Universidad de las Ciencias Informáticas.

Facultad 4

Trabajo de Diploma para Optar por el Título de Ingeniero en Ciencias Informáticas.

Título: Módulo para la Gestión Documental en el Sistema Integral de Perforación de Pozos v3

Autor(a):

Neyisleidy Valdés Sánchez

Tutor:

Ing. Carlos H. Cordoví García

Co-Tutor(a):

Ing. Patricia Ponce de León Atiés

La Habana, junio de 2018

"Año 60 de la Revolución"

"La imaginación es más importante que el conocimiento. El conocimiento es limitado, mientras que la imaginación no"

Albert Einstein.

Declaración de autoría

Declaro ser la única autora del trabajo **Módulo para la Gestión Documental en el Sistema Integral de Perforación de Pozos v3** y se autoriza a la Universidad de las Ciencias Informáticas a hacer el uso que estime pertinente con el mismo.

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

_____

Firma del autor

Neyisleidy Valdés Sánchez

_____ _____

Firma del Tutor
Ing. Carlos H. Cordoví García

Firma del Tutor
Ing. Patricia Ponce de León Atiés

Agradecimientos

Agradecimientos

Primero que todo quiero agradecerles a mis padres Nancy y Jorge, por ser los pilares importantes en mi vida. Por apoyarme en todas mis decisiones y aconsejarme siempre.

Agradecerles a mis tías Elena, Lidia y mi tío Alberto por acompañarme desde muy lejos, en todo el proceso de mi tesis, desde el día uno hasta el final. Así como desde mi comienzo en la universidad. A mi familia por todo su apoyo y dedicación.

Le agradezco a mi hermano por estar siempre preocupado por mí y mis estudios, por su apoyo y paciencia.

Le agradezco a mi tutor Carlos H. por ser guía, amigo, por dedicarme todo el tiempo posible, tanto como los fines de semana, por su paciencia durante toda la tesis.

A mi tutora Patricia, por estar siempre disponible a cada llamado mío desde el primer momento y siempre guiarme por el mejor camino.

Gracias a Elizabeth de la Paz por su tiempo y comprensión, por ser como una tutora.

A Andris por estar siempre disponible a cada minuto, hora, día que lo llamaba.

A mis profesores que contribuyeron a mi formación como profesional en especial: *Karenia por ser mi madre, amiga, una excelente profesora, y por dejarme compartir con ella cada problema que tenía.*

A la profe Elianis por ser una excelente profe guía y amiga.

A Maritza por ser una profesora genial, por estar siempre al pendiente de mí.

A Yordanki por ser un profesor que te obligue a estudiar, gracias por enseñarme a aprender.

Quiero Agradecerle a Dayron por llegar a mi vida en el momento preciso, por tener paciencia conmigo en cada momento de estrés y calmarme con mucho amor. Por apoyarme siempre y en cada momento. Te amo.

A mis amigos en especial: *Malidia y Claudia por apoyarme siempre y ser como hermanas para mí.*

A Gladis por toda su dedicación cada vez que acudía a ella.

A Lester, Ortelio, Jorge Delgado, Alberto, David Pino por ser esos amigos con los que siempre pude contar, por ser mis consejero y guías durante toda mi tesis.

A Denet por ser mi compañera de apartamento, mi amiga, hermana, por aconsejarme siempre, por no hacerlo en ocasiones y dejar que yo sola me diera cuenta de la realidad.

Le agradezco a toda mi familia de Alarcos Teatro por todo su apoyo, consejos, y por todos los momentos que pasamos juntos.

A Carlos Montenegro, Lijandy, Orlay, Yoelkis, Juan Carlos, Norberto, Leydi Laura, Yoisbel, Midiala, Cortizo, Yadelis, a mis compañeras de apartamento, a mis compañeros de aula. En fin, a todas esas personas que siempre me preguntaba cómo iba mi tesis, y a los que se preocupaban por mí.

Dedicatoria

A mi abuela Claudina Gómez que siempre me cuida desde el cielo.

A mi abuelo Adres Valdés por todo su apoyo y siempre motivarme a nunca darme por vencida.

A mis padres Jorge Valdés Gómez y Nancy Sánchez Pozo por todo su amor, comprensión, dedicación, preocupación y todo su apoyo en cada una de las decisiones de mi vida.

Resumen

El Centro de Informática Industrial es un centro productivo especializado en el desarrollo de aplicaciones industriales. El Sistema Integral de Perforación de Pozos perteneciente al centro, cuenta con una aplicación que gestiona el área de negocio de la perforación dentro de la industria del petróleo, la cual brinda cobertura a todo el proceso de perforación realizado en los pozos de petróleo del país. Sin embargo, el Sistema Integral de Perforación de Pozo no permite realizar la gestión documental, es decir no es posible crear, editar y/o eliminar documentos como: planes de trabajos, actas de reuniones, entre otros generados por la gestión del trabajo en los pozos. También se requiere exportar los documentos en formato pdf, así como generar documentos a partir de plantillas previamente creadas. Es necesidad de la empresa que la gestión documental se realice desde el propio sistema. Con el objetivo de solucionar el problema existente se realizó el estudio del estado del arte de las soluciones existentes, las bibliotecas que se utilizan para exportar documentos al formato pdf, lo que permitió la implementación de las funcionalidades solicitadas por el cliente. Como resultado del proceso de investigación y desarrollo se obtuvo un módulo para la gestión documental que al ser integrado al sistema permite la gestión de documentos.

Palabras claves: Gestión Documental, Módulo, Documento, Plantilla.

Abstract

The Industrial Computing Center is a productive center specialized in the development of industrial applications. The Integral Well Drilling System belonging to the center, has an application that manages the drilling business area within the petroleum industry, which covers the entire drilling process carried out in the country's oil wells. However, the Integral Well Drilling System does not allow document management, that is, it is not possible to create, edit and / or eliminate documents such as: work plans, minutes of meetings, among others generated by the management of work in the wells. It is also necessary to export the documents in pdf format, as well as generate documents from previously created templates. It is the company's need for document management to be carried out from the system itself. With the aim of solving the existing problem, the study of the state of the art of the existing solutions was carried out, the libraries that are used to export documents to the pdf format, which allowed the implementation of the functionalities requested by the client. As a result of the research and development process, a document management module was obtained that, when integrated into the system, allows document management.

Key words: Document Management, Module, Document, Template.

Tabla de contenido

Índice de Ilustraciones

Índice de Tablas

Introducción

Las Tecnologías de la Información y las Comunicaciones (TIC), se han convertido en parte de la cotidianidad. Debido a su desarrollo el hombre se ha visto en la necesidad de automatizar los procesos para mejorar la calidad y la eficiencia de sus actividades. Así ocurre con la sustitución de documentos en formato papel por documentos en soporte digital. Observándose así la continua evolución y adopción por la sociedad de estas tecnologías, principalmente por parte de las empresas, a las que ofrece nuevas posibilidades de comunicación.

Este nuevo soporte trajo consigo el intercambio de información y que cada día se generen, reciban y almacenen documentos en soporte digital en un mayor volumen y por disímiles empresas. Por tanto, la Gestión Documental (GD) se ha vuelto imprescindible para garantizar el uso adecuado y oportuno de la documentación.

En Cuba existen un grupo de instituciones encargadas de informatizar los diferentes procesos que se realizan en las empresas del país: una de ellas es la Universidad de las Ciencias informáticas (UCI), en la cual se encuentra el Centro de Informática Industrial (CEDIN), especializado en el desarrollo de sistemas para la industria. El Sistema Integral de Perforación de Pozos (SIPP), es un proyecto desarrollado en el centro.

SIPP cuenta con un sistema de gestión que lleva el mismo nombre del proyecto, este está orientado al área de negocio de la perforación dentro de la industria del petróleo, el cual brinda cobertura al proceso de perforación realizado en los pozos de petróleo del país. Actualmente SIPPv3 no cuenta con un módulo de gestión documental que permita el manejo de documentos directamente desde el propio sistema. Solo se generan hojas de cálculo y documentos en formato PDF *(siglas en inglés Portable Document Format<<formato de documento portátil>>)* a partir de la información almacenada en la base de datos (reportes), pero estos no persisten en la misma. Documentos como actas de reuniones y planes de trabajo son gestionados a través de terceras herramientas. Es necesidad de la empresa que estos documentos sean gestionados desde el SIPP, de modo tal que los procesos y flujo de información, tanto administrativa como de negocio se integren en el propio sistema.

Tomando en consideración la problemática planteada anteriormente se define como **Problema de la investigación:**

¿Cómo realizar la gestión de documentos en el Sistema Integral de Perforación de Pozos v3?

El **objeto de estudio** se enfoca en los sistemas para la gestión documental, definiéndose como **campo de acción:** los módulos para la gestión documental en el Sistema Integral de Perforación de Pozos v3.

Se propone como **objetivo general:** Desarrollar un módulo para la gestión documental en el Sistema Integral de Perforación de Pozos v3.

Para alcanzar el objetivo general propuesto se definen los siguientes **objetivos específicos:**

- Realizar un estudio bibliográfico sobre el estado del arte referente a los sistemas para la gestión documental.
- Realizar el análisis y diseño de la propuesta de solución.
- Implementar la propuesta de solución en el lenguaje de programación PHP.
- Realizar pruebas para garantizar la funcionalidad del sistema.

Para dar cumplimiento al objetivo general se proyectaron las siguientes **tareas de investigación:**

- Capacitación sobre la base tecnológica del SIPP.
- Elaboración del marco teórico conceptual relacionado con los aspectos teóricos que sustentan la investigación.
- Realización del análisis y diseño de la propuesta de solución.
- Implementación de componentes informáticos como soporte al modelo.
- Realización de pruebas funcionales a los componentes implementados.

Para la realización de la investigación se adopta como **idea a defender** que, el desarrollo de un módulo para la gestión documental en el SIPPv3, permitirá la gestión de documentos dentro del propio sistema.

En el desarrollo de la investigación se han utilizado **métodos teóricos y empíricos**. Entre los primeros se emplearon:

El **método analítico-sintético** se utilizó para hacer un análisis de los conceptos relacionados con la gestión documental, los sistemas de gestión documental, así como los componentes o partes que los integran. Permitiendo observar el comportamiento de los sistemas de gestión documental al estudiar sus partes individualmente, en torno al módulo para la gestión documental en el SIPPv3.

El **método de modelación** se empleó para representar los procesos y actividades de la gestión documental mediante diversos modelos y diagramas que permitan el entendimiento y comprensión del problema propuesto logrando así una propuesta de solución que responda los requisitos definidos.

Dentro de los **métodos empíricos** se emplearon:

El **método observación**, este posibilitó la realización de un estudio sobre el funcionamiento de los diferentes sistemas de gestión documental existentes, permitiendo ampliar y corroborar los elementos que se identificaron en la problemática inicial y hacen posible la propuesta de solución.

El **método análisis documental** se empleó para analizar, clasificar, seleccionar y ordenar los elementos a través de un análisis bibliográfico de la literatura, referente al proceso de gestión documental y cómo este permite elevar el control de las actividades de las empresas, por lo que se realizó la revisión de documentos, búsqueda bibliográfica, y estudio de las posiciones de varios autores en relación a la categoría gestión documental.

La presente investigación se estructura en tres capítulos, a continuación, se muestra la distribución por capítulos. Además, se abordan los aspectos tratados en cada uno de ellos:

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA DE LA INVESTIGACIÓN. En este capítulo se puntualizaron los conceptos relacionados con el tema de la gestión documental. Se realizó un estudio del estado del arte a nivel nacional e internacional referente a los sistemas para la gestión documental. Además, se abordó sobre el uso de una metodología y base tecnológica para el desarrollo de la propuesta de solución.

CAPÍTULO 2: ANÁLISIS Y DISEÑO DEL MÓDULO PARA LA GESTIÓN DOCUMENTAL EN EL SISTEMA INTEGRAL DE PERFORACIÓN DE POZOS. En este capítulo se realizó el análisis y diseño de la propuesta de solución, la descripción de requisitos por procesos, el diagrama de despliegue y el diseño de la base de datos.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS DEL MÓDULO PARA LA GESTIÓN DOCUMENTAL EN EL SISTEMA INTEGRAL DE PERFORACIÓN DE POZOS. En este capítulo se exponen aspectos a tener en cuenta en la implementación del módulo, la estrategia de pruebas diseñadas para comprobar el cumplimiento de los objetivos trazados y el análisis de los resultados obtenidos luego de su aplicación.

Capítulo 1 Fundamentación Teórica.

En el presente capítulo se exponen los elementos teóricos y conceptuales necesarios para la implementación de un módulo para la gestión documental en el SIPPv3. Con el objetivo de facilitar la comprensión del proceso y de cada elemento involucrado, se especifican los principales núcleos teóricos de la investigación. Lo que permite identificar las fuentes bibliográficas y áreas del conocimiento que abarcan las tareas de investigación y al campo de acción.

El ciclo de vida de la documentación que se maneja en una organización, constituye un elemento relevante para la celeridad de los procesos que ocurren en la misma (procesos administrativos, productivos, etc.). Por tanto, se hace necesario un adecuado manejo del flujo de la documentación, esto ha dado paso al surgimiento de la gestión documental.

1.1 Gestión Documental.

Durante la investigación se realizó un análisis sobre varias definiciones de gestión documental. A continuación, se citan diversas posiciones de los autores al respecto:

Según (Ruiz, 2015), la gestión documental es el conjunto de tecnologías, normas y técnicas que permiten a la empresa administrar su flujo de documentos a lo largo del ciclo de vida de los mismo, ya sea mediante técnicas manuales o aplicando tecnologías que permiten elevar su rendimiento, funcionalidad y eficiencia.

Por su parte Luis David Fernández Valderrama define la gestión documental como: “el conjunto de instrucciones en las que se detallan las operaciones para el desarrollo de los procesos de la gestión documental al interior de cada entidad. Tales como: producción, recepción, distribución, trámite, organización, consulta, conservación y disposición final de los documentos” (Valderrama, 2001).

Por otro lado (Emprende pyme.net, 2016), define la gestión documental como un medio para compartir, distribuir y gestionar la documentación de una empresa en soporte digital. Con el fin de automatizar los procesos, reducir costes en la empresa, tiempo y espacio, y también mejorar la gestión de los recursos de la misma.

Teniendo en cuenta los elementos comunes de las definiciones abordadas por los autores citados anteriormente y de acuerdo al alcance de la presente investigación, se define la gestión documental como: el conjunto de tecnologías, normas y técnicas que permiten a la empresa administrar su flujo de

documentos a lo largo del ciclo de vida de los mismo. Bien sea mediante técnicas manuales o aplicando tecnologías que permiten elevar su rendimiento y funcionalidad.

El proceso de la gestión documental se ha automatizado en la mayoría de las empresas, por lo que han surgido los sistemas para la gestión documental.

1.2 Sistemas para la Gestión Documental

A través del estudio de la literatura se encontraron diversas posiciones de los autores referentes a la definición sobre sistemas para la gestión documental.

Un sistema de gestión documental permite llevar a cabo la digitalización de documentos en papel. También muchos documentos son creados de forma digital para ser compartidos, tanto de forma interna como externa (EKCIT, 2015).

Por su parte Lluís Codina define a los sistemas de gestión documental como programas de gestión de bases de datos que disponen de una tecnología idónea para el tratamiento de documentos científicos, culturales y técnicos. Estos sistemas difieren en aspectos fundamentales de los de gestión de bases de datos convencionales, o de aplicación general, que se utilizan para la gestión de documentos administrativos (Codina,Lluís, 1993).

Teniendo en cuenta el estudio de las definiciones anteriores sobre sistemas para la gestión documental se adopta como definición para el desarrollo de la presente investigación la combinación de los elementos comunes de las referenciadas anteriormente: se definen a los sistemas de gestión documental como programas de gestión de bases de datos que disponen de una tecnología idónea para el tratamiento de documentos científicos, culturales y técnicos. Estos documentos son creados de forma digital para ser compartidos, tanto de forma interna como externa. Además, los documentos pueden ser creados, recepcionados y conservados.

Los sistemas de gestión documental deben ser capaces de coordinar, controlar, custodiar y recuperar la documentación generada con la eficacia necesaria para que no constituya un obstáculo en la actividad de la empresa.

Existen muchos sistemas para la gestión documental que manejan la documentación en una empresa, a continuación, se describen varios ejemplos.

1.2.1 Sistemas Similares

En la actualidad se han desarrollado varios sistemas que automatizan la gestión documental. A continuación, se enuncian algunas de las soluciones encontradas a través de la revisión bibliográfica realizada.

Nuxeo

Nuxeo es un software que permite implementar un repositorio documental corporativo. Aporta soluciones a las necesidades primarias de gestión documental de las empresas, permitiendo gestionar cómodamente documentos mediante control de versiones, flujos de trabajo asociados, publicación remota o búsqueda avanzada a texto completo, además de integración con las *suites* ofimáticas habituales tales como *Microsoft Office* y *Open Office*. Además, a través de *Nuxeo DAM* se ofrece soporte para imágenes y videos.

La implementación de este sistema es sencilla, si lo que se quiere es cubrir necesidades no muy específicas. Además, al estar desarrollado sobre estándares abiertos, tiene de entrada la facilidad de ampliar su funcionalidad mediante el desarrollo. También resulta interoperable con terceros lo cual pone al alcance de un mayor número de técnicos el conocimiento necesario para trabajar sobre él, ganando así en productividad. (Athento, 2014)

Alfresco

Es una solución versátil compatible con software tanto de la vertiente *Microsoft*, como de la rama *Linux*. Posibilita la creación y gestión de contenidos empresariales desde una gran cantidad de *Content Management System*(*CMSs)*, blogs y paquetes ofimáticos (*Office* y *Open Office*). Además, ofrece una gran variedad de herramientas colaborativas como calendarios individuales y de equipo y tableros de discusión. Su base de programación, al igual que *Nuxeo*, es *Java*, lo que los convierte en soluciones multiplataforma adaptables a cualquier entorno. Está disponible en dos versiones: *Alfresco Community* y *Alfresco Enterprise*. Ambas versiones son de código abierto, aunque su versión *Enterprise*, diseñada para importantes volúmenes de trabajo, es comercializada.

