Universidad de las Ciencias Informáticas

“Facultad 6”

“Componente para exportar e importar estructuras de reportes en el Generador dinámico de reportes en su versión 1.8”

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

Autora: Alejandro Cesar Fernández Garcés
Michel Santos Milanés Luqués

Tutor(es): Ing. Clara Elena Brizuela Figueredo
Ing. Flavio Enrique Roche Rodríguez

La Habana, julio 2016
“Año 58 de la Revolución”

"La vida no es la que uno vivió, sino la que uno recuerda y cómo la recuerda para contarla"

Gabriel García Márquez.

Declaración de autor

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

Para que así conste firmo la presente a los ____ días del mes de _____ del año 2016.

Alejandro Cesar Fernández Garcés

Firma de Autor

Michel Santos Milanés Luqués

Firma de Autor

Ing. Clara Elena Brizuela Figueredo

Firma de Tutora

Ing. Flavio Enrique Roche Rodríguez

Firma de Tutor

Datos de contacto

Tutora: Ing. Clara Elena Brizuela Figueredo

Universidad de las Ciencias Informáticas, La Habana, Cuba

Especialidad de graduación: Ingeniería en Ciencias Informáticas

Correo Electrónico: cebrizuela@uci.cu

Tutor: Ing. Flavio Enrique Roche Rodríguez

Universidad de las Ciencias Informáticas, La Habana, Cuba

Especialidad de graduación: Ingeniería en Ciencias Informáticas

Correo Electrónico: froche@uci.cu

Autor: Alejandro Cesar Fernández Garcés

Universidad de las Ciencias Informáticas, La Habana, Cuba

Correo Electrónico: acfernandez@estudiantes.uci.cu

Autor: Michel Santos Milanés luqués

Universidad de las Ciencias Informáticas, La Habana, Cuba

Correo Electrónico: msmilanes@estudiantes.uci.cu

Dedicatoria

De Alejandro:

Entre mis grandes aspiraciones, metas y sueños siempre estuvo presente ser un graduado de nivel superior. En estos momentos aún no puedo creer que ya lo haya logrado, pero ya es un hecho, ya soy todo un ingeniero, bueno eso dice mi título. Justo ahora me vienen muchas personas a la mente y es cuando me doy cuenta de que este gran logro va dedicado a:

> *En primer lugar, dedico esta tesis a mis padres Ana Gloria Garcés Blanco y Augusto Cesar Fernández, pues sin su apoyo incondicional nada de esto fuese posible, por enseñarme todo lo que sé y prepararme para la vida en todos los sentidos.*
> *En segundo lugar, una mujer que para mí es la mejor profesora que cualquier estudiante puede tener, la profesora que con una sonrisa inmensa llegaba, daba 5 minutos de clases de los cuales salías listo para presentarte a la prueba más difícil del mundo y la otra hora y 25 minutos de lo pasaba criticando y jodiendo a todos en el aula principalmente a los negros.*
> *la profesora Yenelis Benítez.*
> *En tercer lugar, dedico esta tesis a todos los miembros de mis familias adicionales que, aunque no tenemos la misma sangre, siempre me han apoyado y me han dado ánimos para seguir adelante con mis sueños. Realmente tengo una familia increíble y pienso que al menos un pedacito de esta felicidad es gracias a ellos y a ustedes más que a nadie también va dedicada esta tesis.*

De Michel:

A mis padres, mi hermana, mi sobrina, mi cuñado, mi tío Raúl y a mi abuela Ernestina que sé que disfrutaron mucho de este triunfo en mi vida.
A toda aquella persona que en un futuro va a utilizar esta tesis para su desarrollo profesional.

Agradecimientos

De Alejandro:

Un sueño se hace realidad cuando se lucha día a día por él, cuando se pone todo el empeño y dedicación para alcanzarlo. Hoy se hizo realidad mi sueño, después de muchos años de sacrificio he alcanzado la meta que una vez me propuse y por eso quiero agradecer a todas aquellas personas que de una forma u otra han estado apoyándome en todo momento: profesores de ayer y hoy, amigos de ayer y hoy, estando siempre en las buenas y malas junto a mí, animándome a seguir adelante ante cada nuevo reto que la vida me ha presentado, a todos los que un día me dijeron cuenta conmigo, muchísimas gracias, pero especialmente a:

A la mujer más buena y más comprensiva del mundo, mi mamá, aunque no estés presente hoy esto no hubiese sido posible de no ser por todo lo que ha hecho en la vida por mí, por tu amor, confianza, dedicación, apoyo, por ser guía ejemplar y constante, por ser una madre excepcional y sin ser perfecta me has guiado por el buen camino. Te quiero mucho.

A mi papá, por ser el mejor papá del mundo, por brindarme siempre su amor a cambio de nada, por confiar tanto en mí y estar seguro que no lo defraudaría, a pesar de que en muchos momentos difíciles no estabas por circunstancias de la vida no te cambiaría por ningún otro y tranquilo que el inglés es chuchería.

A mi novia, por estar al lado mío estos últimos meses apoyándome, por regañarme y darme fuerzas cuando no tenía ganas de hacer nada, por darme ese amor infinito que alegra cada día de mi vida, por todos los momentos lindos que hemos vivido y lo que faltan por vivir.

A mi familia, por esperar siempre lo mejor de mí, porque cada uno de ellos puso un granito de arena para que pudiera llegar a ser quien soy, en especial a mi hermano Alain que espero que esté orgulloso de su hermano el culo como él dice, a la persona que hoy hace de mi mamá una mujer sumamente feliz y desde que llegó a la familia se ha convertido en un padre para nosotros 'Rudy'. A mis tíos Mirna, Miguel y mi primita Analay que, aunque no están aquí presente hoy siempre me apoyaron para seguir adelante desde el primer que entre a la UCI, a Santiago y Dalia por ser cómplices de este gran sueño, por ser los padres diarios, siempre buscando una solución a todos los problemas que se presentaban y alimentándome con sus ricas croquetas. Yairen y Armandito que se han hecho cargo de mi casa día a día cuando vengo para la UCI y me paso hasta un mes sin ir a la casa gracias a ustedes también. A la que para mí es una familia perfecta Dilema y Ricardo por aceptarme y acogerme en su casa como un hijo más, a esa chica que cuando la conocí me calló súper mal, pero hoy se ha convertido en una persona muy especial y espero un día estar

sentado, así como esta ella aquí ahora viéndola cumplir su sueño Rachel. A todos lo que por un motivo u otro no han podido estar aquí, pero sé que están al tanto de lo que sucede.

Mis queridos tutores Flavio Enrique y Clara Elena, por ser cómplices de este gran esfuerzo, por el apoyo brindado, por ofrecerme sus conocimientos y ayudarme a estar hoy aquí, principalmente a Flavio que nos dio su voto de confianza cuando nos quedamos sin tema de tesis y nos propuso esta, sin importar el poco time que quedaba, dándonos ánimo y siempre estando presente ahí para poder llegar al final, por sus sacrificios en algún tiempo incomprendido, por sus ejemplos de superación incasable, por su comprensión, confianza y amistad incondicional, porque sin su apoyo no hubiera sido posible la culminación de nuestra carrera profesional y sobre todas las cosas por su esmerada dedicación de veras muchas gracias mi hermano. A los profesores del jurado siempre ayudando con sus criterios positivos o negativos para que la tesis saliera lo mejor posible, principalmente la profe Irina con sus 150 comentarios cada vez que le mandaba el documento para ser revisado.

A todas aquellas personas que no dudaron ni un minuto en dejar lo que estuviesen haciendo para prestarme su ayuda y su atención, para aclararme una duda relacionada con la tesis y en ese sentido merece un agradecimiento especial para Alex y Yosbel que lo mismo a las 7 de la mañana que a la 10 de la noche cuando los molestaba me atendían sin problema ninguno. A Elibetsy que me prestaba a Yosbel sin peros a cualquier hora, hasta comida me hizo un día. A la profe Glemnis que con sus críticas educativas nos daba siempre una mejor forma de ver las cosas en todos los dichosos cortes, a todos muchas gracias por todo su apoyo.

A todos mis compañeros que estuvieron conmigo durante estos 6 años de carrera en los que pasamos buenos y malos momentos, pero siempre juntos y contentos, en especial quiero mencionar a los viejos amigos Carlos Chao, Randy, Leonel, Sergio, Karel, la baca Dunia que no pudo estar aquí hoy ya que está esperando a dar a luz a su bebé, la gente del piquete entre otros que hoy no se encuentras aquí, pero son mi gente del primer día en la UCI… a mis compañeras de apartamento por ser mi pequeña familia aquí en la universidad, a todos muchas gracias por soportarme y por cuidar siempre de mí, principalmente a Julio al que siempre estaba jadiando para cualquier cosa lo mismo para arreglar algo de la PC que de comida, también a la canchanfla del apta Randy que con sus consejos fashion siempre estaba criticando o elogiando la vestimenta de todos y ayudando en lo que podía. A mis compañeros de aula en general Yanitza, Yanara, la chica que más jodiamos en el aula Dayi, Angélica, los Juan Plablos, Elian al que siempre jodiamos con las fotos cómicas, Rainer, Ronald, Antonio a todos muchas gracias por formar parte de este sueño y todo su apoyo.

A los que hoy están aquí presente . . . A la mejor tía de apto que tiene la uci, Migdalia la que lo mismo se te metía al baño cuando uno se bañaba para hacer alguna broma que verías y te la encontrabas durmiendo en tu cama o limpiándote el cuarto, a esa que sin más se ha comportado como una madre para nosotros mis más grandes y sinceros agradecimientos.

Por último no puedo dejar de mencionar al tin Roy, a los borrachos del 107, a la mejor familia que me ha dado la vida en estos años de universidad, al correctos Lijandy que con su ideología de jefe nos mantenía a la raya, el loko más loko de todos al lokiño que, en 3 años que conviví con él, no lo vi con chor ni pantalón nunca en la sala, al master chef del apartamento leito que todos los días discutíamos por cualquier cosa y terminaba con la frase "fájate", a los Carlos uno por preparar el mejor café del mundo en las largas noches de tesis y el otro por su palabras diarias "a qué hora se come en este apto". Al gordo del piquete Leonardo que claro siendo el más inteligente era el que ayudaba a todos en las pruebas, el calvo Henry el que más tiempo pasaba frente al espejo peinándose, no sé qué, porque pelo no tiene, agradecer también al que más le temíamos en la uci no por su tamaño sino por su fuerza de cara el cuzo, a la dama de la casa la que no hacía más nada que mandar a buscar helado a todos menos a su príncipe azul, Mónica la antojada de la familia, a su príncipe azul al que más le gustaban que le caigan atrás no tanto las mujeres, si no los hombre, el modelo del apto, mi hermano Migue, el promotor de todas la borracheras, el hijo que acogí desde primer año, el negro más feo de todo los negros el yonki compañero de cuarto este en la lista del apto o no, y no por ser último menos importante, el que más dolores de cabeza me dio en la UCI, yo decía que las mujeres daban dolor de cabeza, donde llegó ese frenó y paró mi dúo de tesis, pero el mejor de todos sin duda alguna. A todos muchas gracias por formar parte de mi vida.

En general, agradezco a todos aquellos que aportaron su granito de arena para que un día como hoy, pudiera dedicarle estas palabras. A todos ustedes, Gracias, Muchas Gracias.

De Michel:

Primeramente, agradecer la UCI por ser mi hogar, por darme la posibilidad de crecerme como persona, por educarme, por darme tantos amigos.
Al tribunal por formar parte de este momento tas especial, especialmente a la profe Irina que es la que más ha vivido nuestra trayectoria por la universidad.
A mis tutores Flavio y Clara Elena por asesorarme y por su ayuda incondicional.
Al oponente Pacheco por contribuir también en este momento.
A los profes por contribuir con mi formación profesional.

A mi compañero de tesis Alejandro por recorrer esta larga trayectoria conmigo y por convertirse en familia en todos estos años.
A todos mis compañeros de aula que me hicieron sonreír y aportaron granos de felicidad a mi estancia.
Ronald, Day, Elian, Henry, los Juan Pablos, Rainer, Antonio, Julio, Randy, Dayana, Lionel, Dunia.
Mis dos Yanis, Yanara y Yanitza por ser más que mis compañeras de bailes en las fiestas se convirtieron en grandes amigas.
A todo ese piquetazo del 6504, los quiero a todos de verdad.
A los tres villaclareños más locos que tiene la UCI.
Lijandy con su frase célebre "Lo mío es batalla de ideas".
Leosdany por creer que es el que mejor que baila del pikete cuando sabe que soy yo.
Miguel Ángel que con sus locuras le arreglaba un día amargo a cualquiera.
A los Carlos, Cuso, Leonardo, Lokiño, Eduardo, Mayi, Indira.
Agradecerle también a alguien que se ha convertido en mi hermanita chiquita, Rachel y al piquetazo de la FICI Iris y Yessica.
Agradecerle a mi hermano del alma el chino y a su familia que su casa ha sido mi segundo hogar, a todo ese pikete de la facultad 1
Justiz, Adrian, Jojhany, Daniel, Susana, mi hermana mayor Wilbia y su mama y a la gente del solar que siempre andan con ella.
A todos los que me ayudaron de una u otra forma, gracias.
A mi Madre y mi Padre por su apoyo y sacrificio, estar siempre a mi lado en todos los momentos de mi vida, por hacer de mí lo que soy hoy,
por ese amor desmedido que me han dado.
A mi hermana Maidelis por estar a mi lado todo este tiempo, sé que está orgullosa de mi y también sé que le di muchos dolores de cabeza.
A mi cuñado Omar por formar parte de esta familia maravillosa.
Agradecerle a lo más lindo que me ha pasado en esta escuela mi sobrinita Adriana, que todos saben aquí como se ha comportado el tío con ella, darle las gracias por haber nacido en un momento como este.
A mi abuela y a mi tío que sé que vivieron cada trayectoria de mi paso por la UCI por todos sus consejos y ayuda en todo el año de mi vida. Los quiero mucho.
En fin, gracias a todos los presentes y a los que me han ayudado a formarme tanto en lo profesional como en lo personal en este largo camino.

Resumen

La gestión de la información es el proceso mediante el cual se logra tener el conocimiento necesario para la toma de decisiones en el momento oportuno, por lo tanto, es considerada uno de los pilares principales de una organización. Muchas entidades disponen de una gran cantidad de información, la cual puede ser gestionada a través de reportes. Este proceso se realiza con herramientas complementarias de los sistemas de información que utilizan un lenguaje transparente al usuario, capaces de realizar consultas a la base de datos para obtener información en forma de reportes. Para el análisis y diseño de reportes en el Sistema Integrado de Gestión Estadística (SIGE) perteneciente al Centro de Tecnología y Gestión de Datos (DATEC), se utiliza el Generador dinámico de reportes (GDR) en su versión 1.8. Este sistema requiere de un componente que facilite el trabajo a la hora de poder reutilizar e incorporar la estructura de los reportes contribuyendo a elevar la productividad.

El objetivo del presente trabajo es desarrollar un componente que permita exportar/importar estructuras de reporte en el Generador dinámico de reporte en su versión 1.8. Como resultado se obtuvo un componente que permite realizar el trabajo de forma más rápida, más eficiente y al mismo tiempo incrementar el número de clientes con el que cuenta SIGE y un correcto funcionamiento luego de haber realizado las pruebas necesarias para comprobar el sistema desarrollado.

Palabras claves: Centro de Tecnología y Gestión de Datos (DATEC), Generador dinámico de reportes (GDR), Sistema Integrado de Gestión Estadística (SIGE).

Abstract

The information management is the process by which manages to have the necessary knowledge for making decisions at the right time, therefore, is considered one of the main pillars of an organization. Many entities have a lot of information, which can be managed through reports. This process is performed with complementary tools of information systems using transparent to the user, able to query the database for information in the form of reports language. For the analysis and design of reports on of the Center for Technology and Data Management (DATEC) Statistics Integrated Management System (SIGE), Dynamic Report Generator (GDR) is used in version 1.8. This system requires a component that facilitates work when you can reuse and incorporate the structure of the reports helping to raise productivity.

The aim of this work is to develop a component that allows export / import reporting structures in the dynamic report generator in version 1.8. As a result, a component that allows work more efficient faster and at the same time increasing the number of customers that account SIGE and proper operation after performing the necessary tests to verify the developed system was obtained.

Keywords: *Center for Technology and Data Management (DATEC), Dynamic Report Generator (DRG), Statistics Integrated Management System (SIGE).*

Índice

Índice Figuras

Índice de tabla

Introducción

Hoy en día la información es considerada un elemento sumamente valioso, de no cuidarse puede encontrarse dañada o perdida en el momento que se necesita consultar. La gestión de la información es un proceso fundamental para cualquier empresa, sobre todo a la hora de tomar decisiones, a raíz de lo planteado anteriormente cobra mayor importancia el proceso de obtener la información correcta, en la forma adecuada, por la persona indicada, al mínimo costo posible, en el momento oportuno y en el lugar indicado para tomar una decisión acertada.

Al incrementarse diariamente el volumen de información existente, en la actualidad las empresas necesitan de una alternativa que les permita consultar la información que tienen en tiempo real y de la manera más eficiente posible, por lo que ha surgido la tendencia de acceder a los datos en forma de reporte. Los reportes no son más que una fotografía en el tiempo y ayudan a las empresas a percatarse en qué lugar se encuentran de acuerdo a los objetivos que tienen planificados, estos son una guía para poder cumplir con los objetivos propuestos (Shimabukuro, 2015). Un reporte está conformado por tres partes fundamentales, el diseño que no es más que el formato con el que se desean visualizar los reportes, la fuente de datos que es el lugar de donde se van a consumir los datos para ser visualizados y el acceso a los datos mecanismo que se utilizará para consultar la información almacenada en las fuentes de datos.

Con la importancia que tiene en la actualidad la visualización de reportes para las empresas han surgido disímiles herramientas para la realización automática de reportes. Ejemplo de ellas son: Gereport, Active Report y el generador dinámico de reporte esta última implementado en la Universidad de las Ciencias Informáticas (UCI) en el Centro de Tecnologías de Gestión de Datos (DATEC).

El Sistema Integral de Gestión Estadística (SIGE) herramienta también desarrollada en la UCI específicamente en el cetro DATEC, brinda facilidades para el trabajo estadístico como son la captación de información a través de formularios matriciales y encuestas, tiene como característica que puede ser utilizado en diversas plataformas. Debido a las ventajas y las facilidades que brinda para el trabajo con los reportes, SIGE utiliza el generador dinámico de reportes como herramienta para visualización y diseñar reportes, la cual permite la generación dinámica de estos, partiendo de datos persistentes en algún origen soportado por el sistema.

Las entidades cubanas no están ajenas a las necesidades de conservar en un orden adecuado la información que manejan, requiriendo de la generación de reportes en tiempo real, para poder realizar los análisis pertinentes en el momento que sea necesario. En la actualidad varias instituciones utilizan para la gestión de la información SIGE, entre las cuales se pueden destacar la Oficina Nacional de Estadística e

Información (ONEI), Ministerio de Industria (MINDUS), Fiscalía General de la República (FGR), El Tribunal Supremo Popular(TSP) entre otras.

Debido al impacto que ha tenido SIGE en las empresas cubanas, y la gran aceptación que ha tenido en las entidades del país. En la actualidad el número de clientes que solicitan la herramienta ha aumentado de forma exponencial. Estos clientes tienen como característica en común, que la mayoría de las funcionalidades que solicitan al equipo de desarrollo están relacionadas con la implementación de nuevos reportes que le ayuden a la hora de tomar decisiones.

El generador dinámico de reporte en su versión 1.8; herramienta utilizada por SIGE para el diseño y la visualización de los reportes, no permite la reutilización de los realizados en otras aplicaciones, lo que implica que cada vez que aparece un nuevo cliente que solicite SIGE al centro DATEC, haya que volver realizar los reportes desde cero, aunque en la mayoría de las ocasiones muchos de los reportes ya existen o han sido implementados anteriormente, pudiendo servir como base o punto de partida para desarrollar un nuevo reporte, lo cual permitiría realizar el trabajo de forma más rápida, más eficiente y al mismo tiempo incrementar el número de clientes con el que cuenta SIGE en la actualidad. Debido a que en la actualidad el equipo de desarrollo con el que cuenta el SIGE no es muy extenso, unido al elevado número de reportes que solicitan los nuevos clientes, se hace imposible aceptar la totalidad de las nuevas solicitudes que llegan al centro DATEC.

Por lo que surge como **problema de investigación:** el generador dinámico de reportes en su versión 1.8 no permite la reutilización e incorporación de estructuras de reportes.

Donde el **objeto de estudio** está dirigido al proceso de gestión de reportes en los sistemas generadores de reportes, enmarcado en el **campo de acción:** el proceso de reutilización e incorporación de estructuras de reportes en el generador dinámico de reporte en su versión 1.8.

Para dar solución al problema de investigación planteado se define como **objetivo general:** Desarrollar un componente que permita exportar e importar las estructuras de un reporte en el generador dinámico de reporte en su versión 1.8.

A partir del objetivo general se desglosan los siguientes **objetivos específicos:**

1. Analizar los fundamentos teóricos que abordan los conceptos relacionados con la gestión de reportes.
2. Realizar el análisis y diseño del Componente para exportar e importar estructuras de reportes en el generador dinámico de reporte en su versión 1.8.

3. Implementar el Componente para exportar e importar estructuras de reportes en el generador dinámico de reporte en su versión 1.8.
4. Validar el Componente para exportar e importar estructuras de reportes en el generador dinámico de reporte en su versión 1.8.

Para el desarrollo de la investigación se definieron las siguientes **tareas de investigación:**

1. Selección de la metodología, herramientas y tecnologías a utilizar en el desarrollo del Componente para exportar e importar estructuras de reportes en el generador dinámico de reportes en su versión 1.8.
2. Análisis y diseño de las funcionalidades a incluir en el Componente para exportar e importar estructuras de reportes en el generador dinámico de reportes en su versión 1.8.
3. Descripción de la estructura base del Componente para exportar e importar estructuras de reportes en el Generador dinámico de reportes en su versión 1.8.
4. Selección de los patrones de diseño a emplear en el desarrollo del Componente para exportar e importar estructuras de reportes en el generador dinámico de reportes en su versión 1.8.
5. Implementación del Componente para exportar e importar estructuras de reportes en el Generador dinámico de reportes en su versión 1.8.
6. Realización de las pruebas de software para comprobar el correcto funcionamiento del Componente para exportar e importar estructuras de reportes en el generador dinámico de reportes en su versión 1.8.

Para el cumplimiento del objetivo general planteado se utilizarán los siguientes **métodos de investigación:**

Métodos teóricos

- Analítico-Sintético: Se utilizó para extraer los elementos fundamentales de las bibliografías consultadas con fines de caracterizar los procesos y soluciones relacionados con el diseño de los reportes, particularmente en el proceso de la exportación e importación de los mismos, permitió realizar el análisis de soluciones similares existentes que ayude al proceso investigativo para la solución planteada.
- Histórico-Lógico: La investigación se realizó a partir del estudio del estado del arte de la problemática analizada, permitiendo conocer la trayectoria y desarrollo de aplicaciones relacionados al trabajo con los diseños reportes, principalmente con el proceso de la exportación e importación de los mismos, lo que dio la posibilidad de seleccionar aquellas herramientas, lenguajes y metodologías lógicas y adecuadas correspondientes para el desarrollo de la aplicación.

Métodos empíricos

1. Análisis Estático: Permitió observar la estructura de generador dinámico de reportes, con el objetivo de clasificar las dificultades existentes y las funcionalidades que debe tener el sistema a implementar.
2. La técnica de entrevista: Se realiza a los especialistas, con el objetivo de obtener información actualizada de los procesos que están informatizados y de las nuevas funcionalidades que se desean incorporar y mejorar; así como para definir los requisitos funcionales y no funcionales del mismo. Se utilizó la entrevista no estructurada debido a que no se basa en un cuestionario rígido y es aplicado a personas que son especialistas en el tema del cual se está preguntando.

