Universidad de Matanzas

Facultad de Ciencias Técnicas

Ingeniería Informática

Curso 2017-2018

Trabajo de diploma para optar por el título de ingeniero informático

“Sistema web para la gestión de la información generada durante el proceso de mantenimiento a los softwares educativos”

Autor: Mario Sergio Santana Rodríguez
Tutor: DrC. Walfredo González

Matanzas
Junio, 2018

Pensamiento

Tu tiempo es limitado, de modo que no lo malgastes viviendo la vida de alguien distinto. No quedes atrapado en el dogma, que es vivir como otros piensan que deberías vivir. No dejes que los ruidos de las opiniones de los demás acallen tu propia voz interior. Y, lo que es más importante, ten el coraje para hacer lo que te dicen tu corazón y tu intuición. Ellos ya saben de algún modo en qué quieres convertirte realmente. Todo lo demás es secundario.

Steve Jobs

Dedicatoria

A mi mamá que sin ella no hubiera llegado tan lejos, por el apoyo incondicional en todo momento, por su sacrificio y entrega.

Agradecimientos

A mi madre y a mi padre por ser guías de mi vida

A mi familia, en general. Gracias a todos por su apoyo

A mi tutor Walfredo por haberme encaminado en la realización de esta investigación, por su ayuda inestimable y su continuo asesoramiento. Gracias

A todos los amigos que me ayudaron de una forma u otra de principio a fin, sin mencionar, para que no quede nadie. Gracias por su ayuda

Declaración de Auditoría

Yo, Mario Sergio Santana Rodríguez, declaro que soy el único autor de este trabajo y autorizo a la Universidad de Matanzas Sede: ¨ Camilo Cienfuegos ¨, especialmente a la Facultad de Ciencias Económicas e Informáticas y al Laboratorio de Tecnología en la Educación, a que hagan el uso que estimen pertinente de él.

Para que así conste firmo la presente a los días del mes de junio del año 2018.

-------------------------------	**------------------------------**
Firma del Autor	**Firma del Tutor**
Mario Sergio Santana Rodríguez	**Walfredo González**

Opinión del cliente y tutor del Trabajo de Diploma

TÍTULO DE LA TESIS: ¨Sistema web para la gestión de la información generada durante el proceso de mantenimiento a los softwares educativos¨

AUTOR. – Mario Sergio Santana Rodríguez.

Tutor: Dr.C. Walfredo González Hernández.

Resumen

El mantenimiento ha estado presente en todo el proceso de desarrollo de softwares específicamente los softwares educativos, por tanto. En nuestros días la gestión de la información generada en el proceso se realiza de forma manual, con el apoyo de aplicaciones tales como Microsoft Excel y Microsoft Word por lo tanto hace el proceso engorroso y lento, tampoco se ha encontrado ningún proyecto que realice de forma automatizada estos procesos de gestión en estos softwares educativos. Debido a esta problemática surge la necesidad de realizar el presente trabajo que tiene como objetivo desarrollar un sistema informático que gestione la información generada en el proceso de mantenimiento a softwares educativos de forma tal que garantice una mayor protección, confidencialidad de los datos y menor tiempo de trabajo al personal encargado del mismo. Se propone una aplicación capaz de insertar, eliminar o modificar dicha información. Para la planificación y perfeccionamiento de este proyecto fue utilizada la metodología de desarrollo ágil SCRUM, como gestor de base de datos MySQL 4.7.0, lenguaje PHP 7.1.7, como servidor web Apache 2.4.26, framework Symfony 3.3.8, el Entorno de Desarrollo Integrado (IDE) utilizado fue JetBrains PhpStorm 2017.1.4.

Summary

The maintenance has been present throughout the development process of softwares specifically the educational softwares, therefore. Nowadays, the management of the information generated in the process is done manually, with the support of applications such as Microsoft Excel and Microsoft Word for what makes the process cumbersome and slow, nor has any project been found to be carried out. automated form these management processes in these educational softwares. Due to this problem arises the need to carry out the present work that aims to develop a computer system that manages the information generated in the maintenance process of educational software in a way that guarantees greater protection, confidentiality of data and shorter work time for the personnel in charge of it. An application capable of inserting, deleting or modifying said information is proposed. For the planning and improvement of this project the SCRUM development methodology was used, as MySQL database manager 4.7.0, PHP 7.1.7 language, as Apache web server 2.4.26, Symfony framework 3.3.8, the Environment of Integrated Development (IDE) used was JetBrains PhpStorm 2017.1.4.

Índice

Introducción

Durante los primeros años del desarrollo de la informática, el software se contemplaba como un añadido. La programación de computadoras era un arte para el que existían pocos métodos sistemáticos. El desarrollo del software se realizaba virtualmente sin ninguna planificación, hasta que comenzaron a aparecer los defectos con sus respectivas soluciones, trayendo como consecuencia un aumento considerable de los costos en el desarrollo del software. Los programadores trataban de hacer las cosas bien, y con un esfuerzo heroico, a menudo salían con éxito. El software se diseñaba a medida para cada aplicación.

Frente a la considerable velocidad con que se ha desarrollado la ingeniería de computadores (hardware), ha crecido el desarrollo del software y aprovechamiento de las herramientas computacionales en la educación ha incrementado también. Es cada vez más frecuente encontrar en los diferentes niveles educativos la utilización de la computadora como objeto de estudio o como herramienta de apoyo para facilitar las labores cotidianas tales como escribir una tarea, elaborar materiales visuales, entre otras.

Todos esos programas, todas esas sentencias tenían que ser corregidos cuando se detectaban fallos, modificados cuando cambiaban los requisitos de los usuarios o adaptados a nuevos dispositivos hardware que se hubieran adquirido. Estas actividades se llamaron colectivamente mantenimiento del software. Definitivamente esta situación dejaba bien claro que muchos de estos softwares debían ser modificados y a esta etapa se le denomina mantenimiento.

Según la terminología ANSI-IEEE, el mantenimiento del software es: "…la modificación de un producto software después de su entrega al cliente o usuario para corregir defectos, para mejorar el rendimiento u otras propiedades deseables, o para adaptarlo a un cambio de entorno" (Rodríguez, 2017, p. 34)

Sin lugar a dudas, en la educación es donde mayores perspectivas actuales existen por la gran diversidad de asignaturas. Para desempeñar las mismas y más tradicionales tareas del profesor como explicar contenidos, formular preguntas sobre los mismos y comprobar los resultados la computadora juega un papel esencial. El interés de estas aplicaciones surgía ante la posibilidad de una instrucción individualizada, fundamentalmente de tipo tutorial. De aquí que la construcción de medios de enseñanza computarizados sea un reto en los momentos actuales y una inversión cuyos resultados se obtienen en tiempo futuro pero que todo país y toda política educacional tiene que tener en cuenta y tiene que desarrollar. Precisamente por tales motivos es que nuestro país se encuentra inmerso en un proceso de desarrollo y perfeccionamiento de la enseñanza a través de los medios informáticos y de la aplicación del software educativo.