Se trata de una plataforma *Enterprise Content Management (ECM*), líder en su mercado con más de 2.500 implantaciones y probada en entornos con más de 100 millones de documentos. (EKCIT, 2016)

Entre sus principales características se encuentran:

- Facilidad de uso.
- Entorno web.

- Soporte para Gestión de Contenidos Empresariales (incluidas Gestión Documental y Gestión de Activos Digitales).
- Soporte para Gestión de Contenido Web (WCM).
- Soporte para Gestión de Contenido Social (Colaboración).

OpenKM

Es un sistema de gestión documental que permite gestionar el contenido empresarial y el flujo de trabajo de una forma eficiente. Los gestores documentales le garantizan la protección de datos al establecer la seguridad de la información del contenido empresarial.

OpenKM permite construir un repositorio con la información de la empresa, para facilitar la creación de conocimiento y mejorar la toma de decisiones en el negocio. Este unifica los grupos de trabajo e incrementa la productividad en la empresa a través de prácticas compartidas. El sistema permite mejorar las relaciones con los clientes, obtener ciclos de ventas más rápidos, además de mejorar el tiempo de comercialización de sus productos y disponer de una mejor información para la toma de decisiones (OpenKm, 2013).

Con el sistema de gestión documental de *OpenKM* se puede:

- Capturar documentos.
- Controlar la vida de los documentos.
- Recopilar información de cualquier fuente digital.
- Colaborar con sus colegas en documentos y proyectos.

En Cuba se han desarrollado varios sistemas que automatizan la gestión documental. A continuación, se describe una de las soluciones existentes, la cual fue desarrollada en la Universidad de las Ciencias Informáticas.

EXcriba

El *GDA EXcriba* o Gestor de Documentos Administrativos EXcriba, es un sistema basado en el (Enterprise Content Management) *ECM Alfresco*, que lleva su contenido a lo largo de su ciclo de vida, permitiendo la gestión de los múltiples documentos de trabajo, ya sean documentos de archivo o administrativos (Vidal, 2016).

El objetivo principal del producto es automatizar procesos documentales y archivísticos que se ejecutan dentro de cualquier entidad, desde la elaboración de un documento en su fase de inicio hasta su conservación en el archivo de gestión. El mismo está compuesto por un módulo para la interfaz de usuario

(Aplicación web), el *ECM Alfresco* como núcleo del sistema informático y herramientas de apoyo para la administración de la solución como valor añadido. Todos estos Componentes enunciados ofrecen a los clientes las bondades que Alfresco provee como repositorio documental y a la vez les brinda una interfaz amigable para interactuar entre el usuario y el software (Suárez, y otros, 2016).

A continuación, se muestran algunas de sus funcionalidades:

- Automatización de los flujos documentales
- Gestión de documentos
- Gestión de carpetas
- Control de versiones
- Control de acceso y permisos
- Notificaciones

Tabla I. Comparación de las soluciones similares [Elaboración propia].

Indicadores/Sistemas para la Gestión Documental	EXcriba	Nuxeo	Alfresco	OpenKM
Código Abierto	Sí	No	Sí	No
Basado en JavaScript	Sí	Sí	Sí	Sí
Ligero	Sí	Sí	Sí	Sí
Utiliza Symfony	Sí	Sí	Sí	Sí

Después del análisis de varios sistemas para la gestión documental, se concluyó que no se puede utilizar uno en específico para abordar la problemática propuesta. Debido a que en el caso de *Alfresco* y *EXcriba* se encuentran instalados en la Unidad Empresarial de Base de Intervención y Perforación de Pozos (UEB-IPP), pero los documentos que se generan no persisten en las tablas de la base de datos de SIPPv3. Debido a esto no se pueden hacer replicados a la UEB-IPP, además habría que trabajar con dos sistemas por separado, uno para las funciones del negocio y otro para las cuestiones administrativas. Cuando se necesita que todo esté unificado en un mismo sistema.

Del mismo modo, a la hora de hacer revisiones, auditorías o reportes, es posible analizar y procesar las réplicas enviadas desde los pozos con la información relacionada con la gestión administrativa realizada en ellos. Además, existe el inconveniente que al utilizar EXcriba para el envío de documentos a través de la red, se requiere un ancho de banda suficiente para mantener la disponibilidad del servicio. Dado que en

la UEB-IPP no se asegura la disponibilidad constante del ancho de banda requerido, es necesario realizar réplicas a las tablas de la base de datos del sistema incluyendo las relacionadas con los documentos administrativos.
En cuanto a los sistemas *OpenKM y Nuxeo*: requieren pago de licencia y no son de código abierto. Por lo que es difícil adaptarlos e integrarlos al SIPPv3.

La implantación de un sistema de gestión documental es un paso decisivo de cualquier organización dentro de su política de calidad y mejora continua, debido a que permite optimizar toda su gestión y la recuperación de la documentación. Para la implementación del módulo se utilizó la siguiente base tecnológica.

1.3 Base Tecnológica del SIPPv3

La propuesta de solución se desarrolló sobre la base tecnológica del Sistema Integral de Perforación de Pozos v3 que incluye:

AUP variación UCI Como metodología de desarrollo

(Sánchez, 2014) definió a la Metodología Proceso Ágil Unificado(AUP) variación para la UCI de la siguiente manera: para el ciclo de vida de los proyectos de la UCI las fases de estos son de Inicio, Ejecución y Cierre.

Entre las disciplinas que propone AUP-UCI para el ciclo de vida de los proyectos de la UCI se encuentran: Modelo, Implementación y Pruebas en este caso estas se dividen en 3 disciplinas: Pruebas Internas, de Liberación y Aceptación.

Algunas de las restantes disciplinas de AUP para la variación UCI asociadas a la parte de gestión, se cubren con las áreas de procesos que define CMMIDEV v1.3 (en sus siglas en ingles Capability Maturity Model Integration for Development) para el nivel 2, serían: Gestión de la Configuración (CM), Planeación de Proyecto (PP) y Monitoreo y Control de Proyecto (PMC). A partir de que el Modelado de negocio propone tres variantes a utilizar en los proyectos Casos de Uso del Negocio (CUN), Descripción de Procesos del Negocio (DPN), Modelo Conceptual (MC) y existen tres formas de encapsular los requisitos, Casos de Usos del Sistema (CUS), Historias de Usuarios (HU), Descripción de Requisitos por Procesos (DRP), surgen cuatro escenarios para modelar el sistema en los proyectos:
Escenario No 1:

Proyectos que modelen el negocio con CUN solo pueden modelar el sistema con CUS.

Escenario No 2:

Proyectos que modelen el negocio con MC solo pueden modelar el sistema con CUS.

Escenario No 3:

Proyectos que modelen el negocio con DPN solo pueden modelar el sistema con DRP.

Escenario No 4:

Proyectos que no modelen negocio solo pueden modelar el sistema con HU.

En el desarrollo del módulo se utilizó el escenario 4, puesto que aplica a los proyectos que no son muy extensos. El cliente es el que describe los requisitos y las HU, por lo que estas pueden estar en constante cambio, para así poder implementarlo, probarlo y validarlo.

Visual Paradigm-UML v8.0

(Leiva Pereira, 2012) expresa que Visual Paradigm es una herramienta de Ingeniería de *Software* Asistida por Computación *(CASE)*. La misma propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. 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. Fue diseñado para una amplia gama de usuarios interesados en la construcción de sistemas de software de forma fiable, a través de la utilización de un enfoque orientado a objetos. Se integra con diferentes entornos de desarrollo como *Eclipse* y *NetBeans* y proporciona código y compatibilidad con diferentes lenguajes como *C++*, *PHP*, *Java* y *Python*.

Symfony como marco de desarrollo

El marco de desarrollo *Symfony* permite optimizar el desarrollo de las aplicaciones web. Para empezar, separa la lógica de negocio, la lógica del servidor y la presentación de la aplicación web. Proporciona a los desarrolladores varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. También, automatiza las tareas más comunes, permitiendo a los desarrolladores dedicarse por completo a los aspectos específicos de cada aplicación. *Symfony* está desarrollado completamente con *PHP5*. Ha sido probado en numerosos proyectos reales, se utiliza en sitios web de comercio electrónico de primer nivel y cuenta con la característica de ser multiplataforma (Zaninoto, 2016).

JQuery

JQuery es una biblioteca *JavaScript* destinada a hacer programación de *JavaScript* más fácil. Una biblioteca de *JavaScript* es un conjunto complejo de código *JavaScript* que simplifica las tareas difíciles y resuelve los problemas entre navegadores. En otras palabras, *JQuery* resuelve los dos problemas de *JavaScript*: la complejidad y la naturaleza de los diferentes navegadores web. Con *JQuery*, puede realizar tareas en una sola línea de código que podría tener cientos de líneas de codificación y muchas horas de pruebas de navegador para lograr con su propio código *JavaScript*. Otra de las características de JQuery es que permite agregar funciones avanzadas a un sitio web con miles de componentes fáciles de utilizar. Por ejemplo, el complemento de la interfaz de usuario de *JQuery* le permite crear muchos elementos complejos tales como paneles con pestañas, menús desplegables y calendarios de selección de fecha emergentes, todos con una sola línea de programación (Farland, 2014).

Php5.6

PHP Hypertext Preprocessor (procesador de hipertexto), es un lenguaje interpretado de alto nivel embebido en páginas *HTML* y ejecutado en el servidor. Ofrece una solución simple y universal para las páginas web. Su diseño elegante lo hace perceptiblemente más fácil de mantener y ponerse al día en el código comparado con otros lenguajes. Debido a su amplia distribución, *PHP* está perfectamente soportado por una gran comunidad de desarrolladores (Leiva Pereira, Yusemí; 2012).

PostgreSQL9.4

PostgreSQL es uno de los Sistemas de Gestión de Bases de Datos Objeto Relacionales (*ORDBMS*) de código abierto más avanzado del mundo. *PostgreSQL* posee muchas características que normalmente sólo se encontraban en las bases de datos comerciales tales como *DB2* u *Oracle* (Korry Douglas, 2005).

Es un Sistema de gestión de bases de datos relacional orientado a objetos y libre, publicado bajo la licencia *PostgreSQL* (Douglas, 2005).

Existen diferentes tipos de herramientas que contribuyeron al desarrollo del módulo, entre ellas se encuentran los editores de textos enriquecidos. Seguidamente se muestra un análisis realizado sobre los editores de texto enriquecido.

1.3.1 Editores de texto o *Bundles* para Textos Enriquecidos.

Los editores de texto o *bundles* para texto enriquecido son componentes web que permiten a los usuarios editar y escribir texto dentro de un navegador web. Se emplean con diferentes propósitos tales como, el mejoramiento de su formulario de entrada o como parte de una aplicación web que permite la entrada de contenido generado por el usuario y con formato. Los editores de texto enriquecido son esencialmente basados en la web (TechClub Tajamar, 2016).

Tabla II. Comparación de los editores de texto [Elaboración propia].

Editores	Código abierto	Ligero	Basado en JavaScript
TinyMCE	Si	Si	No
CK Editor	Si	Si	Si
NicEdit	No	Si	Si
MarkItUp!	No	Si	Si
Damn Small Rich Text Editor	Si	Si	Si

Luego de la comparación realizada por cada uno de los indicadores definidos (Ver Tabla II) se llega a la conclusión que el editor de texto enriquecido adecuado para los propósitos de la presente investigación es el *CK Editor.*

Un documento además de proveer formato, debe ser exportado a formato *PDF*. Por tal motivo a continuación se muestra un estudio de algunas bibliotecas para exportar a este formato.

1.3.2 Bibliotecas para exportar PDF escritas en PHP

FPDF

FPDF es una clase escrita en *PHP* que permite generar documentos *PDF* directamente desde *PHP*, es decir, sin usar la biblioteca *PDFlib.* La *F* de *FPDF* significa *Free* en el inglés (gratis y libre). (FPDF Library, 2017)

Esta biblioteca tiene varias características entre ellas se encuentran:

- Elección de la unidad de medida, formato de página y márgenes
- Gestión de cabeceras y pies de página
- Salto de página automático
- Salto de línea y justificación del texto automáticos
- Admisión de imágenes (*JPEG, PNG* y *GIF*)
- Colores
- Enlaces
- Admisión de fuentes *TrueType*, *Type1* y codificación
- Compresión de página

FPDF no necesita de ninguna extensión para *PHP* (excepto *Zlib* para activar la compresión y *GD* para soporte a *GIF*) y funciona con *PHP5* (≥ 5.1).

Dompdf

En esencia, *dompdf* es un diseño *HTML* compatible y un motor de renderizado escrito en *PHP.* Es un procesador de estilo: descarga y lee hojas de estilo externas, etiquetas de estilo en línea y los atributos de estilo de elementos *HTML* individuales. Para usar *PDFlib* con *dompdf*, se requiere la extensión *PDFlib PECL*. El uso de *PDFlib* mejora el rendimiento y reduce los requisitos de memoria de *dompdf.* (GitHug, 2016)

Requerimientos:

- *PHP 5.4.0* o superior

- Extensión de *DOM*
- Extensión de *GD*
- Extensión de *MBStrign*
- *PHP-font-lib*
- *PHP-svg-lib*

mPDF

Es una biblioteca *PHP* que permite convertir *HTML* a *PDF*. Puede interpretar etiquetas *html*, *CSS* incluidos y externos con lo cual se puede lograr documentos *PDF* que son el reflejo de los *html*. Es más lento que los scripts originales, *p. HTML2FPDF* y produce archivos más grandes cuando se usan fuentes Unicode, pero admite estilos *CSS*. Es posible que se requieran extensiones adicionales para algunas funciones avanzadas como *zlib* para la compresión de recursos incrustados como fuentes o *bcmath* para generar códigos de barras o *xml* para la conversión de conjuntos de caracteres y el manejo de *SVG*. *mPDF* no es compatible con la sobrecarga de función de *PHP*. En cuanto a la importación de archivos *PDF* existentes con *FPDI*, *PHP* requiere extensión *zlib* para la compresión. (GitHub, 2017)

PHPWord

PHPWord es una biblioteca escrita en *PHP* puro que proporciona un conjunto de clases para escribir y leer desde diferentes formatos de archivo de documentos. La versión actual de *PHPWord* admite *Microsoft Office Open XML (OOXML* o *OpenXML*), *OASIS Open Document Format* para aplicaciones de *Office* (*OpenDocument* u *ODF*), *Rich Text Format* (*RTF), HTML* y *PDF*. *PHPWord* es un proyecto de código abierto con licencia bajo los términos de *LGPL* versión 3. *PHPWord* está destinado a ser un producto de software de alta calidad al incorporar integración continua y pruebas unitarias. (GitHug, 2017)

A continuación, se incluyen algunas de las acciones que se pueden realizar con la biblioteca *PHPWord*:

- Crear secciones de documentos con diferentes configuraciones, p. retrato / paisaje, tamaño de página y numeración de página.
- Crear encabezado y pie de página para cada sección.
- Establecer el tipo de fuente, el tamaño de fuente y el estilo de párrafo predeterminados.
- Inserte párrafos, ya sea como texto simple o complejo (una ejecución de texto) que contiene otros elementos.
- Insertar títulos (encabezados) y tabla de contenido.
- Insertar saltos de texto y saltos de página.
- Insertar y formatear imágenes, ya sean locales, remotas o como filigranas de la página.

- Insertar elementos de lista como con viñetas, numerados o multinivel.
- Insertar notas al pie y notas al final.

A continuación, se realiza una comparación entre las anteriores bibliotecas:

Tabla III. Comparación de bibliotecas para exportar a pdf [Elaboración propia].

	FPDF	MPDF	DOMPDF	PHPWORD
Código abierto	Si	Si	Si	Si
Symfony	SI	SI	SI	SI
Formato PDF	Si	Si	Si	SI

Para la comparación de las bibliotecas antes descritas para exportar a formato *pdf*, se tienen en cuenta los siguientes parámetros: que sean de código abierto, si se pueden utilizar en *Symfony* y que el tipo de formato a exportar sea *PDF*. Luego del análisis antes realizado se puede llegar a la conclusión que la biblioteca a utilizar en el desarrollo del módulo es *MPDF* para exportar a *PDF*, porque es una biblioteca que puede interpretar etiquetas *html*, *CSS* incluidos y externos, con lo cual se pueden lograr documentos *PDF* que son el reflejo de los *html.*

1.4 Conclusiones del capítulo

El estudio de los sistemas para la gestión documental, conjuntamente a las soluciones existentes, los *bundles* para texto enriquecido y las bibliotecas para exportar a pdf que se emplean en el mundo permitió la selección de *CK Editor* con el objetivo de proveer el formato del documento y *mPDF* como biblioteca para exportar a *PDF*. Además, se adoptaron AUP-UCI como metodología de desarrollo, Visual Paradigm como herramienta de modelado, Symfony como marco de trabajo, *JQuery* como biblioteca de *JavaScript*, *PHP* como lenguaje de programación y *PostgreSQL* como gestor de base de datos por ser empleadas en el SIPPv3.

Capítulo 2 Análisis y Diseño del Módulo para la Gestión Documental en el Sistema Integral de Perforación de Pozo.

En el siguiente capítulo se lleva a cabo la descripción de la solución propuesta para la presente investigación. Además, se presentan los requerimientos que debe tener el módulo divididos en requisitos funcionales, que muestran las funcionalidades específicas que serán implementadas y representadas a través de las Historias de Usuarios y los requisitos no funcionales, que definirán las características y restricciones que deberá cumplir la aplicación para poder funcionar correctamente.

