“Generador de reportes para la plataforma educativa Zera 2.0”

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

Autor: Rogelio Ausina Díaz

Tutores: Ing. Arcel Labrada Batista

Ing. Evelyn Pérez Rosa

Co-tutor: Ing. Gustavo Crespo Sánchez

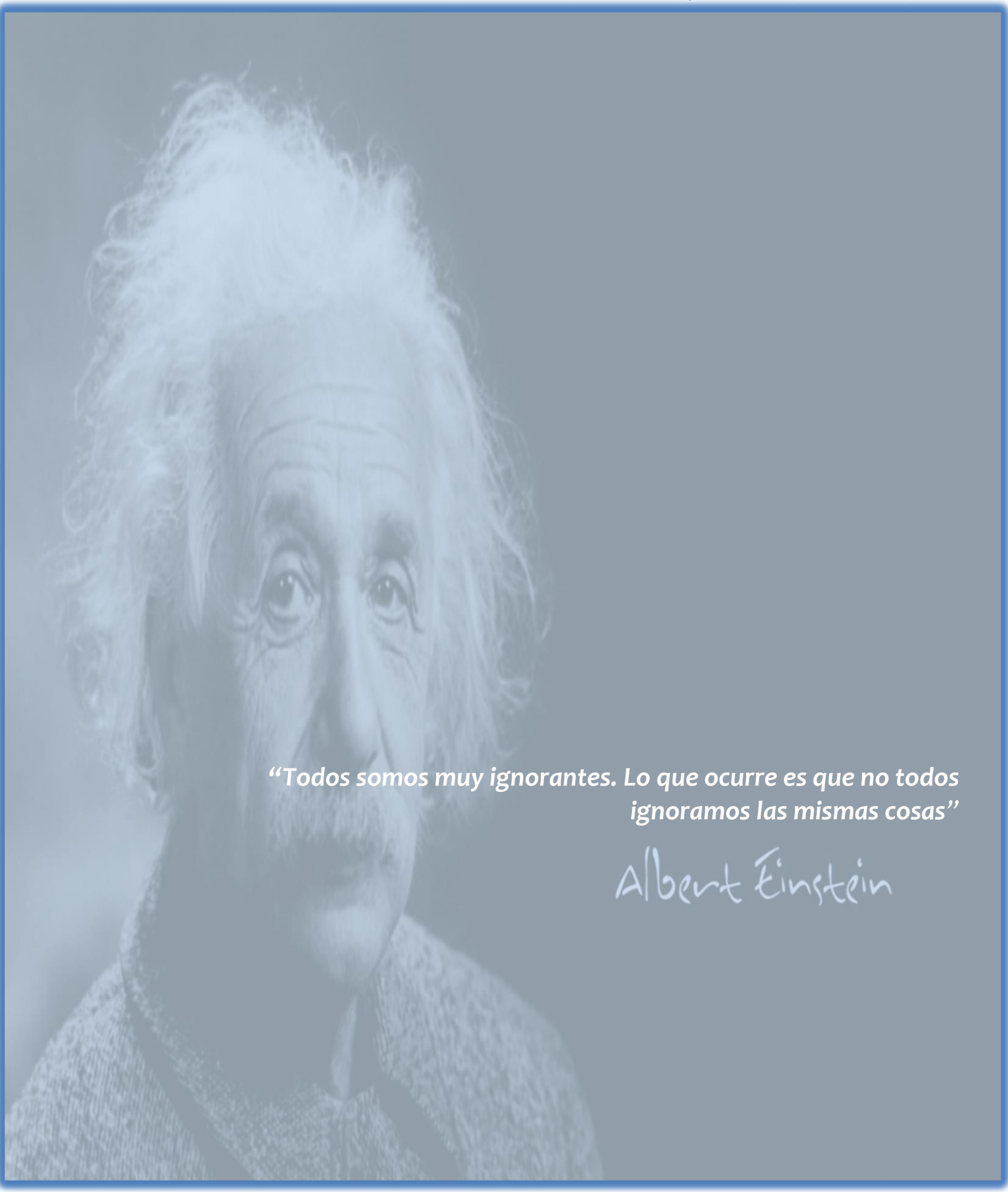
La Habana, Mayo de 2016.

“Año 58 de la Revolución”.

Declaración de Autoría

Declaro ser el autor del presente trabajo de diploma y otorgo 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 _____.

_____	_____
Firma del Autor	Firma del Tutor
Rogelio Ausina Díaz	Ing. Arcel Labrada Batista
_____	_____
Firma del Tutor	Firma del Co-tutor
Ing. Evelyn Pérez Rosa	Ing. Gustavo Crespo Sánchez



"Todos somos muy ignorantes. Lo que ocurre es que no todos ignoramos las mismas cosas"

Albert Einstein

Dedicatoria

A mi familia por estar ahí siempre y a mis amigos por ayudarme cuando me hizo falta.

Resumen

El desarrollo de la plataforma educativa Zera 2.0 no cuenta con un espacio que permita la consulta de la información que es generada por la interacción de los usuarios con la misma, principalmente los estudiantes. Para dar solución a esta problemática se realiza un estudio enfocado a los motores de reportes, así como los principales conceptos asociados al dominio del problema. Se efectúa el estudio y selección de las herramientas, lenguajes de modelado y metodologías de desarrollo a utilizar, para la construcción de objetivo general de la investigación, se tiene en cuenta las características que se ajustan a los requerimientos de la aplicación. Se desarrolló un módulo que permite la integración de un motor para generación de reportes dentro la plataforma educativa Zera 2.0, que brinda una solución al proceso de organización y visualización de la información que se genera dentro de la aplicación a través de reportes. Esta característica proveerá al docente de una herramienta muy útil para el control del progreso educativo y monitoreo de las actividades a través de los reportes. Se utilizaron para el desarrollo de dicha solución Symfony para el desarrollo del módulo y Jasper Report como motor para la generación de los reportes.

Palabras clases: educativa, generación, información, integración, plataforma, reportes.

Contenido

Introducción

A lo largo de la historia la forma en la que se almacena y se accede a la información ha sufrido grandes cambios, esta se encontraba en las bibliotecas de los monasterios en la edad media. A partir de la edad moderna, con el nacimiento de la imprenta, los libros comenzaron a fabricarse en serie y surgieron los periódicos. En el siglo pasado el desarrollo de los dispositivos de cómputo, unido al surgimiento y evolución del Internet, ha posibilitado que aumenten los medios para acceder y compartir la información. La expansión de los conocimientos y la introducción de la web contribuyeron a que el hombre interactúe de forma más rápida con mayor cantidad de datos. El desarrollo de las tecnologías y la globalización de la información contribuyeron al surgimiento de las Tecnologías de la Información y la Comunicación (TIC). La evolución y desarrollo de las TIC como red de comunicación global y el surgimiento y desarrollo de la web como servicio imprescindible para compartir información, creó un excelente espacio para la interacción del hombre con la información hipertextual[1].

Las TIC han contribuido al desarrollo en diferentes esferas de la sociedad entre las que se encuentra la educación, los cursos en línea son de los nuevos modelos de educación que han ganado auge en los últimos tiempos, los mismos potencian el ejercicio de la enseñanza, el aprendizaje de calidad y el desarrollo profesional de los docentes. Los Cursos Masivos Abiertos en Línea (en lo adelante MOOC) son cursos en línea dirigidos a un amplio número de participantes a través de Internet según el principio de educación abierta. Este entorno o contexto en el que se produce el aprendizaje, se basa en proporcionar recursos de apoyo a los que se tiene acceso durante el proceso de aprendizaje, y que por definición se basa en un acceso abierto y regido por el principio de flexibilidad en cuánto al cómo y cuándo accede el estudiante. Asimismo, en las plataformas MOOC el ambiente de aprendizaje es virtual, en cuanto los aprendizajes no se llevan a cabo en un lugar concreto y se reproduce virtualmente un aula o espacio que reúne las condiciones para optimizar el proceso de aprendizaje y la adquisición de contenidos, experiencias y procesos pedagógicos, a partir de actividades dialógicas que facilitan intercambios de conocimientos. Estos tipos de cursos generan gran cantidad de información acerca de los estudiantes, la misma pudiera ser utilizada en beneficio del proceso de enseñanza–aprendizaje.

[1] Estructura secuencial que permite crear, agregar, enlazar y compartir información de diversas fuentes por medio de enlaces asociativos.

Dado el auge que han experimentado este tipo de cursos, aparecieron iniciativas privadas, que con la colaboración de grandes expertos de cada materia, profesores de las más prestigiosas universidades de Estados Unidos, se convirtieron en grandes éxitos como Udacity y Coursera[2]. El caso de Sebastian Thrun[3] fue uno de los más mediáticos y eso contribuyó a una mayor expansión del número de iniciativas y seguidores de los MOOC. Edx, la segunda gran plataforma para MOOC del Instituto Tecnológico de Massachusetts (MIT) vino a confirmar la importancia de la tendencia de los cursos masivos y a reforzar la sensación de que es un movimiento que está llamado a cambiar la educación, puesto que las mejores universidades del mundo se han unido a él. (1)

En nuestro país, el desarrollo del software educativo para apoyar el proceso de enseñanza ha evolucionado teniendo en cuenta los avances tecnológicos en este sentido, por lo que la implementación de este tipo de software está en dependencia del nivel informatización del que dispone nuestra sociedad, en la actualidad se cuenta con una colección de software destinado a la enseñanza.

Desde sus inicios, la UCI dadas las condiciones y el conocimiento científico adquirido en las diferentes áreas, permitió que se crearan centros de producción-investigación donde se concentra personal capacitado. En la misma existen varios centros de desarrollo de software entre los que se encuentra el de Tecnologías para la Formación (FORTES) perteneciente a la Facultad 4, el cual se encarga de ofrecer servicios y productos informáticos para la educación, entre dichos productos se encuentra la plataforma para Cursos Masivos Abiertos en Línea (Zera 2.0), esta plataforma tiene sus bases en las colecciones cubanas de software educativo, una de las características fundamentales de dichas plataformas es el control y seguimiento de la trayectoria de los estudiantes.

Situación problemática:

La plataforma Zera 2.0 cuenta con varios módulos con los que el usuario interactúa, sin embargo, no es posible mantener una trazabilidad de las acciones de los usuarios dentro de la plataforma, lo que no permite conocer cómo interactúan o la evolución que han tenido dentro de la misma. No se tiene control de la participación de los estudiantes en

[2] Plataformas MOOC de mayor prestigio a nivel internacional.

[3] Sebastian Thrun es un innovador, educador, empresario y científico de la computación de Alemania. Fue director general y cofundador de Udacity. Antes de eso, fue vicepresidente en Google, y profesor de informática de la Universidad de Stanford.

los foros, el profesor no conoce la cantidad de estudiantes matriculados en un curso, ni el nombre de los mismos; ni las acciones realizadas por los estudiantes en dichos curso, actividades y foros. Se pierde así un espacio para consultar la información de forma rápida y precisa.

Por lo que se plantea como **problema de investigación**: Inexistencia de un espacio para consultar la información generada en la plataforma educativa Zera 2.0.

Por lo cual se define como **objeto de estudio** los motores para la generación de reportes y el **campo de acción** la generación de reportes dentro de la plataforma educativa Zera 2.0.

Determinándose como **objetivo general:** Desarrollar un módulo para la generación de reportes destinado a ser integrado en la plataforma educativa Zera 2.0.

A partir del objetivo general definido se derivan las siguientes preguntas científicas:

1. ¿Cuáles son los elementos teóricos que sustentan la investigación?
2. ¿Cuáles son las características que debe cumplir el módulo de reportes para la plataforma Zera 2.0?
3. ¿Cuáles son los elementos que intervienen en el diseño del módulo de reportes para la plataforma Zera 2.0?
4. ¿El módulo desarrollado para la plataforma Zera 2.0, cumple con las funcionalidades establecidas?

Para dar respuesta a las preguntas científicas, la investigación debe enmarcarse en relación con las siguientes **tareas:**

1. Asimilación de los principales conceptos y tecnologías que se requieren para el desarrollo del módulo.
2. Elaboración del estado del arte de la tesis.
3. Identificación y especificación de los requisitos funcionales y no funcionales.
4. Estudio y definición de la metodología que se seleccionará para el desarrollo de software.
5. Descripción de las características del módulo.
6. Estudio de las herramientas para implementación del módulo.
7. Implementación de las funcionalidades del módulo.
8. Realización de las pruebas funcionales al módulo.

Al finalizar el presente trabajo de diploma, se espera que el Centro FORTES pueda contar con:

1. Reporte de la investigación (memoria descriptiva).
2. Un módulo de generación de reportes para la plataforma Zera 2.0.

Para la realización de la presente investigación, se hace necesaria la utilización de los siguientes **métodos científicos:**

Métodos teóricos (2)**:**

1. Histórico-Lógico: Se utilizó este método para realizar el estudio del estado del arte, se investigó acerca de otras aplicaciones o herramientas similares, así como los lenguajes y metodologías de desarrollo existentes.
2. Analítico-Sintético: Permitió analizar conceptos y teorías relacionadas con el objeto de estudio. Se empleó en la búsqueda de las tecnologías que se utilizarán para el desarrollo de la propuesta y en la revisión bibliográfica.

Métodos empíricos:

1. Entrevista: Facilitó mediante encuentros con los trabajadores del centro, entender todo el proceso del negocio, el funcionamiento y la estructura de los procesos actuales, además de las dificultades existentes, de esta forma se obtuvo la mayor cantidad de información posible, para determinar las funcionalidades a implementar.

Estructura del documento:

El contenido del presente trabajo está estructurado en tres capítulos distribuidos de la siguiente manera:

Capítulo 1. “Fundamentación teórica”: Contiene la fundamentación teórica de la investigación, se exponen las herramientas, el lenguaje y las metodologías que se van a seleccionar para realizar el Análisis, Diseño e Implementación del Módulo de reportes, se justifica su selección, así como las tendencias actuales en el mundo y en nuestro país, en las plataformas para la gestión del aprendizaje, en conjunto con los conceptos relacionados con el tema.

Capítulo 2. “Descripción de la solución propuesta”: Se efectúa la descripción de la posible solución. Se realiza el levantamiento de los requisitos funcionales y no funcionales y se obtiene su principal artefacto, se modelan las Historias de Usuario, se especifica el diseño y la arquitectura que formará parte de la misma, se detalla el posible funcionamiento del sistema a desarrollar, así como los elementos que lo integran.

Capítulo 3. “Implementación y pruebas”: En este capítulo se exponen los elementos fundamentales del desarrollo ágil del software. Se representa la vista estática y la distribución física del sistema.

Capítulo 1: Fundamentación Teórica

1.1 Introducción

En el presente capítulo se abordarán elementos teóricos relacionados con la investigación y el tema a desarrollar. Se profundizará en la descripción de varios sistemas que gestionan reportes se mencionan algunas de sus principales características y funcionalidades, se destacan también algunas definiciones que se estimaron convenientes para un mejor entendimiento de la posible propuesta de solución. Se seleccionará la metodología y se argumentará el porqué de la misma, así como la selección de las herramientas y tecnologías que se emplearán para una posible propuesta de solución.

1.2 Conceptos asociados al dominio del problema

1.2.1 Generador de Reportes

Los generadores de reportes son herramientas complementarias de los sistemas de información. Por medio de estos se realizan consultas a la base de datos del sistema para obtener información en forma de reporte, además de realizar consultas a las base de datos a través de un lenguaje transparente al usuario, permiten visualizar la información de manera estructurada y/o resumida en un diseño atractivo y fácil de interpretar, permiten además acoplarse a otras técnicas de análisis. En el uso de los generadores de reportes, siempre se tienen en cuenta los sistemas gestores de bases de datos soportados (portabilidad del sistema), y la forma en que se acoplaría a los sistemas y/o aplicaciones que lo usan (integración con aplicaciones externas). (3)

1.2.2 Rasgos generales del funcionamiento de los generadores de reportes

Los generadores de reportes de manera estándar se caracterizan por estar compuestos por dos elementos básicos, un editor de informes y un motor que los genere. Las principales actividades o tareas que debe realizar un generador de reportes son las siguientes:

1. Conformar el diseño de los reportes.
2. Luego de obtener los datos se genera en el formato requerido el informe final de acuerdo a lo diseñado previamente.

1.3 Características de los generadores de reportes

Actualmente existen diversos sistemas de generación de reportes que presentan problemas con el dinamismo a la hora de brindar la información necesaria. Para este trabajo las herramientas son elaboradas con vista a generar un tipo de reporte específico en un tiempo de desarrollo que obliga al usuario a redefinir los requisitos cada vez necesite añadir una nueva funcionalidad. Existen diferentes motores de reportes, para seleccionar el que se utilizará en el presente trabajo se realizó un estudio sobre algunos motores. Entre los cuales se encuentran:

1. Jasper Reports
2. PHP Reports
3. Crystal Reports
4. Active Reports
5. SQL Server Reporting Services

A continuación se explican las principales características de cada uno de ellos.