Entonces, el mantenimiento de software educativo se enfoca en la corrección de errores o el mejoramiento de sus funcionalidades dependiendo de los intereses del cliente como es en cualquier caso de mantenimiento. El SE es uno de los pilares de los sistemas de enseñanza-aprendizaje a distancia que es utilizado como una herramienta para las generaciones futuras de estudiantes, de ahí que su mantenimiento depende no sólo de factores tecnológicos, pudiendo ser los factores pedagógicos los causantes de esta actividad. Para este proceso, como muchos en la ingeniería del software, se hace necesario una metodología que guíe y oriente las acciones a ejecutar para lograr el mantenimiento.

Una metodología ayuda y guía de una manera óptima y factible la creación de dichos SE por lo que se han propuesto diferentes metodologías para la gestión de software educativos (Bernardinia, Porayska-Pomstab, & Smith, 2014; Bul et al., 2015; Coomans & Lacerda, 2015; Choi, Seok, Choi, & Kim, 2015; Suh, 2014; Xie, Tillmann, & Halleux, 2013) y cada uno de ellos han tenido presente la importancia del mantenimiento en sus trabajos.

En efecto, con la utilización de los softwares educativos ofrecen nuevas formas para lograr el aprendizaje por parte de los estudiantes aumentando la probabilidad de cambio en su código, de nuevas estrategias, corrección de errores detectados a lo largo de su vida práctica, actualizaciones o simplemente adaptaciones a nuevos locales, sería muy difícil estas operaciones sin aplicar el mantenimiento de software, convirtiéndose en un pilar fundamental y de los más importantes en la creación de un SE.

En lo que a Cuba respecta las investigaciones sobre procesos ingenieriles en el desarrollo y mantenimiento de software educativo adolecen de empirismo y falta de formalidad. En entrevistas realizadas a los profesores de la enseñanza media se evidencia la necesidad de modificar el software educativo que están implantados en las escuelas por diversas deficiencias que se detectan. Sin embargo, a pesar de existir necesidad de modificarlos no se modifican. En encuestas aplicadas a los miembros de procesos de desarrollo de software educativo en la Universidad de Matanzas y en el resto del país se detectan insuficiencias en el desarrollo del mantenimiento.

Situación problémica

Dada la amplia y concreta investigación realizada, la metodología para el mantenimiento de software “EDUMANTE” presenta una amplia documentación para la implementación de estos procesos. Ello unido a la ausencia de una estructura óptima y factible para el desarrollo web, además de gran cantidad de trabajadores que intervienen; son en general razones precisas que apuntan a la complejidad del problema. Todos estos procesos se llevan a nivel de documentos Word y Excel, los cuales atentan contra la seguridad y la gestión de estos procesos.

Problema científico

Basado en la situación anterior se deduce el siguiente problema:

¿Cómo gestionar el mantenimiento de un software educativo en los laboratorios de tecnologías en la educación?

Objeto de estudio

La gestión de la información generada durante el proceso de mantenimiento de los softwares educativos.

Objetivo

Desarrollar un sistema web que gestione la información del proceso de mantenimiento de un software educativo en los laboratorios de tecnologías en la educación.

Campo de acción

Informatización de la gestión de la documentación asociada al mantenimiento de un software educativo.

Preguntas científicas planteadas

¿Cuál es el marco teórico referencial para el desarrollo de un sistema web que gestione el mantenimiento de un software educativo?

¿Cómo implementar un sistema web para una metodología de mantenimiento a un software educativo en los laboratorios de tecnologías en la educación?

¿Cómo validar un sistema web para gestionar el mantenimiento de un software educativo en los laboratorios de tecnologías en la educación?

Tareas de investigación

Determinar el marco teórico referencial para el desarrollo de un sistema web que gestione el mantenimiento de un software educativo.

Implementar un sistema web para una metodología de mantenimiento a un software educativo en los laboratorios de tecnologías en la educación.

Validar un sistema web para gestionar el mantenimiento de un software educativo en los laboratorios de tecnologías en la educación.

Estructura de la Investigación

La investigación se estructuró en 4 capítulos como se indica a continuación:

Capítulo 1: Marco teórico referencial sobre la gestión del mantenimiento del software educativo.

Se realiza la fundamentación teórica. Esto permite un acercamiento al objeto de estudio. Y se realizará un estudio de los antecedentes del software para comprender porque fue necesario realizar un nuevo sistema y no se usó uno que ya existía.

Capítulo 2: Metodologías de desarrollo y herramientas

Se muestran las tecnologías, herramientas y lenguajes que se usarán para el desarrollo de la aplicación.

Capítulo 3: Descripción de la solución

Apoyándose en la metodología de desarrollo de software utilizada, se abordarán los elementos necesarios para describir la solución propuesta.

CAPÍTULO 1: Marco teórico referencial sobre la gestión del mantenimiento del software educativo.

Introducción

En este capitulo se establecen los elementos esenciales que se deben llevar a cabo en un proceso de mantenimiento. Primeramente se analizan las metodologías de desarrollo de software educativo. Posteriormente se exponen los elementos fundamentales sobre edumante y sus acciones para determinar cuál es el flujo de información de esta metodología.

Necesidad de perfeccionamiento del mantenimiento en softwares educativos

Después de un análisis de las metodologías actuales sobre el desarrollo de los softwares educativos (Aguilar-Vera, Aké, & Ucán-Pech, 2015; Assis & Almeida, 2017; Bernardinia et al., 2014; Bradford, 2012; Bul et al., 2015; Choi et al., 2015) se ha detectado que ninguna de ellas contempla el mantenimiento como una de sus etapas. La ausencia de una descripción detallada de cómo realizarlo en las distintas metodologías de desarrollo de software educativo

La determinación de dicho factor estuvo dada por un análisis realizado (por el autor) de las metodologías desarrollo de SE que se identificaron como aportes relevantes en la época en que se publicaron. En dicho análisis se realizó un estudio comparativo sobre aspectos que se consideran importantes para enfatizar la carencia de detalles en las ya mencionadas metodologías de desarrollo de software educativo durante la realización del mantenimiento y sus actividades, para lo cual fueron establecidos un conjunto de criterios. A continuación se presenta y describe la importancia de los criterios propuestos:

Mención del mantenimiento.