2.1 Propuesta Solución

Se desarrolló un módulo para la gestión documental, el cual satisface la necesidad de gestionar documentos de forma electrónica, permitiendo al usuario interactuar con los mismos.

El módulo centraliza la información de los procesos documentales que se ejecutan dentro de la empresa, desde la elaboración de un documento en su fase de inicio hasta su conservación. El módulo incluye todas las acciones sobre los documentos tales como: crear, editar, mostrar, eliminar, exportar, almacenar documentos en diferentes formatos electrónicos, gestionar los trámites de los documentos que se generan y salvaguardar el patrimonio documental.

El módulo brinda además la posibilidad de crear documentos nuevos, o generar documentos a partir de plantillas existentes y guardadas en base de datos. Permitiendo mostrar un listado de estos, así como modificar y eliminar los documentos creados. Estas acciones de modificar y eliminar se realizan ya sea por corregir un error o para añadirle algún dato e información o porque ya tienen mucho tiempo en la base de datos y deben ser eliminados.
También permite realizar operaciones con los documentos, así como brindar a los usuarios involucrados con cada documento la posibilidad de saber que están relacionados con el documento mediante una notificación.

Con el desarrollo del módulo propuesto se esperan obtener los siguientes beneficios:

- Conservación de la documentación.
- Compartir y aprovechar la información como un recurso colectivo.
- Realizar las acciones sobre un documento tales como: creación, edición, eliminación y exportar un documento a pdf para imprimirlo.
- El documento contenga un formato definido e insertar imágenes en el mismo.

- Se resuelve el problema con la integración, debido a que se gestiona la documentación desde el mismo Sistema Integral de Perforación de Pozos.
- Se resuelve el problema de las réplicas desde el SIPPv3 hacia la UEB-IPP, debido a que la documentación que se genera persiste en la misma base de datos del SIPPv3.

Con el fin de comprender la propuesta de solución a continuación se presenta el modelo de dominio de la misma.

2.2 Modelo de Dominio

El modelo de dominio es utilizado como medio para comprender un dominio específico, debido a que es una representación de los conceptos y elementos de la vida real que serán empleados en el mismo. Se crea con el fin de representar los conceptos claves del dominio del sistema, las propiedades más importantes y las relaciones entre los conceptos facilitando una mejor comunicación entre desarrolladores y clientes (Larman, 2003).

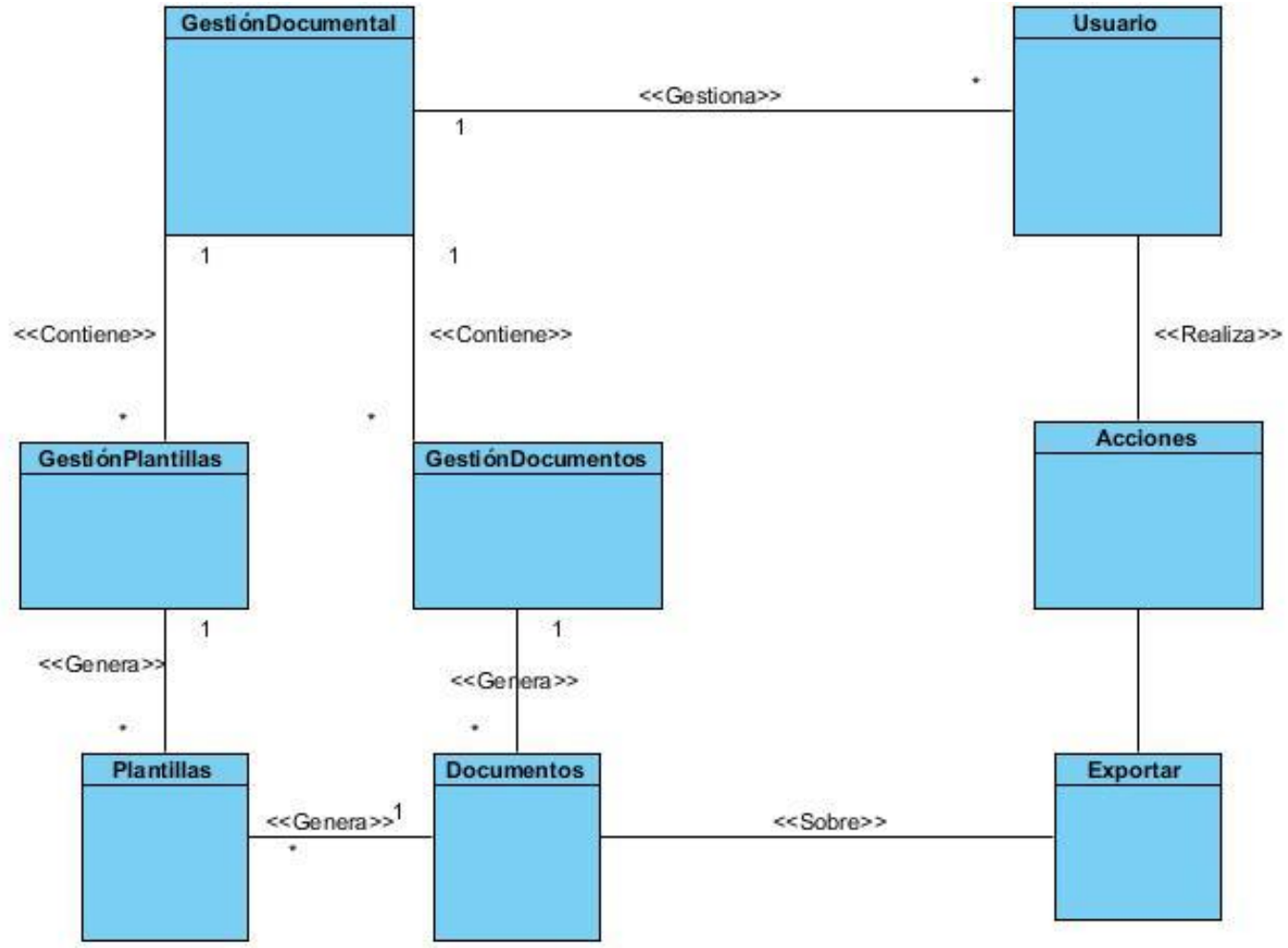


Ilustración 1. Modelo de dominio [Elaboración propia].

A continuación, se describen cada uno de los conceptos del modelo de dominio:

Usuario: Identifica a las personas que tienen acceso al módulo con el objetivo de gestionar la documentación.

GestiónDocumental: Es la vista principal del módulo que contiene las entidades Gestión Documentos y Gestión Plantillas.

GestiónDocumentos: Unidad de datos o documentos almacenados que tienen autor, descripción, nombre, fecha, e involucrados, la cual permite crear, eliminar, modificar, y listar documentos.

GestiónPlantillas: Unidad de datos o documentos almacenados que tienen autor, fecha, descripción y nombre, la cual permite crear, eliminar, modificar, y listar las plantillas.

Plantillas y Documentos: Unidad de datos o información almacenada que tiene un nombre propio y una extensión que lo identifica.

Acciones: Operaciones de importar y exportar que se realizan a los archivos.

Para la implementación del módulo, se realizó el levantamiento de los requisitos los cuales se dividió en dos grupos: funcionales y no funcionales. A continuación, se relacionan dichos requisitos.

2.3 Especificación de requisitos

La ingeniería de requisitos, como todas las demás actividades de la ingeniería del software, debe adaptarse a las necesidades del proceso, el producto y las personas que realicen el trabajo. Además, proporciona el mecanismo apropiado para entender lo que el cliente desea, analizar las necesidades, evaluar la factibilidad, validar la especificación y administrar los requisitos. (PRESSMAN, 2005)

2.3.1 Requisitos Funcionales

Tabla IV. Especificación de requisitos [Elaboración propia].

Nº	Nombre	Descripción	Prioridad	Complejidad	Referencias cruzadas
RF1	Crear Documento	El sistema debe ser capaz de crear un documento nuevo o crear un documento a partir de plantillas previamente creadas.	**Alta**	**Media**	**HU#1**
RF2	Crear documento a partir de una plantilla.	- El sistema permite crear un documento, para crear un documento a partir de una plantilla en el sistema: 1- Si el usuario desea crear un documento desde una plantilla ya predeterminada, este selecciona el botón crear plantilla, se genera la plantilla para crear el documento. - Se guarda el documento.	**Alta**	**Media**	**HU#2**
RF3	Modificar Documento	El sistema debe permitir modificar un documento, esta acción puede realizarse	**Alta**	**Media**	**HU#3**

		seleccionando la opción editar en el menú listado de documento. - Cuando el usuario modifica el documento, selecciona la opción Modificar. - Se muestra un mensaje de confirmación, se presiona el botón aceptar y se guarda el documento modificado.			
RF4	Eliminar Documento	-El sistema permite eliminar un documento, para eliminar un documento el sistema muestra un listado de los mismos permitiendo marcar un elemento a eliminar. -Selecciona la opción eliminar de las acciones mostradas en el menú y presiona la opción Eliminar. - Se muestra un mensaje de confirmación, se presiona el botón aceptar y se elimina el documento.	**Alta**	**Media**	**HU#4**
RF5	Guardar Documento	Si desea guardar un documento se puede realizar seleccionando la opción guardar. -Se muestra un mensaje de confirmación se presiona el	**Alta**	**Media**	**HU#5**

		botón aceptar y se guarda el documento creado.			
RF6	Exportar a PDF los Documentos	- El sistema permite Exportar un documento, para Exportar a PDF, selecciona la opción exportar a PDF de las acciones que aparecen en la vista de mostrar el documento creado.	**Media**	**Alta**	**HU#6**
RF7	Configurar Opciones de formato con HTM desde JavaScript	El sistema permite configurar el formato desde HTML con JavaScript en el documento. - Sistema debe tener un editor de texto enriquecido instalado el cual brinda las opciones de configurar el formato de un documento.	**Alta**	**Alta**	**HU#7**
RF8	Insertar imágenes en el documento	- El sistema permite insertar una imagen en el documento. -Se selecciona la opción de insertar imagen que aparece en sistema. - Se selecciona la imagen, Se da en la opción aceptar, la imagen se inserta en el documento. - Se guarda la operación realizada.	**Alta**	**Alta**	**HU#8**
RF9	Crear notificaciones a partir de la gestión sobre documentos para notificar a los directivos de la empresa.	El sistema debe ser capaz de crear una o más notificaciones y enviarla a los usuarios involucrados con el documento.	**Alta**	**Alta**	**HU#9**

RF10	Asignar permisos de acceso a documentos por usuarios	-El sistema debe ser capaz de asignar permiso de editar, eliminar un documento de acuerdo al usuario que este autenticado en el sistema.	**Alta**	**Alta**	**HU#10**
RF11	. Crear plantillas para generar documento.	-El sistema permite crear una plantilla, para crear una plantilla en el sistema: -El usuario va a la opción adicionar, esta acción permite crear una plantilla nueva, se genera un cuadro de texto donde puede crear la plantilla. -Seguidamente al dar clic en el botón Insertar datos se deben Introducir los metadatos, se presiona el botón aceptar. - Se muestra un mensaje de confirmación se presiona el botón aceptar. -Se guarda la plantilla.	**Alta**	**Alta**	**HU#11**
RF12	Editar plantillas para generar documento.	El sistema debe ser capaz de editar una plantilla, para editar una plantilla se selecciona la opción editar de las opciones que aparecen en el menú listado de plantillas al final de cada plantilla.	**Alta**	**Alta**	**HU#12**
RF13	Eliminar plantillas para generar	- El sistema permite eliminar una plantilla, selecciona la opción eliminar de las acciones en lote y	**Alta**	**Alta**	**HU#13**

	documento.	presiona la opción Elimina. - Se muestra un mensaje de confirmación se presiona la opción aceptar y se elimina la plantilla.			
RF14	Guardar plantillas para generar documento.	Se desea guardar una plantilla se puede realizar seleccionando la opción guardar de las opciones que muestra el propio sistema. - Se muestra un mensaje de confirmación, selecciona la opción Aceptar se guarda la plantilla.			**HU#14**
RF15	Mostrar Documento	El sistema debe permitir mostrar un documento, esta acción puede realizarse seleccionando el icono de Mostrar que aparece al final del documento. El usuario da clic en el icono y se muestra el documento con sus metadatos.	**Alta**	**Media**	**HU#15**
RF16	Mostrar Plantilla	- El sistema debe permitir mostrar una plantilla, esta acción puede realizarse seleccionando el icono de mostrar que aparece al final de la plantilla. El usuario da clic en el icono y se muestra la plantilla con	**Alta**	**Media**	**HU#16**

		sus metadatos.			

2.3.2 Requisitos no funcionales

Son aquellos que imponen restricciones en el diseño, la implementación y estándares de calidad. Para especificar los requerimientos fueron definidas categorías encargadas de recoger las cualidades que debe tener el módulo para su correcto funcionamiento y la prestación adecuada de sus servicios. (Márquez, 2017)

Usabilidad: Cuando actualiza y elimina un documento el módulo muestra un mensaje con el resultado de la acción.

Interfaz: Aplicar las restricciones definidas en el Manual de Identidad de la UEB-IPP.

Seguridad: Que solo tengan acceso al documento los usuarios autorizados.

Funcionalidad: Verificar intentos erróneos de modificación de documentos.

2.4 Historias de Usuarios

Una historia de usuario (*o user story* en inglés) describe una funcionalidad que, por sí misma, aporta valor al usuario. Se compone de: Una descripción escrita de la historia empleada como recordatorio y se planifica (debe ser breve). (Jmbeas, 2011)

Para determinar el tiempo estimado (horas) se aplicó el método comité de experto para el cual se reunieron el jefe del proyecto, el analista, el arquitecto y algunos programadores. Este tiempo estimado se le calculó a cada funcionalidad, basándose en los siguientes indicadores:

- Dificultad de la implementación.
- Cantidad de transacciones con la base de datos.
- Experiencia en el desarrollo.
- Criticidad y prioridad para el proyecto

A continuación, se presentan las historias de usuarios de los requisitos funcionales correspondientes al RF1: Crear Documento y RF3: Eliminar Documento.

Tabla V. Historia de usuario adicionar documento [Elaboración propia].

Número: 1	**Nombre del requisito:** Crear Documento

Programador: Neyisleidy Valdés Sánchez	**Iteración Asignada:** 1ra
Prioridad: Alta	**Tiempo Estimado:** 4 horas
Riesgo en Desarrollo: N/A	**Tiempo Real:** 2 horas

Descripción:

1- Objetivo:

Permitir crear documento en el sistema.

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

Para crear un documento el usuario debe:
- Estar autenticado en el sistema con el rol X.

El sistema permite crear un documento, para crear un documento en el sistema:

1- El usuario va a la opción crear un nuevo documento, esta acción permite crear el documento nuevo, se genera un cuadro de texto donde puede crear el documento.
2- Si el usuario creó el contenido en el cuadro de texto, luego debe introducir los datos después de presionar el botón insertar datos.
3- Se guarda el documento.

Observaciones:

Prototipo de interfaz:

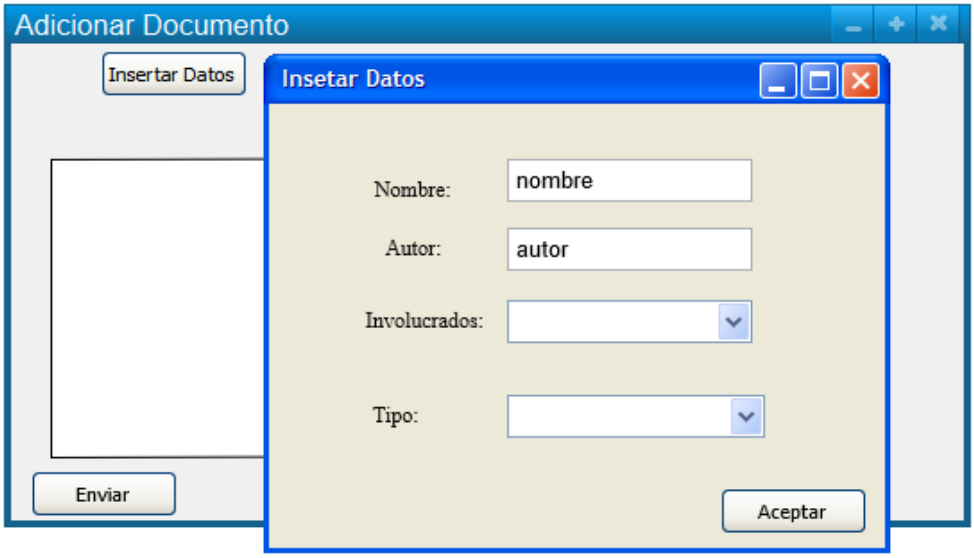


Tabla VI. Historia de usuario eliminar documento [Elaboración propia].

Número: 4	**Nombre del requisito:** Eliminar documento
Programador: Neyisleidy Valdés Sánchez	**Iteración Asignada:** 1era
Prioridad: Alta	**Tiempo Estimado:** 4 horas
Riesgo en Desarrollo:	**Tiempo Real:** 2 horas

Descripción:

1- Objetivo:

Permitir eliminar un documento en el sistema.

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

Para eliminar un documento el usuario debe:

Estar autenticado en el sistema con el rol X.

Debe existir en el sistema al menos un documento.

-Tener permiso para eliminar un documento.