1.3.1 Jasper Reports

Es el motor de reportes más usado actualmente en el mundo del Código Abierto (Open source), puede ser embebido en todo tipo de aplicaciones, desde las que generan reportes a partir de una plantilla o modelo predeterminado hasta las que brindan más libertad al usuario para diseñar sus propios reportes y ejecutar otras operaciones complejas. Es una biblioteca implementada completamente en Java la cual brinda el máximo nivel de portabilidad, con un amplio y expandible grupo de posibles fuentes de datos. Posee además una abundante variedad de formatos de salida, importación, así como una gran comunidad mundial que mantiene y desarrolla la biblioteca. Genera informes para formatos de impresión predeterminados existentes o para reportes continuos a ser visualizados en la web, los reportes pueden ser exportados a formatos como: PDF, XML, HTML, CSV, XLS, RTF, TXT. A través de los sub-reportes es posible manipular y diseñar reportes con un diseño altamente complejo. Soporta a la misma vez varios orígenes de datos incluso aunque sean de tipos distintos. Es posible pasar parámetros desde una aplicación a Jasper Report, esto es muy simple de implementar y brinda una herramienta muy poderosa que permite clasificar, restringir o mejorar los datos que son enviados al usuario basándose en las condiciones de tiempo de ejecución.

La utilización de esta herramienta permite mejorar la gestión de la empresa mediante la creación y gestión de informes. Con una solución de este tipo, se dispone en el tiempo deseado de los datos indispensables para la gestión eficaz de un departamento. También pueden seguirse fácilmente los resultados obtenidos y la manera en que la actividad progresa en función de los objetivos fijados. Pueden descubrirse las tendencias y los factores claves que afectan a la actividad y a los resultados de la empresa. (4)

1.3.2 Php Reports

Php Report es un motor de generación de reportes para PHP, es software libre y provee un alto nivel de productividad en este tipo de aplicaciones. Un reporte en Php Report está dividido en capas que contienen unas a otras. De esta forma un reporte sería un documento, contiene un encabezado, un pie y está formado por páginas, las que a su vez cuentan con un encabezado, un pie de página y puede contener grupos anidados en los que se cargarán los campos obtenidos de la consulta a la base de datos. (5)

Algunas de sus características son:

1. Muestra informes de cualquier fuente de datos la misma puede exportarse como datos (SQL, MongoDB, PHP, etc.)
2. Los reportes pueden ser exportados en HTML, XML, CSV, XLS, JSON, texto plano, o en su propio formato personalizado.
3. Permite añadir parámetros personalizables para un informe (por ejemplo, rango de fechas, palabras clave, etc.)
4. Soporte para gráficos y tablas con la API de visualización de datos de Google.
5. Cambiar fácilmente entre entornos de bases de datos (por ejemplo, la producción, el almacenamiento intermedio, y dev).
6. Crear paneles que combinan varios informes en una sola página.
7. Totalmente ampliable y personalizable.

1.3.3 SQL Server Reporting Services

SQL Server es una plataforma de elaboración de informes basada en servidor que se puede utilizar para crear y administrar informes tabulares, matriciales, de gráficos y de formato libre con datos extraídos de orígenes de datos relacionales y multidimensionales. Se pueden visualizar y administrar mediante una conexión basada en la web. Entre sus características principales están las siguientes:

1. Servidor de informes que aloja y procesa informes en diversos formatos: HTML, PDF, TIFF, Excel, CSV, etc.
2. Los informes incluyen características interactivas basadas en la Web API que permite a los programadores integrar o extender procesamiento de datos e informes en aplicaciones personalizadas.
3. La arquitectura de desarrollo y tiempo de ejecución se ha creado con un diseño modular para admitir extensiones de terceros y posibilidades de integración (*Microsoft Corporation*).
4. Los usuarios pueden enrutar directamente los trabajos de impresión, sin necesidad de exportarlos antes. (6)

1.3.4 Crystal Reports

Crystal Reports es un producto de alta tecnología, además de ser una herramienta potente, es fácil de usar para el diseño y generación de reportes a partir de datos almacenados en una base de datos u otra fuente de información. Crystal Reports es una solución de informes poderosa, dinámica y procesable que ayuda a diseñar, explorar, visualizar y entregar informes por medio de la web. Permite transformar rápidamente cualquier fuente de datos en contenido interactivo, integrar estrechamente capacidades de diseño, modificación y visualización en aplicaciones .NET, Java o COM, y además permite a los usuarios finales acceder e interactuar con los reportes a través de portales Web, dispositivos móviles y documentos de Microsoft Office. (7)

1.3.5 Active Reports

Active Reports es un componente de informes .NET, entre las características claves de este componente figuran:

1. El diseñador de informes de Active Reports se integra perfectamente en los entornos de desarrollo Visual Studio .NET. Una vez instalado el producto en el equipo del desarrollador, la adición de un informe a un proyecto es tan fácil como añadir una clase o un formulario.
2. Los informes se crean dentro de Visual Studio .NET y se compilan directamente en el ejecutable. Por consiguiente, los ensamblados pueden distribuirse mediante despliegue Xcopy o colocarse en la cache de ensamblados global (*Global Assembly Cache*: GAC). El modelo de objetos proporciona el motor de generación de informes como un único ensamblado administrado con nombre

seguro. No es necesario realizar ninguna otra configuración adicional en el servidor o el equipo cliente.

3. Dado que Active Reports es totalmente administrado, no presenta dependencias de aplicaciones de terceros.
4. Active Reports incluye filtros para la exportación a formatos habituales como Adobe PDF, Microsoft Excel, RTF, HTML, texto y TIFF tanto en aplicaciones Windows como en aplicaciones basadas en la Web.
5. Con Active Reports se incluye un control de gráficos que admite tipos comunes de gráficos en 2D y 3D y proporciona prestaciones gráficas avanzadas, además de exportación nativa a diversos formatos de imagen.
6. Active Reports incluye un control de visor de informes que admite zoom y vista previa de informes, múltiples fichas para visualización de hipervínculos, vistas divididas y de múltiples páginas, un panel de índice de contenidos, miniaturas, búsquedas de texto, anotaciones y personalización de barra de herramientas.
7. Active Reports Professional incluye un control de diseñador de informes de usuario final (*End-User Report Designer*) libre de derechos que le permite alojar el diseñador de informes en sus propias aplicaciones con el fin de ofrecer a los usuarios finales capacidad para crear y modificar informes. (8)

1.3.6 Selección del generador de reportes

Son varios los generadores de reportes que existen, para seleccionar el que se utilizará hay que tener en cuenta las necesidades de los usuarios, así como la arquitectura de la que se hará uso. La siguiente tabla de comparación entre los diferentes motores de reportes tendrá en cuenta pautas como formatos en los que exporta, fuentes de datos que soporta, si es multiplataforma y si es libre o no.

Generadores	**Libre**	**Sistema operative**	**Formatos**	**Fuente de datos**
Jasper Reports	Si	Multiplataforma	HTML, PDF, Excel, CSV, ODT, RTF, XLS	postgresql, mysql, oracle, mssql, odbc, interbase,

				informix, xml, csv
Php Reports	Si	Multiplataforma	HTML, PDF, Excel, CSV	postgresql, mysql, oracle, mssql, odbc, interbase, Informix
SQL Server	No	Windows	HTML, PDF, Excel, CSV	postgresql, mysql, oracle, mssql, odbc, interbase, Informix
Crystal Reports	No	Windows	RTF, PDF, Excel, HTML, TIFF	postgresql, mysql, oracle, mssql, odbc, interbase, Informix
Active Reports	No	Windows	HTML, PDF, TIFF, Excel, CSV	postgresql, mysql, oracle, mssql, odbc, interbase, Informix

Tabla 1: Comparación entre los diferentes generadores de reportes.

Como resultado de la comparación se decide escoger el motor de reportes Jasper Reports, ya que es de código abierto, multiplataforma, permite exportar en una mayor cantidad de formatos que los restantes motores de reportes, además cuenta con un

mayor rango de fuentes de datos soportadas, permite generar sub-reportes para facilitar el diseño, cuenta con una amplia comunidad y es el motor de reportes más usado en el marco de software libre. Al estar sujeto el sistema a la creación de múltiples reportes es recomendable utilizar un motor de reportes dinámico, en este caso Jasper Report satisface dichas necesidades.

1.3.7 SDR 1.0

El Servidor Dinámico de Reportes (SDRv1.0) es un software que permite de manera eficiente la gestión, compilación, publicación y exportación de reportes. El principio del SDRv1.0 es brindar una capa de servicios que facilita su integración con cualquier otro sistema, web o de escritorio, con independencia de sus plataformas de desarrollo. A través de este los clientes pueden generar reportes en una gran cantidad de formatos sobre múltiples fuentes de datos. Estos reportes se mantendrán almacenados en el servidor, con ello se logra mejores tiempos de respuesta y acceso al historial de los reportes. De forma general se pueden presentar las principales características y funcionalidades del sistema de la siguiente manera:

1. Permite la generación de reportes en múltiples formatos: PDF, XML, HTML, CSV, XLS, XLSX, RTF, JPG, ODT, PPTX, DOCX, ODS y XML4SWF, a partir de diferentes fuentes de datos.
2. Las conexiones a las fuentes de datos de los reportes se limitan solo por el soporte de los API JDBC (*Java Data Base Connection*) de Java.
3. Permite la generación de sub-reportes.
4. Visualiza las vistas previas de los reportes.
5. Posibilita la selección de estilos de estructura para un grupo de reportes.
6. No limita el tamaño del diseño del reporte para la generación de reportes.
7. Se encarga de procesar peticiones de aplicaciones web o de escritorio a través de servicios REST.
8. Constituye un sistema multiplataforma.
9. Utiliza un gestor de base de datos, que puede ser PostgreSQL, SQLite, MySQL o SQL, para persistir la información del servidor y los datos de los reportes.
10. Implementa un conjunto de políticas de seguridad para el acceso a los recursos y servicios a varios niveles.
11. Permite la programación de tareas para la exportación automática de reportes en un momento específico, y su envío por correo electrónico o por FTP.

1.4 Metodologías y herramientas de desarrollo.

Las metodologías de la investigación constituyen un medio indispensable para dirigir la utilización de herramientas teórico-prácticas en la solución de problemas mediante el método científico. Estas son de vital importancia en el desarrollo de cualquier proyecto, aunque las mismas no aseguran del todo obtener los resultados esperados. Esto se debe principalmente a la gran variedad de metodologías que existen en la actualidad. Son muchos los factores a tener en cuenta para la selección de la misma, para que esta se ajuste al contexto de desarrollo del proyecto, y facilite en vez de entorpecer, para alcanzar los resultados esperados. (9)

Cada metodología de desarrollo tiene más o menos su propio enfoque a la hora de aplicarla a un proyecto de desarrollo de software. Pero todas ellas se basan en una serie de enfoques generalistas. (10)

Entre las metodologías de desarrollo se encuentran:

Tradicionales:

1. RUP (*Rational Unified Process*)
2. MSF (*Microsoft Solution Framework*)
3. Win-Win Spiral Model
4. Iconix

Ágiles:

1. XP *(eXtreme Programming*)
2. Scrum
3. Crystal
4. DSDM (*Dynamic Systems Development Method*)
5. FDD (*Feature Deriven Development*)
6. Extreme Modeling
7. AUP (*Agile Unified Process*)

En la siguiente tabla se podrán apreciar las principales diferencias entre las metodologías ágiles y tradicionales:

Metodologías ágiles	Metodologías tradicionales
Menos énfasis en la arquitectura del software	La arquitectura del software es esencial y se expresa mediante modelos
No existe contrato tradicional o al menos es bastante flexible	Existe un contrato prefijado
Pocos roles	Más roles
Basadas en heurísticas provenientes de prácticas de producción de código	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo
Grupos pequeños (<10 integrantes) que trabajan en el mismo sitio	Grupos grandes y posiblemente distribuidos
Pocos artefactos	Más artefactos
Especialmente preparados para cambios durante el proyecto	Cierta resistencia a los cambios
Impuestas internamente (por el equipo)	Impuestas externamente
El cliente es parte del equipo de desarrollo	El cliente interactúa con el equipo de desarrollo mediante reuniones
Proceso menos controlado, con pocos principios	Proceso mucho más controlado, con numerosas políticas/normas

Tabla 2. Comparación entre las metodologías ágiles y tradicionales.

1.4.1 Selección de metodología

Las metodologías de desarrollo ayudan a obtener un software que responde y documenta los requerimientos de los distintos clientes. Para el desarrollo de este componente se hizo un estudio para seleccionar cual se adaptaría mejor al desarrollo de la propuesta de solución. Para la selección de la metodología de desarrollo a utilizar, se tuvieron en cuenta las metodologías ágiles ya que éstas entregan productos en un corto período de tiempo, realizan un mejoramiento constante de la calidad del software,

detectan errores, permiten conocer el estado actual del proyecto y mantienen un control del mismo por parte del cliente y el equipo de desarrollo, por lo que se ajusta de mejor manera al producto que se desea desarrollar.

1.4.2 eXtreme Programing (XP)

La programación extrema es una perspectiva deliberada y disciplinada al desarrollo de software. Es una metodología ligera de desarrollo de software basada en la simplicidad, la comunicación, la retroalimentación o reutilización del código desarrollado. Se caracteriza por ser una metodología ágil, ya que está diseñada para entregar el software que los clientes necesitan y cuando lo necesitan.

Se destaca en lograr la satisfacción del cliente, diferenciándose de las metodologías tradicionales, principalmente en que pone más énfasis en la adaptabilidad y previsibilidad. La misma tiene como objetivos principales disminuir los costes y aumentar el trabajo en grupo, en el que tanto los jefes de proyecto, los clientes y desarrolladores, son parte del equipo y están involucrados en el desarrollo del software. A través de los cuales ha obtenido el éxito como metodología ágil. (11)

Los desarrolladores que emplean XP se comunican con sus clientes y entre ellos mismos, para mantener un diseño limpio y simple. Obtiene retroalimentación, mediante pruebas al software las cuales se realizan desde el primer día. Esta metodología ha sido diseñada para equipos pequeños de desarrolladores y darle solución al eterno problema del desarrollo de software por encargo; entregar el resultado que el cliente necesita a tiempo.

El ciclo de vida ideal de XP consta de seis fases:

1. Exploración
2. Planificación de la entrega
3. Iteraciones
4. Producción
5. Mantenimiento
6. Muerte del proyecto

Características de XP, esta metodología se basa en:

Pruebas Unitarias: se basa en las pruebas realizadas a los principales procesos, de tal manera que se adelanta en algo hacia el futuro, se puede de esta manera hacer pruebas

de las fallas que pudieran ocurrir. Es como si se adelantara a obtener los posibles errores.

Refabricación: se basa en la reutilización de código, para lo cual se crean patrones o modelos estándares, lo que facilita el cambio.

Programación en pares: una particularidad de esta metodología es que propone la programación en pares, la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no realiza en ese momento. (12)

1.4.3 Scrum

Scrum es una metodología ágil de desarrollo de proyectos, a menudo se percibe como una metodología; pero en lugar de eso se puede ver como un *framework* para la gestión de procesos. En lugar de proporcionar completas y detalladas descripciones de todo lo que hay que hacer en un proyecto, gran parte de ella se deja al equipo de desarrollo de software Scrum. Esto es debido a que el equipo conocerá la mejor forma de resolver el problema que se presentan.

Por esta razón, en el desarrollo Scrum, por ejemplo, una reunión de planificación del sprint se describe en términos del resultado deseado (un compromiso con un conjunto de características que se desarrollarán en el siguiente sprint) en lugar de un conjunto de criterios de ingreso, las definiciones de tareas, la validación criterios, criterios de salida (ETVX) y así sucesivamente. (12)

Scrum cuenta con un equipo de auto-organización, funciones cruzadas. El equipo de Scrum es auto-organización en la que no hay un líder del equipo en general que decide qué persona va a hacer qué tarea o cómo un problema será resuelto. Esos son los temas que se deciden por el equipo en su conjunto. Los equipos de Scrum son apoyados por dos funciones específicas, el primero es un *ScrumMaster*, que es como un entrenador para el equipo, ayudando a los miembros del equipo que utilizan el proceso de Scrum para llevar a cabo al más alto nivel. El dueño del producto (PO) es el otro papel, y en el desarrollo de software Scrum, representa los negocios, clientes o usuarios, y guía al equipo hacia la construcción el producto adecuado. (13)

1.4.4 AUP-UCI

AUP-UCI es una metodología ágil de desarrollo definida para proyectos pertenecientes a la UCI. Esta metodología de desarrollo de software tiene entre sus objetivos aumentar

la calidad del software que se produce, se apoya en el Modelo CMMI-DEV v1.3. Constituye una guía para aplicar las mejores prácticas en una entidad desarrolladora. Estas prácticas están centradas en el desarrollo de productos y servicios de calidad.

De las 4 fases que propone AUP (Inicio, Elaboración, Construcción, Transición) se decide para el ciclo de vida de los proyectos de la UCI mantener la fase de Inicio, pero se modifica el objetivo de la misma, se unifican las restantes 3 fases de AUP en una sola, a la que llamaremos Ejecución y se agrega la fase de Cierre. Quedando de la siguiente forma:

Inicio: Durante el inicio se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente, que permite obtener información fundamental acerca del alcance del proyecto, realizándose las estimaciones de tiempo, esfuerzo y costo y se decide si se ejecuta o no.