Descripción: Permite identificar si la metodología o modelos contemplan el mantenimiento incluyendo los que lo describen a grandes rasgos.

Justificación: Aspecto que potencia la justificación de la necesidad de mantenimiento.

Detalles del mantenimiento

Descripción: Permite identificar si la metodología o modelos describen en detalles todo lo referente al proceso de mantenimiento desde los procesos de apoyo y soporte hasta las actividades y tareas del mismo.

Justificación: Aspecto de interés como referencia y guía para el desarrollo del objetivo de nuestro trabajo.

Incorporación de un mecanismo pedagógico.

Descripción: Permite identificar si la metodología o modelos, se encuentran apoyados en los aspectos pedagógicos del proceso de enseñanza-aprendizaje.

Justificación: Este aspecto es de suma importancia puesto que se pretende el desarrollo de SE, y de acuerdo a las propuestas revisadas es importante no descuidar el aspecto pedagógico de los materiales que se utilizan, así como la forma en que a los alumnos se les brinda esa información.

El mantenimiento de software

El mantenimiento del software es una de las actividades menos atendidas y reconocidas en la industria del software. Sin embargo para Piattini (2004) muchos proyectos europeos fracasan por no tenerlo en cuenta. El estándar IEEE 1219 define el Mantenimiento del Software como "… la modificación de un producto software después de haber sido entregado [a los usuarios o clientes] con el fin de corregir defectos, mejorar el rendimiento u otros atributos, o adaptarlo a un cambio en el entorno" (Epping & Lott, 1994, p. 10).

En el estándar ISO 12207, de Procesos del Ciclo de Vida del Software se establece que "… el Proceso de Mantenimiento contiene las actividades y tareas realizadas por el mantenedor. Este proceso se activa cuando el producto software sufre modificaciones en el código y la documentación asociada, debido a un problema o a la necesidad de mejora o adaptación debido al cambio de los requisitos o de la organización para la cual fue realizado. El objetivo es modificar el producto software existente preservando su integridad. Este proceso incluye la migración y retirada del producto software. El proceso termina con la retirada del producto software" (Jain & Suman, 2015, p. 234). El mantenedor es la organización que proporciona el servicio de mantenimiento. En el caso del SE los requisitos cambiantes del cliente pueden ser de índole institucional, de concepción, de conocimiento, del nivel de los estudiantes; así como de otra índole que pueden llevar a realizar cambios en los sistemas que funcionan sin errores.

El Estándar Internacional ISO/IEC 12207.

La norma ISO/IEC 12207 (ISO/IEC, 1995) es un marco de referencia estándar que cubre todos los aspectos del proceso del ciclo de vida del software. La norma es aplicable a la adquisición de sistemas, productos y servicios software. Describe la arquitectura de los procesos del ciclo de vida software incluyendo el mantenimiento, pero no entra en los detalles de cómo implementar o realizar las actividades y tareas incluidas en tales procesos.

El Proceso de Mantenimiento en ISO-12207.

Como se ha visto, la norma ISO considera el mantenimiento como uno de los procesos principales del ciclo de vida del software. En la definición de ISO, el mantenimiento incluye la migración y la retirada del software y consta de las siguientes actividades:

- Implementación del proceso
- Análisis de problemas y modificaciones
- Implementación de las modificaciones
- Revisión y aceptación del mantenimiento
- Migración
- Retirada de software

Estándares para mantenimiento

En esta sección se analizan el estándar IEEE 1219 y el estándar ISO/IEC 14764. Dichos estándares a diferencia de los anteriormente vistos son específicos para mantenimiento.

El Estándar IEEE 1219 para Mantenimiento

Este estándar IEEE describe "… un proceso iterativo para gestionar y ejecutar actividades de mantenimiento de software [...]. Es aplicable tanto a la planificación del mantenimiento de software que está todavía en desarrollo como a la planificación y ejecución de actividades de mantenimiento de software para productos software existentes" (Pereira dos Santos, 2016, p. 227).

Tal proceso iterativo consta de siete fases para actividades de mantenimiento de software:

1. Identificación y clasificación del problema o la modificación.

2. Análisis.

3. Diseño.

4. Implementación.

5. Pruebas del sistema.

6. Pruebas de aceptación.

7. Entrega.

Descripción del Proceso de Mantenimiento en software educativo propuesto en EDUMANTE.

A continuación se presenta el proceso de mantenimiento propuesto por EDUMANTE, donde primero se muestran los participantes de éste proceso y luego se ofrece una visión general del proceso. Finalmente se presenta la estructura detalla de la metodología.

Participantes en el Proceso de Mantenimiento.

EDUMANTE define a sus participantes en un modelo basado en roles, esto quiere decir, que se definen a un/unos integrante(s) con un conjunto de tareas y responsabilidades para que se desempeñe dentro del proceso de mantenimiento el cual estará supervisado en todo momento por el equipo pedagógico.

Cliente: Es la organización propietaria, por lo tanto, es quien recibe el servicio de mantenimiento. Dicha organización cuenta entre otros con un integrante:

Propietario del Producto: Representa a todos los interesados en el producto final. El propietario del producto por lo general formula peticiones de modificación del tipo perfectivo o adaptativo.

Usuario: Es quien utiliza el software. Propone las peticiones de modificación correctivas (urgentes o no urgentes) y perfectivas.

Equipo de Mantenimiento: Es el grupo de personas que implementa las peticiones de mantenimiento. Tiene autoridad para reorganizarse y definir las acciones necesarias o sugerir remoción de impedimentos. También puede proponer peticiones de mantenimiento preventivo. Este equipo cuenta con:

Gestor de Peticiones: Papel jugado por el ingeniero de software; es quien acepta o rechaza las peticiones de modificación y decide el tipo de mantenimiento que corresponde.

Responsable de Mantenimiento: Es el responsable del mantenimiento, prepara el proceso y establece las normas y procedimientos necesarios para aplicar la metodología. Es quien interactúa con el cliente y el equipo de mantenimiento. Además, es miembro del equipo de mantenimiento y trabaja a la par con el resto de los miembros, coordina los encuentros permanentes del equipo, y se encarga de eliminar eventuales obstáculos.