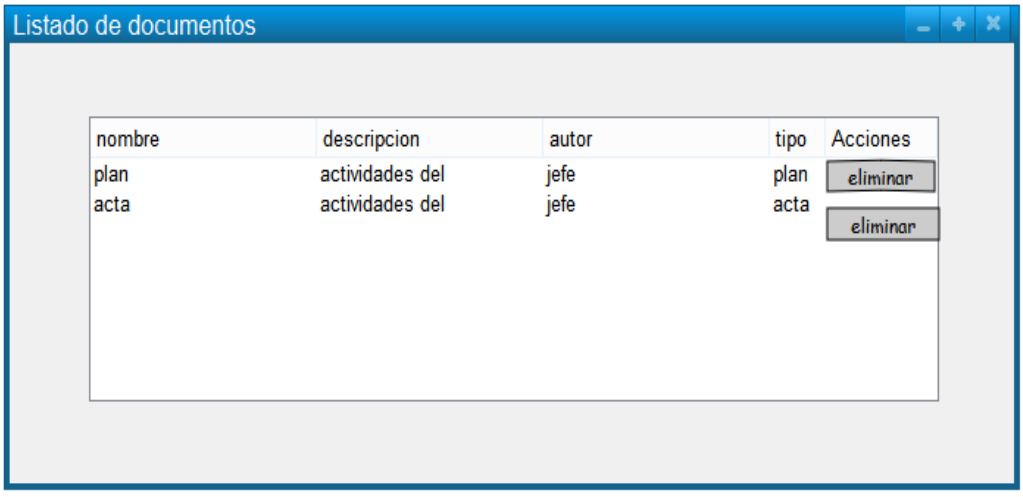
El sistema permite eliminar un documento, para lo cual muestra un listado de los documentos existentes, permitiendo marcar el elemento a eliminar.

1 - Selecciona la opción eliminar de las acciones mostradas en el menú.

2 -Se muestra un mensaje de confirmación, se presiona el botón aceptar y se elimina el documento

Observaciones:

Prototipo de interfaz:



2.5 Diagramas de Clases

El diagrama de clases recoge las clases de objetos y sus asociaciones. En este diagrama se representa la estructura y el comportamiento de cada uno de los objetos del sistema y sus relaciones con los demás objetos. (manuel.cillero.es, 2015)

Diagrama de clases del CRUD Gestionar Documento

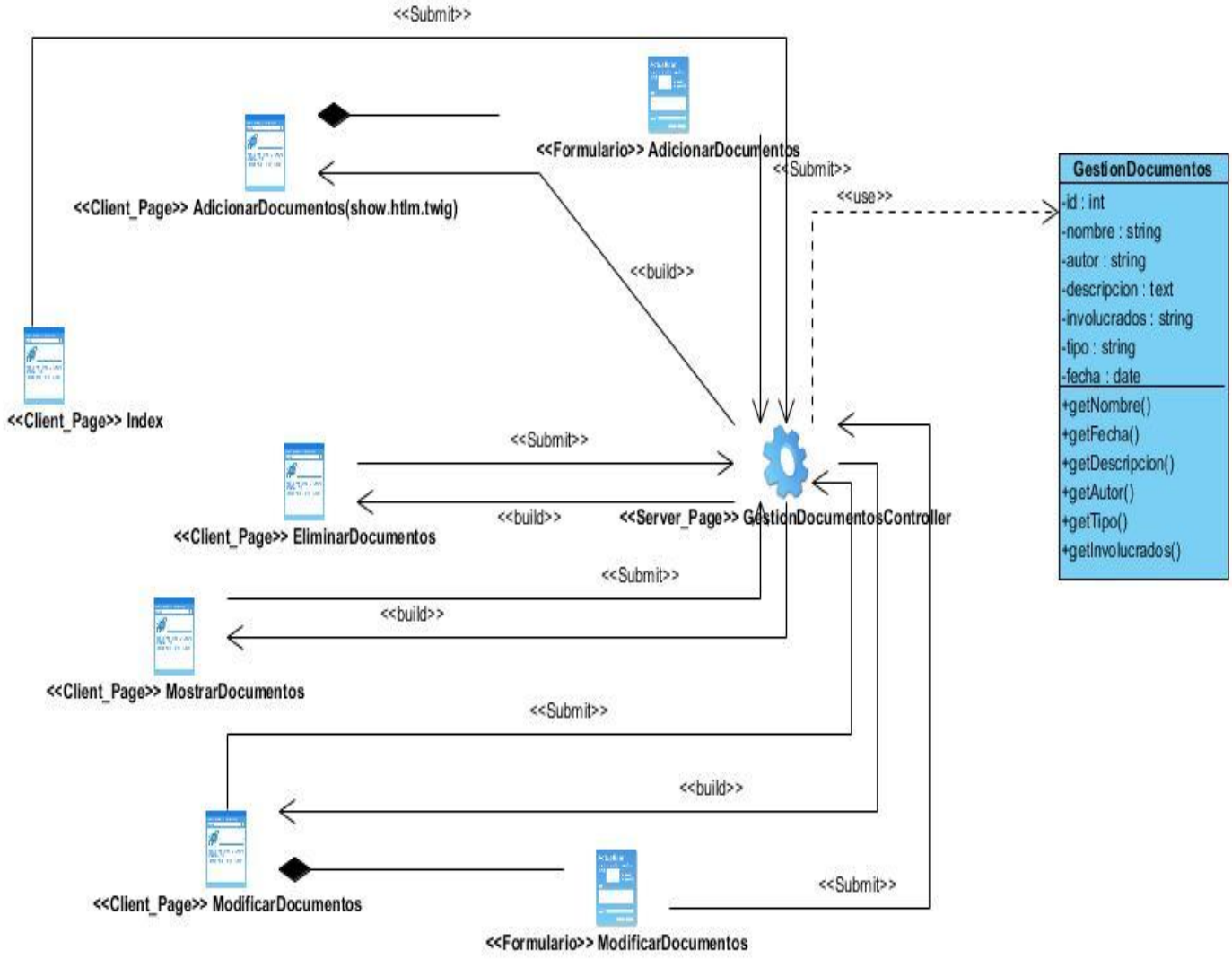


Ilustración 2. Diagrama de clases GestiónDocumentos [Elaboración propia].

Diagrama de clases del CRUD Gestionar Plantillas

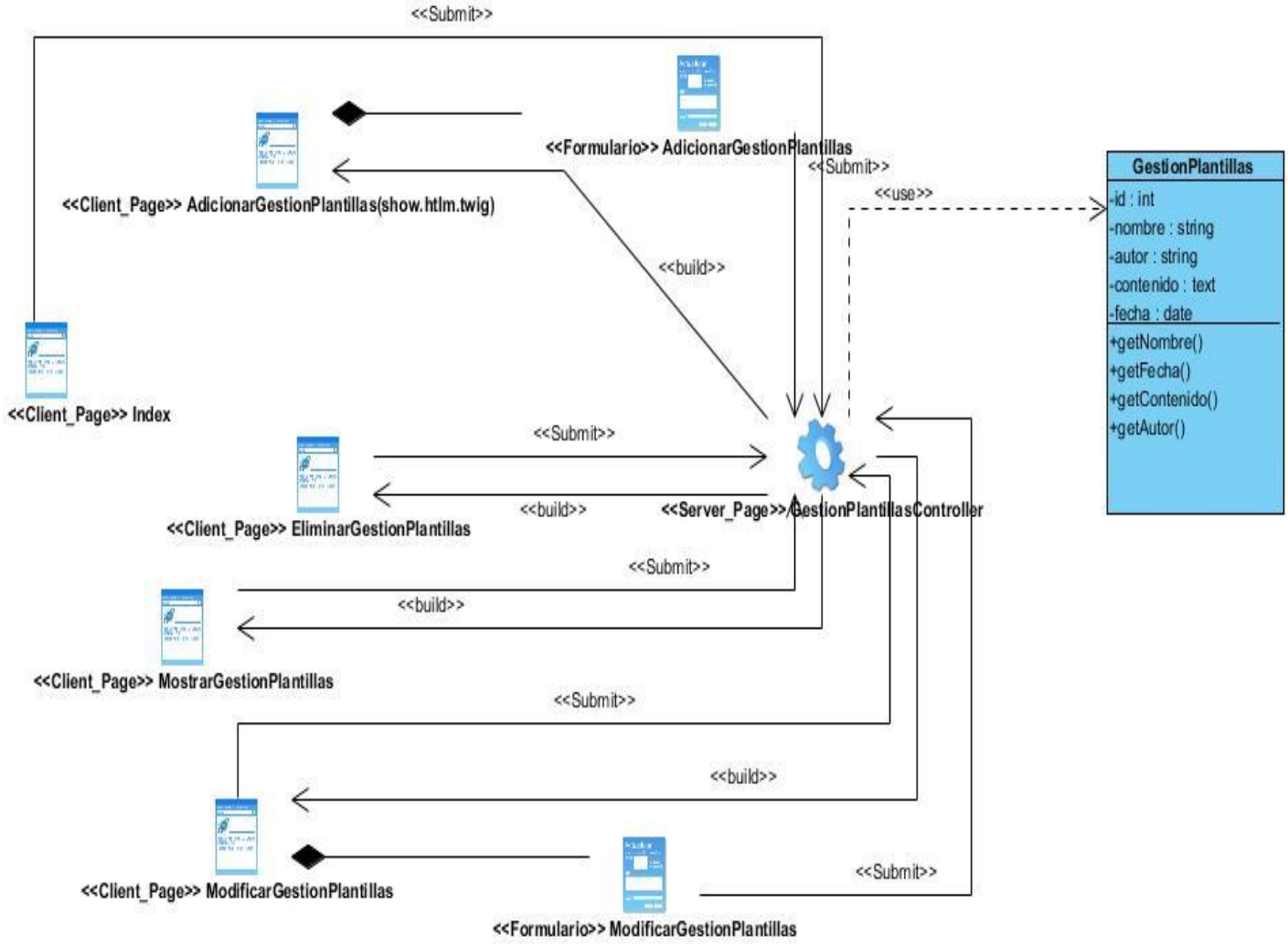


Ilustración 3. Diagrama de clases GestionPlantillas [Elaboración propia].

2.6 Patrón Arquitectónico

En Symfony, el patrón Modelo Vista Controlador (MVC) viene dado de la siguiente manera: por cada entidad de la base de datos se crea un controlador, que no es más que una clase en *PHP* con sus respectivos métodos llamadas acciones (*índex, update, register*), estos métodos renderizan las vistas o a otros métodos del modelo. (Elivar, 2017)

Modelo: El modelo contiene la parte de acceso a los datos, cuenta con las clases GestionDocumentos y GestionPlantillas que mapean los registros a la base de datos.

La Vista: Crea por cada entidad de la base de datos varias subcarpetas, dentro de cada una de las carpetas están los formularios y los contenidos twig.phtml que contienen las acciones de *show, edit* y *new* en una entidad.

El Controlador: Es la parte que interactúa con la vista y el modelo, se encuentra en el nivel medio entre la vista y el modelo, todas las peticiones que hace el cliente (vista) son pasadas al controlador GestionDocumentosController.php y GestionPlantillasController.php, el controlador redirecciona la petición, por ejemplo: si es una consulta hacia la base de datos, la envía al modelo, por su parte el modelo no devuelve esos datos a la vista directamente, sino que se los pasa al controlador y este se encarga de enviarlos a la vista.

2.7 Diagrama de Despliegue

Muestra la estructura en tiempo de ejecución del sistema, la configuración del hardware y cómo el software se distribuye en él.

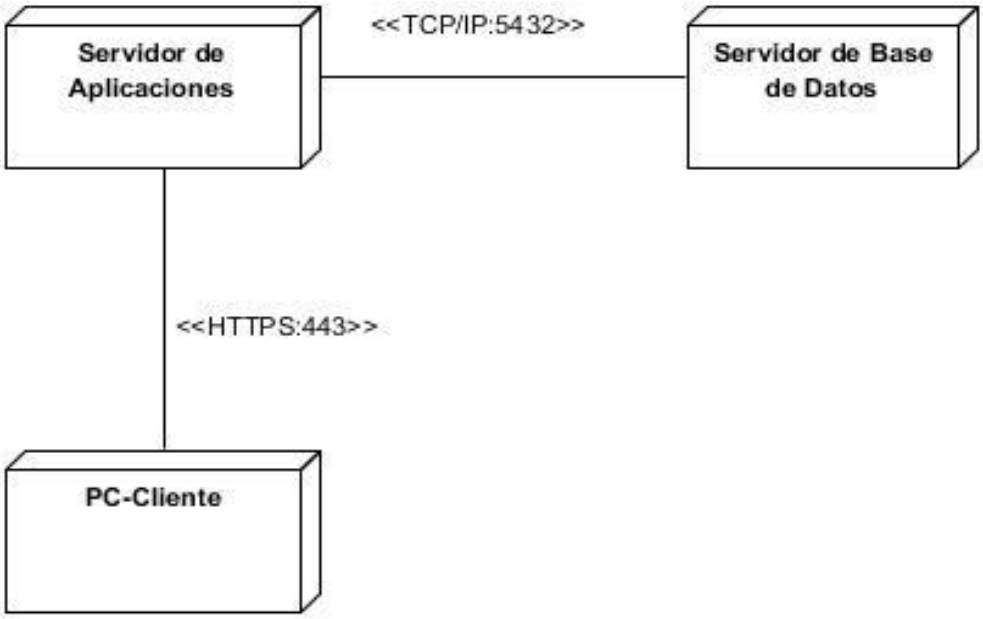


Ilustración 4. Diagrama de despliegue [Elaboración propia].

2.8 Modelo de Datos

Este modelo representa la realidad a través de entidades, que son objetos que existen y que se distinguen de otros por sus características. (Base de datos, 2016)

Para implementar la solución propuesta se añadieron tres tablas al modelo de datos del SIPPv3 las cuales tienen relación con la tabla usuario y la tabla pozo del modelo de datos del SIPPv3. En la siguiente ilustración se presentan las tablas utilizadas en la base de datos del Módulo para la Gestión Documental en el Sistema Integral de Perforación de Pozos.

Diagrama Entidad Relación del Módulo para la Gestión Documental en el Sistema Integral de Perforación de Pozos:

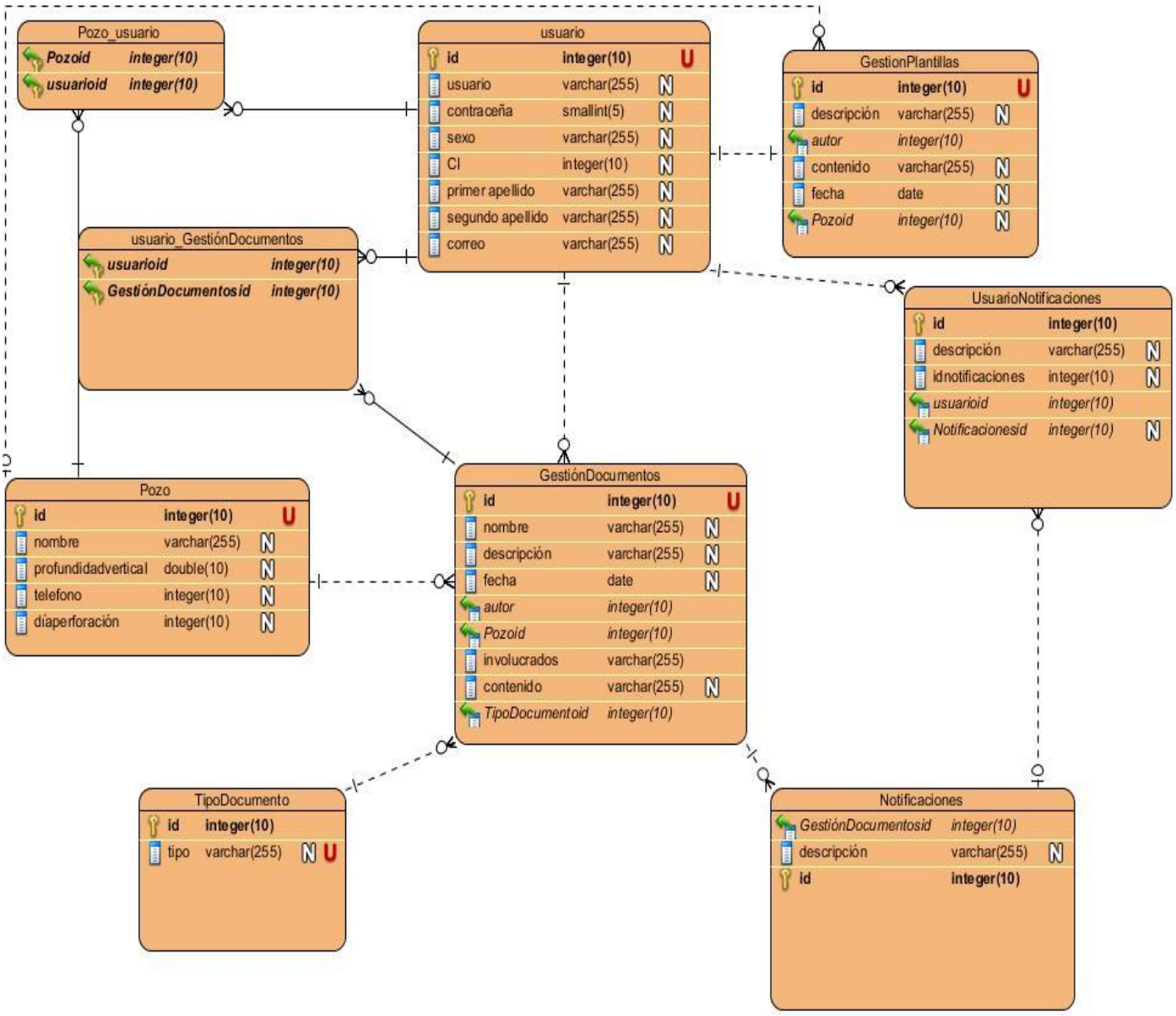


Ilustración 5. Diagrama Entidad Relación [Elaboración propia].