Ejecución: En esta fase se ejecutan las actividades requeridas para desarrollar el software, donde se incluyen los ajustes de los planes del proyecto y se consideran los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto.

Cierre: En esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

El ciclo de vida:

Se decide para el ciclo de vida de los proyectos de la UCI tener 7 disciplinas, definidos a un nivel más atómico que en AUP. Estas son:

Modelado de negocio: El Modelado del Negocio es la disciplina destinada a comprender los procesos de negocio de una organización. Se comprende cómo funciona el negocio que se desea informatizar para tener garantías de que el software desarrollado va a cumplir su propósito.

Requisitos: El esfuerzo principal en dicha disciplina es desarrollar un modelo del sistema que se va a construir, la misma comprende la administración y gestión de los requisitos funcionales y no funcionales del producto. Existen tres formas de encapsular los requisitos [Casos de Uso del Sistema (CUS), Historias de usuario (HU) y Descripción de requisitos por proceso (DRP)], agrupados en cuatro escenarios condicionados por el Modelado de negocio.

Análisis y diseño: En esta disciplina, si se considera necesario, los requisitos pueden ser refinados y estructurados para conseguir una comprensión más precisa de estos, y una descripción que sea fácil de mantener y ayude a la estructuración del sistema (incluida su arquitectura). En esta disciplina se modela el sistema y su forma (incluida su arquitectura) para que soporte todos los requisitos, se incluyen además los requisitos no funcionales. Los modelos desarrollados son más formales y específicos que el de análisis.

Implementación: En la implementación, a partir de los resultados del Análisis y Diseño se construye el sistema.

Pruebas internas: En esta disciplina se verifica el resultado de la implementación, se prueba cada implementación, se incluyen tanto las implementaciones internas como intermedias, así como las versiones finales a ser liberadas. Se deben desarrollar artefactos de prueba como: diseños de casos de prueba, listas de chequeo y de ser posible, componentes de prueba ejecutables para automatizar las pruebas.

Pruebas de liberación: Pruebas diseñadas y ejecutadas por una entidad certificadora de la calidad externa, a todos los entregables de los proyectos antes de ser entregados al cliente para su aceptación.

Pruebas 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 usuarios finales para ejecutar aquellas funciones y tareas para las cuales el software fue construido. (14)

1.4.5 Metodología seleccionada

AUP-UCI es una metodología flexible que no requiere de una gran cantidad de desarrolladores. Genera solo la documentación necesaria y no la especificada para cada flujo de trabajo como lo hace RUP. Está diseñada para trabajar en proyectos pequeños donde la atención se centra en las actividades que realmente son importantes. Permite el uso de herramientas de cualquier tipo, se incluyen aquí las de código abierto. Es una metodología que se ajusta y aprovecha las ventajas que brindan las metodologías ágiles.

Aunque XP es una metodología popular y que brinda muchas ventajas presenta también desventajas. Ejemplo de estas es que no se genera documentación y puede ser complejo actualizar o realizar ingeniería inversa al proyecto. En la actualidad en cualquier sistema por más simple que sea el cliente solicita que le sea entregado,

además del software, manuales para los usuarios y otras documentaciones, XP por otra parte, no es viable en el desarrollo de sistemas que no requieren de la presencia del usuario, pues este requiere que el usuario esté siempre presente como un miembro más del equipo de desarrollo.

Se propone AUP-UCI ya que la misma se ajusta al tipo de módulo que se pretende realizar, por demás es la metodología adoptada en los proyectos de la UCI con el fin de lograr estandarizar el proceso de desarrollo de software, lo que hace de esta manera mucho más fácil integrar los artefactos que se generen durante el desarrollo de dicho módulo.

1.5 Framework de desarrollo

En el desarrollo de software, un *framework* es una estructura conceptual y tecnológica de soporte definida, normalmente con artefactos o módulos de software concretos, en base a la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, librerías y un lenguaje interpretado entre otros programas para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio. Provee una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio. (15)

1.5.1 Symfony

Symfony es un *framework* diseñado para optimizar el desarrollo de las aplicaciones web de acuerdo a algunas de sus características. Primeramente, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Presenta varias herramientas y clases que permiten reducir el tiempo de desarrollo de las aplicaciones web complejas. Symfony está desarrollado completamente en PHP 5; es compatible con la mayoría de los gestores de bases de datos, como MySQL, PostgreSQL, Oracle y Microsoft SQL Server. Se puede ejecutar tanto en plataformas Unix y Linux, como en plataformas Windows. Alguna de las características que presenta son:

1. Sigue las mejores prácticas de patrones de diseño para la web.
2. Código fácil de leer y permite un mantenimiento sencillo.
3. Independiente del sistema gestor de bases de datos.
4. Utiliza programación orientada a objetos.

5. Fácil de instalar y configurar en plataformas como Windows y Linux.
6. Posee una potente línea de comandos que facilitan la generación de código, lo cual permite ahorrar tiempo de trabajo.
7. Fácil de extender, lo que permite su integración con las bibliotecas de otros fabricantes.
8. Utiliza la arquitectura Modelo – Vista – Controlador (MVC) como la capa de abstracción de base de datos, el controlador frontal y las acciones.

Symfony automatiza la mayoría de elementos comunes de los proyectos web, como, por ejemplo:

1. La capa de internacionalización que incluye Symfony permite la traducción de los datos y de la interfaz, así como la adaptación local de los contenidos.

2. La capa de presentación utiliza plantillas y diseños que pueden ser creados por diseñadores HTML sin ningún tipo de conocimiento del *framework*. Las ayudas incluidas permiten minimizar el código utilizado en la presentación, pues encapsulan grandes bloques de código en llamadas simples a funciones.

3. Los formularios incluyen validación automatizada y relleno automático de datos, lo que asegura la obtención de datos correctos y mejora la experiencia de usuario.

4. Los datos incluyen mecanismos de escape que permiten una mejor protección contra los ataques producidos por datos corruptos.

5. La gestión de la caché reduce el ancho de banda utilizado y la carga del servidor.

6. La autenticación y la gestión de credenciales simplifican la creación de secciones restringidas y la gestión de la seguridad de usuario.

7. Las interacciones con AJAX son tan fáciles de implementar mediante las ayudas que permiten encapsular los efectos JavaScript compatibles con todos los navegadores en una única línea de código. (16)

1.5.2 Laravel

Laravel se inició el año 2011, y aprovecha las mejoras de PHP 5.3, ofrece una sintaxis clara y simple en la creación de código PHP. Se pueden escribir aplicaciones web con muy pocas líneas de código que además fáciles de entender, incluso para un programador recién iniciado. Laravel nace confiando en Symfony, pero también

depende de otras grandes librerías como SwiftMailer, Carbon o Doctrine. Usa además la metodología tradicional MVC. Sin embargo, el *framework* propone una vía más rápida en PHP, la cual consiste en programar la interacción HTTP directamente como una función anónima asociada a una Ruta, la cual permite simplificar el código.

Las principales características que ofrece este *framework* son:

1. Una completa y concisa documentación que es muy sencilla de leer y comprender. Con código de ejemplo que es elegante y expresivo, facilita significativamente aprendizaje del *framework*, incluso sólo con observar el código.
2. Un ORM para manejar la capa de persistencia de datos de manera muy simple, con sólo un par de líneas de código se puede hacer mucho. Maneja con efectividad las distintas relaciones entre las tablas de una base de datos.
3. Un poderoso administrador de extensiones (*Bundles*[4]), algunos valiosos *Bundles* ya están disponible en la propia página de Laravel (17)
4. Es un proyecto Open Source con licencia MIT, de uso libre.

1.5.3 CodeIgniter

CodeIgniter es una aplicación web realizada en el lenguaje de programación PHP con el objetivo de servir como marco de trabajo para construir otras aplicaciones web bajo una cierta arquitectura y un lenguaje de programación como PHP. Está compuesto por un conjunto de librerías que se utilizan para construir las aplicaciones las cuales siguen los patrones que establece el *framework* para conseguir mayores resultados en las mismas. (18)

CodeIgniter facilita en gran medida los tiempos de aprendizajes y los periodos de desarrollo. Su modelo arquitectónico posibilita una mayor organización y estructuración de las soluciones. Algunos de los factores más importantes que lo definen son:

2. **Versatilidad**: funciona correctamente en disímiles ambientes de despliegue, incluso en ambientes compartidos, donde solo se tiene acceso mediante FTP y no se posee una administración total del mismo.
3. **Compatibilidad**: el *framework* es compatible con las principales versiones de PHP, lo cual permite poder desplegarlo sin problemas en diferentes ambientes.

[4] Es un directorio que almacena todo lo relacionado con una función específica, incluyendo clases PHP, configuración, e incluso hojas de estilo y archivos de Javascript.

4. **Facilidad de instalación**: la instalación en los diferentes ambientes se realiza solamente mediante la copia de los archivos fuentes del proyecto hacia el ambiente de despliegue. Una vez copiado dichos ficheros, se modifica el fichero de configuración de la base de datos y automáticamente, la aplicación queda lista para ser utilizada.
5. **Flexibilidad**: es un *framework* de desarrollo extremadamente flexible, es decir, si usted quiere seguir las pautas que define el *framework*, logrará un desarrollo guiado y robusto, pero si quiere tener toda la libertad posible, el *framework* le permite utilizar las pautas que defina conveniente.
6. **Ligereza**: el núcleo de CodeIgniter es extremadamente ligero, lo cual posibilita que los tiempos de respuesta sean sumamente atractivos. A partir de exámenes de velocidad de respuesta que se le han realizado al *framework* en diferentes situaciones, se ha llegado a la conclusión de que CodeIgniter es uno de los *framework* de desarrollo web sobre PHP más rápido que existen.
7. **Documentación**: la documentación para desarrollo web es sumamente fácil de acceder, podemos encontrarla fácilmente en toda la web o más específicamente en su sitio oficial (http://www.codeigniter.com). Una gran ventaja con que cuenta la documentación oficial del *framework*, es que posee una guía totalmente enlazada y permite acceder a sus principales tópicos a través de búsquedas inteligentes. (19)

1.5.4 Selección del framework de desarrollo

Symfony es el *framework* sobre el que se desarrolla el marco de trabajo Xalix, debido a que es un conjunto de componentes independientes, que cohesionados resuelven con mayor facilidad los problemas comunes del desarrollo web. Además, permite el desarrollo de aplicaciones extensibles, flexibles y configurables. Symfony constituye un *framework* maduro en el desarrollo de aplicaciones y existe mucha información sobre su uso, disponibles en varios libros gratuitos y decenas de tutoriales, lo que propicia un mejor aprendizaje y facilidad de uso por parte de los desarrolladores. Debido a la experiencia que se tiene con esta tecnología y a las características antes expuestas se decide desarrollar el componente sobre Symfony.

1.6 Ambiente de desarrollo

Un entorno de desarrollo integrado IDE (*Integrated Development Environment*) es un programa informático compuesto por un conjunto de herramientas de programación.

Consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI).

Algunas de las características de los IDE son:

1. Multiplataforma.
2. Soporte para diversos lenguajes de programación.
3. Integración con sistemas de control de versiones.
4. Reconocimiento de sintaxis.
5. Extensiones y componentes para el IDE.
6. Integración con *frameworks* populares.
7. Importar y exportar proyectos.
8. Múltiples idiomas.
9. Manual de usuarios y ayuda. (20)

1.6.1 PhpStorm

PhpStorm es un IDE de programación desarrollado por JetBrains. Es uno de los entornos de programación más completos de la actualidad, permite editar código no sólo del lenguaje de programación php como lo indica su nombre y es multiplataforma. Es compatible con *framework* tales como Symfony, Zend Framework, Laravel, Magento, CakePHP, Yii, entre otros.

Tiene soporte para todas las características del lenguaje PHP para proyectos nuevos o viejos. Proporciona un buen auto completamiento de código, refactorización, prevención de errores en tiempo de ejecución. Aprovecha al máximo las tecnologías *front-end* como HTML5, CSS, Sass, Less, Stylus, CoffeeScript, TypeScript, Emmet, and JavaScript con refactorización, depuración y pruebas unitarias disponibles. Realiza muchas tareas rutinarias desde el IDE, gracias a la integración de sistemas de control de versión, el apoyo a la implementación remota, bases de datos, herramientas de línea de comandos, Vagrant, Composer, Rest client, entre otras herramientas.

Puede refactorizar su código de forma segura con cambio de nombre seguro, mover, eliminar, extraer método, variable en línea, firma de cambio y muchas otras. Las refactorizaciones específicas del idioma le ayudan a realizar cambios en todo el proyecto en cuestión de unos pocos clics y se pueden deshacer de forma segura. (21)

1.6.2 NetBeans

Netbeans es una herramienta que se utiliza para desarrollar aplicaciones web, móvil y de escritorio para diferentes lenguajes de programación como son Java, C++, Ruby y PHP, entre otros. Es de código abierto, es multiplataforma, multilenguaje, contiene servidores web y es fácil de instalar y de utilizar (22).

Características de NetBeans:

1. Formado por un IDE de código abierto y una plataforma de aplicación que permite a los desarrolladores crear con rapidez aplicaciones web.
2. Ofrece documentación y recursos de formación exhaustivos.
3. Dispone de soporte para crear interfaces gráficas de forma visual, desarrollo de aplicaciones web, control de versiones, colaboración entre varias personas.
4. Multiplataforma.
5. Está desarrollado para usarse generalmente con lenguaje Java, aunque permite su uso con otros lenguajes de programación como PHP.

1.6.3 Selección del ambiente de desarrollo

NetBeans es una herramienta escrita en Java, para programar, compilar y depurar programas. Es un producto libre y gratuito. Permite el desarrollo de aplicaciones de escritorio, para celulares y para la web. Es un IDE multilenguaje completo y modular. Cuenta con depurador, perfilador, herramientas para refactorizaciones y completamiento de código. Brinda soporte nativo a los *frameworks* Symfony y jQuery. Además, es multiplataforma por lo que es posible utilizarlo desde cualquier sistema operativo si se tiene instalada la máquina virtual de Java. Se seleccionó para el desarrollo del componente debido a que posee un gran soporte para la edición en HTML5, tiene una interfaz con múltiples opciones, un aceptable completamiento de código del *framework* Symfony y es una herramienta gratuita. Por estas características se decide utilizar como IDE para el desarrollo del módulo el NetBeans.

1.7 Tecnologías a utilizar

1.7.1 XML

XML, el Lenguaje de Marcas Extensible, es la sintaxis de documentos más robusta, fiable y flexible inventada hasta ahora. Define un protocolo genérico para marcar datos con etiquetas sencillas y de fácil lectura que en la práctica se ha convertido en el modelo

a seguir para los nuevos formatos de documentos diseñados en casi todas las aplicaciones de ordenador. Esta obra es la referencia integral para conocer el mundo en constante crecimiento de XML, desde los fundamentos sintácticos más básicos, hasta los detalles de la creación de DTD y esquemas y las API que se pueden utilizar para leer y escribir documentos XML en una considerable variedad de lenguajes de programación. Tanto los diseñadores web que usan XML para generar páginas web y archivos PDF como los programadores C++ que utilizan REST o SOAP para transmitir datos entre sistemas. Algunas de las ventajas de XML son las siguientes:

1. Las aplicaciones se pueden generar rápidamente y su mantenimiento es sencillo.
2. La información es más accesible y reutilizable.
3. Separa los datos de la presentación y del proceso, lo que permite mostrar y procesar los datos.
4. Ofrece un formato para la descripción de datos estructurados lo que facilita que las declaraciones de los contenidos sean más precisas y los resultados de las búsquedas sean más significativos en varias plataformas. (23)

1.7.2 Java Script

Es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas, esta es una de las principales ventajas que presenta sobre el HTML, además es muy fácil de aprender. Es técnicamente un lenguaje de programación interpretado por lo que no es necesario compilar los programas para ejecutarlos. Permite integrarlo directamente en páginas HTML. Los programas escritos en este lenguaje se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. Algunas de las principales características son las siguientes:

1. Es interpretado por el navegador del cliente.
2. Está basado en objetos. Por lo cual emplea herencia, típicas de la Programación Orientada a Objetos (POO).
3. Su código se integra en las páginas HTML, incluido en las propias páginas.
4. No es necesario declarar los tipos de variables que van a utilizarse.
5. Las referencias a objetos se comprueban en tiempo de ejecución, por lo tanto, no se compila. (24)

1.7.3 CSS

Hojas de Estilo en Cascada (*Cascading Style Sheets*), es un mecanismo simple que describe la manera que se muestra un documento en la pantalla. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos. CSS se utiliza para dar estilo a documentos HTML y XML, separando el contenido de la presentación. Los estilos definen la forma de mostrar los elementos HTML y XML. CSS permite a los desarrolladores Web controlar el estilo y el formato de múltiples páginas Web al mismo tiempo. Cualquier cambio en el estilo marcado para un elemento en la CSS afectará a todas las páginas vinculadas a esa CSS en las que aparezca ese elemento (25).

1.7.4 PHP5

PHP5[5] utiliza la interpretación y compilación para ofrecer a los programadores mejor rendimiento y flexibilidad. Compila para el código una serie de instrucciones siempre que estas son accedidas. Las instrucciones son ejecutadas una por una hasta que el script termine. Esto es diferente a la manera convencional de compilación de lenguajes como C++. Es recompilado cada vez que se solicita un script.

Se usa principalmente en interpretación del lado del servidor para la generación de páginas web dinámicas. Las principales características de PHP5 son: rapidez, facilidad de aprendizaje, soporte multiplataforma tanto de diversos sistemas operativos, como servidores HTTP y bases de datos, además se distribuye de forma gratuita bajo una licencia abierta. Algunas de sus principales ventajas son (26):

1. Tiene manejo de excepciones.
2. Es completamente expandible. Está compuesto de un sistema principal, un conjunto de módulos y una variedad de extensiones de código.
3. Capacidad de conexión con la mayoría de los motores de base de datos, destaca su conectividad con MySQL y PostgreSQL.
4. Permite aplicar técnicas de programación orientada a objetos.
5. Es utilizado como módulo de Apache, lo que lo hace extremadamente veloz.

[5] PHP (acrónimo recursivo de *PHP: Hypertext Preprocessor*) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML.

6. Es de código abierto, lo cual le permite al usuario realizar cambios para lograr una mejor eficiencia en lo que necesite sin necesidad de depender de una compañía.

1.7.5 Bootstrap 3.3.1

Es el *framework* para CSS más popular que existe para el desarrollo web, posee una amplia documentación, haciéndose extensible su uso. Provee de forma fácil las mejores prácticas en el diseño de plantillas, se generan vistas muy agradables y ligeras para el cliente. Es de código abierto, con una comunidad que lo mantiene a través de gitHub. (27)

1.8 Gestor de Base Datos

Un Sistema Gestor de base de datos (SGBD) es un conjunto de programas que permiten crear y mantener una base de datos, asegurando su integridad, confidencialidad y seguridad. Permite almacenar y posteriormente acceder a los datos de forma rápida y estructurada. Las aplicaciones más usuales son para la gestión de empresas e instituciones públicas, además de ser ampliamente utilizado en entornos científicos con el objeto de almacenar la información experimental.

1.8.1 PostgreSQL

PostgreSQL es un SGBD Objeto-Relacional basado en el proyecto POSTGRES, de la universidad de Berkeley. Es una derivación libre de este proyecto y utiliza el lenguaje SQL. Fue pionero en muchos de los conceptos del sistema objeto-relacional actual. Este proyecto lleva más de una década de desarrollo, es hoy en día el sistema libre más avanzado, el mismo soporta la gran mayoría de las transacciones SQL y control concurrente.

Entre sus principales ventajas se destacan las siguientes:

1. Permite la gestión de diferentes usuarios y utiliza los permisos asignados a cada uno de ellos.
2. Posee una gran escalabilidad, lo que lo hace idóneo para su uso en sitios Web que atienden un gran número de solicitudes.
3. Puede ser instalado un número ilimitado de veces sin temor de sobrepasar la licencia.
4. Posee estabilidad y confiabilidad legendarias.
5. Es extensible a través del código fuente disponible sin costos adicionales.

6. Es multiplataforma, disponible en la mayoría de los sistemas operativos.
7. Permite implementar reglas, vistas, disparadores, sub-consultas y funciones.
8. Posee herramientas para generar SQL portable para compartir con otros sistemas compatibles con SQL. (28)

1.8.2 MySQL

Es un sistema de gestión de bases de datos relacional, licenciado bajo la GPL de la GNU. Su diseño multihilo le permite soportar una gran carga de forma muy eficiente. MySQL fue creada por la empresa sueca MySQL AB, que mantiene el copyright del código fuente del servidor SQL, así como también de la marca. Este gestor de bases de datos es, probablemente, el gestor más usado en el mundo del software libre, debido a su gran rapidez y facilidad de uso. Esta gran aceptación es debida, en parte, a que existen infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación, además de su fácil instalación y configuración (29).

Las principales características de este gestor de bases de datos son las siguientes:

1. Aprovecha la potencia de sistemas multiprocesador, gracias a su implementación multihilo.
2. Soporta gran cantidad de tipos de datos para las columnas.
3. Ofrece API's en gran cantidad de lenguajes (C, C++, Java, PHP).
4. Gran portabilidad entre sistemas.
5. Soporta hasta 32 índices por tabla.
6. Gestión de usuarios y *passwords*, manteniendo la seguridad en los datos.

1.8.3 Selección del Gestor de Base Datos

En la propuesta de solución se escoge PostgreSQL por su gran rapidez y facilidad de instalación, configuración y uso que brindan un mayor rendimiento. Además de ser el gestor de base datos definido en la arquitectura del proyecto para la plataforma.

1.9 Administrador de Base datos

1.9.1 pgAdmin III

Es una aplicación gráfica para gestionar el gestor de base de datos PostgreSQL, siendo la más completa y popular con licencia *Open Source*. Está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta

desarrollar bases de datos complejas. La interfaz gráfica soporta todas las características de PostgreSQL y facilita la administración. Es desarrollado en todo el mundo por una comunidad de expertos de PostgreSQL y está disponible en más de una docena de idiomas. Es de software libre liberado bajo la licencia de PostgreSQL. (30)

Tiene una enorme variedad de características entre las que se encuentran:

1. Multiplataforma.
2. Diseñado para las versiones de PostgreSQL múltiples y derivados.
3. Una amplia documentación.
4. Acceso a los datos.
5. Acceso a todos los objetos PostgreSQL.
6. Soporta la mayoría de las codificaciones del lado del servidor.

1.9.2 PostgreSQL Studio

PostgreSQL Studio permite llevar a cabo las tareas esenciales de desarrollo sobre una base datos PostgreSQL desde una consola basada en web. La misma se ejecuta en un entorno de nube, PostgreSQL Studio permite el trabajo con sus bases de datos sin la necesidad de abrir el firewall. Algunas de sus características son (31):

1. Fácil de instalar.
2. Fácil de personalizar.
3. El acceso seguro a las bases de datos.
4. Fácil navegador esquema de base.
5. Acceso a múltiples bases de datos.
6. Permite Administrar vistas, tablas foráneas, funciones, secuencias, tipos de datos, tablas, índices, restricciones, *triggers,* reglas de la tabla, datos de la tabla.
7. Estadísticas de mesa.
8. Ejecutar comandos SQL.
9. Compatibilidad entre plataformas.
10. El apoyo a un gran número de servidores Java.
11. Documentación.

1.9.3 phpPgAdmin

El phpPgAdmin es una aplicación web, escrita en PHP, para administrar bases de datos PostgreSQL, provee una manera conveniente a los usuarios para crear bases de datos, tablas, alterarlas y consultar sus datos usando el lenguaje estándar SQL, estuvo basado

en phpMyAdmin, pero hoy día ya no comparte código con él; incluso provee las mismas funcionalidades y más a los usuarios del servidor de base de datos PostgreSQL. Algunas de sus características son (32):

1. Administra varios servidores.
2. Facilita la manipulación de datos.
3. Soporta el volcado de los datos de las tablas en varios formatos: SQL, XML, XHTML, CSV, pestañas, pg_dump.
4. Importa scripts SQL, XML, CSV y pestañas.
5. Excelente compatibilidad de idiomas.
6. Fácil de instalar y configurar.

1.9.4 Selección de administrador de base datos

Se decide utilizar como administrador de base de datos pgAdmin III. Una característica interesante de pgAdmin III es que cada vez que realiza una modificación en un objeto, escribe la sentencia SQL correspondiente, lo que hace que sea una herramienta muy útil y a la vez didáctica.

1.10 Servidor web

Un servidor web sirve de contenido estático a un navegador, carga un archivo y lo sirve a través de la red al navegador de un usuario. Este intercambio es mediado por el navegador y el servidor que hablan el uno con el otro mediante HTTP (Protocolo de transferencia de hipertexto, es el protocolo usado en cada transacción de la World Wide Web) (33).

1.10.1 Nginx

Se trata de un servidor web y proxy inverso de código abierto ligero de alto rendimiento, que también incluye servicios de correo electrónico con acceso al *Internet Message Protocol* (IMAP) y al servidor *Post Office Protocol* (POP). Además, NGINX está listo para ser utilizado como un proxy inverso. En este modo, NGINX se utiliza para equilibrar la carga entre los servidores *back-end*, o para proporcionar almacenamiento en caché para un servidor *back-end* lento (34).

Son muchas las características que ofrece este servidor web, pero una de las más importantes es que se trata de un software que es asíncrono, a diferencia de Apache que está basada en procesos. La ventaja principal de ser asíncrono, es su escalabilidad.

En un sistema basado en procesos, cada conexión simultánea requiere de un hilo, lo que puede llevar a sobrecargar el servidor, mientras que en un servidor asíncrono se gestionan las peticiones en muy pocos hilos, reduce así las posibilidades de sobrecarga en el servidor.

Otras características que ofrece el servidor NGINX son:

1. Capaz de manejar más de 10.000 conexiones simultáneas con un uso bajo de memoria.
2. Balanceo de carga, lo que distribuye la carga entre los servidores que forman parte de la estructura y redirige cada petición hacia aquella máquina que tenga una menor carga.
3. Alta tolerancia a fallos.
4. Soporte para TSL, SSL, FastCGI, SCGI o uWSGI, entre otros.
5. Compatible con el nuevo estándar de direcciones IPv6.
6. Compresión y descompresión con Gzip, que permite comprimir al vuelo los archivos y datos que se mueven por la red, desde el servidor web hasta el navegador del usuario.
7. Reescritura de urls, para crear urls amigables que ayuden en el proceso del posicionamiento web, aunque a diferencia de Apache, NGINX no hace uso del fichero .htaccess, sino que las reglas de reescritura las carga directamente en su configuración.
8. Permite limitar el número de conexiones concurrentes.

1.10.2 Apache Tomcat7

Es un servidor de aplicaciones web basado en código abierto para tecnologías Java Servlet 3.0 y Java Server Pages 2.2 (JSP). Está publicado bajo licencia Apache versión 2. Posee una comunidad de desarrollo muy selectiva, utiliza Catalina como contenedor de servlets a partir de la versión 4. Es multiplataforma, siendo compatible con todo sistema operativo que posea instalada la máquina virtual de Java. Apache Tomcat permite el monitoreo y la administración remota de las aplicaciones mediante una conexión segura, servicios de virtual hosting, clustering y equilibrado de cargas en las aplicaciones. (35)

1.11 Herramienta de modelado

CASE, (acrónimo de *Computer Aided Software Engineering*) es el tipo de herramienta destinadas a aumentar la productividad en el desarrollo de software. El uso de las

mismas reduce los gastos y el tiempo en el ciclo de desarrollo del software ya que permiten modelar fácilmente procesos y sistemas, con previo dominio de al menos un lenguaje de modelado soportado por la herramienta. Estas incluyen un conjunto de programas que facilitan la optimización de un producto que ofrece apoyo permanente a los analistas, ingenieros de software y desarrolladores.

CASE es la aplicación de métodos y técnicas quedan utilidades a los programas, por medio de otros, procedimientos y su respectiva documentación. En esta investigación se hace referencia a las herramientas que ayudan a la gestión de requisitos; es decir al proceso de identificación, asignación y seguimiento de los mismos, incluyendo interfaz, verificación, modificación y control de cada requisito, durante el ciclo de vida del proyecto. Las actualizaciones de requisitos deben ser gestionados para asegurar que se mantenga la calidad del producto. (36)

1.11.1 Visual Paradigm 8.0

Es una herramienta de ingeniería de software multiplataforma, que propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, generación del código fuente de los programas y la documentación de los mismos. Su propósito general es soportar el ciclo de vida del proceso de desarrollo del software a través de la representación de todo tipo de diagramas.

Esta herramienta genera diseños enfocados al negocio las que contribuyen a un software de mayor calidad. Usa un lenguaje estándar y común a todo el equipo de desarrollo que facilita la comunicación. Mantiene capacidades de ingeniería directa e inversa y posibilita que el modelo y el código permanezcan sincronizados en todo el ciclo de desarrollo.

Se decidió usar Visual Paradigm debido a las características antes mencionadas y que es una herramienta CASE profesional. También se tuvo en cuenta que esta es la herramienta usada por la Universidad. La misma posee una licencia comercial y la UCI cuenta con la misma para su uso. (37)

1.11.2 Balsamiq Mockup

Es una herramienta que permite realizar *wireframes* para webs fácilmente. Un *wireframe* (aplicado a la Web) es una representación esquemática de la solución que se llevará adelante, sin entrar en etapas posteriores como el diseño gráfico o la programación web.

Permite acordar con el cliente aspectos claves de la solución a desarrollar, como la distribución general de los elementos, sus jerarquías y la navegación de los mismos. Provee representaciones de todos los elementos utilizados para la construcción de una web, como pantallas de navegadores, títulos, menús, imágenes, videos, entre otros (38).

Por las características de flexibilidad, eficiencia, simplicidad e interfaz amigable se escoge esta herramienta para el diseño de los prototipos de interfaces de usuario

1.12 Conclusiones parciales

1. El marco teórico elaborado permitió fundamentar y elaborar teóricamente los conceptos relacionados con la investigación.
2. El estudio de los elementos asociados a los generadores de reporte con el uso de métodos científicos permitió definir el motor de reportes a utilizar.
3. Durante el desarrollo del módulo se generará la documentación y los artefactos necesarios para su entendimiento, mantenimiento, posterior escalabilidad y reutilización futura guiados por la metodología AUP-UCI.
4. El análisis comparativo establecido entre las metodologías, lenguajes de programación, herramientas y tecnologías más usadas en el desarrollo de sitios web, permitió seleccionar las más apropiadas para guiar y desarrollar el módulo deseado.

Capítulo 2: Descripción de la solución propuesta

2.1 Introducción

El levantamiento de requisitos y la descripción de las historias de usuario son tareas que tienen un impacto significativo en la caracterización y en el esclarecimiento del componente que se desea implementar. Para llevar esta información a lo que realmente deben codificar los programadores es necesario realizar un buen diseño arquitectónico que represente correctamente la estructura de datos y los componentes necesarios para construir el módulo. Con este propósito, en el presente capítulo se describe la solución propuesta, se describen los usuarios que interactúan con dicho módulo, así como los requisitos funcionales y no funcionales. También guiados por la metodología AUP-UCI se recogen los artefactos generados durante las fases de la misma.

2.2 Actores del negocio

Un actor del negocio es cualquier individuo, grupo, entidad, organización, máquina o sistema de información externos; con los que el negocio interactúa. A continuación se relacionan los actores del negocio del módulo de reportes (39):

Actores del negocio	Descripción
Administrador	Es la persona que se encarga de crear, eliminar, modificar reportes y dar permisos a los usuarios que accederán a los mismos.
Profesor principal	Visualiza los reportes disponibles en la plataforma.
Profesor editor	Visualiza los reportes disponibles en la plataforma.

Tabla 3 Descripción de actores del negocio

2.3 Modelado de dominio

El modelo de dominio es un artefacto de la disciplina de análisis, construido con las reglas de UML durante la fase de concepción, en la tarea construcción del modelo de dominio, representado como uno o más diagramas de clases y que contiene, no conceptos propios de un sistema de software sino de la propia realidad física, estos

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. Similares a los mapas mentales utilizados en el aprendizaje, el modelo de dominio es utilizado por el analista como un medio para comprender el sector industrial o de negocios al cual el sistema va a servir.

El modelo de dominio puede ser tomado como el punto de partida para el diseño del sistema. Esto se debe a que cuando se realiza la programación orientada a objetos, se supone que el funcionamiento interno del software va a imitar en alguna medida a la realidad, por lo que el mapa de conceptos del modelo de domino constituye una primera versión del sistema.

2.3.1 Diagrama de modelo de dominio

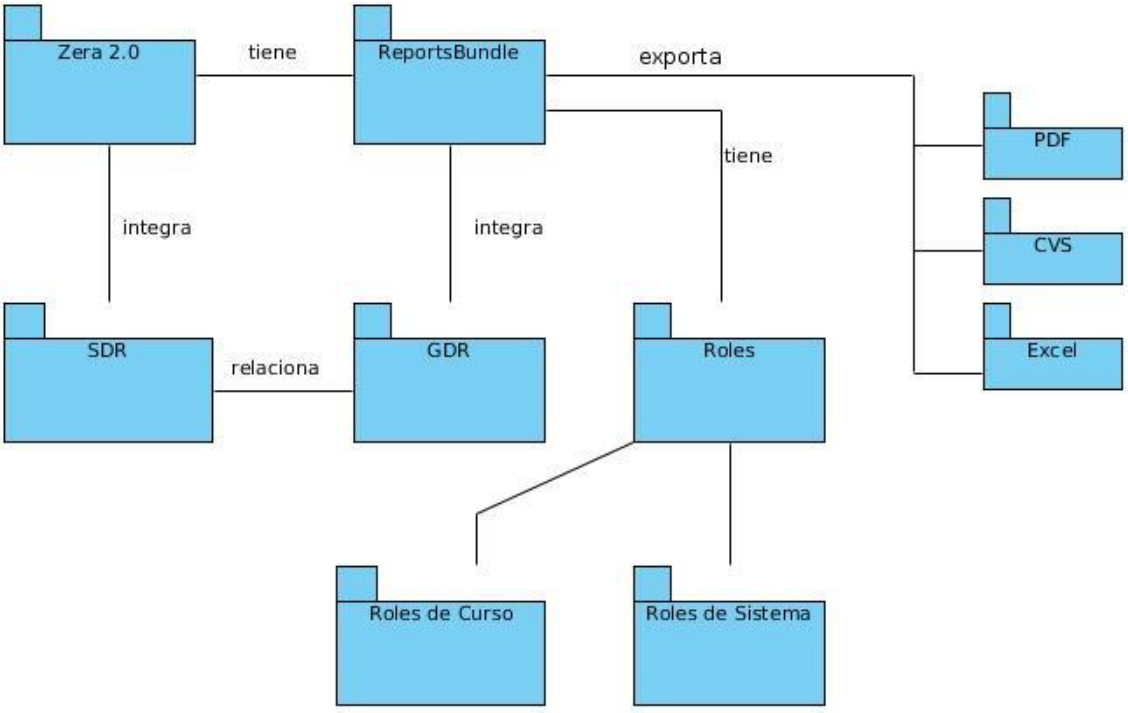


Imagen 1. Diagrama modelo de dominio

2.4 Propuesta de solución

Al seleccionarse Jasper Reports como motor de reportes, es necesario la implementación de un módulo que permita la integración con la plataforma ZERA 2.0, el mismo facilitará al usuario la consulta de la información que se genera dentro de la plataforma a través de reportes, se utiliza el Servidor de Reportes Dinámico 1.0 (SDR) para mostrar dichos reportes a través de servicios web de tipo *REST*. Para el diseño de los reportes se recomienda el uso de herramientas como el Generador de Reportes Dinámicos 2.0 (GDR) o el iReport, los cuales no son más que *front-end*[6] gráficos para la edición de informes.

2.5 Descripción del sistema propuesto

Todos los usuarios que accedan al módulo deben estar autenticados en la plataforma para una mayor seguridad del sistema y evitar que un usuario que no esté registrado en la base de datos y no tenga permisos, pueda acceder al módulo. El usuario según sus privilegios podrá acceder al módulo y visualizar los reportes permitidos en dependencia del nivel de acceso que tenga. El usuario podrá seleccionar el tipo de reporte deseado, el cual puede ser exportado en la extensión deseada seleccionada por el usuario (PDF, CSV, XLS).

De manera general este módulo permitirá gestionar todo lo referente a la generación de los reportes para los diferentes roles dentro de la plataforma, permitiendo visualizar la información que se genera por la interacción de los usuarios con la misma.

2.6 Requisitos del software

Los requisitos de software cumplen un papel primordial en el proceso de producción de software, debido a que se enfoca un área fundamental: la definición de lo que se desea producir. Ian Sommerville[7] presenta una definición acerca de lo que es un requisito: “Un requerimiento es simplemente una declaración abstracta de alto nivel de un servicio que debe proporcionar el sistema o una restricción de éste”. (40)

Los requisitos de software se dividen en dos categorías, los requisitos funcionales y los no funcionales, los primeros son los encargados de describir las funciones que deberá

[6] En diseño de software el front-end es la parte del software que interactúa con el o los usuarios.
[7] Ian Sommerville F es un académico británico, autor de varios libros sobre la ingeniería de software. Trabajó como profesor de Ingeniería de Software de la Universidad de St. Andrews en Escocia hasta 2014 y es un destacado investigador en el campo de la ingeniería de sistemas, la fiabilidad del sistema y la informática social.

ser capaz de realizar el sistema y los no funcionales tienen que ver con las características que de una forma u otra puedan limitar el funcionamiento del mismo, ya sea rendimiento, interfaces de usuario, fiabilidad, mantenimiento, seguridad, portabilidad y estándares.

2.6.1 Requisitos funcionales

RF 1: Gestionar parámetros de conexión con el servidor SDR

RF 1.1: Editar configuración de conexión

RF 1.2: Autenticar usuario

RF 2: Gestionar reportes en el sistema

RF 2.1: Adicionar reporte

RF 2.2: Modificar reporte

RF 2.3 Eliminar reporte

RF 2.4 Visualizar reporte

RF 3: Generar reportes de prueba

RF 3.1: Generar reporte de la cantidad de cursos asociados a los usuarios

RF 3.2: Generar reporte de la cantidad de usuarios asociados a los roles

RF 3.3: Generar reporte de la cantidad de usuarios matriculados en los cursos

RF 3.4: Generar reporte de la cantidad de cursos asociados a las categorías

RF 3.5: Generar reporte de la cantidad de cursos asociados a las escalas de calificación

RF 3.6: Generar reporte de la cantidad de usuarios que han participado en un foro

RF 3.7: Generar reporte de la cantidad de temas asociados a un foro

RF 3.8: Generar reporte de los usuarios con mejores evaluaciones en un foro

RF 3.9: Generar reporte de la cantidad de cuestionarios en el que ha sido incluido el ejercicio

RF 3.10: Generar reporte de la cantidad de usuarios que han resuelto el ejercicio

RF 3.11: Generar reporte de la cantidad de usuarios que han resuelto el cuestionario

RF 3.12: Generar reporte del promedio de calificaciones obtenidas por ejercicio

RF 3.13: Generar reporte de la cantidad de actividades asociadas a las rúbricas

RF 3.14: Generar reporte de la cantidad de rúbricas asociadas a los aspectos

RF 3.15: Generar reporte de la cantidad de cursos asociados a las instituciones

RF 3.16: Generar reporte de la cantidad de usuarios asociados a las instituciones

2.6.2 Requisitos no funcionales

Los requisitos no funcionales son los encargados de determinar las cualidades que el producto debe tener o cumplir, por lo que es de vital importancia que el cliente y los usuarios puedan valorar las características no funcionales del producto: usabilidad, seguridad y apariencia, y puedan marcar la diferencia entre un software bien aceptado y uno con poca aceptación (41).

Existen varios tipos de requisitos no funcionales. Los seleccionados para la solución propuesta son:

RNF 1: Funcionalidad

El sistema cumplirá con requisitos de seguridad tales como:

1. Brindar protección contra accesos no autorizados a la información.
2. Garantizar el acceso a las funcionalidades definidas para los usuarios de acuerdo a los roles que posean.
3. Mantener el sistema disponible evitando que los mecanismos de seguridad impidan el acceso a la información requerida por los usuarios autorizados.

RNF 2: Usabilidad

1. El sistema podrá ser usado por cualquier persona que posea conocimientos básicos en el manejo de la computadora y de un ambiente web en sentido general.
2. El sistema debe tener acceso al menú general desde cualquiera de sus páginas.
3. Se deben mostrar las rutas de acceso según la navegación que tenga el usuario.

4. El sistema debe permitir a los usuarios con permisos administrativos realizar ciertas acciones de configuración en dependencia de su rol.

RNF 3: Seguridad

1. La plataforma debe proteger los productos educativos que gestione, de forma que restrinja el acceso a usuarios no autorizados.
2. Cada usuario va a tener asignado un rol en el sistema.
3. Niveles de acceso determinados por los roles válidos dentro de la plataforma, los cuales fueron definidos en consenso con los desarrolladores de la plataforma.
4. Con respecto a la seguridad del código: utilizar ofuscadores de código fuente. Asegurar la plataforma de los ataques de hackers, tales como *Cross Site Scripting* (XSS), *SQL Injection*, *Cross-Site Request Forgeries* (CSRF), mediante la escritura de código seguro.

RNF 4: Restricciones de diseño

1. La arquitectura que se usará será Modelo-Vista-Controlador.
2. El lenguaje de programación que se debe emplear: PHP 5.3.3 ó superior.
3. El marco de trabajo base de desarrollo que se utilizará es: Symfony 2.7.

RFN 5: De software

1. PostgreSQL 9.4.x.
2. Servidor Ngnix
3. Apache Tomcat 7.
4. Sistema operativo: Distribución de CentOS última versión estable.

El sistema será accesible desde estaciones de trabajo de escritorio, laptop, tablets y smartphones. Estos deberán contar con un navegador web moderno (Navegadores web: Firefox, Chrome, Opera, navegadores de dispositivos móviles actualizados) para acceder a la aplicación y tener instalado el *plugin* de Java.

RFN 6: De hardware

1. Servidor de bases de datos relacional con memoria RAM: 16 GB, disco Duro: 100 GB y microprocesador: 6 x 800 Ghz.

2. Servidor de aplicaciones Ngnix: 2.7.x con memoria RAM: 16 GB, disco Duro: 500 GB y microprocesador: 6 x 800 Ghz.

2.7 Historias de usuario

Una historia de usuario describe una funcionalidad que, por sí misma, aporta valor al usuario. (42) Las HU son creadas a partir de las funcionalidades definidas. En las siguientes tablas se mostrarán algunas de las HU de los diferentes Requisitos Funcionales (RF), que presentan prioridad alta por constituir principales funcionalidades del módulo. Para documentar las HU se utiliza una plantilla con los aspectos definidos a continuación:

1. **Número**: posee el número asignado a la HU.
2. **Nombre**: atributo que contiene el nombre de la HU.
3. **Usuario**: usuario del sistema que utiliza o protagoniza la HU.
4. **Prioridad en el negocio**: evidencia el nivel de prioridad de la HU en el negocio.
5. **Puntos estimados**: estimación hecha por el equipo de desarrollo del tiempo de duración de la HU.

A continuación, se muestran las descripciones textuales de la HU: Editar configuración de conexión. Las restantes HU están presentes en el Anexo 1.

Número: 1	**Nombre del requisito:** Editar configuración de conexión
Programador: N/A	**Iteración Asignada:** 1era
Prioridad: Alta	**Tiempo Estimado:** -
Riesgo en Desarrollo: N/A	**Tiempo Real:** -

Descripción:

3- Objetivo:

Permitir configurar los parámetros de conexión con el SDR.

2- Acciones para lograr el objetivo (precondiciones y datos):

Para configurar parámetros de conexión hay que:

- Definir en el archivo service.yml los parámetros necesarios para la conexión.

3- Comportamientos válidos y no válidos (flujo central y alternos):

Campos para configurar los parámetros de conexión:

- sdr_server: Localización del servidor SDR.
- platform_database: Localización de la BD de la plataforma.
- sdr_database: Localización de la BD del SDR.

4- Flujo de la acción a realizar: - Definir los parámetros sdr_server, platform_database y sdr_database en el archivo service.yml perteneciente al bundle de reportes.
Observaciones:

Tabla 4. Historia de usuario editar configuración de conexión.

2.8 Patrones

Un patrón es una colaboración parametrizada junto con las pautas sobre cuando utilizarlo. Un parámetro se puede sustituir por diversos valores, para producir distintas colaboraciones. Los parámetros señalan generalmente las ranuras para las clases. Cuando se instancia un patrón, sus parámetros están ligados a clases reales de un diagrama de clases o a los roles dentro de una colaboración más amplia. (43)

2.8.1 Patrón arquitectónico Modelo – Vista – Controlador en Symfony

El *framework* Symfony está basado en el patrón arquitectónico Modelo-Vista-Controlador (MVC). Este patrón separa en tres niveles las funcionalidades de una aplicación con el objetivo de aumentar la usabilidad de las mismas. Estos niveles son:

Modelo: administra y maneja todo lo relacionado con los datos del sistema, da respuesta a peticiones de información sobre el estado de la aplicación (normalmente desde la vista), y responde con instrucciones de cambio de estado (usualmente desde el controlador) a la vista.

Vista: gestiona lo relacionado con mostrar la información al usuario. La vista está representada por ficheros escritos en PHP que se encargan de construir la página HTML con la que interactúa el usuario.

Controlador: interpreta los eventos que son lanzados por la entrada estándar del usuario (normalmente mouse y teclado), informa de los mismos al modelo y/o la vista para que se ejecuten los cambios apropiadamente. (44)

El siguiente esquema muestra el funcionamiento interno de estos niveles en Symfony:

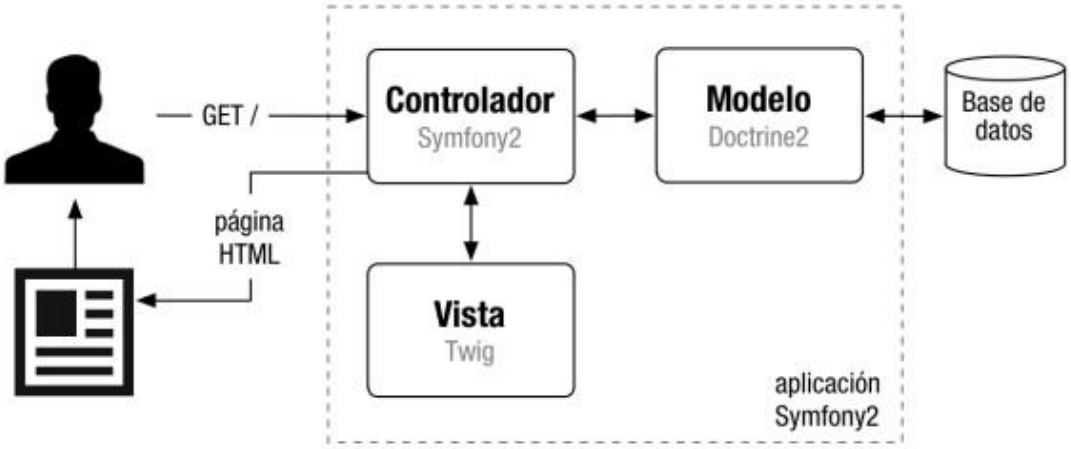


Imagen 2. Funcionamiento interno de Symfony 2.0

2.8.2 Patrón de diseño

Los subsistemas dentro de una arquitectura de software, al igual que sus interrelaciones, consisten normalmente de unidades arquitectónicas más pequeñas, estas se describen a través de patrones de diseño. Un patrón de diseño define un esquema de refinamiento de los subsistemas o componentes dentro de un sistema, o las relaciones entre estos, los mismos describen una estructura común y recurrente de componentes interrelacionados que resuelve un problema general de diseño dentro de un contexto particular. Los patrones de diseño trabajan a una escala intermedia y son menores en escala que un patrón de arquitectura, pero logran ser independientes del lenguaje de programación. La aplicación de un patrón de diseño no tiene efectos sobre la estructura fundamental del sistema (arquitectura), pero puede tener una fuerte influencia sobre la arquitectura de un subsistema. (40)

2.8.3 Patrones GRASP

General Responsibility Asignment (GRASP por su acrónimo en inglés) describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. El nombre se eligió para indicar la importancia de captar estos principios, si se quiere diseñar eficazmente el software orientado a objetos. Favorecen la reutilización de código y ayudan a construir software basado en la reutilización. Los utilizados en la solución son los que a continuación se muestran (45):

Bajo Acoplamiento: El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas.

Acoplamiento bajo significa que una clase no depende de muchas clases y se evidencia en el hecho de que los controladores heredan únicamente de la clase Default Controller.

Experto: Este patrón se basa en la asignación de responsabilidades, las cuales se asignan a las clases que posean la información necesaria para llevarlas a cabo. Es utilizado en la capa de abstracción del modelo de datos. Con el uso del ORM Doctrine, Symfony genera automáticamente las clases que representan las entidades de nuestro modelo de datos. Asociado a cada una de estas clases son generadas un conjunto de funcionalidades que las relacionan de forma directa con la entidad que representan. Estas clases contienen toda la información necesaria de la tabla que representan en la base de datos.

Creador: El patrón 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 debe conectar con el objeto producido en cualquier evento, por lo que es utilizado en los controladores, en ellos se encuentran las acciones definidas para el sistema.

Controlador: 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. Dentro del *framework* el patrón se evidencia en las clases que forman la capa Controlador del patrón arquitectónico MVC. En Symfony todas las peticiones son procesadas por un solo controlador frontal, este es el único punto de entrada de una aplicación en un entorno determinado.

Alta Cohesión: cada elemento del diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos y auto-identificable. La información que almacena una clase debe ser coherente y estar en mayor medida relacionada con la clase. Estas clases mejoran la claridad y la facilidad con que se entiende el diseño.

2.9 Modelo de análisis

El objetivo del análisis es traducir los requisitos a una especificación que describe cómo implementar el sistema. El análisis consiste en obtener una visión del sistema que se preocupa de ver qué hace, de modo que sólo se interesa por los requisitos funcionales. (43)

2.9.1 Diagrama de clases y colaboración del análisis

Los diagramas de clases del análisis ofrecen una especificación más precisa de los requisitos que la obtenida como resultado de la captura de requisitos y la estructura de modo que facilita su comprensión, modificación y mantenimiento.