Equipo pedagógico: encargados de generar (de ser necesario), supervisar y evaluar. Las soluciones de mantenimiento pedagógico derivadas de las peticiones de modificación y por las de cualquier otra índole si repercuten en la estrategia pedagógica y didáctica del producto software, esto mediante los conocimientos de cada uno en los campos que representan cada uno de sus integrantes que son:

Pedagogo: Estructura y clasifica las necesidades del cliente en las diferentes tendencias pedagógicas para establecer las pautas del sistema a desarrollar.

Psicopedagogo: Es el especialista en el desarrollo del aprendizaje del estudiante así como los diferentes niveles de desarrollo de los estudiantes.

Especialista en el área del conocimiento: Es el encargado de mostrar el desarrollo del área del conocimiento que debe mostrarse en el software así como las regularidades de su aprendizaje.

Especialista en tecnología educativa: Traduce las aspiraciones del cliente en un producto desarrollable.

Programador: Encargado de realizar las tareas relacionadas con la modificación del código desencadenadas por alguna petición de modificación.

Editor: Se ocupa de las tareas propias de la edición, conversión de videos a diferentes formatos, captura de filmaciones, conformación de videos.

Diseñador: Revisa los elementos estéticos del producto.

Visión general del Proceso de Mantenimiento.

Precisando el concepto de “proceso de mantenimiento” se indica que está formado por:

1. Un conjunto de actividades destinadas a preparar y planificar las actividades de mantenimiento.
2. El conjunto de intervenciones que producen modificaciones sobre el software.
3. Reuniones para medir el estado de avance y resolver problemas dentro de las intervenciones.
4. Un conjunto de actividades que deben realizarse con posterioridad a cada modificación sobre el software.
5. Opcionalmente:
    a. Tareas que incorporen interfaces con los procesos auxiliares establecidos.
    b. Una actividad que guíe en la finalización de la prestación del servicio de mantenimiento.

El proceso de mantenimiento se apoya en procesos de gestión y soporte con los cuales establece interfaz en determinados momentos. (Ver 2.3 Interfaces con otros procesos). Dicho proceso de mantenimiento comienza cuando el mantenedor y el cliente comienzan a trabajar en conjunto, es decir, asignan responsables, criterios y explicación de cómo se va a trabajar. Estas tareas se agrupan en la actividad llamada Planificación del Proceso. Al concluir la actividad mencionada anteriormente se inicia el ciclo que cada petición de mantenimiento tendrá que seguir. La primera actividad a realizar en el ciclo es atención de la petición mediante la cual se formula ó recibe la petición de mantenimiento, que luego pasa a manos del Gestor de Peticiones para que el mismo ejerza sus funciones. La petición puede entrar a dos diferentes tipos de Mantenimiento, uno corto para las peticiones del tipo no planificable y otro más largo para las del tipo planificable. Dentro del mantenimiento se realizaran una serie de reuniones con el fin de obtener su estado de avance y posibles problemas que puedan ocurrir dentro de su ejecución. Cuando una petición ha sido resuelta se finaliza el ciclo con la actividad finalizar intervención. La finalidad de esta actividad es la validación y verificación del producto por parte del cliente, el paso del producto a producción, registro de documentos y reuniones de retrospección.

Para finalizar el proceso de mantenimiento, se cuenta con una actividad final llamada Finalización del servicio, que sirve para una cesión de actividades por parte del mantenedor de forma que no repercuta negativamente en la organización cliente. En algunas ocasiones, antes de llevar a cabo la finalización es necesario realizar la actividad de Retirada con el fin de aplicar el plan de retirada del software. Las actividades de planificación del proceso, retirada y finalización del servicio, se desarrollarán tan solo una vez y no entorpecerán la agilidad del proceso.

Por otro lado, están los dos tipos de mantenimiento según la planificación que serán un conjunto repetitivo de actividades. Esto no quiere decir que los cambios no estén permitidos, todo lo contrario son bienvenidos y existe un mecanismo para incorporarlos gracias a que existe un feedback rápido con el cliente, junto con una entrega rápida y periódica de atención a las peticiones de mantenimiento.

A continuación para una mejor comprensión del proceso de mantenimiento descrito se muestran las actividades que se desarrollan a lo largo del mismo

Elementos que conforman el mantenimiento en EDUMANTE

Actividades y tareas iniciales comunes

En esta actividad se realizan las primeras tareas que dan origen al inicio del proceso de mantenimieto de EDUMANTE.

Planificación del proceso

El propósito de esta actividad es llevar a cabo la planificación del proceso de mantenimiento. La actividad inicial planificación del proceso solo se ejecuta cuando el cliente contacta con el mantenedor para que realice el proceso de mantenimiento.

Asignar Responsables

Se asignan los roles del proceso de mantenimiento a las personas involucradas en el mismo: El Propietario del Producto, el Gestor de Peticiones, el Responsable de Mantenimiento, el conocedor del tema y el equipo pedagógico, al igual que el resto del equipo de mantenimiento quedan claramente establecidos e identificados.

Definir procedimientos de petición de modificación.

El usuario o el cliente plantea la necesidad de mantenimiento, la cual es presentada a la entidad encargada del mantenimiento por el propietario del producto, en respuesta a lo cual el equipo de mantenimiento genera el documento que se presentará para formalizar la petición. Dicho documento será llamado Petición de Modificación (DOC1) y recibido por el Gestor de Peticiones.

Determinar los motivos que generan la petición de modificación.

Múltiples son los motivos que originan la petición de modificación o mantenimiento en el software educativo:

Errores del sistema: son aquellos que provocan paros, fallos o afectaciones en el software provocando inestabilidad e incoherencia en el mismo.

Motivos de corte pedagógico: son aquellos los cuales conllevan a un mantenimiento por cambios y modificaciones en aspectos como:

-Políticas pedagógicas.
-Contenido y su estructura.
-Sistema de motivación y evaluación.
-Concepciones psicológicas y didácticas.
-Medios y materiales de enseñanza.
-Objetivos tanto de disciplina, programa o de cada actividad.
-Estrategia didáctica.

Motivos de corte Tecnológicos:

Son aquellos los cuales conllevan a un mantenimiento por cambios en el conjunto de teorías y de técnicas que permiten el aprovechamiento práctico del conocimiento científico como:

Infraestructura: por ejemplo cambios en la red interna.

Informáticos: por ejemplo la introducción de técnicas de Inteligencia artificial como es el caso de agentes, redes neuronales y sistemas de razonamiento basados en casos.