A continuación, en las tablas VI, VII y VIII se hace una descripción de las tablas creadas en el modelo de dato:

Tabla VII. Descripción de modelo [Elaboración propia].

Nombre	Descripción
GestionDocumentos	Contiene todos los datos de las funcionalidades para gestionar un documento.
GestionPlantillas	Contiene todo los datos de las funcionalidades para crear una plantilla.

Tabla VIII. Descripción de la tabla gestión documentos [Elaboración propia].

Atributos	Tipo	Descripción
id	Integer	Contiene todos los datos de las funcionalidades para gestionar un documento.
nombre	Varchar	Nombre del documento que fue creado.
descripción	Varchar	Campo donde se escribe el documento
fecha	Date	Fecha en la que fue creada el documento
autor	Varchar	Nombre del usuario que esta autenticado en ese momento
involucrados	Varchar	Usuarios involucrados con el documento
Pozo id	Integer	Id que muestra la relación entre las dos tablas

Tabla IX. Descripción de la tabla gestión plantillas. Elaboración propia.

Atributos	Tipo	Descripción
id	Integer	Contiene todos los datos de las funcionalidades para gestionar una plantilla.

contenido	Varchar	Campo donde se escribe la plantilla
fecha	Date	Fecha en la que fue creada la plantilla
autor	Varchar	Nombre del usuario que esta autenticado en ese momento
Pozo id	Integer	Id que muestra la relación entre las dos tablas

2.9 Conclusiones del Capítulo

Se realizó la identificación de los requisitos funcionales y no funcionales del sistema para orientar la implementación de las distintas funcionalidades del módulo. Se estudió el patrón arquitectónico Modelo Vista Controlador a través de la utilización del marco de trabajo Symfony para generar una estructura de sistema y establecer la relación entre los Componentes. Los artefactos generados durante el diseño de la solución contribuyeron a la realización del sistema para dar paso a la implementación de la solución propuesta.

Capítulo 3 Implementación y Pruebas del Módulo para la Gestión Documental en el Sistema Integral de Perforación de Pozos.

En el presente capítulo se describen los estándares de código fundamentales utilizados en la implementación de la solución propuesta. Además, se modela el diagrama de componentes para mostrar la organización y las dependencias que existen entre los ficheros que contienen código fuente. Una vez culminado el proceso de implementación se procede a aplicar un conjunto de pruebas al módulo implementado.

3.1 Implementación

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

3.1.1 Estándar de codificación

Un estándar de programación es una forma de normalizar el código la cual permite definir la estructura y organización del mismo, también facilita al programador la modificación de su propio código fuente, aunque no esté trabajando en equipo, así como definir las formas en que se deben declarar las variables, las clases y los comentarios (Emanuel, 2013).

Seguidamente se definen las pautas o estándares de codificación utilizados en la implementación de la propuesta de solución:

Estructura:

- Solo se añade un espacio después de cada delimitador coma.
- Solo se añade un espacio alrededor de los operadores.
- Se añade una coma después de cada elemento del arreglo.
- Se añade una línea en blanco antes de las declaraciones *return*.
- Se declara las propiedades de clase antes que los métodos.
- Se declaran explícitamente la visibilidad de los métodos utilizando las palabras clave*: private, protected o public.*

- Se emplean llaves para indicar la estructura del cuerpo de control, independientemente del número de declaraciones que contenga.

Convenciones de nomenclatura:

- El nombre de las clases siempre comienza con letra mayúscula, si es un nombre compuesto por más de una palabra, cada una debe comenzar con mayúscula y sin espacios. Ejemplo: la entidad <<GestionDocumentos>>
- El nombre de las variables siempre comienza con el carácter especial "$", sin espacio y escrito en minúsculas. En caso de ser un nombre compuesto por más de una palabra, cada una debe escribirse en minúscula, sin espacio y sin guiones. Ejemplo: $id y $nombre.
- El nombre de los métodos o funciones comienza con letra minúscula, si es un nombre compuesto por más de una palabra cada una debe escribirse con mayúscula y sin guiones. Con la excepción del método ajax_request (). Ejemplo: editAction (). (Symfony2 Spanish Documentation, 2011)

3.2 Patrones de Diseño

Los patrones de diseño son soluciones genéricas a los problemas comunes en el desarrollo de aplicaciones de software. Los componentes de Symfony usan muchos de los patrones de diseño debido que son necesarios para mejorar las habilidades como programador. (Hamon, Hugo, 2014), (Zaninotto, 2009). Entre los patrones de diseño se encuentran:

3.2.1 Patrones GRASP (Patrones Generales de Software para Asignar Responsabilidades)

Experto: Expresa que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. Se evidencia su uso en la ilustración, donde la clase <<GestionDocumentos>> es la responsable de conocer toda la información necesaria para realizar las acciones de crear, editar y eliminar un documento.

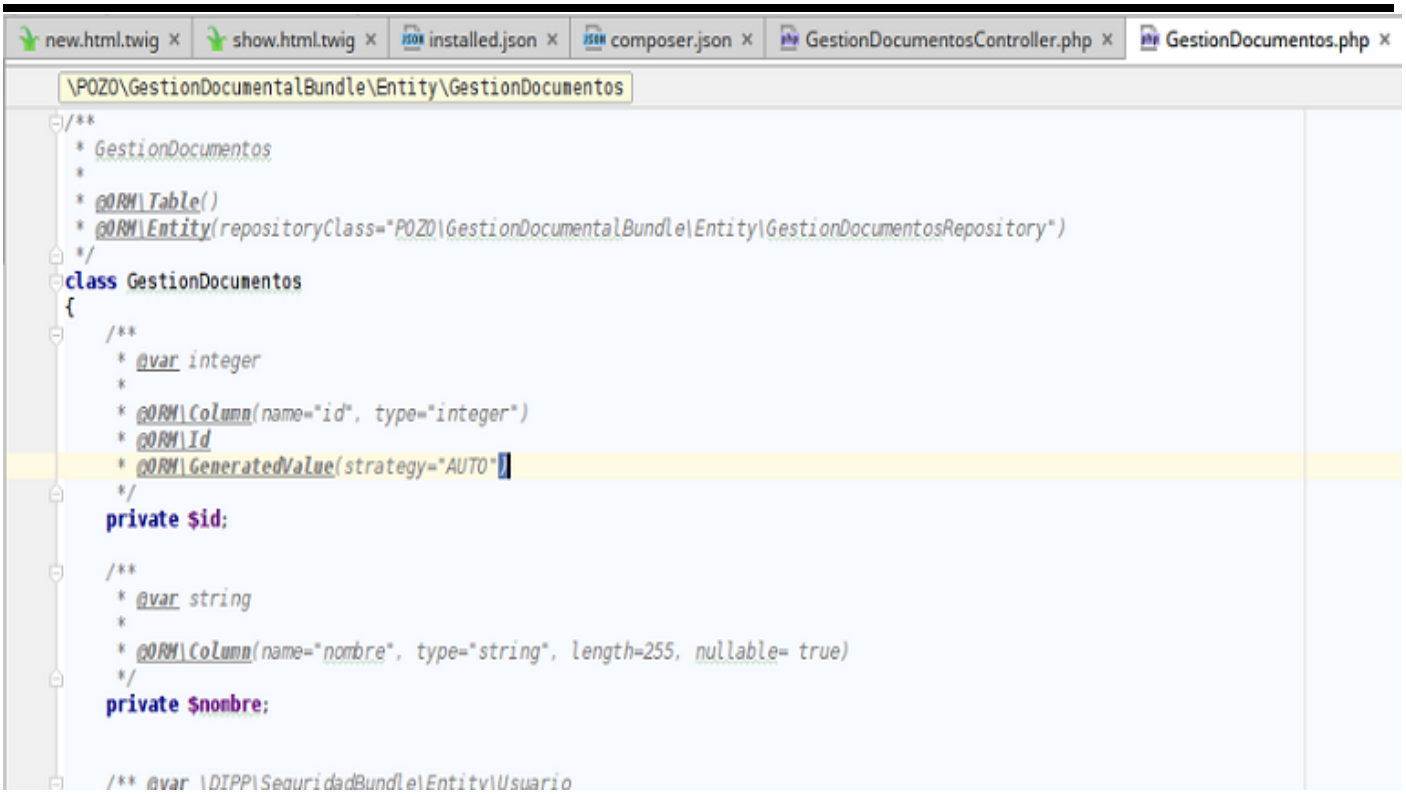


Ilustración 6. Ejemplo de uso del patrón experto [Elaboración propia].

Creador: En la clase GestionDocumentos.Controller se encuentran las acciones definidas para el sistema tales como de crear, editar, eliminar, exportar a pdf, actualizar, mostrar un documento. En dichas acciones se crean los objetos de las clases que representan las entidades, lo que evidencia que la clase Controller. php es “creador” de dichas entidades.

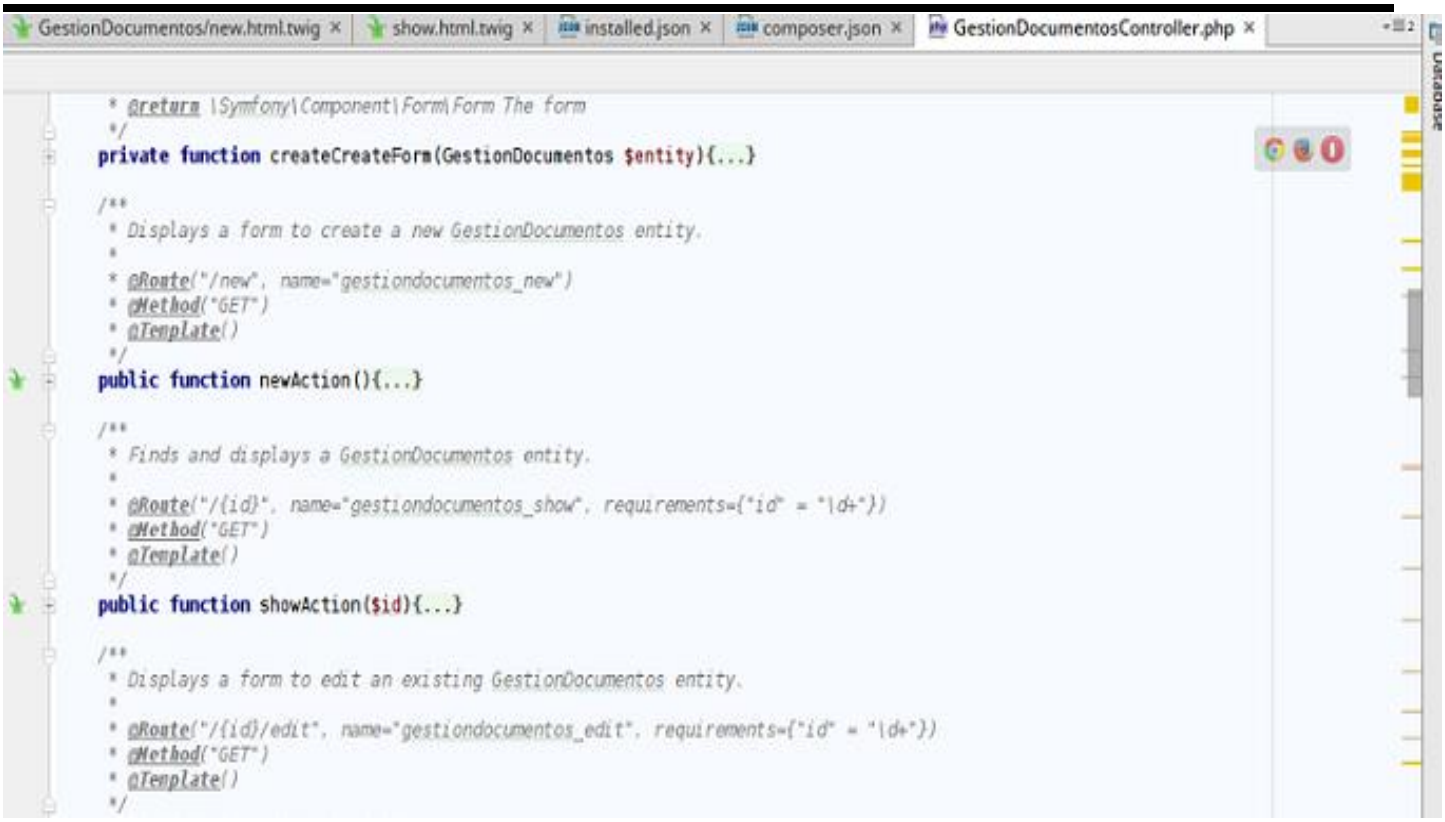


Ilustración 7. Ejemplo de uso del patrón creador [Elaboración propia].

Alta Cohesión: *Symfony* permite la organización del trabajo en cuanto a la estructura del proyecto y la asignación de responsabilidades con alta cohesión. Se evidencia el uso de este patrón en la clase GestionDocumentosController.php, la cual está formada por varias funcionalidades que están estrechamente relacionadas, siendo la misma para realizar diferentes operaciones, instanciar objetos y acceder a las propiedades.

Ilustración 8. Ejemplo de uso del patrón alta cohesión [Elaboración propia].

Bajo Acoplamiento: Plantea que las clases deben estar lo menos ligadas entre sí, posibilitando que se pueda modificar alguna clase teniendo la menor repercusión posible en el resto del módulo. Se evidencia la utilización de este patrón en el módulo, dado que la clase <<GestionDocumentosController.php>> hereda únicamente de <<Controller.php>>, además del uso de dependencia entre clases utilizando relaciones de tipo <<use>> mostrando un bajo acoplamiento de clases.

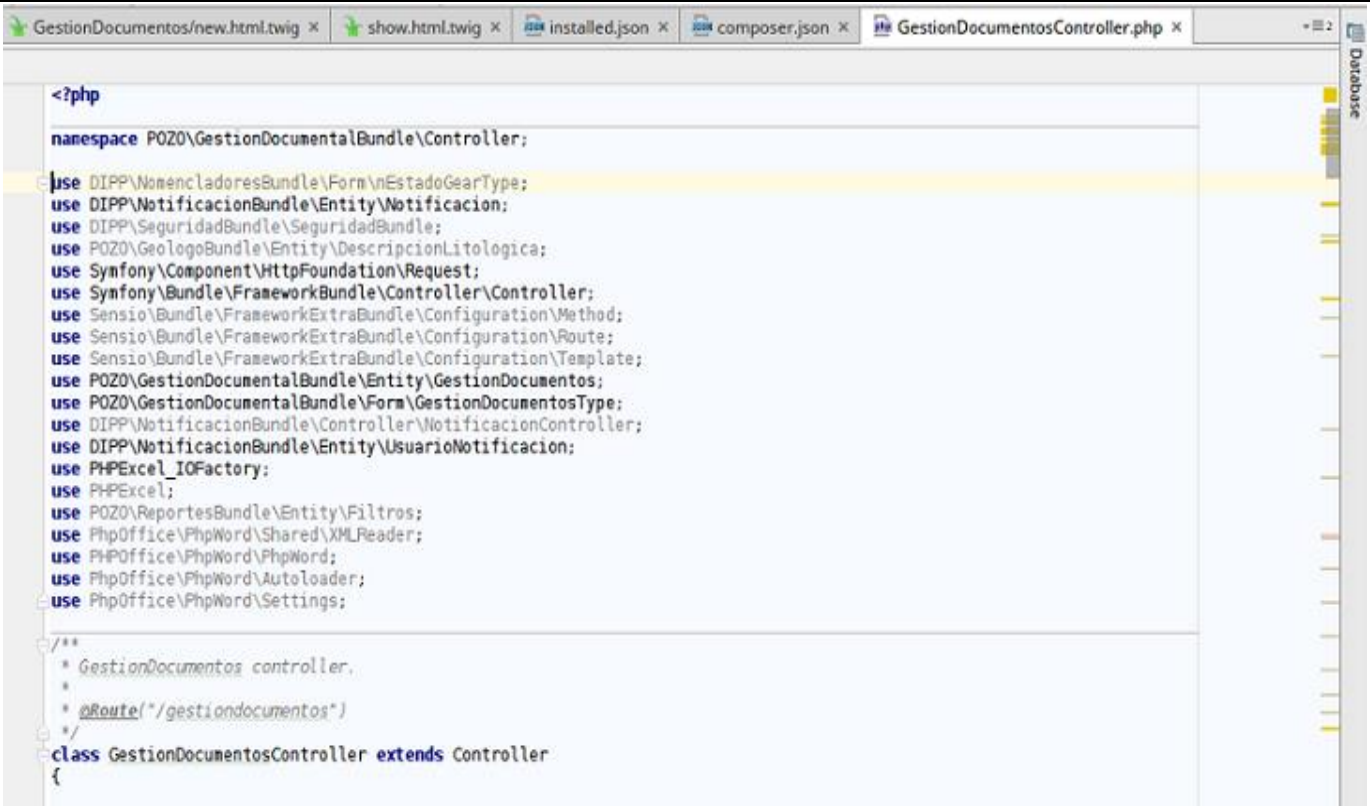


Ilustración 9. Ejemplo uso del patrón bajo acoplamiento [Elaboración propia].

Controlador: Todas las peticiones son manejadas por un solo controlador frontal (app_dev.php), que es el punto de entrada único de toda la aplicación en un entorno determinado. Cuando el controlador frontal recibe una petición, utiliza el sistema de enrutamiento para asociar el nombre de una acción y el nombre de un módulo con el Localizador Uniforme de Recursos (*URL*) entrada por el usuario. Este patrón se puede observar en la clase GestionDocumentosController.php ya que la misma se encarga de gestionar todo el flujo de trabajo con un documento.

