Facultad 2

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

MÓDULO DE GENERACIÓN DE REPORTES PARA LA PLATAFORMA DE SEGURIDAD EN LAS TECNOLOGÍAS DE LA INFORMACIÓN

Autores:

Acela María Sanamé Pérez

Manuel Hernández Montiel

Tutores:

Ing. Yeilenia Iris Pérez Vázquez

Ing. Luis Eduardo Gallardo Concepción

La Habana, junio de 2016

"Año 58 de la Revolución"

Todos y cada uno de nosotros paga puntualmente su cuota de sacrificio consciente de recibir el premio en la satisfacción del deber cumplido…"

Ché

Declaración de autoría

Declaramos que somos los únicos autores del trabajo de diploma Módulo para generar reportes en la Plataforma de la Seguridad en las Tecnologías de la Información y autorizamos al Centro de Telemática (TLM) de la Facultad 2 y a la Universidad de las Ciencias Informáticas (UCI) a hacer el uso del mismo en su beneficio.

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

Firma del Autor	**Firma del Autor**
Acela María Sanamé Pérez	Manuel Hernández Montiel
Firma del Tutor	**Firma del Tutor**
Ing. Yeilenia Iris Pérez Vázquez	Ing. Luís Eduardo Gallardo Concepción

Datos de contacto

Acela María Sanamé Pérez
Correo: amsaname@estudiantes.uci.cu
La Habana, Cuba

Manuel Hernández Montiel
Correo: mmontiel@estudiantes.uci.cu
La Habana, Cuba

Tutores

Ing. Yeilenia Iris Pérez Vázquez
Correo: yiperez@uci.cu
Universidad de las Ciencias Informáticas, La Habana, Cuba

Ing. Luís Eduardo Gallardo Concepción
Correo: legallardo@uci.cu

Universidad de las Ciencias Informáticas, La Habana, Cuba

Dedicatoria

Dedico este trabajo a toda mi familia y amigos, en especial a mis padres por haber sido mi inspiración y por quienes hoy me he convertido finalmente en profesional, por el amor que me han brindado y por toda la confianza que siempre han depositado en mí.

Acela María Sanamé Pérez

Dedico este trabajo a toda mi familia y amigos, en especial a mis padres y abuelos por haber sido mi inspiración y por quienes hoy me he convertido finalmente en profesional, por el amor que me han brindado y por toda la confianza que siempre han depositado en mí.

Manuel Hernández Montiel

Agradecimientos

Acela María Sanamé Pérez

A mis padres por el apoyo incondicional que me han brindado en todo momento.

A toda mi familia por preocuparse por mí y brindarme buenos consejos.

A mis compañeros de universidad que han compartido cinco años a mi lado en las buenas y en las malas.

A mis tutores por haberme ayuda a realizar mi sueño de ser ingeniera en las ciencias informáticas.

Manuel Hernández Montiel

A mis padres por el apoyo incondicional.

A mi hermana.

A toda mi familia en generar.

A mis amistades y profesores.

A mis tutores por haberme ayuda a realizar mi sueño.

Resumen

La generación de reportes es un tema común a tratar en el mundo de la informática, debido a que la mayoría de las herramientas de gestión de la información necesitan mostrar sus datos a los usuarios mediante la confección de informes o reportes.

Xilema-PlatSi es una plataforma de seguridad creada para realizar auditorías de aplicaciones web. La Plataforma actualmente no cuenta con una herramienta que le permita mostrar sus resultados mediante la confección de reportes capaces de mostrar la información de manera esclarecedora y entendible para los directivos de la Empresa de las Telecomunicaciones de Cuba. Teniendo en cuenta lo anteriormente planteado, se propone desarrollar un módulo para Xilema-PlatSI con el objetivo de generar reportes que muestren de manera específica e interpretativa la información contenida en la base de datos de la Plataforma, permitiendo que los directivos de la empresa para la que fue creada la aplicación, consulten la información de manera rápida y concisa. El proceso de desarrollo del módulo fue guiado mediante la metodología ágil XP que garantizó la simplicidad del mismo, para la implementación se utiliza el lenguaje de desarrollo PHP, haciendo uso del modelo arquitectónico de dos capas cliente-servidor. El resultado obtenido fue el Módulo para Generar Reportes de Auditorías Web en Xilema-PlatSI. Módulo que probado mediante la estrategia de pruebas definida, arrojó resultados satisfactorios demostrando estar capacitado para las funcionalidades requeridas.

Palabras clave*:* XILEMA-PlatSi, reporte, auditoria web, módulo.

Índice de contenido

Índice de tablas

Índice de figuras

Introducción

Durante el siglo XX, el desarrollo organizacional recorrió las fases más importantes para reconocer sus principales elementos motores: el hombre, el trabajo y el conocimiento. La transformación más relevante ocurrió en el sector empresarial, donde se comenzaron a idear mecanismos que posibilitaran el aumento de la eficiencia laborar y el correcto funcionamiento organizacional (Visbal, 2009).

La informatización de los procesos habituales en el sistema empresarial, ha sido una de las estrategias ideadas por el hombre para, haciendo uso de las nuevas tecnologías, responder a las necesidades y requerimientos de los sistemas. Una de las necesidades surgidas en las instituciones es la de gestionar toda la información que se genera en los sistemas implementados, permitiendo conocer el comportamiento histórico de la institución a través de la estadística, reflexionar sobre objetivos y metas, pronosticar resultados y reconsiderar paradigmas actuales; con el fin de facilitar la toma de decisiones empresariales (Cañevate, 2007).

Escoger la mejor alternativa de entre las posibles, necesita información sobre cada una de ellas y sus consecuencias respecto al objetivo de la empresa. La toma de decisiones es un factor decisivo para definir el futuro de las instituciones y obtener los resultados esperados, pero para ello, se ha de contar con sistemas de información que permitan identificar y resolver los problemas que se les presenta a toda organización (Canós, 2012).

En la actualidad, existen infinidades de sistemas informáticos para gestionar la información en las empresas. La mayoría de estos sistemas, arrojan sus resultados mediante la confección de reportes. Los reportes son parte de la respuesta a la necesidad de una mayor responsabilidad ante los grupos de interés y de un mejoramiento en la transparencia de las relaciones de las empresas con estos grupos. Es cada vez mayor la necesidad de generar reportes para las empresas con el objetivo de ayudar en la toma de decisiones, medir el trabajo que se está realizando y brindar, de la mejor forma, el conocimiento a los clientes.

Generar reportes persigue la creación de una administración eficiente, que satisfaga las necesidades reales de los usuarios, tales como: poder almacenar la información o realizar reportes basados en distintos indicadores. Este proceso se hace posible con la introducción de mecanismos que promuevan el desarrollo de servicios de mayor calidad, además de sistemas de control de la

información que otorguen una plena transparencia en el análisis de datos para perfeccionar el sistema de información mejorando la agilidad y uso de la misma.

Cuba es un país que se encuentra en proceso de informatización, debido a la gran cantidad de información que deben de procesar las empresas nacionales. En gran medida estas instituciones, se apoyan en sistemas informáticos que les permitan proporcionar información útil y en tiempo real al personal. Muchos de estos sistemas informáticos son desarrollados en la Universidad de las Ciencias Informáticas (UCI), cuya misión es formar profesionales comprometidos con su Patria, altamente calificados en la rama de la informática, producir aplicaciones y servicios informáticos a partir de la vinculación estudio-trabajo como modelo de formación y servir de soporte a la industria cubana de la informática (UCI, 2012). La producción de dicha universidad cuenta con una amplia red de centros productivos dentro de los que se encuentra el Centro de Telemática (TLM), especializado en sistemas y servicios informáticos en la rama de las telecomunicaciones y la seguridad informática. Dentro de los proyectos desarrollados por el centro se encuentra la Plataforma de Seguridad en las Tecnologías de la Información o Xilema–PlatSI, proyecto en desarrollo, que permite detectar e informar posibles brechas de seguridad, mediante la integración de herramientas de pruebas.

La Plataforma tiene como objetivo, integrar herramientas de seguridad que faciliten la detección de vulnerabilidades que puedan originar un fallo en los pilares de la seguridad informática (Confidencialidad, Integridad y Disponibilidad de la información) (Concepción., 2016). El sistema informático[1] se desarrolla a solicitud de la Empresa de Telecomunicaciones de Cuba (ETECSA) para la realización de auditorías a las aplicaciones web y sistemas de información que son alojados en sus Centros de Datos.

La Plataforma está compuesta por 3 subsistemas:

- ✓ Subsistema de Administración, que garantiza el control de acceso a la Plataforma, la gestión de usuarios y los permisos para el acceso de estos, además de la configuración de herramientas y tareas de administración.

[1] Sistema. Conjunto ordenado de normas y procedimientos que regulan el funcionamiento de un grupo o colectividad.
Sistema informático. Un **sistema informático** (SI) es un **sistema** que permite almacenar y procesar información; es el conjunto de partes interrelacionadas: hardware, software y personal **informático**.

- ✓ Subsistema de Auditorías, encargado de brindar la posibilidad de ejecutar las herramientas Acunetix, Nikto, W3af, Arachni y Zap para las pruebas de seguridad de forma remota, además da a conocer en tiempo real el estado de las pruebas (ejecución, detenidas, canceladas).

- ✓ Subsistema de Reporte, el cual permite generar un informe técnico con los resultados obtenidos en las pruebas de seguridad realizadas. Esta muestra los tipos de vulnerabilidades que afectan al sistema y el impacto que pueden ocasionar, además evalúa el nivel de seguridad del sistema según los resultados generados.

A pesar de que la solución tiene un impacto considerable en ETECSA, no cuenta con un mecanismo que, de manera rápida y efectiva, permita consultar información específica sobre indicadores como: tiempo, cantidad de auditorías, estado de las auditorías, usuarios del sistema, informes, entre otros. Otra de las limitantes que presenta la plataforma es la forma de visualizar la información, que se realiza mediante tablas, resultando inadecuado para los directivos con tiempo escaso y dificultando la búsqueda en grandes volúmenes de datos, obstaculizando la toma de decisiones en la empresa en cuanto al desempeño de sus trabajadores y el control de sus procesos de auditoría.

En base a lo anterior se plantea como **problema de la investigación:** ¿Cómo contribuir, a partir de la información contenida en PLATSI, a la toma de decisiones de los directivos del Departamento de Seguridad Informática de ETECSA?

Se define como **objeto de estudio:** Los procesos de generación de reportes en Sistemas de Gestión de Información, como **objetivo general:** Desarrollar el módulo para generar reportes en la Plataforma de Seguridad en las Tecnologías de la Información, en el **campo de acción:** El proceso de generación de reportes en la Plataforma de Seguridad en las Tecnologías de la Información. Para dar solución al problema se define como **objetivos específicos:**

- ✓ Fundamentar la investigación, a partir del análisis de conceptos relacionados con el tema de generación de reportes.
- ✓ Realizar un estudio de sistemas similares en el contexto nacional e internacional.
- ✓ Fundamentar la metodología de desarrollo y las herramientas y tecnologías a utilizar para la implementación del módulo.
- ✓ Analizar las distintas fases de desarrollos establecidas por la metodología seleccionada.
- ✓ Implementar el módulo de generación de reportes para PlatSI.

- ✓ Aplicar una estrategia de pruebas para validar las funcionalidades del módulo.

Para dar cumplimiento a los objetivos específicos anteriormente planteado, se definen las siguientes **tareas de investigación:**

- ✓ Análisis de conceptos relacionados con el tema para una mejor interpretación.
- ✓ Análisis de distintas herramientas existentes en el ámbito nacional e internacional para generar reportes.
- ✓ Justificación de la metodología de desarrollo a utilizar para guiar las fases de desarrollo del módulo.
- ✓ Selección del lenguaje de programación y tecnologías a utilizar para implementar el módulo.
- ✓ Presentación de la propuesta de solución para responder a la problemática planteado.
- ✓ Generación de los artefactos que establece cada una de las fases de desarrollo de la metodología para caracterizar el módulo.
- ✓ Selección de la arquitectura de software para el diseño del módulo.
- ✓ Definición de los patrones de diseño más adecuados para la construcción del módulo propuesto.
- ✓ Definición de la estrategia de prueba a seguir para verificar el correcto funcionamiento del módulo.

Para el desarrollo de la investigación se hizo uso de los siguientes **métodos científicos**.

Métodos teóricos

Analítico-Sintético: Se empleó para identificar los conceptos y las definiciones más importantes relacionadas con el tema de reportes, así como para analizar las tecnologías y herramientas a utilizar.

Histórico-Lógico: Se empleó para estudiar lo más notable en el plano teórico acerca de las herramientas informáticas que permitan general reportes. También se utilizó para conocer los tipos de reportes existentes y para estudiar el estado actual de sistema Xilema-PlatSi.

Métodos empíricos

Revisión de Documentos: Se utilizó para lograr una mayor comprensión referente al contenido de generación de reportes, así como para definir una forma de trabajo.

Para lograr una mayor comprensión de la investigación realizada se decidió estructurarlo de la siguiente manera:

Capítulo 1: "Fundamentación Teórica"

En este capítulo se hace un análisis de los principales conceptos relacionados con el objeto de estudio y el campo de acción. Además, contiene un análisis de aplicaciones similares vinculadas al campo de acción. También presenta la metodología de desarrollo de software utilizada además de las herramientas, lenguajes y tecnologías a tener en cuenta para el proceso de desarrollo del sistema.