Esta actividad se realiza además de para detectar los motivos o causas que generan la petición de mantenimiento también para identificar cuáles de estas involucran cambios específicamente en los aspectos pedagógicos-didácticos del software aunque no se deriven directamente de dichos aspectos. De esta forma se podrá dar un mejor enfoque a la posterior atención de la solución y acciones de mantenimiento.

Diagnóstico de la necesidad de mantenimiento.

La información obtenida en el desarrollo de esta tarea será de utilidad en actividades posteriores. Esta tarea permitirá conocer la situación real. El Gestor de Peticiones decide aceptar o rechazar la petición, para lo cual se apoya en sus conocimientos y los resultados de las actividades descritas a continuación para cada motivo que generó la petición de modificación:

Para toda petición: La realización de encuestas y formularios a los usuarios y/o propietario del producto donde se tratarán los aspectos inherentes a cada cual dependiendo del motivo que genera la petición de modificación con el objetivo de esclarecer el problema o necesidad de modificación.

Para petición generada por motivos de corte pedagógicos-didácticos: Se toman los criterios emitidos por el equipo pedagógico sobre la viabilidad o no de los posibles cambios o modificaciones que pueda ocasionar la petición de modificación a los aspectos pedagógicos-didácticos señalados en la tarea anterior.

Para petición generada por un Error: La confección de una guía de observación del software y la realización de un estudio del manual de usuario, lo que permitirá una mejor comprensión del error y también evaluar la posibilidad de que se esté operando mal o que no se ha comprendido la funcionalidad del software. La realización de un Estudio de Factibilidad (esto en cualquier otro caso) que contemple aspectos como el costo, la designación y consumo de recursos y tiempo.

Adquirir conocimiento de la aplicación.

El Equipo de mantenimiento compila información del software que se va a mantener, para lo cual: a) realiza las actividades propias de cada una de las vertientes ya sea pedagógica o informática como el estudio de la documentación, código fuente, referencias cruzadas, principios pedagógicos didácticos, teoría de aprendizaje, estrategias educativas y didácticas empleadas en el software, observa cómo trabaja éste, b) se entrevista con los usuarios, con el fin de obtener el conocimiento necesario y suficiente del producto software a mantener. Durante el tiempo que dure esta tarea no se realizará ningún tipo de mantenimiento. Al finalizar esta tarea, el Equipo de mantenimiento entrega al Cliente un informe acerca del estado de su software, de manera que el cliente verifique que el Equipo de Mantenimiento ha adquirido un conocimiento adecuado del software.

Preparar entornos de pruebas.

En esta tarea el equipo de mantenimiento prepara el entorno de pruebas o sea las instalaciones, hardware, software, firmware, medios de evaluación pedagógica, procedimientos y documentación necesarios para la cualificación y otras pruebas de software. Se pueden incluir simuladores, analizadores de código, generadores de casos de prueba, analizadores de rutas, además de todos los incluidos en el entorno de Ingeniería del software. También realiza copias del software, prepara las bases de datos y archivos (el entorno), que sean semejantes a la realidad y que cubran la totalidad de las funcionalidades del sistema. El objetivo es que el software pueda funcionar en un ambiente aislado, para que no afecte la operación normal de los sistemas en uso.

Atención de la petición de modificación.

El propósito de ésta actividad es recibir una petición de modificación, a la cual se le asigna un tipo de mantenimiento. En esta actividad comienza el ciclo que cada petición de modificación tendrá que seguir para ser atendida.

Recibir petición de modificación.

El cliente entrega una Petición de modificación (DOC1) que como se ha visto es inducida por cualquiera de los motivos anteriormente definidos y derivados de las necesidades del usuario o el cliente, dicha petición es recibida y registrada por el

Gestor de Peticiones, pero pasa por un proceso de supervisión por parte del equipo pedagógico. El Gestor de Peticiones deberá asignar un identificador único a cada Petición de Modificación.

Decidir el tipo de mantenimiento.

Los diferentes tipos de mantenimiento tienen grados de importancia, criticidad y formas de actuar diferentes. Con el fin de unificar criterios e identificar unívocamente cada situación concreta, hablaremos de los siguientes dos tipos de mantenimiento

Correctivo no urgente: Se produce cuando existe un error en el producto software que no es crítico, pero que tal vez impida el funcionamiento de la aplicación o el normal funcionamiento de la empresa en un periodo de tiempo relativamente corto.

Correctivo urgente, es aquel que se da en situaciones en que existe un error en el producto software que bloquea la aplicación o el proceso de funcionamiento de la empresa, que debe ser resuelto con brevedad. Estas peticiones deben ser rápidamente servidas.

Análisis del error

Investigar y Analizar Causas del error: Esta actividad está orientada a determinar los motivos que generan la petición de modificación, verifica el problema con la colaboración del Usuario que realizó la petición y lo reproduce. Además estudia diferentes alternativas para implementar la modificación para la corrección del error. También se construye una lista de los elementos software a corregir (módulos, rutinas, documentos, entre otras).

Realizar acciones correctivas.

El equipo de mantenimiento ejecuta las acciones necesarias para corregir el problema detectado. Se debe identificar todos los componentes del producto software (rutinas, bases de datos, entre otras) afectadas por la intervención.

Ejecutar pruebas unitarias.

El Equipo de mantenimiento debe comprobar el correcto funcionamiento de todos los cambios realizados. Las pruebas realizadas se deben documentar en el Documento de Pruebas Unitarias Realizadas (DOC2). Esta tarea sirve para comprobar la correcta operación del modulo al que se le han practicado las acciones correctivas.

Mantenimiento No Planificable.

El propósito de esta actividad es brindar atención urgente a las peticiones de modificación que bloquean o interrumpen el funcionamiento del producto software. El mantenimiento no planificable se ejecuta cuando se asume un mantenimiento correctivo urgente. Estas actividades se ejecutan cuando el error presentado en la petición de modificación paraliza de manera seria el funcionamiento normal del software, de forma que la corrección del error deba ser inminente. Se recomienda la ejecución de ciclos cortos de entre uno y siete días (dependiendo del tipo de error) con reuniones todos los días.

Mantenimiento Planificable.

El propósito de esta actividad es brindar atención a las peticiones de modificación que afectan de alguna manera el funcionamiento del producto software. El mantenimiento planificable se ejecuta cuando se asume los mantenimientos: correctivo no urgente, perfectivo, preventivo, adaptativo y evolutivo. Al ser un mantenimiento con menor urgencia que el anterior se recomienda la ejecución ciclos más largos, de entre ocho y quince días (dependiendo del tipo de mantenimiento) con reuniones cada dos días.

Análisis de la petición

Si se trata de un mantenimiento correctivo no urgente o perfectivo (CP), se documenta la causa del error y se indican las posibles alternativas de implementación de la solución en el Documento de Diagnóstico del Error y Posibles Soluciones (DOC3). Si se trata de un mantenimiento preventivo (P) o adaptativo (A), solamente se indican las posibles alternativas de implementación en el Documento de Diagnostico del Error y Posibles Soluciones (DOC3). Luego de analizar cada una de las posibles soluciones de la petición de modificación se elige la alternativa de implementación adecuada, se rellena el Documento de Diagnostico del Error y Posibles Soluciones (DOC3), indicando la alternativa elegida para la corrección

Intervención y pruebas

El equipo de mantenimiento debe ejecutar las acciones necesarias para ofrecer una solución a la petición de modificación conforme con la alternativa seleccionada, la cual está registrada en el documento Diagnostico del Error y Posibles Soluciones (DOC3). Se debe identificar todos los componentes del producto software (rutinas, bases de datos, entre otras) afectadas por la intervención.

Ejecutar pruebas unitarias y de integración

El Equipo de mantenimiento realiza las pruebas unitarias y de integración sobre el producto software intervenido. Se debe comprobar que la Petición de Modificación (registrada en el DOC1) queda atendida y de que los diferentes elementos software funcionan correctamente en forma conjunta. Una vez finalizadas, se genera el Documento de Pruebas Unitarias Realizadas (DOC2). El propósito es probar el correcto funcionamiento del software tanto en módulos independientes como en todo el sistema.

Ejecutar paralelamente el software antiguo y nuevo

El equipo de mantenimiento ejecuta acciones reales en el producto software antiguo y en el modificado para detectar y prevenir posibles errores de proceso. Pueden aplicarse pruebas de no regresión, de manera que no se repitan errores anterior a la intervención.

Seguimiento del Mantenimiento.

El propósito de esta actividad es conducir reuniones de control para hacer un seguimiento del estado de avance y resolver problemas de las intervenciones realizadas mediante el Mantenimiento. En esta actividad se lleva a cabo la revisión del trabajo realizado y se solucionan las dificultades encontradas. Estas actividades son comunes para los dos tipos de mantenimiento (planificable y no planificable).

Reuniones habituales.

En estas reuniones de al menos 15 minutos, se reúne todo el equipo de mantenimiento, en el que cada miembro del equipo expone solo los siguientes temas:

¿Qué es lo que hizo desde la última reunión?

¿Qué es lo que va hacer hasta la siguiente reunión

¿Cómo se va a llevar a cabo, te hace falta algo?

¿Se mantienen los criterios pedagógicos descritos en la última reunión?

Seguimiento de los cambios

Se realiza el control de los cambios producidos en la ejecución del mantenimiento, este control abarca los siguientes aspectos:

- Realizar la traza de los cambios que la petición de modificación ha provocado a lo largo de los procesos de desarrollo implicados.
- Verificar que se han realizado satisfactoriamente las pruebas unitarias, de integración y del sistema que se consideraron necesarias para los componentes a modificar.
- Comprobar que sólo se ha modificado lo establecido y, en caso contrario, justificar el motivo.
- Comprobar si la modificación cumple con los criterios pedagógicos establecidos.
- Llevar el control de los distintos desarrollos existentes en paralelo sobre un mismo componente, con el fin de coordinar las modificaciones incluidas en cada uno de ellos, y asegurar que en el paso a producción se implantan correctamente

Actividades finales.

El propósito de esta actividad es socializar los cambios realizados con el fin de que el cliente verifique la corrección de la Petición de modificación. Este conjunto de actividades y tareas es común para todos los mantenimientos y se denominan finales ya que se ejecutan al terminar el mantenimiento propuesto por una petición de modificación.

Verificar y validar corrección con el cliente.

El Equipo de mantenimiento y la Organización Usuaria del sistema se reúnen para comprobar que el producto intervenido funciona correctamente. En esta tarea se debe haber interacción con el cliente para recabar impresiones, sugerencias, mejoras y su relevancia sobre el producto que se ha entregado. También se evalúan y registran nuevos posibles cambios en el Registro de Peticiones.

Pasar a producción.

El Equipo de mantenimiento pasa al entorno de producción el software corregido para su utilización por parte de los usuarios.

Documentar manual de usuario.

El Equipo de mantenimiento debe documentar o re-documentar el manual de usuario, si es que ha cambiado el modo de operación del software o si ha agregado nuevas funcionalidades.

Registro de la intervención.

La intervención de modificación queda registrada, según los procedimientos de la organización.

Reunión de retrospección.

Se deben extraer las mejores prácticas de la última intervención. Aquí el Encargado de Mantenimiento pregunta a su Equipo: ¿Qué fue bien en el último mantenimiento? ¿Qué se puede mejorar? Toda esta información es tomada para optimizar el próximo mantenimiento. Una labor muy importante a realizar en esta tarea es la definición y anuncio del próximo mantenimiento a ejecutar.

Desarrollar plan de retirada.

El Equipo de mantenimiento redacta un documento en el que describe cuándo se llevará a cabo la retirada del software.

Notificar futura retirada.

El Equipo de mantenimiento notifica al Cliente el momento en el que se ejecutará la retirada del software

Ejecutar paralelo

El Usuario, con el visto bueno del Cliente y bajo la supervisión del Equipo de mantenimiento, realiza operaciones reales sobre el software que se va a retirar y el software nuevo a implantar (si es que aquél va a ser sustituido).

Notificar retirada.

Se notifica la inminencia de la retirada. El producto software es retirado y el nuevo es el que queda en funcionamiento para explotación.

Almacenar Datos del Software Antiguo

Los datos del software antiguo son almacenados.

Finalización del servicio.

El propósito de esta actividad es dar por finalizado de manera formal el servicio de mantenimiento de software al cliente. Esta actividad se realiza cuando el mantenedor deja de prestar sus servicios a la organización cliente. Esta actividad consta de las siguientes tareas.

Entrega del inventario y de la documentación.

Se entregan los productos de software generados y modificados al cliente.

Cesión definitiva del servicio.

La Organización de mantenimiento deja de prestar sus servicios al Cliente con carácter definitivo.