3.2.2 Patrones GOF (Banda de los Cuatro)

Patrón Decorator: Añade funcionalidad a una clase, dinámicamente. Se evidencia el uso de este patrón en la plantilla.base.html. twig, que también se denomina plantilla global. Esta almacena el código *HTML*, *CSS* y *JavaScript* que es común a todas las páginas de la aplicación, para evitar su repetición en cada página. La plantilla global se encarga de decorar todas las vistas del sistema incluyendo las llamadas *layout*, siempre y cuando estas hereden de la plantilla.base.html. twig.

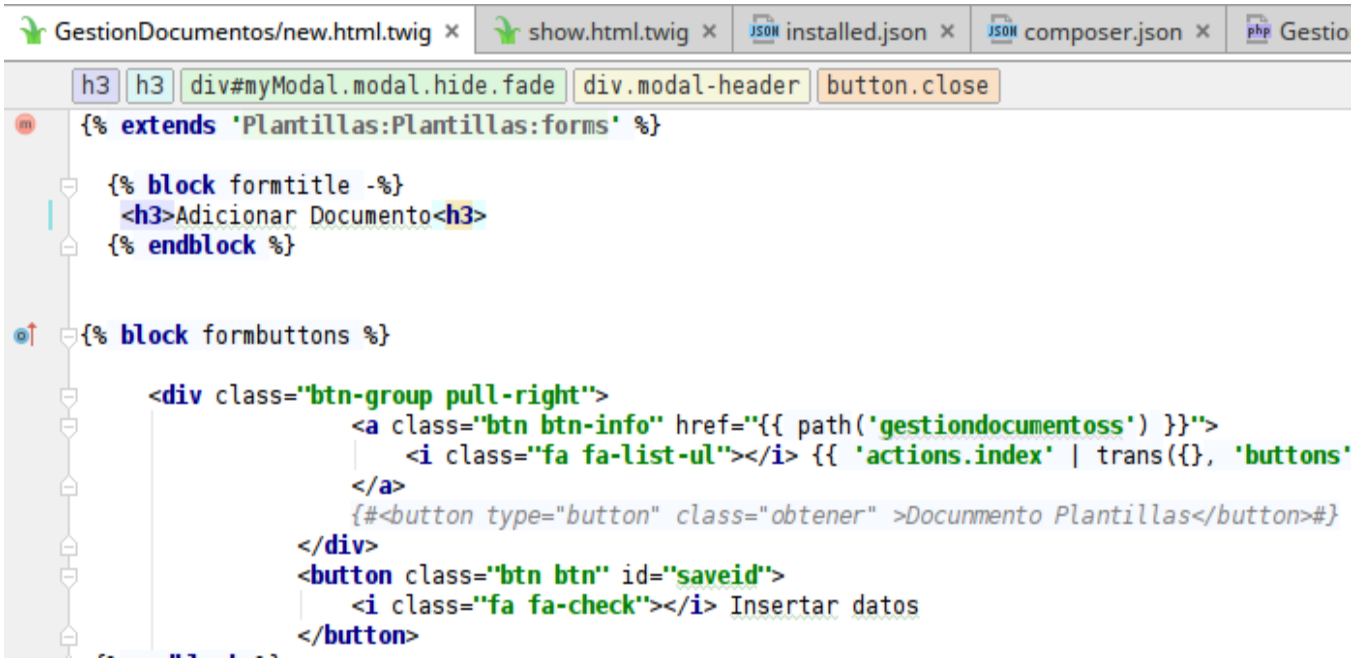


Ilustración 10. Ejemplo de uso del patrón decorador [Elaboración propia].

3.3 Diagrama de Componentes

Un diagrama de componentes describe la organización de los componentes físicos en el sistema. Estos están compuestos por componentes, interfaces y dependencias. Estos diagramas pueden incluir paquetes que permiten organizar la construcción del sistema de información en subsistemas y que recogen aspectos prácticos relacionados con la secuencia de compilación entre componentes, la agrupación de elementos en librerías (Catedra de proyecto, 2016).

Seguidamente se muestra el diagrama de Componentes del Módulo:

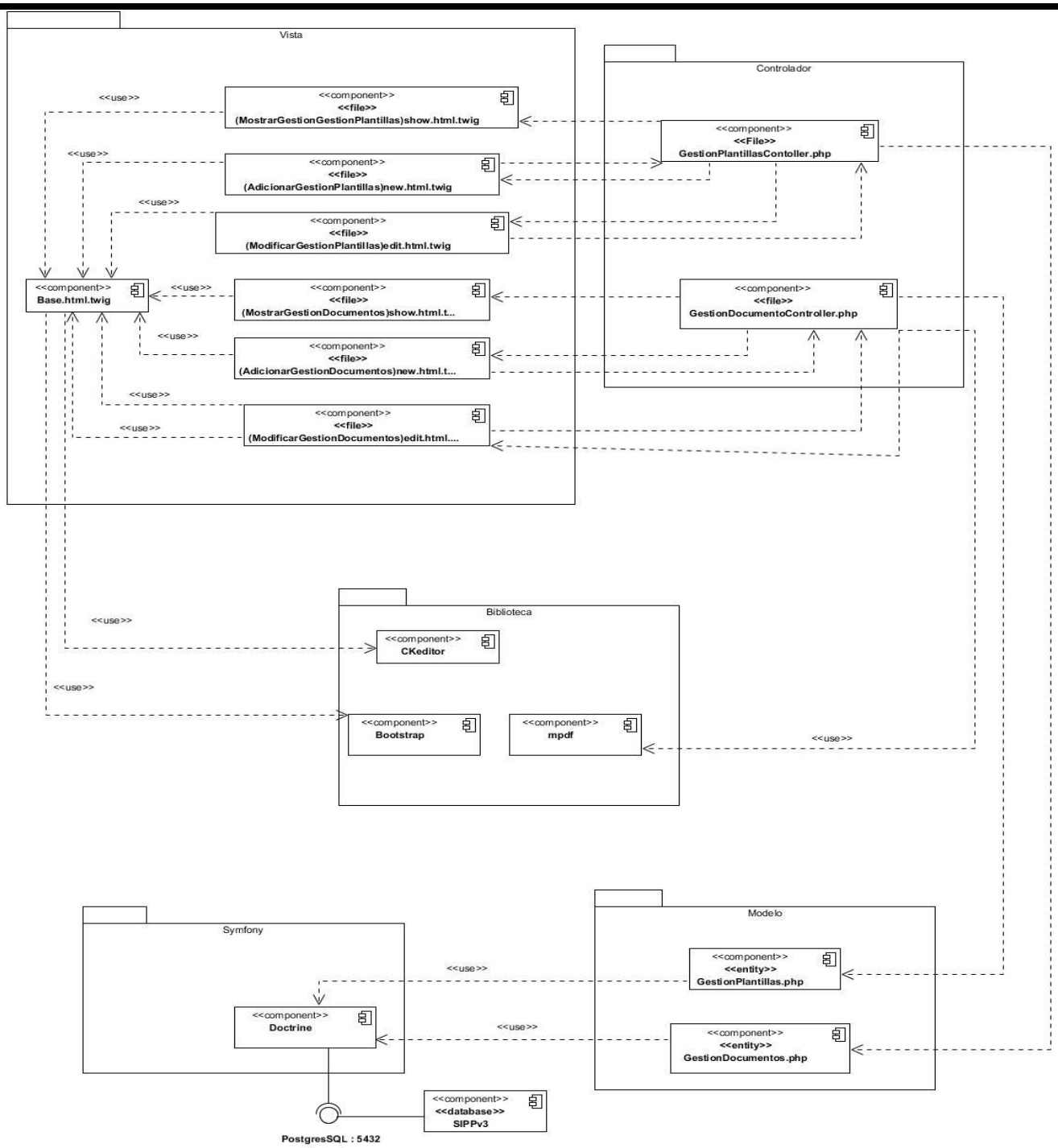


Ilustración 11. Diagrama de Componentes [Elaboración propia].

Twig: Es un motor de plantillas rápido y eficiente para *PHP*. Se recomienda para el desarrollo de aplicaciones en *Symfony2* el empleo de *Twig* para crear todas las plantillas de la aplicación. Puesto que

uno de los objetivos de *Twig* es conseguir que los diseñadores sean capaces de crear todas las plantillas de la aplicación de forma autónoma, sin ayuda de los programadores. De esta forma se acelera el desarrollo de las aplicaciones y se mejora la productividad (Pacheco, 2013).

Doctrine: Se encarga de mapear la base de datos, accede a los datos de las tablas (entidades) para consultarlos. Realiza acciones sobre los datos tales como: insertar, borrar y actualizar (Lazáro, 2015).

3.4 Pruebas

El proceso de pruebas se centra en los procesos lógicos internos del software, asegurando que todas las sentencias se han comprobado, y en los procesos externos funcionales, es decir, la realización de las pruebas para la detección de errores. En el caso de una herramienta de *software*, es necesario tener etapas de pruebas tanto para la parte funcional del *software*, como para la parte aplicativa del mismo (Pressman, 2008).

3.4.1 Estrategia de prueba

Una estrategia de prueba de software proporciona una guía que describe los pasos que deben realizarse como parte de la prueba. Aplicándose cuando se planean y se llevan a cabo dichos pasos, y cuanto esfuerzo, tiempo y recursos se requieran. Por tanto, cualquier estrategia de prueba debe incorporar la planificación de la prueba, el diseño de casos de prueba, la ejecución de la prueba y la recolección y evaluación de los resultados (In. SilderSahre,Tomado del libro Roger Pressman, 2013).

En la estrategia de pruebas se deben definir los niveles, métodos, técnicas y tipos de pruebas a realizar además de otros aspectos como condiciones de parada y ambiente de ejecución. Partiendo del análisis realizado y lo que establece el proyecto se determinó realizar pruebas de caja blanca, aplicando la técnica de camino básico. Luego se realizarán pruebas funcionales a nivel de sistema con el método de caja negra empleando la técnica partición equivalente. Además de realizar pruebas de aceptación.

3.4.2 Pruebas de caja blanca

Al módulo desarrollado se le aplicó la prueba de caja blanca, haciendo uso de la técnica camino básico, con el objetivo de evaluar la complejidad lógica de un diseño procedimental y usar esta medida como guía para la definición de un conjunto básico de caminos de ejecución (PRESSMAN, 2002). Esta prueba permite garantizar que los casos de pruebas obtenidos a través del camino básico se ejecuten cada sentencia del programa por lo menos una vez.

Las pruebas de caja blanca se centran en los detalles procedimentales del software, por lo que su diseño está fuertemente ligado al código fuente. Estas pruebas fueron realizadas, pero no están documentadas ya que se aplicaban cada vez que se implementaba una nueva funcionalidad, estas se realizaban escogiendo distintos valores de entrada para examinar cada uno de los posibles flujos de ejecución del programa y cerciorarse de que se devolvieran los valores de salida adecuados. En caso de que no devolviera la salida esperada se corregía.

3.4.3 Aplicación del método caja blanca

Como parte de la aplicación de la técnica camino básico se presentan los resultados de cada uno de los pasos que establece tomando como ejemplo el método *ShowAction*.

Paso 1. Representar el programa en el grafo de flujo

A continuación, se marcan los nodos en el código:

```
public function showAction($id)
{
    $em = $this->getDoctrine()->getManager();
1
    $entity = $em->getRepository('POZOGestionDocumentalBundle:GestionPlantillas')->find($id);

2   if (!$entity) {

        $this->get('service.message')->addError($this->get('translator')
3       ->trans('gestionplantilla.error.notfound'));
    }

    $deleteForm = $this->createDeleteForm($id);

4   return array(
    'entity'      => $entity,
    'delete_form' => $deleteForm->createView(),
  );
}
```

A continuación, se muestra el grafo de flujo asociado al método *ShowAction*:

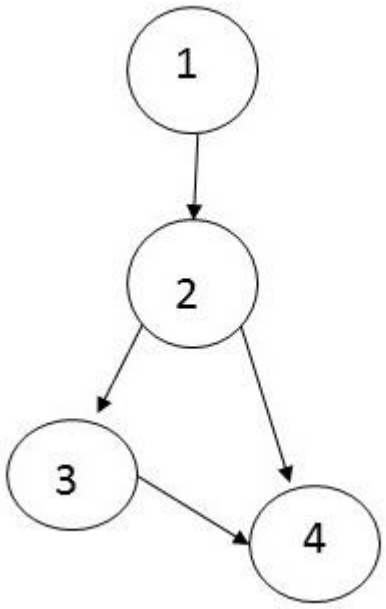


Ilustración 12. Grafo de flujo asociado al método ShowAction [Elaboración propia]

Paso 2. Calcular la complejidad ciclomática para el grafo de flujo de la Ilustración 12.

$V(g) = (R+1)(I)$ $V(g) = (a-n+2)(II)$ $V(g) = (p+1)(III)$

Siendo "V(g)" el valor de la complejidad ciclomática “a” la cantidad total de aristas, “n” la cantidad total de nodos, “p” la cantidad total de nodos predicados (nodos de los cuales parten dos o más aristas) y “R” la cantidad total de regiones, se incluye el área exterior del grafo como una región más.

Para ecuación (I)

$V(g) = 1+1 = 2$

Para la ecuación (II)

$V(g) = (4-4)+2 = 0+2 = 2$

Para la ecuación (III)

$V(g) = 1+1 = 2$

La evaluación de las formulas I, II y III arroja un valor de complejidad ciclomática igual a 2, de manera que existen 2 posibles caminos por donde el flujo puede circular. Este valor representa el número mínimo de casos de prueba para el procedimiento tratado. Seguidamente, es necesario especificar los caminos básicos que puede tomar el algoritmo durante su ejecución.

Paso 3. Identificar los caminos independientes

Tabla X . Densificación de caminos independientes [Elaboración propia].

No. de trayectorias	Trayectorias
1	1, 2, 4
2	1, 2, 3,4

Paso 4. Generar los casos de prueba

Se procede a ejecutar los casos de pruebas para cada uno de los caminos básicos determinados en el grafo de flujo (Ver Tablas XI-XII). Para definir los casos de prueba es necesario tener en cuenta:

Trayectoria: Se escribe la trayectoria del camino al que se describe.

Descripción: Se describe el caso de prueba y se especifican los aspectos fundamentales de los datos de entrada.

Resultados Esperados: Se especifica el resultado que debe arrojar el procedimiento.

Tabla XI. Caso de prueba para la trayectoria 1[Elaboración propia].

Caso de prueba para la Trayectoria Básica 1	
Trayectoria	1, 2, 4
Descripción de los parámetros de entrada	Se realiza en primera instancia una consulta ORM para luego con doctrine mediante el id solicitar todo lo que contiene la entidad. Si dicha entidad existe se crea un formulario view.(permite que una vez presionado el botón mostrar el sistema responda a la acción y retorna un arreglo con la entidad , el formulario view y el formulario delete) .
Resultado esperado	Se retorna un arreglo con la entidad y el formulario edit.

Tabla XII. Caso de prueba para la trayectoria 2 [Elaboración propia].

Caso de prueba para la Trayectoria Básica 2	
Trayectoria	1, 2, 3,4
Descripción de los parámetros de entrada	Se realiza en primera instancia una consulta ORM para luego con doctrine mediante el id solicitar todo lo que contiene la entidad. Si dicha entidad no existe se crea la siguiente excepción: Imposible encontrar la entidad
Resultado esperado	El sistema muestra un mensaje de error

	anunciando que es imposible encontrar la entidad.

3.4.4 Aplicación y resultado de la prueba de caja blanca

Se realizaron 3 iteraciones de las pruebas unitarias que arrojaron los resultados siguientes:

Las no conformidades detectadas en la realización de las pruebas de caja blanca fueron clasificadas según el nivel de complejidad para el desarrollo del módulo, en altas y medias. (DIRECCIÓN DE CALIDAD, 2016)

Altas

Error de interfaz: Son las no conformidades de tipo visual.

Excepciones: El artefacto muestra una página o mensaje de error fatal

Funcionalidad: El resultado mostrado a realizar una acción no es el esperado

Opciones que no funcionan: No muestra ningún resultado a realizar alguna acción

Validación: Son los errores relativos a la validación de los datos en las aplicaciones y funcionalidades asociadas.

Medias

Correspondencia con otro artefacto: Lo que se muestra en el artefacto que se revisa no se corresponde con lo descrito en el artefacto de apoyo.

Error de Idioma: Son aquellos errores relativos al uso de dos o más idiomas en una aplicación.

Ortografía: Existen palabras con errores ortográficos.

Redacción: Errores de redacción.

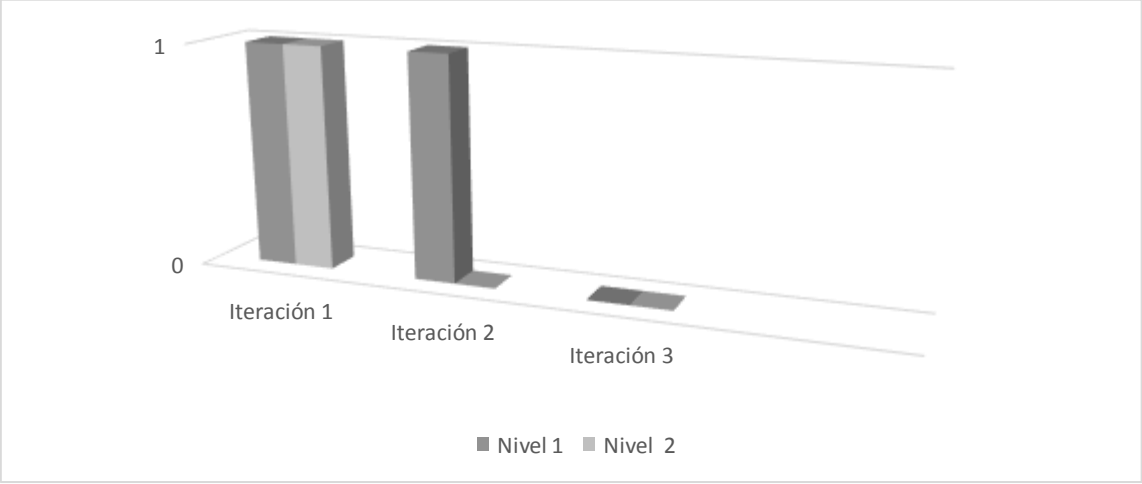


Ilustración 13. Resultado de la prueba de caja blanca [Elaboración propia].