Capítulo 2: "Características del sistema"

En este capítulo se presenta la propuesta de solución a la problemática anteriormente planteada, se desglosan las características funcionales que debe cumplir el sistema y se muestran las historias de usuario. Se estima y planifica el desarrollo del módulo mediante la creación de los artefactos correspondientes a las fases de desarrollo que plantea la metodología XP.

Capítulo 3: "Diseño del módulo"

En este capítulo se define la arquitectura, el patrón de arquitectura y los patrones de diseño a utilizar. Se definen los patrones de diseño a utilizar y se elaboran las tarjetas Clase-Responsabilidad-Colaborador (CRC).

Capítulo 4: "Implementación y Prueba"

En este capítulo se realizan las tareas de ingeniería a implementar, se definen los estándares de codificación que identificarán el código implementado, se realiza la implementación del módulo y se traza una estrategia de prueba para validar el trabajo realizado.

Capítulo 1: Fundamentación teórica

1.1 Introducción

En el presente capítulo se realiza un estudio de la fundamentación teórica de la investigación que se desarrolla, tratando los conceptos fundamentales que se manejan durante el proceso de elaboración del sistema. Se plantea el estado actual en que se encuentra la Plataforma de la Seguridad de las Tecnologías de la Información y su funcionamiento. Se justifica la metodología de desarrollo a utilizar para guiar la realización del proyecto y se definen las distintas herramientas y tecnologías que se utilizarán en el desarrollo del módulo.

1.2 Conceptos Fundamentales

Reporte

El reporte es un documento informativo que sirve para comunicar información que sea relevante, este puede ser un material creado por una empresa, organización o un trabajo de clases que sirve para dar mayor información sobre un tema en específico, por otro lado, también la información que se divulga a través de un medio de comunicación, este puede ser visual o textual (Reserved, 2011).

Reportes informáticos

Los reportes informáticos son informes que organizan y exhiben la información contenida en la base de datos (Reserved, 10).

Auditoría

Existen numerosas definiciones…La Real Academia Española de la Lengua define como “una revisión sistemática de una actividad o de una situación para evaluar el cumplimiento de las reglas o criterios objetivos a las que aquellas deben someterse” (Rosa, 2005).

Auditoría web

Una auditoría web es dar a conocer el estado actual de la página que se está verificando en cuanto a las vulnerabilidades que en ellas se encuentran, dando como resultado una serie de reportes que ayudarán al web master o al propietario a comprender varias áreas del sitio como son la seguridad, la usabilidad, contenido o calidad. La auditoría web ayuda a conocer el estado actual de la página web ya que, analiza principalmente la usabilidad, optimización y seguridad de la aplicación (Pascual, 2007).

Seguridad informática

La seguridad informática es la disciplina que se ocupa de diseñar las normas, procedimientos, métodos y técnicas, orientados a proveer condiciones seguras y confiables, para el procesamiento de datos en sistemas informáticos. Consiste en asegurar que los recursos del sistema de información (material informático o programas) de una organización sean utilizados de la manera que se decidió y que el acceso a la información allí contenida, así como su modificación, sólo sea posible a las personas que se encuentren acreditadas y dentro de los límites de su autorización (MENDOZA, 2011).

Tipos de gráficos que se visualizarán en el módulo

Gráfico de columna

Un gráfico de barras es una representación gráfica en un eje cartesiano de las frecuencias de una variable cualitativa o discreta. En uno de los ejes se posicionan las distintas categorías o modalidades de la variable cualitativa o discreta y en el otro el valor o frecuencia de cada categoría en una determinada escala (Estadísticas, 2016).

Gráfico de líneas

Un gráfico de líneas es una representación gráfica en un eje cartesiano de la relación que existe entre dos variables reflejando con claridad los cambios producidos. Se suelen usar para presentar tendencias temporales. En el eje horizontal se ha de posicionar la variable que indica las unidades de tiempo y en el vertical se introduce la escala de la variable cuya variación en el tiempo (Estadísticas, 2016).

Gráfico de pastel

Un diagrama de pastel es un círculo dividido en partes, donde el área de cada parte es proporcional al número de datos de cada categoría. La gráfica de pastel se usa para representar variables cualitativas o categóricas, de preferencia nominales. Se utiliza para mostrar la proporción le corresponde a cada categoría (Universidad Nacional Autónoma de Mexico, 2015).

1.3 Estado actual del sistema XILEMA-PlatSI

Xilema-PlatSI integra varias herramientas de detección de vulnerabilidades y gestiona todo el proceso de auditoría a sistemas informáticos. Actualmente gestiona el proceso de auditorías a sistemas web, e integra las herramientas de penetración a sistemas web Acunetix, Nikto, W3af, Arachni y Zap.

Estas herramientas, reportan las vulnerabilidades detectadas en los sistemas e incluyen información sobre elementos del sistema auditado que no son vulnerabilidades pero que pudieran influir en la seguridad, por ejemplo: la utilización de versiones obsoletas de librerías o programas, paquetes instalados con vulnerabilidades conocidas y reportadas. También puede ocurrir que las herramientas detecten una vulnerabilidad en los sistemas que no comprometa su seguridad debido a que la infraestructura donde se encuentra desplegado el sistema no permite que sea explotada durante un ataque informático.

Dentro de los roles que interactúan con el sistema está el rol especialista. Los procesos que realiza un especialista al encontrar una vulnerabilidad es probar que realmente pueda ser explotada durante un ataque informático. La información detallada de las vulnerabilidades es fundamental para saber cómo y dónde un sistema puede ser vulnerable. Datos como la clasificación o la propia descripción de la vulnerabilidad ayudan a enfocarse en los lugares en los que un posible atacante pueda afectar la disponibilidad, confidencialidad e integridad de un sistema. No todas las herramientas muestran la clasificación y las descripciones de las vulnerabilidades, el especialista debe describirlas de forma única por cada auditoría que realice, ya que estas descripciones no son almacenadas por Xilema-PlatSI.

Actualmente la Plataforma, brinda la posibilidad de crear un informe con los resultados arrojados por las herramientas integradas en cada auditoría, luego los especialistas de seguridad informática deben realizar el análisis manual de cada uno de estos resultados y definir si son o no vulnerabilidades con el suficiente impacto para clasificar el sistema como vulnerable.

1.3.1 Funcionamiento del sistema XILEMA-PlatSI

PlatSI, cuenta con cinco herramientas para realizar las auditorias webs: Acunetix, Nikto, W3af, Arachni y Zap. En ellas, interactúan diferentes roles como: planificador, especialista y solicitante, encargados de realizar el proceso de auditoría. Cada rol tendrá acceso solo a la información con la que deben interactuar.

El proceso de auditoría comienza con el rol solicitante, encargado de crear la solicitud. Inicialmente esta se encuentra en estado PENDIENTE y no pasará al estado ATENDIDA hasta que el planificador cree la auditoría general. Al crear la auditoría general, esta se encuentra en estado CRAEDA y una vez se realice la asignación a uno o varios especialistas, así como al responsable de la misma, esta pasa al estado INICIADA.

Las asignaciones cuentan de tres estados, NUEVA, ACEPTADA y RESUELTA. En un principio la asignación está en estado NUEVA, previamente a que el especialista comience a crear la auditoría, este modifica su estado a ACEPTADA y la auditoría general pasará a estado EJECUTANDO. Seguidamente el especialista procederá a ejecutar las herramientas de pruebas de seguridad luego editará las vulnerabilidades y realizará su correlación. Luego de terminada la auditoría, el especialista modificará la asignación como resuelta y la auditoría general como COMPLETADA. Culminado el trabajo, el especialista creará un informe técnico con las vulnerabilidades detectadas. En caso en que la auditoría general no se encuentre en estado CREADA en su fecha fin establecida, ésta pasará a estado RETRASADA. El proceso finalizará con el rol responsable, quien estará encargado de confeccionar y exportar a PDF[2], un informe final que contendrá toda la información obtenida sobre la auditoría realizada.

1.4 Herramientas para generar reportes a escala internacional

Tableau (Public)

Tableau, es una herramienta de visualización de datos. Cuenta con una arquitectura de cliente-servidor de n niveles altamente escalable que presta servicios a clientes móviles, clientes web y software instalado en equipos de escritorio. Incluye un número de conectores de datos optimizado para bases de datos como Microsoft Excel, SQL Server, Oracle, Teradata y muchas más. La API[3] de la extracción de datos de Tableau viene incorporada en el SDK de Tableau. El SDK[4] de Tableau funciona en sistemas operativos como Windows, Mac OS X (10.9 y posteriores) y en las siguientes distribuciones de Linux: Fedora 18 y posteriores, CentOS 6.5 y posteriores, Ubuntu 12.04 y posteriores. El SDK admite los idiomas siguientes: C, C++, Java y versiones de Python 2.6 y 2.7, no es compatible con Python 3.3. Es una herramienta privada con versión de pago bastante elevada.

[2] PDF (siglas en ingles de Portable Document Format) es un formato de almacenamiento para documentos digitales independiente.

[3] API (siglas en ingles de Application Programming Interface) es un conjunto de funciones y procedimientos que cumplen uno o más funciones con el fin de ser utilizadas por otro software.

[4] SDK (siglas en ingles de Software Development Kit) es un Kit de desarrollo de software mediante el cual se puede desarrollar aplicaciones.

Kit. Es un conjunto de herramientas de desarrollo de software que le permita al programador crear aplicaciones para un sistema en concreto.

La versión gratuita de Tableau Public no permite el manejo de un grupo amplio de información, además corre solo sobre Windows, inconveniente que no permite que sea utilizada en PlatSI que corre sobre GNU/Linux (Abella, 2016).

amCharts

AmCharts está centrada en el desarrollo de herramientas, sobre todo orientado al desarrollador de programación para la visualización de datos mediante tablas y mapas. Está centrando todo el desarrollo en torno a JavaScript y tecnologías relacionadas. Como estándar multiplataforma, JavaScript es apoyado por gran mayoría de dispositivos y plataformas y navegadores móviles.

Todos los productos amCharts están disponibles como descarga gratuita y la versión comercial disponible para su compra. Versiones libres son completamente funcionales. La única diferencia entre sus homólogos comerciales es que muestran un pequeño enlace de la marca. El doble de licencias va a la par con amCharts, la filosofía de hacer sus productos con todas las funciones disponibles para todos, incluso aquellos que no brindaría normalmente adquirir una licencia comercial (Martinez, 2010).

1.4.1 Herramientas para generar reportes utilizadas a escala nacional

Generador Dinámico de Reporte (GDR)

El Generador Dinámico de Reportes fue creado en la Universidad de las Ciencias Informáticas. Es una aplicación web para la gestión de la información de cualquier empresa o institución, mediante el cual muestra el cumplimiento de las metas establecidas y con ello contribuir a la toma de decisiones. Posibilita diseñar reportes tabulares (con gráficos incluidos), tabla pivote y cruzada desde los gestores de Bases de Datos: SQL Server, SQLite, Oracle, MySQL y PostgreSQL. Dicha herramienta permite a los usuarios abstraerse de los conocimientos relacionados con los gestores de bases de datos y generar reportes en varios formatos con gran variedad de opciones en su diseño, marcando una diferencia entre los reportes tradicionales y los reportes dinámicos.

El producto ha sido desarrollado utilizando el lenguaje de programación PHP[5], el framework Symfony v1.2.7 y para el diseño de su interfaz gráfica de usuario se utilizó Ext Js[6] 2.2, lo cual a pesar de ser una aplicación web le confiere características muy similares a las de cualquier aplicación de escritorio, facilitando así la interacción de los usuarios con esta. Consta además de una arquitectura

[5] PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de programación web de código abierto.

[6] Ext Js (pronunciado como "ekst") es una biblioteca de JavaScript para el desarrollo de aplicaciones web.

cliente-servidor basada en el modelo Modelo-Vista-Controlador (MVC), así como de varias pluing[7] de integración con otras aplicaciones (UCI, 2012).

El GDR se compone de seis módulos dentro de lo que se encuentra: Diseñador de reportes y Visor de reportes, cada uno contienen sus funcionalidades específicas. El Diseñador de reportes posee gran flexibilidad permitiendo al usuario final el diseño de reportes personalizados desde la web. Para exportar a formatos PDF los reportes diseñados, es necesario contar con el módulo Visor de reportes, quien es encargado de visualizar, exportar y filtrar reportes diseñados en el módulo anterior. Por lo planteado anteriormente se descarta al GDR de ser utilizado en PlatSI, ya que lo que el cliente pide es un módulo de reporte que contenga en el mismo módulo las funcionalidades generar, visualizar y exporte los reportes solicitados.

GEReport v1.0

GEReport v1.0 fue creada en La Universidad de Cienfuegos, es una herramienta capaz de construir reportes dinámicos que cumplan con los estándares mundiales de estos tipos de aplicaciones, y cuya utilización no se restringiese debido a las licencias de software, se ejecutase sobre un entorno web, se minimizaran las dependencias tecnológicas y fuese multiplataforma.

La arquitectura que utiliza es Cliente-Servidor. El lado del cliente se escribió utilizando código JavaScript, para lo cual, se usó Dojo[8] 1.8, además, se utilizaron lenguajes como XHTML y CSS para el diseño de las vistas. El lado del servidor se desarrolló mediante código escrito en lenguaje PHP, para lo cual se utilizó CodeIgniter como marco de trabajo, característica que impide su utilización en PlatSi ya que esta utiliza el marco de trabajo Symfony2 a solicitud del cliente (GeReport: Sistema de Gestión de Reportes Dinámicos ., 2014).

1.4.2 Herramienta seleccionada

Después de haber realizado un estudio a escala nacional e internacional a un pequeño grupo de herramientas para generar reportes, se selecciona amCharts a utilizar en la realización del módulo. amCharts puede ser integrado a Symfony2 mediante amCharts Bundle[9] que le proporcionando

[7] Pluing. Es un programa que incrementa o aumenta las funcionalidades de un programa principal.

[8] Dojo. Es un framework para el desarrollo de aplicaciones web.

[9] Bundle. Un **bundle** es un concepto similar al de los plugins en otras aplicaciones, pero todavía mejor.

extensiones Twig[10] y objetos de PHP para hacer el trabajo pesado. El paquete incorpora la biblioteca de gráficos amCharts JS y resecar su código gráfico escribiendo todo en PHP.

1.5 Metodología de Desarrollo

Se entiende por metodología de desarrollo una colección de documentación formal referente a los procesos, las políticas y los procedimientos que intervienen en el desarrollo del software (2008). La finalidad de una metodología de desarrollo es garantizar la eficacia (p.ej. cumplir los requisitos iniciales) y la eficiencia (p.ej. minimizar las pérdidas de tiempo) en el proceso de generación de software. A continuación, se justifica la metodología que se utilizará para guiar el desarrollo del módulo.

1.5.1 Metodología seleccionada XP

XP, es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores y propiciando un buen clima de trabajo. Dentro de las características por las que el equipo de desarrollo escoge dicha metodología se tiene que; el equipo de desarrollo está compuesto por dos integrantes. Permite la realización de ciclos muy cortos tras los cuales se muestran resultados, minimizando el tener que rehacer partes que no cumplen con los requisitos y ayudando a los programadores a centrarse en los que son más importantes. El cliente y sus exigencias juegan un papel clave, por lo que esta metodología ofrece una gran flexibilidad ante el constante cambio de los requisitos. Se enfoca más en el desarrollo del software que en la documentación, ya que el proyecto es pequeño y cuenta de poco tiempo para su desarrollo. Permite definir en cada iteración las historias de usuario a desarrollar (Rodrígez, 2014).

1.6 Herramientas asociadas al desarrollo de software

1.6.1 Lenguaje de programación

Un lenguaje de programación consiste en todos los símbolos, caracteres y reglas de uso que permiten a las personas "comunicarse" con las computadoras. Existen varios cientos de lenguajes y dialectos de programación diferentes. Algunos se crean para una aplicación especial, mientras que otros son herramientas de uso general más flexibles que son apropiadas para muchos tipos de aplicaciones. En todo caso los lenguajes de programación deben tener instrucciones que pertenecen a las categorías ya familiares de entrada/salida, cálculo/manipulación de textos, lógica/comparación y

[10] Twig. Es un motor de creación de plantillas para utilizar con PHP.

almacenamiento / recuperación (Thomas, 2007). Los siguientes lenguajes se utilizarán en la confección del módulo:

Lenguaje de marcas de hipertexto: HTML 5

HTML5 es el sucesor del HTML4.01 y es el último lenguaje de marcado de hipertexto[11] para los sitios web desarrollados por World Wide Web Consortium (W3C) y Grupo Web de Hipertexto Aplicación de Tecnología de trabajo (WHATWG). Dentro de las características que se aprovechan de este lenguaje están que: Funciona en conjunto con CSS3 y todavía sigue en desarrollo, reduce la necesidad de plugins externos, mejor manejo de errores, es independiente de dispositivos y el proceso de desarrollo es público. Hoy en día todos los grandes navegadores web (Chrome, Mozilla Firefox, Safari, Opera, Internet Explorer) ofrecen compatibilidad con HTML5 (Thomas, 2007).

Lenguaje de hojas de estilo: CSS3

El CSS sirve para definir la estética de un sitio web en un documento externo y eso mismo permite que modificando ese documento (la hoja CSS) podamos cambiar la estética entera de un sitio web, el mismo sitio web puede variar totalmente de estética cambiando solo la CSS, sin tocar para nada los documentos HTML, jsp[12] o asp[13] que lo componen (Thomas, 2007).

Dentro de las ventajas que ofrece CSS3 está que:

- ✓ Se obtiene un mayor control de la presentación del sitio al poder tener todo el código CSS reunido en uno, lo que facilita su modificación.
- ✓ Pueden mostrarse distintas hojas de estilo según el dispositivo que se está utilizando.
- ✓ Es estrechamente compatible con HTML5.

Lenguaje de Programación: PHP v5.4.44

PHP, es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos.

[11] Hipertexto. Sistema de organización y presentación de datos que se basa en la vinculación de fragmentos textuales o gráficos a otros fragmentos, lo cual permite al usuario acceder a la información no necesariamente de forma secuencial sino desde cualquiera de los distintos ítems relacionados.

[12] Jsp (siglas en ingles de JavaServer Pages) es una tecnología que ayuda a los desarrolladores de software a crear páginas web dinámicas basadas en HTML, XML, entre otros tipos de documentos. JSP es similar a PHP, pero usa el lenguaje de programación Java.

[13] Asp (siglas en ingles de Active Server Pages) es la tecnología desarrollada por Microsoft para la creación de páginas dinámicas del servidor.

El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página Web resultante.

PHP se considera uno de los lenguajes más flexibles, potentes y de alto rendimiento conocidos hasta el día de hoy lo que ha atraído el interés de múltiples sitios con gran demanda de tráfico. Lo mejor de utilizar dicho lenguaje es su extrema simplicidad para el principiante, pero a su vez ofrece muchas características avanzadas para los programadores profesionales. Otras de las ventajas que ofrece PHP es que: Consta de muchísimas librerías externas que facilitan parte del trabajo y es un lenguaje multiplataforma que correr sobre las dos plataformas más importantes, Linux y Microsoft (Thomas, 2007).

1.6.2 Framework

El uso de un framework puede servirle a un equipo de desarrolladores para recortar tiempo con un resultado mejor: código de mayor calidad, proyectos escalables, aplicaciones web rápidas y robustas. Usar un framework permite tener ordenado en carpetas todo el código, disponer de librerías y funcionalidades que enriquecerán el proyecto final y aumentar la seguridad (Gallego, 2010). A continuación, se muestra los framework de desarrollos a utilizar en la realización del módulo:

Symfony v2.3.9

Symfony es un framework para el desarrollo de aplicaciones web mediante un Modelo-Vista-Controlador. Es un marco distribuido bajo una licencia de código abierto MIT. Está desarrollado por completo en PHP, se puede ejecutar en plataformas UNIX y Windows y es compatible con la mayoría de los sistemas de gestión de bases de datos más conocidos: MySQL, Microsoft SQL Server (Gallego, 2010).

Características:

- ✓ Permite el cambio de sistema de gestión de base de datos en cualquier momento del desarrollo.
- ✓ Usa programación orientada a objetos.
- ✓ Usa el patrón Modelo-Vista-Controlador.
- ✓ Es un marco de desarrollo que facilita mucho la interoperabilidad.
- ✓ Tiene un sistema de caché basado en HTTP que mejora el rendimiento de las aplicaciones desarrolladas con Symfony.

Symfony2 contiene un lenguaje de plantillas aún más potente llamado Twig. *Twig* te permite escribir plantillas concisas y fáciles de leer para los diseñadores web y, de varias maneras, más poderosas que las plantillas PHP. Twig viene con una larga lista de etiquetas y filtros que están disponibles de forma predeterminada. Incluso se puede agregar propias extensiones a Twig, según sea necesario.

JavaScript

JavaScript es un lenguaje que puede ser utilizado por profesionales y para quienes se inician en el desarrollo y diseño de sitios web. No requiere de compilación ya que el lenguaje funciona del lado del cliente, los navegadores son los encargados de interpretar estos códigos. Muchos confunden el JavaScript con el Java, pero ambos lenguajes son diferentes y tienes sus características singulares. JavaScript tiene la ventaja de ser incorporado en cualquier página web, puede ser ejecutado sin la necesidad de instalar otro programa para ser visualizado (Valdéz, 2007).

Framework JavaScript: jQuery v1.9.2

jQuery es una biblioteca de JavaScript, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM[14], manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX[15] a páginas web. En el desarrollo del módulo se utilizará jQuery para incorporar elementos tales como menús desplegables o de selección múltiple (Gallego, 2010).

Tanto jQuery es un software libre y de código abierto distribuido por la Fundación jQuery posee un doble licenciamiento bajo la Licencia MIT[16] y la Licencia Pública General de GNU, permitiendo su uso en proyectos libres y privados.

Framework de CSS: Bootstrap v2.3.2

Bootstrap es un framework o conjunto de herramientas de Código abierto para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como, extensiones de JavaScript opcionales adicionales.

[14] DOM (siglas en ingles de Document Object Model) es un conjunto de utilidades específicamente diseñadas para manipular documentos XML.

[15] AJAX (acrónimo de Asynchronous JavaScript and XML) es una técnica que permite la comunicación asíncrona entre un servidor y un navegador en formato XML mediante programas escritos en JavaScript**.**

16 MTI (siglas en ingles Massachusetts Institute of Technology) es una de las tantas licencias de software.

Existe un concepto de compatibilidad parcial que hace disponible la información básica de un sitio web para todos los dispositivos y navegadores. Por ejemplo, las propiedades introducidas en CSS3 para las esquinas redondeadas, gradientes y sombras son usadas por Bootstrap a pesar de la falta de soporte de navegadores antiguos. Esto extiende la funcionalidad de la herramienta, pero no es requerida para su uso (Gallego, 2010).

1.6.3 Entorno de Desarrollo Integrado (IDE)

Un entorno de desarrollo integrado, es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (wordpress). El siguiente IDE es el que se utilizará para desarrollar el módulo:

PhpStorm v9.0

JetBrainsPhpStorm es un IDE comercial, multi-plataforma para PHP desarrollado sobre la plataforma JetBrainsIntelliJ IDEA. PhpStorm se basa en IntelliJ IDEA, que está escrito en Java. Los usuarios pueden ampliar el IDE mediante la instalación de plugins creados para la plataforma de IntelliJ o escribir sus propios plugins.

Todas las funciones disponibles en WebStorm[17] se incluyen en PhpStorm, que añade soporte para PHP y bases de datos, barcos WebStorm con plugins preinstalados JavaScript (como Node.js), que están disponibles para PhpStorm sin costo.

PhpStorm está disponible para los desarrolladores individuales, y las empresas y organizaciones. Licencias reducidas y gratuitas adicionales están disponibles para nuevas empresas, estudiantes y profesores, y los proyectos de código abierto no comerciales. Estas licencias gratuitas requieren aprobación y no incluyen la cláusula de retorno perpetuo.

1.6.4 Sistema Gestor de Base de Datos

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

17 WebStorm. Es un IDE para JavaScript.

seguridad e integridad de los mismos (Avila). El SGBD relacionado con el módulo de gestión de reportes es:

PostgreSQL v9.1

PostgreSQL es un Sistema de gestión de bases de datos relacional orientado a objetos, libre y publicado bajo la licencia PosgreSQL1, similar a la Bases de Datos o la MIT. Utiliza un modelo cliente-servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. Este es el mismo sistema gestor de base de datos utilizado en la versión actual de Xilema-PlatSI, por lo que será con este con quien interactúe el módulo.

1.6.5 Herramienta CASE

Visual Paradigm v8.0

Es una herramienta CASE que soporta el modelado mediante UML y proporciona asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del Ciclo de Vida de desarrollo de un Software (Ingeniería de Software, 2014). Visual Paradigm ha sido concebida para soportar el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas. Se caracteriza por ser un sistema multiplataforma, distribuido bajo una licencia comercial y gratuita. Dicha herramienta se selecciona en su versión 8.0 como herramienta de apoyo, para realizar el diagrama de clases que se visualizarán más adelante.

Lenguaje Unificado de Modelado (UML)

Prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos y describe la semántica esencial de estos diagramas y los símbolos en ellos utilizados. Es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software (Orallo, 2015). Este lenguaje permite diseñar diferentes tipos de diagramas como, diagramas de clases.

1.6 Conclusiones Parciales

A partir del estudio realizado a Xilema-PlatSI y su funcionamiento, se conoció el estado en que se encuentra actualmente la plataforma. Se trataron diferentes conceptos relacionados con el objeto de estudio y el campo de acción, conceptos que ayudaron en la interpretación del tema a tratar en el transcurso del trabajo. Se realizó un análisis de distintas herramientas que pudieran dar solución a la

problemática planteada donde el resultado obtenido fue, seleccionar la biblioteca amCharts para realizar los gráficos que se representarán en el módulo Se justificó la metodología a utilizar para guiar el desarrollo del proyecto y finalmente se seleccionaron las herramientas y tecnologías a utilizar para crear el Módulo para Generar Reportes en la Plataforma de la Seguridad de la Información.