Conclusiones del Capitulo

Se determinaron los conceptos fundamentales del mantenimiento como fase en el proceso de desarrollo de un software educativo así como los procesos que tienen lugar en él.

Se asume la metodología de mantenimiento EDUMANTE por ser la única metodología para el mantenimiento del software educativo y se describen sus fases y procesos lo que posibilita determinar un flujo de procesos asi como la información que se introduce en el sistema y la que se genera.

CAPÍTULO 2: Metodologías de desarrollo y herramientas

Introducción

Para llevar a cabo el desarrollo del sistema se realiza una investigación para escoger un lenguaje de programación y un framework que permita adecuarse a las peticiones del cliente que posea una amplia bibliografía que facilite su uso.

En la actualidad existen distintos tipos de lenguajes de programación y cada uno tiene diferentes entornos de trabajo (Frameworks). En clases estudiamos el lenguaje php (Hypertext Pre-Processor) orientado al desarrollo web de contenido dinámico lo cual resulta una ventaja en el desarrollo de nuestro sistema debido a que se profundizara en el estudio del lenguaje y no se inicia desde cero con su compresión, ahorra tiempo de estudio y que adelanta tiempo de trabajo. Este lenguaje al ser vinculado a trabajo con el servidor permite la conexión con varios gestores de Base de Datos.

Metodología de desarrollo

Una metodología de desarrollo de software indica las acciones a realizar, los documentos que se generan, los artefactos que se utilizan para describir las acciones, los roles de quienes intervienen en el proceso, así como las métricas utilizadas durante el proceso.

Pesadas:

Teniendo en cuenta la filosofía de desarrollo de las metodologías, aquellas con mayor énfasis en la planificación y control del proyecto, en especificación precisa de requisitos y modelado, reciben el apelativo de Metodologías Tradicionales o Pesadas. Entre las metodologías tradicionales o pesadas podemos citar:

RUP (Rational Unified Procces): constituye la metodología estándar más utilizada para el análisis, diseño, implementación y documentación de sistemas orientados a objetos. Es conjunto de metodologías adaptables al contexto y necesidades de cada organización. Incluye información entrelazada de diversos artefactos y descripciones de las diversas actividades (Rabbi & Mannan, 2016).

MSF (Microsoft Solution Framework): Es una metodología personalizable que ayuda a los equipos a enfrentarse directamente a las causas más habituales de fracaso de los proyectos tecnológicos y mejorar así las tasas de éxito, Se centra en: alinear los objetivos de negocio y de tecnología, establecer de manera clara los objetivos, los roles y las responsabilidades, implementar un proceso iterativo controlado por hitos o puntos de control, controlar los riesgos de manera proactiva, responder con eficacia ante los cambios (Rios Herrera, 2016).

Win-Win Spiral Model: Es una adaptación del modelo de espiral que se hace hincapié explícitamente situados en la participación del cliente en un proceso de negociación en la génesis del desarrollo de productos. Idealmente, el desarrollador simplemente preguntar al cliente lo que se requiere y el cliente proporcionaría el suficiente detalle para proceder. Por desgracia esto rara vez sucede y negociaciones significativas entre ambas partes son necesarias para equilibrar la funcionalidad, rendimiento, entre otros... con los costos de salida al mercado (Rios Herrera, 2016).

Iconix: Es una metodología pesada que unifica un conjunto de métodos de orientación a objetos con el objetivo de tener un control estricto sobre todo el ciclo de vida del producto a realizar. Presenta claramente las actividades de cada fase y exhibe una secuencia de pasos que deben ser seguidos: modelo de dominio, modelo de casos de uso, prototipo de interfaz de usuario. Cuenta con tres características fundamentales: iterativo e incremental, trazabilidad y dinámica del UML (Rios Herrera, 2016).

Ágiles:

Relativamente un nuevo paradigma de desarrollo software con múltiples técnicas para llevarlo a la práctica. Entre los más notables se encuentran:

> **Scrum (1986):** Scrum es un marco de trabajo que da libertad en cómo implementarlo. Empezar a usar Scrum es fácil y aplicarlo en el desarrollo de proyectos aporta beneficios no sólo a la empresa que fabrica el producto sino al cliente y al equipo de desarrollo. El impacto que producen los cambios en el desarrollo es menor para la empresa, el cliente puede empezar a usar las partes más valiosas del producto antes empezando así a recuperar parte de la inversión.

Ver y usar parte del producto ayuda al cliente a darse cuenta más rápido de los cambios necesarios, y al ser detectados antes tendrán menor impacto y coste en el desarrollo. El equipo se auto gestiona y cada persona puede crecer profesionalmente adquiriendo conocimiento y aportando valor en múltiples áreas del proyecto. Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales (Armendáriz Barreno & Guaraca, 2013).

Crystal: Crystal no es una metodología en si misma sino una familia de metodologías con un “código genético” común. Crystal da vital importancia a las personas que componen el equipo de un proyecto, y por tanto sus puntos de estudio son: aspecto humano del equipo, tamaño de un equipo (número de componentes), comunicación entre los componentes, distintas políticas a seguir, espacio físico de trabajo (Armendáriz Barreno & Guaraca, 2013).

Programación Extrema (en inglés eXtreme Programming o XP): es una metodología de desarrollo de la ingeniería de software, la programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos (Armendáriz Barreno & Guaraca, 2013).

Método de Desarrollo de Sistemas Dinámicos (Dynamic Systems Development Method o DSDM): Es una metodología ágil que es apoyada por la continua implicación del usuario en un desarrollo iterativo y creciente. Sensible a los requerimientos cambiantes, para desarrollar un sistema que reúna las necesidades de la empresa en tiempo y presupuesto es caracterizada por su rapidez de desarrollo atendiendo a las demandas de tecnología de forma eficaz y eficiente previendo que transcurra mucho tiempo y la tecnología cambie. Es ideal para proyectos de sistemas de información cuyos presupuestos y agendas son muy apretados (Armendáriz Barreno & Guaraca, 2013).

Selección para el desarrollo del trabajo y justificación de la misma

Metodología Ágil

Las metodologías pesadas proponen un marco de trabajo planificado y no cambiable. Están destinadas a grandes proyectos, no resulta factible su utilización para el desarrollo de esta aplicación. Las Metodologías Agiles proponen una forma de trabajo rápida y orientada a proyectos de cualquier tamaño grande, mediano o pequeño. Por lo tanto, la más indicada a este proyecto es la Metodología Ágil.