El Trabajo de Diploma está estructurado de la siguiente manera: Introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas, bibliografía y anexos.

Capítulo 1: Fundamentación Teórica.
En este capítulo se presenta la definición del marco teórico de la investigación. Se exponen los temas referentes a los sistemas estadísticos existentes en el mundo y en Cuba. Además, se aborda el estudio de la metodología y herramientas que serán utilizadas para el desarrollo del componente.

Capítulo 2: Análisis y Diseño.
En este capítulo se describe de forma detallada cómo deberá funcionar el componente a desarrollar y las especificaciones de su implementación.

Capítulo3: Implementación y Pruebas de la solución.
Comienza con el resultado del diseño, implementación del 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

En este capítulo se describen los principales conceptos relacionados con las herramientas de generación de reportes. Se exponen características de algunos sistemas relacionados con exportar e importar archivos, abordándose a su vez la metodología, herramientas y tecnologías a utilizar en el desarrollo de la solución propuesta.

1.1 Conceptos asociados al dominio del problema

Datos: se define como datos a los hechos que describen sucesos y entidades. Los datos son símbolos que describen condiciones, hechos y situaciones. Se caracterizan por no contener ninguna información. Un dato puede significar un número, una letra, un signo ortográfico o cualquier símbolo que represente una cantidad, una medida, una palabra o una descripción. La importancia de los datos está en su capacidad de asociarse dentro de un contexto para convertirse en información. Por si mismos los datos no tienen capacidad de comunicar un significado y por tanto no pueden afectar el comportamiento de quien los recibe. Para ser útiles, los datos deben convertirse en información para ofrecer un significado, conocimiento, ideas o conclusiones. Los datos pueden consistir en números y estadísticas (D'Ambrosio, 2010).

Estadística: la estadística, en general, es la ciencia que trata de la recopilación, organización, presentación, análisis e interpretación de datos numéricos con el fin de realizar una toma de decisiones más efectiva. Una de las formas para gestionar los datos estadísticos es a través de la generación de reportes (Muñoz, 2008).

Reportes: se denomina reporte a un cuerpo de información destinado a servir de análisis sobre un tópico determinado. Un reporte puede revestir diversas formas, ya sea como escrito, como charla, como informe televisivo o como una película documental. Además, contiene una serie de rutinas asociadas para la generación del mismo (Shimabukuro, 2015). Un reporte está conformado por tres partes fundamentales, el diseño que no es más que el formato con el que se desean visualizar los reportes, la fuente de datos que contienen los datos para ser visualizados y el acceso a los datos mecanismo que se utilizará para consultar la información almacenada en dicha fuente de dato.

Rutinas: serie de instrucciones para un ordenador que le permite llevar a cabo una tarea. Los programadores pueden utilizar una serie de rutinas ya escritas y probadas a la hora de realizar sus programas, estas son una secuencia invariable de instrucciones que puede utilizarse una y varias veces. En este sentido, la rutina se presenta como un sub algoritmo dentro del algoritmo principal (el programa), en general, se utilizan rutinas para mejorar el rendimiento global del sistema de gestión de bases de datos,

permitiendo que las funciones de las aplicaciones se realicen en el servidor de bases de datos. Las rutinas también son conocidas como funciones o subrutinas (Diclib, 2015).

Función: teniendo una base de datos con información útil para la empresa, es importante saber cómo extraerla y hacer uso de ella. Uno de los mecanismos que se utiliza para esto son las funciones. Las mismas son objetos de la base de datos que están programados con un lenguaje procedural específico y que pueden facilitar la administración de la base de datos y la visualización de la información contenida en ella. Además, son un grupo de instrucciones con un objetivo en particular y que se ejecuta al ser llamada desde otra función o procedimiento. Una función puede llamarse múltiples veces e incluso llamarse a sí misma. Se compilan en el Sistema Gestor de Base de Datos(SGBD) cuando se crean, por tanto, se ejecutan con mayor rapidez que las instrucciones SQL individuales. En la implementación de una función se puede hacer uso de consultas y vistas, dos mecanismos de obtención de información que además pueden utilizarse individualmente. A continuación, se hace una breve descripción de cada uno (Alegsa, 2010).

Consulta: las consultas tienen como objetivo recuperar datos específicos de las tablas, los cuales suelen estar dispersos y a través de ellas, se pueden ver en una sola hoja de datos. Este mecanismo de obtención de información permite agregar criterios para filtrar los datos hasta obtener solo los registros que se deseen. Las consultas pueden ser de lectura o escritura. Las de lectura (SELECT), simplemente recuperan los datos y hacen que estén disponibles para su uso. Las de escritura (INSERT, UPDATE, DELETE), pueden servir para crear tablas nuevas, agregar datos a tablas existentes y actualizar o eliminar datos (Alegsa, 2010)

Vistas: la vista es una tabla virtual resultante de una consulta SQL en la cual se cargan los datos en el momento de ser llamada, sólo se almacena de ella la definición. A través de su implementación se pueden obtener datos de una o varias tablas. Presenta desventajas en cuanto a rendimiento, porque el SGBD debe traducir las consultas elaboradas a la vista en consultas realizadas a las tablas fuentes, por lo que pueden tardar mucho tiempo en completarse. Estas se crean cuando se necesitan hacer varias sentencias para devolver una tabla final. En resumen, una vista es una tabla virtual derivada de las tablas reales de una base de datos (Computer Systems Research , 2012).

1.2 Sistemas informáticos existentes

Active Report: es una herramienta de plataforma propietaria que se caracteriza por soportar gráficos de tipo 2D y 3D, imágenes y sub-reportes. Es capaz de emplear una amplia variedad de orígenes de datos, además cuenta con un lenguaje de programación propio. Permite la adición de nuevos reportes sin

necesidad de recompilar la aplicación, dando soporte tanto a aplicaciones web como de escritorio y exporta los reportes en diferentes formatos como: PDF, XML y HTML (Olivares, 2008)

Sistema de gestión de reporte dinámico (Gereport): es al diseño, generación y configuración de los reportes relacionados con los datos históricos almacenados en una fuente de datos. Además, luego de contar con toda la información del reporte es posible exportarlo como imagen y en los formatos HTML, PDF y Excel. Para la interacción con las aplicaciones externas, el sistema implementa un servicio que expone los metadatos de los reportes para poder utilizarlos sin restricciones de lenguajes y plataformas. (RCCI, 2014)

Estos sistemas contienen las funcionalidades necesarias para dar solución al problema planteado, por motivos de ser privativos y de no ser integrables con SIGE solo fueron analizados y estudiados aportando el conocimiento necesario para determinar cuáles son los formatos adecuados para utilizar en el desarrollo del componente.

Generador dinámico de reporte (GDR): es una aplicación desarrollada sobre el framework Symfony y escrita en PHP. Es multiplataforma. Soporta imágenes, gráficas, así como varios orígenes de datos. Proporciona a los usuarios, entre otras opciones, agilizar la toma de decisiones, generar reportes y con gran variedad de opciones en su diseño, marcando una diferencia entre los reportes tradicionales y los reportes dinámicos, objetos de este producto. Este está compuesto por varias aplicaciones entre las que se encuentran el Visor de reportes, Diseñador de modelos y el Diseñador de reporte (Valle Laborde, et al., 2011).

Actualmente, para diseñar reportes en el GDR en su versión 1.8 el usuario debe realizar tres procesos fundamentales, el diseño, la fuente de datos y la manera de consumir estos datos. Para poder obtener la información que se desean visualizar se deben implementar disimiles rutinas. Las rutinas están compuestas por funciones y vistas, este proceso se realiza a nivel de base de datos, lo cual conlleva a que se emplee más tiempo en el diseño y realización de las mismas.

Se decide utilizar el GDR para el desarrollo del componente ya que esta embebido dentro de SIGE y es el utilizado para visualizar y diseñar reportes.

1.3 Formato del archivo a exportar

Formato XML: Es una tecnología diseñada para administrar y compartir datos estructurados en un archivo de texto legible para el usuario. XML sigue directrices estándares de sector y puede ser procesado por una amplia gama de bases de datos y aplicaciones. Representar información estructurada en la web, de modo

que esta información pueda ser almacenada, transmitida, procesada, visualizada e impresa, por diversos tipos de aplicaciones y dispositivos. El uso de XML permite a los diseñadores de aplicaciones crear sus propias etiquetas, estructuras de datos y esquemas personalizados. En resumen, XML facilita considerablemente la definición, transmisión, validación e interpretación de datos entre bases de datos, aplicaciones y organizaciones (C/ Albasanz, 14 Bis).

Con el estudio realizado se decidió escoger el formato XML ya que posee una estructura más abierta y extensible. Los identificadores pueden crearse de manera más simple y pueden ser adaptados tanto para la Web como para una intranet por medio de un procesador de documentos (*parser*). También se pueden gestionar desde el propio cliente Web, algo que otros formatos es difícil realizar como es el caso de HTML. Otra de las facilidades que aporta XML es que puede acceder a los datos de manera más precisa, ya que como la información está mucho más estructurada facilita enormemente la labor a los buscadores. Este separa radicalmente el contenido y el formato de presentación. En resumen, XML es el lenguaje más apropiado, además de ser el formato estándar que utiliza SIGE para exportar archivos utilizando la librería php_XML operando con Simplexml extensión de php para obtener los datos, los cuales se obtiene al instalar php5.

1.4 Metodología y herramientas

Es indispensable disponer de metodologías y herramientas que permitan valorar las mejoras introducidas y comparar el “estado de situación” antes y después de la introducción de los cambios (Foundation, ITIL®, 2015). Para desarrollar el componente se utilizarán las tecnologías y herramientas definidas en el GDR en su versión 1.8 dando esto grandes facilidades a la hora de integrar el componente.

Metodología de desarrollo de software.

Una metodología de desarrollo de software define un conjunto de filosofías, fases, procedimientos, reglas, técnicas, herramientas, documentación, y aspectos de formación para los desarrolladores de sistemas de información. Por tanto, una metodología de desarrollo de software es un conjunto de componentes que especifican (Aguila, 2013).

- Cómo se debe dividir un proyecto en etapas.
- Qué tareas se llevan a cabo en cada etapa.
- Qué salidas se producen y cuando se deben producir.
- Qué restricciones se aplican.
- Qué herramientas se van a utilizar.

- Cómo se gestiona y controla un proyecto.

Proceso Unificado Abierto u OpenUP (*Open Unified Process*)

Es un proceso modelo y extensible, dirigido a gestión y desarrollo de proyectos de software basados en desarrollo iterativo, ágil e incremental apropiado para proyectos pequeños y de bajos recursos; y es aplicable a un conjunto amplio de plataformas y aplicaciones de desarrollo (Aguila, 2013).

Se divide en cuatro fases:

1. Concepción: Primera de las 4 fases en el proyecto del ciclo de vida, acerca del entendimiento del propósito y objetivos y obteniendo suficiente información para confirmar que el proyecto debe hacer. El objetivo de ésta fase es capturar las necesidades de los *stakeholder* en los objetivos del ciclo de vida para el proyecto.
2. Elaboración: Es la segunda de las 4 fases del ciclo de vida del OpenUP donde se trata los riesgos significativos para la arquitectura. El propósito de esta fase es establecer la base de la arquitectura del sistema.
3. Construcción: Esta fase está enfocada al diseño, implementación y prueba de las funcionalidades para desarrollar un sistema completo. El propósito de esta fase es completar el desarrollo del sistema basado en la Arquitectura definida.
4. Transición: Es la última fase, cuyo propósito es asegurar que el sistema es entregado a los usuarios, y evalúa la funcionalidad y performance del último entregable de la fase de construcción.

Propone 6 flujos de trabajo:

1. Requerimientos: Se realizan entrevistas con el cliente para comprender el problema a resolver y se definen los requerimientos.
2. Análisis y Diseño: Se realiza el diseño de los requisitos que serán después implementados.
3. Implementación: Se realiza la implementación del sistema basándose en el diseño realizado.
4. Prueba: Busca los defectos a lo largo del ciclo de vida.
5. Gestión del proyecto: Involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.
6. Gestión de configuración y cambios: Describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto en cuanto a: utilización y actualización, control de versiones, etcétera.

Beneficios de su uso:

1. Es apropiado para proyectos pequeños y de bajos recursos, permite disminuir las probabilidades de fracaso en los proyectos pequeños e incrementar las probabilidades de éxito.
2. Permite detectar errores tempranos a través de un ciclo iterativo.
3. Evita la elaboración de documentación, diagramas e iteraciones innecesarias requeridas en la metodología RUP.
4. Por ser una metodología ágil tiene un enfoque centrado al cliente y con iteraciones cortas.

Por sus características y beneficios OpenUP es la metodología adecuada para el desarrollo del componente debido a que es más eficiente y apropiada para proyectos pequeños de bajos recursos, lo cual permite con eficiencia disminuir las probabilidades de fracaso. También permite detectar errores tempranos a través de un ciclo iterativo, evitando la elaboración de documentos, diagramas e iteraciones innecesarias, además de ser una metodología ágil con un enfoque centrado al cliente, unido a que es la metodología utilizada por el proyecto en SIGE.

Lenguaje de modelado UML

UML 2.1: El Lenguaje de Modelado Unificado o UML (acrónimo en inglés de *Unified Modeling Lenguaje*) es un lenguaje de modelado visual, destinado a ser utilizado no solo para el modelado de procesos de negocio y similares; sino también para el análisis, diseño e implementación de sistemas basados en software (uml-diagrams.org, 2009).

UML es un lenguaje común para los analistas, arquitectos y desarrolladores de *software*, utilizado para describir, especificar, diseñar nuevos procesos de negocio (uml-diagrams.org, 2009). Además, proporciona valiosa ayuda a los profesionales visualizando, comunicando y aplicando sus diseños. A lo largo de los años se ha visto evolucionar al UML logrando convertirse en una valiosa herramienta de prestigio y que ahorra mucho tiempo, llevándola a crecer y evolucionar a su versión 2.1.

Se utiliza este lenguaje de modelado ya que podemos tener una orientación en cuanto al orden de las actividades de un equipo, ayudando esto a dirigir las tareas de los desarrolladores individuales y del equipo en conjunto, además de ofrecer criterios para el seguimiento y la medición de los productos y las actividades de un proyecto.

Herramienta CASE

Visual Paradigma 8.0: *Visual Paradigma for UML Standard Edition* (VP-UML SE) es una plataforma de modelado diseñado para apoyar a los arquitectos de sistemas, desarrolladores y diseñadores de UML.

Además, acelera el proceso de análisis y diseño de aplicaciones empresariales complejas que facilita la visualización UML en diferentes tipos de diagramas, así como diagramas entidad relación, diagramas de requisitos y el análisis textual (Softwaresea, 2014).

Las ventajas que proporciona Visual Paradimg for UML son:

- Facilita el modelado de UML, ya que proporciona herramientas específicas para ello.
- Esto también permite la estandarización de la documentación, ya que la misma se ajusta al estándar soportado por la herramienta.
- Controla que el modelado con UML sea correcto.
- Al disponer de un repositorio común, es posible visualizar el mismo elemento en varios diagramas, evitando duplicidades.
- Permite integrar se con otras aplicaciones, como herramientas ofimáticas, lo cual aumenta la productividad.
- Permite el trabajo en grupo, proporcionando herramientas de compartición de trabajo.
- Facilita la reutilización, ya que disponemos de una herramienta centralizada donde se encuentran los modelos utilizados para otros proyectos.
- Permite generar código de forma automática, reduciendo los tiempos de desarrollo y evitando errores en la codificación del software.
- Permite generar diversos informes a partir de la información introducida en la herramienta.

Se escoge la herramienta Visual Paradigma ya que soportar el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas. Además, de ser utilizada por SIGE en su implementación.

Lenguajes de programación

PHP 5: PHP, acrónimo de "PHP: *Hypertext Preprocessor*", es un lenguaje de 'scripting'[1] de propósito general y de código abierto que está especialmente pensado para el desarrollo web y que puede ser embebido en páginas HTML. Su sintaxis recurre a C, Java y Perl, siendo así sencillo de aprender. La meta principal de este lenguaje es permitir a los desarrolladores web escribir dinámica y rápidamente páginas web generadas (Achour, 2011).

[1] Tipo de lenguaje de programación que es generalmente interpretado.

JavaScript: JavaScript es un lenguaje de script multiplataforma orientado a objetos. Es un lenguaje pequeño y ligero; no es útil como un lenguaje independiente, más bien está diseñado para una fácil incrustación en otros productos y aplicaciones, tales como los navegadores Web. Dentro de un entorno anfitrión, JavaScript puede ser conectado a los objetos de su entorno para proveer un control programable sobre éstos (Mozilla Developer Network, 2014).

Para la implementación del componente se escoge el lenguaje PHP5. El mismo es un lenguaje simple para el desarrollo de aplicaciones web, pero que a su vez ofrece características avanzadas para programadores expertos. Por otra parte, el lenguaje JavaScript se selecciona debido a su adaptabilidad y uso, tanto del lado del cliente como del servidor.

Marco de trabajo

ExtJS 3.4: ExtJS es una librería *JavaScript* que permite construir aplicaciones complejas en Internet. Esta librería incluye componentes de Interfaz de Usuario (UI) del alto performance y personalizables, modelo de componentes extensibles, una Interfaz de Programación de Aplicaciones (API) fácil de usar y licencias *Open Source* y comerciales. Una de las grandes ventajas de utilizar ExtJS es que permite crear aplicaciones complejas utilizando componentes predefinidos, así como un manejador de *layouts* (composición) similar al que provee *Java Swing*, gracias a esto provee una experiencia consistente sobre cualquier navegador, evitando el tedioso problema de validar que el código escrito funcione bien en cada uno (Firefox, IE, Safari, 2011).

Symfony 1.1.7: *Symfony* es un marco de trabajo de desarrollo web para PHP. Permite a los desarrolladores construir y mantener más fáciles sitios web con PHP (Github, 2014). Está basado en la arquitectura Modelo-Vista-Controlador (MVC), surgió como un proyecto de software libre el cual se ha convertido en uno de los *framework* más populares de PHP que existe en la actualidad. Se utiliza por defecto al Mapeo Objeto-Relacional (ORM) Propel que se encarga por defecto de gestionar el modelo. En las aplicaciones *Symfony*, el acceso y la modificación de los datos almacenados en las bases de datos se realiza mediante objetos; de esta forma nunca se accede de forma explícita a las bases de datos (Fabien Potencier, 2008).

El uso de *Symfony* garantiza rapidez y confiabilidad a la hora de la creación de productos usando PHP. Es compatible con la mayoría de los gestores de bases de datos, como: PostgreSQL, Oracle, SQL Server. Además, las aplicaciones desarrolladas con él pueden ejecutarse en disímiles plataformas, como son Windows y sistemas Unix/Linux.

Como marcos de trabajo se utiliza ExtJS 3.4 debido a que posee una base de componentes amplia que permite la reutilización y extensión de los mismos, así como también una vasta documentación y comunidad.

Por otra parte, se utiliza *Symfony* 1.1.7 debido a su destacado uso del patrón arquitectónico Modelo-Vista-Controlador que facilita la separación de estos elementos.

Entorno de desarrollo integrado

NetBeans 8.0: NetBeans es un proyecto de código abierto con una gran base de usuarios y una comunidad en constante crecimiento. NetBeans IDE es una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito para Java, pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el NetBeans IDE. NetBeans IDE es un producto libre y gratuito sin restricciones de uso (Oracle Corporation, 2011).

Este IDE es idóneo para el desarrollo del componente ya que se usa fundamentalmente por ser multilenguaje, pero además por ser de código abierto y gratuito.

Sistema gestor de base de datos.

PostgreSQL 9.1: PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD[2] y con su código fuente disponible libremente. PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multi-hilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando (Martinez, 2010).

PgAdmin III: PgAdmin cuenta con una rica administración en código abierto y desarrollo de plataforma para PostgreSQL. La aplicación puede utilizarse en sistemas operativos como Linux, FreeBSD, Solaris, Mac OSX y las plataformas de Windows para administrar PostgreSQL, así como las versiones comerciales y derivados de PostgreSQL como Postgres Plus Advanced Server y bases de datos Greenplum.

Para la implementación del componente se escoge como gestor de bases de datos PostgreSQL debido a que admite un gran ilimitado de bases de datos. Mientras que la herramienta PgAdmin permite el trabajo con las bases de datos de forma sencilla, brindando facilidades en la escritura de consultas y el manejo de las bases de datos.

Servidor de aplicaciones

Apache 2.2: Apache, también conocido como Apache HTTP Server, es una norma establecida en la distribución en línea de servicios de sitios web. Se trata de una plataforma de servidor web de código abierto, lo que garantiza la disponibilidad en línea de la mayoría de los sitios web activos hoy día. El servidor está dirigido a servir a una gran cantidad de plataformas web que trabajan sobre sistemas operativos como Unix,

[2] Licencia de software otorgada principalmente para los sistemas BSD (Berkeley Software Distribution)

Windows, Linux, Solaris, Novell NetWare, FreeBSD, Mac OS X, Microsoft Windows, OS / 2. (Hosting, Module for, 2013).

Se decide Apache por ser un servidor web confiable que ofrece nuevos y más flexibles módulos de autenticación de usuario, además de su configurabilidad y robustez, es casi universal al estar en una multitud de Sistemas Operativos. Trabaja con gran cantidad de lenguajes, PHP y otros lenguajes de script, Java y páginas JSP, teniendo todo el soporte que se necesita para tener páginas dinámicas, además de ser el utilizado en SIGE para su realización.

1.5 Conclusiones del capítulo:

El análisis de algunas herramientas existentes en la actualidad relacionadas con exportar a importar datos, permitió establecer los principales elementos a tener en cuenta para la realización del componente que se desea desarrollar. Además, de ayudar a definir el formato más adecuado a escoger, para realizar la exportación de los reportes. El estudio de las herramientas y metodología permitió seleccionar UML 2.1 como lenguaje de modelado y Visual Paradigma 8.0 como herramienta CASE. Como lenguajes de programación PHP 5 y JavaScript; Symfony 1.1.7 y ExtJS 3.4 como *framework* de desarrollo; NetBeans 8.0 como IDE de desarrollo. Para el tratamiento de las bases de datos se identificaron PostgreSQL 9.1 y PgAdmin III; y como servidor de aplicaciones Apache 2.2. Además, se definió XML como formato a utilizar para exportar/importar los reportes de acuerdo a las características que presenta y las facilidades que genera a la hora del desarrollo del componente y OpenUP como metodología de desarrollo de software.

Capítulo 2: Análisis y Diseño

En el presente capítulo se describe mediante el modelo de dominio el problema planteado, así como describir los principales conceptos que serán de análisis para la creación del módulo. Se presentan los requisitos funcionales y no funcionales lo cuales conforman y ayudan a elaborar el diagrama de caso de uso del sistema identificando la relación entre los actores y casos de uso. Se representan los diagramas de clases del diseño correspondientes a cada caso de uso, además de los diagramas de secuencias propios a los requisitos funcionales identificados.

2.1 Propuesta Solución

Se propone el desarrollo de un componente para exportar e importar estructuras de reportes en GDR en su versión 1.8, lo cual permitirá reutilizar e incorporar los reportes, pudiendo servir como base o punto de partida para desarrollar un nuevo reporte, lo cual permitiría realizar el trabajo de forma más rápida, más eficiente y al mismo tiempo incrementar el número de clientes con el que cuenta SIGE en la actualidad. Además, a la hora de gestionar un reporte específico permite al especialista acceder a la estructura del mismo de manera ágil y fácil. El proceso de exportación e importación de estructuras de reporte será intuitivo y fácil para los usuarios, permitiendo que los mismos se familiaricen rápidamente con el componente y aumenten su productividad al trabajar con el mismo.