Capítulo 2: Características del sistema

2.1 Introducción

En el presente capitulo se abordarán las características del módulo a desarrollar. Se define la propuesta de solución y las historias de usuarios. Se plantean las funcionalidades que realizará el módulo y la lista de reserva del producto, donde se plantean requisitos necesarios para la instalación y funcionamiento del módulo. Se generan lo artefactos necesarios por cada una de las fases de desarrollo por la que transita el proyecto.

2.2 Propuesta de solución

XILEMA-PlatSI, está creada con la concepción de integrar varios módulos para realizar y gestionar auditorías. Actualmente la Plataforma, solo cuenta con el Módulo de Auditorías a Aplicaciones Web (MAAWeb). La propuesta de solución, se desarrolla con la finalidad de integrar un Módulo para Generar Reportes de Auditorías a Aplicaciones Webs (MGRA) a PlatSI. El módulo contará con una serie de funcionalidades que permitirán al usuario, visualizar los datos arrojados por las auditorías a aplicaciones webs realizadas, mediante la confección de un reporte formado por un conjunto de gráficas y tablas.

A continuación, se muestra una imagen con la estructura actual de la Plataforma y los diferentes niveles hasta llegar al módulo que contiene la propuesta de solución**:**

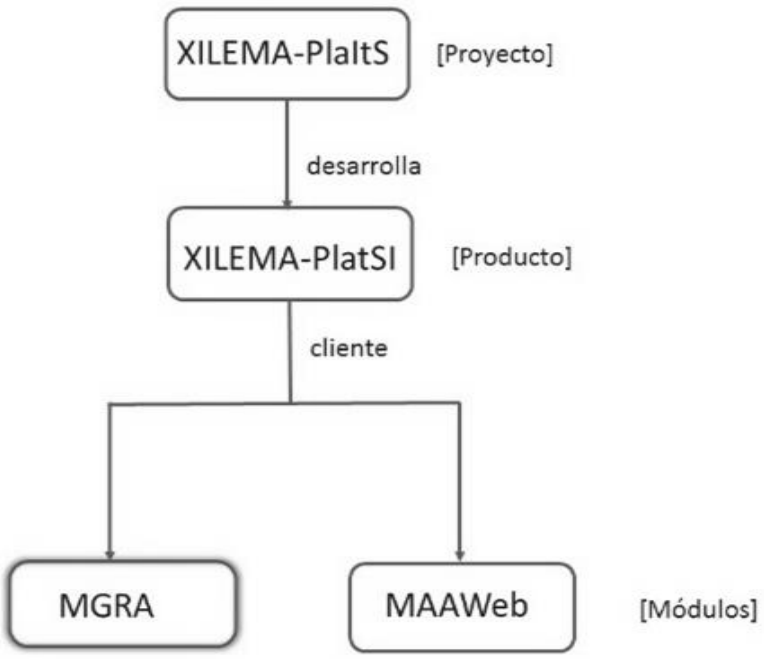


Figura 1: Composición de Xilema-PlatSI

La solución propone que el módulo solo contará con un rol específico para acceder a dicha información, el rol de planificador, el cual, contará con la opción de seleccionar una serie de datos para especificar la información que necesite adquirir. Primeramente, seleccionará el tipo de reporte y luego el intervalo de tiempo del cual desea obtener la información. El módulo, se encargará de generar el reporte, el cual mediante gráficas mostrará los datos solicitados y una tabla conformada por**:** Las auditorías con más tiempo de ejecución, especialista y responsables con más asignaciones, aplicaciones con más vulnerabilidades y riesgos e informes de auditorías con más riesgos. Finalmente, el usuario contará con la opción de exportar a PDF toda la información adquirida o solo la de su interés.

2.3 Lista de funcionalidades del módulo

- ✓ **1:** Generar reportes de auditorías generales según el intervalo de tiempo (anual, mensual o diario). Este reporte contendrá:
    - Gráfica de pastel del estado actual de las auditorías generales.
    - Gráfica de pastel del estado actual de las solicitudes.
    - Gráfica de barras del estado actual de las asignaciones.
    - Gráfica de línea del estado histórico de las auditorías generales.
    - Exportar a formato PDF.

  Mostrará mediante tablas:
    - Las 10 auditorías con más tiempo de ejecución.
    - Los 10 productos con más asignaciones.
    - Los 10 responsables con más asignaciones.

- ✓ **2:** Generar reportes de auditorías técnicas según el intervalo de tiempo (anual, mensual o diario). Este reporte contendrá:
    - Gráfica de pastel del estado actual de las auditorías técnicas.
    - Gráfica de línea del estado histórico de las auditorías técnicas.
    - Gráfica de barras de la cantidad de ejecuciones por herramientas.
    - Gráfica de barras de la cantidad de vulnerabilidades detectadas por herramientas.

- Exportar a formato PDF.

Mostrará mediante tablas:

- Las 10 auditorías técnicas con más tiempo de ejecución.
- Los 10 productos con más riesgo de vulnerabilidad.
- Los 10 productos con más vulnerabilidades.

✓ **3:** Generar reportes por especialista según el intervalo de tiempo (anual, mensual o diario). Este reporte contendrá:

- Gráfica de pastel del estado actual de las auditorías técnicas asignadas al especialista.
- Gráfica de línea del estado histórico de las auditorías técnicas.
- Gráfica de barra de la cantidad de ejecuciones por herramientas.
- Gráfica de barra de la cantidad de vulnerabilidades detectadas por herramientas.
- Exportar a formato PDF.

Mostrará mediante tablas:

- Las 10 auditorías con más tiempo de ejecución.
- Los 10 productos con más riesgo de vulnerabilidad.
- Los 10 productos con más vulnerabilidades.

✓ **4:** Generar reportes por responsable según el intervalo de tiempo (anual, mensual o diario). Este reporte contendrá:

- Gráfica de pastel del estado actual de las auditorías generales asignadas al responsable.
- Gráfica de línea del estado histórico de las auditorías generales.
- Exportar a formato PDF.

Mostrará mediante una tabla:

- Los 10 informes de auditorías generales con más riesgo de vulnerabilidad.

✓ **5:** Generar reportes por solicitante según el intervalo de tiempo (anual, mensual o diario). Este reporte contendrá:

- Gráfica de pastel del estado actual de las solicitudes realizadas por el solicitante.

- Gráfico de línea del estado histórico de las solicitudes en función del intervalo de tiempo.
- Exportar a formato PDF.

2.4 Lista de reserva del producto

Usabilidad

- ✓ Es necesario una preparación previa para trabajar con el módulo, pues a pesar de ser intuitivo su manejo, se necesita un cierto grado de conocimiento en computación.
- ✓ Contará con mensajes de alerta, de errores o de información para guiar al usuario que esté interactuando con la aplicación.

Software

- ✓ Navegador web para acceder al sistema mediante conexión de red.
- ✓ Instalación de un servidor web Apache que contenga PHP v5.4.44.
- ✓ Instalación de un servidor de base de datos PostgreSQL v9.1.

Hardware

- ✓ Computadora de 512 MB de RAM o superior y 10 GB o más de espacio en el disco duro.

Seguridad

- ✓ Se necesita autenticación para acceder al módulo. Para ello se debe tener conocimiento del usuario y contraseña del rol autorizado para acceder al módulo.

Interfaz de usuario

- ✓ La aplicación contará con un conjunto de interfaces que le permitirán al usuario interactuar con el módulo.

2.5 Usuario relacionado con el módulo.

Un usuario es un individuo que utiliza una computadora, sistema operativo, servicio o cualquier sistema informático (Alegsa, 2012). Puede ser cualquier persona relacionada con el sistema, ya sea vinculada al desarrollo del mismo o que de una forma u otra interactúa con la aplicación, incluyendo a los que mantendrán el sistema funcionando y actualizado.

En la Plataforma, actualmente interactúan los usuarios que se identifican con el rol de: Administrador, Especialista, Solicitante o Planificador. El usuario que se identifique con el rol de Especialista, será quien tenga el acceso a operar en las funcionalidades que contendrá el módulo una vez implementado e integrado el mismo.

2.6 Fase de exploración

La fase de Exploración es la primera fase definida por la metodología XP. Esta fase comienza por la creación de una serie de historias, llamadas Historias de Usuarios (HU), las cuales definen mediante su redacción qué es lo que verdaderamente necesita el cliente. Es aquí donde se define el alcance real, permitiendo una familiarización del equipo de desarrollo con las herramientas y tecnologías a usar en la solución (Rodrígez, 2014).

2.7 Historias de Usuarios

Las Historias de Usuarios es la técnica que utiliza la metodología XP para especificar los requisitos del software. Estas son escritas desde la perspectiva del cliente, donde describen de forma breve las características que el componente debe realizar, aunque también los desarrolladores pueden brindar su ayuda en la identificación de las mismas. El contenido de las historias debe ser concreto, sencillo, dinámico y flexible. Cada una de ellas es lo suficientemente comprensible, como para que los programadores puedan estimar con un reducido margen de riesgo, su tiempo de desarrollo. Es el cliente el encargado de asignarle una prioridad a cada Historia de Usuario y es el equipo de desarrollo el encargado de asignarle un costo, este se traduce en las semanas que llevará el desarrollo de las mismas. Por otra parte, es bueno destacar que las Historias de Usuarios nuevas pueden describirse en cualquier momento, con esto se comprueba la flexibilidad de la metodología (Malfará, 2006).

2.7.1 Clasificación de las historias de usuarios

Prioridad en el negocio:

Alto: Se le otorga a las HU que resultan funcionalidades fundamentales en el desarrollo del componente, a las que el cliente define como principales para el control integral.

Medio: Se le otorga a las HU que resultan para el cliente como funcionalidades a tener en cuenta, sin que estas tengan una afectación sobre el componente que se esté desarrollando.

Bajo: Se le otorga a las HU que constituyen funcionalidades que sirven de ayuda al control de elementos asociados al equipo de desarrollo, a la estructura y no tienen nada que ver con el componente en desarrollo.

El riesgo en su desarrollo:

Alto: Cuando en la implementación de las HU se considera la posible existencia de errores que lleven la inoperatividad del código.

Medio: Cuando pueden aparecer errores en la implementación de la HU que puedan retrasar la entrega de la versión.

Bajo: Cuando pueden aparecer errores que serán tratados con relativa facilidad sin que traigan perjuicios para el desarrollo del proyecto*.*

Las HU serán representadas mediante tablas conformadas por los siguientes campos:

Número: Número de la historia de usuario incremental en el tiempo.

Nombre de Historia de Usuario: El nombre de la historia de usuario sería para identificarlas mejor entre los desarrolladores y el cliente.

Modificación de Historia de Usuario Número: Si sufrió alguna modificación anterior.

Usuario: Personas involucradas en el desarrollo de la HU.

Iteración Asignada: Número de la iteración.

Prioridad en negocio: Alto, Medio o Bajo.

Riesgo en Desarrollo: Alto, Medio o Bajo.

Puntos estimados: Tiempo estimado que se demorará el desarrollo de la HU; cada punto es considerado como una semana de trabajo.

Puntos Reales: Tiempo que se demoró en realidad el desarrollo de la HU.

Descripción: Breve descripción de la HU.

Observaciones: Señalamiento o advertencia del componente.

Prototipo de interfaz: Prototipo de interfaz si aplica.

Tabla 1: Generar reportes de auditorías generales

Historia de Usuario	
Número: 1	**Nombre de historia:** Generar reportes de auditorías generales según el intervalo de tiempo (anual, mensual o diario).
Modificación de historia de usuario: Ninguna	
Prioridad en el negocio: Alto	**Riesgo de desarrollo:** Alto
Puntos estimados: 3	**Iteración asignada:** 1
Puntos reales: 3	
Usuarios: Acela María Sanamé Pérez y Manuel Hernández Montiel.	
Descripción: Permite al planificador generar reportes de auditorías generales según el intervalo de tiempo (anual, mensual o diario). Estos reportes contendrán: • Gráfica del estado actual de las auditorías generales. • Gráfica del estado actual de las solicitudes. • Gráfica del estado actual de las asignaciones. • Gráfica del histórico de las auditorías generales. • Las 10 auditorías con más tiempo de ejecución. • Los 10 especialistas con más asignaciones. • Los 10 responsables con más asignaciones.	

Interfaz:

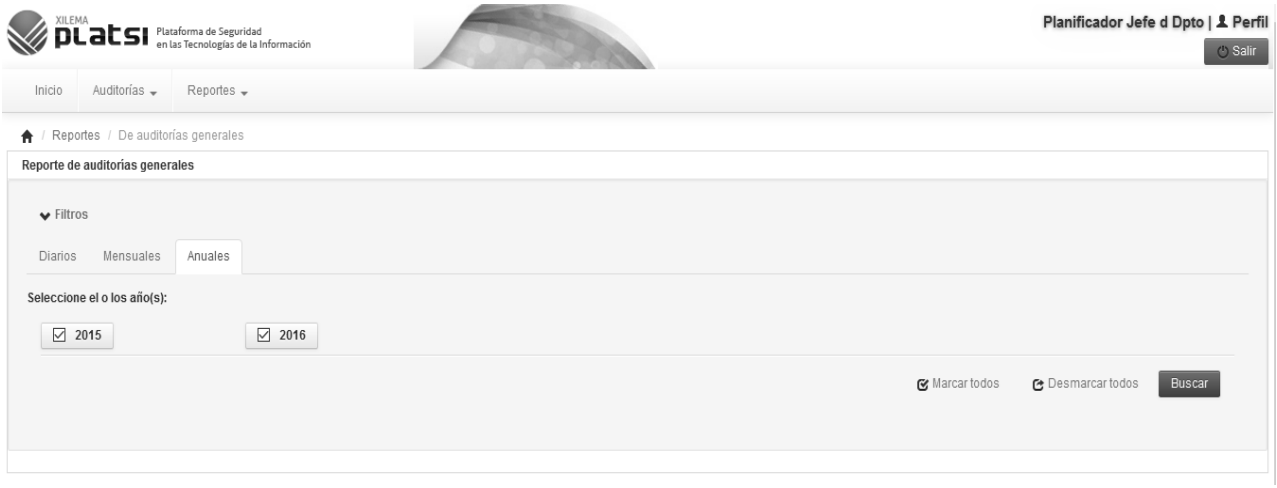
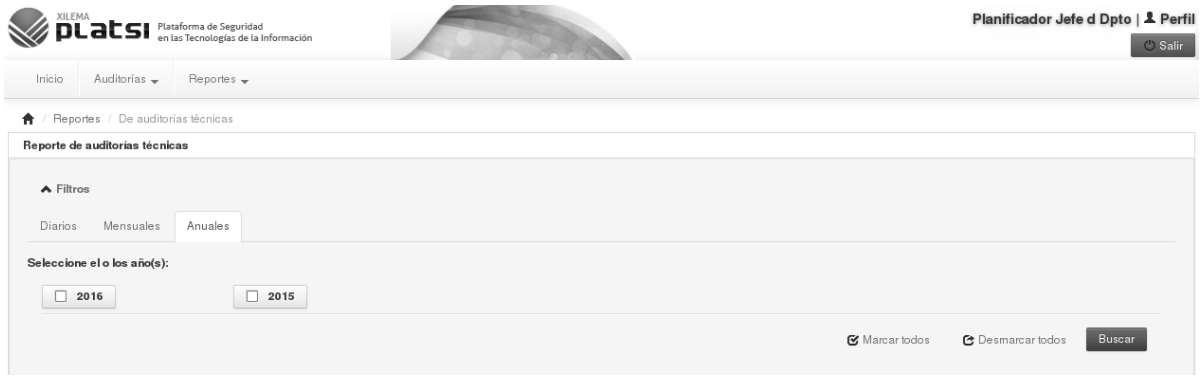


Tabla 2 : Generar reportes de auditorías técnicas

Historia de Usuario	
Número: 2	**Nombre de historia:** Generar reportes de auditorías técnicas según el intervalo de tiempo (anual, mensual o diario).
Modificación de historia de usuario: Ninguna	
Prioridad en el negocio: Alto	**Riesgo de desarrollo:** Alto
Puntos estimados: 3	**Iteración asignada:** 2
Puntos reales: 3	
Usuarios: Acela María Sanamé Pérez y Manuel Hernández Montiel.	
Descripción: Permite al planificador generar reportes de auditorías técnicas según el intervalo de tiempo (anual, mensual o diario). Estos reportes contendrán: • Gráfica del estado actual de las auditorías técnicas. • Gráfica del histórico de las auditorías técnicas. • Gráfica de cantidad de ejecuciones por herramientas. • Gráfica de cantidad de vulnerabilidades detectadas por herramientas.	

- Las 10 auditorías técnicas con más tiempo de ejecución.
- Los 10 productos con más riesgo.
- Los 10 productos con más vulnerabilidades.

Interfaz:



En el *Anexo I* se encuentran las restantes tablas de HU.

2.8 Fase de planificación del sistema

En esta fase el cliente establece la prioridad de cada historia de usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas, esto se expresa utilizando como medida el punto, el cual es considerado como una semana de trabajo ideal para los programadores. Por otra parte, el equipo de desarrollo mantiene un registro de la "velocidad" de desarrollo, establecida en puntos por iteración, basándose principalmente en la suma de puntos correspondientes a las historias de usuario que fueron terminadas en la última iteración (Letelier, 2010).

2.8.1 Estimación del esfuerzo por historia de usuario

Tabla 3 : Estimación por esfuerzo

Historia de usuario	Estimación
1. Generar reporte de auditoria generales.	3 semanas

2. Generar reporte de auditorías técnicas.	3 semanas
3. Generar reporte por especialista.	3 semanas
4. Generar reporte por responsable.	3 semanas
5. Generar reportes por solicitante.	3 semanas

2.9 Plan de iteraciones

Luego de ser identificadas y descritas las Historias de Usuarios, se procede a la planificación de la fase de implementación, donde se realizarán cinco iteraciones con una duración de aproximadamente tres semanas, las cuales se describen a continuación:

Iteración #1

En la presente iteración se realizará la HU número 1. Esta iteración será el inicio de las nuevas funcionalidades para el módulo que el cliente necesita.

Iteración #2

En esta iteración se realizará la HU 2. Perteneciente a la opción de generar reportes por cada uno de las auditorías técnicas.

Iteración #3

En esta iteración se realizará la HU 3. Perteneciente a la opción de generar reportes por cada responsable.

Iteración #4

En esta iteración se realizará la HU 4. Perteneciente a la opción de generar reportes por cada especialista.

Iteración #5

En esta iteración se realizarán las HU 5. Perteneciente a la opción de generar reportes por solicitante. Concluida esta iteración el módulo estar listo para su evaluación y puesta en práctica en un entorno real.

2.9.1 Plan de duración de las iteraciones.

Tabla 4 : Plan de duración de las iteraciones

Iteración	Orden de HU por implementar	Duración total
1	✓ Generar reporte de auditoria generales.	3 semanas
2	✓ Generar reporte de auditorías técnicas.	3 semanas
3	✓ Generar reporte por especialista.	3 semanas
4	✓ Generar reporte por responsable.	3 semanas
5	✓ Generar reportes por solicitud.	3 semanas

2.10 Plan de entrega

En este plan se detalla la fecha fin de cada iteración, mostrando una versión desarrollada del producto en ese momento hasta lograr el producto final en la fecha establecida. A continuación, se muestra el plan de entrega para cada iteración:

Tabla 5 : Plan de entrega

Módulo	Número de iteración	Fechas de entregas
Módulo de Generación de Reportes para la Plataforma de Seguridad en las Tecnologías de la Información.	1	(11/3/2016)
	2	(1/4/2016)
	3	(22/4/2016)
	4	(13/5/2016)
	5	(3/5/2016)

2.11 Conclusiones Parciales

Luego de planteada la propuesta de solución, se estimó y planificó el esfuerzo para la implementación del módulo, para ello se generaron varios artefactos como: plan de iteración, plan de duración de las iteraciones y plan de entrega. Estos artefactos traducen las necesidades del cliente en funcionalidades, recogen las tareas que deben realizar cada desarrollador agrupadas por iteración y el tiempo máximo del que dispone para ejecutarlas. Finalmente se obtuvo como resultado, un tiempo estimado de 15 semanas para implementar el módulo que generará los reportes en la Plataforma.

Capítulo 3: Diseño del módulo

3.1 Introducción

En este capítulo se describe la fase de diseño del módulo propio de la metodología XP. Se define el patrón de arquitectura a seguir, así como los patrones de diseño a utilizar para la implementación del módulo y quedarán definidas las tarjetas CRC como técnica de diseño. Además, se efectuarán pruebas, con el objetivo de detectar errores y fallas en el módulo desarrollado, asegurando la calidad del producto.

3.2 Arquitectura

En su forma más sencilla, la arquitectura es la estructura de organización de los componentes de un programa (Módulo), la forma en la que estos interactúan y la estructura de datos que utilizan. Sin embargo, en un sentido más amplio, los componentes se generalizan para que representen los elementos de un sistema grande y sus interacciones (Pressman, 2010).

Actualmente la arquitectura general de la Plataforma, está compuesta por varios Bundles que le permiten realizar parte de las tareas para la que fue creada. Unos de estos Bundles contendrán el módulo (MGRA), el cual estará guiado por la arquitectura Cliente-Servidor y como patrón de arquitectura Modelo-Vista-Controlador, debido a que es el patrón que utiliza el framework de desarrollo Symfony2.

3.2.1 Arquitectura Cliente-Servidor.

El modelo arquitectónico Cliente-Servidor es un modelo de sistema en el que dicho sistema se organiza como un conjunto de servicios y de servidores asociados, más unos clientes que acceden y usan los servicios (Pressman, 2010).

3.2.2 Patrón de arquitectura

Los patrones arquitectónicos, o patrones de arquitectura, son patrones de software que ofrecen soluciones a problemas de arquitectura de software en ingeniería de software. Especifican un conjunto predefinido de subsistemas con sus responsabilidades y una serie de recomendaciones para organizar los distintos componentes (Venete, 2011).

El patrón MVC consiste en tres tipos de objetos: el modelo que es el objeto de la aplicación, la vista que es su representación y el controlador que define el modo en que la interfaz reacciona a la entrada

del usuario. De una manera poco ortodoxa se dice que el controlador controla el flujo de la aplicación, pide al modelo aquello que el usuario solicita, y le devuelve (al usuario) una representación del modelo a través de la vista. El siguiente gráfico muestra la cooperación entre los tres objetos:

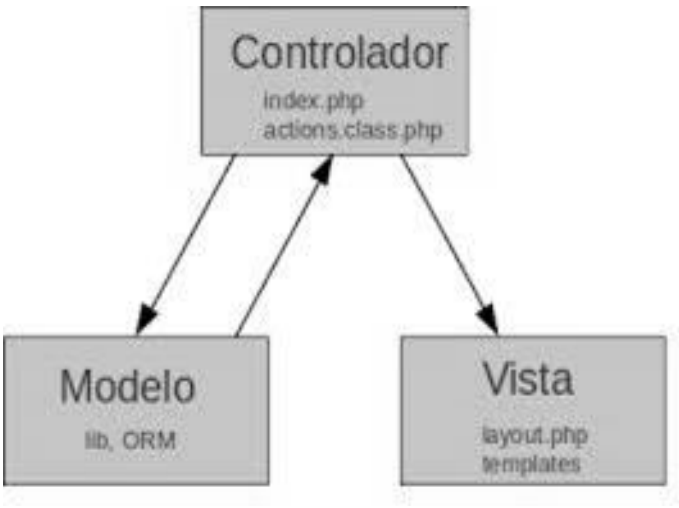


Figura 2: Arquitectura MVC

Dentro de las estructuras que propone el framework de desarrollo Symfony2, se encuentra la estructura de carpeta reflejada en la siguiente Figura.

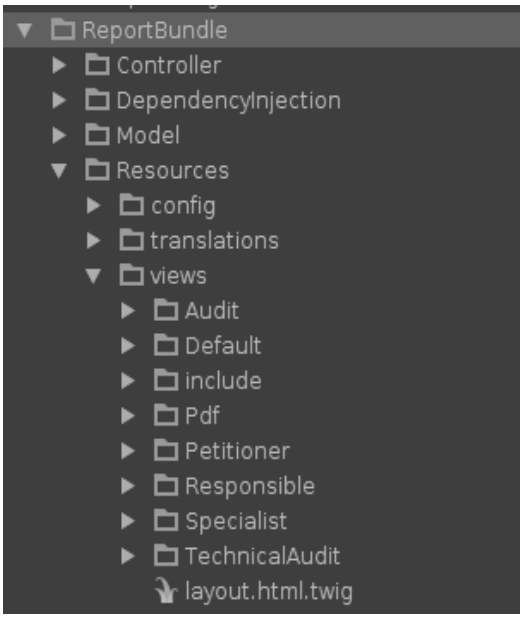


Figura 3: Estructura de carpetas de symfony2

A continuación, se muestra mediante diagramas de clases una representación de las clases que componen cada una de las capas del patrón MVC.

Modelo: Contiene todas las clases que representan las entidades del Bundle.

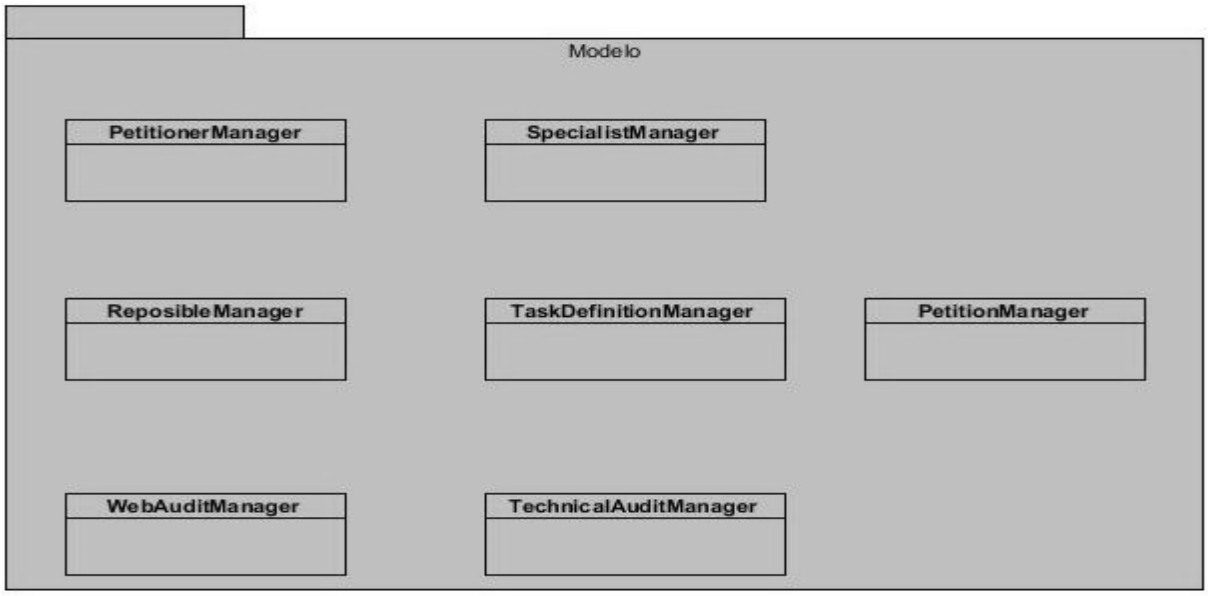


Figura 4: Clases de capa del modelo del módulo

Controlador: Recibe entradas, normalmente como eventos. Los eventos son traducidos a solicitudes de servicio, bien para el modelo o bien para la vista.

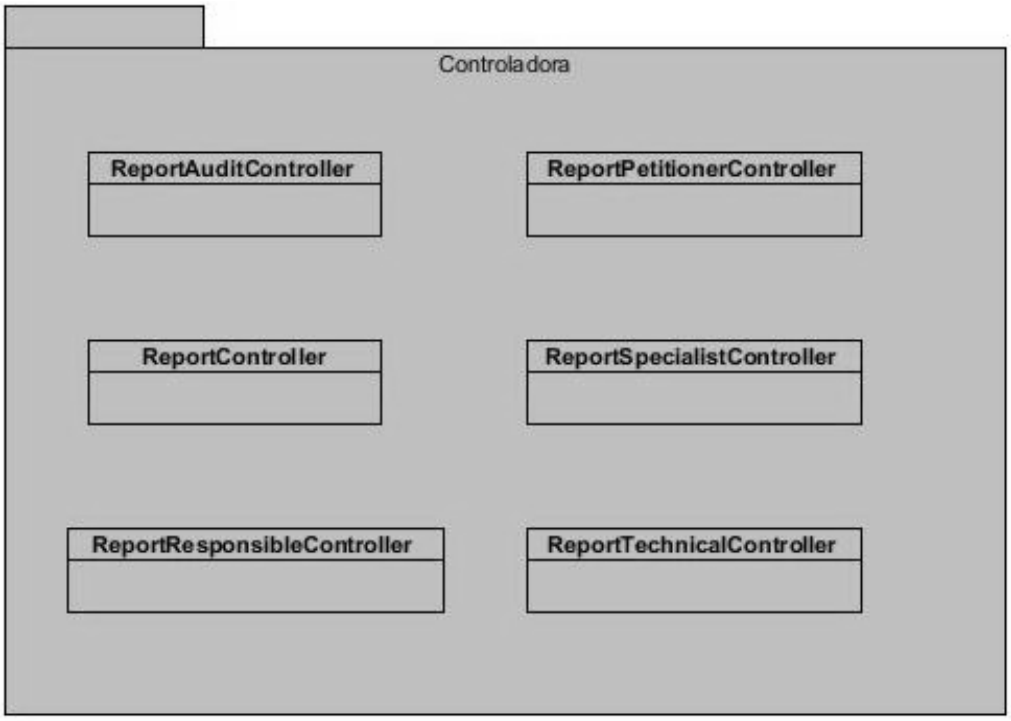


Figura 5: Clases de la capa controladora de módulo

Vista: Muestra la información al usuario. Pueden existir múltiples vistas del modelo, cada una teniendo asociado un componente controlador.

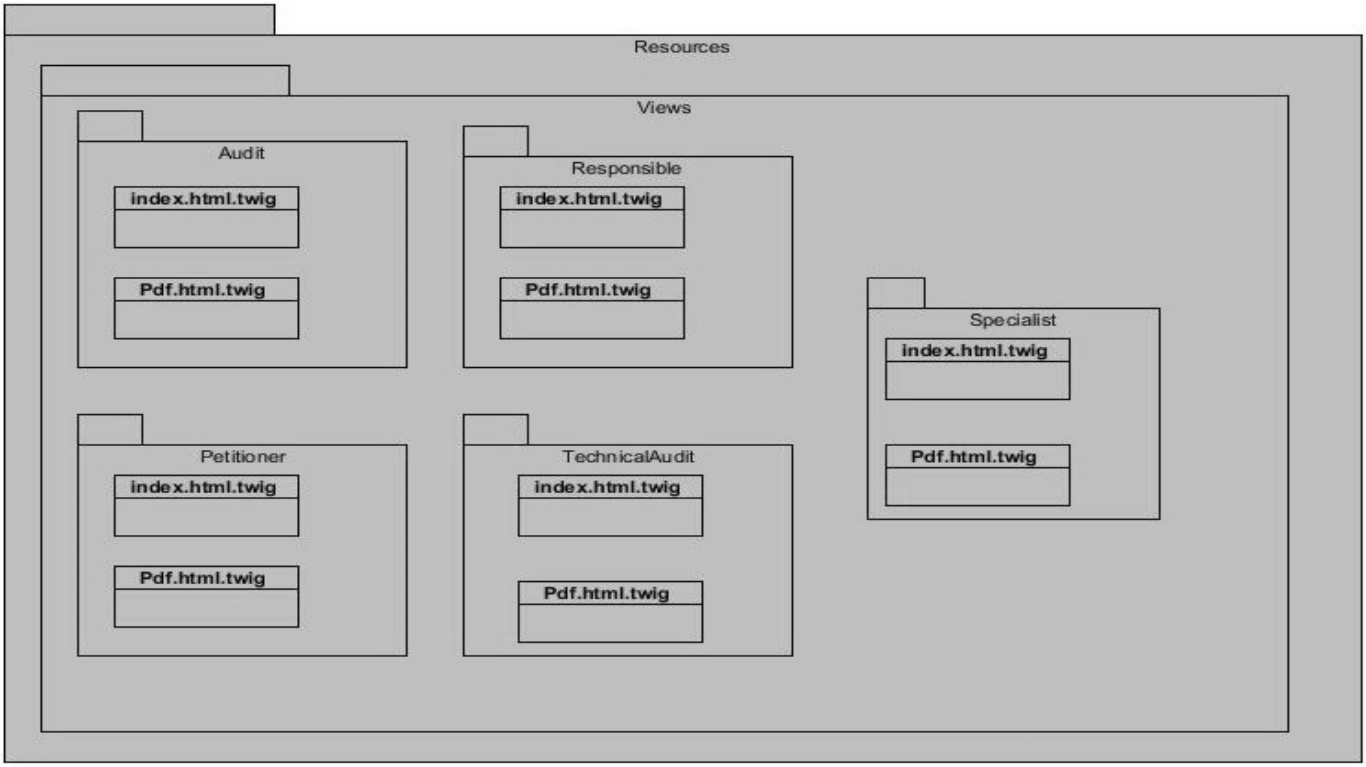


Figura 6: Clases de la capa controladora del módulo

3.3 Patrones de diseño

Un patrón de diseño constituye un esquema para refinar subsistemas o componentes. Es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular. Un patrón de diseño identifica: clases, instancias, roles, colaboraciones y la distribución de responsabilidades, además de que ayuda a construir clases y a estructurar sistemas de clases. Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces (Gamma, 2003).

3.4 Patrones GRASP (General Responsibility Assignment Software Patterns)

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos y constituyen la base del cómo se diseñará el sistema **(Gamma, 2003)**. Los patrones GRASP usados para el diseño son del presente módulo son**:**

Experto

Problema: ¿Cuál es un principio general para asignar responsabilidades a los objetos?

Solución: asignar una responsabilidad a la clase que tiene la información necesaria para realizar la responsabilidad, siendo esta clase el experto en información.

Ejemplo: el patrón se evidencia en la capa Controladora la cual está dividida en distintas clases cada una con distintas funcionalidades asignadas y con la información necesaria para realizarlas. Ejemplo de estas clases son: ReportAuditController.php, ReportResponcibleController.php y las demás clases que conforman la capa controladora del módulo.

Bajo Acoplamiento

Problema: ¿Cómo dar soporte a las bajas dependencias y al incremento de la reutilización?

Solución: diseñar con el objetivo de tener las clases lo menos ligadas entre sí que se pueda. De tal forma que, en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases.

Ejemplo: El patrón bajo acoplamiento se evidencia en la distribución de cada uno de los controladores de la aplicación, los cuales están diseñados por separados para la realización de cada una de las funcionalidades en específico, lográndose de esta manera que un fallo en alguna de estas clases controladoras no afecte a las demás funcionalidades de los otros controladores. Ejemplo de estas clases son ReportSpecialistController.php, ReportTechnicalController.php y las demás clases que conforman la capa Controladora.

Alta Cohesión

Problema: ¿Cómo mantener la complejidad manejable?

Solución: Asignar responsabilidades de manera que la información que almacena una clase sea coherente y esté relacionada con la clase.

Ejemplo: Este patrón se ve evidenciado cuando cada clase controladora utiliza su modelo específico mediante el cual accede a los datos que necesita para la realización de una funcionalidad en concreto con la información que realmente le hace falta y posteriormente pasársela a la vista. Un ejemplo más detallado se visualiza en la clase ReportAuditController.php que accede a los datos mediante la clase AuditManager.php.

Controlador

Problema: ¿Quién gestiona un evento del sistema?

Solución: Asignar la responsabilidad de gestionar un mensaje de un evento del sistema a una clase.

Ejemplo: se requirió el empleo del patrón en todas las clases controladoras del módulo ya que estas son las responsables de gestionar las funcionalidades del sistema.

3.5 Tarjetas CRC (Clases-Responsabilidad-Colaboración)

Diseñar un sistema utilizando la metodología XP solo es posible si se siguen tres principios: Cargo o Clase, Responsabilidad y Colaboración (CRC). Estos tres principios hacen posible que todo el equipo de desarrollo esté inmerso en la tarea de diseño.

Las tarjetas CRC representan objetos que se transforman en clases bien estructuradas con responsabilidades bien definidas. Son simples y adaptables, características que facilitan la interacción entre el cliente y el equipo de desarrollo (2015). Las tarjetas CRC generadas para el diseño de la solución propuesta se muestran a continuación:

Tabla 6 : ReportAuditController

Tarjeta CRC	
Clase: ReportAuditController	
Descripción: Clase encargada de generar los reportes de auditorías generales.	
Responsabilidades:	**Colaboradores:**
✓ Generar reporte de auditorías generales para los filtros anuales, mensuales y diarios. ✓ Exportar reporte a PDF.	✓ AuditManager Audit ✓ AssignmentManager Assignment

Tabla 7 : ReportController

Tarjeta CRC	
Clase: ReportSpecialistController	
Descripción: Clase encargada de generar los reportes por especialista.	
Responsabilidades:	**Colaboradores:**

✓ Generar reporte por especialista para los filtros anuales, mensuales y diarios. ✓ Exportar reporte a PDF.	✓ AuditManager Audit ✓ PetitionManager

Tabla 8 : ReportPetitionController

Tarjeta CRC	
Clase: ReportResponsibleController	
Descripción: Clase encargada de generar los reportes por responsable.	
Responsabilidades:	**Colaboradores:**
✓ Generar reporte por responsable para los filtros anuales, mensuales y diarios. ✓ Exportar reporte a PDF.	✓ AuditManager Audit

Las restantes tablas de Tarjetas CRC se visualizan en los *Anexo II*.

3.6 Conclusiones parciales

La determinación de los patrones arquitectónicos y de diseño usados permitió lograr una mayor organización en los elementos que conforman la aplicación. Se definieron los patrones presentes en el código que componen el módulo. Se definieron las tareas de ingeniería que describen las HU a implementar. La confección de las tarjetas CRC definidas para el desarrollo del sistema permitió conocer el comportamiento de cada una de las clases.

Capítulo 4: Implementación y prueba

4.1 Implementación

En esta fase XP propone tener en cuenta una serie de aspectos como son la disponibilidad del cliente y el desarrollo en pareja para lograr mayores resultados en la implementación del *software*.

Disponibilidad del cliente: El cliente formó parte del equipo de desarrollo, describió las HU, guío la toma de decisiones, aprobó las versiones del producto y verificó el cumplimiento de los objetivos trazados.

Desarrollo en pareja: Toda la implementación fue realizada por dos personas que trabajaron en forma conjunta.

4.2 Tarea de ingeniería

Las Tareas de la ingeniería son utilizadas como apoyo a las historias de usuarios. Estas detallan con más profundidad la implementación de las HU, por eso son la base para la implementación del módulo. Las tareas de ingeniería están descritas mediante tablas que contienen los siguientes campos:

Número de la tarea: Los números han de ser consecutivos.

Número de Historia de Usuario: Número de historia de usuario a la que pertenece la tarea.

Nombre de la tarea: Nombre que identifica la tarea.

Tipo de tarea: Las tareas pueden ser de Desarrollo, Mejora, Corrección, ext.

Puntos estimados: Tiempo estimados en días que se le asignará a su desarrollo.

Programadores responsables: Nombre y apellidos de los programadores**.**

Descripción: Breve descripción de la tarea.

Tabla 9 : Generar reportes de auditorías generales en el intervalo de tiempo anual

Tareas de ingeniería	
Número de tarea: 1	**Número de HU:** 1

Nombre de la tarea: Generar reportes de auditorías generales en el intervalo de tiempo anual.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 4/5
Programadores responsables: Acela María Sanamé Pérez y Manuel Hernández Montiel	
Descripción: Generar los reportes con los elementos contenidos en la descripción de la HU a la que pertenece, pero con los resultados comprendidos en el intervalo de tiempo anual seleccionado.	

Tabla 10 : Generar reportes de auditorías generales en el intervalo de tiempo mensual

Tareas de ingeniería	
Número de tarea: 2	**Número de HU:** 1
Nombre de la tarea: Generar reportes de auditorías generales en el intervalo de tiempo mensual.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 4/5
Programadores responsables: Acela María Sanamé Pérez y Manuel Hernández Montiel	
Descripción: Generar los reportes con los elementos contenidos en la descripción de la HU a la que pertenece, pero con los resultados comprendidos en el intervalo de tiempo mensual seleccionado.	

Tabla 11 : Generar reportes de auditorías generales en el intervalo de tiempo diario

Tareas de ingeniería	
Número de tarea: 3	**Número de HU:** 1
Nombre de la tarea: Generar reportes de auditorías generales en el intervalo de tiempo diario.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 4/5

Programadores responsables: Acela María Sanamé Pérez y Manuel Hernández Montiel
Descripción: Generar los reportes con los elementos contenidos en la descripción de la HU a la que pertenece, pero con los resultados comprendidos en el intervalo de tiempo diario seleccionado.

A continuación, se muestran dos tareas pertenecientes a la primera iteración, las demás se podrán visualizar en *Anexo III*.

4.3 Estándar de codificación

El código es la mejor documentación que tiene el sistema. Es por este motivo que se deben establecer reglas para los programadores y estas deben ser seguidas estrictamente, por eso existen los estándares de codificación. Un estándar de codificación es un estilo de programación homogéneo a seguir por los desarrolladores para permitir que los programas se acerquen lo mejor posible al lenguaje natural. Facilita la lectura y la modificación del código por cualquier miembro del equipo de desarrollo e incorpora las mejores prácticas de la codificación (Calleja, Manuel Arias, 2012).

El módulo desarrollado se rige por los estándares que utiliza el framework empleado. Symfony2 sigue los estándares definidos en los documentos psr-0, psr-1 y psr-2 que pueden ser consultados en el repositorio oficial del estándar https://github.com/php-fig/fig-standards. A continuación, se muestran algunos de los estándares de codificación utilizados en el módulo:

Estructura

- ✓ Añade un solo espacio después de cada delimitador coma;
- ✓ Añade una línea en blanco antes de las declaraciones return, a menos que el valor devuelto solo sea dentro de un grupo de declaraciones (tal como una declaración if);
- ✓ Declara las propiedades de clase antes que los métodos;

Convenciones de nomenclatura

- ✓ Utiliza mayúsculas intercaladas —sin guiones bajos— en nombres de variable, función, método o argumentos;
- ✓ Utiliza espacios de nombres para todas las clases;

Licencia

- ✓ Symfony se distribuye bajo la licencia MIT, y el bloque de la licencia tiene que estar presente en la parte superior de todos los archivos PHP, antes del espacio de nombres.

4.4 Pruebas al sistema

Uno de los pilares de la metodología XP es el uso de pruebas para comprobar el funcionamiento de los códigos que se hayan implementado. Las pruebas son una actividad en la cual un sistema o componente es ejecutado bajo condiciones o requerimientos especificados, los resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema o componente. La prueba de software es un elemento crítico para la garantía de la calidad del software y presenta una revisión final de las especificaciones del diseño y de la codificación (Pressman, 2010).

4.4.1 Estrategia de prueba

La metodología XP define solo dos niveles de pruebas, de aceptación y pruebas unitarias. Para comprobar el funcionamiento del módulo se realizarán las pruebas establecidas. Los métodos a utilizar serán: para las pruebas de aceptación el método de caja negra y método de caja blanca para las unitarias.

4.4.2 Pruebas unitarias

Las pruebas unitarias son una de las piedras angulares de XP. Son la escala más pequeña de las pruebas, está basada en la funcionalidad de los módulos del programa, como funciones, procedimientos, módulos de clase. La elaboración de código está dirigida por este tipo de prueba, las cuales se ejecutan constantemente ante cada modificación del sistema (Letelier, 2010).

Symfony2 provee soporte para escribir y ejecutar pruebas unitarias, basada en PHPUnit. Este es un entorno para realizar pruebas unitarias automatizadas en el lenguaje de programación PHP. Las pruebas unitarias fueron implementadas luego de concluir cada una de las historias de usuario y pueden ser consultadas en la estructura del módulo dentro de una carpeta llamada Test.

Luego de terminada la implementación de la historia de usuario definida en la 1ra iteración se realizó la primera iteración de pruebas unitarias en donde se obtienen un total de 11 no conformidades tales como parámetros incorrectos, las cuales fueron resueltas y no quedó ninguna pendiente por resolver. Luego se ejecutó una segunda iteración donde no se detectaron no conformidades.

Una vez concluida la implementación de la historia de usuario definida en la 2da iteración se realizó la primera iteración de pruebas unitarias en donde se obtienen un total de 12 no conformidades tales

como parámetros incorrectos, las cuales fueron resueltas y no quedó ninguna pendiente por resolver. Posteriormente se ejecutó una segunda iteración donde no se detectaron no conformidades.

Luego de realizar la implementación de la historia de usuario definida en la 3ra iteración se realizó la primera iteración de pruebas unitarias en donde se obtienen un total de 8 no conformidades tales como retornos de datos incorrectos, las cuales fueron resueltas y no quedó ninguna pendiente por resolver. Después se ejecutó una segunda iteración donde no se detectaron no conformidades.

Luego de terminada la implementación de la historia de usuario definida en la 4ta iteración se realizó la primera iteración de pruebas unitarias en donde se obtienen un total de 5 no conformidades tales como parámetros incorrectos, las cuales fueron resueltas y no quedó ninguna pendiente por resolver. Después se ejecutó una segunda iteración donde no se detectaron no conformidades.

Finalmente, se implementó la historia de usuario definida en la 5ta iteración se realizó la primera iteración de pruebas unitarias en donde se obtienen un total de 4 no conformidades tales como patrones parámetros incorrectos, las cuales fueron resueltas y no quedó ninguna pendiente por resolver. Después se ejecutó una segunda iteración donde no se detectaron no conformidades.

Los resultados se muestran en la Ilustración número 7:

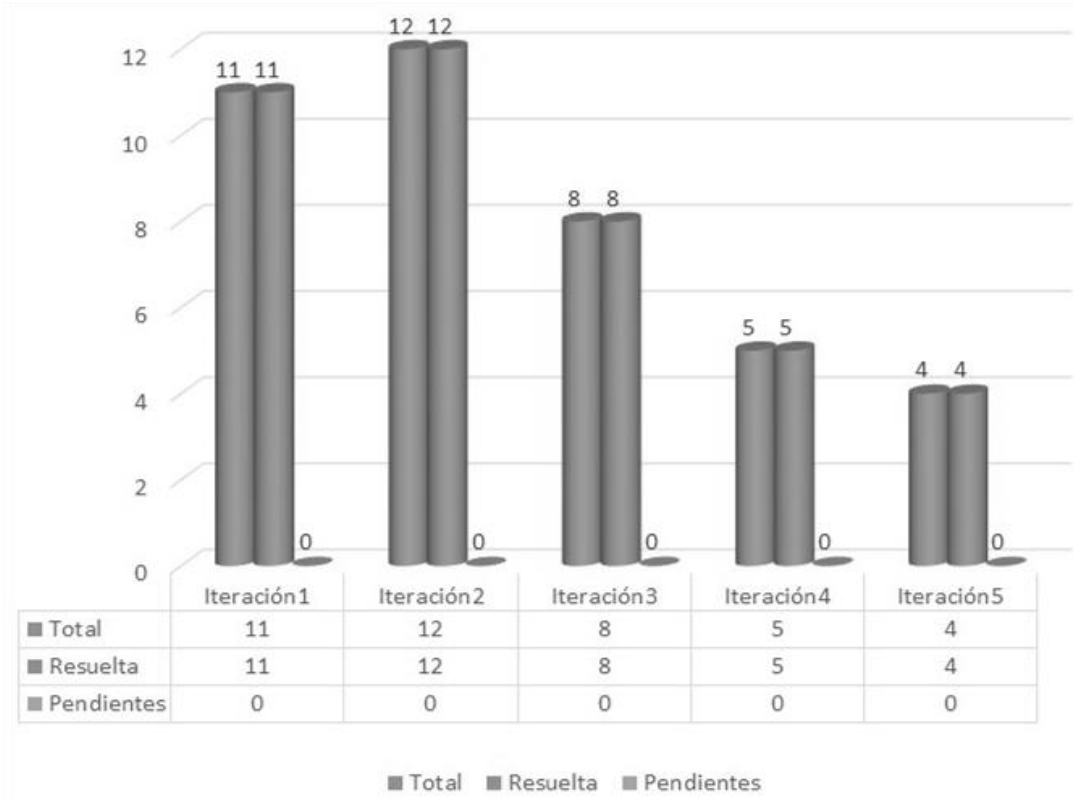


Figura 7: Resultado de las pruebas unitarias

4.4.3 Pruebas de aceptación

Su objetivo principal es verificar las características funcionales del sistema, las cuales son la principal fuente de información a la hora de construir las pruebas de aceptación.

Las pruebas de aceptación son creadas a partir de las historias de usuario. Durante una iteración la historia de usuario seleccionada en la planificación de iteraciones se convertirá en una prueba de aceptación. El cliente o usuario especifica los aspectos a testear cuando una historia de usuario ha sido correctamente implementada (Malfará, 2006).

Una historia de usuario puede tener más de una prueba de aceptación, tantas como sean necesarias para garantizar su correcto funcionamiento y no se considera completa hasta que no supera sus pruebas de aceptación. Esto significa que debe desarrollarse un nuevo test de aceptación para cada iteración o se considerará que el equipo de desarrollo no realiza ningún progreso. Una prueba de aceptación es como una caja negra. Cada una de ellas representa una salida esperada del sistema. Es responsabilidad del cliente verificar la corrección de las pruebas de aceptación y tomar decisiones

acerca de las mismas. La garantía de calidad es una parte esencial en el proceso de XP (Malfará, 2006).

Los casos de prueba (CP) elaborados comprenden las siguientes secciones:

Código: Identificador del CP, es sugerente al nombre de la prueba a la que hace referencia.

Historia de Usuario: HU a la que hace referencia.

Nombre: Nombre que se le da a la prueba a realizar.

Descripción: Se describe la funcionalidad que se desea probar.

Condiciones de Ejecución: Condiciones que deben cumplirse para poder llevar a cabo el caso de prueba.

Entradas / Pasos de Ejecución: Descripción de cada uno de los pasos seguidos durante el desarrollo de la prueba, se tendrá en cuenta cada una de las entradas que hace el usuario con el objetivo de ver si se obtiene el resultado esperado.

Resultado esperado: Breve descripción del resultado que se espera obtener con la prueba.

Evaluación de la prueba: Acorde al resultado de la prueba realizada se emitirá una evaluación.

Resultado obtenido: Breve descripción del resultado obtenido con la prueba.

A continuación, se presentan los CP elaborados.

Tabla 12 : Caso de prueba HU1_P1

Prueba de aceptación	
Código: Código: HU1_P1	**Historia de Usuario:** Generar reportes de auditorías generales según el intervalo de tiempo (anual, mensual o diario).
Nombre: Generar reportes de auditorías generales en el intervalo de tiempo anual.	
Descripción: Permite generar auditorías generales por años.	
Condiciones de Ejecución: Tener rol de planificador.	
Entradas / Pasos de Ejecución:	

✓ El planificador selecciona del menú principal la opción Reporte. ✓ El planificador pulsa el vínculo “De auditorías generarles”. ✓ El planificador pulsa el vínculo “Filtro”. ✓ El planificador selecciona el intervalo de tiempo anual. ✓ El planificador selecciona él o los años. ✓ El planificador presiona el botón Buscar. ✓ El planificador selecciona la información que desee obtener (Paso Opcional). ✓ El planificador selecciona la opción descargar. ✓ El planificador selecciona la opción descargar como PDF.
Resultado esperado: Se generará el reporte que muestra las auditorias generales por el intervalo de tiempo anual y se exportará a formato PDF.
Evaluación de la prueba: Satisfactoria

Tabla 13 : Caso de prueba HU1_P2

Pruebas de aceptación	
Código: Código: HU1_P2	**Historia de Usuario:** Generar reportes de auditorías generales según el intervalo de tiempo (anual, mensual o diario).
Nombre: Generar reportes de auditorías generales en el intervalo de tiempo mensual.	
Descripción: Permite generar auditorías generales por meses.	
Condiciones de Ejecución: Tener rol de planificador.	
Entradas / Pasos de Ejecución: ✓ El planificador selecciona del menú principal la opción Reporte. ✓ El planificador pulsa el vínculo “De auditorías generarles”. ✓ El planificador pulsa el vínculo “Filtro”. ✓ El planificador selecciona el intervalo de tiempo mensual. ✓ El planificador selecciona el año. ✓ El planificador selecciona él o los meses. ✓ El planificador presiona el botón Buscar.	