En el gráfico anterior se evidencian 2 no conformidades detectadas al realizar la primera iteración, estas fueron clasificadas según su nivel de afectación en el funcionamiento del módulo, en nivel 1 y nivel 2. Definidas como nivel 1 aquellas no conformidades altas, como nivel 2 las no conformidades medias. Se detectó en el nivel 1 un error de variables no inicializadas. En el nivel 2 se detectó un error de variables creadas repetidas. Al concluir la corrección de los errores detectados se dio fin a la primera iteración y se prosiguió a comenzar una nueva, para comprobar si existían errores.

Una vez realizada la segunda iteración no se detectaron errores en el nivel 2 y un error que en un método no estaban cerradas, las llaves en el caso del nivel 1. Al concluir la corrección de los errores detectados se dio fin a la segunda iteración y se prosiguió a comenzar una nueva, para comprobar si existían errores Por último en la tercera iteración no se detectaron no conformidades. Los resultados generales de las pruebas se pueden ver en la (Ilustración 13).

3.4.5 Pruebas de caja negra

Las pruebas que se realizan son pruebas funcionales o pruebas de caja negra, con el fin de asegurar que el software cumpla con las especificaciones requeridas y eliminar los posibles defectos que este pudiera tener. Teniendo un enfoque en el análisis de los datos de entrada y salida. La técnica de prueba que se utilizó es la Técnica Partición de Equivalencia:

Esta técnica intenta dividir el dominio de entrada de un programa en un número finito de clases de equivalencia en el cual el campo de entrada de un programa es la sumatoria de las clases de datos. El diseño de casos de prueba para la partición equivalente se basa en una evaluación de las clases de equivalencia para una condición de entrada. Una clase de equivalencia representa un conjunto de estados válidos o inválidos para condiciones de entrada.

3.4.6 Diseño de los casos de prueba

De acuerdo con (GAVIRIA, 2015), el diseño de casos de prueba para esta técnica cuenta con dos pasos:

1. Identificar las clases de equivalencias.

2. Identificar los casos de prueba.

3.4.7 Casos de Pruebas

Tabla XIII. Caso de prueba adicionar documento [Elaboración propia].

Escenario	Descripción	Nombre	Autor	Involucrados	Tipo	Respuesta del sistema	Flujo central
EC1.1 Adicionar Documento	*Se adiciona un nuevo documento*	*V* *Plan de trabajo*	*V* *Admin*	*V* *Jefe*	*V* *Plan de trabajo*	*Se adiciona un nuevo documento, con los campos especificados, y se muestra la interfaz MOSTRAR DOCUMENTO.*	*Siguiendo la ruta mostrada a la izquierda en la interfaz principal "Otras Operaciones"-> "Gestión de Documentos", se muestra la interfaz LISTADO DE DOCUMENTOS y el botón ADICONAR al presionar el botón ¨Adicionar ¨ se muestra la interfaz ¨Crear Documento¨ para crear el documento. Luego se presiona el botón insertar datos, se introducen los valores en el campo correctamente, para después presionar el botón ¨Guardar. Muestra un mensaje de confirmación , se presiona el botón aceptar y se guarda el documento.*
		V *Acta de Reunión*	*V* *Jefe*	*V* *Supervisor*	*V* *Acta de Reunión*		
EC1.2 Adicionar Documento/valor en los campos incorrecto	*Se adiciona un documento con un valor en los campos involucrados y tipo incorrectos*	*V*	*V*	*I*	*I*	*Detecta que el valor de los campos Involucrados, Tipo no son correctos por lo que se muestra un mensaje de información y no se adiciona*	*Siguiendo la ruta mostrada a la izquierda en la interfaz principal "Otras Operaciones"-> Gestión de Documentos, se muestra la interfaz LISTADO DE*
		Plan de trabajo	*Admin*	*Sunmncb*	*Acta123*		

						el documento.	*DOCUMENTOS y el botón ADICONAR al presionar el botón ¨Adicionar ¨ se muestra la interfaz ¨Crear Documento¨ para crear el documento. Luego se presiona el botón insertar datos, que se introducen los valores en el campo incorrecto, para después presionar el botón ¨Guardar. Muestra un mensaje de información con los errores.*

Tabla XIV. Descripción de las variables del caso de prueba adicionar documento [Elaboración propia].

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	*Nombre*	*Campo de texto*	*No*	*Admitirá letras y números es un campo obligatorio.*

2	*Autor*	*Campo no editable*	*No*	*El autor se obtiene por defecto, es el usuario que esta autenticado en el sistema y ya sale en el campo.*
3	*Involucrados*	*Campo de selección*	*No*	*Se selecciona uno o más involucrados que estos pueden ser: Jefe, Supervisor, Secretaria, Nomenclador. Campo obligatorio.*
4	*Tipo*	*Campo de selección*	*No*	*Se selecciona un tipo de documento, este puede ser Plan de Trabajo o Acta de Reunión o Minuta de Reunión. Campo obligatorio.*

Tabla XV Caso de prueba adicionar plantilla [Elaboración propia].

Escenario	**Descripción**	**Nombre**	**Autor**	**Respuesta del sistema**	**Flujo central**
EC1.1 Adicionar Plantilla	*Se adicionar una nueva plantilla*	*V* *Plan de trabajo*	*V* *Admin*	*Se adiciona una nueva Plantilla, con los campos especificados, y se muestra la interfaz Mostrar Plantilla.*	*Siguiendo la ruta mostrada a la izquierda en la interfaz principal "Otras Operaciones" ->"Gestión de Plantillas", se muestra la interfaz ¨Adicionar Plantilla ¨ para crear la plantilla. Luego se presionar el botón*
		V *Acta de Reunión*	*V* *Jefe*		

					insertar datos, se introducen los valores en los campos correctamente, para después presiona el botón ¨Guardar. Muestra un mensaje de confirmación se presiona el botón aceptar y se guarda la plantilla
EC2.2 Adicionar Plantilla/valor en el campo ya existente	*Se intenta adicionar una plantilla con un valor en el campo nombre y existente*	*I* *Plan de trabajo*	*V* *Admin*	*Detecta que el valor del campo nombre ya existe. Muestra un mensaje de información de los errores, por lo que no adiciona la plantilla*	*Siguiendo la ruta mostrada a la izquierda en la interfaz principal "Otras Operaciones"->"Gestión de Plantillas", se muestra la interfaz ¨Crear Plantilla¨ para crear la plantilla. Luego se presiona el botón insertar datos, se introducen los valores repetidos en el campo, para después presionar el botón ¨Guardar¨. Se muestra un mensaje de información de los errores.*

Tabla XVI Descripción de variables del caso de prueba adicionar plantillas [Elaboración propia].

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	*Nombre*	*Campo de texto*	*No*	*Campo alfabético, permite letras y números.*

2	*Autor*	*Campo no editable*	*No*	*El autor se obtiene por defecto, es el usuario que esta autenticado en el sistema y sale en el campo.*

3.4.8 Aplicación y resultado de las pruebas de caja negra

Las no conformidades detectadas en la realización de las pruebas de caja negra fueron clasificadas según el nivel de complejidad para el desarrollo del módulo, en altas y medias (DIRECCION DE CALIDAD, 2016).

Altas

Error de interfaz: Son las no conformidades de tipo visual.

Excepciones: El artefacto muestra una página o mensaje de error fatal.

Funcionalidad: El resultado mostrado a realizar una acción no es el esperado.

Opciones que no funcionan: No muestra ningún resultado a realizar alguna acción.

Validación: Son los errores relativos a la validación de los datos en las aplicaciones y funcionalidades asociadas.

Medias

Correspondencia con otro artefacto: Lo que se muestra en el artefacto que se revisa no se corresponde con lo descrito en el artefacto de apoyo.

Error de Idioma: Son aquellos errores relativos al uso de dos o más idiomas en una aplicación.

Ortografía: Existen palabras con errores ortográficos.

Redacción: Errores de redacción.

Las pruebas aplicadas comprueban los aspectos funcionales del sistema. Las no conformidades de clasificación altas y medias detectadas se pueden apreciar en la gráfica que a continuación se presenta:

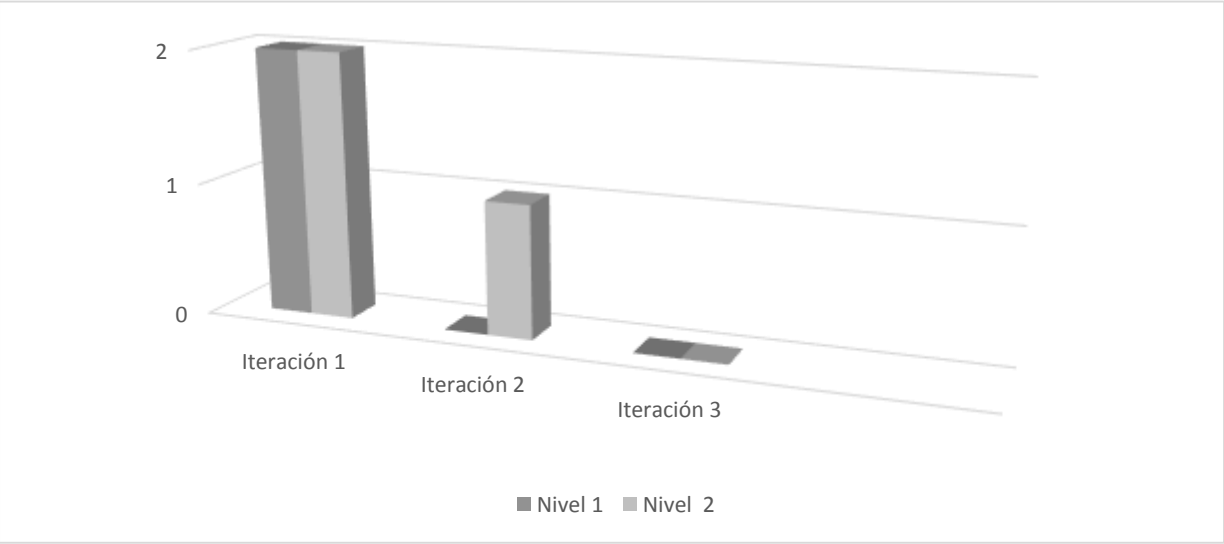


Ilustración 14. Resultado de las pruebas de caja negra [Elaboración propia].

En el gráfico anterior se evidencian 4 no conformidades detectadas al realizar la primera iteración, estas fueron clasificadas según el nivel de afectación en el funcionamiento del módulo. Definidas como nivel 1 aquellas no conformidades altas, como nivel 2 las no conformidades medias. Se detectaron en el nivel 1, un error al exportar un documento a *PDF*, un error al insertar imagen en el documento, en el nivel 2 se detectaron un error que en la misma interfaz aparecen dos idiomas indistintamente y un error de ortografía de falta de una tilde.

Al concluir la corrección de los errores detectados se dio fin a la primera iteración y se prosiguió a comenzar una nueva, para comprobar si existían errores. Una vez realizada la segunda iteración no se detectaron errores en el nivel 1 y un error ortografía de utilización incorrecta de mayúsculas y minúsculas, en el caso del nivel 2. Al concluir la corrección de los errores detectados se dio fin a la segunda iteración y se prosiguió a comenzar una nueva, para comprobar si existían errores. Por último, en la tercera iteración no se detectaron no conformidades. Los resultados generales de las pruebas se pueden observar en la (Ilustración 14).

3.4.9 Pruebas de aceptación

Estas pruebas se realizan para probar que el sistema cumpla con los requerimientos y expectativas del cliente. Estas se pueden distinguir entre dos pruebas:

- Pruebas alfa: las realiza el usuario en presencia del personal de desarrollo del proyecto haciendo uso de una máquina preparada para tal fin.

- Pruebas beta: las realiza el usuario después de que el equipo de desarrollo les entregue una versión casi definitiva del producto.

Las pruebas de aceptación realizadas fueron pruebas alfa, donde los usuarios encargados de realizarlas fue el personal de calidad del CEDIN.

3.5 Conclusiones del Capítulo

Después de haber realizado la implementación y las pruebas al Módulo para la Gestión Documental en el Sistema Integral de Perforación de Pozos, se arribó a las siguientes conclusiones:
Se desarrolló un estudio de los estándares de codificación y de los patrones de diseño, para obtener habilidades al desarrollar el módulo de forma organizada. El diseño del diagrama de componentes se realizó para definir la organización física de cada uno de los componentes del módulo. Se realizaron correcciones a las no conformidades obtenidas en el progreso de las pruebas para que el módulo desarrollado tuviese las características solicitadas por el cliente.

Conclusiones generales

- El estudio del estado del arte referido a la gestión documental permitió identificar las funcionalidades como: crear, editar, modificar, persistencia, mostrar, exportar, insertar imagen, en un documento.

- Se desarrolló un módulo para la gestión documental en el Sistema Integral de Perforación de Pozos que le brinda valor agregado a este, permite la creación, edición, eliminación, persistencia, y exportar a formato pdf un documento.

- El estudio de los *bundles* para texto enriquecido y las bibliotecas para exportar a pdf que se emplean en el mundo permitió la selección de *CK Editor* con el objetivo de proveer el formato del documento y *mPDF* como biblioteca para exportar a *PDF*.

- Las pruebas realizadas al módulo demostraron que los requisitos fueron cumplidos satisfactoriamente.

Por lo anteriormente expuesto, se puede concluir que el objetivo general de la presente investigación se ha cumplido satisfactoriamente.

Referencias bibliográficas

1. **Athento. 2014.** Athento. *Descubre los beneficios de Nuxeo.* [En línea] Athento, 2014. [Citado el: 15 de noviembre de 217.] http://www.athento.com/es/nuxeo/.
2. **Base de datos. 2016.** *www.aiu.edu/cursos/base de datos/pdf leccion 2.* s.l. : MIS, 2016. 308.
3. **Catedra de proyecto. 2016.** Diagrama UML. *Diagrama de Componente.* [En línea] 2016. [Citado el: 5 de mayo de 2018.] http://www.teatroabadia.com/es/uploads/documentos/iagramas_del_uml.pdf.
4. **Cillero, Manuel. 2015.** Manuel.Cillero.es. *Diagrama de Componente.* [En línea] 15 de enero de 2015. [Citado el: 3 de mayo de 2018.] https://manuel.cillero.es/doc/metrica-3/tecnicas/diagrama-de-componentes/.
5. *Codina, Lluís.* **documental, Qué es un sistema de gestión. 1993.** 6, Barcelona : El Profecional de la Informacion, Indexada por ISI y Scopus, 1993. ISSN1386-6710.
6. **Días, Carlo,Luis.** *Analisis y Diseño O.O.* s.l. : Fuente Inicial: Arquitectura de Software, Julio Carreño/Cesar Bustocara.
7. **DIRECCION DE CALIDAD. 2016.** *Manual del Probador.* la habana : s.n., 2016.
8. **Donatien. 2011.** Metodología de desarrollo. La Habana : s.n., 2011.
9. **—. 2009.** Sistema para la Identificación de aguas en Pozos Petroleos(SIAPP). La Habana ,Universidad de Ciencias Informatica : s.n., 2009.
10. **Douglas, Korry. 2005.** *S.D. PostgreSQL.* Estados Unidos : 2da edicion ed., 2005.
11. **EKCIT. 2016.** Alfresco, gestor de contenidos de código abierto. *tic.Portal.* [En línea] EKCIT, 2016. [Citado el: 15 de noviembre de 2017.] https://www.ticportal.es/temas/sistema-gestion-documental/programas-gestion-documental/alfresco.
12. **—. 2015.** tic.Portal. *Sistema de Gestión Documental.* [En línea] EKCIT, 2015. [Citado el: 14 de noviembre de 2017.] https://www.ticportal.es/temas/sistema-gestion-documental/que-es-sistema-gestion-documental.
13. **Elivar, Largo. 2017.** Plantilla Modelo Vista Controlador (MVC) en PHP. *Como-crear-una-plantilla-modelo-vista-controlador-para-tus-proyectos-en-php.* [En línea] 6 de Marzo de 2017.

[Citado el: 25 de 3 de 2018.] https://www.ecodeup.com/como-crear-una-plantilla-modelo-vista-controlador-para-tus-proyectos-en-php/.