2.2 Modelo de dominio

Un modelo de dominio no es más que un artefacto de la disciplina de análisis, construido con las reglas de UML durante la fase de concepción, presentado como uno o más diagramas de clases y que contiene, no solo conceptos propios de un sistema de software sino de la propia realidad física. Los modelos de dominio pueden utilizarse para capturar y expresar el entendimiento ganado en un área bajo análisis como paso previo al diseño de un sistema, ya sea de software o de otro tipo (Synergix, 2008).

Diagrama conceptual del Modelo Dominio

No están bien definidos los procesos del negocio por lo que se realiza el modelo de dominio, el cual tiene como objetivo fundamental comprender y describir los conceptos más importantes dentro del contexto del sistema, es una representación visual del entorno real del proyecto.

El modelo de dominio ocupa un rol protagónico en el desarrollo de software y además puede ser tomado como punto de partida para el diseño del sistema. El siguiente modelo de dominio expresa de manera clara en forma de concepto el estado actual del sistema.

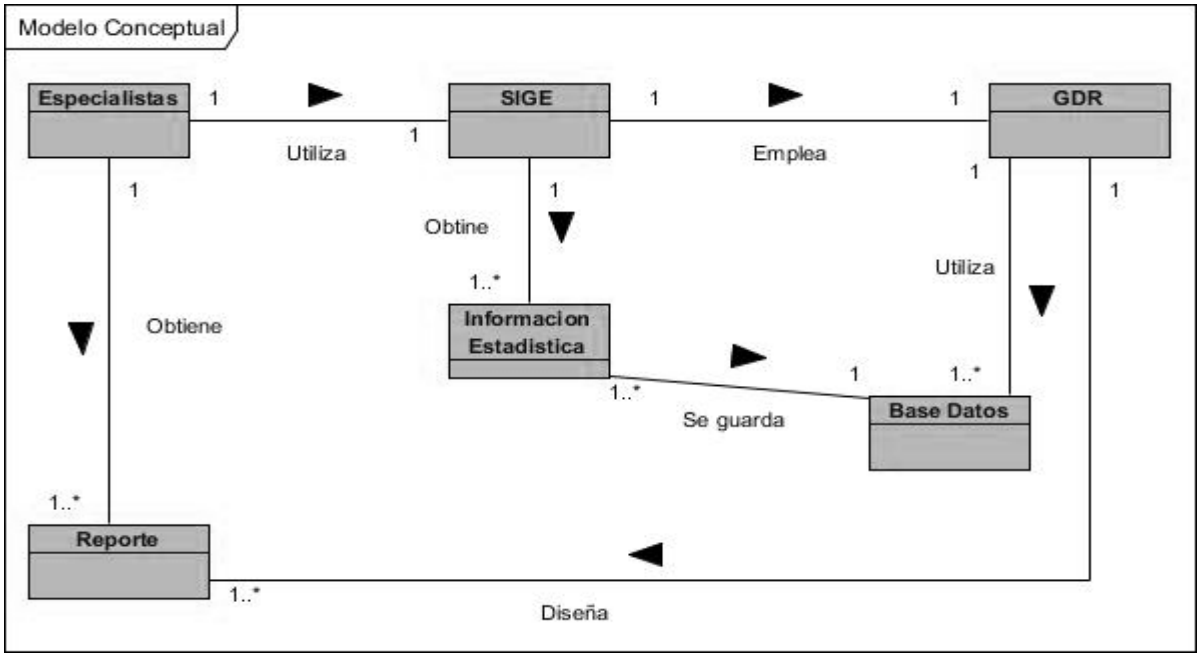


Figura 1: Modelo Dominio

Definición de clases del Modelo Dominio

Especialistas: Persona autorizada a utilizar el sistema. Varía en dependencia de los niveles en los que está estructurado SIGE. Son los encargados de utilizar SIGE para gestionar la información.

SIGE: Sistema Integrado de Gestión Estadística el cual automatiza los procesos estadísticos de la Oficina Nacional de Estadística e Información. Se apoya en GDR como herramienta para la generación de reporte.

GDR: Generador dinámico de reporte, herramienta utilizada por SIGE para gestionar reportes estadísticos. Utiliza la información estadística captada para la generar reportes.

Información Estadística: Análisis provenientes de una muestra representativa de datos, utilizada para la toma de decisiones en áreas de negocios o instituciones gubernamentales, la cual se guarda en base de datos.

Base de datos: utilizada por el sistema para almacenar la información obtenida de las instituciones externas con el objetivo de consultarla atreves del GDR para generar reporte.

Reportes: Los reportes muestran los análisis y resultados de las empresas, por lo que se consideran un importante requisito en el negocio.

2.3 Especificación de requisitos del sistema

Los requerimientos de un sistema de software son un conjunto de pautas que expresan lo que se desea realizar. Estos se clasifican de dos formas: funcionales y no funcionales, los cuales parten de la descripción de las clases representadas en el modelo de domino y las necesidades del cliente.

2.3.1 Requisitos Funcionales

Son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que éste debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares. En algunos casos, los requerimientos funcionales de los sistemas también pueden declarar explícitamente lo que el sistema no debe hacer (Can, 2010).

Se identificaron los siguientes requisitos funcionales:

RF1: Exportar Plantilla a XML
RF2: Exportar reporte a XML
RF3: Lista Plantilla
RF4: Lista reportes
RF5: Buscar Plantilla
RF6: Buscar reporte
RF7: Importar Plantilla en formato XML
RF8: Importar reporte en formato XML

Tabla. 1: Requisitos funcionales del sistema

Patrones de casos de uso utilizados

Los patrones constituyen una guía para el diseño del software. Su objetivo es la solución de problemas que ocurren repetidamente dentro de un contexto muy bien definido. Además, deben ser reusables, lo que significa que son aplicables a diferentes problemas de diseño en distintas circunstancias. Son de gran utilidad para describir las mejores prácticas, buenos diseños y encapsulan la experiencia permitiendo su reutilización. Son utilizados generalmente como plantillas que describen cómo deben ser estructurados y organizados los casos de uso, por lo que su empleo constituye una buena práctica en el modelado de casos de uso (Sommerville, 2005). En la presente investigación se empleó para el modelado de los casos de uso el siguiente patrón:

- **CRUD Parcial:** Este se evidencia en los casos de uso Administrar Plantilla y Administrar Reporte compuestos por un listar, buscar, exportar e importar respectivamente.

2.4 Modelo de Caso de uso del sistema

El modelo de casos de uso del sistema describe la funcionalidad propuesta del nuevo sistema, donde cada caso de uso representa una unidad discreta de interacción entre un usuario (humano o máquina) y el sistema. (Sparx, 2007). Este está compuesto por actores, casos de uso y la relación que existe entre ellos.

Caso de Uso: Los casos de uso son artefactos narrativos que describen, bajo la forma de acciones y reacciones, el comportamiento del sistema desde el punto de vista del usuario. Por lo tanto, establece un acuerdo entre clientes y desarrolladores sobre las condiciones y posibilidades (requisitos) que debe cumplir el sistema (Sommerville, 2005).

Actores del sistema: Un actor es una entidad externa al sistema que se modela y que puede interactuar con él. Puede ser una persona o un grupo de personas homogéneas, otro sistema, o una máquina. Los actores son externos al sistema que vamos a desarrollar. Por lo tanto, al identificarlos, estamos comenzando a delimitar el sistema y a definir su alcance. Es importante resaltar que los actores son abstracciones de papeles o roles y no necesariamente tienen una correspondencia directa con personas (Jojooa , 2014).

Actor	Descripción
Usuario	El usuario es cualquier persona autorizada a utilizar el sistema para trabajar en el módulo. Varía en dependencia de los niveles en los que está estructurado GDR.

Tabla. 2 Actores del sistema

Diagrama de caso uso del sistema

Los CU son tareas específicas que se realizan a raíz de la orden de un actor. Un diagrama de casos de uso del sistema representa gráficamente a las funciones del sistema y su interacción con los actores (Sommerville, 2005).

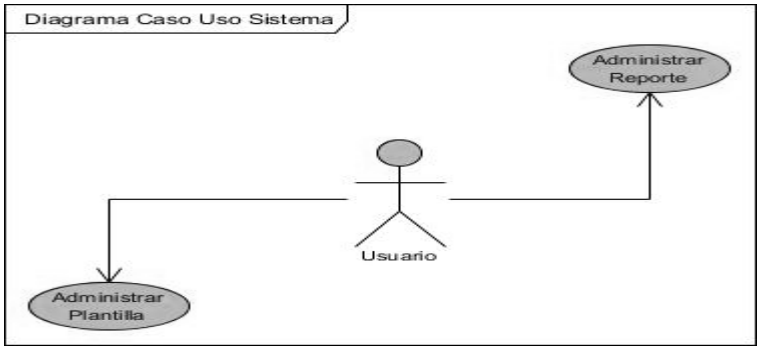


Figura.2: Diagrama caso uso del sistema.

Descripción textual de los casos de Uso del Sistema

Un caso de uso describe como el usuario finalmente interactúa con el sistema.

Tabla. 3: Descripción del Caso de Uso Administrar Plantilla

Objetivo	Administrar Plantilla
Actores	Administrador
Resumen	El caso de uso se inicia cuando el usuario selecciona la acción Plantilla.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	
Pos_condiciones	Administrar Plantilla
Flujo de eventos	
Flujo básico "Administrar Plantilla"	
Actor	**Sistema**
El caso de uso inicia cuando el actor selecciona el componente "exportar_importar".	El sistema muestra una interfaz para seleccionar "Plantilla o Reporte"
El actor selecciona la opción "Plantilla".	El sistema muestra una interfaz con las lista de las plantillas
El actor selecciona la opción que desea realizar con las plantillas como exportar, exportar, eliminar o buscar.	El sistema realiza una de las siguientes opciones: • Selecciona "Exportar": ver sección 1 "Exportar" • Selecciona Importar: ver sección 2 "Importar " • Selecciona Buscar: ver sección 3 "Buscar"
Sección 1: Exportar	
El actor selecciona Exportar luego de haber seleccionado la plantilla.	El sistema muestra una interfaz seleccionar la opción de guardar o visualizar la plantilla.
El actor selección la opción deseada y selecciona aceptar o cancelar.	El sistema realiza una de las siguientes opciones: • Selecciona Aceptar: Ver sección 4 "Aceptar" • Selecciona Cancelar: Ver sección 5 "Cancelar "

Sección 4: Seleccionar Aceptar	
El actor selecciona la Aceptar	El sistema muestra un mensaje de satisfacción
	El sistema muestra la interfaz con las lista de las plantillas
Sección 5: Selecciona Cancelar	
El actor selecciona la Cancelar	El sistema muestra la interfaz con las lista de las plantillas
Sección 2: Importar	
El actor selecciona la opción Importar	El sistema muestra una interfaz para seleccionar la plantilla desde el origen donde se encuentra en la pc.
El actor selecciona la plantilla	
El actor selección la opción aceptar o cancelar	El sistema realiza una de las siguientes opciones: • Selecciona Aceptar: Ver sección 6 “Aceptar” • Selecciona Cancelar: Ver sección 7“Cancelar “
Sección 6: Seleccionar Aceptar	
El actor selecciona la Aceptar	El sistema muestra un mensaje de satisfacción
	El sistema muestra una interfaz con las lista de las plantillas y la nueva importada.
Sección 7: Selecciona Cancelar	
El actor selecciona la Cancelar	El sistema muestra una interfaz con las lista de las plantillas
Sección 3: Buscar	
El actor introduce el nombre o el id de la plantilla buscar	El sistema realiza la búsqueda de la plantilla
	El sistema muestra el resultado de la búsqueda
Sección 1	

Tabla. 4: Descripción del Caso de Uso Administrar Reporte

Objetivo	Administrar Reporte
Actores	Administrador
Resumen	El caso de uso se inicia cuando el usuario selecciona la acción Reporte.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	
Poscondiciones	Administrar Reporte
Flujo de eventos	
Flujo básico "Administrar Reporte"	
Actor	**Sistema**
El caso de uso inicia cuando el actor selecciona el componente "Exportar Importar"	El sistema muestra una interfaz para seleccionar "Plantilla o Reporte"
El actor selecciona la opción "Reporte"	El sistema muestra una interfaz con las lista de las reportes
El actor selecciona el reporte que desea exportar y selecciona la opción exportar o directamente selecciona Importar o Buscar	El sistema realiza una de las siguientes opciones: • Selecciona Exportar: Ver sección 1 "Exportar" • Selecciona Importar: Ver sección 2 "Importar " • Selecciona Buscar: ver sección 3 "Buscar"
Sección 1: Exportar	
El actor selecciona Exportar luego de haber seleccionado el reporte.	El sistema muestra una interfaz para seleccionar el origen donde quiere guardar en la PC.
El actor selección la dirección donde quiere guardar el reporte y selecciona la opción aceptar o cancelar	El sistema realiza una de las siguientes opciones: • Selecciona Aceptar: Ver sección 4 "Aceptar" • Selecciona Cancelar: Ver sección 5 "Cancelar "
Sección 4: Seleccionar Aceptar	

El actor selecciona la Aceptar	El sistema muestra un mensaje de satisfacción.
	El sistema muestra una interfaz con las lista de los reportes.
Sección 5: Selecciona Cancelar	
El actor selecciona la Cancelar	El sistema muestra una interfaz con las lista de los reportes.
Sección 2: Importar	
El actor selecciona la Importar	El sistema muestra una interfaz para seleccionar el reporte desde el origen donde se encuentra en la PC.
El actor selecciona el reporte	
El actor selección la opción Aceptar o Cancelar.	El sistema realiza una de las siguientes opciones: • Selecciona Aceptar: Ver sección 6 “Aceptar” • Selecciona Cancelar: Ver sección 7“Cancelar “
Sección 6: Seleccionar Aceptar	
El actor selecciona la Aceptar.	El sistema muestra un mensaje de satisfacción.
	El sistema muestra una interfaz con las lista de las reportes y la nueva importada.
Sección 7: Selecciona Cancelar.	
El actor selecciona la Cancelar.	El sistema muestra una interfaz con las lista de los reportes.
Sección 3: Buscar	
El actor introduce el nombre o el id del reporte que necesita buscar.	El sistema realiza la búsqueda de los reportes.
	El sistema muestra el resultado de la búsqueda.
Sección 1	

Requisitos no Funcionales:

Son restricciones de los servicios o funciones ofrecidos por el sistema. Incluyen restricciones de tiempo, sobre el proceso de desarrollo. Los requisitos no funcionales a menudo se aplican al sistema en su totalidad. Estos normalmente se emplean en características o servicios individuales del sistema (Can, 2010).

Restricciones de Diseño:

RNF1: Lenguaje y marco de trabajo (*framework*) para el desarrollo del sistema del lado del servidor.
El componente deberá ser implementado en el lenguaje de programación PHP en su versión 5 o superior. Como *framework* de desarrollo se usará Symfony 1.1.7 el cual propone una arquitectura modular en tres capas: el modelo, la vista y el controlador.

RNF2: lenguaje y marco de trabajo para el desarrollo del sistema del lado del cliente.
Una de las bibliotecas fundamentales en el desarrollo de la herramienta es la de ExtJS 3.4 la cual permite el diseño de interfaces visuales interactivas usando metodologías como AJAX y permiten crear aplicaciones WEB con la apariencia de escritorio.

Interfaz:

RNF3: Interfaces de usuario.
Las interfaces de usuario serán diseñadas a modo de aplicaciones RIA (*Rich Internet Application*) lo que permite a los usuarios contar con aplicaciones web muy similares a las aplicaciones de escritorios. Para lograr este objetivo se usará la librería *Javascript, ExtJS*, la cual conjuga una serie de componentes visuales que proveen funcionalidades que ayudan al diseño de este tipo de aplicaciones WEB con apariencia de escritorio.

RNF4: Interfaces de Hardware.

Cliente

- Ordenador Pentium IV o superior, con 1.7 GHz de velocidad de microprocesador.
- Memoria RAM mínimo 256MB.

Servidor

- Ordenador Pentium IV o superior, con 1.7 GHz de velocidad de microprocesador.
- Memoria RAM mínimo 1GB.

RNF5: Interfaces de Software.

Un servidor en el que se instale únicamente SIGE con los siguientes elementos.

- Sistema operativo GNU/Linux Ubuntu 10.10 o versión superior.
- Servidor de aplicaciones Apache, versión 2.
- Lenguaje de programación PHP y librerías versión 5.
- Sistemas Gestor de Base de Datos PostgreSQL, versión 8.4.
- En los clientes Mozilla Firefox 3.4 o superior.

Usabilidad:

RNF6: El usuario podrá importar y exportar de manera ágil y sencilla, además de poder obtener la estructura de un reporte sin necesidad de ser un experto en el uso de la herramienta.

Licencia:

RNF7: No se posee ningún requisito de licencia o restricción en el uso del software puesto que se desarrollará con la utilización de herramientas libres como: Symfony 1.7, Postgres 8.4, PgAdminIII, NetBeans 8.0 y ExtJS 3.4.

2.5 Diseño del Sistema

El diseño del sistema es el arte de definir la arquitectura del hardware y software, componentes, módulos y datos de un sistema de cómputo para satisfacer ciertos requerimientos. Dichos requerimientos son u conjunto de actividades encaminadas a obtener las características necesarias que deberá poseer el nuevo sistema, para comprender cómo trabaja y dónde es necesario efectuar mejoras o cambios considerables (Corporation, 2015).

Modelos del diseño

El modelo de diseño describe la realización física de los casos de uso centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a desarrollar. El modelo de diseño es esencialmente el diseño mismo. Es el puente entre los requerimientos y la implementación del sistema.

Patrones arquitectónicos

Al desarrollarse el Componente para exportar e importar estructuras de reportes en el GDR en su versión

1.8 usando como framework de desarrollo Symfony en su versión 1.1.7 lo que implica utilizar como patrón arquitectónico Modelo-Vista-Controlador (MVC). Este está formado por tres niveles:

El modelo representa la información con la que trabaja la aplicación, es decir, su lógica de negocio. La base de datos pertenece a esta capa.

La vista transforma el modelo en una página web que permite al usuario interactuar con ella, se tienen presente todas las clases Javascript correspondientes a la interfaz. En Symfony la capa de la vista está formada principalmente por plantillas en PHP.

El controlador se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista. Este responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista. Incluye todas las clases.php implementadas en el sistema.

La arquitectura MVC separa la lógica de negocio (el modelo) y la presentación (la vista) por lo que se consigue un mantenimiento más sencillo de las aplicaciones. El controlador se encarga de aislar al modelo y a la vista de los detalles del protocolo utilizado para las peticiones (HTTP, consola de comandos, email). 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 base de datos utilizado por la aplicación (Potencier, 2015).

Diagrama de clases de diseño

El diagrama de clases de diseño expresa de manera gráfica todas las especificaciones de las clases del software y las interfaces de la aplicación. Esto es una representación de toda la implementación que se desea realizar. Los componentes que se utilizan de Symfony están modelados en un subsistema llamado Symfony, también Propel que no es más que el encargado de realizar el mapeo relacional de los objetos.

Se utiliza el modelo vista controlador uno de los patrones más utilizados en Symfony. El Modelo representa la información con la que trabaja la aplicación, es decir, su lógica de negocio. La Vista transforma el modelo en una página web que permite al usuario interactuar con ella y el Controlador se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

A partir de la descripción detallada de los casos de usos del sistema, se modelaron los diagramas de clases del diseño. A continuación, se muestra un ejemplo:

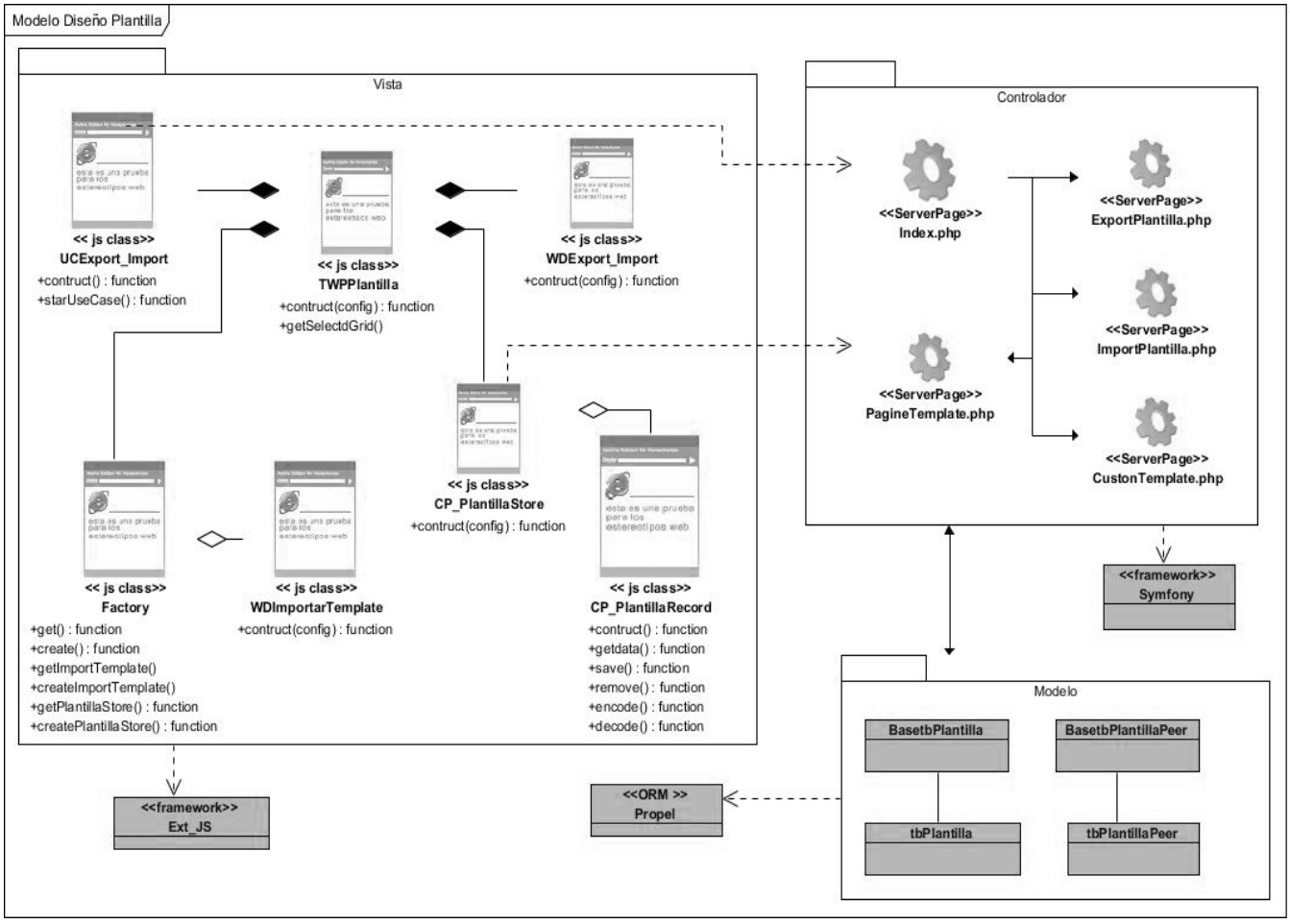


Figura 3: Diagrama de clase del diseño del caso uso Administrar Plantilla

El diagrama de clases de diseño contiene todas las clases y relaciones respectivas a la implementación del caso de uso Administrar Plantilla, además de representar sus métodos y atributos adecuados. En el diagrama se refleja el patrón arquitectónico MVC correspondiente al *framework* Symfony dando grandes facilidades al desarrollo del sistema. Por parte de la vista se evidencian todas las clases relacionadas con la interfaz de usuario desarrolladas en lenguajes Javascript utilizando los componentes del subsistema ExtJS. Por otra parte, en el paquete controlador es necesario considerar los lenguajes que se encontrarán del lado del servidor entre ellos está PHP. En Symfony la capa del controlador, que contiene el código que une la capa de negocio con la de presentación, está dividida en varios componentes que se utilizan para diversos propósitos, aquí es donde aparece como primer elemento el controlador frontal Index.php, que es el único punto de entrada a la aplicación, carga la configuración y determina la acción a ejecutarse. Además, esta capa cuenta con las acciones, las cuales contienen la lógica de la aplicación, verifica la integridad de las peticiones y preparan los datos requeridos por la capa de presentación, utilizando los componentes de Symfony. Luego de asociar la acción, esta se comunica con la capa donde radica el paquete de clases con

sus métodos y funciones necesarios para responder las peticiones del usuario, donde esta nueva capa interactúa con el modelo para el trabajo con los datos. Los elementos correspondientes al modelo pertenecen a la clase generada por el ORM Propel encargados de realizar todas a las acciones correspondientes con la base de datos, dando facilidades a la hora de consultar los datos.

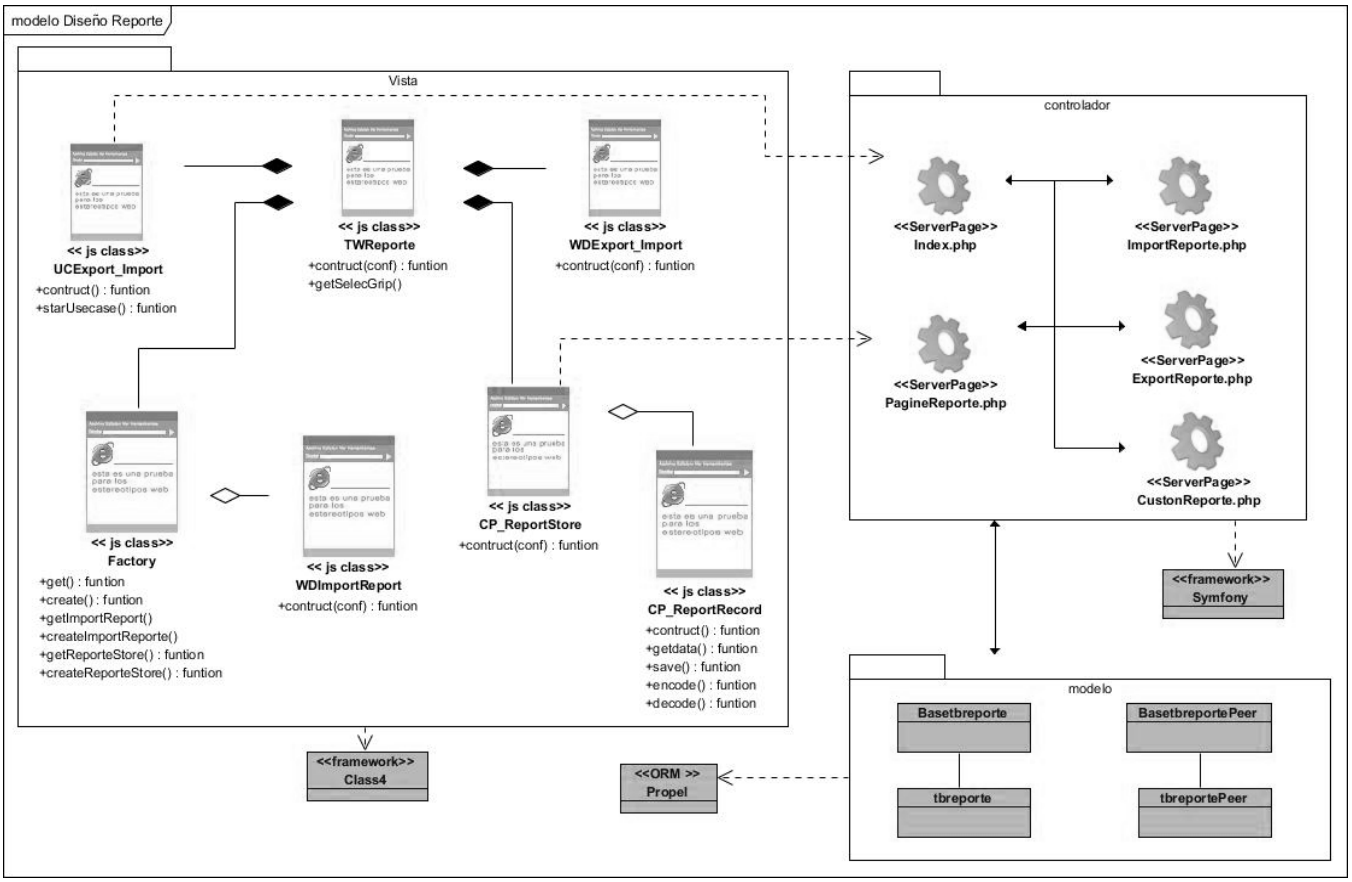


Figura 4: Diagrama de clase del diseño del caso uso Administrar Reporte

El diagrama de clases de diseño contiene todas las clases y relaciones respectivas a la implementación del caso de uso Administrar Reporte, además de representar sus métodos y atributos adecuados. En el diagrama se refleja el patrón arquitectónico MVC correspondiente al *framework* Symfony dando grandes facilidades al desarrollo del sistema. Por parte de la vista se evidencias todas las clases relacionadas con la interfaz de usuario desarrollada en lenguaje Javascript utilizando los componentes del subsistema ExtJS. Por otra parte, en el paquete controlador es necesario considerar los lenguajes que se encontrarán del lado del servidor entre ellos está PHP. En Symfony la capa del controlador, que contiene el código que une la capa de negocio con la de presentación, está dividida en varios componentes que se utilizan para diversos propósitos, aquí es donde aparece como primer elemento el controlador frontal Index.php, que es el único punto de entrada a la aplicación, carga la configuración y determina la acción a ejecutarse. Además, esta

capa cuenta con las acciones, las cuales contienen la lógica de la aplicación, verifica la integridad de las peticiones y prepara los datos requeridos por la capa de presentación, haciendo uso de los componentes de Symfony. Luego de asociar la acción, esta se comunica con la capa donde radica el paquete de clases con los métodos y funciones necesarios para responder las peticiones del usuario, donde esta nueva capa interactúa con el modelo para el trabajo con los datos. Los elementos correspondientes al modelo, pertenecen a la clase generada por el ORM Propel encargados de realizar todas a las acciones correspondientes con la base de datos, dando facilidades a la hora de consultar los datos.

Symfony está concebido de tal manera que obliga el uso de diferentes patrones de diseño. Los patrones de diseño empleados en la solución pertenecen al conjunto de patrones GRASP y los GOF, la utilización de estos patrones ha ayudado a refinar el diseño y a asignar las responsabilidades de las distintas clases de diseño, haciéndolas más sencillas y reutilizables.

Patrones de diseño

Patrones GRASP: Se denominan "Patrones GRASP" por sus siglas en inglés: *General Responsibility Assignment Software Patterns*. Como su nombre lo indica, estos patrones nos indican cual es la manera más fácil de asignar responsabilidades a objetos software. Los patrones se componen de los siguientes elementos nombre, problema, solución y consecuencia, Además de poder relacionarse y aplicarse de forma simultánea en muchos casos (Pressman, 2009)

Experto: Consiste en asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad. Experto es un patrón que se utiliza más que cualquier otro al asignar responsabilidades; es un principio básico que suele utilizarse en el diseño orientado a objetos. (Larman, 2004).

El patrón se evidencia en las clases TWPlantilla y TWReporte donde cada una cuenta con la información necesaria para cumplir con sus responsabilidades. Dichas responsabilidades son listar, buscar, exportar e importar plantilla o reportes respectivamente.

Figura 5: Patrón Experto

Alta Cohesión: Consiste en asignar una responsabilidad de modo que la cohesión siga siendo alta. En la perspectiva del diseño orientado a objetos, la cohesión es una medida de cuan relacionadas y enfocadas están las responsabilidades de una clase. Se dice que una clase tiene alta cohesión si la misma tiene responsabilidades moderadas en un área funcional y colabora con las otras para llevar a cabo las tareas (Larman, 2004).

Symfony organiza el trabajo en cuanto a la estructura del proyecto, lo cual permite crear y trabajar con clases con una alta cohesión. Este patrón se evidencia en las clases del modelo, las cuales tienen solo los atributos y métodos necesarios para que estas cumplan con su función donde existen dos tipos de clases fundamentales:

- Las clases que se encargan de la abstracción de datos que son las responsables de realizar todas las operaciones con la base de datos.

- Las clases de acceso a datos que son las responsables de interactuar con las clases de abstracción de datos, devuelven los objetos que necesitan los controladores en su forma original.

Symfony genera 4 clases por cada tabla en la base de datos, en el componente a realizar tendrá una tabla denominada Reporte, por tanto, se generarán las siguientes clases: Reporte, “BaseReporte”, “ReportePeer” y “BaseReportePeer”. De estas cuatro clases, se percibe que las clases que trabajan directamente con la base de datos, son las terminadas en “Peer”, estas clases son las encargadas de hacer las consultas a la base de datos utilizando Propel, por tanto, estas clases como clases de abstracción de datos son las que tienen los atributos necesarios para realizar dicha función, por tanto, deben implementar la responsabilidad de realizar las acciones directamente con la base de datos y aquí es donde se aplica la alta cohesión.

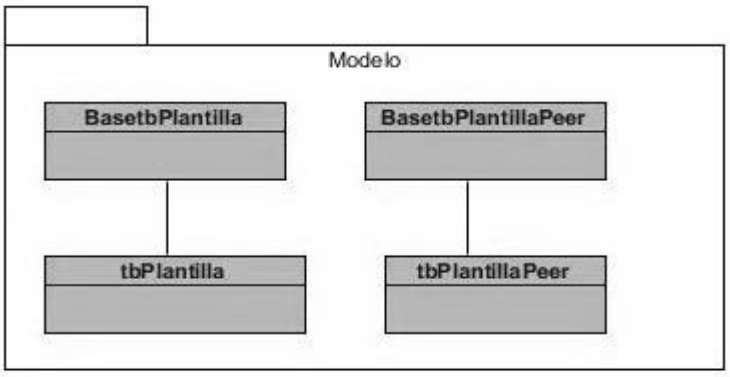


Figura 6: Patrón Alta Cohesión

Creador: Guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que debemos conectar con el objeto producido en cualquier evento (Larman, 2004).

El patrón se evidencia en la clase Factory la cual crea un objeto de la clase WDImportarTemplate y le asigna responsabilidades.

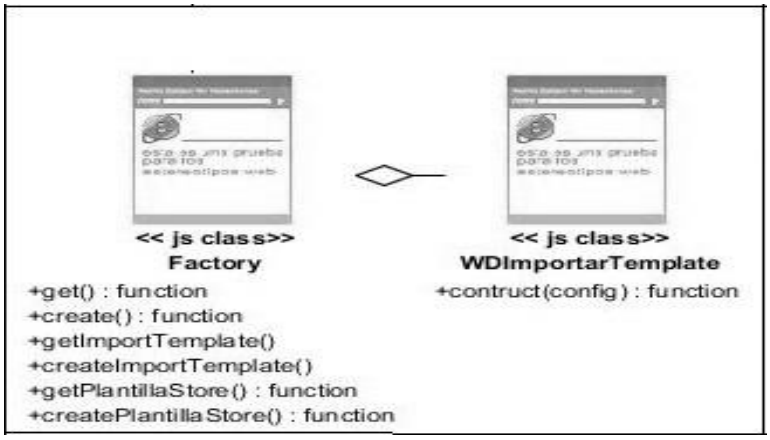


Figura 7: Patrón Creador

Controlador: Consiste en asignar la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas. Un controlador es un objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema, define además el método de su operación (Larman, 2004).

El patrón controlador Consiste en asignar la responsabilidad de controlar el flujo de eventos del sistema, este se evidencia en las clases que controlan las acciones de la aplicación, es la clase index.php se encarga de llamar las clases específicas, las cuales manejan las acciones requeridas para satisfacer las peticiones realizadas por el usuario.

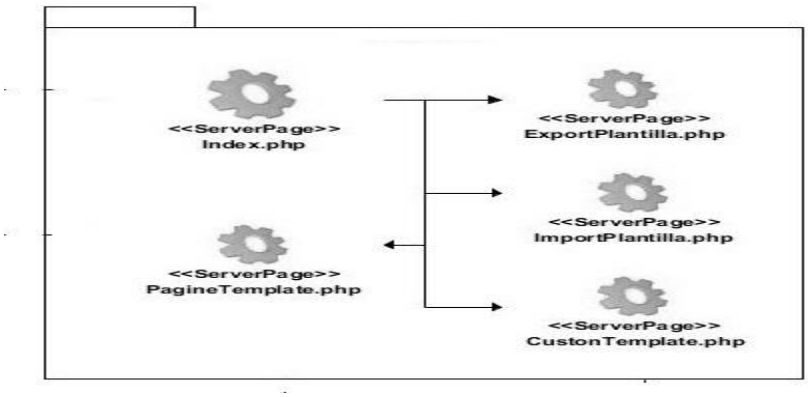


Figura 8: Patrón Controlador

Patrones GoF

Los patrones *GoF (Gang of Four),* estos patrones se utilizan basados en la experiencia acumulada al resolver problemas reiterativos. Esto ayuda a construir de manera eficiente software reutilizando código fuente ya existente. Existen varios tipos de clasificación los de creación, encargados de la creación de objetos, los de estructuras que no son más que los administradores de la composición de la clase y los objetos y están los de comportamiento que representan el modo en que las clases y objetos interactúan y se reparten las responsabilidades (Larman, 2004). Donde se evidencia en la creación, estructura y comportamiento del componente.

Fachada: El patrón fachada pertenece al grupo estructural, consiste en una interfaz común para un conjunto de implementaciones o actividades dispares (Larman, 2004). Esta evidencia en la clase TWPlantilla y TWReporte interfaces comunes para un conjunto o actividades dispares.

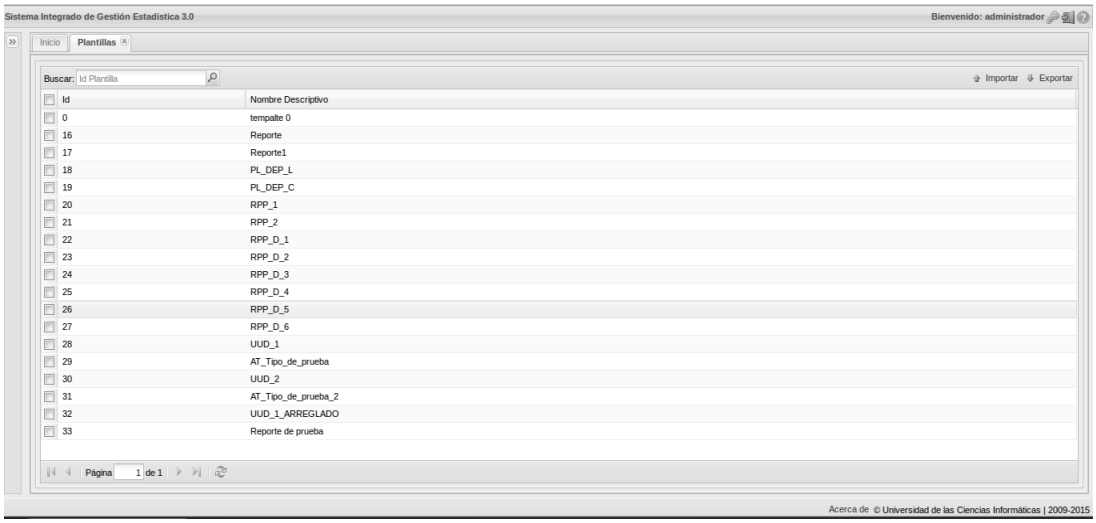


Figura 9: **Patrón Fachada TWPlantilla**

Figura 10: **Patrón Fachada TWReporte**

Factory Abstracta: Presente en los patrones de tipo creación, encargado de crear familias de objetos relacionados o que dependen entre sí, la fábrica facilita la reutilización de código. (Larman, 2004). Se evidencia en la clase Factory la cual crea un objeto de la clase WDImportarTemplate y asigna responsabilidades para dar solución a las funcionalidades de los CU.

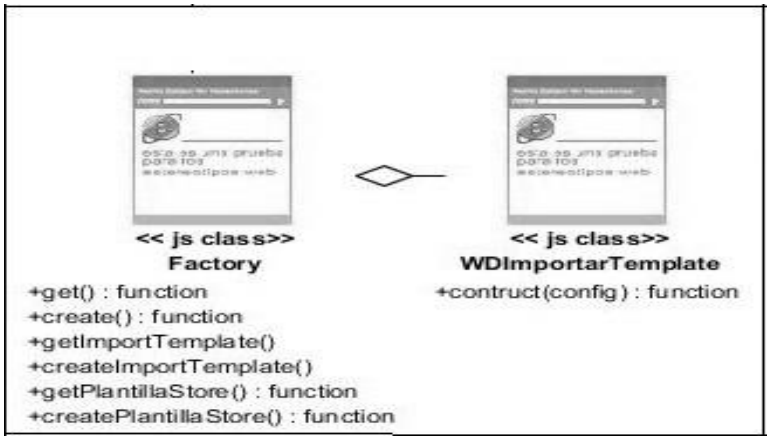


Figura 11: **Patrón Fábrica**

Diagrama de Secuencia

En el diagrama de secuencia es donde comienzan a intervenir los métodos que llevarán las clases del sistema, esto se debe a que podemos ver cómo van interactuando los objetos de las clases con los actores y con otros objetos de manera dinámica. Además, es donde se empiezan a ver qué métodos llevarán las

clases del sistema. Es un diagrama de interacción que destaca el orden temporal de los mensajes. Gráficamente, un diagrama de secuencia es una tabla que representa objetos, dispuestos a lo largo del eje X, y mensajes, ordenados según se suceden en el tiempo, a lo largo del eje Y (Pressman, 2009).

A continuación, se muestra un ejemplo:

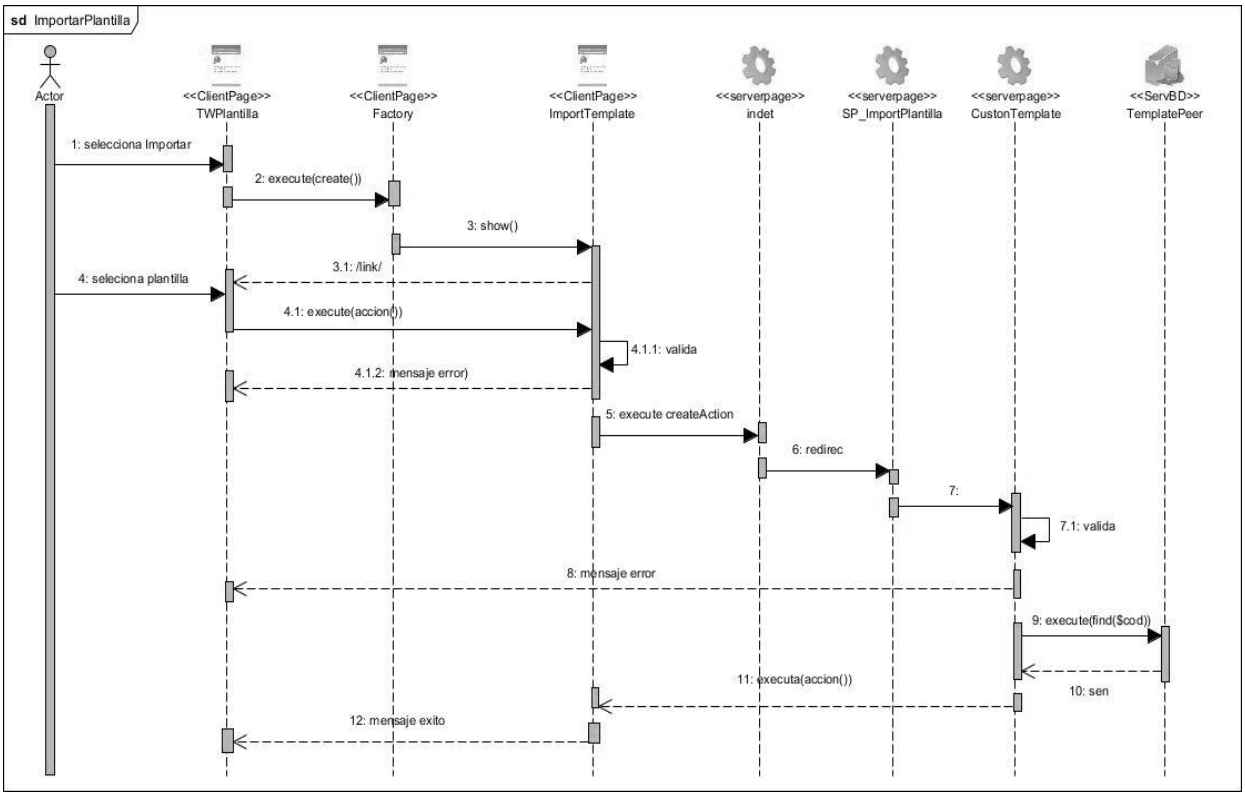


Figura 12: **Diagrama de secuencia correspondiente al escenario Importar Plantilla**

El diagrama de secuencia que se muestra en la figura 12 representa la vida en ejecución de requisito funcional Importar plantilla del caso de uso Administrar Plantilla. El usuario selecciona la opción importar en los escenarios de la interfaz del sistema correspondiente a la clase TWPlantilla la cual accede a la clase Factory del método ImportarPlantilla que a su vez acede a la clase ImportTemplate la cual solicita al usuario seleccionar la plantilla a importar. El usuario selecciona la plantilla y posteriormente la opción aceptar, la clase ImportTemplate valida que la plantilla esté en formato adecuado, de no ser así envía un mensaje de error al usuario que se muestra en la clase TWPlantilla. Si la plantilla cumple con el formato adecuado se solicita a la clase Índex.php que ejecute el método ImportarTemplate correspondiente a la clase CustonTemplate. La clase CustonTemplate valida los datos de la plantilla, de ser incorrectos envía un mensaje de error a la clase TWPlantilla en caso de no ser salvada en la base de datos. Finalmente se actualiza la lista que se muestra en la clase TWPlantilla y muestra un mensaje de éxito.

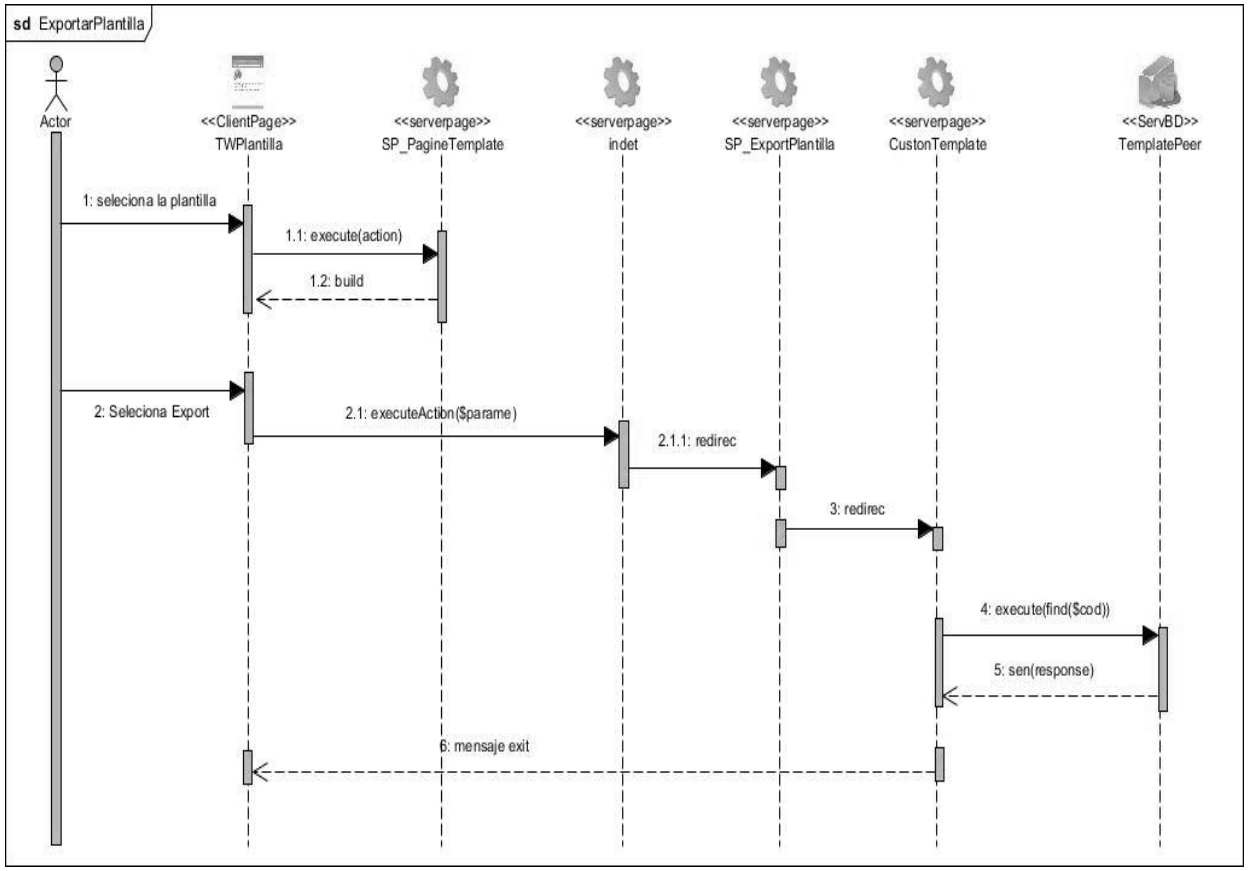


Figura 13: **Diagrama de secuencia correspondiente al escenario Exportar Plantilla**

El diagrama que se muestra en la figura 13 representa el ciclo de vida del requisito funcional Exportar Plantilla asociado en el caso de uso Administrar Plantilla. El usuario es el encargado de seleccionar la plantilla que desea exportar en la clase TWPlantilla la cual selecciona en la SPPagineTemplate la plantilla exportar, luego el usuario selecciona la opción exportar la cual se muestra en los escenarios de la interfaz del sistema correspondiente a la clase TWPlantilla la cual accede a la clase controladora Index.php y selecciona la clase PHP ExportPlantilla.php para ejecutar la acción, la cual llama el método ExportPlantilla que se encuentra en la clase CustonTemplate que es la encargada de importar la plantilla. Esta es la encargada de obtener los datos de la base de datos a través del ORM Propel y luego exporta la plantilla guardándola en la ubicación adecuada y mostrando un mensaje de satisfacción al usuario en la TWPlantilla.

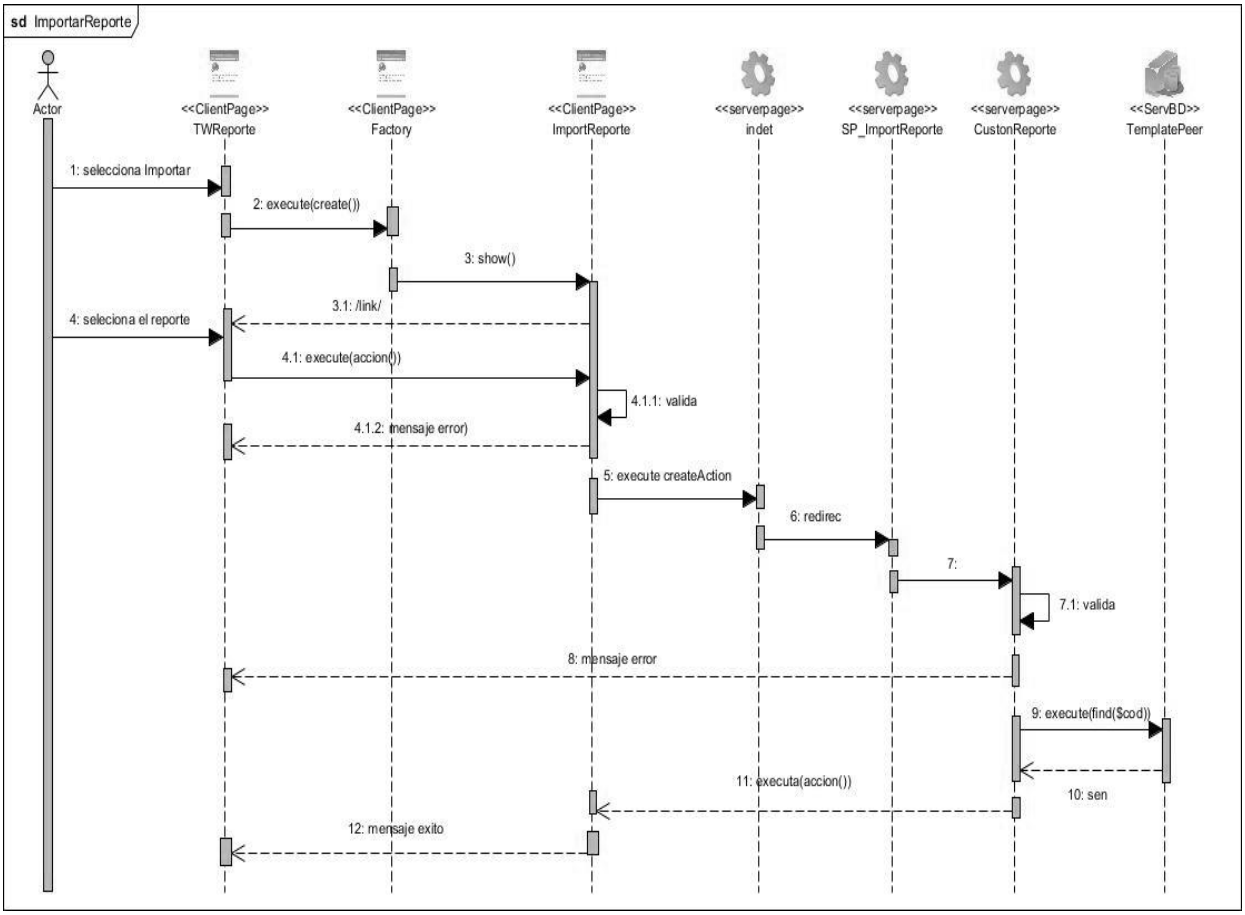


Figura 14: **Diagrama de secuencia correspondiente al escenario Importar Reporte**

El diagrama que se muestra en la figura 14 representa el ciclo de vida del requisito funcional Importar Reporte del caso de uso Administrar Reportes. El usuario selecciona la opción Importar en los escenarios de la interfaz del sistema correspondiente a la clase TWReporte, la cual accede a la clase Factory del método ImportarReporte el cual a su vez accede a la clase ImportReporte que es la encargada de solicitar al usuario seleccionar el reporte a importar. El usuario selecciona el reporte y posteriormente la opción aceptar, la clase ImportReporte valida que la reporte esté en formato adecuado, de no ser así envía un mensaje de error al usuario que se muestra en la clase TWReporte. Si el reporte cumple con el formato adecuado se solicita a la clase Index.php que ejecute el método ImportarReporte correspondiente a la clase CustonReporte. La clase CustonReporte valida los datos del reporte, mostrando un mensaje de éxito en la clase TWReporte.

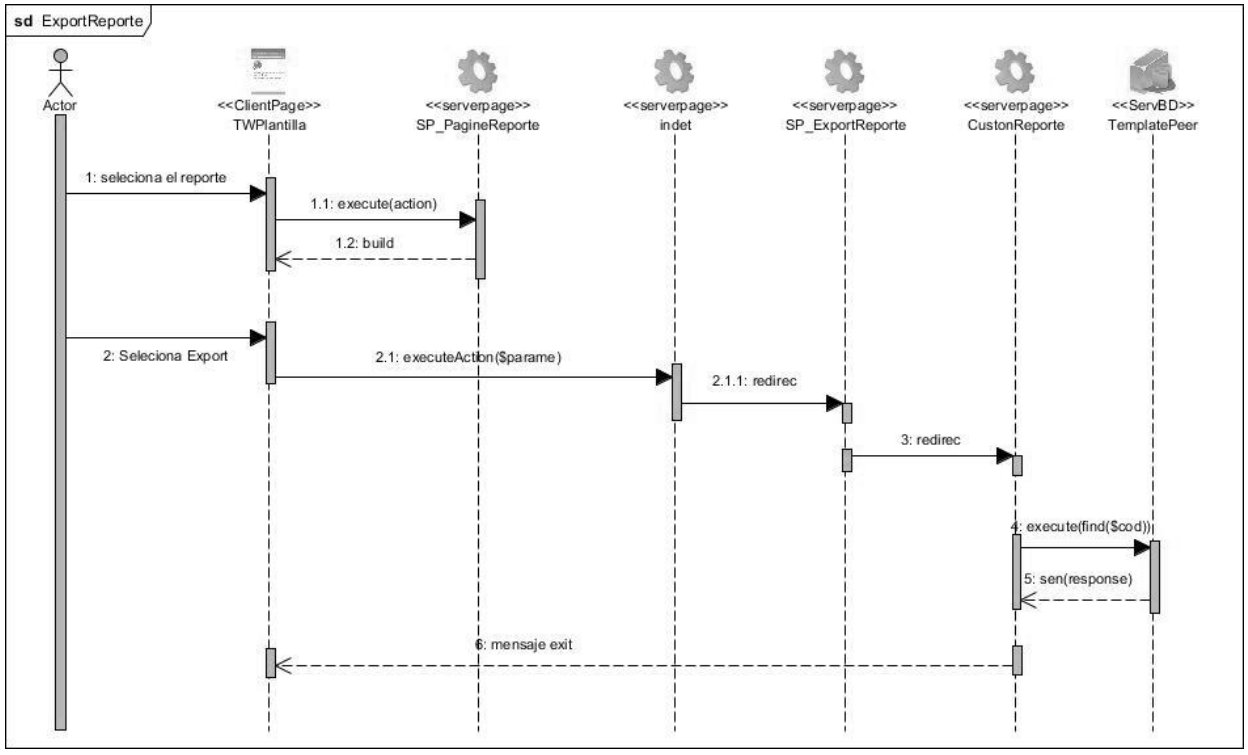


Figura 15: **Diagrama de secuencia correspondiente al escenario Exportar Reporte**

El diagrama que se muestra en la figura 15 representa el ciclo de vida del requisito funcional Exportar Reporte asociado al caso de uso Administrar Reporte. El usuario es el encargado de seleccionar el reporte que desea exportar en la clase TWReporte, la cual selecciona en la SP_PagineReporte el reporte a exportar, luego el usuario selecciona la opción Exportar, la cual se muestra en los escenarios de la interfaz del sistema correspondiente a la clase TWReporte. Esta clase accede a la controladora Index.php y selecciona la clase PHP ExportReporte.php la cual llama a ejecutar el método ExportReporte que se encuentra en la clase CustonTemplate y es la encargada de importar el reporte. Esta obtiene los datos de la base de datos a través del ORM Propel y luego exporta el reporte guardándolo en la ubicación adecuada y muestra un mensaje de satisfacción al usuario en la TWReporte.

2.6 Conclusiones del capitulo

La realización del análisis del componente permitió obtener una representación visual de las clases conceptuales del entorno a través del modelo de dominio. A partir del análisis de las funcionalidades y capacidades que debe cumplir el sistema posibilitó identificar ocho requisitos funcionales agrupados en dos casos de usos siguiendo el patrón CDRUD parcial. Se identificó el patrón arquitectónico MVC a ser utilizado en la implementación de la solución, así como los patrones de diseño Grasp y GoF utilizados en la realización de los diagramas de clases del diseño.

Capítulo 3: Implementación y Prueba

En este capítulo se desarrolla el flujo de trabajo de Implementación, se describen sus principales artefactos, destacando el Modelo de Implementación que incluye componentes, subsistemas de implementación, diagramas de componentes y el diagrama de despliegue correspondiente. Después de la implementación del diagrama de componente y el diagrama de despliegue, se realizan las pruebas necesarias con el objetivo de validar la solución implementada.

3.1 Modelo de implementación

El Modelo de implementación está comprendido por un conjunto de componentes y subsistemas que constituyen la composición física de la implementación del sistema. Entre los componentes podemos encontrar datos, archivos, ejecutables, código fuente y los directorios. Fundamentalmente, se describe la relación que existe desde los paquetes y clases del modelo de diseño a subsistemas y componentes físicos (Hernandez, 2013).

Diagrama de componente

Los diagramas de componentes describen cómo se organizan los componentes de acuerdo con los mecanismos de estructuración disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados, y cómo dependen los componentes unos de otros. Debido a que los diagramas de componentes son más parecidos a los diagramas de casos de uso, son utilizados para modelar la vista estática y dinámica de un sistema. Muestran la organización y las dependencias entre un conjunto de componentes. A continuación, teniendo en cuenta que no es necesario que un diagrama incluya todos los componentes del sistema, se muestran los diagramas para cada caso de uso identificado.

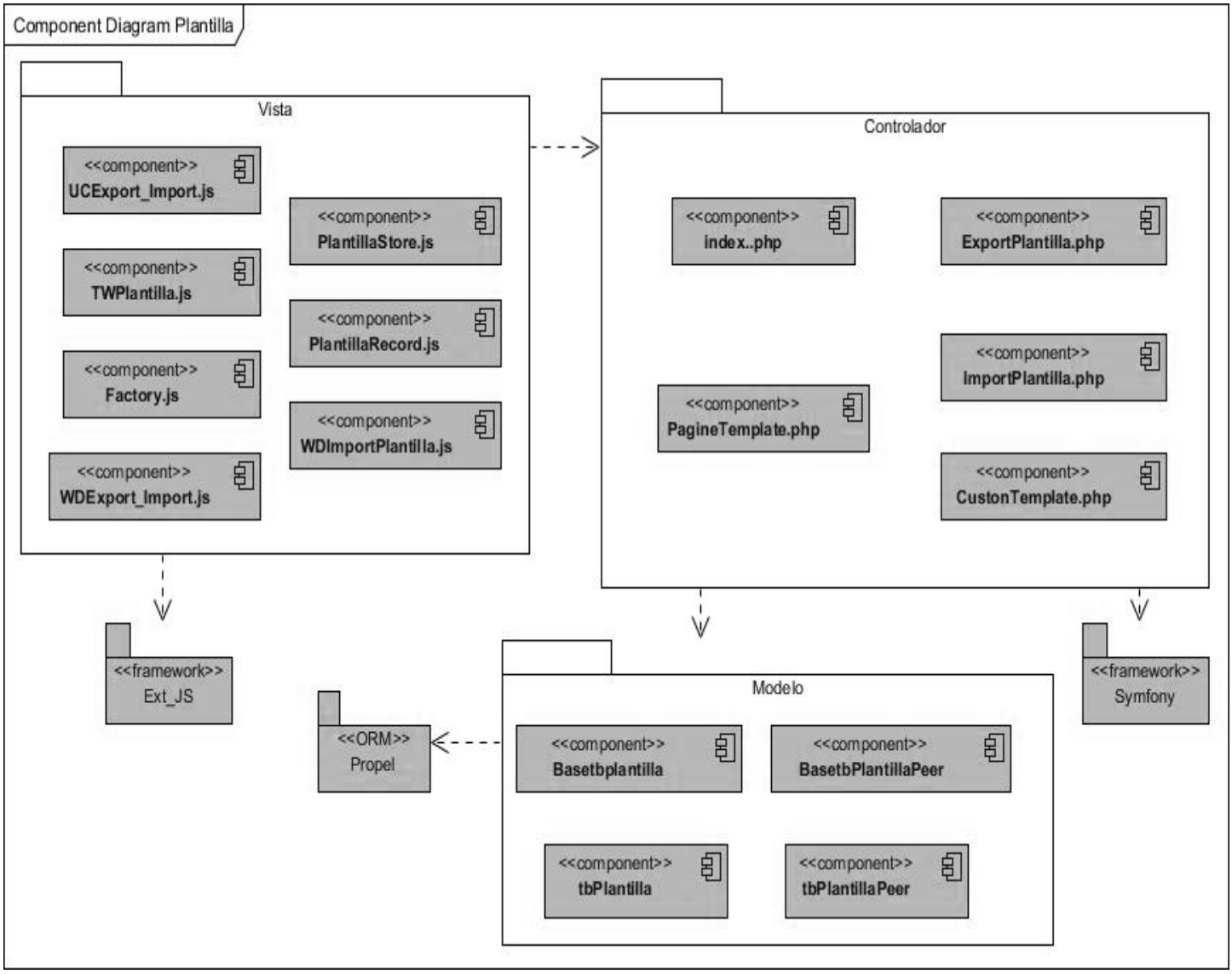


Figura. 16: Diagrama de Componente del CU Administrar Plantilla

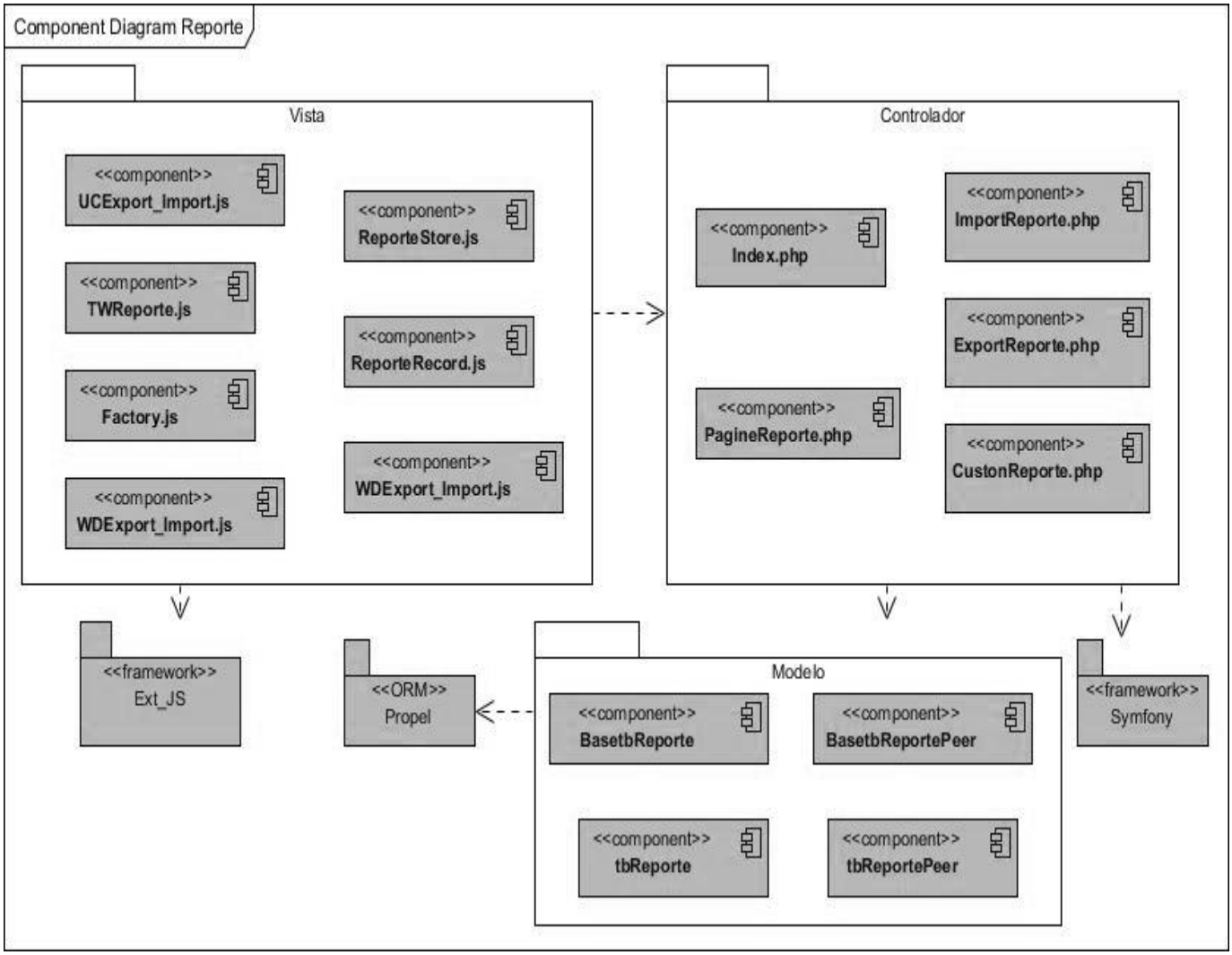


Figura. 17: Diagrama de Componente del CU Administrar Plantilla

Los diagramas de componentes realizados anteriormente representan una descripción de cómo se organizan los componentes y cómo dependen unos de otros. En la capa Vista se encuentran reflejado el paquete de componentes Vista, el cual lleva incluido el archivo que representará la interfaz de usuario de acuerdo al caso de uso diseñado, utilizando los componentes de ExtJS. El paquete de componentes Controlador contiene las acciones, las cuales responden a los servicios requeridos por los casos de uso a través del componente controlador frontal, que es la única puerta de entrada y salida a la aplicación. En este proceso el controlador utiliza el archivo componente Symfony. En la carpeta "lib" se encuentra el paquete de componentes de clases que asocian las acciones definidas por el caso de uso con la clase de la capa del modelo que mapea la tabla correspondiente de la base de datos patdsi2. El paquete de componentes Modelo utiliza Propel para el trabajo con los datos, el cual facilita la labor de desarrollo de aplicaciones web.

Modelo de despliegue

Un Diagrama de despliegue modela la arquitectura en tiempo de ejecución de un sistema. Esto muestra la configuración de los elementos de hardware (nodos) y muestra cómo los elementos y artefactos del software se trazan en esos nodos.

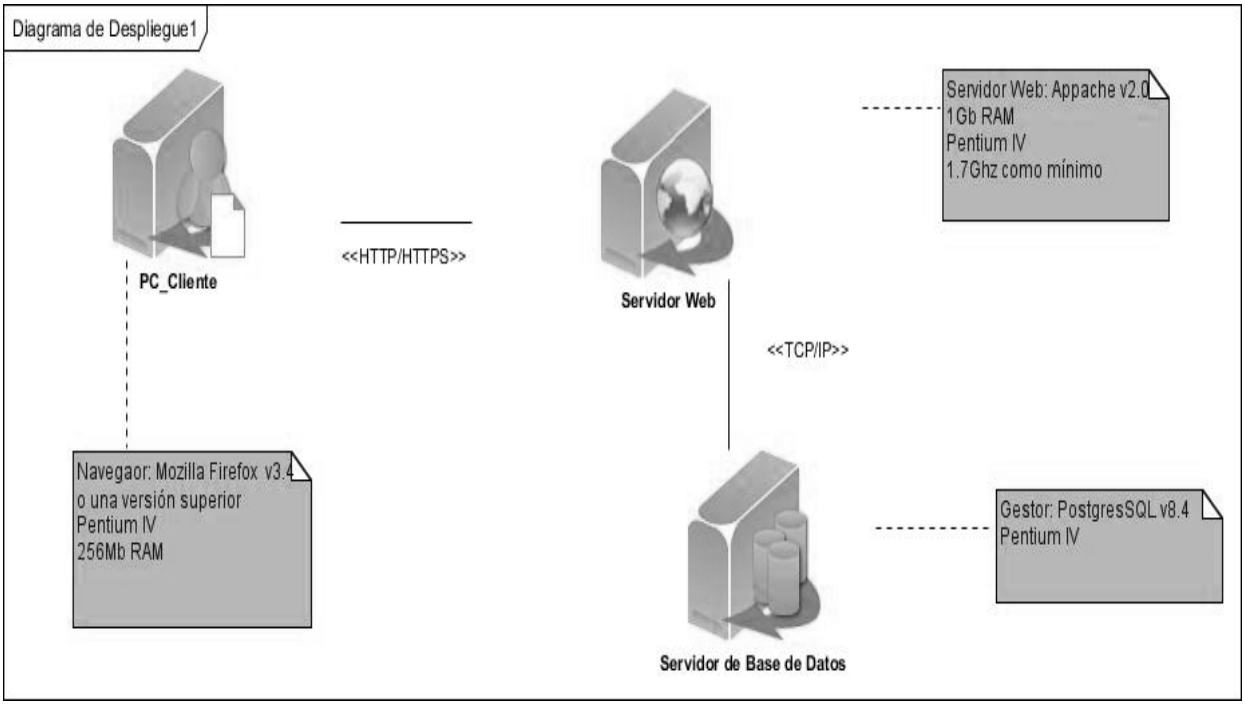


Figura. 18: Diagrama de Despliegue

Pc Cliente: se refiere a las estaciones de trabajo que el usuario utilizará para acceder a la aplicación Web y transcribir sus datos.

Servidor de Web: 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) y HTTPS (Hypertext Transfer Protocol Secure) como protocolos 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.

Conexión HTTP/HTTPS: son los protocolos utilizados entre los browsers de los clientes y el servidor Web. Este elemento de la arquitectura representa un tipo de comunicación no orientado a la conexión entre los clientes y servidor.

Conexión 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 usando el puerto 5432.

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. El servidor de base de datos elegido es PostgreSQL, el cual está disponible para Linux y Windows.

3.2 Código fuente

El código fuente de un programa informático (o software) es un conjunto de líneas de texto (instrucciones), que debe seguir la computadora para ejecutar dicho programa. Este primer estado no es directamente ejecutable por la computadora, sino que debe ser traducido a otro lenguaje o código binario. Para esta traducción se usan los llamados compiladores, ensambladores y otros sistemas de traducción (Pressman, 2009).

Estándares de codificación

Se definen estándares de codificación porque son un estilo de programación homogéneo. Debido a que la forma de escribir código es propia de cada programador y completamente diferente a la forma de cualquier otro. De la forma usada depende la facilidad para entender el código y retomar ciertas partes realizadas por otros integrantes, así como la depuración de las mismas y permite que todos los participantes del proyecto puedan entenderlo en menos tiempo.

Estándares de codificación utilizados

1. Los bloques de código siempre deben estar encerrados por llaves, si consta de una línea no es necesario utilizar llaves.

```
public static function metodSacarDatos($template) {
    $xml = simplexml_load_string($template);
    $schema = $xml->sourceReport;
    $atributos = (array) $schema->attributes();
    $atributos = $atributos['@attributes'];

    return $atributos;
}
```

2. Los nombres de funciones y variables deben empezar siempre con una letra minúscula utilizando la forma "camelCase" el cual es un estilo de escritura que se aplica a frases o palabras compuestas.

```
public static function exportReporteXML($idtemplate) {...119 lines }

public static function importReporteFromXML($filename) {...77 lines }
```

3. Se recomienda en sentido general la elocuencia, los nombres de las funciones deben ser elocuentes como para describir su propósito y comportamiento.

```
public static function exportReporteXML($idtemplate) {...119 lines }

public static function importReporteFromXML($filename) {...77 lines }
```

4. Los métodos de los objetos deben ser nombrados utilizando la forma ‘camelCase ‘.

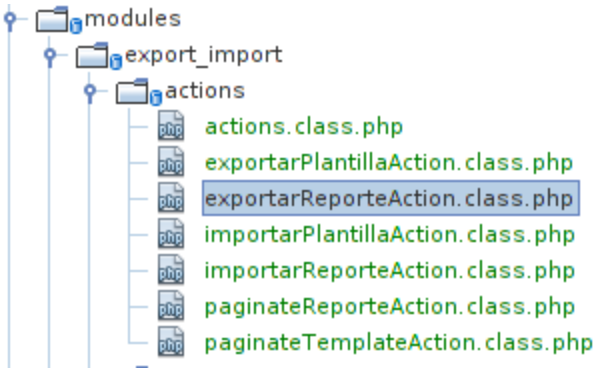
5. Los ficheros tienen que concluir con la extensión .js, y no deben incluir signos de puntuación excepto – (guión medio) o _ (guión bajo).

6. Los números están permitidos en los nombres de clase, pero no son convenientes a la hora de programar.

7. Las cadenas tienen que ser definidas utilizando comillas simples siempre que sea posible, para obtener un mejor rendimiento.

```
try {
    $xml = simplexml_load_file(sfConfig::get() . '/var/www/html/Sige-Trunk/data/' . $filename);
    unlink(sfConfig::get() . '/var/www/html/Sige-Trunk/data/' . $filename);
```

8. ¿Todas las etiquetas php deben ser completas (<? php?>) ... no reducidas (<? ?>).



9. Los nombres de clases pueden contener sólo caracteres alfanuméricos.

3.3 Pruebas de software

La prueba del software es un elemento crítico para la garantía de calidad del software y representa una revisión de las especificaciones, del diseño y de la codificación. Una vez generado el código fuente es necesario probar el software para descubrir y corregir la mayor cantidad de errores posibles antes de ser entregado. Su objetivo es diseñar una serie de casos de prueba que tengan una alta probabilidad de encontrar errores (Pressman, 2009).

El objetivo principal de la ejecución de las pruebas está dado a:

- Descubrir tantos errores como sea posible.
- Notificar acerca de los riesgos percibidos del proyecto.
- Cumplir con los requerimientos específicos del cliente, en cuanto a la ejecución de las pruebas.

En la presente investigación se define la estrategia de prueba que a continuación se especifica.

Estrategia de prueba

La estrategia de pruebas no es más que describir el enfoque y los objetivos generales de las actividades de prueba que se realizan con el objetivo de tener una mejor planificación y claridad. Estas incluyen los niveles de pruebas y tipos de pruebas a ser ejecutadas (Pressman, 2009).

- Técnicas de pruebas y herramientas a ser usadas.
- Conocimientos y formación de quienes ejecutarán las pruebas.
- Qué criterios de éxitos y culminación de la prueba serán usados.

Niveles de Pruebas

El ámbito o destino de las pruebas se software variar en tres niveles

1. Pruebas Unitarias.
2. Pruebas Integración.
3. Pruebas de validación o funcionales.

Pruebas unitarias: Las pruebas unitarias presentan el proceso de verificación desde la más mínima unidad del diseño del software: el módulo. Está orientada a caja blanca y este paso se puede llevar a cabo en paralelo para múltiples módulos. Además, se concentran en probar cada componente individualmente para asegurar que funcione de manera apropiada como unidad (Pressman, 2009).

Pruebas de Caja blanca

La prueba de la caja blanca es un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para derivar los casos de prueba. Se realiza un examen minucioso de los detalles procedimentales, comprobando los caminos lógicos del programa, comprobando los bucles y condiciones, y examinado el estado del programa en varios puntos (Pressman, 2009)

Las pruebas de caja blanca intentan garantizar que:

- Se ejecutan al menos una vez todos los caminos independientes de cada componente.
- Se utilizan las decisiones en su parte verdadera y en su parte falsa.
- Se ejecuten todos los bucles en sus límites.
- Se utilizan todas las estructuras de datos internas.

Existen varios métodos de pruebas de cajas blanca entre los que se encuentra Camino Básico y las Pruebas de estructura de control.

Como tipo de prueba se decide aplicar el método **Camino Básico**. El método Camino Básico permite obtener una medida de la complejidad de un diseño procedimental, y utilizar esta medida como guía para la definición de una serie de caminos básicos de ejecución, diseñando casos de prueba que garanticen que cada camino se ejecute al menos una vez. A continuación de evidencian una serie de pasos:

- Obtener el grafo de flujo, a partir del diseño o del código del componente.
- Obtener la complejidad ciclomática del grafo de flujo.
- Definir el conjunto básico de caminos independientes.
- Determinar los casos de prueba que permitan la ejecución de cada uno de los caminos anteriores.
- Ejecutar cada caso de prueba y comprobar que los resultados son los esperados.

A continuación, se muestra un ejemplo basado en un diagrama de flujo que representa la estructura de control del programa en el método ExportarPlantilla.

```
this.btnExport = new Ext.Button({
    text: 'Exportar',
    scope: this,
    iconCls: 'exportar',
    tooltip: 'Exportar',
    handler: function() {
        if (!this.getSelectedGrid().getSelectionModel().getCount()) {
            Ext.msg.show('Debe seleccionar la plantilla a exportar.', Ext.msg.STATUS_WARNING);
        } else {
            var template = this.getSelectedGrid().getSelectionModel().getSelected();
            window.open('/tools.php/export_import/exportarPlantilla' + '?idtemplate=' + template.get('idtemplate'));
        }
    }
});
```

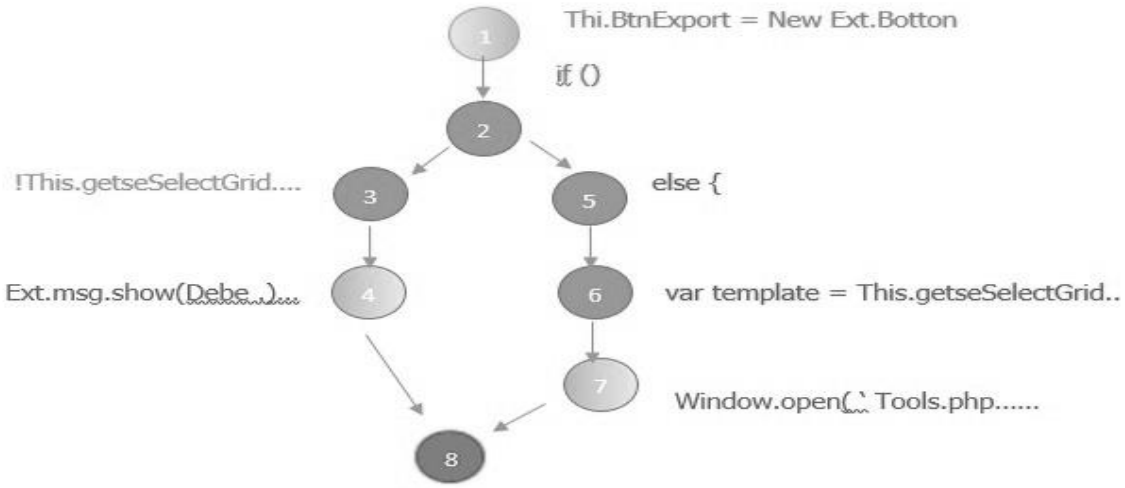


Figura. 19: Código y Gráfico método ExportarPlantilla

Luego se calcula la complejidad ciclomática que no es más que una métrica de software que proporciona una medida cuantitativa de la complejidad lógica del software. El valor calculado mediante la complejidad ciclomática define el número de rutas independientes y proporciona un límite superior para el número de pruebas que debe aplicarse.

G – Gráfica, E – número de aristas, N—número de nodos.

V(G) = E-N+2 V(G)=8-8+2 V(G)=2

A partir del valor de la complejidad ciclomática obtenemos el número de caminos independientes, que nos dan un valor límite para el número de pruebas que tenemos que diseñar. En el ejemplo, el número de caminos independientes es 2, y los caminos independientes son:

Caminos 1: (1-2-3-4) Camino2:(1-2-5-6-7-8)

A continuación, se muestra otro ejemplo basado en un diagrama de flujo que representa la estructura de control del programa en el método Vista que se encarga de obtener las vistas del reporte para posteriormente agregarlas al XML y exportarlas.

```
public static function vistas($template) {
    $xml = simplexml_load_string($template);
    $parameters = (array) $xml->sourceReport->parameters;
    $arrayParameters = $parameters["parameter"];
    $bdh = AgentExplorer::pdoSIGE();
    $resultParams = array();
    foreach ($arrayParameters as $param) {
        $atributos = (array) $param->attributes();
        $atributos = $atributos['@attributes'];
        if ($atributos["source"] == "field") {
            $temp = array();
            $data = $param->data->field;
            $atributosData = (array) $data->attributes();
            $atributosData = $atributosData['@attributes'];
            $schema = $atributosData["schema"];
            $nombre = $atributosData["table"];
            $sql = "select viewname as nombre ,viewowner as propietario,definition as definicion from pg_views where viewname=" . "'" . $nombre . "'
and schemaname=" . "'" . $schema . "';";

            $stmt = $bdh->prepare($sql);
            $stmt->execute();
            $temp["schema"] = $schema;
            $temp["nombre"] = $nombre;
            $fijo = "CREATE OR REPLACE VIEW " . $schema . "." . $nombre . " AS ";
            while ($row = $stmt->fetch(PDO::FETCH_NAMED)) {
                $definicion = $row["definicion"];
            }
            $temp["codigo"] = $fijo . $definicion;
            $resultParams[] = $temp;
        }
    }
    return $resultParams;
```

Figura. 20: Código fuente del método Vista

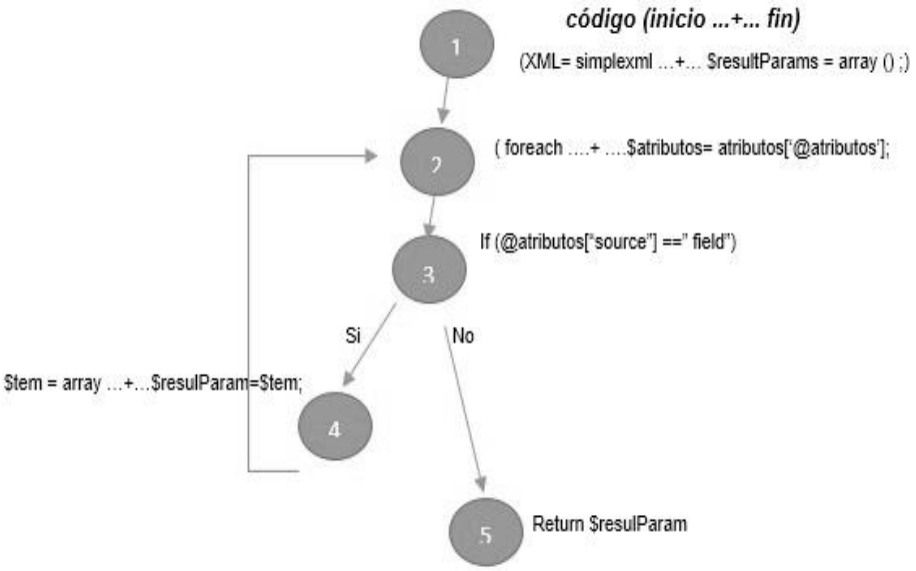


Figura. 21: Gráfico método Vista

Luego se calcula la complejidad ciclomática que no es más que una métrica de software que proporciona una medida cuantitativa de la complejidad lógica del software. El valor calculado mediante la complejidad ciclomática define el número de rutas independientes y proporciona un límite superior para el número de pruebas que debe aplicarse. G – Gráfica, E – número de aristas, N—número de nodos.

V(G) = E-N+2 V(G) = 5-5+2 V(G)= 2

A partir del valor de la complejidad ciclomática obtenemos el número de caminos independientes, que nos dan un valor límite para el número de pruebas que tenemos que diseñar. En el ejemplo, el número de caminos independientes es 2, y los caminos independientes son:

Caminos 1: (1-2-3-4) Camino2:(1-2-3-5)

Al realizar las pruebas unitarias por el método camino básico se identificaron los caminos independientes correspondientes a cada método ejecutado, obteniendo como resultado el número de posibles caminos a recorrer por el método al ser ejecutado. Logrando como efecto un correcto funcionamiento de los métodos seleccionados para la prueba.

Pruebas de integración: El objetivo de las pruebas de integración es verificar el correcto ensamblaje entre los distintos módulos que componen la solución una vez que han sido probados unitariamente con el fin de comprobar que interactúan correctamente a través de sus interfaces internas y externas, que cubren la

funcionalidad establecida y se ajustan a los requisitos no funcionales especificados en las verificaciones correspondientes. En esta prueba se comprueba la compatibilidad y funcionalidad de los interfaces entre las distintas 'partes' que componen el desarrollo de la solución.

Como tipo de prueba se decide aplicar las pruebas de **Integración Ascendente**, que no es más que empezar la construcción y la prueba con módulos atómicos (es decir, componentes de los niveles más bajos de la estructura del programa). Ya que los componentes se integran de abajo hacia arriba, siempre está disponible el procesamiento requerido para los componentes subordinados a un determinado nivel y se elimina la necesidad de resguardarlos (Pressman, 2009).

Pasos para realizar la integración.

1. Ir a las tablas de la base de datos en el esquema de seguridad, incluir el nombre (**nameApp**) de la nueva app en la tabla **napp** y el del módulo (**nameModule**) correspondiente a la app creada con el identificador que lo relacione con la misma en la tabla **nmodule**.
2. Abrir la consola y dar todos los permisos al directorio donde está ubicado el proyecto **chmod -R 777 + (ruta del proyecto).**
3. Ubicarse dentro del proyecto por consola (cd + (ruta del proyecto)) y ejecutar los siguientes comandos
    a. *php Symfony generate: app nameApp*
    b. *php Symfony generate: module nameApp nameModule*
4. Crear la jerarquía de carpetas en la ruta web/js/padtsi2 carpeta padre *nameApp* y carpeta hija *nameModule* dentro de esta última se incluyen todos los .js correspondientes al módulo creado (incluir el archivo MODNameApp.js dentro de la carpeta *nameApp)*
5. Incluir las rutas de los .js creados en el fichero indexTestSuccess.php encontrado en la ruta apps/patdsi3/modules/menu/templates/ indexTestSuccess.php
6. Incluir los namespace en el archivo patdsi-base.js en la ruta web/js/patdsi2/ patdsi-base.js

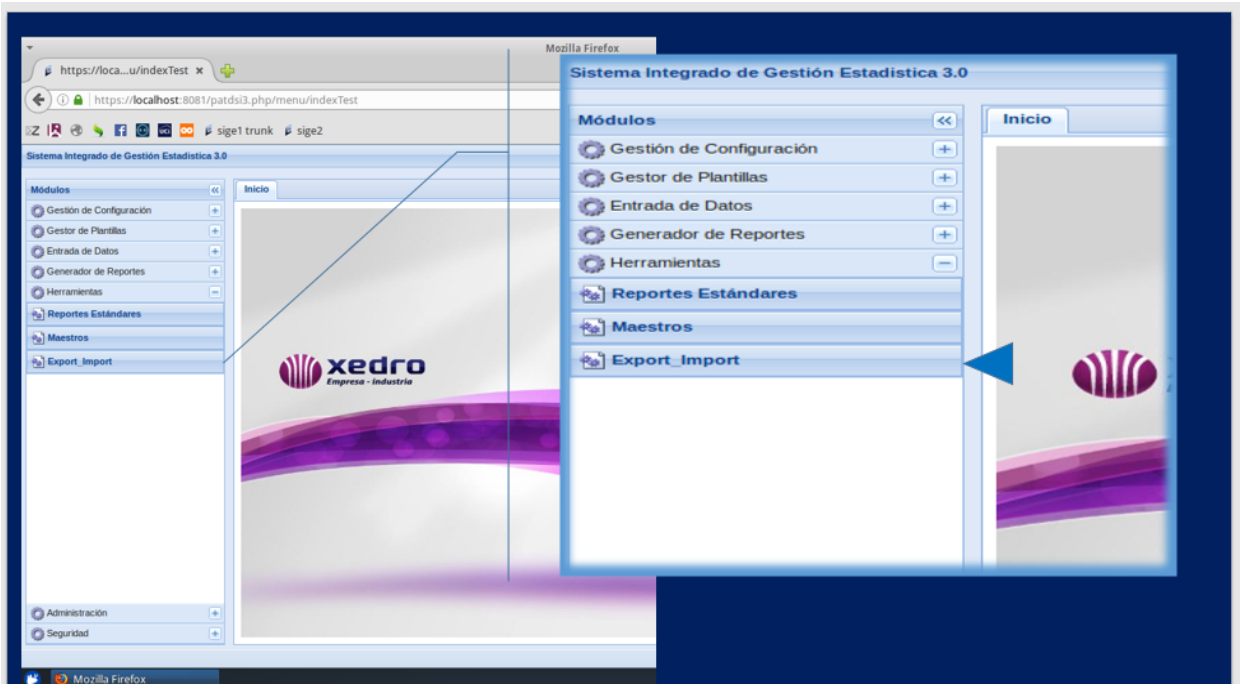


Figura. 22: Integración del sistema

Pruebas de Caja negra

En la prueba de caja negra, los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce una salida correcta. Estas se llevan a cabo sobre la interfaz del software, obviando el comportamiento interno y la estructura del programa.

Los casos de prueba de la caja negra pretenden demostrar que:

- Las funciones del software son operativas.
- La entrada se acepta de forma correcta.
- Se produce una salida correcta.
- La integridad de la información externa se mantiene.

Existen varios métodos de pruebas de cajas negra entre los que se encuentra métodos gráficos de prueba, análisis de valores límites, partición equivalente, entre otras.

Pruebas funcionales o validación: Las pruebas funcionales son un aseguramiento final de que el software cumple con todos los requisitos funcionales, se centran en los requisitos funcionales del software permitiendo al ingeniero obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos del programa, es decir consideran la función para la cual fue creado el producto (lo que hace).

Se llevan a cabo sobre la interfaz del sistema reduciendo el número de casos de prueba mediante la elección de entradas y salidas válidas y no válidas que ejercitan toda la funcionalidad del sistema (Pressman, 2009)

Dentro de esta clasificación se encuentra la técnica **Partición Equivalente** el cual es un método de prueba de caja negra, divide el dominio de entrada de un programa en clases de datos de los que pueden derivarse casos de prueba, también:

- Se dirige a la definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar.
- El diseño de casos de prueba para la partición equivalente se basa en una evaluación de las clases de equivalencia para una condición de entrada.
- Una clase de equivalencia representa un conjunto de estados válidos y no válidos para condiciones de entrada, que no son más que valores numéricos específicos, un rango de valores, un conjunto de valores relacionados o una condición lógica.

Casos de prueba: "Conjunto de entrada de prueba, condición de ejecución, y resultado esperado desarrollados para un objetivo en particular, como el ejercicio de una ruta de programa en particular o para verificar el cumplimiento de un requisito específico, y como documentación que especifique las entradas, los resultados previos, y un conjunto de condiciones de ejecuciones de un elemento de prueba". (IEEE) Para la siguiente investigación se definieron diez casos de prueba en correspondencia con los CU. A continuación, se presenta un ejemplo de un escenario del caso de prueba Administrar Plantilla.

Tabla. 5: SC Exportar Plantilla.

Escenario	Descripción	Variable 1	Variable 2	Respuesta del Sistema	Flujo central
EC 1.1 Exportar Plantilla	En este escenario se exporta la plantilla al formato XML.	NA	NA	El sistema exporta la plantilla deseada.	1. Seleccionar la plantilla a exportar. 2. Luego la opción "Exportar" en la interfaz. 3.Seleccionar la opción aceptar
EC 1.2 Exportar Plantilla (no está seleccionada la plantilla a exportar)	Este escenario no debe permitir exportar la plantilla si la misma no está seleccionada.	NA	NA	El sistema muestra una notificación "Debe seleccionar la plantilla a Exportar".	1. No seleccionar ninguna plantilla. 2. Luego la opción "Exportar" en la interfaz. 3.Seleccionar la opción aceptar

Tabla. 6: SC Listar Plantilla.

Escenario	Descripción	Variable 1	Variable 2	Respuesta del Sistema	Flujo central
EC 1.1 Listar Plantilla	En este escenario se lista las plantillas correctamente.	NA	NA	El sistema lista las plantillas.	1.Seleccionar modulo. 2. Se selecciona la opción plantilla. 3.Se da click en la opción aceptar.

Tabla. 7: SC Buscar Plantilla

Escenario	Descripción	Variable 1	Variable 2	Respuesta del Sistema	Flujo central
EC 1.1 Buscar plantilla	En este escenario se visualiza la plantilla filtrada.	V id plantilla	V nombre	El sistema visualiza la plantilla filtrada.	1. Introducir id de la plantilla. 2. Seleccionar la opción buscar
EC 1.2 Buscar Plantilla (campo vacío)	Este escenario no debe mostrar resultados.	I	NA	El sistema no muestra ningún resultado.	1. Dejar campo id vacío. 2. Seleccionar opción buscar
EC 1.2 Buscar Plantilla (datos incorrectos)	Este escenario no debe mostrar resultados.	I	NA	El sistema no muestra ningún resultado.	1. Introducir valor no valido. 2. Seleccionar opción buscar

Tabla. 8: SC Importar Plantilla.

Escenario	Descripción	Variable 1	Variable 2	Respuesta del Sistema	Flujo central
EC 1.1 Buscar plantilla	*En este escenario se visualiza la plantilla filtrada.*	*V* *id plantilla*	*V* *nombre*	*El sistema visualiza la plantilla filtrada.*	*1. Introducir id de la plantilla. 2. Seleccionar la opción buscar*
EC 1.2 Buscar Plantilla (campo vacío)	*Este escenario no debe mostrar resultados.*	*I*	*NA*	*El sistema no muestra ningún resultado.*	*1. Dejar campo id vacío. 2. Seleccionar opción buscar*
EC 1.2 Buscar Plantilla (datos incorrectos)	*Este escenario no debe mostrar resultados.*	*I*	*NA*	*El sistema no muestra ningún resultado.*	*1. Introducir valor no valido. 2. Seleccionar opción buscar*

Luego de ser revisados los casos de pruebas correspondientes al sistema Componente para exportar e importar estructuras de reportes en GDR en su versión 1.8 se detectaron un total de 12 no conformidades, nueve de ellas de valor significativas críticas, un ejemplo de estas consiste en que, a la hora de importar un reporte, este no se visualizaba en el visor de reporte del componente de GDR debido a que se deben insertar datos específicos del reporte en tablas correspondientes al visor de reporte para poder ser visualizados, tales como> id_categoria a la que pertenece el reporte e id_modelo entre otros. Por otra parte, se detectaron tres no conformidad de diseño las cuales estuvieron relacionadas con la interfaz del sistema. La siguiente

gráfica muestra en tres iteraciones el resultado obtenido. Dándole solución a todas las no conformidades detectadas cumpliendo así con el correcto funcionamiento de los requisitos identificados.

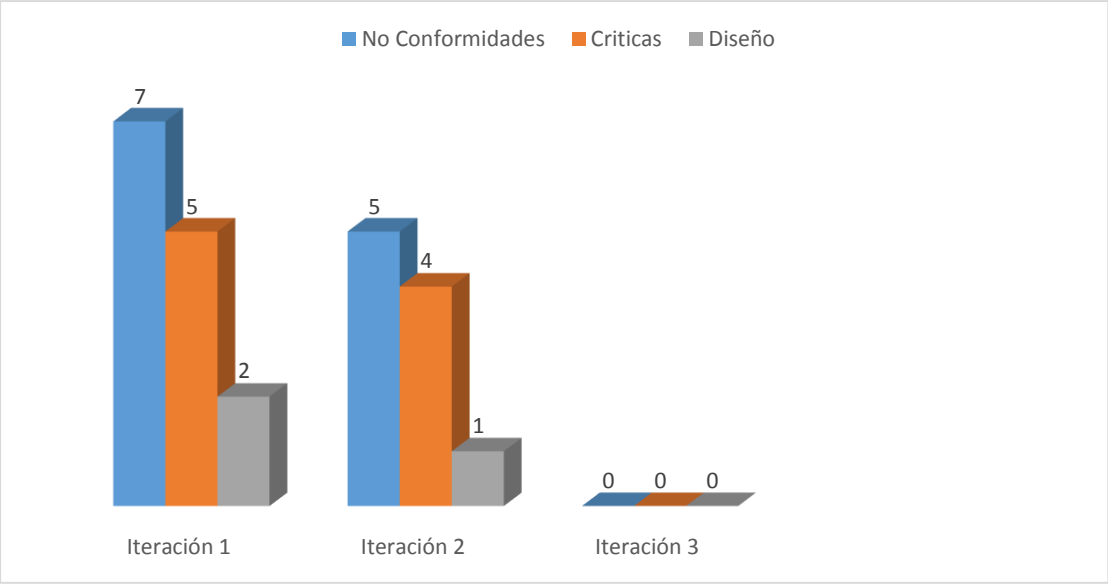


Figura. 23: Resultados de las pruebas funciónales

Pruebas de aceptación

La prueba de aceptación es la prueba final antes del despliegue del sistema. Su objetivo es verificar que el software está listo y que puede ser usado por los usuarios finales para ejecutar aquellas funciones y tareas para las cuales fue construido (PRESSMAN, 2009).

Las pruebas de aceptación son aquellas que son diseñadas por el propio equipo de desarrollo en base a los requisitos funcionales especificados, y ejecutadas por el propio usuario de manera que este dé validez y conformidad al producto que se le está entregado en base a lo que se acordó inicialmente. De forma general las pruebas de aceptación pueden afrontarse mediante dos tipos de procedimiento para realizarlas: pruebas alfa y pruebas beta (PRESSMAN, 2009).

Se decide aplicar las pruebas alfa ya que en ellas se le entrega a un usuario final el producto terminado, junto a su documentación correspondiente para que éste, en presencia del desarrollador y en entornos previamente preparados para el proceso, vaya informando de las inconsistencias y errores que detecte. En este caso el usuario final es el proyecto, el cual con el líder del proyecto y los desarrolladores efectuaron las pruebas de aceptación, ver anexo 1 fig:23.

3.4 Conclusiones del capítulo

En el presente capítulo se trataron aspectos importantes sobre el desarrollo del componente, se obtuvieron los diagramas de componentes a través de la estructura que presenta los diagramas de clases del diseño permitiendo mostrar las dependencias que existen entre estos componentes físicos incluyendo archivos, módulos, ejecutables y paquetes. De esta forma se realizó la implementación del sistema en términos de componentes, ofreciendo solución a los requisitos especificados en el capítulo anterior. Se realiza el diagrama de despliegue, permitiendo una descripción física de los componentes a lo largo de la infraestructura del sistema. Una vez realizadas las pruebas de funcionales y de aceptación se identificaron 12 no conformidades de ella 9 criticas y 3 de diseño las cuales fueron resueltas satisfactoriamente logrando obtener una aplicación que cumple con los requerimientos solicitados por el cliente.

Conclusiones Generales

A partir de la investigación realizada y los resultados obtenidos se arriba a las siguientes conclusiones:

- El análisis de los fundamentos teóricos referentes a la gestión de reportes permitió la correcta selección de la metodología y herramientas a utilizar para el desarrollo de la solución brindada, así como las principales características que debía cumplir la misma.
- El análisis y diseño del componente, proporcionó el punto de partida para las actividades de implementación y desarrollo del sistema.
- Se implementó el componente para exportar/importar estructuras de reportes posibilitado la reutilización e incorporación de las estructuras de reportes existentes para crear nuevos reportes.
- La realización de las pruebas permitió validar el correcto funcionamiento del componente demostrando que los indicadores de calidad cumplieron satisfactoriamente con los requerimientos del cliente.

Por todo lo anterior se concluye que los objetivos propuestos para la investigación fueron cumplidos satisfactoriamente.

Recomendaciones

Después de haber alcanzado los objetivos que se trazaron al principio de este trabajo se propone la siguiente recomendación.

- Agregar al componente implementado la funcionalidad de reutilizar e incorporar la estructura de los reportes desarrollados en los otros gestores de bases de datos, MySQL y Oracle que son los soportados por el GDR en su versión 1.8.

Referencias

C*l* Albasanz, 14 Bis. EXES. *Manual de XML.* [En línea] Area de producion y desarrollo. http://www.mundolinux.info/que-es-xml.htm.

Achour, Mehdi. 2011. Manual de PHP. [En línea] 2011. http://php.net/manual/es/preface.php.

Aguila, Rolando Morales. 2013. Repositorio Instituional, Tesis. [En línea] 2013. http://repositorio_institucional.uci.cu/jspui/handle/ident/TD_07022_13.

Alegsa. 2010. Definición de lenguaje de programación. [En línea] 2010. http://www.alegsa.com.ar/Dic/lenguaje%20de%20programacion.php.

—. 2010. Definicion de SGBD. [En línea] 2010. http://www.alegsa.com.ar/Dic/sgbd.php.

—. 2010. Servidor de Aplicaciones. [En línea] 2010. http://www.alegsa.com.ar/Dic/servidor%20de%20aplicaciones.php.

Alegsa, Leandro. 2010. Definicion de consulta. *DICCIONARIO DE INFORMÁTICA Y TECNOLOGÍA.* [En línea] 5 de diciembre de 2010. http://www.alegsa.com.ar/Dic/funcion.php.

Biithahh, Gas. 2011. Entornos de desarrollo integrados. [En línea] 2011. http://es.slideshare.net/GhaBiithahh/entornos-de-desarrollo-integrados.

Can, Carolina Novelo. 2010. Requisitos Funcionales. *Escarega, Campache.* [En línea] 6 de septiembre de 2010. http://es.scribd.com/doc/37187866/Requerimientos-funcionales-y-no-funcionales#scribd.

CASE, Herramientas. 2015. Herramientas CASE. [En línea] monografias.com, 2015. http://www.monografias.com/trabajos14/herramicase/herramicase.shtml#herr.

Cientificas, Series. EcuRed. *Series Cientificas.* [En línea] publicaciones.uci.cu/index.php. 2495.

Computer Systems Research . 2012. Computer Systems Research Group (CSRG). [En línea] 2012. http://csrg.inf.utfsm.cl/~jfuentes/_build/html/lectures/week7/lecture27.html.

Congreso HispaLinux 2000. Definicion RTF. [En línea] www.ibiblio.org/pub/linux/docs/LuCaS/Presentaciones/200002hispalinux/conf-04/04-html/formato_RTF.html.

Copyright. 2014. PDF Definicion. *MasAdelante .com.* [En línea] MasAdelante .com, 2014. http://www.masadelante.com/faqs/pdf.

Corporation. 2015. slideshare. [En línea] 2015. http://es.slideshare.net/JUANESTEFA/diseo-de-sistemas.

D'Ambrosio, Sergio. 2010. El Centro de Tesis. *Concepto de datos.* [En línea] © Monografias.com S.A., 2010. http://www.monografias.com/trabajos14/datos/datos.shtml.

Delgado, M.C. Gabino Estevez y Ing. Eduardo , Ochoa Hernández. 2001. La Coordinación de Innovación Educativa o por los Autores. [En línea] 11 de diciembre de 2001. http://dieumsnh.qfb.umich.mx/gesinfo/.

Desarrollo de Software Interasystem S.A. de C.V. 2015. Softmant. [En línea] Desarrollo de Software Interasystem S.A. de C.V., 2015. http://www.softmant.com/index.php/es/.

Diclib. 2015. Rutina. *Definicion de rutina.* [En línea] 2015. http://www.diclib.com/cgi-bin/d1.cgi?l=es&base=alkonaeconomia&page=showid&id=5593#ixzz49gGT3dia.

EcuRed. 1985. [En línea] 1985. http://www.ecured.cu/Gesti%C3%B3n_de_la_Informaci%C3%B3n.

Fabien Potencier, François Zaninotto. 2008. *Symfony la guía definitiva.* 2008.

Fiestas, Jhonattan. 2014. ElevenPaths. [En línea] 2014. [Citado el: 8 de Abril de 2016.] http://blog.elevenpaths.com/2014/09/qa-pruebas-para-asegurar-la-calidad-del.html.

Firefox, IE, Safari. 2011. [En línea] 2011. http://dev.sencha.com/deploy/ext-3.4.0/examples/.

Foundation, ITIL®. 2015. Metodologia y Herraminetas. [En línea] 2015. http://itilv3.osiatis.es/proceso_mejora_continua_servicios_TI/herramientas_metodologias.php.

Genbeta. 2014. Genbeta:dev. [En línea] 14 de julio de 2014. http://www.genbetadev.com/metodologias-de-programacion/patrones-de-diseno-que-son-y-por-que-debes-usarlos.

Gil, Manuel Torres. Universidad de Almeria. *Universidad de Almeria.* [En línea] [Citado el: 20 de Abril de 2016.] http://indalog.ual.es/mtorres/LP/index.php?opcion=inicio.

Github. 2014. What is Symfony? [En línea] 2014. https://github.com/symfony/symfony.

Hernandes Barrio, Carmelio Jose. 2014. slideshare. [En línea] 2014. http://es.slideshare.net/carmeloh2/metodologa-open-up-39321348.

Hernandez, Leovigilda. 2013. MODELO DE IMPLEMENTACIÓN. [En línea] 1 de junio de 2013. [Citado el: 1 de abril de 2016.] http://ithleovi.blogspot.com/2013/06/unidad-5-modelo-deimplementacion-el.html.

Hosting, Module for. 2013. What is Apache HTTP Server? [En línea] 2013. http://www.modulehosting.com/apache.html.

IEEE, Estándar. Departamento de Lenguajes y Sistemas Informáticos. [En línea] [Citado el: 24 de abril de 2016.]

Jojooa . 2014. tecnología, marketing y crm. [En línea] 2014. https://sites.google.com/site/jojooa/analisis-de-sistemas/definicion-de-actor-que-es-un-actor.

Larman, Craig. 2004. *UML Y PATRONES.* 2004.

—. 2004. *UML Y PATRONES.* 2004.

—. 2004. *UML Y PATRONES.* 2004.

—. 2004. *UML Y PATRONES .* 2004.

Martinez, Rafael. 2010. Sobre PostgreSQL. [En línea] 2 de octubre de 2010. http://www.postgresql.org.es/sobre_postgresql.

Mozilla Developer Network. 2014. JavaScript. [En línea] 2014. https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Introducci%C3%B3n.

Multiplataforma. 2012. ¿Que denominamos Framework en Informática? [En línea] 2012. http://desarrollomovilmultiplataforma.blogspot.com/2012/08/aspectos-teoricos-framework.html.

Muñoz, David Ruiz. 2008. *Manual Estadisticas.* 2008.

Olivares, José Rolando Lafaurie. 2008. Sistema para la generación de reportes . [En línea] 2008. https://en.wikipedia.org/wiki/ActiveReports.

OMG, Object Management Group. 2005. Introduction to OMG's Unified Modeling Language™ (UML®). [En línea] junio de 2005. http://www.omg.org/gettingstarted/what_is_uml.htm.

Oracle Corporation. 2011. NetBeans Policies and Terms of Use. [En línea] 2011. https://netbeans.org/index_es.html.

Potencier. 2015. *LibroWeb.* 2015.

Pressman. 2009. Cap_13_Estrtegia_Prueba. 2009.

PRESSMAN, Roger. 2009. *Ingeniería del Software. Un enfoque práctico.* s.l. : McGraw-Hill/Interamericana, 2009.

Pressman, Roger. 2009. *Sistema Ingeneria de Software.* 2009.

RCCI. 2014. Gereport. [En línea] Rev cuba cienc informat vol.8 no.4 La Habana, oct-dic de 2014. http://scielo.sld.cu/scielo.php?pid=S2227-18992014000400007&script=sci_arttext.

Reporte, Definición de. Definición de Reporte. *Definición de Reporte.* [En línea] http://definicion.mx/reporte/.

Rodríguez, Saúl Cuesta. 2014. SG Buzz. [En línea] 2014. [Citado el: 12 de Febrero de 2016.] http://sg.com.mx/.

—. 2014. SG Buzz. [En línea] 2014. [Citado el: 12 de Febrero de 2016.] http://sg.com.mx/.

Shimabukuro, Oscar. 2015. Marketin Digital (Smartec). [En línea] 26 de junio de 2015. http://www.smartec.la/blog/la-importancia-de-los-reportes-para-los-clientes.

—. 2015. Marketin Digital (Smartec). [En línea] 26 de junio de 2015. http://www.smartec.la/blog/la-importancia-de-los-reportes-para-los-clientes.

Softwaresea. 2014. Visual Paradigm for UML estándar. [En línea] 2014. http://www.ie.inf.uc3m.es/grupo/docencia/reglada/Is1y2/PracticaVP.pdf.

Sommerville, Ian. 2005. *Ingenieria de software. 7ma edición.* Madrid : Person Educación, 2005.

—. 2005. *Ingenieria de software. 7ma edición.* Madrid : Person Educación, 2005.

Sparx. 2007. Modelo Caso Uso. [En línea] 2007. http://www.sparxsystems.com.ar/resources/tutorial/use_case_model.html.

Synergix. 2008. Tecnología y Synergix. *Modelo de dominio.* [En línea] 10 de julio de 2008. https://synergix.wordpress.com/2008/07/10/modelo-de-dominio.

uml-diagrams.org. 2009. The Unified Modeling Language. [En línea] 2009. uml-diagrams.org.

Valle Laborde, Mabel, Alfonso Valdés, Javier y Saballo López, Luis Ernesto. 2011. Repositorio Instituional, Tesis. [En línea] 2011. http://repositorio_institucional.uci.cu/jspui/handle/ident/TD_04379_11.

Venemedia. 2015. Definicion HTML. *Definicion HTML.* [En línea] 26 de enero de 2015. http://conceptodefinicion.de/html/.

Bibliografía

C*l* Albasanz, 14 Bis. EXES. *Manual de XML.* [En línea] Area de producion y desarrollo. http://www.mundolinux.info/que-es-xml.htm.

Achour, Mehdi. 2011. Manual de PHP. [En línea] 2011. http://php.net/manual/es/preface.php.

Aguila, Rolando Morales. 2013. Repositorio Instituional, Tesis. [En línea] 2013. http://repositorio_institucional.uci.cu/jspui/handle/ident/TD_07022_13.

Alegsa. 2010. Definición de lenguaje de programación. [En línea] 2010. http://www.alegsa.com.ar/Dic/lenguaje%20de%20programacion.php.

—. 2010. Definicion de SGBD. [En línea] 2010. http://www.alegsa.com.ar/Dic/sgbd.php.

—. 2010. Servidor de Aplicaciones. [En línea] 2010. http://www.alegsa.com.ar/Dic/servidor%20de%20aplicaciones.php.

Alegsa, Leandro. 2010. Definicion de consulta. *DICCIONARIO DE INFORMÁTICA Y TECNOLOGÍA.* [En línea] 5 de diciembre de 2010. http://www.alegsa.com.ar/Dic/funcion.php.

Biithahh, Gas. 2011. Entornos de desarrollo integrados. [En línea] 2011. http://es.slideshare.net/GhaBiithahh/entornos-de-desarrollo-integrados.

Can, Carolina Novelo. 2010. Requisitos Funcionales. *Escarega, Campache.* [En línea] 6 de septiembre de 2010. http://es.scribd.com/doc/37187866/Requerimientos-funcionales-y-no-funcionales#scribd.

CASE, Herramientas. 2015. Herramientas CASE. [En línea] monografias.com, 2015. http://www.monografias.com/trabajos14/herramicase/herramicase.shtml#herr.

Cientificas, Series. EcuRed. *Series Cientificas.* [En línea] publicaciones.uci.cu/index.php. 2495.

Computer Systems Research . 2012. Computer Systems Research Group (CSRG). [En línea] 2012. http://csrg.inf.utfsm.cl/~jfuentes/_build/html/lectures/week7/lecture27.html.

Congreso HispaLinux 2000. Definicion RTF. [En línea] www.ibiblio.org/pub/linux/docs/LuCaS/Presentaciones/200002hispalinux/conf-04/04-html/formato_RTF.html.

Copyright. 2014. PDF Definicion. *MasAdelante .com.* [En línea] MasAdelante .com, 2014. http://www.masadelante.com/faqs/pdf.

Corporation. 2015. slideshare. [En línea] 2015. http://es.slideshare.net/JUANESTEFA/diseo-de-sistemas.

D'Ambrosio, Sergio. 2010. El Centro de Tesis. *Concepto de datos.* [En línea] © Monografias.com S.A., 2010. http://www.monografias.com/trabajos14/datos/datos.shtml.

Delgado, M.C. Gabino Estevez y Ing. Eduardo , Ochoa Hernández. 2001. La Coordinación de Innovación Educativa o por los Autores. [En línea] 11 de diciembre de 2001. http://dieumsnh.qfb.umich.mx/gesinfo/.

Desarrollo de Software Interasystem S.A. de C.V. 2015. Softmant. [En línea] Desarrollo de Software Interasystem S.A. de C.V., 2015. http://www.softmant.com/index.php/es/.

Diclib. 2015. Rutina. *Definicion de rutina.* [En línea] 2015. http://www.diclib.com/cgi-bin/d1.cgi?l=es&base=alkonaeconomia&page=showid&id=5593#ixzz49gGT3dia.

EcuRed. 1985. [En línea] 1985. http://www.ecured.cu/Gesti%C3%B3n_de_la_Informaci%C3%B3n.

Fabien Potencier, François Zaninotto. 2008. *Symfony la guía definitiva.* 2008.

Fiestas, Jhonattan. 2014. ElevenPaths. [En línea] 2014. [Citado el: 8 de Abril de 2016.] http://blog.elevenpaths.com/2014/09/qa-pruebas-para-asegurar-la-calidad-del.html.

Firefox, IE, Safari. 2011. [En línea] 2011. http://dev.sencha.com/deploy/ext-3.4.0/examples/.

Foundation, ITIL®. 2015. Metodologia y Herraminetas. [En línea] 2015. http://itilv3.osiatis.es/proceso_mejora_continua_servicios_TI/herramientas_metodologias.php.

Genbeta. 2014. Genbeta:dev. [En línea] 14 de julio de 2014. http://www.genbetadev.com/metodologias-de-programacion/patrones-de-diseno-que-son-y-por-que-debes-usarlos.

Gil, Manuel Torres. Universidad de Almeria. *Universidad de Almeria.* [En línea] [Citado el: 20 de Abril de 2016.] http://indalog.ual.es/mtorres/LP/index.php?opcion=inicio.

Github. 2014. What is Symfony? [En línea] 2014. https://github.com/symfony/symfony.

Hernandes Barrio, Carmelio Jose. 2014. slideshare. [En línea] 2014. http://es.slideshare.net/carmeloh2/metodologa-open-up-39321348.

Hernandez, Leovigilda. 2013. MODELO DE IMPLEMENTACIÓN. [En línea] 1 de junio de 2013. [Citado el: 1 de abril de 2016.] http://ithleovi.blogspot.com/2013/06/unidad-5-modelo-deimplementacion-el.html.

Hosting, Module for. 2013. What is Apache HTTP Server? [En línea] 2013. http://www.modulehosting.com/apache.html.

IEEE, Estándar. Departamento de Lenguajes y Sistemas Informáticos. [En línea] [Citado el: 24 de abril de 2016.]

Jojooa . 2014. tecnología, marketing y crm. [En línea] 2014. https://sites.google.com/site/jojooa/analisis-de-sistemas/definicion-de-actor-que-es-un-actor.

Larman, Craig. 2004. *UML Y PATRONES.* 2004.

—. 2004. *UML Y PATRONES.* 2004.

—. 2004. *UML Y PATRONES.* 2004.

—. 2004. *UML Y PATRONES .* 2004.

Martinez, Rafael. 2010. Sobre PostgreSQL. [En línea] 2 de octubre de 2010. http://www.postgresql.org.es/sobre_postgresql.

Mozilla Developer Network. 2014. JavaScript. [En línea] 2014. https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Introducci%C3%B3n.

Muñoz, David Ruiz. 2008. *Manual Estadisticas.* 2008.

Olivares, José Rolando Lafaurie. 2008. Sistema para la generación de reportes . [En línea] 2008. https://en.wikipedia.org/wiki/ActiveReports.