✓ El planificador selecciona la información que desee obtener (Paso Opcional). ✓ El planificador selecciona la opción descargar. ✓ El planificador selecciona la opción descargar como PDF.
Resultado esperado: Se generará el reporte que muestra las auditorias generales por el intervalo de tiempo mensual y se exportará a formato PDF.
Evaluación de la prueba: Satisfactoria

Tabla 14 : Caso de prueba HU1_P3

Pruebas de aceptación	
Código: Código: HU1_P3	**Historia de Usuario:** Generar reportes de auditorías generales según el intervalo de tiempo (anual, mensual o diario).
Nombre: Generar reportes de auditorías generales en el intervalo de tiempo diario.	
Descripción: Permite generar auditorías generales por días.	
Condiciones de Ejecución: Tener rol de planificador.	
Entradas / Pasos de Ejecución: ✓ El planificador selecciona del menú principal la opción Reporte. ✓ El planificador pulsa el vínculo “De auditorías generarles”. ✓ El planificador pulsa el vínculo “Filtro”. ✓ El planificador selecciona el intervalo de tiempo diario. ✓ El planificador selecciona el año. ✓ El planificador selecciona el mese. ✓ El planificador selecciona el o los días. ✓ El planificador presiona el botón Buscar. ✓ El planificador selecciona la información que desee obtener (Paso Opcional). ✓ El planificador selecciona la opción descargar. ✓ El planificador selecciona la opción descargar como PDF.	

Resultado esperado: Se generará el reporte que muestra las auditorias generales por el intervalo de tiempo diario y se exportará a formato PDF.
Evaluación de la prueba: Satisfactoria

En el Anexo IV se muestran las restantes Pruebas de aceptación.

Los resultados de las pruebas de aceptación se muestran en la siguiente Ilustración.

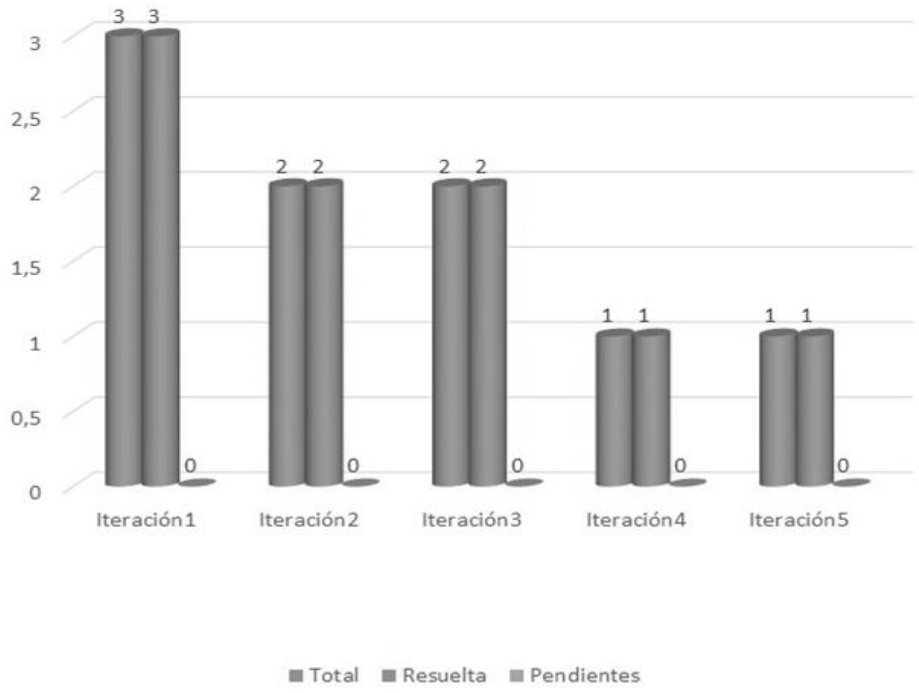


Figura 8: Resultado de las pruebas de aceptación

Se realizaron dos iteraciones de prueba por cada una de las HU. En la primera iteración perteneciente a la HU 1 de pruebas se obtuvo tres no conformidades entre ellas botones con textos en inglés, estas fueron resueltas y en la segunda no conformidad no se encontró cero no conformidades. En la primera iteración de prueba pertenecientes a la HU 2 se detectaron dos no conformidades que consistían en que las funcionalidades no devolvían parte de los datos esperados como: las 10 auditorías técnicas con más tiempo de ejecución, los 10 productos con más riesgo o los 10 productos con más vulnerabilidades, estas no conformidades fueron resueltas y en la segunda

iteración de prueba no se encontraron no conformidades. La primera iteración de prueba perteneciente a la HU 3 arrojó dos no conformidades dentro de las que se encuentran problemas al seleccionar el intervalo de tiempo deseado, las no conformidades fueron corregidas y la segunda iteración no se encontró no conformidades. En la primera iteración de prueba de las HU 4 y HU 5 se encontró una no conformidad estas fueron corregidas y en la segunda iteración de prueba de cada una de ellas no se encontraron no conformidades.

4.5 Conclusiones parciales

Con la realización de la fase de implementación y prueba planteada por la metodología XP, se logró comprobar que el sistema creado cumpliera con los requerimientos planteados. La realización de las tareas de ingeniería proporcionó solución a todas las HU logrando una mayor organización y rapidez en el desarrollo del sistema. El uso de un estándar de codificación permitió una mejor legibilidad de la aplicación y que esta fuera de fácil entendimiento para posteriores desarrolladores de este sistema en versiones futuras. La culminación del desarrollo de las pruebas unitarias y de aceptación con resultados satisfactorios garantizó una buena calidad del sistema.

Conclusiones

El aporte principal de este trabajo consiste en el diseño e implementación de una aplicación que sea capaz de generar reportes en Xilema-PlatSi. Después de concluida la investigación y ser analizados los resultados obtenidos, las conclusiones a las que se arribaron son las siguientes:

- ✓ Las herramientas estudiadas que presentaban un comportamiento similar al deseado por el cliente, no eran consideradas adecuadas para dar solución al problema de la investigación (exceptuando amCharts, la cual se utilizó como librería de apoyo para realizar el módulo), por lo que resultó necesario la creación de una nueva herramienta.
- ✓ La selección de la metodología de desarrollo ágil XP permitió un fácil entendimiento entre los distintos participantes en la producción de software a lo largo del ciclo de vida del proyecto, teniendo en cuenta su papel y responsabilidad.
- ✓ Las herramientas, los lenguajes de programación y la tecnología utilizada facilitaron el trabajo realizado.
- ✓ La propuesta de solución definida contiene cinco funcionalidades descritas en cinco HU divididas en cuatro iteraciones con un entregable por iteración, que a través de servicios que tiene Symfony2 se integra la solución con Xilema-PlatSI. La estimación del esfuerzo para la realización de cada HU arrojó una duración aproximada de dos meses para la implementación del trabajo de diploma.
- ✓ Se implementó a partir de los requerimientos de software establecidos con el cliente, el módulo para Generar Reportes de Auditorias Web en Xilema-PlatSI responsable de generar, mostrar y exportar reportes.
- ✓ El uso de patrones arquitectónicos y de diseño propició tener estructuras de código más completas y mejor elaboradas, permitiendo mejorar en gran medida los resultados esperados por el cliente.
- ✓ Las pruebas realizadas comprobaron el grado de cumplimiento respecto a las especificaciones iniciales del sistema, garantizando la calidad del software.
- ✓ La ventaja que ha traído el módulo en Xilema-PlatSI es la posibilidad de mostrar la información almacenada en los centros de datos de la Plataforma de manera específica mediante la conformación de reportes.

Recomendaciones

Una vez concluida la investigación se proponen las siguientes recomendaciones:

- ✓ Realizar pruebas de estrés una vez implementado todo el módulo.
- ✓ Para nuevas versiones del módulo incluir la opción de exportar a otros formatos además de PDF.
- ✓ Filtrar la información por el nombre de la aplicación web a la que se le haya realizado la auditoría.

Referencias bibliográficas

1. Pascual, Efrén Santos, *Auditoria de Páginas Web*, 2007.
2. Calleja, Manuel Arias, *Centro de Investigación sobre Sistemas Inteligentes de Ayuda a la Decisión*, 2012, http://www.cisiad.uned.es/carmen/estilo-codificacion.pdf.
3. MENDOZA, MARTHA ASCENCIO, *Desarrollo de una Propuesta Metodológica para Determinar la Seguridad en una* (Prira, 2011).
4. Orallo, Enrique Hernández, *El Lenguaje Unificado de Modelado (UML)* (España, 2015).
5. wordpress, *Fergarciac*, n.d., https://fergarciac.wordpress.com/2013/01/25/entorno-de-desarrollo-integrado-ide/.
6. UCI, “Generador Dinámico de Reporte” (2012).
7. Cienfuegos., Univercidad de, “GeReport: Sistema de Gestión de Reportes Dinámicos.” (2014).
8. Universidad Nacional Autónoma de Mexico, *Gráfico de pastel* (Mexico, 2015).
9. Orallo, Enrique Hernández, “Ingeniería de Software” (2014).
10. Pressman, Roger S., *Ingeniería de Software Un enfoque práctico Séptima edición.* (Mexico, n.d.).
11. Venete, Adriana, *Introducción a los patrones de arquitectura.*, 2011.
12. Avila, Katty, *Kavisi*, n.d., http://www.cavsi.com/preguntasrespuestas/que-es-un-sistema-gestor-de-bases-de-datos-o-sgbd/.
13. Visbal, Sara M. Artiles, *La gestión documental, de información y el conocimiento en la empresa. El caso de Cuba* (La Habana, 2009).
14. Rodrígez, Dayana Bustamante y Jean, *Metodología actuela XP*, 2014.
15. Letelier, Patricio, *Metodologías Ágiles para el desarrollo de software:eXtreme Programming (XP)*, 2010.
16. Letelier, Patricio, *Metodologías Ágiles para el desarrollo de software:eXtreme Programming (XP)*, 2010.
17. Gamma, Erich, *Patrones de Diseño.Elementos de software orientado a objetos reutilizables*, 2003.
18. Ing. Yeilenia I. Pérez Vázquez. Ing. Luís Eduardo Gallardo, *Perfil del Trabajo de Diploma* (La Habana, 2016).
19. Cañevate, Antonio Muños, *Sistema de información en las empresas*, 2007.
20. Malfará, Dayvis, *Testing en XP*, 2006.

21. Estadísticas, Instituto Nacional de, “Tipos de gráficos” (2016).

22. Canós, Lourdes, *Toma de decisiones en la empresa: proceso y clasificación*, 2012.

23. UCI, “Universidad de las Ciencias Informáticas (UCI)” (2012), http://www.uci.cu/mision.

Bibliografía

Avila, Katty. *Kavisi*, n.d. http://www.cavsi.com/preguntasrespuestas/que-es-un-sistema-gestor-de-bases-de-datos-o-sgbd/.

Calleja, Manuel Arias. *Centro de Investigación sobre Sistemas Inteligentes de Ayuda a la Decisión*, 2012. http://www.cisiad.uned.es/carmen/estilo-codificacion.pdf.

Canós, Lourdes. *Toma de decisiones en la empresa: proceso y clasificación*, 2012.

Cañevate, Antonio Muños. *Sistema de información en las empresas*, 2007.

Cienfuegos., Univercidad de. “GeReport: Sistema de Gestión de Reportes Dinámicos.” (2014).

Estadísticas, Instituto Nacional de. “Tipos de gráficos” (2016).

Gamma, Erich. *Patrones de Diseño.Elementos de software orientado a objetos reutilizables*, 2003.

Ing. Luís Eduardo Gallardo, Ing. Yeilenia I. Pérez Vázquez. *Perfil del Trabajo de Diploma*. La Habana, 2016.

Letelier, Patricio. *Metodologías Ágiles para el desarrollo de software:eXtreme Programming (XP)*, 2010.

———. *Metodologías Ágiles para el desarrollo de software:eXtreme Programming (XP)*, 2010.

Malfará, Dayvis. *Testing en XP*, 2006.

MENDOZA, MARTHA ASCENCIO. *Desarrollo de una Propuesta Metodológica para Determinar la Seguridad en una*. Prira, 2011.

Orallo, Enrique Hernández. *El Lenguaje Unificado de Modelado (UML)*. España, 2015.

———. “Ingeniería de Software” (2014).

Pascual, Efrén Santos. *Auditoria de Páginas Web*, 2007.

Pressman, Roger S. *Ingeniería de Software Un enfoque práctico Séptima edición.* Mexico, n.d.

Rodrígez, Dayana Bustamante y Jean. *Metodología actuela XP*, 2014.

UCI. “Generador Dinámico de Reporte” (2012).

———. “Universidad de las Ciencias Informáticas (UCI)” (2012). http://www.uci.cu/mision.

Universidad Nacional Autónoma de Mexico. *Gráfico de pastel*. Mexico, 2015.

Venete, Adriana. *Introducción a los patrones de arquitectura.*, 2011.

Visbal, Sara M. Artiles. *La gestión documental, de información y el conocimiento en la empresa. El caso de Cuba*. La Habana, 2009.

wordpress. *Fergarciac*, n.d. https://fergarciac.wordpress.com/2013/01/25/entorno-de-desarrollo-integrado-ide/.