Los diagramas de colaboración, por su parte, son una representación más concreta y detallada que los diagramas de clases del análisis, aunque también representan la parte estática del sistema, este contiene las clases y sus relaciones. Son empleados para representar las relaciones que se establecen entre las clases. (43)

A continuación, se presenta el diagrama de clases y colaboración del análisis correspondiente a las historias de usuario.

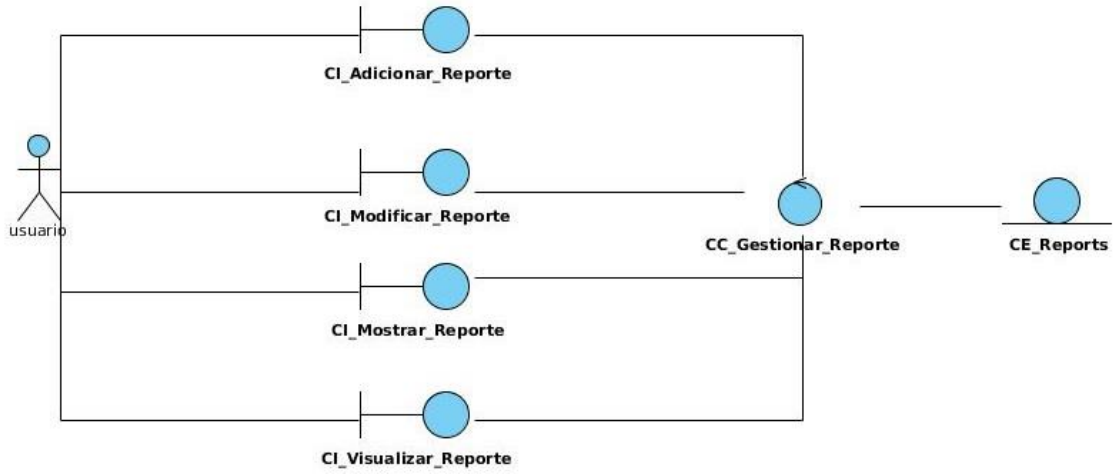


Imagen 3. Diagrama de clases de análisis

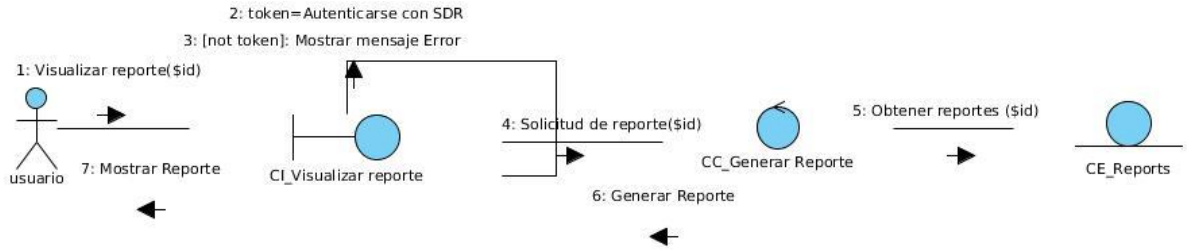


Imagen 4. Diagrama de colaboración

2.10 Modelo de diseño

Los diagramas de clases del diseño son una representación más concreta y detallada que los diagramas de clases del análisis, aunque también representan la parte estática

del sistema estos contienen las clases y sus relaciones. Son empleados para representar las relaciones que se establecen entre las clases. (43)

2.10.1 Diagrama de clases y secuencia del diseño

Los diagramas de clases del diseño son empleados para representar las relaciones que se establecen entre las clases, mientras que los de secuencia muestran la interacción de un conjunto de objetos en una aplicación a través del tiempo.

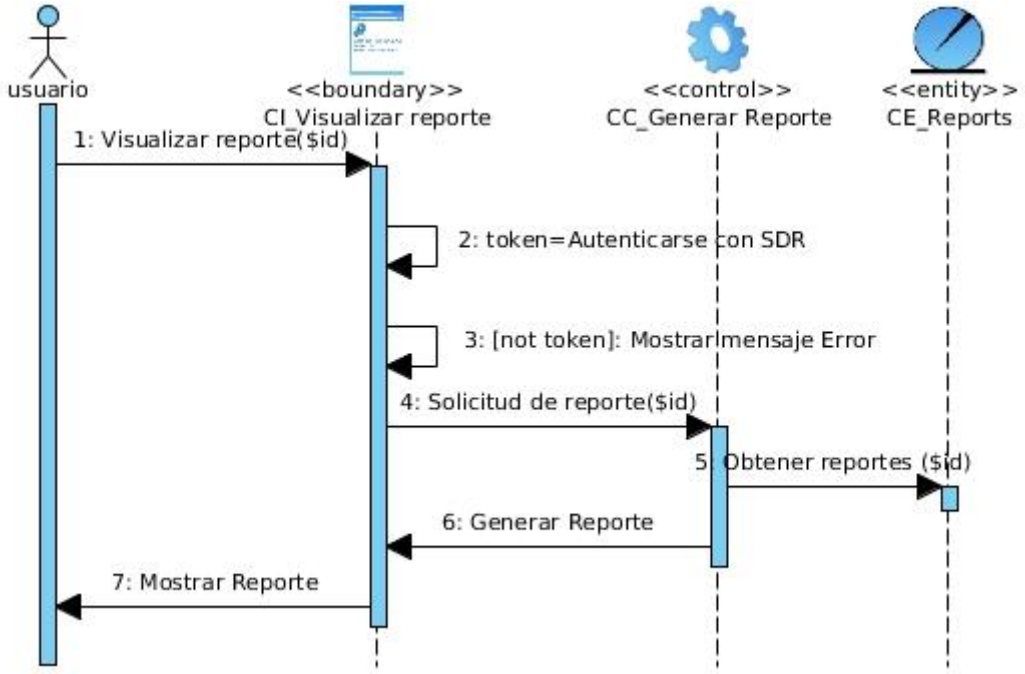


Imagen 5. Diagrama de Secuencia

2.11 Diagrama de despliegue

El modelo de despliegue define la arquitectura física del sistema. Se usa para modelar de manera detallada los nodos físicos y las asociaciones de comunicación que existen entre ellos (41). Del mismo modo, queda especificado qué hardware, sistemas

operativos, software de interfaces y soporte conformarán el nuevo sistema. A continuación se presenta el diagrama de despliegue propuesto para el sistema:

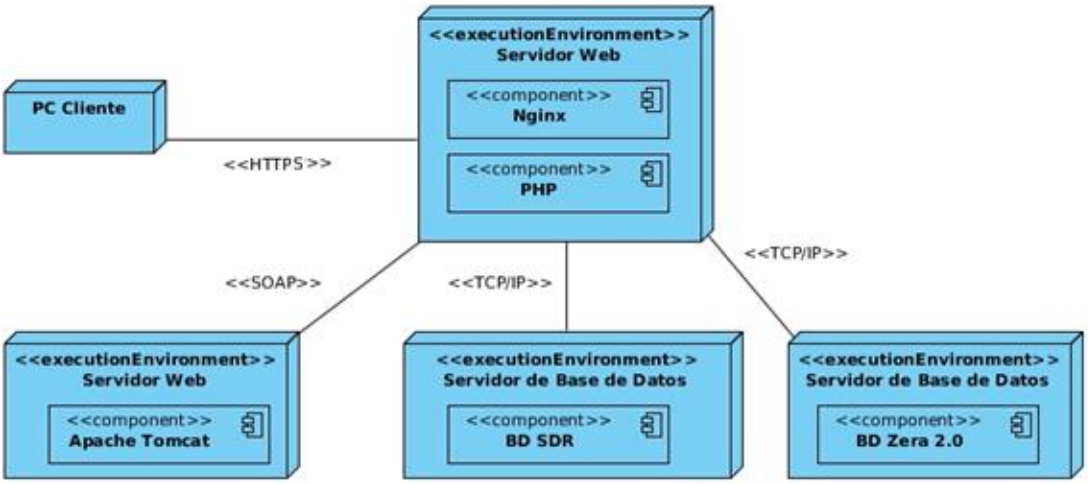


Imagen 6. Diagrama de despliegue

2.12 Modelado de los datos

Otra de las etapas del diseño es la elaboración del modelo de datos, con el propósito de garantizar que los datos persistentes sean almacenados coherente y eficazmente y definir el comportamiento que debe ser implementado en la base de datos. (46)

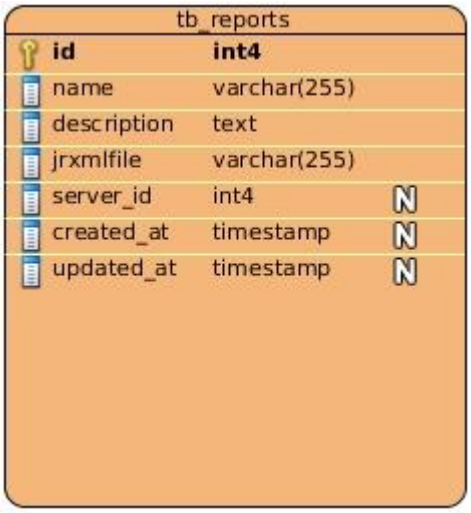


Imagen 7. Modelado de Datos

2.12.1 Descripción de las tablas de la base de datos

Se seleccionaron las tablas que almacenaban información de interés para la investigación.

tb_reports		
Descripción: en la siguiente tabla se agrupa la información correspondiente a los reportes existentes en la plataforma.		
Atributos	Tipo	Descripción
id	Integer	Etiqueta única que identifica al objeto en la tabla.
name	character varying(255)	Almacena el nombre del reporte en la plataforma
description	Text	Almacena la descripción del reporte
created_at	timestamp without time zone	Almacena la fecha de creación.
updated_at	timestamp without time zone	Almacena la fecha de actualización.
jrxmlfile	character varying(255)	Almacena el nombre del archivo de reporte (jrxml).
server_id	Integer	Almacena el identificador único del reporte en el SDR.

Tabla 5. Descripción de la tabla de la base de datos

2.13 Prototipos de interfaz

Al conjunto de elementos de la pantalla que permiten al usuario realizar acciones sobre el sitio web que es visitado, de le denomina interfaz, se considera parte de la interfaz a sus elementos de identificación, de navegación, de contenidos y de acción. Todos ellos deben estar preparados para ofrecer servicios determinados al usuario, con el fin de que este obtenga lo que vino a buscar cuando visitó el sitio web. Cada uno de los elementos

que sean integrados dentro de la interfaz debe estar pensado para causar un efecto sobre el usuario y deben ser utilizados con un propósito. (47)

A continuación, se muestran las interfaces propuestas para el desarrollo del módulo:

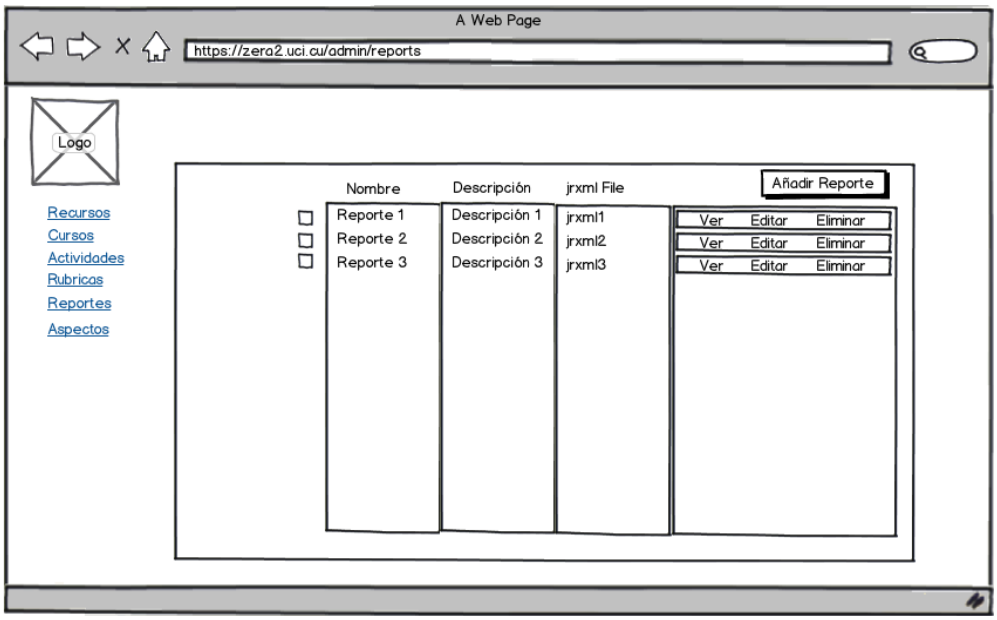


Imagen 8. Prototipo de interfaz de administración de los reportes.

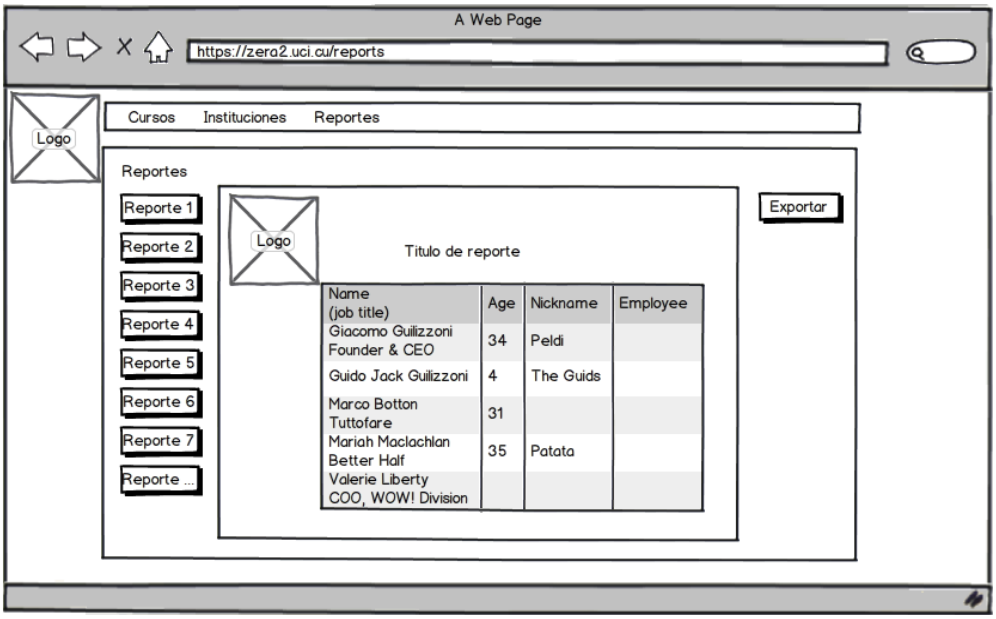


Imagen 9. Prototipo de interfaz para la visualización de reportes

2.14 Conclusiones parciales

1. Se realizó un estudio de las principales actividades que se manejan en el proceso de generación de reportes, las cuales se representan en el modelo de

dominio descrito dando una visión clara de las relaciones entre los conceptos asociados a la generación de reportes.

2. Se definieron los requerimientos funcionales y no funcionales, que recogen las principales necesidades del sistema a desarrollar, identificándose 22 requisitos funcionales y 6 requisitos no funcionales.
3. Se describieron los actores que interactúan con el sistema.
4. Se describió el uso de patrones presentes en el diseño del módulo, los mismos garantizan la reutilización de las funcionalidades y el uso de buenas prácticas en la codificación.
5. Se obtuvieron los artefactos Modelo de diseño, de base de datos y de despliegue los cuales representan la estructura interna del componente así cómo los elementos que intervendrán en la implantación de la solución.

Capítulo 3. Implementación y pruebas

3.1 Introducción

En el flujo de trabajo de implementación se desarrolla la arquitectura y el sistema como un todo. Este flujo de trabajo tiene como propósito definir la organización del código, implementar clases, objetos y subsistemas en términos de componentes y subsistemas de implementación, probar los componentes desarrollados e integrarlos a un sistema ejecutable. Durante el flujo de trabajo de prueba se verifica que el sistema implementa de verdad la funcionalidad descrita en las historias de usuario y que satisface los requisitos del componente, esto permite verificar el resultado de la implementación (41).

3.2 Diagrama de componentes

Un diagrama de componentes muestra los elementos de un diseño de un sistema de software. Permite visualizar la estructura de alto nivel del sistema y el comportamiento del servicio que estos componentes proporcionan y usan a través de interfaces (6).

Los componentes físicos incluyen archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables, o paquetes. Los diagramas de componentes prevalecen en el campo de la arquitectura de software, pero pueden ser usados para modelar y documentar cualquier arquitectura de sistema.

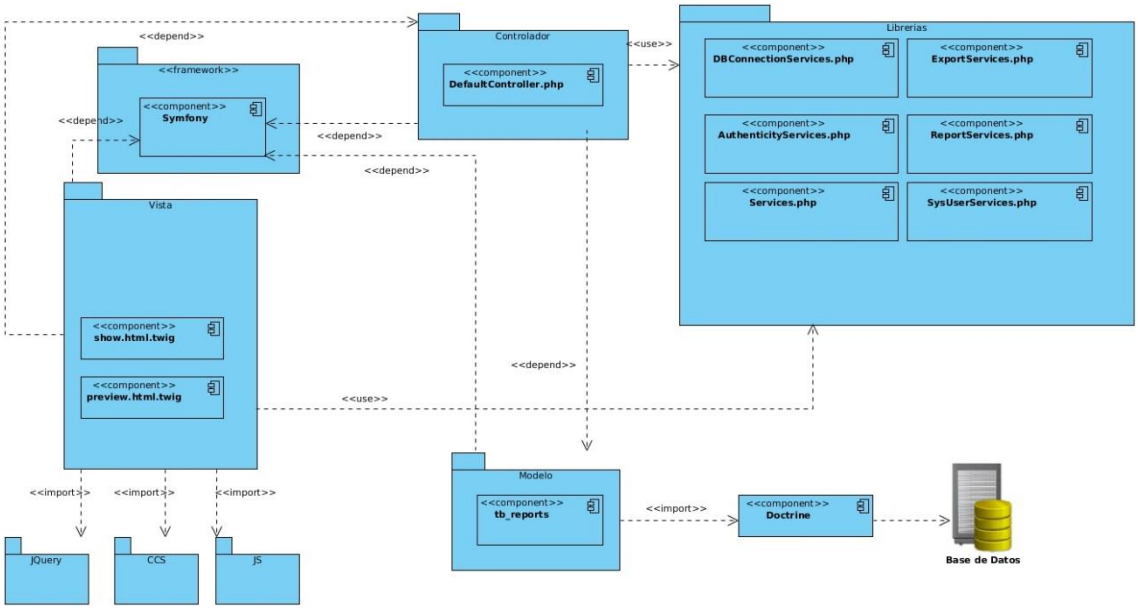


Imagen 10. Diagrama de componente

Capítulo 3. Implementación y pruebas

3.3 Estándares de codificación

Un estándar de codificación comprende los aspectos de la generación de código. Al inicio de un desarrollo de software, es necesario establecer un estándar de codificación para asegurar que los programadores del proyecto trabajen de forma coordinada. Establecer un estándar de codificación asegura que un equipo de programadores mantenga un código de calidad. A continuación se enumeran algunos de ellos (48):

1. Se utiliza el tabulador para la alineación, excepto en los archivos YAML donde se utilizan dos espacios.
2. Para el caso de las tablas en las base de datos deben ponerle el prefijo xl_, por ejemplo si una tabla se llama *comments* en la base de datos se le pone *xl_comments*.
3. Añade una coma después de cada elemento del arreglo en un arreglo multilínea, incluso después del último.
4. Declara las propiedades de clase antes que los métodos.
5. PHP debe usar las etiquetas <?php ¿> o la etiqueta corta <?= ¿>, no debe usarse otra variante.
6. Debe haber una línea en blanco después de los espacios de nombres y debajo de los bloques “use”.
7. Los nombres de las propiedades no deberían tener como prefijo el carácter “_” para indicar la visibilidad *protected* o *private*.

Debe haber un espacio después de la palabra clave de la estructura de control.

3.4 Implementaciones significativas

La imagen muestra el código para obtener y guardar el *token*, elemento de seguridad necesario para realizar todas las interacciones con el SDR.

```
//      Obteniendo token de autenticacion
        $configurator = $this->container->get('reports.sdr.conf');
        $auth = new AuthenticityServices($configurator);
        $login = $auth->login();
        $token = json_decode($login)->items[0]->token;
```

Imagen 11. Acción obtener y guardar *token*.

La imagen muestra el código para establecer la conexión con la base de datos de la plataforma paso necesario para obtener el *id* de la conexión y poder exportar posteriormente los reportes.

```
//Conectando a la BD de la plataforma
$dbcs = new DBConnectionServices();
$plataform_db = $this->container->getParameter('platform_server')

$conn_params = array(
    'token' => $token,
    'items' => array(
        'name' => time(),
        'password' => 'postgres',
        'url' => 'jdbc:postgresql://' . $plataform_db,
        'username' => 'postgres'
    )
);

$conn = $dbcs->create($conn_params);

$json_conn = json_decode($conn);

$id_conn = $json_conn->items[0]->id;

$em = $this->getDoctrine()->getManager();
$report = $em->getRepository('ReportsBundle:Reports')->find($id);

if (!$report) {
    throw $this->createNotFoundException();
}
```

Imagen 12. Acción conexión a la base de datos de la plataforma

La imagen muestra el código para obtener la vista previa del reporte seleccionado.

```php
// Generando Vista previa del reporte
$url_server = $this->getParameter('platform_server');
$params = array();
$connection = array(
    'name' => time(),
    'password' => $this->getParameter('database_password'),
    'url' => 'jdbc:postgresql://' . $url_server,
    'username' => $this->getParameter('database_user')
);

$params_report = array(
    'token' => $token,
    'file' => $this->container->getParameter('reports_directory')
    'connection' => $connection,
    'params' => $params
);

$report_data = $rs->preview($params_report);


return new Response($report_data);
```

Imagen 13. Acción mostrar vista previa del reporte

La imagen muestra el código para exportar el reporte seleccionado en el formato deseado.

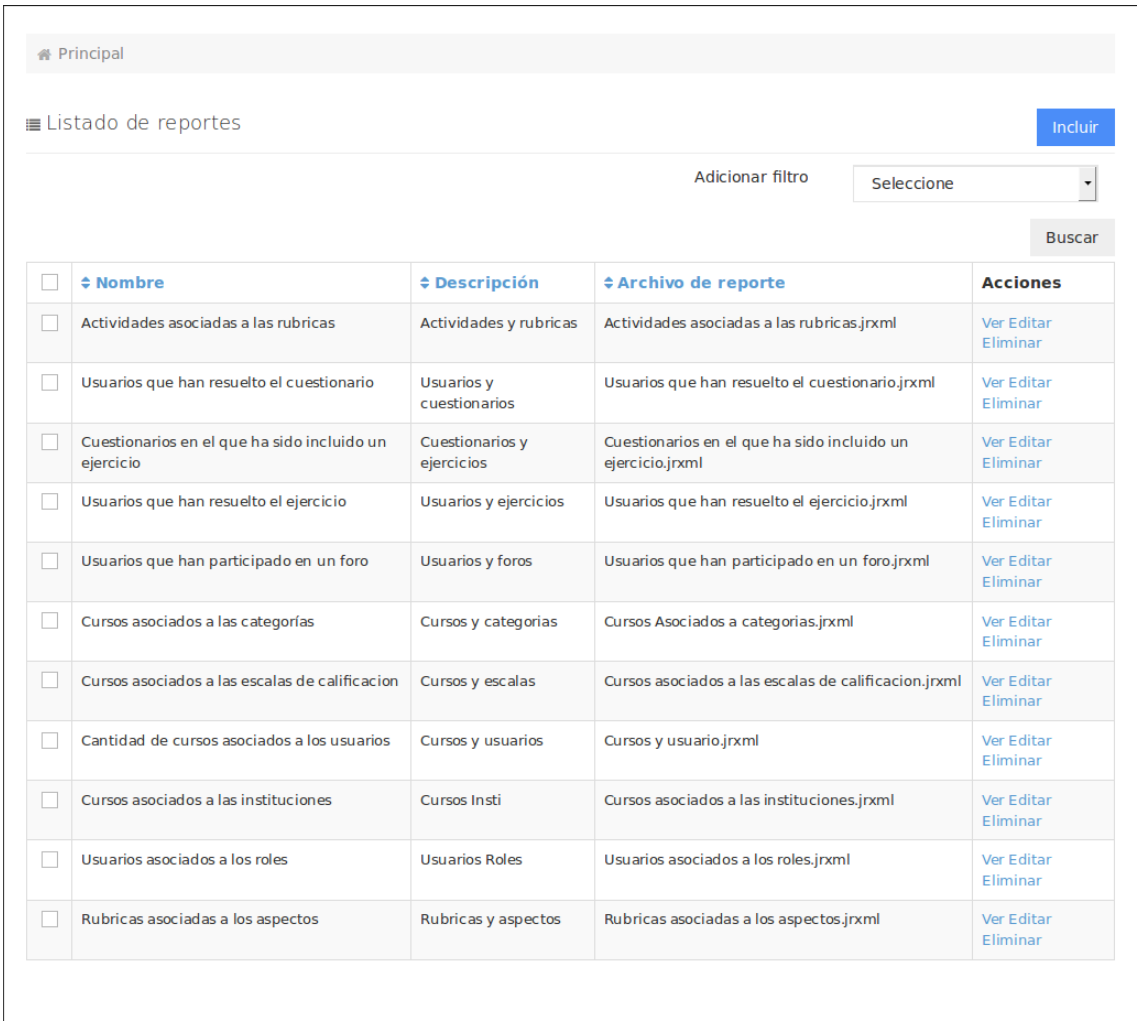
```php
//exportar reporte
$item = array();
$item['reportId'] = $server_id;
$item['dbConnectionId'] = $id_conn;
$export[] = $item;
$data = array();
$data['export'] = $export;
$data['format'] = $format;
$params_report_export = array(
    'token' => $token,
    'items' => $data
);

$info = $rs->export($params_report_export);
$arrayInfo = json_decode($info, TRUE);
$url = $arrayInfo['url'];
return $this->redirect('http://' . $url);
```

Imagen 14. Acción exportar reporte

Capítulo 3. Implementación y pruebas

3.5 Interfaces principales

La siguiente imagen muestra el resultado final de la interfaz para gestionar los reportes en la plataforma.



Nombre	Descripción	Archivo de reporte	Acciones
Actividades asociadas a las rubricas	Actividades y rubricas	Actividades asociadas a las rubricas.jrxml	Ver Editar Eliminar
Usuarios que han resuelto el cuestionario	Usuarios y cuestionarios	Usuarios que han resuelto el cuestionario.jrxml	Ver Editar Eliminar
Cuestionarios en el que ha sido incluido un ejercicio	Cuestionarios y ejercicios	Cuestionarios en el que ha sido incluido un ejercicio.jrxml	Ver Editar Eliminar
Usuarios que han resuelto el ejercicio	Usuarios y ejercicios	Usuarios que han resuelto el ejercicio.jrxml	Ver Editar Eliminar
Usuarios que han participado en un foro	Usuarios y foros	Usuarios que han participado en un foro.jrxml	Ver Editar Eliminar
Cursos asociados a las categorías	Cursos y categorias	Cursos Asociados a categorias.jrxml	Ver Editar Eliminar
Cursos asociados a las escalas de calificacion	Cursos y escalas	Cursos asociados a las escalas de calificacion.jrxml	Ver Editar Eliminar
Cantidad de cursos asociados a los usuarios	Cursos y usuarios	Cursos y usuario.jrxml	Ver Editar Eliminar
Cursos asociados a las instituciones	Cursos Insti	Cursos asociados a las instituciones.jrxml	Ver Editar Eliminar
Usuarios asociados a los roles	Usuarios Roles	Usuarios asociados a los roles.jrxml	Ver Editar Eliminar
Rubricas asociadas a los aspectos	Rubricas y aspectos	Rubricas asociadas a los aspectos.jrxml	Ver Editar Eliminar

Imagen 15. Interfaz para gestionar los reportes.

La siguiente imagen muestra la interfaz para editar los datos de los reportes (nombre, descripción y el archivo de reporte).

Capítulo 3. Implementación y pruebas

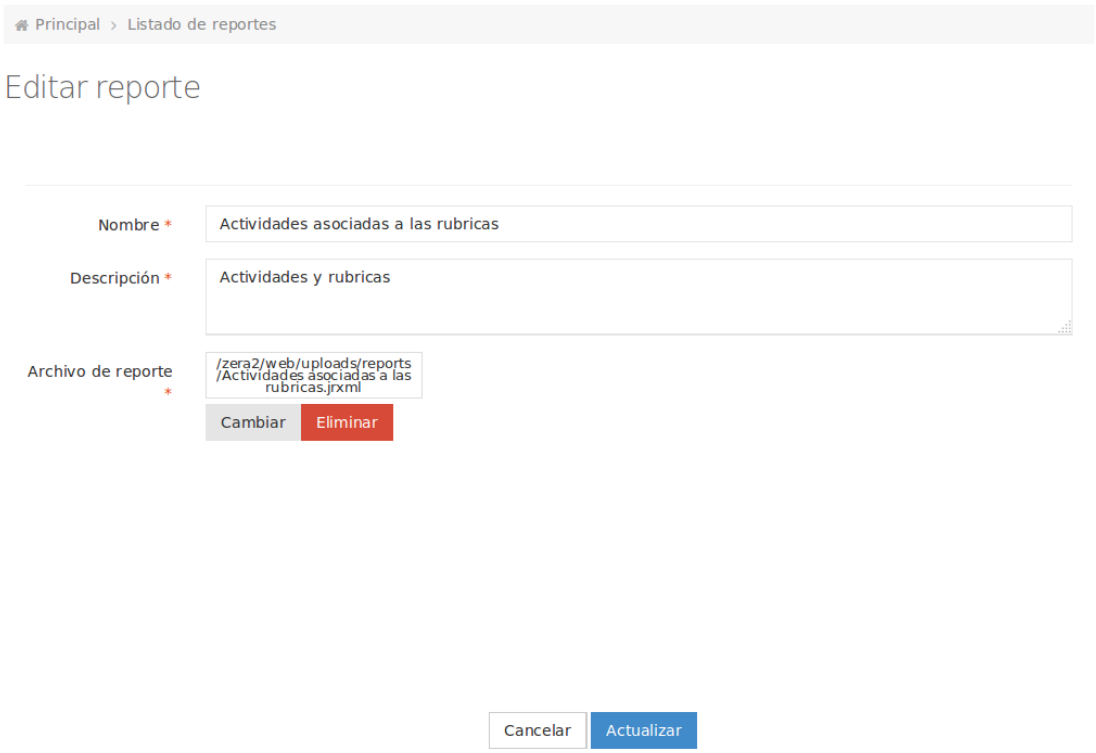


Imagen 16. Interfaz editar reporte.

La siguiente imagen muestra la interfaz mostrar reporte.

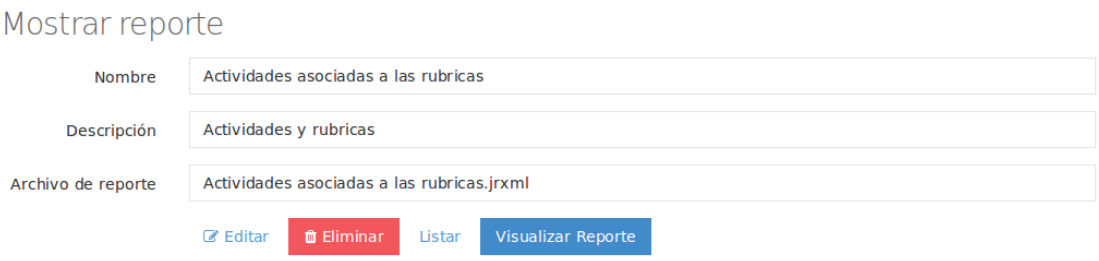


Imagen 17. Interfaz mostrar reporte

La siguiente imagen muestra la interfaz de visualización de los reportes.