14. **Emanuel. 2013.** SliderShare. *Estandar de codificación.* [En línea] Pixel16, 10 de septiembre de 2013. [Citado el: 5 de mayo de 2018.] https://es.slideshare.net/PiXeL16/estandares-de-codigo-emanuel.
15. **Emprende pyme.net. 2016.** Emprende pyme.net. *Documentos para empresas > La gestión documental en la empresa.* [En línea] 2016. [Citado el: 15 de 02 de 2018.] https://www.emprendepyme.net/la-gestion-documental-en-la-empresa.html.
16. **Farland, Mc. 2014.** *Java Script &JQuery: The Misnis Manual.* Estados Unidos : 3ra edicion ed., 2014. ISBN-131987-491-94707-4.
17. **FPDF Library. 2017.** FPDF Library ,FPDF Generator. *FPDF Generator.* [En línea] Vincent Pontainer, noviembre de 2017. [Citado el: 10 de abril de 2018.] http://www.fpdf.org/?lang=es.
18. **GAVIRIA, FLORIAN, BEATRIZ. 2015.** *PRUEBAS FUNCIONALES USANDO TÉCNICAS DE CAJA NEGRA – PARTE I.* Nueva Paz : Universidad del Valle, 2015.
19. **GitHub. 2016.** dompdf. *GitHub.* [En línea] GitHub, 2016. [Citado el: 10 de abril de 2018.] https://github.com/dompdf/dompdf.
20. **—. 2017.** mpdf. *GitHub.* [En línea] GitHub, 2017. [Citado el: 10 de abril de 2018.] https://mpdf.github.io/.
21. **—. 2017.** PHPWord. *GitHub.* [En línea] GitHub, 2017. [Citado el: 10 de abril de 2018.] https://github.com/PHPOffice/PHPWord#requirements.
22. **Gomez, M. 2013.** *Patrones Arquitectónico.* [En línea] 16 de 09 de 2013. [Citado el: 5 de 03 de 2018.] de ingenio DS:http://ingenios ds.wordpress.com/2013/09/16/patrones-arquitectonicos.
23. **Goodman, Amy y Denis, Moyniham. 2018.** Cubadebate. *La despiadada crueldad del presupuesto de Trump.* [En línea] 18 de Febrero de 2018. [Citado el: 18 de Febrero de 2018.] http://www.cubadebate.cu/especiales/2018/02/18/la-despiadada-crueldad-del-presupuesto-de-trump/.
24. **Hamon,Hugo. 2014.** Symfony.es. *Applyign Design Pattems to Symfony.* [En línea] 3 de mayo de 2014. [Citado el: 22 de 3 de 2018.] http://symfony.es/noticias/2014/05/03/los-patrones-de-diseno-de-los-componentes-symfony/.

25. **Hernández, Leovigilda. 2013.** Modelo de Implementación. *Unidad5-Modelo Implementación.* [En línea] 1 de Junio de 2013. [Citado el: 3 de mayo de 2018.] http://ithleovi.blogspot.com/2013/06/unidad-5-modelo-deimplementacion-el.html.
26. **In. SilderSahre,Tomado del libro Roger Pressman. 2013.** Estrategias prueba de software. *In. SilderSahre.* [En línea] 5 de diciembre de 2013. [Citado el: 19 de junio de 2018.] https://es.slideshare.net/celibano/estrategias-prueba-de-sw.
27. **Jmbeas. 2011.** Jmbeas. *Historias de Usuarios.* [En línea] 23 de 05 de 2011. [Citado el: 3 de 03 de 2018.] http://jmbeas.es/guias/historias-de-usuario/.
28. **Larman, Craig y Hall,Prentice. 2003.** El Modelo de Diseño. [En línea] 2003. [Citado el: 20 de febrero de 2018.] http://is.ls.fi.ump.es/docencia/is2/documentación/ModeloDominio.pdf..
29. **Lazáro, Diego. 2015.** *Guía de PHP, Introducción a Doctrine ORM.* 2015.
30. **Leiva Pereira, Yusemí. 2012.** *Diseño e Implementación del Módulo de graficar del Sistema Mnejo Integrar de Pozos.* UCI, La Habana : Tesis, 2012.
31. **Luis Sanchez, Hugo. 2015.** *Doble jueves.* La Habana : Ediciones Union, 2015. 978-959-308-200-6.
32. **manuel.cillero.es. 2015.** Mi Circunstancia Digital. *Diagrama de Clases.* [En línea] 18 de 7 de 2015. [Citado el: 15 de 03 de 2018.] https://manuel.cillero.es/doc/metrica-3/tecnicas/diagrama-de-clases/.
33. **Márquez, Escalona, Danilo, Adrian. 2017.** *Módulo Gestor de Archivos para el Agente Transitario de Cargas Transcargo.* 2017.
34. **Montesdeoca, Murillo,Raquel. 2015.** Ingenieria de Software. *DIAGRAMAS DE PAQUETES Y DE SECUENCIAS.* [En línea] 8 de Julio de 2015. [Citado el: 5 de Abril de 2018.] http://jraquelm2.wixsite.com/ingenieriadesoftware/single-post/2015/07/08/-TEMA-9-DIAGRAMAS-DE-PAQUETES-Y-DE-SECUENCIAS.
35. **OpenKm. 2013.** OpenKm. *Gestión Documental.* [En línea] OpenKm, 2013. [Citado el: 15 de novimbre de 2017.] https://www.openkm.com/es.html.
36. **Pacheco, Nacho. 2013.** *Manual de Symfony en Español.* s.l. : Sphinx 1.1.3. , 2013.
37. **Pressman, Roger. 2008.** *ANALISIS Y DISEÑO DE SISTEMAS, METODOLOGÍA DE ROGER PRESSMAN.* 2008.
38. **PRESSMAN, Roger S. 2005.** *Ingienería de sofware, un enfoque práctico. McgGraw-Hill Companies.* s.l. : ISBN 8448132149.Podware.2016.Definicion de ERP., 2005.

39. **PRESSMAN, ROGER S. 2002.** *Ingienria de Software: Un Enfoque Práctico.* s.l. : 5ta S.I: McGraw-Hill Companies, 2002. ISBN 8448132149.
40. **Rodriguez, J. 2012.** Unidad 7: Profundizando en la arquitectura MVC de Symfony — index 1.00 documentation. [En línea] 31 de junio de 2012. [Citado el: 5 de 03 de 2018.] e Unidad 7: Profundizando en la arquitectura MVC de Symfony — index 1.00 documentation:: http://juandarodriguez.es/cursosf14/unidad7.html.
41. **Ruiz, Javier,Francisco. 2015.** dataprius. *Definición de Gestión Documental.* [En línea] Dataprius, 2015. [Citado el: 20 de septiembre de 2017.] https://dataprius.com/gestion-documental.html.
42. **Sánchez, Rodríguez, Tamara. 2014.** *Programa de Mejora. Metodología de desarrollo para la Actividad productiva en la UCI.* La Habana : s.n., 2014.
43. *Sistemas de Gestión de Bases de Datos.* **Codina, Lluís. 1993.** 4, s.l. : ISI y Scopus, 1993, Vol. II. ISBN 1386-6710.
44. **Suárez, Díaz,,Yamila, labarcena, Olivares,Lisandra y Fernández, Blanco,Yaniel. 2016.** *Solución para la Implementacion y Despliegues de Flujo de Trabajo Avanzado en Sistema de Gestión Documental Eléctrónico.* La Habana : Informática 2016, 2016.
45. **Symfony2 Spanish Documentation. 2011.** test-sf-doc-es.readthedocs.io. *Estandares de Codificación.* [En línea] 2011. [Citado el: 5 de mayo de 2018.] http://test-sf-doc-es.readthedocs.io/en/latest/contributing/code/standards.html. 5682369d968fe93c45040b5addbac67576f46007. .
46. **TechClub Tajamar. 2016.** TechClub Tajamar. *Integración de editores WySiwyG en ASP.Net MVC – TinyMCE.* [En línea] TechClub tajamar, 23 de marzo de 2016. [Citado el: 20 de noviembre de 2017.] https://techclub.formaciontajamar.com/integracion-editores-wysiwyg-asp-net-mvc-tinymce/.
47. **Terrera, Gustavo. 2017.** TestingBaires. *Pruebas de Caja Negra y un enfoque práctico.* [En línea] 26 de febrero de 2017. [Citado el: 7 de mayo de 2018.] https://testingbaires.com/pruebas-caja-negra-enfoque-practico/.
48. **Valderrama, Luis David Fernández. 2001.** *IESA Instituto de Estudios Superiores en Administracion.* Caracas-Venezuela : Sociedad de la Información, 2001. ISSN: 1578-326x.
49. **Vidal, Zenia Camps. 2016.** El eXcriba una herramienta necesaria. *UCI,El eXcriba una herramienta necesaria.* [En línea] 07 de 3 de 2016. [Citado el: noviembre de 15 de 2017.] http://www.uci.cu/el-excriba-una-herramienta-necesaria.

50. **Yáñez, Hernández,Luis. 2013-2014.** *Fundamentos de la Programación.* s.l. : Licencia, 2013-2014. by-nc-sa/3.0.

51. **Zaninoto, Francois. 2016.** Symfony, La Gía Definitiva. [aut. libro] Franco F.P. *Symfony, La Gía Definitiva.* 2016.

52. **Zaninotto, F y POTENCIER, F. 2009.** *Symfony 1.1, la guía definitiva.* 2009.

Anexos

Historias de Usuarios

Número: 2	**Nombre del requisito:** Crear Documento a partir de plantilla
Programador: Neyisleidy Valdés Sánchez	**Iteración Asignada:** 1ra
Prioridad: Alta	**Tiempo Estimado:** 4 horas
Riesgo en Desarrollo: N/A	**Tiempo Real:** 2 horas

Descripción:

1- Objetivo:

Permitir crear documento en el sistema.

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

Para crear un documento el usuario debe:

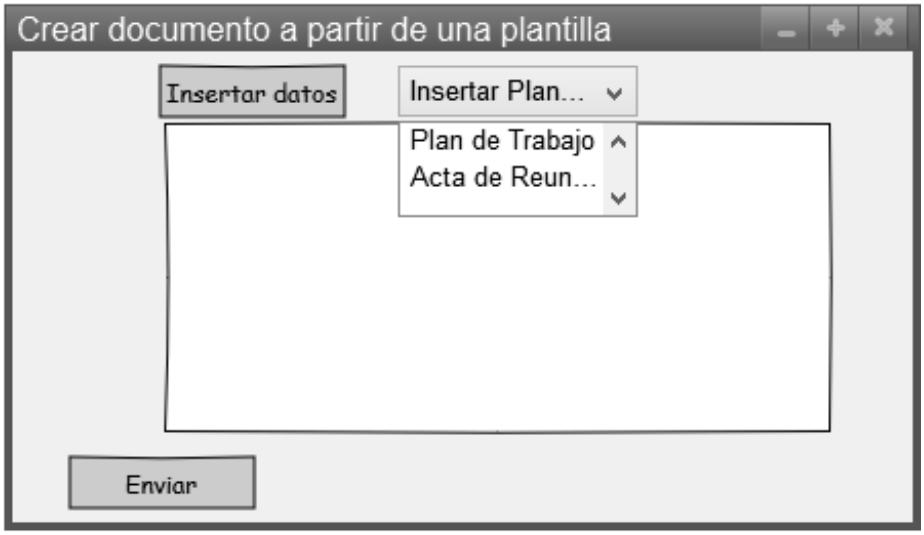
- Estar autenticado en el sistema con el rol X.

- El sistema permite crear un documento, para crear un documento a partir de una plantilla en el sistema:

1- Si el usuario desea crear un documento desde una plantilla ya predeterminada, este selecciona el botón crear plantilla, se genera la plantilla para crear el documento.

- Se guarda el documento.

Prototipo de interfaz:



Número: 3	**Nombre del requisito:** Modificar documento
Programador: Neyisleidy Valdés Sánchez	**Iteración Asignada:** 1era
Prioridad: Alta	**Tiempo Estimado:** 4 horas
Riesgo en Desarrollo:	**Tiempo Real:** 2 horas

Descripción:

1- Objetivo:

Permitir modificar documento.

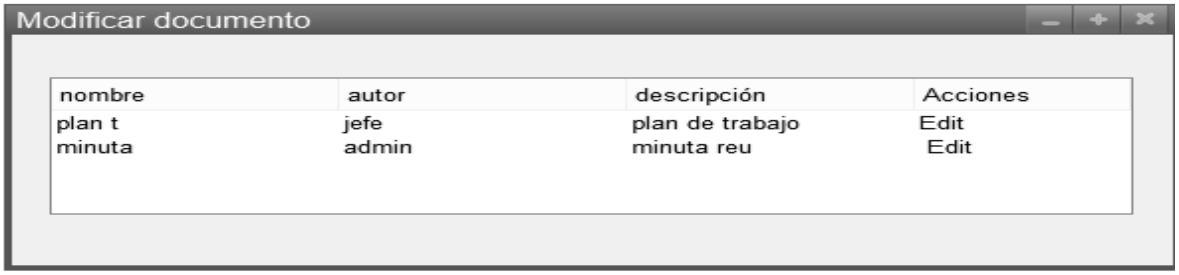
2- Acciones para lograr el objetivo (precondiciones y datos):

Para modificar un documento el usuario debe:

- Estar autenticado en el sistema con el rol X.
- Debe existir en el sistema al menos un documento.
- El sistema debe permitir modificar un documento, esta acción puede realizarse seleccionando la opción editar en el menú o desde la vista previa de la propia del documento.
- Cuando el usuario modifica de forma correcta los documentos y selecciona la opción Actualizar.

Observaciones:

Prototipo de interfaz:



Número: 5	**Nombre del requisito:** Guardar Documento
Programador: Neyisleidy Valdés Sánchez	**Iteración Asignada:** 1era
Prioridad: Alta	**Tiempo Estimado:** 4 horas
Riesgo en Desarrollo:	**Tiempo Real:** 2 horas

Descripción:

1- Objetivo:

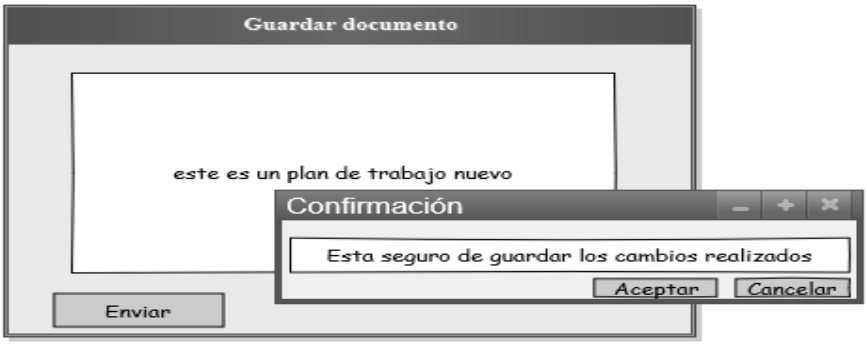
Permitir guardar un documento en el sistema.

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

Para Guardar un documento el usuario debe:

- Estar autenticado en el sistema con el rol X.
- Debe crear un documento en el sistema.
- Si desea guardar un documento se puede realizar seleccionando la opción guardar,
-Se muestra un mensaje de confirmación se presiona el botón aceptar y se guarda el documento creado.

Prototipo de interfaz:



Número: 15	**Nombre del requisito:** Mostrar Documento
Programador: Neyisleidy Valdés Sánchez	**Iteración Asignada:** 1era
Prioridad: Alta	**Tiempo Estimado:** 4 horas
Riesgo en Desarrollo:	**Tiempo Real:** 2 horas

Descripción:

1- Objetivo:

Permitir mostrar un documento.

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

Para mostrar un documento el usuario debe:

- Estar autenticado en el sistema con el rol X.

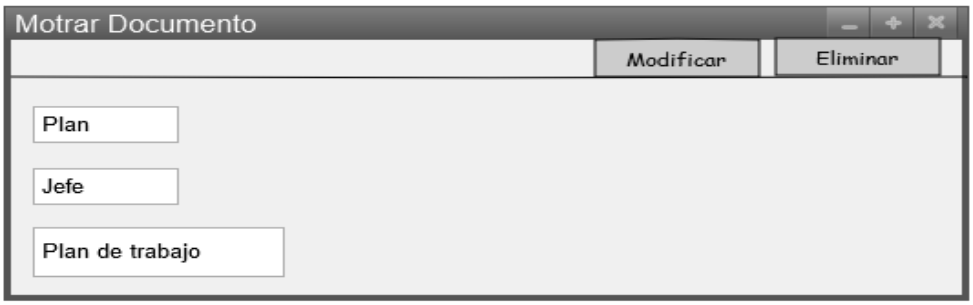
- Debe existir en el sistema al menos un documento.

-El sistema debe permitir mostrar un documento, esta acción puede realizarse seleccionando el icono de Mostrar que aparece al final del documento.

-El usuario da clic en el icono y se muestra el documento con sus metadatos.

Observaciones:

Prototipo de interfaz:



Número: 6	**Nombre del requisito:** Exportar a PDF los documentos
Programador: Neyisleidy Valdés Sánchez	**Iteración Asignada:** 1era
Prioridad: M	**Tiempo Estimado:** 4 horas
Riesgo en Desarrollo:	**Tiempo Real:** 2 horas
Descripción:	

1- Objetivo:

Permitir Exportar a PDF los documentos en el sistema.

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

Para Exportar Los Documentos el usuario debe:

- Estar autenticado en el sistema con el rol X.
- Debe existir en el sistema al menos un documento.
- El sistema permite Exportar un documento, para Exportar a PDF, selecciona la opción exportar a PDF de las acciones que aparecen en la vista de mostrar el documento creado.

Prototipo de interfaz:

Número: 11	**Nombre del requisito:** Crear Plantilla para Generar Documento
Programador: Neyisleidy Valdés Sánchez	**Iteración Asignada:** 1ra
Prioridad: Alta	**Tiempo Estimado:** 4 horas
Riesgo en Desarrollo: N/A	**Tiempo Real:** 2 horas

Descripción:

1- Objetivo:

Permitir crear documento en el sistema.

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

Para crear una plantilla para generar un documento el usuario debe:

- Estar autenticado en el sistema con el rol X.

- El sistema permite crear una plantilla, para crear una plantilla en el sistema:

-El usuario va a la opción adicionar, esta acción permite crear una plantilla nueva, se genera un cuadro de texto donde puede crear la plantilla.

Seguidamente al dar clic en el botón Insertar datos se deben Introducir los metadatos, se acepta

- Se muestra un mensaje de confirmación se presiona el botón aceptar.
- Se guarda la plantilla.

Observaciones:

Prototipo de interfaz: