Las herramientas avanzadas de diseño y desarrollo de formularios permiten a Office InfoPath 2007 simplificar procesos complejos mediante el uso de formularios dinámicos e interactivos que conecten a los usuarios con los datos correctos. El formato nativo con el que InfoPath trabaja es XML, de tal manera que cualquier formato que se diseña y del cual se recolecta información se obtendrá en dicho formato, para su sencilla distribución y conexión con cualquier otra aplicación de negocio que soporte comunicación vía servicios Web y soporte XML. (6)

Entre las principales características de InfoPath se encuentran:

- ✓ Desarrolla y administra soluciones de manera más fácil: InfoPath ofrece todo un ambiente de desarrollo robusto para crear y adaptar rápidamente formularios de captura de información. El ambiente de desarrollo está enfocado a facilitar la creación de formularios de captura que permita a la organización un cierto nivel de flexibilidad para construir rápidamente soluciones que admitan recopilar información de cualquier tipo. En algunos casos integrarla a los procesos o sistemas, como consecuencia la organización tendrá una tecnología que le permitirá ser más flexible a la hora de responder con agilidad a los cambios que el entorno demanda.

[1] Batch: En DOS, OS/2 y Microsoft Windows batch es un archivo de procesamiento por lotes. Se trata de archivos de texto sin formato, guardados con la extensión BAT que contienen un conjunto de comandos MS-DOS.

- ✓ Conecta personas, información y sistemas: InfoPath ofrece soporte para habilitar escenarios de integración y comunicación con sistemas mediante el uso de servicios Web, ofreciendo la capacidad de obtener de otras fuentes de datos información para reforzar la integridad de los datos y también para integrar en los sistemas actuales nueva información recopilada.
- ✓ Captura información de manera flexible y clara: InfoPath provee de una experiencia de captura muy similar a la de Microsoft Word con el valor adicional de mantener una estructura de datos para la captura que puede contener validaciones de datos, comparaciones, filtrados, eventos, conexiones, búsquedas, etc. sin tener que escribir una sola línea de código de programación.

SPSS

Paquete Estadístico para las Ciencias Sociales (SPSS - *Statistical Package for the Social Sciences*) es un programa estadístico informático muy usado en las ciencias sociales y las empresas de investigación de mercado. Como programa estadístico es muy utilizado debido a la capacidad de trabajar con bases de datos de gran tamaño, además de permitir la re-codificación de las variables y registros según las necesidades del usuario. Fue adquirido por la compañía IBM en el año 2009, es soportado en varios sistemas operativos entre ellos Windows, Macintosh, Linux y Unix. (7)

La última versión liberada de SPSS Statistics es la 19.0 con grandes mejoras de rendimiento, procedimientos y funciones más fáciles de usar, simplificando y acelerando sus tareas analíticas.

Entre las principales características de SPSS se encuentran:

- ✓ Soluciona todo el proceso analítico desde la planificación y la preparación de los datos al análisis, informes y distribución.
- ✓ Ofrece una funcionalidad adaptada e interfaces personalizadas para los distintos niveles de conocimiento y responsabilidades funcionales de los usuarios comerciales, analistas y estadígrafos.
- ✓ Incluye opciones flexibles de distribución desde la versión independiente de escritorio a las versiones para servidor de empresa.
- ✓ Funciona con todos los tipos de datos comunes, lenguajes de programación externos, sistemas operativos y tipos de archivos.
- ✓ Ofrece una amplia gama de técnicas especializadas para acelerar la productividad y aumentar la eficacia.

SAS

En un mercado tan competitivo como el actual, las organizaciones deben focalizar sus escasos recursos en las estrategias más adecuadas para conducir a la compañía al éxito. El software de Sistema de Análisis Estadístico (SAS - *Statistical Analysis System*) ayuda a conseguir este objetivo, completando la inversión ya realizada en sistemas operativos.

SAS es un sistema de entrega de información que provee acceso transparente a cualquier fuente de datos, incluyendo archivos planos, jerárquicos y los más importantes proveedores de bases de datos relacionales. También incluye su propia base de datos para almacenar y manejar la información, además de un motor robusto e integrado de transformación de los datos, que permite antes de su utilización en la toma de decisiones, aplicar cierta lógica desde diferentes ambientes operacionales. Cuenta con herramientas de primera clase para análisis e inteligencia de negocios, que incluyen análisis multidimensional, consultas y reportes, minería de datos y otros. Soporta un amplio rango de aplicaciones, destacándose los siguientes: análisis estadístico, análisis gráfico de datos, análisis de datos guiado, mejoramiento de la calidad, diseño experimental, administración de proyectos, programación lineal y no-lineal, generación de reportes y gráficas, manipulación y despliegue de imágenes, Sistemas de Información geográfica, visualización multidimensional de datos, aplicaciones de multimedia y Sistemas de Información ejecutiva. (8)

Recoge información de diversas formas:

- ✓ A través del registro, encuestas y otros formularios en línea.
- ✓ Como parte de un proceso de ventas continuo.
- ✓ Mientras se facilita soporte técnico, consultoría, durante chats, o información de productos
- ✓ A través del proceso de mantenimiento y modernización de nuestros productos.

1.2.1 Comparación de las herramientas analizadas con la aplicación a desarrollar

Las herramientas de encuestas explicadas, tienen características muy esenciales y poderosas en el momento de la toma de decisiones de las empresas. No se puede dejar pasar por alto las facilidades que ofrecen para un mejor funcionamiento en el procesamiento y captura de informaciones, tales como:

- ✓ Recopilar información de un modo eficiente y confiable.
- ✓ Centralizar el control y la administración de formularios.

- ✓ Optimizar los procesos empresariales basados en formularios.

A continuación se muestra una tabla de comparación con los principales criterios tomados de las características que presentan cada una de las herramientas analizadas:

Criterios	CSPro	InfoPath	SSPS	SAS
Tipo de aplicación.	Desktop	Desktop	Desktop	Desktop
Múltiples registros.	Si	Si	Análisis estadísticos.	Análisis estadísticos.
Integridad de datos.	Si	Si	Si	Si
Tipos de variables.	Todo tipo de variables	Todo tipo de variables	Valores numéricos, notaciones científicas, cadenas y símbolos matemáticos.	Valores numéricos, notaciones científicas, cadenas y símbolos matemáticos.
Plataforma	Windows	Windows	Windows, Macintosh, Linux y Unix.	Windows
Captura de datos.	Validaciones, comparación y corrección inmediata de errores, corregir y producir	Validaciones de datos, comparaciones, filtrados, eventos, entre otros.	Validaciones de datos, comparaciones, filtrados, eventos, conexiones, búsquedas, entre	Validaciones de datos, comparaciones, filtrados, eventos, conexiones, búsquedas, entre

	tabulaciones de datos censales o de una encuesta.		otros.	otros.

Tabla 1: Tabla de comparaciones de las herramientas para el diseño de encuestas.

Estos software en común aportan seguridad, fiabilidad y continuidad en operativas tradicionalmente complejas como es el uso distribuido e integración de la misma con otros sistemas. A partir de la comparación realizada se pueden destacar aspectos comunes que forjan la base de los posibles servicios que brindará el sistema a desarrollar. Los sistemas mostrados en el estudio teórico presentan la desventaja de que son aplicaciones de escritorio, lo que constituye el factor impulsor para realizar la propuesta de trabajo como aplicación Web. Otras de las características que se debe recalcar además de la captura de datos que presentan estas aplicaciones y la integridad del proceso de captura de datos, es que sea multiplataforma y que los formularios contengan múltiples registros. La propuesta de este trabajo constituye el resultado de asimilar características esenciales, encontradas en los sistemas de su tipo más utilizados en el mundo y combinarlas con las ventajas de la Web.

1.3 Metodología de desarrollo

Para asegurar el éxito durante el desarrollo de software no es suficiente contar con notaciones de modelado y herramientas, hace falta un elemento importante: la metodología de desarrollo, la cual provee de una dirección a seguir para la correcta aplicación de los demás elementos.

Las metodologías de desarrollo clásicas proponen un ciclo de desarrollo muy rígido, en el que las fases están muy bien definidas y una vez que se pasa de una a otra resulta muy difícil volver a fases anteriores del proceso sin incurrir en retrasos, mayores costes, entre otros. Para dar respuesta las cambiantes necesidades del mundo actual, en el que continuamente se deben realizar cambios en las especificaciones, surgieron un conjunto de metodologías comúnmente conocidas como metodologías ágiles. Existen un número elevado de estas metodologías ágiles, pero en general todas siguen los conceptos especificados en el “Manifiesto ágil[2] en mayor o menor medida y comparten una serie de características, entre las que conviene destacar: (9)

- ✓ Desarrollo basado en iteraciones: Este tipo de metodologías defienden que se deben realizar iteraciones de entre uno a tres meses a lo sumo, al final de las cuales se deben entregar soluciones parciales que aporten valor de negocio al cliente.

[2] Manifiesto ágil: documento que describe los fundamentos de las metodologías ágiles redactado en una reunión desarrollada en Utah-EEUU en febrero del 2001, con la participación de 17 expertos del mundo de la industria de software

- ✓ Se valora más a las personas que a los procesos: En este enfoque se da más valor a las personas y al conocimiento que aportan que a la definición de procesos estáticos.
- ✓ Involucrar al cliente: En este tipo de metodologías la interacción con el cliente debe ser constante, si el cliente no se involucra en el proceso de desarrollo de una manera activa difícilmente se puede garantizar la calidad del producto final, y lo que es peor, que de verdad realice las tareas que se esperaba de él.
- ✓ Simplicidad: Es imprescindible realizar primero lo más sencillo que se pueda, y posteriormente ir añadiendo funcionalidades en forma de incrementos.

Proceso Unificado Abierto (OpenUp).

OpenUp es una metodología de desarrollo basada en la metodología de Proceso Unificado Relacional (RUP - *Rational Unified Process*). Es el subconjunto de esta última que contiene el conjunto mínimo de prácticas que ayuden a un equipo de desarrollo de software a realizar un producto de alta calidad y de una forma más eficiente. OpenUp utiliza un punto de vista pragmático y una filosofía ágil que se centraliza en la naturaleza colaborativa del proceso de desarrollo del software. Una de sus principales características es su alto grado de adaptabilidad a las necesidades de un proyecto en particular. Intenta incluir dentro de su proceso de desarrollo únicamente el contenido imprescindible para garantizar un proceso de desarrollo de calidad y eficiente. (9)

OpenUp tiene las características de un Proceso Unificado al cual se aplican enfoques iterativos e incrementales dentro de un ciclo de vida estructurado. El proceso de desarrollo en OpenUp utiliza casos de uso, escenarios, gestión del riesgo y un enfoque centrado en la arquitectura. (9)

Los cuatro principios básicos en los que se fundamenta OpenUp son los siguientes:

- ✓ Promover la colaboración para alinear los intereses comunes y difundir el conocimiento.
- ✓ Balanceo de las prioridades competitivas para maximizar el valor de los participantes del proyecto.
- ✓ Desarrollo de una arquitectura al principio con el fin de minimizar riesgos y planificar el desarrollo.
- ✓ Evolucionar para obtener una retroalimentación continua en aras de una mejora continua.

El objetivo de OpenUp es ayudar al equipo de desarrollo a través de todo el ciclo de vida, de modo que este sea capaz de añadir valor de negocio para los clientes de una forma predecible: con la entrega de

un software operativo y funcional al final de cada iteración. Partiendo de lo antes expuesto el grupo de arquitectura del proyecto SiGOB propuso como metodología de desarrollo Open UP para el progreso de la aplicación.

1.4 Lenguaje de programación

Un lenguaje de programación es un conjunto de símbolos, reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos o expresiones, es utilizado para controlar el comportamiento físico y lógico de una máquina. Además, permite especificar de manera precisa sobre qué datos debe operar una computadora, cómo estos datos deben ser almacenados o transmitidos y qué acciones debe tomar bajo otras circunstancias.

Actualmente existen diferentes lenguajes de programación para desarrollar en la Web, estos han ido surgiendo debido a las tendencias y necesidades de las plataformas. Desde los inicios de Internet, fueron apareciendo diferentes demandas por los usuarios y se dieron soluciones mediante lenguajes estáticos. Con el transcurso del tiempo, las tecnologías fueron desarrollándose y surgieron nuevos problemas a los que darle solución. Esto dio lugar a desarrollar lenguajes de programación para la Web dinámica, que permitieran interactuar con los usuarios y utilizaran sistemas de Bases de Datos. Algunos ejemplos de lenguajes de programación Web son: PHP, JavaScript, HTML y otros más. (10)

1.4.1 Lenguaje de programación del lado del servidor

Existen diferentes lenguajes concebidos para la Web, cada uno de ellos explota más a fondo ciertas características que lo hacen más o menos útiles para desarrollar distintas aplicaciones.

La versatilidad de un lenguaje está íntimamente relacionada con su complejidad. Un lenguaje de complejo aprendizaje permite en general realizar un espectro de tareas más amplio y más profundo. Es por ello que a la hora de elegir el lenguaje que se quiere utilizar es necesario conocer claramente lo que hay que hacer y si el lenguaje en cuestión lo permite o no. El lenguaje del lado del servidor seleccionado para el desarrollo de la aplicación es PHP.

PHP

PHP es un lenguaje de programación interpretado, diseñado para la creación de páginas Web dinámicas de manera ágil y eficiente, es un acrónimo recursivo que significa "Hypertext Pre-processor".

Presenta como ventajas que es un lenguaje multiplataforma completamente orientado al desarrollo de aplicaciones Web dinámicas con acceso a información almacenada en una Base de Datos. Tiene capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la

actualidad, destacando su conectividad con MySQL y PostgreSQL, el código fuente escrito en PHP es transparente al navegador y al cliente, ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador y es una alternativa de fácil acceso para todos, pues es libre. (11)

PHP ha seguido evolucionando. Desde julio del 2004 fue lanzado PHP 5, que utiliza el motor Zend Engine 2.0. Se selecciona PHP 5 porque permite mejorar los mecanismos de Programación Orientada a Objetos (POO) para solucionar las carencias de las anteriores versiones, excelente soporte de acceso a datos, además este nuevo motor incluye mejoras de rendimiento y el manejo de excepciones. (11)

1.5 Lenguaje de Modelado: UML

Se definió para el diseño del componente el Lenguaje Unificado de Modelado (UML - *Unified Modeling Language*). UML es la sucesión de una serie de métodos de análisis y diseño orientados a objetos que aparecen a finales de los años ochenta y principios de los noventa. Es llamado como lenguaje de modelado y no se considera un método. Incrementa la capacidad de lo que se puede hacer con otros métodos de análisis y diseño orientados a objetos. Los autores de UML apuntaron también al modelado de sistemas distribuidos y concurrentes para asegurar que el lenguaje maneje adecuadamente estos dominios. (15)

El lenguaje de modelado es la notación (principalmente gráfica) que usan los métodos para expresar un diseño. El proceso indica los pasos que se deben seguir para llegar a un diseño.

Una de las metas principales de UML es avanzar en el estado de la integración institucional proporcionando herramientas de interoperabilidad para el modelado visual de objetos. Sin embargo para lograr un intercambio exitoso de modelos de información entre herramientas, se requirió definir a UML una semántica y una notación. Una herramienta de UML debe mantener la consistencia entre los diagramas en un mismo modelo. (15)

Posee mayor rigor en la especificación, permite realizar una verificación y validación del modelo realizado. Se pueden automatizar determinados procesos y permite generar código a partir de los modelos y a la inversa (a partir del código fuente generar los modelos). Esto permite que el modelo y el código estén actualizados, con lo que siempre se puede mantener. (15)

1.6 Tecnologías y Herramientas

Las tecnologías de información y las herramientas que soportan el desarrollo de una aplicación Web, juegan un papel clave en su evolución. Representan el apoyo a la administración del proyecto, adaptándose a las características y objetivos propios de la organización. Seguido de la necesidad de realizar un software con la calidad requerida se realizó un análisis de las tecnologías y herramientas necesarias para la implementación del componente.

Herramientas CASE

Las herramientas de Ingeniería de Software Asistida por Computadora (CASE) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el costo de las mismas en términos de tiempo y de dinero. Estas herramientas apoyan en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el proceso de realizar un diseño del proyecto, cálculo de costes, documentación o detección de errores, entre otras. (20)

Entre sus principales objetivos se encuentran:

- ✓ Mejorar la productividad en el desarrollo y mantenimiento del software.
- ✓ Aumentar la calidad del software.
- ✓ Ayuda a la reutilización del software, portabilidad y estandarización de la documentación.
- ✓ Gestión global en todas las fases de desarrollo de software con una misma herramienta.
- ✓ Facilitar el uso de las distintas metodologías propias de la ingeniería del software.

Bajo la decisión del Grupo de Arquitectura del proyecto SiGOB se utilizará para el modelado del sistema a implementar la herramienta CASE: Visual Paradigm 6.1.

Visual Paradigm es una herramienta de modelado que soporta UML y permite modelar todos los diagramas necesarios para representar gráficamente el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. (21)

Algunas características de Visual Paradigm:

- ✓ Es profesional
- ✓ Contiene facilidades para redactar especificaciones de casos de uso del sistema
- ✓ Sincronización entre diagramas de entidad-relación y diagramas de clases
- ✓ Generación de código / ingeniería inversa

- ✓ Generación de documentos.
- ✓ Interoperabilidad con otras aplicaciones
- ✓ Integración con distintos Entornos de Desarrollo Integrados (IDEs), entre ellos está NetBeans.

Netbeans 6.9

NetBeans es una aplicación de código abierto (*open source*) diseñada para el desarrollo de aplicaciones fácilmente portable entre las distintas plataformas, haciendo uso de la tecnología Java. Dispone de soporte para crear interfaces gráficas de forma visual, desarrollo de aplicaciones Web, control de versiones, colaboración entre varias personas, creación de aplicaciones compatibles con teléfonos móviles y funcionalidades ampliables mediante la instalación de packs.

Cuenta con el apoyo de una vibrante comunidad de desarrolladores y ofrece una variada selección de terceros plug-ins, además es un entorno de desarrollo integrado para Windows, Mac, Linux y Solaris.

Esta nueva versión cuenta con nuevas facilidades lo que ha conllevado a su preferencia para el desarrollo de la aplicación, entre ellas se destacan que las aplicaciones basadas en esta plataforma generan instaladores para los sistemas operativos más usados. Cuenta con un soporte mejorado para consumir servicios Web y conectarse a bases de datos, soporte para servicios Web REST en aplicaciones RPC, mejoras al debugger. Presenta además un nuevo soporte para Applets de Java, búsqueda de usos para CSS y lenguajes tipo HTML, soporte para PHP Zend Framework. (22)

Marco de trabajo (framework) Symfony

Symfony fue seleccionado debido a que está desarrollado completamente en PHP 5, es un framework diseñado para optimizar el desarrollo de las aplicaciones Web. Una de sus principales características es que separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación Web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación Web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación.

Symfony ha sido probado en numerosos proyectos reales y se utiliza en sitios Web de comercio electrónico de primer nivel. Es compatible con la mayoría de los gestores de bases de datos, como MySQL, PostgreSQL, Oracle y Microsoft SQL Server e incluso permite el cambio en cualquier fase del proyecto.

Presenta otras características favorables, como son su fácil instalación y configuración en la mayoría de las plataformas. Está preparado para aplicaciones empresariales, adaptables a las políticas y

arquitecturas propias de cada empresa, conteniendo una potente línea de comandos que facilitan la generación de código para el ahorro del tiempo de trabajo. (16)

XML

Es el estándar de Lenguaje de Marcado Extensible (XML - *Extensible Markup Language*), conformado por un conjunto de reglas para definir etiquetas semánticas orientadas a organizar un documento en diferentes partes. Permite al usuario definir sus propios lenguajes de anotación adaptados a sus necesidades y contiene tres características muy importantes que son: extensibilidad, estructura y validación. (13)

Ventajas de XML:

- ✓ Las aplicaciones se pueden generar rápidamente y su mantenimiento es más sencillo.
- ✓ Separa los datos de la presentación y del proceso, lo que permite mostrar y procesar los datos al gusto deseado con sólo aplicar distintas hojas de estilo y aplicaciones.
- ✓ La información es más accesible y reutilizable, por la flexibilidad de las etiquetas de XML sin tener que amoldarse a reglas específicas de un fabricante. (13)

YAML

El formato YAML (YAML Ain't Markup Language) es utilizado para serializar datos, resultando fácil de procesar por las máquinas, de lectura para los usuarios y de interactuar con los lenguajes de script. Es una interesante herramienta que permite crear de manera simple, la estructura de sitios Web en XHTML y CSS, respetando todos los estándares y permitiendo una visualización correcta en cualquier navegador. YAML fue creado bajo la creencia de que todos los datos pueden ser representados adecuadamente como combinaciones de listas, mapeos (*hashes*) y datos escalares (valores simples). La sintaxis es relativamente sencilla y fue diseñada teniendo en cuenta que fuera muy legible, pero que a la vez fuese fácilmente interpretable a los tipos de datos más comunes en la mayoría de los lenguajes de alto nivel. (14)

Ext.JS

A principios de 2006, Jack Slocum desarrolló un conjunto de servicios públicos de extensión a la biblioteca de YUI, con el pasar del tiempo fueron tomando popularidad y se organizaron rápidamente en una biblioteca independiente llamada YUI-Ext. En el 2007, Jack formó una compañía para seguir avanzando en el desarrollo de la extensión, que en algún momento a partir de entonces comenzó a ser conocido como Ext.JS.

Ext.JS es el motor que permite crear Aplicaciones Ricas en Internet (RIA) mediante Javascript, usando tecnologías como AJAX, DHTML y DOM. Es ligera y de alto rendimiento, compatible con la mayoría de navegadores y es mejor conocido por la colección de primera clase de widgets[3] de interfaz de usuario.

Ventajas de Ext.JS:

- ✓ Existe un balance entre cliente - servidor. La carga de procesamiento se distribuye, permitiendo que el servidor, al tener menor carga, pueda manejar más clientes al mismo tiempo.
- ✓ Comunicación asíncrona. En este tipo de aplicación el motor de render puede comunicarse con el servidor sin necesidad de estar sujeta a un clic o una acción del usuario, dándole la libertad de cargar información sin que el cliente se dé cuenta.
- ✓ Eficiencia de la red. El tráfico de red puede disminuir al permitir que la aplicación elija que información desea transmitir al servidor y viceversa, sin embargo la aplicación que haga uso de la pre-carga de datos puede que revierta este beneficio por el incremento del tráfico.

Ext JS ha sufrido algunos cambios de licencias, pero existe la opción de fuente abierta que está disponible, permitiendo su uso de forma gratuita en los términos de la GNU Licencia Pública General (GPL) 3.0.

Desde la versión 3.0, el equipo de Ext.JS lanzó el producto Core Ext para ofrecer distintos navegadores API y realizar la mayoría de las tareas como: la manipulación del DOM, CSS gestión, el control de eventos, Ajax, y animaciones. Ext. Core es que está bajo la licencia MIT liberal, que es una licencia de código abierto. (17)

RPC

El mecanismo de comunicación para las aplicaciones cliente servidor, como es el caso del componente a desarrollar, se proporcionará a través del paquete Llamada a Procedimientos Remotos (RPC - *Remote Procedure Call*). RPC se desarrolló por Sun Microsystems, creado por Andrew D. Birrell y Bruce Jay Nelson en 1984, es una colección de herramientas y librerías de funciones. Es un protocolo que permite a un programa de ordenador ejecutar código en otra máquina remota sin tener que preocuparse por las comunicaciones entre ambos.

[3] Widgets: Son pequeñas aplicaciones o programas, usualmente presentado en archivos o ficheros pequeños que son ejecutados por un motor de *widgets* o *Widget Engine*. Entre sus objetivos están los de dar fácil acceso a funciones frecuentemente usadas y proveer de información visual.

Un servidor RPC consiste en una colección de procedimientos que un cliente puede solicitar enviando una petición al servidor junto con los parámetros. El servidor invocará el procedimiento indicado en nombre del cliente, entregando el valor de retorno, si hay alguno. RPC confía en sockets estándar UDP y TCP para transportar los datos en formato Representación de Datos Externos (XDR) hacia el host remoto. (18)

Funcionamiento de RPC:

RPC usa el mecanismo de solicitud respuesta presentado previamente. Oculta los detalles relacionados con la creación envío y recepción de los mensajes. El cliente y el servidor se comunican mediante dos stubs, un stub es una interfaz de comunicación que implementa el protocolo RPC, especifica cómo se construyen y cómo se intercambian los mensajes. Los stubs son generados por un compilador de protocolo enlazándolos con los clientes y los servidores

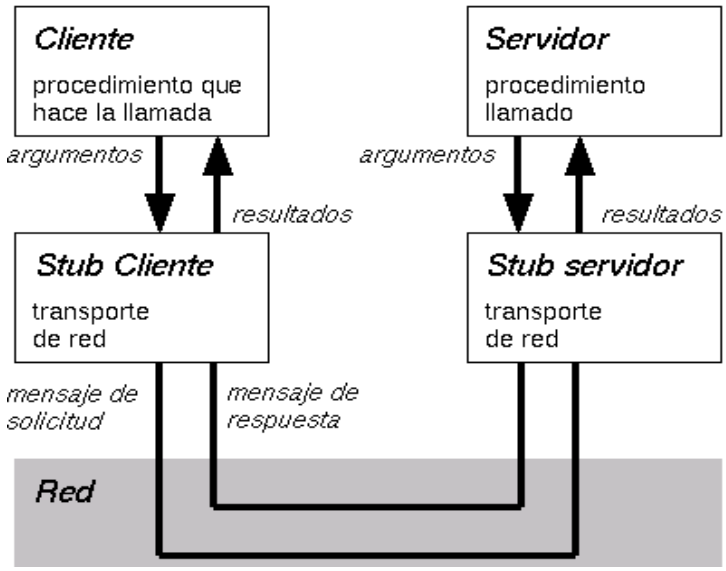

Figura 1: Llamada de Procedimiento Remoto (RPC)

Ventajas de RPC:

- ✓ Compartición de recursos
- ✓ Concurrencia
- ✓ Alta escalabilidad
- ✓ Tolerancia a fallos
- ✓ Ofrece un mecanismo de nivel más alto que los sockets

JSON

El formato ligero de Notación de Objetos de JavaScript (JSON - *JavaScript Object Notation*) es utilizado para el intercambio de datos, lo que proporciona que sea entendible a las máquinas para interpretarlo y generarlo. Está basado en un subconjunto del Lenguaje de Programación JavaScript. JSON es un formato de texto que es completamente independiente del lenguaje pero utiliza convenciones que son ampliamente conocidos por los programadores de la familia de lenguajes C, incluyendo C, C++, C#, Java, JavaScript, Perl, Python, y muchos otros.

Entre las ventajas se define que su procesamiento por parte de los ordenadores es rápido; se necesitan librerías muy pequeñas para trabajar con él (siendo posible incluso procesarlo sin librería); dada su naturaleza es ideal para entornos Ajax. (12)

Propel

Propel representa el mapeo de objetos relacionales (ORM - *Object-Relational Mapping*) para PHP que facilita la labor de desarrollo de aplicaciones Web, gracias a la capa que transforma el tratamiento de la base de datos mediante objetos, con la que se puede recuperar, insertar y modificar datos. Permite realizar consultas complejas y manipular bases de datos sin escribir una sola cláusula de Lenguaje de Consultas Estructurado (SQL - *Structured Query Language*). Además, hace más fácil la escritura de aplicaciones, su despliegue o migrar si alguna vez la situación lo amerita. Con Propel sólo es necesario definir la base de datos en formato XML u obtener la definición desde una base de datos ya existente. Proporciona un inteligente y comprensivo servicio de manejo de datos con un mínimo costo de realización para una aplicación en PHP. (19)

PGAdmin

Se selecciona el PGAdmin por ser una herramienta de código abierto que tiene como propósito general diseñar, mantener y administrar las bases de datos en Postgres. Está diseñado para responder a las necesidades de la mayoría de los usuarios, desde escribir simples consultas SQL hasta desarrollar bases de datos complejas. Está disponible en más de una docena de lenguajes y para varios sistemas operativos, incluyendo Microsoft Windows, Linux, FreeBSD, Mac OSX y Solaris. (23)

Características incluidas:

- ✓ Entradas SQL aleatorias.
- ✓ Pantallas de información y 'Ayudas' para bases de datos, tablas, índices, secuencias, vistas, programas de arranque, funciones y lenguajes.

- ✓ Preguntas y respuestas para configurar Usuarios, Grupos y Privilegios.
- ✓ Control de revisión con mejora de la generación de script.
- ✓ Configuración de las tablas de Microsoft MSysConf.
- ✓ Informes predefinidos en bases de datos, tablas, índices, secuencias, lenguajes y vistas.

PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado. PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilo para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. (24)

Entre sus principales características se encuentran:

- ✓ Soporta distintos tipos de datos.
- ✓ Incorpora una estructura de arreglos de datos.
- ✓ Incorpora funciones de diversa índole.
- ✓ Permite la declaración de funciones propias, así como la definición de disparadores.
- ✓ Soporta el uso de índices, reglas y vistas.
- ✓ Incluye herencia entre tablas (aunque no entre objetos, ya que no existen), por lo que a este gestor de bases de datos se le incluye entre los gestores objeto-relacionales.
- ✓ Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.

Conclusiones parciales

En este capítulo se realizó un análisis de los elementos fundamentales de las diferentes herramientas propuestas, arrojando como resultado las características claves que servirán de base para el diseño de la propuesta de trabajo. Debido a que el proyecto cuenta con un pequeño equipo de desarrollo y poco tiempo de desarrollo disponible, se selecciona la metodología OpenUP como guía para la documentación de SiGOB junto a Visual Paradigm como herramienta Case. Como lenguaje de programación del lado del servidor se escogió PHP en su versión 5 y el framework para el desarrollo de aplicaciones Web Symfony, que implementa el estilo arquitectónico Modelo - Vista - Controlador.

Dicho framework tiene integrado el ORM Propel, el cual garantiza una auditoría de seguridad en el código. El formato seleccionado para la definición de la encuesta y para el intercambio con los demás módulos de SiGOB es el estándar JSON, que acompañado de la tecnología RPC provee la flexibilidad y escalabilidad deseadas para el componente. El gestor de base de datos elegido es PostgreSQL por ser la herramienta de código abierto más potente del mercado. En aras de lograr la organización del código se seleccionó el entorno integrado de desarrollo Netbeans 6.9 con soporte para los lenguajes de programación a utilizar, cumpliendo con el estándar de codificación definido por DATEC.

CAPÍTULO 2: DISEÑO DE LA SOLUCIÓN

Este capítulo describe la propuesta de solución para el problema científico de la investigación. Mediante conceptos y pautas, se explicará el modelo de dominio como representación de la estructura de lo que se desea obtener. A partir de comprender el negocio, se obtienen los requisitos funcionales y no funcionales. La traducción de los mismos a una especificación que describe cómo implementar el componente, permite realizar el diseño de la aplicación generando los artefactos necesarios para cada fase. Se realizan los diagramas de clases y los de interacción para los casos de uso arquitectónicamente significativos, terminando en el diseño de la aplicación con la estructura física del componente para la captura de datos que se propone mediante el modelo de despliegue.

2.1 Modelo de Dominio

En la fase de inicio se determinó que los procesos del negocio no están lo suficientemente definidos, por lo que se decidió separar y representar los conceptos más importantes relacionados con la realidad física del sistema mediante un Modelo de Dominio (Figura 2). Este modelo permite mostrar de manera visual los principales conceptos que se manejan en el dominio del problema y de esta forma utilizar un vocabulario común que ayude a usuarios, clientes, desarrolladores e interesados a entender el contexto en que se ubica el sistema.

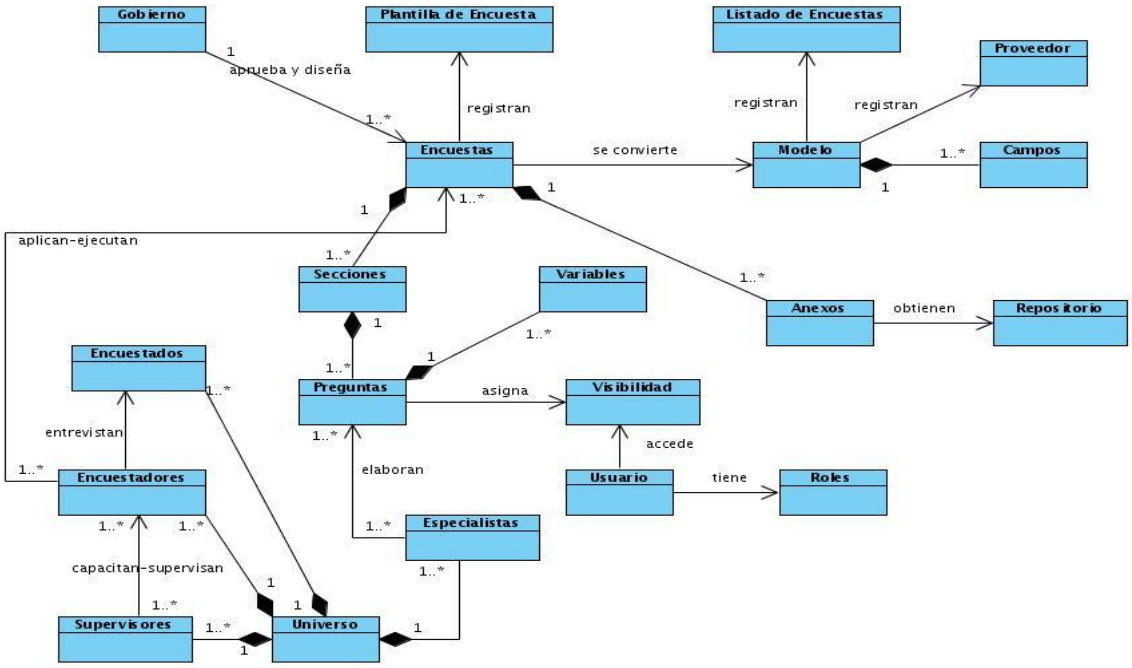

Figura 2: Modelo de Dominio

Conceptos del Modelo de Dominio:

Clase Gobierno: Asigna la responsabilidad de diseñar las encuestas y aprueba las propuestas de encuestas elaboradas.

Clase Universo: Contiene, como su nombre lo indica, el universo de personas involucradas en todo el proceso de encuesta a lo largo del ciclo de vida de la misma: especialistas, encuestadores, encuestados.

Clase Supervisores: Personal responsable de capacitar, supervisar y controlar a los encuestadores.

Clase Encuestadores: Personal responsable de aplicar la encuesta.

Clase Especialistas: Profesional encargado de elaborar las preguntas que contiene una encuesta, definir la muestra a la que se aplicará la encuesta, así como revisar los resultados obtenidos.

Clase Encuestados: Personal sobre el que se realiza la encuesta.

Clase Encuesta: Existen dos tipos de encuestas: las que se realizan a la población y las económicas. Estas contienen un conjunto de preguntas elaboradas por especialistas y una vez aplicadas a los encuestados arrojan resultados que sirven para estudios posteriores. Una encuesta está conformada por las bases metodológicas, por el encabezado, que constituye un tipo especial de sección, por secciones las cuales contienen preguntas y estas a su vez contienen las variables. La encuesta tiene un ciclo de vida, inicialmente en el estado no activado se aplica el proceso de diseño de la encuesta, luego de aprobado el diseño esta pasa al estado activado donde se pone en marcha al proceso de pre-visualización de la encuesta, se publica y se le da un seguimiento mientras se encuentra activada la misma.

Clase Secciones: Se refiere a las partes en las que está estructurada una encuesta. En las secciones se recogen las preguntas por temas.

Clase Preguntas: Representa las preguntas contenidas en las secciones de una encuesta y son diseñadas por especialistas.

Clase Variable: Las variables son los campos que contienen las preguntas.

Clase Visibilidad: Se refiere al estado que pueda tener una encuesta o parte de ella y que usuario o rol, según los privilegios asignados, puede acceder a dicha parte.

Clase Usuario: Es el personal que accederá a la encuesta cumpliendo con las políticas que plantea la AAA (Autenticación, Autorización y Atributos).

Clase Roles: Define el nivel de acceso que pueda tener un usuario siguiendo las políticas de la AAA.

Clase Anexos: Constituye un tipo de información de soporte enciclopédico que aporta información relacionada con la encuesta.

Clase Repositorio: Lugar donde estarán almacenados los anexos que pueden adjuntarse a las encuestas.

Clase Plantilla de Encuesta: Una vez diseñada y aprobada la encuesta se procede al proceso de digitación de la misma, quedando como resultado la plantilla en formato digital, la cual se llenará posteriormente.

Clase Modelo: Una vez digitada y conformada la plantilla de encuesta, el modelo sería el resultado de llenar dicha plantilla con las respuestas de los encuestados a las preguntas realizadas por los encuestadores, formato que define la información estadística.

Clase Campos: Atributos que tendrá el modelo, pueden ser atributo llave o atributos comunes: el nombre, tipo, restricciones de valores, descripción, estado.

Clase Proveedor: Unidad física, host donde se almacenarán los datos de las encuestas. Físicamente un Proveedor es una fuente de datos (Base de Datos) de la cual se obtiene la información que posteriormente se ha de mostrar en el reporte.

Clase Listado de Encuestas: Una vez que se ha diseñado la encuesta se ha obtenido la plantilla, por lo que se procede al proceso de llenado de la encuesta. El resultado que se obtiene es el listado de dichas encuestas con las respuestas que recogieron los encuestadores.

2.2 Requerimientos

Partiendo de la descripción de las clases más importantes dentro del contexto del sistema representado en el Modelo de Dominio, se determinaron los requisitos del sistema de información. Un requisito 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. (25)

Requerimientos funcionales

Se identificaron los siguientes requisitos funcionales :

RF 1 Administrar Plantilla de Encuesta

RF 1.1 Consultar registro de plantillas

RF 1.2 Guardar plantilla de encuesta

RF 1.3 Eliminar plantilla de encuesta

RF 1.4 Publicar plantilla de encuesta

RF 1.5 Deshabilitar – Habilitar plantilla de encuesta

RF 1.6 Gestionar categoría

RF 1.6.1 Insertar categoría

RF 1.6.2 Eliminar categoría

RF 1.6.3 Renombrar categoría

RF 1.6.4 Actualizar propiedades de categoría

RF 2 Digitar encuesta

RF 2.1 Insertar datos de encuesta

RF 2.2 Actualizar datos de encuesta

RF 3 Administrar Encuesta

RF 3.1 Editar encuesta

RF 3.2 Exportar encuesta

RF 3.3 Guardar encuesta

RF 3.4 Eliminar encuesta

Requerimientos no funcionales

Los requerimientos 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. Se identificaron a raíz de estos elementos los siguientes requisitos no funcionales: (25)

Portabilidad

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

Fiabilidad

RNF 1 Disponibilidad

El sistema debe poder permanecer constantemente funcionando excepto cuando sea necesario reiniciarlo o detenerlo por tareas de mantenimiento o cambio de configuración.

RNF 1.1 Disponibilidad de Servicios

Debido a que el sistema mantendrá en todo momento una cola de eventos, es necesario que este permanezca en línea constantemente para atender las solicitudes desde un usuario directamente. El sistema permanecerá fuera de servicio sólo en caso que se estén realizando tareas de mantenimiento o de configuración.

Eficiencia

RNF 2 Capacidad

El sistema debe permitir la concurrencia de al menos 100 usuarios conectados a la base de datos.

PostgreSQL sin modificaciones acepta hasta 100 conexiones simultáneas escribiendo datos en la Base de datos. Configurando PostgreSQL puede soportar 500 usuarios como máximo al mismo tiempo sin afectar el rendimiento o funcionalidad del mismo.

Restricciones de diseño

RNF 3 Restricciones del diseño e implementación

El sistema deberá ser implementado en el lenguaje de programación PHP versión 5.2 o superior, como framework de desarrollo se usará Symfony en su versión 1.1.7 y la librería de Java Script Ext JS versión 3.0. Como herramienta de desarrollo se usará NetBeans 6.9 y como sistema gestor de base de datos se utilizará PostgreSQL 8.4.

Requerimiento de software

RNF 4 Requerimiento de software.

- Todo el software debe correr sobre el Sistema Operativo Linux.
- El software base debe responder a las políticas de Software Libre y de Fuente Abierta.
- Los clientes tendrán acceso al sistema a través del navegador Mozilla Firefox.

Software

RNF 5 Software

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

- Sistema Operativo: GNU/Linux preferentemente Ubuntu GNU/Linux 8.04 o superior, Debian 4.0 GNU/Linux o superior.
- Usuario con privilegios de administración del sistema operativo.

El servidor donde se instalará la base de datos debe cumplir con los siguientes requisitos:

- Sistema Operativo: GNU/Linux preferentemente Ubuntu GNU/Linux 8.04 o superior, Debian 4.0 GNU/Linux o superior.
- PostgreSQL versión 8.3 o superior.
- PGAdminIII o algún administrador para PostgreSQL.
- Usuario con privilegios para instalar la base de datos.
- PostgreSQL debe estar correctamente configurado para aceptar conexiones vía TCP/IP

Hardware

RNF 6 Hardware

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

- Procesador Intel Pentium 4 a 3.0 GHz
- 2 GB de RAM.
- 160 GB de espacio en disco duro.

El servidor donde se instalará la base de datos debe cumplir con los siguientes requisitos:

- Procesador Intel Pentium 4 a 3.0 GHz
- 2 GB de RAM.
- 160 GB de espacio en disco duro.

La máquina utilizada para acceder a la aplicación debe cumplir con los siguientes requisitos:

- Procesador Intel Pentium 4 1.7 GHz o AMD similar.
- 512 MB de RAM o superior.

- 80 GB de espacio en disco duro o superior. (30)

2.3 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 usuario, el cual incluye los actores, los casos de uso y sus relaciones. A partir de los requisitos funcionales identificados anteriormente se modeló el siguiente Diagrama de Casos de Uso:

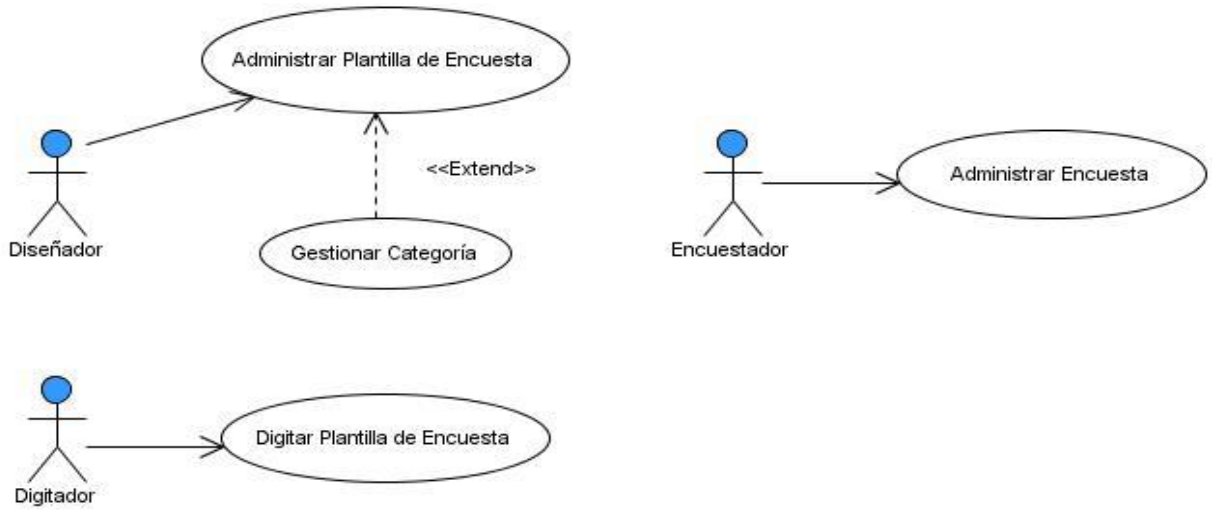

Figura 3: Diagrama de Casos de Uso del Sistema

Actor	Descripción
Diseñador	Ente encargado de diseñar la plantilla de encuesta.
Encuestador	Ente dedicado a aplicar la encuesta a la muestra seleccionada. Es el encargado de consultar el registro de encuesta y gestionar todo lo relacionado a la encuesta, dígase eliminar, modificar o digitar.
Digitador	Ente con la responsabilidad de registrar en la plantilla de encuesta la información capturada en la realización de la encuesta a la muestra.

Tabla 2: Descripción de los actores del sistema

Descripción de un caso de uso significativo

<table>
<tr><td>Caso de Uso:</td><td colspan="2">Gestionar Categoría</td></tr>
<tr><td>Actores:</td><td colspan="2">Diseñador</td></tr>
<tr><td>Resumen:</td><td colspan="2">El CU se inicia cuando el Diseñador decide realizar una de la siguientes acciones:
Insertar categoría, el diseñador decide insertar una nueva categoría, el sistema inserta la nueva categoría y termina el CU.
Eliminar categoría, el diseñador decide eliminar una nueva categoría, el sistema elimina la categoría y termina el CU.
Renombrar categoría, el diseñador decide renombrar una categoría, especifica el nuevo nombre y el sistema guarda los cambios, terminando así el CU.
Actualizar propiedades de categoría, el diseñador decide revisar las propiedades de las categorías existentes, el sistema muestras las propiedades que pueden ser editables, terminando así el CU.</td></tr>
<tr><td>Precondiciones:</td><td colspan="2">Debe haberse iniciado el CU Diseñar Plantilla Encuesta y el CU Gestionar Plantilla Encuesta.</td></tr>
<tr><td>Referencias:</td><td colspan="2">RF 1.7, RF1.7.1, RF1.7.2, RF1.7.3, RF1.7.4.</td></tr>
<tr><td>Prioridad:</td><td colspan="2">Crítico</td></tr>
<tr><td colspan="3">Flujo Normal de Eventos</td></tr>
<tr><td>Acción del Actor</td><td colspan="2">Respuesta del Sistema</td></tr>
<tr><td>1. El diseñador selecciona a través de un menú contextual la opción que desee realizar de las categorías:
• Insertar categoría
• Eliminar categoría
• Renombrar categoría
• Propiedades de categoría</td><td colspan="2">2. El sistema en dependencia de la opción selecciona muestra la interfaz correspondiente:
• Insertar categoría, ir a la sección “Insertar categoría”.
• Eliminar categoría, ir a la sección “Eliminar categoría”.
• Renombrar categoría, ir a la sección “Renombrar categoría”.
• Propiedades de categoría, ir a la sección “Actualizar propiedades de</td></tr>
</table>

	categoría".
Sección "Insertar categoría"	
Acción del Actor	**Respuesta del Sistema**
1. A través de un menú contextual el diseñador selecciona insertar una categoría nueva al sistema.	2. El sistema muestra una ventana para especificar Nombre y Descripción de la categoría.
3. El diseñador especifica todos los datos y selecciona la opción Aceptar.	4. El sistema inserta la nueva categoría y termina el CU.
Sección "Eliminar categoría"	
Acción del Actor	**Respuesta del Sistema**
1. A través de un menú contextual el diseñador selecciona eliminar una categoría nueva al sistema.	2. El sistema muestra un mensaje de alerta especificando si se desea que se elimine una categoría.
3. El diseñador selecciona la opción Aceptar del mensaje.	4. El sistema elimina la categoría, vuelve al entorno de trabajo y termina el CU.
Sección "Renombrar categoría"	
Acción del Actor	**Respuesta del Sistema**
1. El diseñador decide cambiarle el nombre a una categoría determinada.	2. El sistema muestra una interfaz nueva para indicar el nombre nuevo.
3. El diseñador especifica el nombre nuevo y selecciona la opción Aceptar.	4. El sistema cambia el nombre de la categoría y termina el CU.
Sección "Actualizar propiedades de categoría"	
Acción del Actor	**Respuesta del Sistema**
1. El diseñador decide revisar las propiedades de una categoría.	2. El sistema muestra una interfaz nueva con las propiedades de la categoría.
3. El diseñador realiza o no los cambios que desee sobre las propiedades y selecciona la opción Aceptar.	4. El sistema modifica en caso necesario, retorna al entorno de trabajo y termina el CU.
Poscondiciones	En correspondencia con la acción del Diseñador se: • Insertan categorías • Eliminan categorías

	• Renombran categorías • Se consultan o modifican las propiedades de una categoría

Tabla 3: Descripción del caso de uso Gestionar Categorías

Para obtener la descripción detallada de la realización de los casos de uso ver en el expediente digital del proyecto SiGOB el documento Modelo del Sistema. (27)

2.4 Objetivos del diseño

El diseño es el centro de atención al final de la fase de elaboración y el comienzo de las iteraciones de construcción. Esto contribuye a una arquitectura estable y sólida, creando un plano del modelo de implementación. El modelo de diseño está muy cercano al de implementación, lo que es necesario mantener el modelo de diseño a través de todo el ciclo de vida completo 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.
- 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. (28)

2.5 Diagrama de Clases del diseño

El diagrama de clases del diseño muestra los atributos y métodos de cada clase, representando de forma sencilla la colaboración y las responsabilidades de cada una de ellas entorno al sistema que conforman. A partir de la descripción detallada de los casos de uso del sistema, se modeló el siguiente diagrama de clases del diseño:

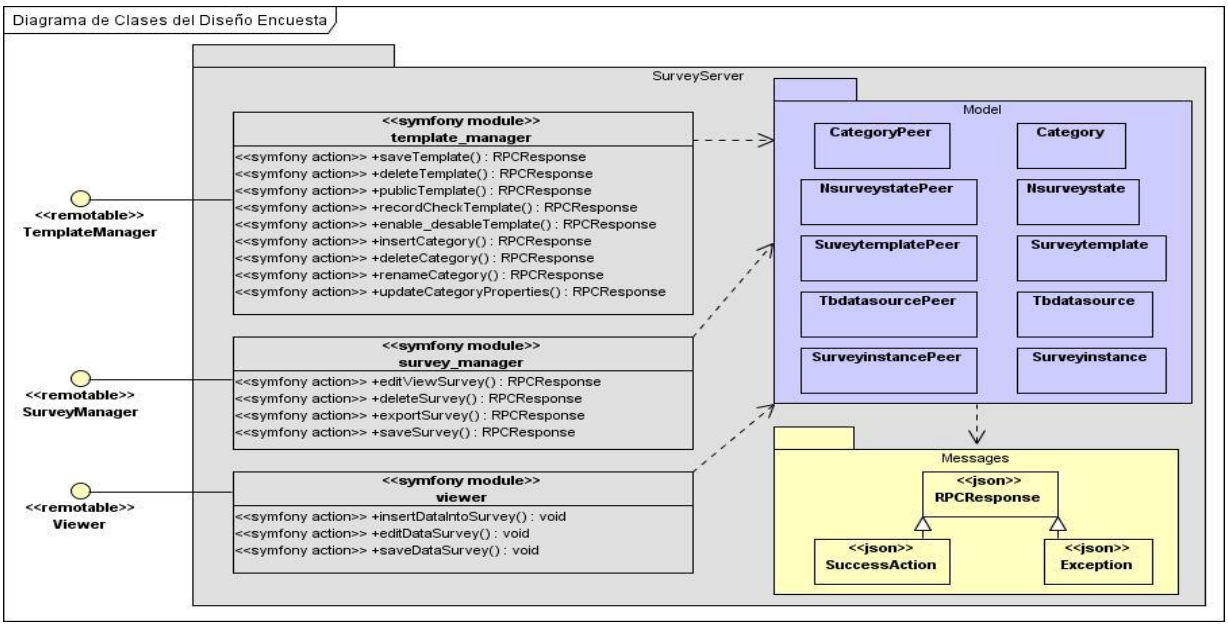

Figura 4: Diagrama de Clases del Diseño del Sistema

Descripción de las principales clases.

template_manager: Es el módulo de Symfony encargado de responder las peticiones Ajax del lado del servidor para la gestión de plantillas de encuestas.

Nombre: template_manager	
Tipo de Clase: Controladora.	
Para cada responsabilidad:	
Nombre	**Descripción**
+ saveTemplate()	Permite guardar las plantillas de encuesta, manteniéndolas a salvo en el servidor.
+ deleteTemplate()	Brinda la posibilidad de eliminar una o más plantillas que se encuentran en el servidor.
+ publicTemplate()	Una vez terminada de diseñar la plantilla el sistema le permite al usuario hacerla pública y de esta forma estar lista para su digitación.
+ recordCheckTemplate()	Permite al usuario consultar el registro de plantillas de encuestas.

+ enable_desableTemplate()	Debe ser capaz de activar o no la plantilla de encuesta para posibles acciones sobre ella, una plantilla pública ya está habilitada por defecto.
+ insertCategory()	Brinda la posibilidad de insertar la categoría deseada.
+ deleteCategory()	Posibilita la eliminación de la categoría indicada.
+ renameCategory()	Permite al usuario renombrar la categoría deseada.
+ updateCategoryPropeties()	Brinda la posibilidad de mostrar las propiedades de las categorías y a su vez permitir modificarlas.

Tabla 4: Módulo de Symfony template_manager

survey_manager: Es el módulo de Symfony encargado de responder las peticiones Ajax del lado del servidor para la gestión de encuestas.

Nombre: survey_manager	
Tipo de Clase: Controladora.	
Para cada responsabilidad:	
Nombre	**Descripción**
+ editViewSurvey ()	Permite al usuario acceder a una encuesta para revisar y transformar en caso necesario los datos de la misma con el fin de editarla. También debe permitir mostrar o visualizar los datos de determinada encuesta a través de HTML.
+ deleteSurvey ()	Permite al usuario eliminar la encuesta, o sea, suprimir o excluir la misma del servidor.
+ exportSurvey()	Exporta los datos que han sido introducidos en la encuesta.

+ saveSurvey()	Permite al usuario guardar la encuesta. Si se trata de una encuesta anteriormente cargada o visualizada esta es actualizada, si se trata de un intento de guardar la encuesta como si fuese nueva se emite un reporte de error dado su existencia.

Tabla 5: Módulo de Symfony survey_manager

viewer: Es el módulo de Symfony encargado de responder las peticiones Ajax del lado del servidor para digitar la plantilla de encuesta.

Nombre: viewer	
Tipo de Clase: Controladora.	
Para cada responsabilidad:	
Nombre	**Descripción**
+ insertDataIntoTemplate ()	Permite al usuario insertar los datos en la plantilla de encuesta seleccionada.
+ saveDataTemplate ()	Brinda al usuario la posibilidad de guardar los datos de la plantilla de encuesta y guardarlos en la dirección deseada.
+ editDataTemplate()	Permite al usuario modificar los datos en la plantilla de encuesta.

Tabla 6: Módulo de Symfony viewer

Patrones

Para hacer un diseño eficiente se tomaron en cuenta un conjunto de patrones, ya que los mismos constituyen una guía para resolver problemas comunes en programación. Para que una solución sea considerada un patrón debe poseer ciertas características, entre las cuales se encuentra la comprobación de su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reusable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

Entre los patrones arquitectónicos más aplicados para el diseño Web se encuentra el Modelo-Vista-Controlador (MVC) el cual es utilizado en la presente solución. El MVC permite separar las capas:

Modelo (representa la información con la que trabaja la aplicación, es decir, su lógica de negocio), Vista (transforma el modelo en una página Web que permite al usuario interactuar con ella) y Controlador (se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista).

La calidad de diseño de la interacción de los objetos y la asignación de responsabilidades presentan gran variación. Las decisiones poco acertadas dan origen a sistemas y componentes pocos robustos y difíciles de mantener, entender, reutilizar o extender. Una implementación hábil se funda en los principios cardinales que rigen un buen diseño orientado a objetos. En los patrones generales de software para asignación de responsabilidades (GRASP) se codifican algunos de los principios, que se aplican al diseñar los diagramas de interacción. Una de las principales ventajas a señalar en el uso del framework Symfony es que está basado en patrones de diseño. Los patrones de diseño empleados en la solución pertenecen al conjunto de patrones GRASP, su utilización ayudó a refinar el diseño y a asignar las responsabilidades de las distintas clases de diseño, haciéndolas más sencillas, reutilizables y encapsuladas. A continuación se explican los patrones utilizados:

Experto: El patrón experto en información es el principio básico de asignación de responsabilidades. Indica que la 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 (disminución del acoplamiento). 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.

Creador: En las clases Actions se encuentran las acciones definidas para el sistema de información a desarrollar divididas de forma organizada por módulos. En las acciones se crean los objetos de las clases que representan las entidades, evidenciando de este modo que las clases Actions representan el patrón "creador" de dichas entidades.

Alta Cohesión: Symfony permite asignar responsabilidades con una alta cohesión, por ejemplo la clase Actions tiene la responsabilidad de definir las acciones para las plantillas y colabora con otras para realizar diferentes operaciones, instanciar objetos y acceder a las propiedades, es decir, está formada por diferentes funcionalidades que se encuentran estrechamente relacionadas proporcionando que el software sea flexible frente a grandes cambios.

Controlador: Todas las peticiones Web son manejadas por un solo controlador frontal (sfAction), que es el único punto de entrada 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.

2.6 Vista Lógica

La Vista Lógica muestra los componentes principales de diseño y sus relaciones de forma independiente de los detalles técnicos y de cómo la funcionalidad será implementada en la plataforma de ejecución (ejemplo, .NET Framework). Describe la solución en términos de paquetes y clases de diseño, como también la realización de los casos de uso**,** subsistemas de los casos de uso más arquitectónicamente significativos.

A continuación se muestra la vista lógica de las clases que abarcan el comportamiento arquitectónicamente significativo:

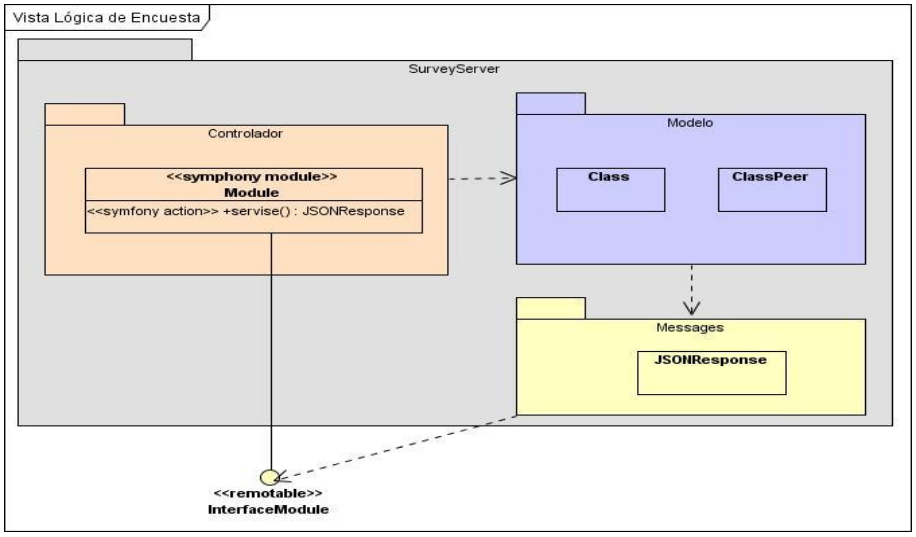

Figura 5: Vista Lógica

Para simplificar el desarrollo de la aplicación se utilizó la automatización de uno de los patrones más utilizados en el framework Symfony: el patrón Modelo-Vista-Controlador.

Como se trata de una aplicación Web, es necesario considerar fundamentalmente los lenguajes que se encontrarán del lado del servidor entre ellos está PHP. Las interfaces de red establecidas por cada módulo del sistema permiten al servidor que ejecute el servicio de enrutamiento y acceso remoto para la comunicación. Los servicios en Symfony se definen como las acciones (representadas anteriormente con el estereotipo “symfony action”) agrupadas en el correspondiente módulo (clase con

el estereotipo “symfony module”) al que pertenecen, se destaca la relación que existe de flujo de información entre el componente módulo del lado del servidor. Las solicitudes enviadas desde el cliente son respondidas por el servidor específicamente en formato JSON.

Con JSON se intercambian los datos estructurados entre diferentes sistemas sobre protocolos como HTTP. Define como se deberán codificar los objetos de un determinado lenguaje para convertirlos en cadenas de texto, con el propósito de enviarlos hacia otras aplicaciones y que estas puedan reconstruirlos como objetos nativos, utilizando la decodificación definida por el mismo formato. Para el llamado de procedimientos hacia el servidor desde el cliente se emplea RPC, el cual a través del formato JSON.

JSON-RPC es un protocolo de llamadas a procedimientos remotos similar a XML-RPC, pero más ligero. Se trata de un protocolo para un mecanismo típico de RPC en donde hay un cliente y un servidor que establecen una conexión. El cliente invoca métodos remotos publicados por el servidor enviando un objeto request [4] con el nombre del método y sus parámetros. El servidor contesta con un response que contiene los datos de respuesta. Tanto los objetos request como los response [5]son paquetes HTTP cuyo recurso (información transmitida) son objetos JSON. Se representan los diferentes mensajes enviados del lado del servidor: Exception y SuccesAction. Puesto que es necesario tener en cuenta el hecho que cuando se invoca un servicio, donde esta invocación es asíncrona se hace imprescindible disponer de un observador que esté al tanto de la ocurrencia o del fallo de la operación, por esta razón es necesario definir los mensajes de tipo Exception.

El objeto request en formato JSON con el juego de datos de la encuesta está estructurado como se muestra en la siguiente figura:

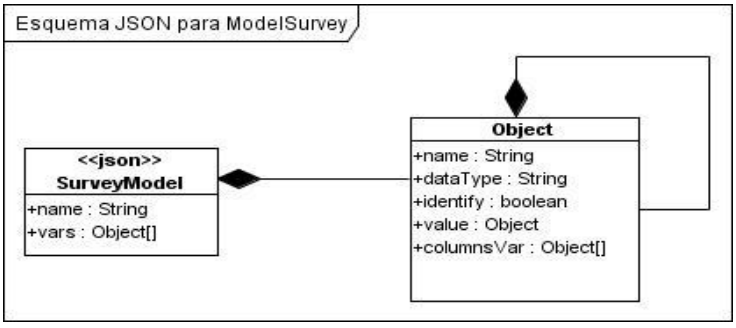

Figura 6: Formato JSON para los datos de la encuesta.

[4] Objeto Request: gestiona todo lo relativo a entrada de datos al script por parte del usuario (formularios), provenientes de otra URL, del propio servidor o del browser.
[5]Objeto Response: es una facilidad construida sobre el objeto Request que permite manipular respuestas de llamadas asíncronas en formato JSON.

El modelo se encarga de la abstracción de la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes de, por ejemplo, el tipo de gestor de bases de datos utilizado por la aplicación. Las clases de la capa del modelo se generan automáticamente, en función de la estructura de datos de la aplicación. La librería Propel se encarga de esta generación automática, ya que crea el esqueleto o estructura básica de las clases y genera automáticamente el código necesario. Cada tabla del modelo de datos genera 4 clases pero las verdaderas clases presentan la estructura de Clase.php y ClasePeer.php.

Las relaciones fundamentales entre estos elementos son de dependencia y las de flujo de datos. La relación de flujo de datos entre la interface y el modelo representa la correspondencia de las invocaciones del cliente con las acciones que del lado del servidor darán respuesta a las mismas, se transfiere tanto en la solicitud como en la respuesta el objeto con todos los atributos que se envían. (26)

2.7 Diagramas de Secuencia.

El diagrama de secuencia muestra las interacciones entre objetos, ordenadas en secuencia temporal durante un escenario concreto. Proporciona una realización física de la realización de casos de uso en términos de clases del diseño. A continuación se muestran los diagramas de secuencias de los escenarios del caso de uso Gestionar Categorías.

Sección Insertar categorías:

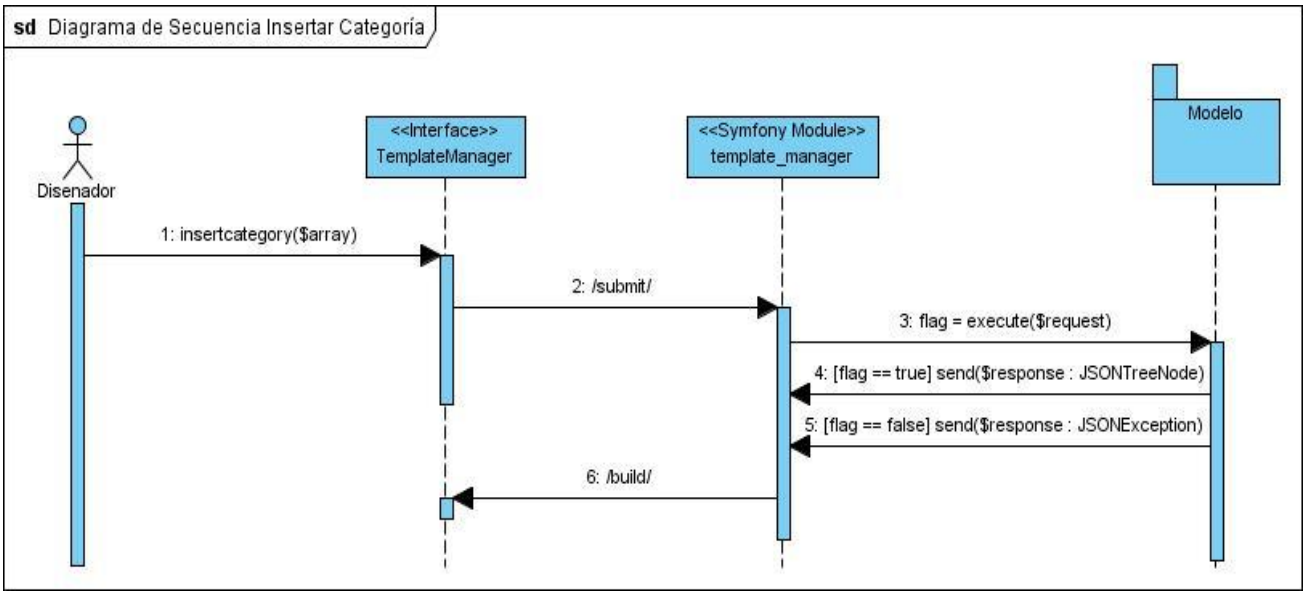

Figura 7: Diagrama de Secuencia de la sección Insertar categoría del CU Gestionar categoría

Sección Eliminar categoría:

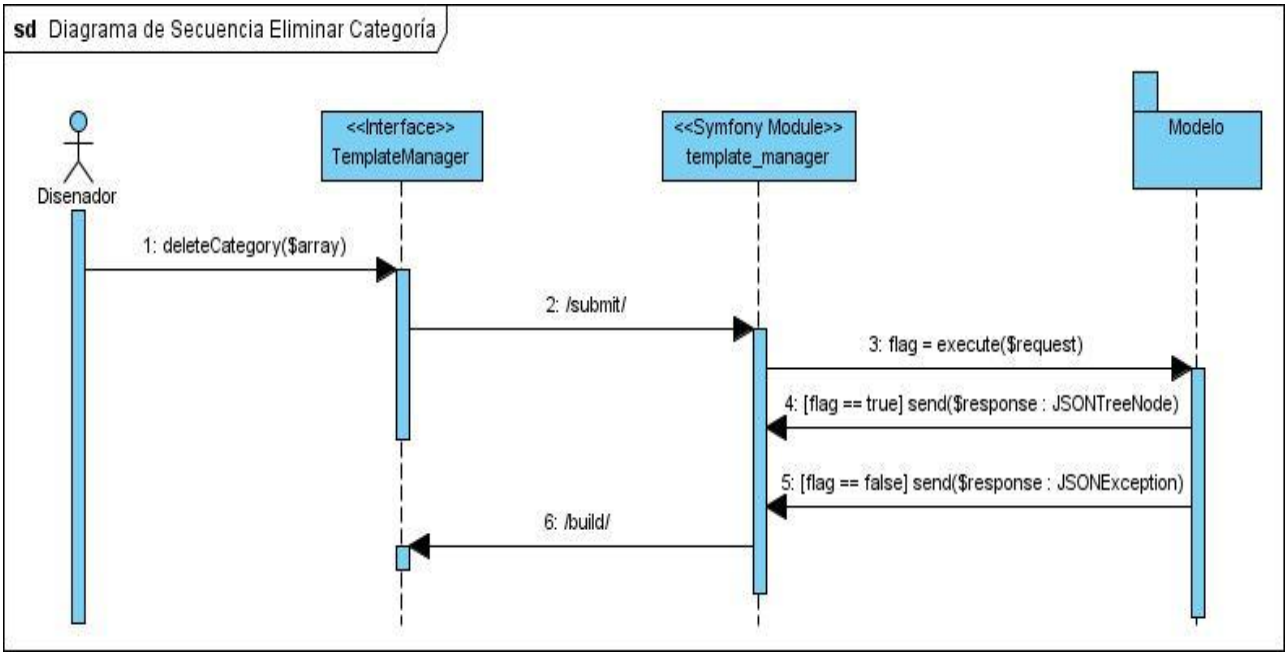

Figura 8: Diagrama de secuencia de la sección Eliminar categoría del CU Gestionar categoría

Sección Actualizar propiedades de la categoría:

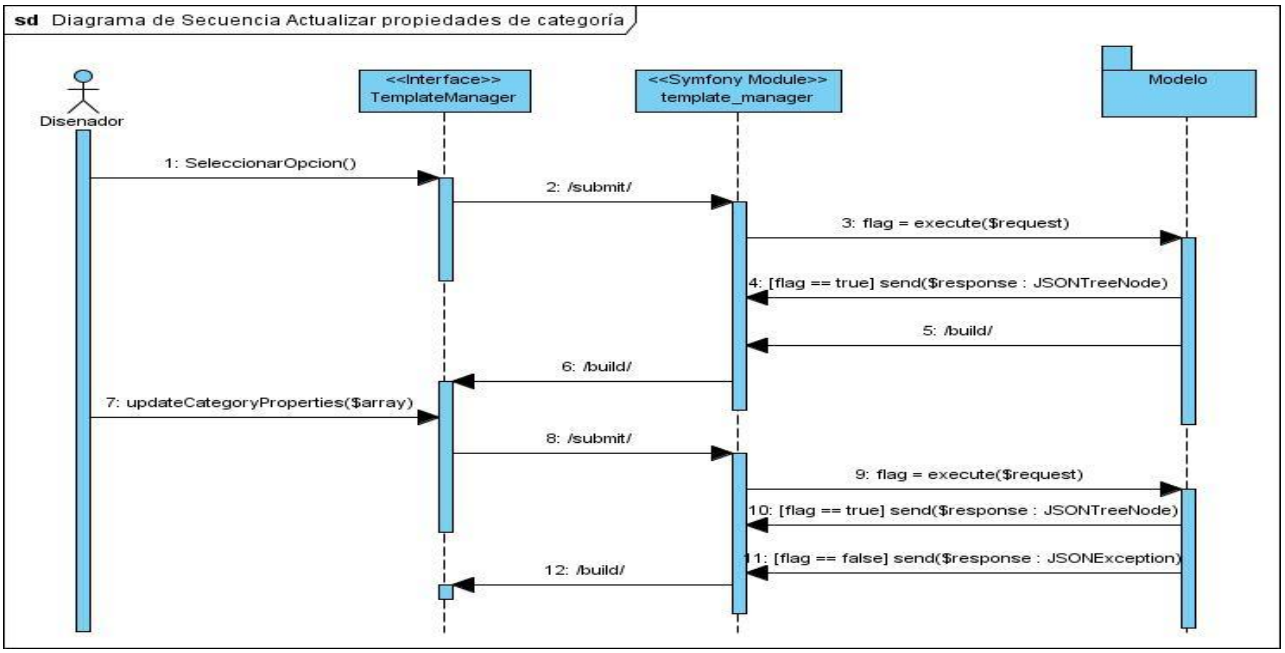

Figura 9: Diagrama de secuencia de la sección de Actualizar propiedades de la categorías del CU Gestionar categoría

Sección Renombrar categoría:

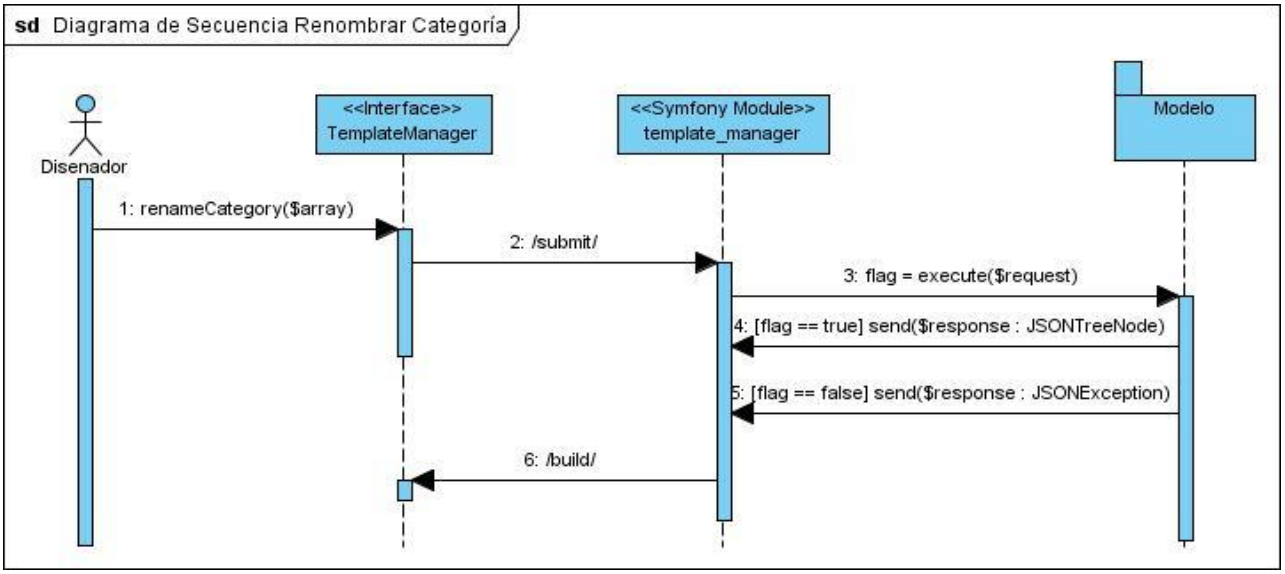

Figura 10: Diagrama de secuencia de la sección Renombrar categoría del CU Gestionar categoría

En symfony, la capa del controlador que contiene el código que une la lógica de negocio con la presentación, está dividida en varios componentes que se utilizan para diversos propósitos:

- El controlador frontal es el único punto de entrada a la aplicación. Carga la configuración y determina la acción a ejecutarse.
- Las acciones contienen la lógica de la aplicación. Verifican la integridad de las peticiones y preparan los datos requeridos por la capa de presentación.

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 escrita (o seleccionada) por el usuario. Luego de asociar la acción utiliza el modelo y define variables para la vista. Cuando se realiza una petición Web en la aplicación symfony, la URL define la acción y los parámetros de la petición.

El proceso de precisión de la acción al terminar es ejecutado en la capa del modelo, en el cual el acceso y la modificación de los datos almacenados en la base de datos se realizan mediante objetos; de esta forma nunca se accede de forma explícita a la base de datos. Este comportamiento permite un alto nivel de abstracción y permite una fácil portabilidad. La utilización de objetos en vez de registros y de clases en vez de tablas, tiene la ventaja de permitir añadir métodos de acceso en los objetos que no tienen relación directa con una tabla.

Cuando termina la ejecución de la acción, el valor retornado por el método de la acción determina como será producida la vista. Esta, definida en el diseño como Interface, se encarga de producir las páginas que se muestran como resultado de las acciones. Está compuesta por diversas partes, estando cada una de ellas especialmente preparada para que pueda ser fácilmente modificable por la persona que normalmente trabaja con cada aspecto del diseño de las aplicaciones.

2.8 Vista de Despliegue.

El diagrama de Despliegue provee un modelo detallado de la forma en la que los componentes se desplegarán a lo largo de la infraestructura del sistema. Detalla las capacidades de red, las especificaciones del servidor, los requisitos de hardware y otra información relacionada al despliegue del sistema propuesto. Esta vista representa el mapeo de componentes de software ejecutables con los nodos de procesamiento (hardware). Además, tiene en cuenta los siguientes requerimientos: disponibilidad del sistema, rendimiento y escalabilidad.

Un nodo es un objeto físico que representa un recurso informático, este recurso generalmente dispone de datos persistentes y capacidad de proceso. Las conexiones entre nodos muestran las líneas de comunicación con las que el sistema tendrá que interactuar. A continuación se muestra la estructura de los componentes que se desplegarán a lo largo de la infraestructura del sistema:

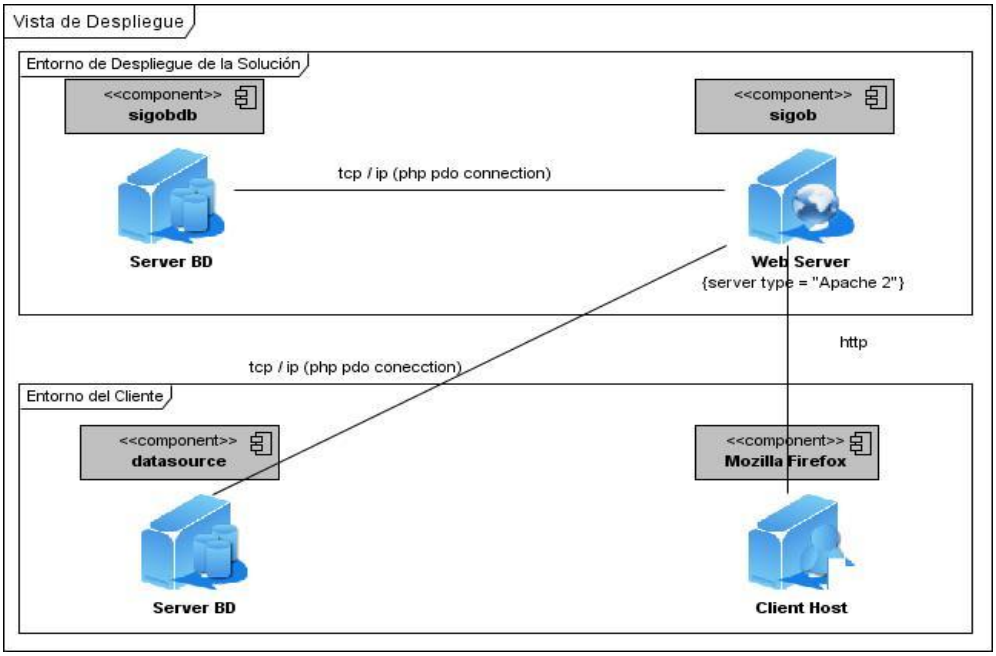

Figura 11: Vista de Despliegue de la Aplicación

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 HTTP (Hypertext Transfer Protocol) 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.

Protocolos de comunicación

Un protocolo de comunicación es un conjunto de reglas establecidas entre dos dispositivos para permitir la comunicación entre ambos.

- Conexión HTTP

  Es el protocolo utilizado entre los browser de los clientes y el servidor Web. Este elemento de la arquitectura representa un tipo de comunicación no orientado a la conexión entre clientes y servidor.

- TCP/IP

  Es la base del Internet que sirve para enlazar computadoras. El protocolo TCP/IP es utilizado para establecer la conexión entre el servidor de aplicación y el servidor de base de datos.

Servidor de Base de Datos

Se refiere a un servidor que radica en cada nodo regional en el cual el cliente define que sean guardados los datos. En el servidor central estarán almacenados todos los datos recopilados por todos los nodos regionales.

Conclusiones parciales

En el presente capítulo se tomó como punto de partida el modelo de dominio, los requisitos funcionales y no funcionales, quedando conformado el diagrama de casos de uso del sistema y el modelo de diseño. En el marco concebido se modeló la vista lógica enfatizada en las características que posee la arquitectura de Symfony por el patrón Modelo-Vista-Controlador. Como parte de la realización de los

casos de uso se modelaron los diagramas de secuencia, los cuales muestran una sucesión lógica de acciones para dar respuesta a las peticiones del cliente. La forma en que los componentes se desplegarán a lo largo de la infraestructura del sistema quedó descrita mediante el diagrama de despliegue, dando paso a la implementación de la aplicación.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA

En el capítulo correspondiente a las disciplinas de implementación y pruebas, se analizarán los artefactos principales como el Modelo de Implementación y el Modelo de datos generado a partir de las clases persistentes. Se comenzará a implementar el sistema en términos de componentes y se realizarán las pruebas para validar el mismo.

3.1. 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 representan información de larga duración, es decir, es la información que se requiere almacenar para gestionarla 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. Una de 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. A continuación se muestra el diagrama para el sistema a desarrollar: (31)

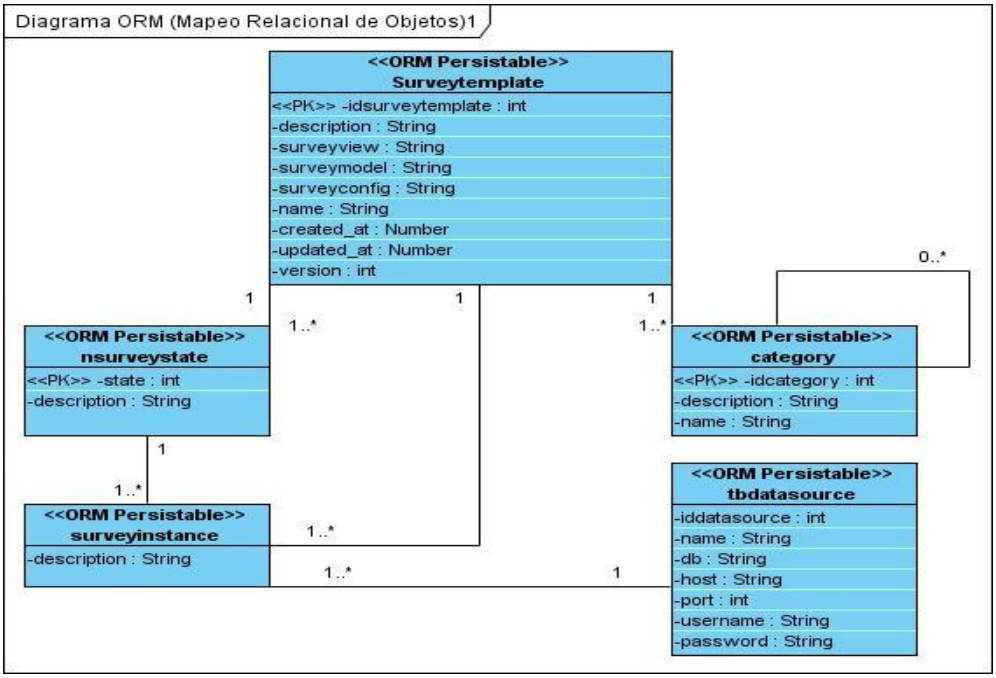

Figura 12: Diagrama de clases persistentes

3.2. Modelo de datos

Los modelos de datos aportan la base conceptual para diseñar aplicaciones que hacen un uso intensivo de datos. De igual forma, brindan la base formal para las herramientas y técnicas empleadas en el desarrollo y uso de sistemas de información. En general, 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. A continuación se muestra el modelo de datos del módulo Encuesta:

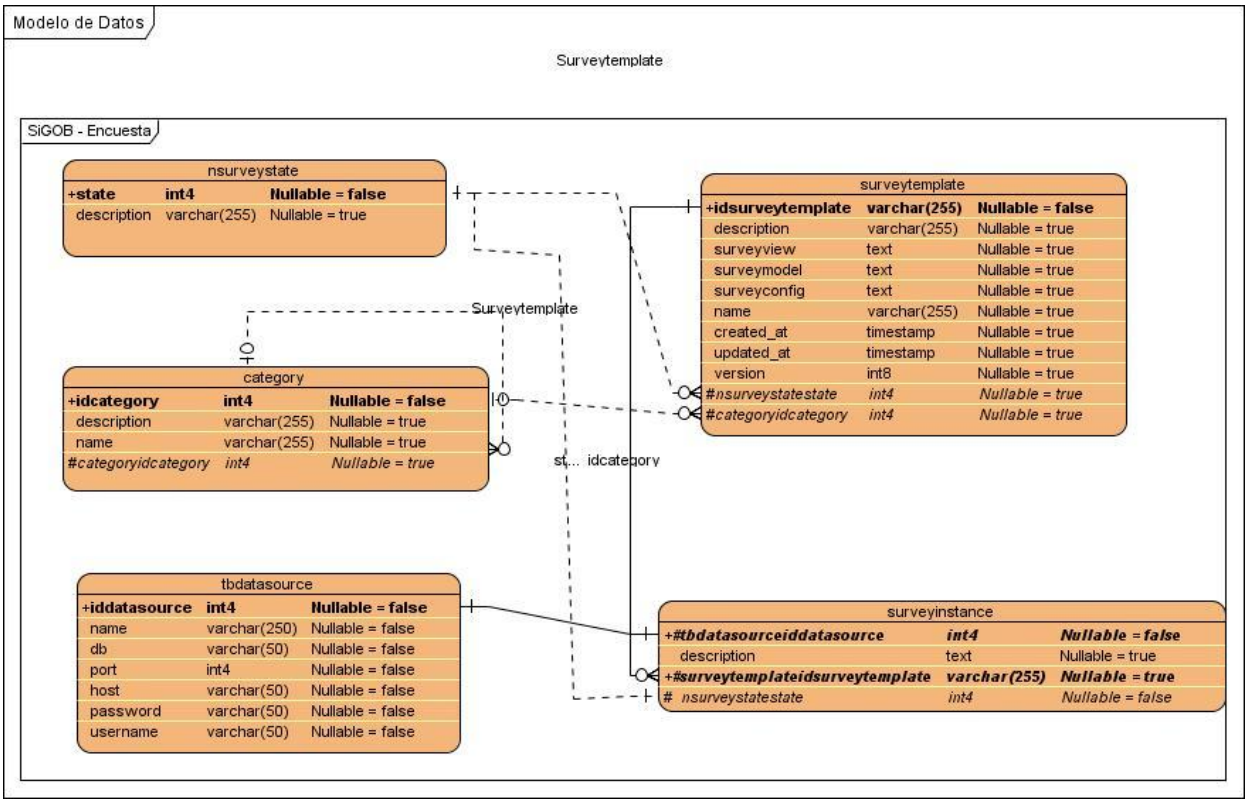

Figura 13: Modelo de Datos del módulo Encuesta

El tratamiento a los datos es algo primordial en un sistema, de cómo se persiste finalmente los resultados capturados por la encuesta. Para cada encuesta se crean un conjunto de tablas en correspondencia con el diseño de la misma. Se mantiene en una única tabla primaria los valores primitivos de la encuesta que se recuperan como un registro íntegro representado una instancia de captura. Algunos de estos campos (variables de la encuesta) primitivos identifican la instancia de la encuesta lo que se decide al diseñar la encuesta. Para los campos de múltiples valores como tablas o listas de múltiple selección, la información se almacena en tablas apartes donde la relación es respecto a la tabla primaria de 1 a muchos. Este modelo se crea físicamente en la base de datos en el momento en que se publica la encuesta y facilita el posterior procesamiento.

Un ejemplo de las tablas generadas para una encuesta de 6 variables donde las variable 5 es una tabla de cuatro columnas, y la variable 6 es una lista de selección múltiple. El resto de las variables son de tipos primitivos, donde se asume con identidad de la encueta al valor de la variable 4.

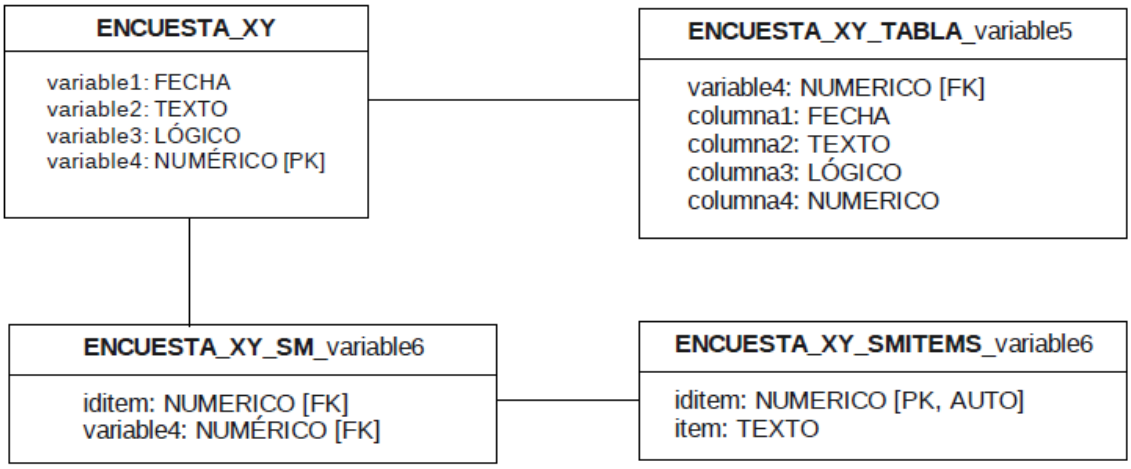

Figura 14: Ejemplo de representación para las tablas de la captura de los datos de las encuestas

Descripción de las tablas

Tabla 7: Tabla nsurveystate

Nombre: nsurveystate		
Descripción: Tabla nomencladora la cual almacena los 3 posibles estados de una plantilla.		
Atributo	**Tipo**	**Descripción**
state	int4	Identificador auto-incremental
description	varchar(255)	Contiene la descripción del estado de la plantilla

Tabla 8: Tabla category

Nombre: category		
Descripción: Almacena las categorías en las cuales se clasifican las plantillas de encuestas.		
Atributo	**Tipo**	**Descripción**
idcategory	int4	Identificador auto-incremental
description	varchar(255)	Almacena la descripción de la categoría de la plantilla
name	varchar(255)	Contiene el nombre de la categoría
categorysuper	int4	Llave foránea de categoría en caso de

		contener otra categoría

Tabla 9: Tabla surveytemplate

Nombre: surveytemplate		
Descripción: Almacena las plantillas y los datos de las encuestas.		
Atributo	**Tipo**	**Descripción**
idsurveytemplate	int4	Identificador auto-incremental
description	varchar(255)	Almacena la descripción de la plantilla
surveyview	text	Contiene la vista de la encuesta correspondiente
surveymodel	text	Contiene el modelo representado por la encuesta
surveyconfig	text	Contiene la configuración de la encuesta
name	varchar(255)	Almacena el nombre de la plantilla
created_at	timestamp	Contiene la fecha de la creación de la plantilla
updated_at	timestamp	Contiene la fecha de la actualización de la plantilla
version	int8	Versión en la cual se encuentra la plantilla
nsurveystate	int4	Llave foránea que representa el estado de la plantilla
idcategory	int4	Llave foránea que representa la categoría en la que se encuentra la plantilla

Tabla 10: Tabla tbdatasource

Nombre: tbdatasource		
Descripción: Almacena los datos del nodo origen para el cual se almacenarán las informaciones de las encuestas.		
Atributo	**Tipo**	**Descripción**
iddatasource	int4	Identificador auto-incremental
db	varchar(50)	Contiene el nombre de la base de datos
name	varchar(255)	Contiene el nombre de la provincia donde estará ubicado el nodo origen

port	int4	Puerto por la cual se conectará el servidor central
host	varchar(50)	Número de IP del nodo origen
username	varchar(50)	Usuario por el cual se conectara al nodo origen
password	varchar(50)	Contraseña del usuario

Tabla 11: Tabla surveyinstance

Nombre: surveyinstance		
Descripción: Almacena las instancias de las plantillas de encuestas publicadas en el nodo origen especificado.		
Atributo	**Tipo**	**Descripción**
iddatasource	int4	Identificador auto-incremental la cual es foránea de la tabla tbdatasource
description	varchar(255)	Almacena la descripción de los estados que puede tomar la instancia de la plantilla en una región del municipio o provincia.
idsurveytemplate	varchar(255)	Identificador que representa a la vez llave foránea de la tabla surveytemplate
nsurveystate	int4	Identificador auto-incremental la cual es foránea de la tabla nsurveystate que representa el estado de la instancia de la plantilla

3.3. 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 otros. Describe también la organización de los componentes de acuerdo a los mecanismos de estructuración y modularización disponibles en el entorno de implementación y lenguajes de implementación empleados, también cómo dependen los componentes unos de otros. (32)

3.3.1. Diagrama de Componentes

Un componente es el empaquetamiento físico de los elementos de un modelo, como son las clases en el modelo de diseño. “Los diagramas de componentes describen los elementos físicos y sus realizaciones en el entorno” de implementación y en el lenguaje o lenguajes de programación utilizados, y cómo dependen los componentes unos de otros. Muestra la organización y las dependencias entre un conjunto de componentes. (32)

Capa de Negocio

La composición de este componente se representa en la Figura 10. La aplicación Survey está compuesta por 3 paquetes de componentes los cuales representan las disímiles acciones o servicios que brinda el servidor, separadas por los diferentes módulos establecidos para el desarrollo del proyecto SiGOB. La carpeta lib recoge la librería dsExtDirectPlugin y el framework Symfony. El dsExtDirectPlugin es una adaptación de Symfony del router Ext.Direct en el marco Ext Js 3.x para crear aplicaciones de forma rápida y fiable. Incluye un generador de API que permite generar fácilmente el código JavaScript necesario para ejecutar métodos Ext.Direct JSON-RPC. El framework Symfony representado como paquete de componentes es necesario en la ejecución de las diferentes aplicaciones que representan los ficheros .php con las acciones y objetos del negocio que le dan funcionalidad al sistema.

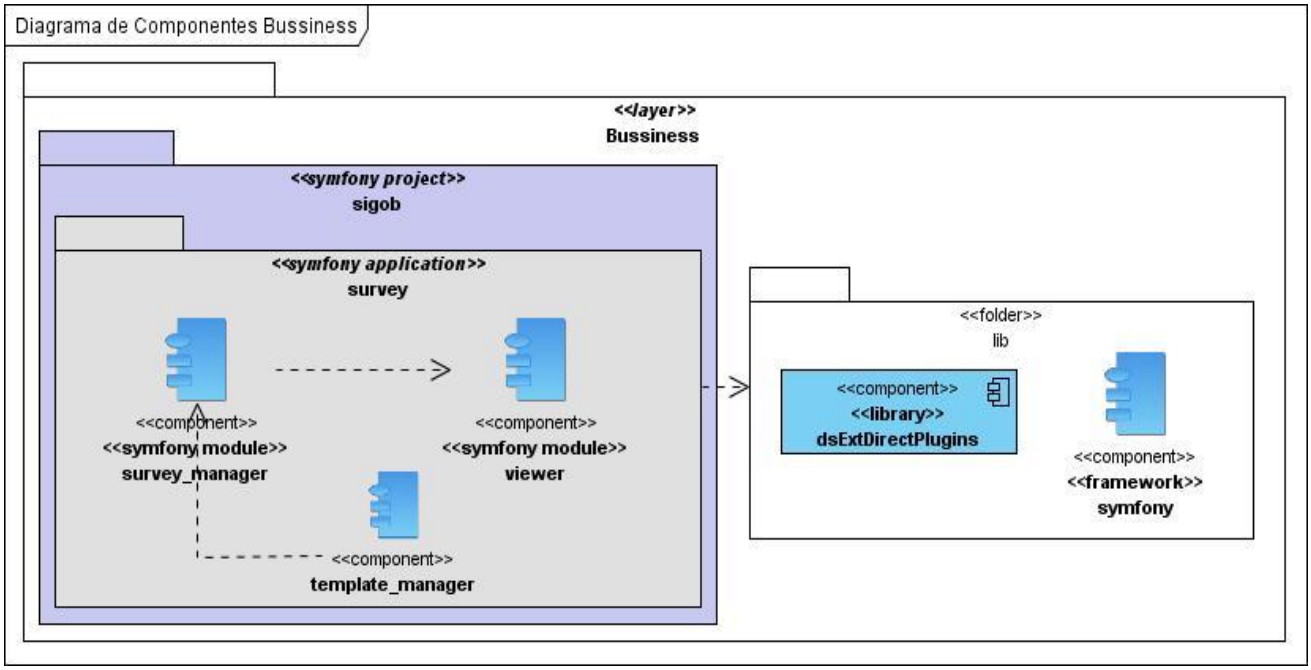

Figura 15: Diagrama de componentes de la capa de Negocio

Capa de Acceso a Datos

De manera idéntica a la capa de negocio se utiliza el intérprete de PHP integrado al servidor Apache con la extensión php5_pgsql para acceder al servidor de base de datos PostgreSQL. La capa de acceso a datos está compuesta por los componentes comunes del framework de acceso a datos Propel y por las librerías resultado del mapeo de la respectiva base de datos de SiGOB. La librería Propel genera automáticamente las clases de la capa del modelo, en función de la estructura de datos de la aplicación, ya que crea el esqueleto o estructura básica de las clases y genera automáticamente el código necesario. La abstracción de la base de datos es completamente invisible al programador, ya que la realiza otro componente específico llamado Creole. De esta manera, si se cambia el sistema gestor de bases de datos en cualquier momento, no se debe reescribir ni una línea de código, tan sólo es necesario modificar un parámetro en un archivo de configuración. Estas clases generadas se archivan en las carpetas de Modelo de Objetos (OM) y Mapeo a la Base de Datos (MAP). La carpeta OM contiene las clases base del modelo, generadas una vez que se analiza el esquema de la base de datos. Con respecto a la carpeta MAP la cual contiene meta información relativa a las tablas, es imprescindible para la ejecución de la aplicación. Los componentes que se encuentran en el Model representan al conjunto de clases de objetos que se encargan del acceso a datos.

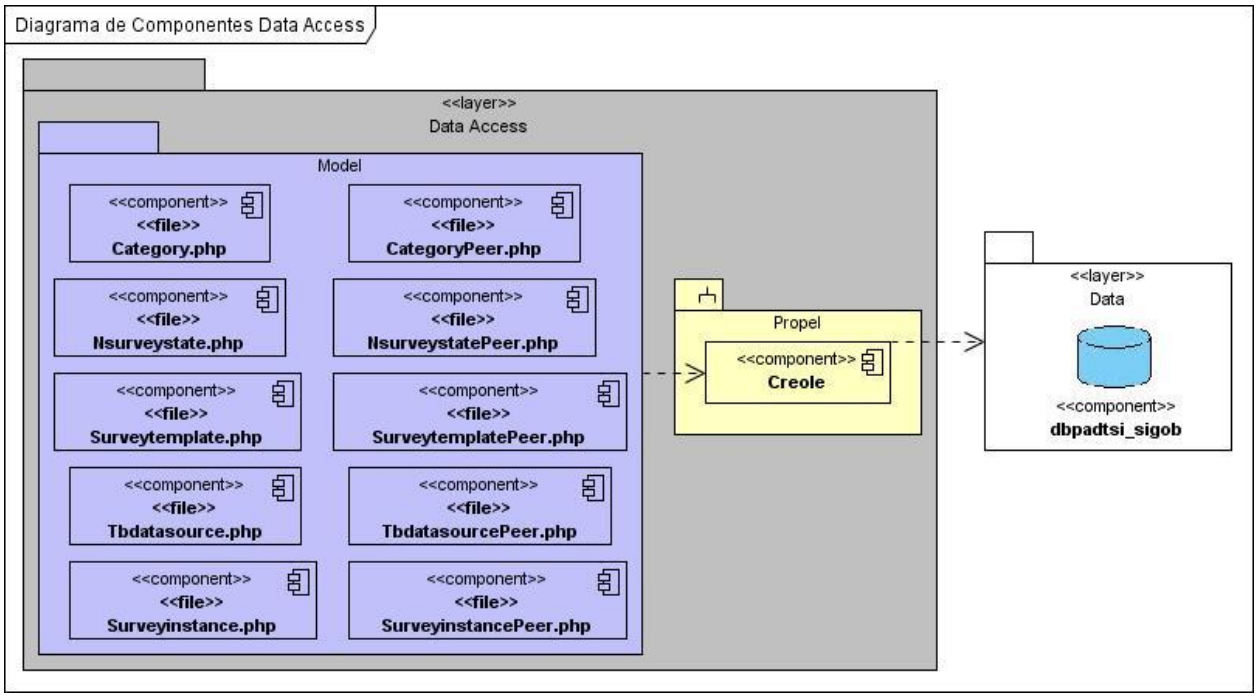

Figura 16: Diagrama de componentes de la capa de Acceso a Datos

3.4. Ejemplos de Códigos

El servicio de insertar la categoría recibe cómo parámetro un objeto request en formato JSON con los valores que construyen al objeto Category. Los valores son captados por el identificador del componente visual, entre los que se encuentra el nombre y la descripción de la categoría de tipo "String" y el identificador de otra categoría a la cual puede o no pertenecer. Luego de haber establecido los parámetros se llama al método remoto "insertCategry" del módulo "template_manager". Finaliza la ejecución con el retorno de un objeto response que contiene el formato JSON con el tipo de mensaje en dependencia si se efectuó o no satisfactoriamente el servicio.

```
/**
 * @extdirect-method insertCategory
 * @extdirect-enable
 * @extdirect-len 1
 */

public function executeInsertCategory($request)
{
    try
    {
        $description = $request->getParameter('description');
        $name = $request->getParameter('name');
        $categorysup = $request->getParameter('categorysuper');

        if (CategoryPeer::getCategory($name))
        {
            $category = new Category();
            $category->setDescription($description);
            $category->setName($name);
            $category->setCategorysuper($categorysup);
            $category->save();
        }

        $this->result = array(
            "success"  => true);
        return sfView::SUCCESS:

    } catch (Exception $exc)
    {
        $this->result = array(
            "success"  => false,
            "error" => array(
                "code" => $exc->getCode(),
                "message" => $exc->getMessage()));
        return sfView::SUCCESS;
    }
```

Figura 17: Ejemplo de código del método insertCategory

El servicio de renombrar la categoría recibe cómo parámetro un objeto request en formato JSON con los valores que contiene el objeto Category. Los valores son captados por el identificador del componente visual, entre los que se encuentra el nuevo nombre que se le asigna y el que lo representaba anteriormente de tipo "String". Luego de la entrada de estos valores, se llama al método remoto "insertCategry" del módulo "template_manager" y el sistema busca la existencia de la categoría seleccionada a través del método "getByName" perteneciente a la clase que mapea la entidad. Cuando encuentra el objeto categoría, se modifica el valor y se retorna un objeto response que contiene el formato JSON, con el tipo de mensaje en dependencia si se efectuó o no satisfactoriamente el servicio.

```
/**
* @extdirect-method renameCategory
* @extdirect-enable
* @extdirect-len 1
*/

public function executeRenameCategory($request)
{
    try
    {
        $nemaold = $request->getParameter('nameold');
        $name = $request->getParameter('name');
        $category = CategoryPeer::getByName($name);
        if($name != null)
        {
            $category->setName($name);
            $category->save();
            $this->result = array(
            "success"  => true);
            return sfView::SUCCESS;
        }
        else
            new Exception ("Datos Incompletos");
    } catch (Exception $exc)
    {
        $this->result = array(
            "success"  => false,
            "error" => array(
                "code" => $exc->getCode(),
                "message" => $exc->getMessage()));
        return sfView::SUCCESS;
    }
```

Figura 18: Ejemplo de código del método renameCategory

3.5. Modelo de Prueba

Las pruebas de aceptación se establecen por cada caso de uso para validar su correcta implementación. Cada prueba es especificada mediante un documento que establece las condiciones de ejecución, las entradas de la prueba y los resultados esperados.

El propósito de esta disciplina es:

- Proporcionar retroalimentación temprana y frecuente sobre si el sistema cumple los requisitos.
- Objetivamente medir los progresos realizados en pequeños incrementos.
- Identificar los problemas con la solución, aumentando la probabilidad de que el sistema se comportará correctamente.
- Asegurar que los cambios en el sistema no introducen nuevos defectos.
- Mejorar la velocidad, facilitando el descubrimiento de los temas con los requisitos, diseños e implementaciones lo más pronto posible.

La disciplina de pruebas es iterativa e incremental. Las pruebas se producen en cada iteración del ciclo de vida, comenzando con las primeras versiones de sistema. Esta disciplina desafía los supuestos riesgos e incertidumbres inherentes en el desarrollo de artefactos altamente técnicos y direcciones de esos problemas mediante el uso de demostración concreta y la evaluación imparcial.

En la evaluación dinámica del sistema se aplicó el nivel de trabajo Pruebas de Desarrollador, el cual está diseñado e implementado 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. La técnica de prueba seleccionada para evaluar los servicios que ofrece el sistema es la de Prueba funcional, la cual tiene como principal objetivo asegurar el trabajo apropiado de los requisitos funcionales, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados. A través del método de Caja Negra, también conocido como Pruebas de Comportamiento, se ejecutará cada caso de uso usando datos válidos e inválidos, para verificar que los resultados esperados ocurran cuando se usen datos válidos y se desplieguen los mensajes apropiados de error.

Para confeccionar los casos de prueba de Caja Negra se utilizó el criterio de la técnica de Partición de Equivalencia, donde se divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software. En esencia, esta técnica intenta dividir el dominio de entrada de un programa en un número finito de variables de equivalencia. De tal modo que se pueda asumir razonablemente que una prueba realizada con un valor representativo de cada variable es equivalente

a una prueba realizada con cualquier otro valor de dicha variable. Las variables de equivalencia representan un conjunto de estados válidos y no válidos para las condiciones de entrada de un programa. Se definen dos tipos de variables de equivalencia: las válidas, que representan entradas válidas al programa, y las no válidas, que representan valores de entrada erróneos, aunque pueden existir valores no relevantes a los que no sea necesario proporcionar un valor real.

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 que recoge el caso de uso en cuestión. 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 4 casos de uso del sistema, plasmándose en la documentación del proyecto. (Ver planilla Diseño de Casos de Prueba)

A continuación se presenta la tabla de secciones a probar en el caso de uso Gestionar Categorías:

Tabla 12: Secciones a probar en el Caso de Uso

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC1: Actualizar categoría	EC 1.1: Actualizar categoría	El diseñador decide cambiar elementos de la categoría, el sistema envía un mensaje indicando que la petición se realizó exitosamente, terminando así el CU.
SC2 Guardar categoría	EC 2.1: Guardar categoría	El diseñador solicita guardar la categoría, enviando el sistema un mensaje indicando que la petición se ejecutó exitosamente.

SC3 Eliminar categoría	EC 3.1: Eliminar categoría	El sistema elimina la categoría seleccionada. Envía el sistema un mensaje indicando que la petición se ejecutó exitosamente, terminando así el CU.
SC4 Renombrar categoría	EC 4.1: Renombrar categoría	Se introduce el nuevo nombre de la categoría, el sistema muestra un mensaje de confirmación de la petición, culminando así el CU.

Partiendo de esta descripción, se detallan las variables que se encuentran en las interfaces asociadas al caso de uso.

Tabla 13: Descripción de la variable

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	idcategory	Entero	No	El identificador de la categoría que representa un auto-incremental.
2	name	String	No	Obtiene el nombre de la categoría, está compuesto solo por letras y algunos símbolos de separación como _ ( ) , ; . : / ^ - [ ]
3	description	String	Si	Obtiene la descripción de la categoría, está compuesto solo por letras y algunos símbolos de separación como _ ( ) , ; . : / ^ - [ ]
4	newname	String	No	Obtiene el nuevo nombre de la categoría, está compuesto solo por letras y algunos símbolos de separación como _ ( ) , ; . : / ^ - [ ]

Esta descripción posibilitó que se realizara una matriz de datos, donde se evaluó y probó la validez de cada uno de los datos introducidos en el sistema, específicamente en la sección que se estuvo

probando. Utilizando un juego de datos válidos e inválidos se identificó el empleo de la técnica de partición de equivalencia.

Matriz de Datos

Tabla 14: SC 1 Actualizar categoría

Escenario	Variables (Enumeradas según descripción de la variable)				Respuesta del Sistema	Resultado de la Prueba	Flujo Central
	1	2	3	4			
Actualizar categoría	V	V Cultura	NA	V Danza	El sistema guarda los nuevos cambios de la encuesta en BD, enviando un mensaje	Satisfactorio	1. El sistema envía los datos codificados en formato JSON, retornando al cliente la respuesta de confirmación de la ejecución.
	V	I Deporte	NA	V Salud	El sistema devuelve un mensaje tipo Exception debido a que ha ocurrido un error con la búsqueda del parámetro en la BD.	Satisfactorio	

Tabla 15: SC 2 Guardar categoría

Escenario	Variables (Enumeradas según descripción de la variable)			Respuesta del Sistema	Resultado de la Prueba	Flujo Central
	1	2	3			

Guardar categoría	V	V Cultura	NA	El sistema guarda la categoría en BD, enviando un mensaje	Satisfactorio.	2. El sistema envía los datos codificados en formato JSON, retornando al cliente la respuesta de confirmación de la ejecución.
	V	I Cultura	NA	El sistema devuelve un mensaje tipo Exception debido a que ha ocurrido un error con la entrada del parámetro en la BD.	Satisfactorio	

Tabla 16: SC 3 Eliminar categoría

Escenario	**V 2**	**Respuesta del Sistema**	**Resultado de la Prueba**	**Flujo Central**
Eliminar categoría.	V Cultura	El sistema elimina la categoría y devuelve un mensaje tipo success verdadero.	Satisfactorio	1. El sistema elimina la plantilla de encuesta y devuelve un mensaje de confirmación de la petición realizada.
	I Danza	El sistema devuelve un mensaje tipo success false debido a datos incorrectos de	Satisfactorio	

Tabla 17: SC 4 Renombrar categoría

Escenario	**Variables (Enumeradas según descripción de la variable)**		**Respuesta del Sistema**	**Resultado de la Prueba**	**Flujo Central**
	2	**4**			

Renombrar categoría	V Cultura	V SLD	El sistema guarda el nuevo nombre de la categoría, enviando un mensaje tipo success verdadero.	Satisfactorio.	3. El sistema envía los datos codificados en formato JSON, retornando al cliente la respuesta de confirmación de la ejecución.

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 4 no conformidades significativas, las cuales fueron resueltas, culminando el presente trabajo con resultados satisfactorios.

Conclusiones parciales

Como resultado de este capítulo se obtuvo la implementación del sistema en términos de componentes, proporcionando solución a los requisitos especificados en el capítulo anterior. Para ellos se representaron las clases persistentes y la descripción de la estructura de tablas sobre las que se establece el modelo de datos del sistema, así como sus atributos y relaciones. Además, se estructuraron las clases del diseño en paquetes de componentes mostrando la organización y sus dependencias entre los componentes que conforman el sistema. Una vez concebida la estructura y la implementación del sistema, se realizaron los casos de pruebas para validar la completitud de los requerimientos, obteniéndose resultados satisfactorios.

CONCLUSIONES

Para el desarrollo del componente para la captura de datos del Sistema de Información de Gobierno (SiGOB), Módulo Encuesta:

- Se analizaron los preceptos teóricos y técnicos que sustentaron la investigación, 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 modelo de diseño basado en la arquitectura de referencia del departamento de Integración de Soluciones, proporcionando la entrada apropiada y el punto de partida para las actividades de implementación.
- Se implementó el componente que posibilitará la captura de datos para Sistema de Información de Gobierno (SiGOB): Módulo Encuesta.
- Las pruebas realizadas para validar las funcionalidades que fueron implementadas, demostraron que los indicadores de calidad cumplieron satisfactoriamente con los requisitos, garantizando su correcto funcionamiento.

RECOMENDACIONES

Luego de haber analizado los resultados del presente trabajo de diploma, surgen algunas ideas que podrían ser incorporadas en un futuro con el objetivo de fortalecer el sistema desarrollado, por lo que se recomienda:

- Integrar la interfaz del Módulo Encuesta perteneciente al proyecto Sistema de Información del Gobierno (SiGOB) con el componente para la captura de datos desarrollado.
- Adicionar e implementar un requisito correspondiente a lograr el uso desconectado de la capa de presentación con el componente desarrollado en el presente Trabajo de Diploma.
- Realizar pruebas de carga y estrés al componente para la captura de datos.

REFENCIAS BIBLIOGRÁFICAS

1. **Allendes, José Miguel Santibáñez**. [En línea] 27 de Octubre de 2009. [Citado el: 10 de Enero de 2011.] http://jms.caos.cl/si/si02.html.
2. **Valle, José Guillermo. Monografías**. [En línea] 2005. [Citado el: 10 de Enero de 2010.] http://www.monografias.com/trabajos24/arquitectura-cliente-servidor/arquitectura-cliente-servidor.shtml?monosearch.
3. **Hernández, Marta Alelú. AUM.es**. [En línea] 2010. [Citado el: 12 de Noviembre de 2010.] http://www.uam.es/personal_pdi/stmaria/jmurillo/InvestigacionEE/Presentaciones/Encuesta_ppt.pdf.
4. **Empresariales, Departamento Docente Central de Ciencias**. Entorno Virtual de Aprendizaje. [En línea] 1 de Enero de 2011. [Citado el: 12 de Enero de 2011.] http://eva.uci.cu.
5. **U.S. Census Bureau. U.S. Census Bureau, Population Division**. [En línea] 11 de Junio de 2010. [Citado el: 15 de Noviembre de 2010.] http://www.census.gov/ipc/www/cspro/download/start40.pdf.
6. **Microsoft, Corporation.** Microsoft Office. [En línea] 2010. [Citado el: 10 de Noviembre de 2010.] http://office.microsoft.com/es-es/infopath-help/informacion-general-del-producto-microsoft-office-infopath-2007-HA010165634.aspx.
7. **IBM, Corporation.** SPSS. [En línea] 2010. [Citado el: 12 de Noviembre de 2010.] http://www.spss.com/es/software/stat.
8. **Institute, SAS.** SAS. [En línea] 2010. [Citado el: 15 de Noviembre de 2010.] http://www.sas.com/offices/latinamerica/andean/software/index.html.
9. **Valencia, Néstor Díaz.** Centro de Competencia Morfeo. *Formación Morfeo.* [En línea] 2009. [Citado el: 11 de Noviembre de 2010.] http://formacion.morfeo-project.org/wiki/index.php/PT3:GPSL:UF6.
10. **Enciclopedia Libre**. [En línea] 11 de Diciembre de 2008. [Citado el: 12 de Noviembre de 2010.] http://enciclopedia.us.es/index.php/Lenguaje_de_programaci%C3%B3n.
11. **Sitio Tuturiales.** [En línea] 2010. [Citado el: 12 de Noviembre de 2010.] http://www.sitiotutoriales.com/#cphp.
12. JSON. [En línea] 2010. [Citado el: 12 de Noviembre de 2010.] http://www.json.org/json-es.html.
13. GS1 Panama. [En línea] Junio de 2006. [Citado el: 10 de Noviembre de 2010.] http://www.gs1pa.org/boletin/2005/junio/boletin-jun05-art3.html.
14. **Evan, Clark.** YAMAL. [En línea] 1 de Octubre de 2009. [Citado el: 10 de Noviembre de 2010.] http://www.yaml.org/spec/1.2/spec.html.
15. **Cornejo, José Enrique González.** Document Information Retrieval Systems. *DocIris.* [En línea] 1998. [Citado el: 13 de Noviembre de 2010.] http://www.docirs.cl/uml.htm.
16. **Eguiluz, Javier.** Symfony.es. [En línea] 2010. [Citado el: 12 de Noviembre de 2010.] http://www.symfony.es/que-es-symfony/.
17. Sencha. [En línea] 2010. [Citado el: 16 de Noviembre de 2010.] http://www.extjs.com/products/extjs/
18. **Arévalo, Carlos.** ICA Instituto de Cálculo Aplicado. [En línea] Enero de 2001. [Citado el: 12 de Noviembre de 2010.] http://www.ica.luz.ve/carevalo/procesamiento-distribuido-1/c31.html.

19. **Proyejt, Propel.** Propel Guía de Usuario. [En línea] 2004. [Citado el: 15 de febrero de 2011.] http://propel.phpdb.org/docs/es/user_guide/.
20. **Wikipedia**. [En línea] 16 de Noviembre de 2010. [Citado el: 18 de Noviembre de 2010.] http://es.wikipedia.org/wiki/Herramienta_CASE.
21. **Sitio de Descarga de Software**. [En línea] 5 de Marzo de 2007. [Citado el: 16 de Noviembre de 2010.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/.
22. **NetBeans**. [En línea] 2010. [Citado el: 13 de Noviembre de 2010.] http://netbeans.org/community/releases/69/.
23. **pgADMIN** . [En línea] 2010. [Citado el: 16 de Noviembre de 2010.] http://www.pgadmin.org/docs/.
24. **PostgreSQL-es**. [En línea] 2 de Octubre de 2010. [Citado el: 12 de Noviembre de 2010.] http://www.postgresql-es.org/sobre_postgresql.
25. **Tecnología y Synergix**. [En línea] 2009. [Citado el: 2011 de Febrero de 16.] http://synergix.wordpress.com/2008/07/07/requisito-funcional-y-no-funcional/.
26. **Lobo, Armando Robert.** Arquitectura de Software para el Sistema Integrado de Gestión Estadística 2.0 Nuragas. Ciudad Habana : Trabajo de Diploma, 2009.
27. **Productiva, Dirección de Calidad de la Infraestructura.** Modelo del Sistema. [En línea] 5 de Mayo de 2010. [Citado el: 24 de febrero de 2011.] https://repositorio.datec.prod.uci.cu/svn/cbd_gestion/Datos%202009-2010/3%20Proyecto/4%20SiGOB/expediente_proyecto/1.%20Ingenier%C3%ADa/1.1%20Requisitos/.
28. **Pressman, Roger S**.; 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.
29. **Potencier, Fabien.** Libros Web. [En línea] 24 de Diciembre de 2008. [Citado el: 2011 de Febrero de 26.] http://www.librosWeb.es/symfony_1_1.
30. **Productiva, Dirección de Calidad de la Infraestructura.** Repositorio DATEC. [En línea] 2011. [Citado el: 15 de Febrero de 2011.] https://repositorio.datec.prod.uci.cu/svn/cbd_gestion/Datos%202009-2010/3%20Proyecto/4%20SiGOB/expediente_proyecto/1.%20Ingenier%C3%ADa/1.2%20Arquitectura%20y%20Dise%C3%B1o/.
31. **JACOBSON, G. B. J. R. I.** *El Proceso Unificado del Desarrollo de Software.* 2000.
32. **JACOBSON, I.; G. BOOCH**, et al. El proceso Unificado de Desarrollo de Software. p.xviii Addison Wesley Object Technology.

BIBLIOGRAFÍA

- **JACOBSON, I.; G. BOOCH**, et al. *El proceso Unificado de Desarrollo de Software*. p.xviii Addison Wesley Object Technology.
- **LOBO, Armando Robert.** *Arquitectura de Software para el Sistema Integrado de Gestión Estadística 2.0 Nuragas*. Ciudad Habana: Trabajo de Diploma, 2009.
- **PRESMAN, Roger S**.; 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.
- **PRODUCTIVA, Dirección de Calidad de la Infraestructura.** Repositorio DATEC. [En línea] 2011. [Citado el: 15 de Febrero de 2011.] https://repositorio.datec.prod.uci.cu/svn/cbd_gestion/Datos%202009-2010/3%20Proyecto/4%20SiGOB/expediente_proyecto/1.%20Ingenier%C3%ADa/1.2%20Arquitectura%20y%20Dise%C3%B1o/.
- **POTENCIER, Fabien y Zaninotto, Francois.** *Symfony 1.2 La guía Definitiva.* France: Sensio SA, 2008.

ANEXOS

Figura 19: Diagrama de Secuencia de la sección Consultar registro de plantillas del CU Administrar Plantilla

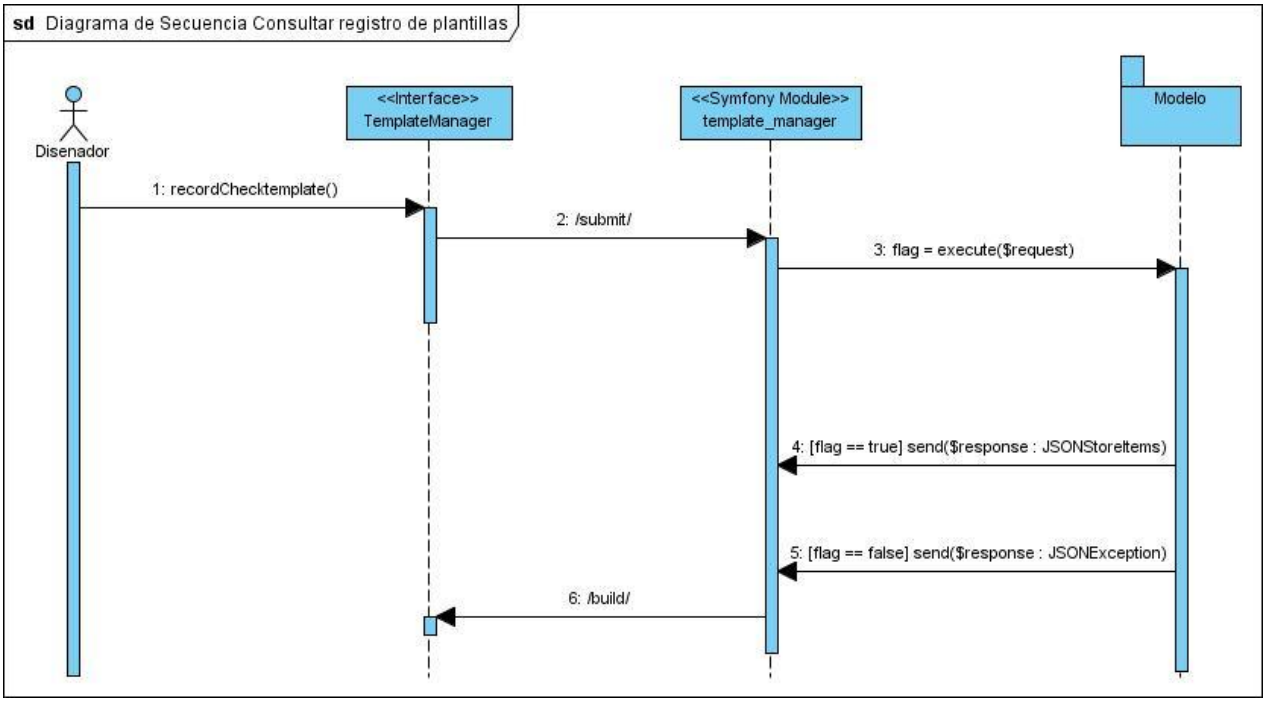

Figura 20: Diagrama de Secuencia de la sección Guardar plantillas del CU Administrar Plantilla

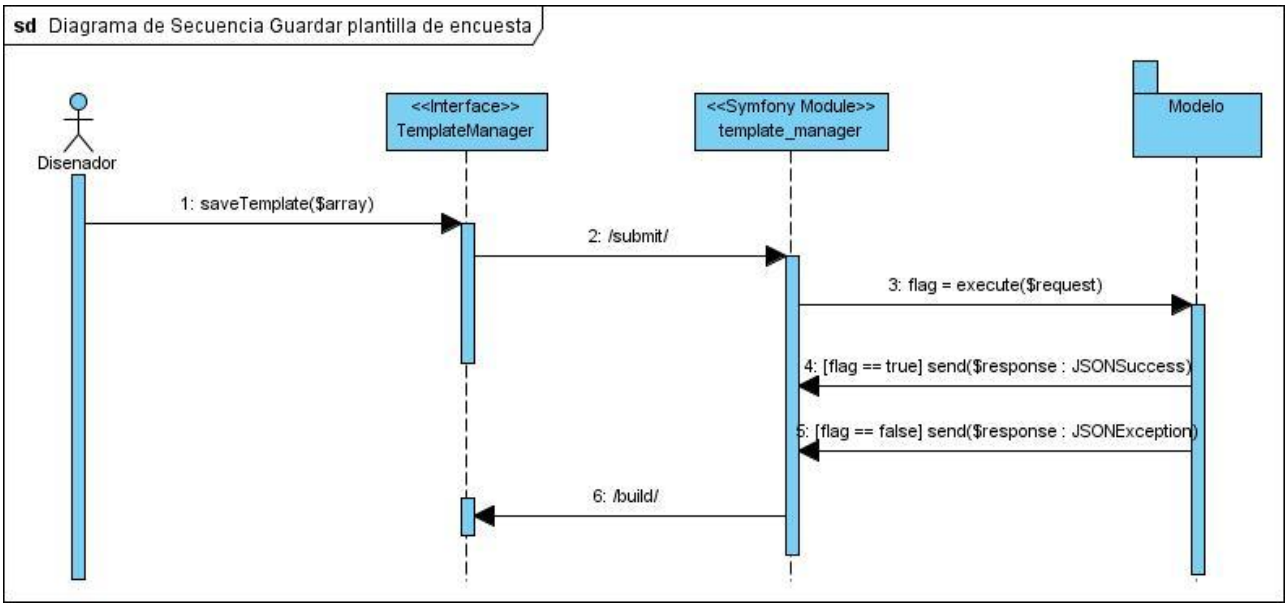

Figura 21: Diagrama de Secuencia de la sección Eliminar plantillas del CU Administrar Plantilla

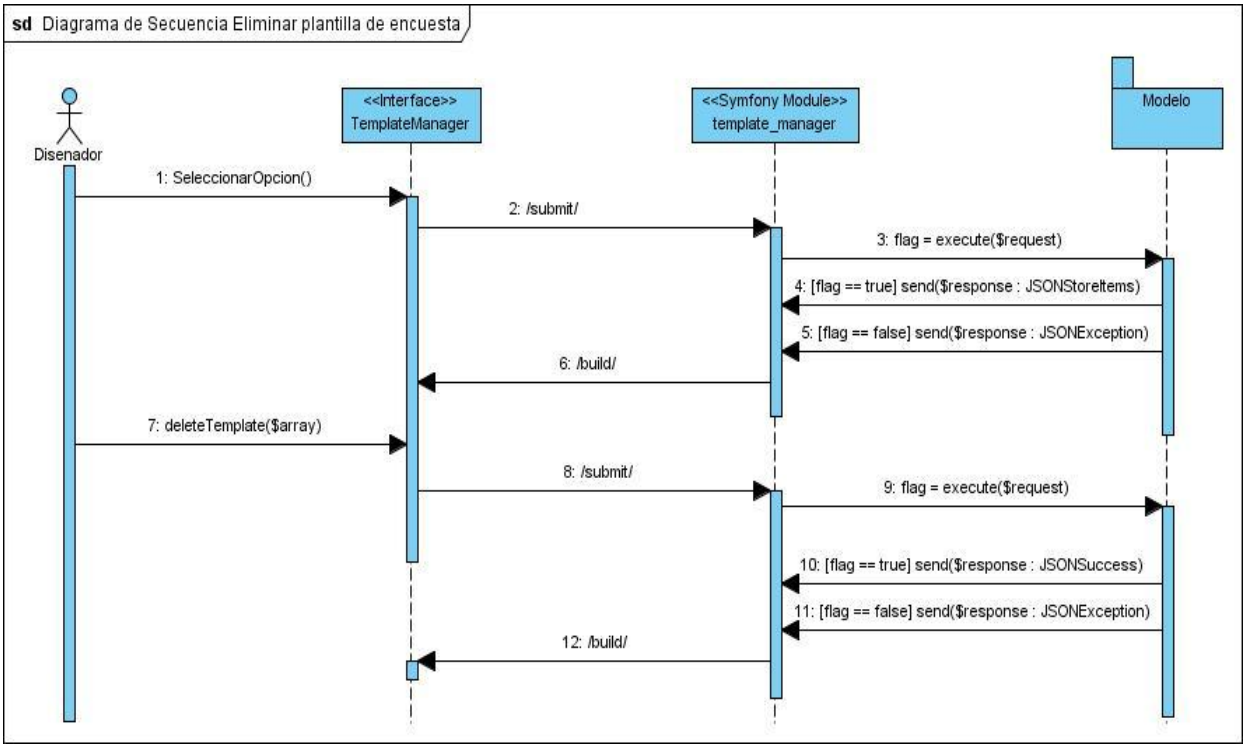

Figura 22: Diagrama de Secuencia de la sección Publicar plantillas del CU Administrar Plantilla

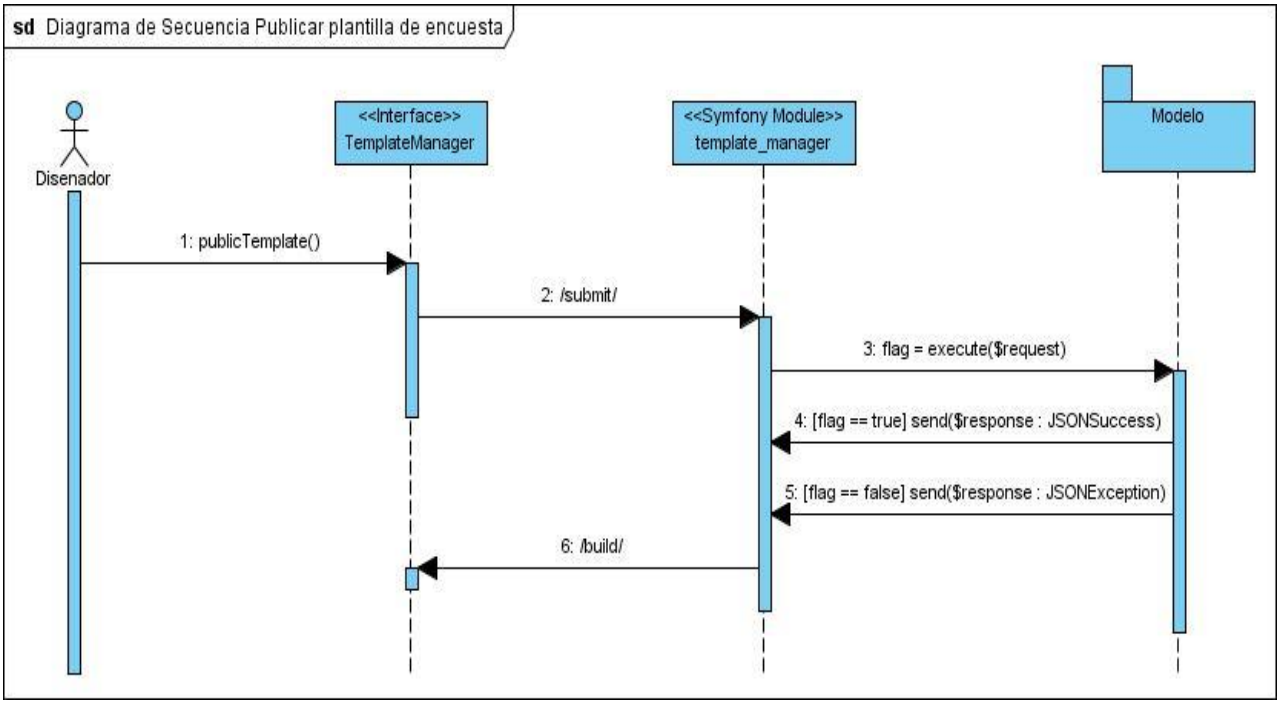

Figura 23: Diagrama de Secuencia de la sección Deshabilitar/Habilitar plantillas del CU Administrar Plantilla

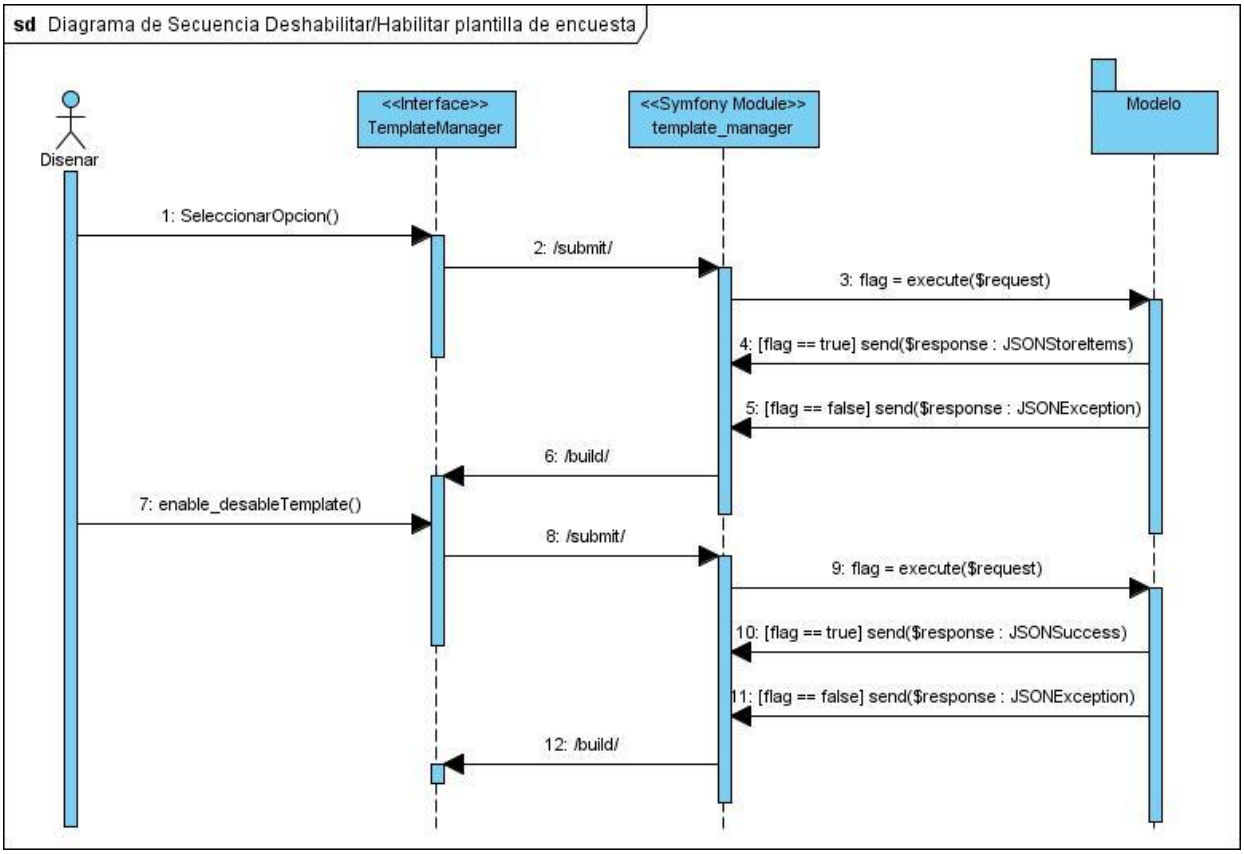

Figura 24: Diagrama de Secuencia de la sección Insertar Datos de encuesta del CU Digitar Plantilla de Encuesta

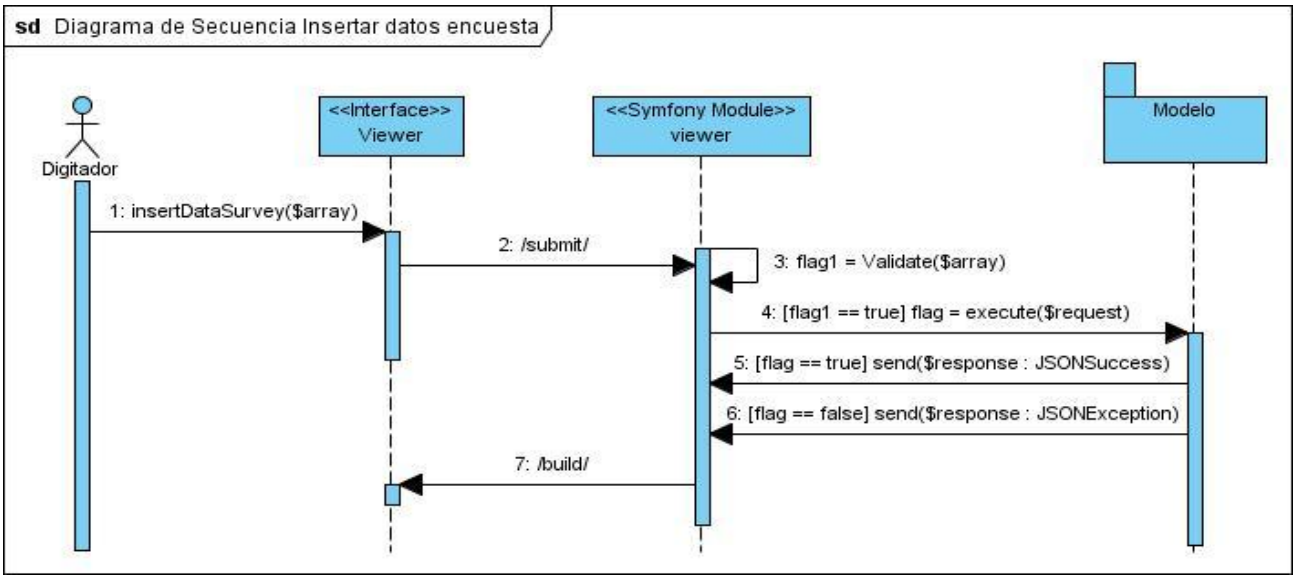

Figura 25: Diagrama de Secuencia de la sección Exportar encuesta del CU Administrar Encuesta

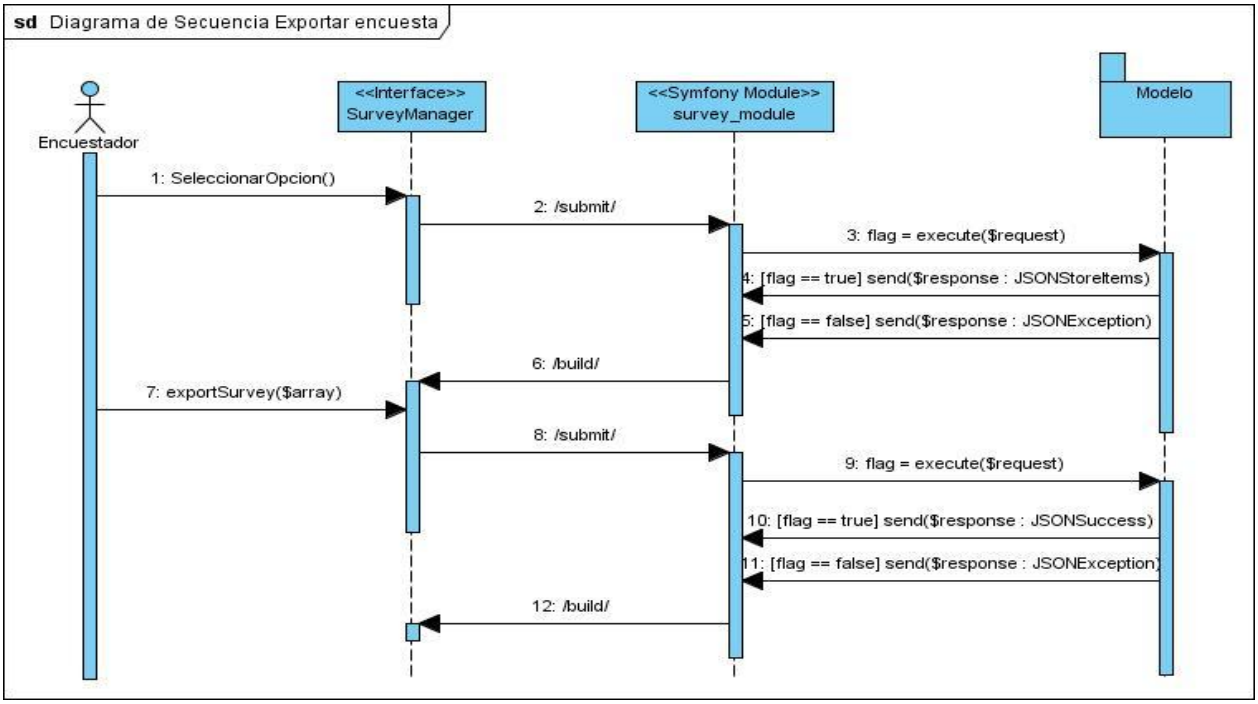

Figura 26: Diagrama de Secuencia de la sección Guardar encuesta del CU Administrar Encuesta