Capítulo 3. Implementación y pruebas

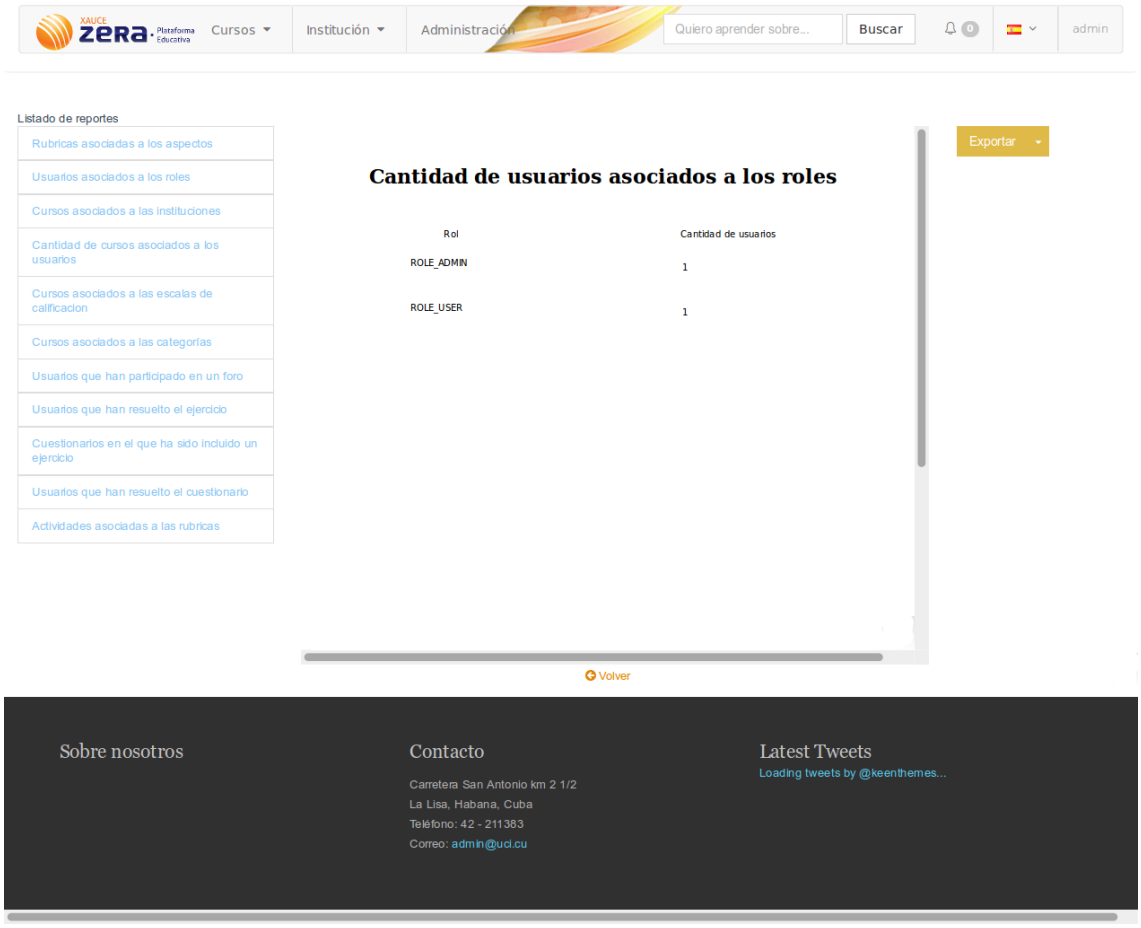


Imagen 18. Interfaz visualizar reporte

3.6 Pruebas de software

El único instrumento adecuado para determinar el *status* de la calidad de un producto software es el proceso de pruebas. En este proceso se ejecutan pruebas dirigidas a componentes del software o al sistema de software en su totalidad, con el objetivo de medir el grado en que el software cumple con los requerimientos. Al ejecutarse esta actividad se persigue descubrir defectos en el sistema que hagan que este tenga un comportamiento incorrecto o no deseable. (49)

3.6.1 Métodos de prueba

Las pruebas son de gran importancia en la garantía de la calidad del software. Los objetivos principales de realizar una prueba son:

1. Detectar un error
2. Tener un buen caso de prueba
3. Descubrir un error no descubierto antes

Los métodos de prueba del software tienen el objetivo de diseñar pruebas que descubran diferentes tipos de errores con menor tiempo y esfuerzo.

3.6.2 Diseños de casos de prueba

Se trata de diseñar pruebas que tengan la mayor probabilidad de encontrar el mayor número de errores con la mínima cantidad de esfuerzo y de tiempo. Cualquier producto de ingeniería se puede probar de dos formas:

Pruebas de caja negra: Realizar pruebas de forma que se compruebe que cada función es operativa.

Pruebas de caja blanca: Desarrollar pruebas de forma que se asegure que la operación interna se ajusta a las especificaciones, y que todos los componentes internos se han probado de forma adecuada.

En la prueba de la 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. (50)

En la prueba de caja blanca se realiza un examen minucioso de los detalles procedimentales los cuales comprueban los caminos lógicos del programa, así como los bucles y condiciones, examinado el estado del programa en varios puntos. (50)

Descripción general			
Permitir generar reporte de la cantidad de cursos asociados a los usuarios en el sistema.			
Condiciones de ejecución			
Para generar el reporte de la cantidad de cursos asociados a los usuarios hay que: - Estar autenticado en el sistema. - Debe existir en el sistema al menos un curso. - Debe existir en el sistema al menos un usuario asociado a un curso.			
SC 1 Generar el reporte de la cantidad de cursos asociados a los usuarios			
Escenario	**Descripción**	**Respuesta del sistema**	**Flujo central**
EC 1.1 Seleccionar la opción Generar reportes	Selecciona la opción Generar reportes.	El sistema muestra el listado de reportes que permite generar	Inicio/Reportes
EC 1.2 Seleccionar el reporte deseado	Permite seleccionar el reporte que el usuario desea generar	Se muestre un listado con los siguientes datos: - Usuario - Cantidad de cursos asociados	Inicio/Reportes/Reporte deseado
EC 1.3 Exportar a varios formatos	Se selecciona la opción de exportar al formato deseado por el usuario(pdf, odt, cvs, doc, ppt, xls)	El sistema exporta al formato seleccionado todos los datos del reporte que desee generar	Inicio/Reportes/Reporte deseado/Exportar

Tabla 6. Caso de pruebas generar reporte de la cantidad de cursos asociados a los usuarios en el sistema.

3.6.3 Resultados de las pruebas de caja negra

En la prueba de la 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 (50). En las pruebas realizadas al sistema mediante el método de caja negra, se pudo comprobar el cumplimiento de los requisitos funcionales y no funcionales del software obtenido.

A continuación, se presenta el resultado de las pruebas realizadas:

Iteraciones	***Cantidad de casos de prueba***	***No conformidades detectadas***			
		Alta	***Media***	***Baja***	***Total***
1	16	4	3	1	8
2	16	3	3	1	7
3	16	0	0	0	0

Tabla 7. Resultado de las pruebas de caja negra

Las no conformidades encontradas en las iteraciones 1 y 2 estuvieron relacionadas a la autenticación con el servidor y conexión a la base de datos del SDR, la configuración de algunos de los parámetros para realizar dicha conexión, la creación y visualización de los reportes generados, la actualización de los campos y validación de los mismos; fueron corregidos además algunos errores en la interfaz y en las funciones visibles al usuario. Debido al impacto que tenían en la calidad del componente, el equipo de trabajo se enfocó en solucionar dichas no conformidades, logrando en cada iteración de prueba disminuir el número de defectos en el componente. Al finalizar no quedaron no conformidades sin solucionar.

3.7 Conclusiones parciales

1. Como resultado de este capítulo se han modelado los artefactos relacionados con los flujos de trabajo implementación y pruebas.

2. Se describieron los estándares de codificación empleados en el desarrollo del módulo, con la utilización de los mismo se obtuvo un código de calidad.
3. Se mostraron algunos de los fragmentos de código de mayor importancia para el funcionamiento del módulo.
4. Para la validación del software se realizaron las pruebas al sistema mediante la técnica de caja negra, con esta se detectaron a tiempo un grupo de no conformidades en cada iteración, la solución de las mismas favorecieron la obtención de una mejor implementación de las funcionalidades.

Conclusiones generales:

Con la realización del presente trabajo se arribaron a las siguientes conclusiones:

1. El análisis de conceptos asociados al objeto de estudio, ayudó a seleccionar las herramientas para completar el desarrollo del módulo.
2. Una vez culminado el trabajo es posible afirmar que se cumplieron los objetivos trazados para el mismo:
    - Se realizó el análisis y diseño del módulo de reportes para la plataforma Zera 2.0.
    - Se realizó la implementación del módulo de reportes, el cual cumple con los requisitos identificados.
    - Se realizaron pruebas funcionales para validar el correcto funcionamiento del módulo de reportes.
3. Como producto final de la investigación se obtuvo el componente para la generación de reportes en la plataforma Zera 2.0, el mismo pude ser integrado con cualquier proyecto que tenga como línea base el marco de desarrollo Xalix.

Recomendaciones:

1. Para una nueva versión del módulo incorporar un sistema de filtrado a los reportes, como, por ejemplo: la cantidad de usuarios matriculados en un curso en específico, debido a la cantidad de información que pueden llegar a existir en la plataforma lo que podría hacer compleja la búsqueda de la información deseada.
2. Integrar solución con nueva versión en desarrollo del SDR.

Referencias bibliográficas

1. *Los Cursos en Línea Masivos y Abiertos (MOOC) como alternativa.* **González, Hector Matías.** s.l. : GECONTEC: Revista Internacional de Gestión del Conocimiento y la Tecnología, 2014, Vol. 2. ISSN 2255-5684.

2. **Dr Alvarez de Zayas, Carlos.** *Metodología de la Investigación Científica.* Santiago de Cuba : Centro de estudios de educación superior "Manuel F. Gran", 1995.

3. **Ennis Bouly, Yanet Pérez Solá, Nara Lidia Abreu Medina, Aldis Joan.** *Componentes para la Generación de Reportes Dinámicos en el GDR sobre Bases de Datos en Access y Oracle 11g.* 2015.

4. **Jasper.** JasperReport. *JasperReport.* [En línea] [Citado el: 15 de Marzo de 2016.] http://community.jaspersoft.com/project/jasperreports-library.

5. **github.** github. [En línea] [Citado el: 15 de marzo de 2016.] https://jdorn.github.io/php-reports/.

6. **Microsoft.** msdn.microsoft.com. [En línea] 2015. https://msdn.microsoft.com/es-es/library/dd409390.aspx.

7. **Report, Crystal.** Crystal Report. [En línea] [Citado el: 15 de marzo de 2016.] http://www.crystalbook.com/crtcr/program/book/intro_2008.asp.

8. **Report, Active.** Active Report. *Active Report.* [En línea] [Citado el: 15 de marzo de 2016.] http://activereports.grapecity.com/.

9. **Ricardo Grau, Cecilia Correa, Mauricio Rojas.** *Metodología de la Investigación.* s.l. : Fondo Editorial Coruniversitaria, 2004. ISBN: 958-8028-10-8.

10. **Hermenegildo, Romero.** http://es.slideshare.net. [En línea] 7 de Febrero de 2012. [Citado el: 12 de Febrero de 2015.] http://es.slideshare.net/MeneRomero/metodologias-de-desarrollo.

11. **Solís, Manuel Calero.** GoogleCode. [En línea] 2003. [Citado el: 13 de Marzo de 2016.] https://uma-cms.googlecode.com/svn/trunk/docs/ExplicaXp.pdf.

12. **Kniberg, Henrik.** *Scrum y XP desde las trincheras.* 207. ISBN: 978-1-4303-2264-1 .

13. **Schwaber, Ken y Sutherland, Jeff.** Scrum. [En línea] [Citado el: 13 de Marzo de 2016.] www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-ES.pdf.

14. **Sánchez, Tamara Rodríguez.** *Metodología de desarrollo para la Actividad productiva de la UCI.*

15. **Alegsa, Leandro.** Alegsa.com. [En línea] 12 de Mayo de 2010. [Citado el: 10 de Marzo de 2016.] http://www.alegsa.com.ar/Dic/framework.php.

16. **Potencier, Fabien y Zaninotto, François.** Sunshine. [En línea] [Citado el: 22 de Enero de 2016.] http://sunshine.prod.uci.cu/gridfs/sunshine/books/symfony_guia_definitiva.pdf.

17. **Laravel.** Laravel. [En línea] [Citado el: 15 de Marzo de 2016.] http://bundles.laravel.com/.

18. **Codeigniter.** Codeigniter. [En línea] [Citado el: 15 de marzo de 2016.] http://www.codeigniter.com/.

19. **Codeigniterb.** Codeigniterb.com. [En línea] [Citado el: 15 de marzo de 2016.] http://codeigniterb.com/caracteristicas-generales-de-codeigniter/.

20. **Garcia, Fernando.** fergarciac. [En línea] 25 de Enero de 2013. https://fergarciac.wordpress.com/2013/01/25/entorno-de-desarrollo-integrado-ide/.

21. **Jetbrains.** Jetbrains. [En línea] [Citado el: 26 de Enero de 2016.] https://www.jetbrains.com/phpstorm/.

22. *Herramienta de Desarrollo Netbeans.* **Mendoza González, Geovanny.**

23. **Harold, Elliotte Rusty y Means, W. Scott.** *XML.* s.l. : Anaya Multimedia-Anaya Interactiva, 2005. ISBN: 8441518122 ISBN-13: 9788441518124.

24. **Eguíluz Pérez, Javier.** Libros.es. [En línea] [Citado el: 15 de abril de 2016.] https://librosweb.es/libro/javascript/.

25. **w3c.** w3c. [En línea] [Citado el: 15 de Abril de 2016.] http://www.w3c.es/Divulgacion/GuiasBreves/HojasEstilo.

26. **Rosalba.** Arquitectura del computador. [En línea] [Citado el: 15 de Abril de 2016.] http://rosalba-rouse1.blogspot.com/.

27. **Bootstrap.** [En línea] [Citado el: 10 de Febrero de 2016.] http://getbootstrap.com.

28. **Postgresql.** Postgresql.org. [En línea] 10 de 2 de 2010. [Citado el: 20 de marzo de 2016.] http://www.postgresql.org.es/sobre_postgresql.

29. **Daniel Pecos Martínez.** geekWare. *PostGreSQL vs. MySQL.* [En línea] http://danielpecos.com/documents/postgresql-vs-mysql/.

30. **Pgadmin.** Pgadmin. [En línea] [Citado el: 20 de mayo de 2016.] http://www.pgadmin.org/index.php.

31. **PostgreSQL-Studio.** PostgreSQL Studio. [En línea] [Citado el: 16 de Mayo de 2016.] http://www.postgresqlstudio.org/about/index.html.

32. **Sourceforge.** sourceforge. [En línea] [Citado el: 16 de Mayo de 2016.] http://phppgadmin.sourceforge.net/doku.php.

33. **Gregorio, Jose.** slideshare. [En línea] 28 de Enero de 211. [Citado el: 10 de Mayo de 2016.] http://www.slideshare.net/josegregoriob/servidor-web-8451426.

34. **ACENS, [prod.].** *Servidor web Nginx, una clara alternativa a APACHE.* [Documento]

35. **Tomcat, Apache.** tomcat. [En línea] [Citado el: 12 de Marzo de 2016.] http://tomcat.apache.org/tomcat-7.0-doc/.

36. *Herramientas CASE para ingeniería de Requisitos.* **Alarcon, Andrea y Sandoval, Erika.** 6, Boyacá : s.n., 2008.

37. **Nazar, Fabian.** Slideshare.net. [En línea] [Citado el: 20 de marzo de 2016.] http://es.slideshare.net/fabiannazar1/uml-tutorialvisualparadigm.

38. **Jeremy Toeman.** Balsamiq. [En línea] Balsamiq Studios, 2008. [Citado el: 12 de Mayo de 2016.] https://balsamiq.com/products/mockups/.

39. **González, Anaisa Hernández.** rii.cujae.edu.cu. [En línea] Marzo de 2005. http://rii.cujae.edu.cu/index.php/revistaind/article/viewFile/129/114.

40. **Sommerville, Ian.** *Ingeniería del software.* s.l. : Pearson Educación, 2011. ISBN 978-607-32-0603-7.

41. **Ivar Jacobson, Grady Booch, James Rumbaugh.** *El proceso Unificado del desarrollo de software.* s.l. : Pearson Educación, 2000. ISBN: 84-7829-036-2.

42. **Cohn, Mike.** *User Stories Applied.* s.l. : Addison Wesley, 2004. ISBN 0-321-20568-5.

43. **Rumbaugh J., Jacobson I., Booch G.** *El lenguaje Unificado de Modelado. Manual de Referencia.* Madrid : Pearson Educación, 2000. ISBN: 84-7829-037-0.

44. **Potencier, F.** *Symfony la guía definitiva.* 2009.

45. **Visconti, Marcello y Astudillo, Hernán.** *Fundamentos de la Ingeniería de Software.* s.l. : Universidad Técnica Federico Santa María.

46. **JACOBSON, Ivar.** *A PPLYING UML IN T HE U NIFIED P ROCESS.* s.l. : Rational Software Corporation, 1999.

47. **Digital, Guia.** [En línea] [Citado el: 18 de abril de 2016.] http://www.guiadigital.gob.cl/articulo/que-es-una-interfaz.

48. **Pereda Díaz, Ernesto Vladimir y Miranda Pons, Yaismel.** *Pautas de codificación. Centro FORTES. Versión 1.0.* La Habana : Universidad de las Ciencias Informáticas.

49. **Software, Pruebas del.** pruebasdesoftware. [En línea] [Citado el: 12 de Abril de 2016.] http://www.pruebasdesoftware.com/laspruebasdesoftware.htm.

50. **Lemus, Guillermo.** es.slideshare.net. [En línea] 27 de abril de 2012. [Citado el: 14 de abril de 2016.] http://es.slideshare.net/GuillermoLemus/tipos-de-pruebas-de-software.

51. **Torossi, Gustavo.** *El Proceso Unificado de Desarrollo de Software.*

52. **Microsoft.** Microsoft. [En línea] [Citado el: 15 de marzo de 2016.] https://msdn.microsoft.com/es-es/library/ms159106(v=sql.120).aspx.

53. **Sánchez, Javier Zapata.** pruebasdelsoftware.wordpress.com. [En línea] Enero de 2013. [Citado el: 12 de Abril de 2016.] https://pruebasdelsoftware.wordpress.com/.

54. **Corporation, Oracle.** Bienvenido a NetBeans y www.netbeans.org. [En línea] NetBeans. [Citado el: 28 de 11 de 2013.] https://netbeans.org/index_es.html.

55. **Rumbaugh, James, Jacobson, Ivar y Booch, Grady.** *El Proceso Unificado de Desarrollo de Software.* Madrid : Addison-Wesley, 2000.