OMG, Object Management Group. 2005. Introduction to OMG's Unified Modeling Language™ (UML®). [En línea] junio de 2005. http://www.omg.org/gettingstarted/what_is_uml.htm.

Oracle Corporation. 2011. NetBeans Policies and Terms of Use. [En línea] 2011. https://netbeans.org/index_es.html.

Potencier. 2015. *LibroWeb.* 2015.

Pressman. 2009. Cap_13_Estrtegia_Prueba. 2009.

PRESSMAN, Roger. 2009. *Ingeniería del Software. Un enfoque práctico.* s.l. : McGraw-Hill/Interamericana, 2009.

Pressman, Roger. 2009. *Sistema Ingeneria de Software.* 2009.

Reporte, Definición de. Definición de Reporte. *Definición de Reporte.* [En línea] http://definicion.mx/reporte/.

Rodríguez, Saúl Cuesta. 2014. SG Buzz. [En línea] 2014. [Citado el: 12 de Febrero de 2016.] http://sg.com.mx/.

—. 2014. SG Buzz. [En línea] 2014. [Citado el: 12 de Febrero de 2016.] http://sg.com.mx/.

Shimabukuro, Oscar. 2015. Marketin Digital (Smartec). [En línea] 26 de junio de 2015. http://www.smartec.la/blog/la-importancia-de-los-reportes-para-los-clientes.

—. 2015. Marketin Digital (Smartec). [En línea] 26 de junio de 2015. http://www.smartec.la/blog/la-importancia-de-los-reportes-para-los-clientes.

Software, Departamento Ingeniería y Gestión . 2014. 2014.

Softwaresea. 2014. Visual Paradigm for UML estándar. [En línea] 2014. http://www.ie.inf.uc3m.es/grupo/docencia/reglada/Is1y2/PracticaVP.pdf.

Sommerville, Ian. 2005. *Ingenieria de software. 7ma edición.* Madrid : Person Educación, 2005.

—. 2005. *Ingenieria de software. 7ma edición.* Madrid : Person Educación, 2005.

Sparx. 2007. Modelo Caso Uso. [En línea] 2007. http://www.sparxsystems.com.ar/resources/tutorial/use_case_model.html.

Synergix. 2008. Tecnología y Synergix. *Modelo de dominio.* [En línea] 10 de julio de 2008. https://synergix.wordpress.com/2008/07/10/modelo-de-dominio.

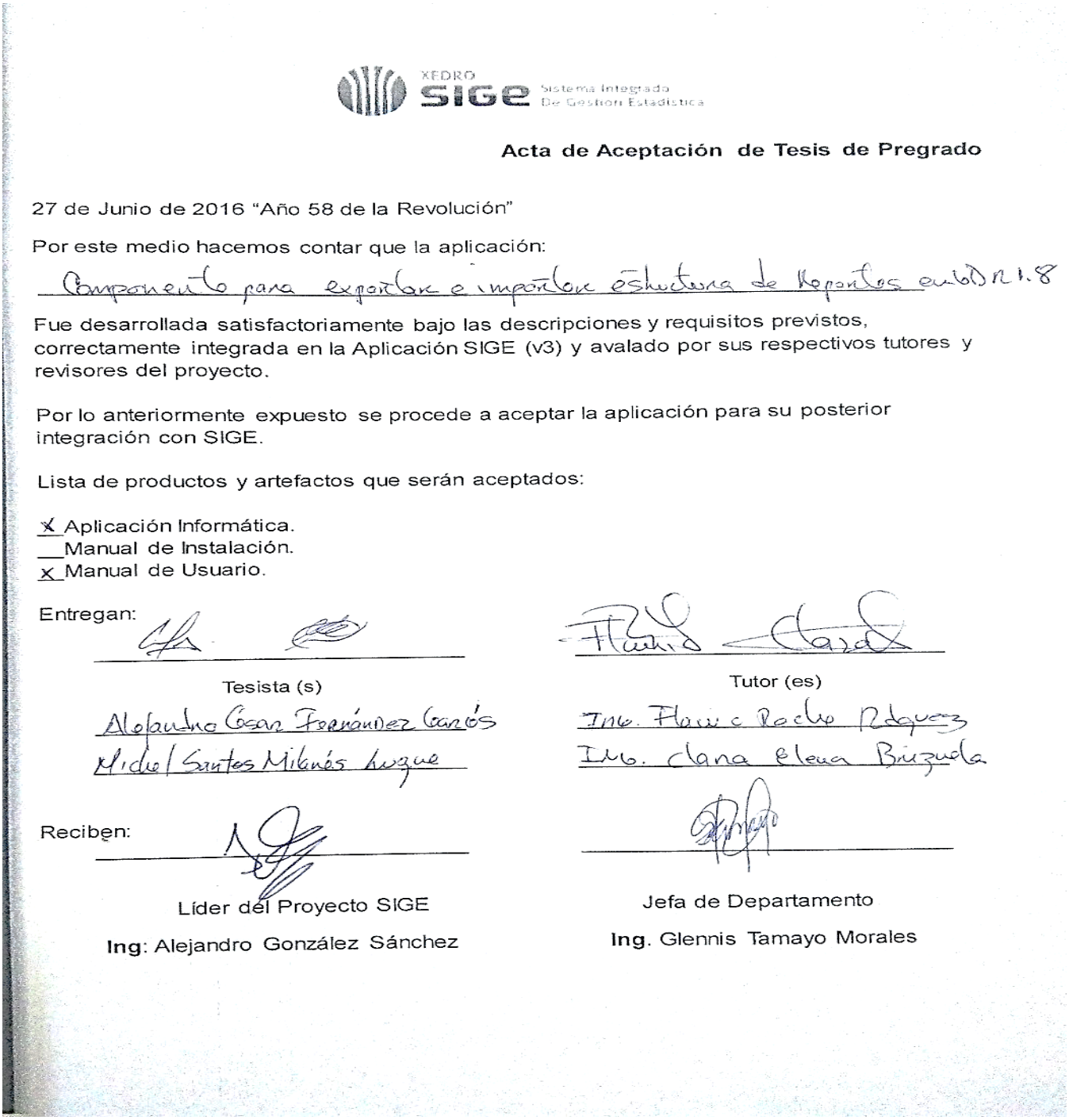
Taringa. Patrones Grasp. [En línea] http://www.taringa.net/post/apuntes-y-monografias/18339620/Patrones-de-Diseno-GRASP.html.

uml-diagrams.org. 2009. The Unified Modeling Language. [En línea] 2009. uml-diagrams.org.

Valle Laborde, Mabel, Alfonso Valdés, Javier y Saballo López, Luis Ernesto. 2011. Repositorio Instituional, Tesis. [En línea] 2011. http://repositorio_institucional.uci.cu/jspui/handle/ident/TD_04379_11.

Venemedia. 2015. Definicion HTML. *Definicion HTML.* [En línea] 26 de enero de 2015. http://conceptodefinicion.de/html/.

Anexos



XEDRO SIGE Sistema Integrado De Gestión Estadística

Acta de Aceptación de Tesis de Pregrado

27 de Junio de 2016 "Año 58 de la Revolución"

Por este medio hacemos contar que la aplicación:

Componente para exportar e importar estructura de Reportes en GDR 1.8

Fue desarrollada satisfactoriamente bajo las descripciones y requisitos previstos, correctamente integrada en la Aplicación SIGE (v3) y avalado por sus respectivos tutores y revisores del proyecto.

Por lo anteriormente expuesto se procede a aceptar la aplicación para su posterior integración con SIGE.

Lista de productos y artefactos que serán aceptados:

X Aplicación Informática.
__Manual de Instalación.
X Manual de Usuario.

Entregan:

Tesista (s)

Alejandro César Fernández Canós
Midel Santos Milanés Luque

Tutor (es)

Ing. Flavio Rocha Rdguez
Ing. Clara Elena Brizuela

Reciben:

Líder del Proyecto SIGE

Ing: Alejandro González Sánchez

Jefa de Departamento

Ing. Glennis Tamayo Morales

Figura. 24: Planilla de prueba de aceptación

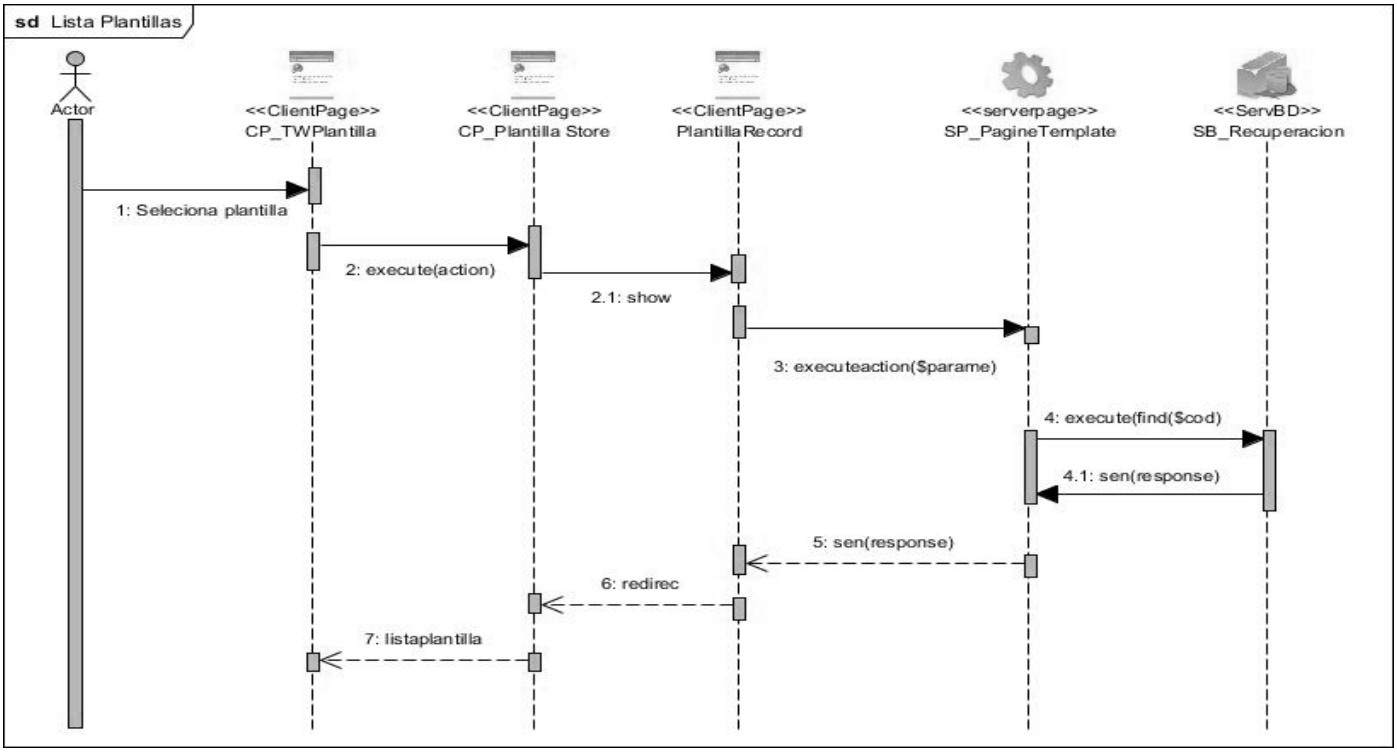


Figura. 25: Diagrama secuencia listar plantilla

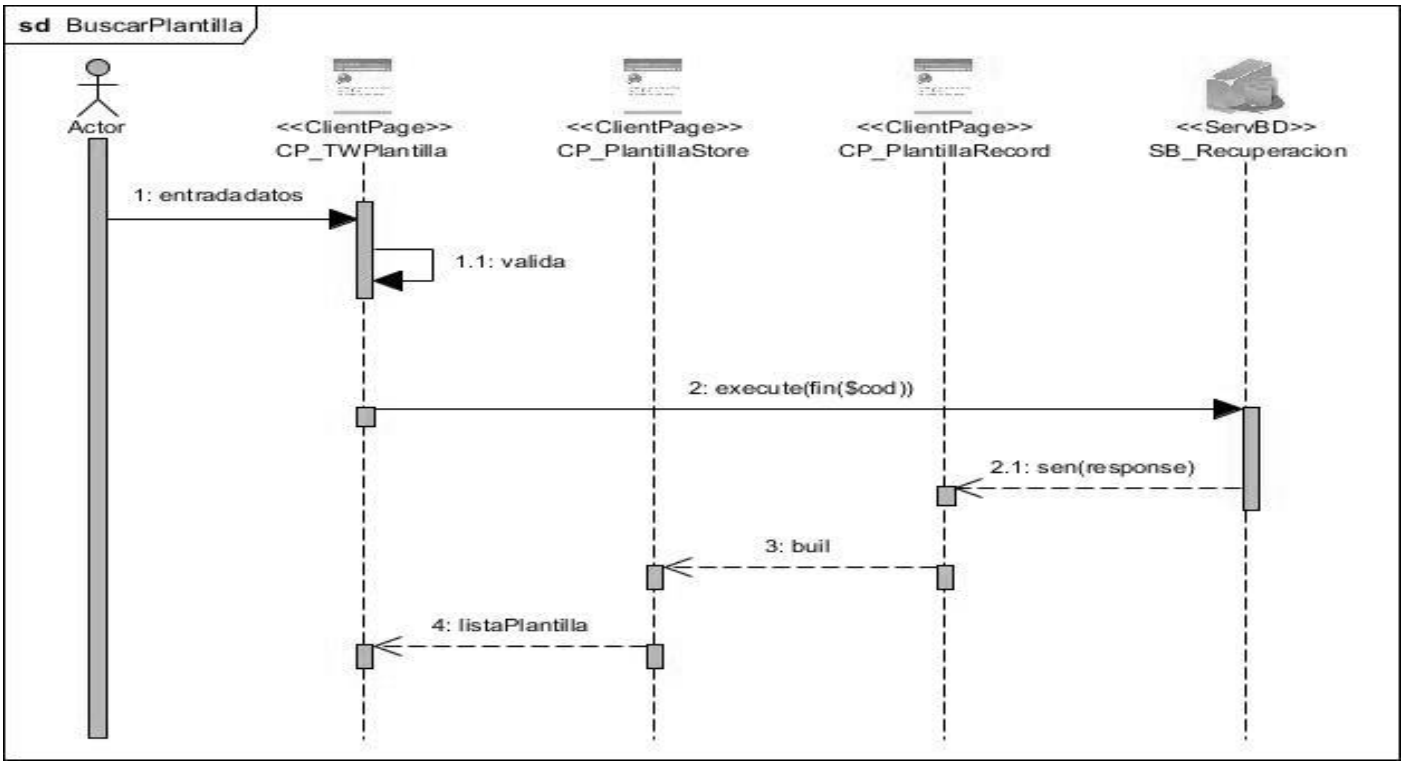


Figura. 26: Diagrama secuencia buscar plantilla

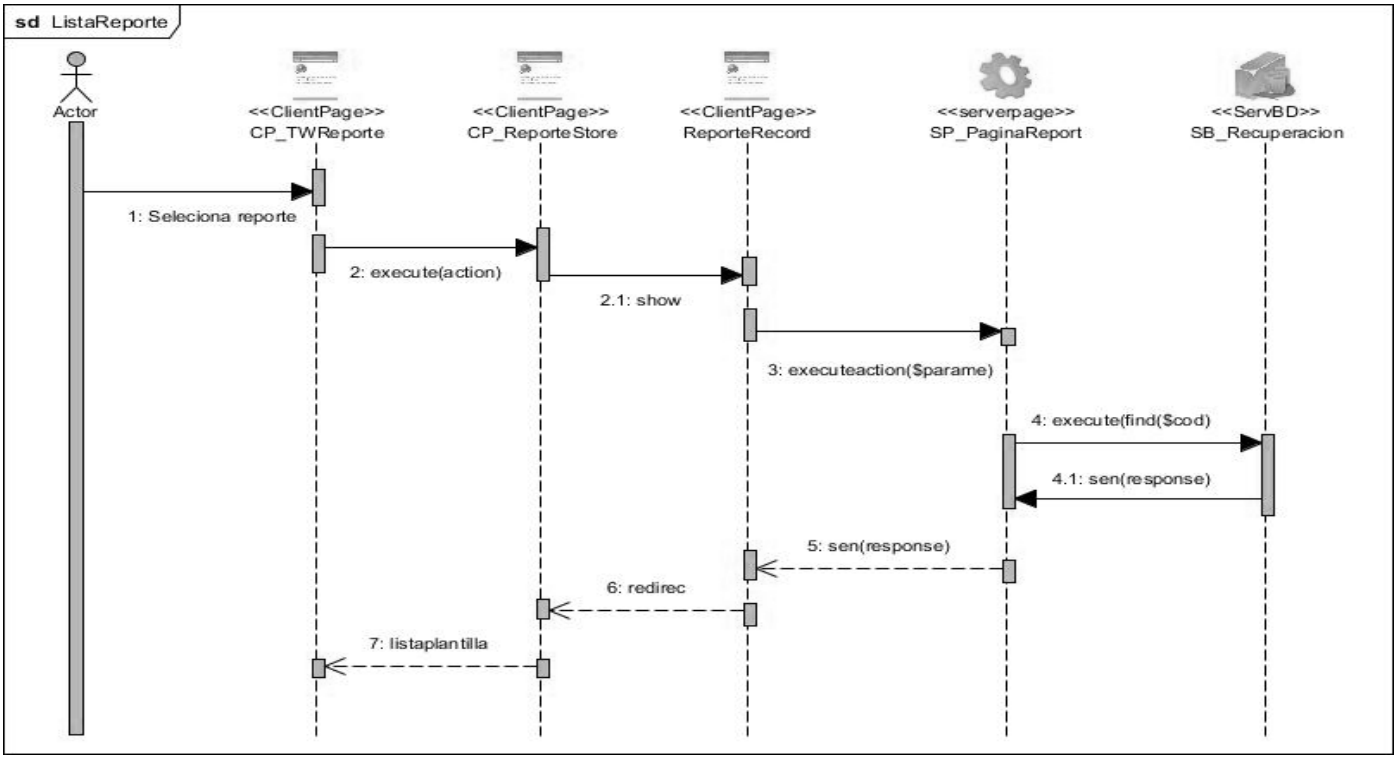


Figura. 27: Diagrama secuencia listar reporte

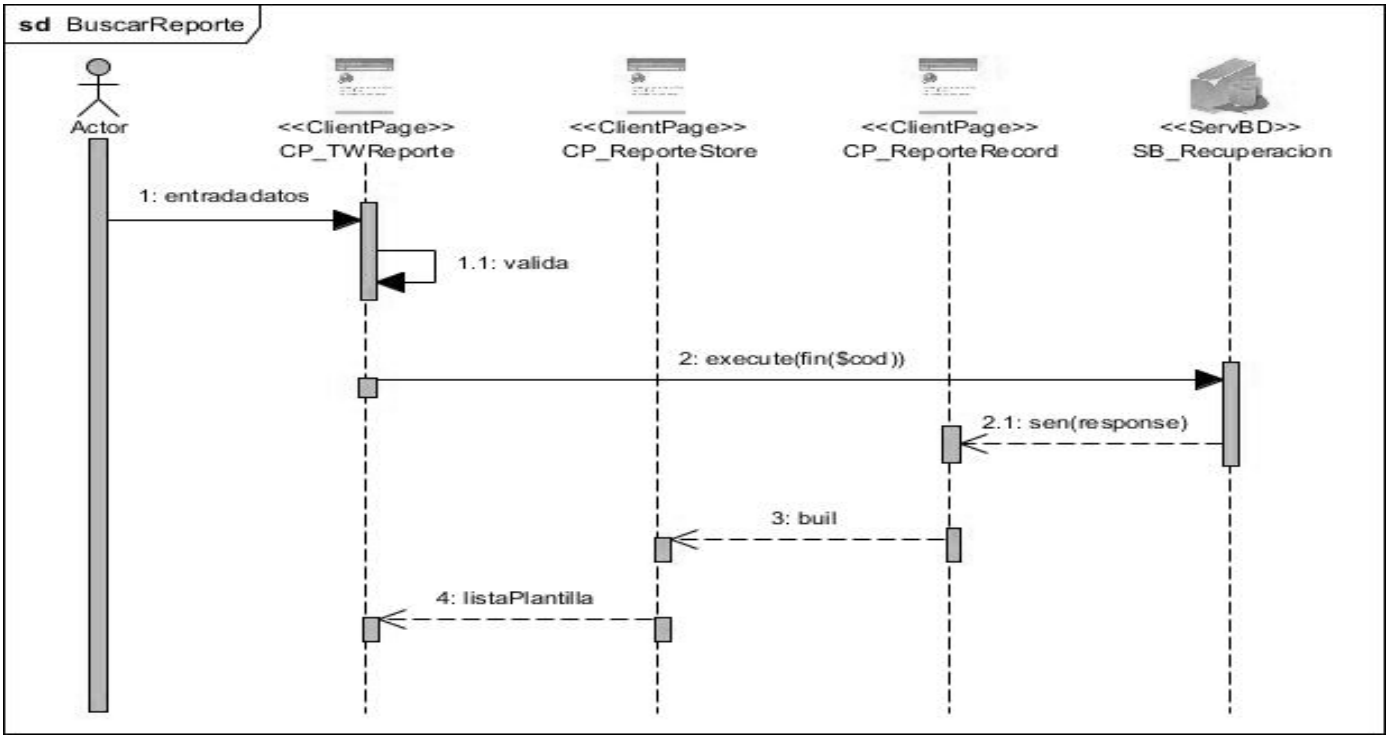


Figura. 28: Diagrama secuencia buscar reporte

Glosario de Términos

AJAX: (*Asynchronous Javascript And XML*), Javascript asíncrono y XML, es una técnica de desarrollo web para crear aplicaciones interactivas.

Apache: Es un software libre, servidor HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etcétera), Windows y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual.

BD: Base de datos.

CASE: (*Computer Aided Software Engineering*). Ingeniería de Software Asistida por Computación.

CU: Caso de Uso.

DATEC: Centro de Tecnologías de Gestión de Datos.

DOM: (*Document Object Model*), Modelo de Objetos del Documento o Modelo en Objetos para la representación de Documentos, es esencialmente una interfaz de programación de aplicaciones (API) que proporciona un conjunto estándar de objetos para representar documentos HTML y XML.

GRASP: (*General Responsibility Assignment Software Patterns*), son patrones generales de software para asignación de responsabilidades. Aunque se considera que más que patrones propiamente dichos, son una serie de "buenas prácticas" de aplicación recomendable en el diseño de software.

HTML: (*Hyper Text Markup Language*). Lenguaje de Marcado de Hipertexto, es el lenguaje de marcado predominante para la elaboración de páginas web.

HTTP: (*Hypertext Transfer Protocol*). Protocolo de transferencia de hipertexto usado en cada transacción de la World Wide Web.

JQuery: Biblioteca o framework de Javascript, creada inicialmente que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web.

MVC: Modelo Vista Controlador, es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

PHP: (*Hypertext Preprocessor*), es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas.

PostgreSQL: Es un sistema de gestión de base de datos relacional orientada a objetos y libre.

Propel: Es un ORM para PHP que facilita la labor de desarrollo de aplicaciones web, gracias a la capa que transforma el tratamiento del BD mediante objetos, con la que se puede recuperar, insertar y modificar datos.

TCP/IP: Protocolo de Control de Transmisión (TCP) y Protocolo de Internet (IP). El TCP/IP es la base de Internet, y sirve para enlazar computadoras que utilizan diferentes sistemas operativos.

URL: (*Uniform Resource Locator*). Localizador de Recurso Uniforme, dirección global de documentos y de otros recursos en la World Wide Web.

Windows: Es un sistema operativo con interfaz gráfica para computadoras personales propiedad de la empresa Microsoft. Windows es el sistema operativo más utilizado en el mundo.

XML: (*Extensible Markup Language*) Lenguaje de Marcas Extensible, es un metalenguaje extensible de etiquetas.