Scrum:

El proceso en Scrum

En Scrum un proyecto se ejecuta en bloques temporales cortos y fijos (iteraciones que normalmente son de 2 semanas, aunque en algunos equipos son de 3 y hasta 4 semanas, límite máximo de feedback y reflexión). Cada iteración tiene que proporcionar un resultado completo, un incremento de producto final que sea susceptible de ser entregado con el mínimo esfuerzo al cliente cuando lo solicite. El proceso parte de la lista de objetivos/requisitos priorizada del producto, que actúa como plan del proyecto. En esta lista el cliente prioriza los objetivos balanceando el valor que le aportan respecto a su coste y quedan repartidos en iteraciones y entregas.

Las actividades que se llevan a cabo en Scrum son las siguientes:

Planificación de la iteración

El primer día de la iteración se realiza la reunión de planificación de la iteración. Tiene dos partes:

1. **Selección de requisitos** (4 horas máximo). El cliente presenta al equipo la lista de requisitos priorizada del producto o proyecto. El equipo pregunta al cliente las dudas que surgen y selecciona los requisitos más prioritarios que se compromete a completar en la iteración, de manera que puedan ser entregados si el cliente lo solicita.

2. **Planificación de la iteración** (4 horas máximo). El equipo elabora la lista de tareas de la iteración necesarias para desarrollar los requisitos a que se ha comprometido. La estimación de esfuerzo se hace de manera conjunta y los miembros del equipo se autoasignan las tareas.

Ejecución de la iteración

Cada día el equipo realiza una reunión de sincronización (15 minutos máximo). Cada miembro del equipo inspecciona el trabajo que el resto está realizando (dependencias entre tareas, progreso hacia el objetivo de la iteración, obstáculos que pueden impedir este objetivo) para poder hacer las adaptaciones necesarias que permitan cumplir con el compromiso adquirido. En la reunión cada miembro del equipo responde a tres preguntas:

- ¿Qué he hecho desde la última reunión de sincronización?
- ¿Qué voy a hacer a partir de este momento?
- ¿Qué impedimentos tengo o voy a tener?

Durante la iteración el Facilitador (Scrum Master) se encarga de que el equipo pueda cumplir con su compromiso y de que no se merme su productividad.

- Elimina los obstáculos que el equipo no puede resolver por sí mismo.
- Protege al equipo de interrupciones externas que puedan afectar su compromiso o su productividad.

Durante la iteración, el cliente junto con el equipo refinan la lista de requisitos (para prepararlos para las siguientes iteraciones) y, si es necesario, cambian o replanifican los objetivos del proyecto para maximizar la utilidad de lo que se desarrolla y el retorno de inversión.

Inspección y adaptación

El último día de la iteración se realiza la reunión de revisión de la iteración. Tiene dos partes:

1. **Demostración** (4 horas máximo). El equipo presenta al cliente los requisitos completados en la iteración, en forma de incremento de producto preparado para ser entregado con el mínimo esfuerzo. En función de los resultados mostrados y de los cambios que haya habido en el contexto del proyecto, el

cliente realiza las adaptaciones necesarias de manera objetiva, ya desde la primera iteración, replanificando el proyecto.

2. **Retrospectiva** (4 horas máximo). El equipo analiza cómo ha sido su manera de trabajar y cuáles son los problemas que podrían impedirle progresar adecuadamente, mejorando de manera continua su productividad. El Facilitador se encargará de ir eliminando los obstáculos identificados.

Framework

En el desarrollo de software, un entorno de trabajo o framework es una estructura conceptual y tecnológica de asistencia definida, normalmente, con artefactos o módulos concretos de software, que puede servir de base para la organización y desarrollo de software. Típicamente, puede incluir soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Laravel: maneja una sintaxis expresiva, elegante, con el objetivo de eliminar la molestia del desarrollo web facilitando las tareas comunes, como la autenticación, enrutamiento, sesiones y caché. Proporciona, potentes herramientas accesibles necesarias para construir grandes aplicaciones robustas, con un contenedor de controles de inversión, sistema de migración expresiva, y el apoyo de las pruebas unitarias estrechamente integrada

Yii: es un acrónimo de" **Yes, ¡It is!**" o "**Si, ¡Lo es!** ", es un framework de desarrollo de aplicaciones libres para la web, de código abierto escrito en PHP5 que promueve el diseño limpio y motiva el desarrollo rápido. Se trabaja para optimizar su desarrollo de aplicaciones y ayuda a garantizar un producto final extremadamente eficiente y extensible.

CodeIgniter: cuenta con un amplio conjunto de librerías para **tareas comúnmente necesarias**, así como una interfaz sencilla y la estructura lógica para acceder a estas bibliotecas. Este **Framework** usa la arquitectura MVC, que permite una gran separación entre la lógica y la presentación, especialmente útil para los proyectos en los que los diseñadores están trabajando en los archivos de plantilla.

Symfony: Este **Framework PHP** es útil para acelerar la creación y el mantenimiento de sus aplicaciones web. Hace uso de las normas existentes de PHP. Proporciona un conjunto de elementos prefabricados que se pueden integrar rápidamente en su aplicación, combinada con una metodología clara para ayudarle a trabajar de forma eficiente y eficaz en las tareas más complejas.

Phalcon: Está basado en **PHP5**, y se implementa como una **extensión de C** para ofrecer un menor consumo de recursos y alto rendimiento. No hay necesidad de aprender o utilizar el lenguaje C, ya que las funcionalidades se brindan como clases PHP listas para usar. Como **Phalcon** se acopla libremente, se puede usar el Framework completo, o sólo partes específicas del mismo como componentes armables

Zend Framework 2: Es un FrameWork de código abierto para desarrollar aplicaciones web, utilizando el código orientado a objetos. Los componentes de la biblioteca estándar forman una poderosa herramienta extensible cuando se combinan, ofreciendo una aplicación MVC de alto rendimiento y bastante robusta.

Selección para el desarrollo del trabajo y justificación de la misma
Symfony

Rápido y consumo menor de memoria: Un error común en el desarrollo de aplicaciones es la de considerar el rendimiento o velocidad de la aplicación cuando la implementación ha terminado. Symfony fue concebido desde el principio para favorecer el rendimiento. Symfony es aproximadamente tres (3) veces más rápido que la versión 1.4 o de 1.10 de Zend Framework, usando la mitad de consumo de memoria.

Flexibilidad ilimitada: Symfony es adaptable.

• Gestión de la complejidad: Permite desarrollar aplicaciones complejas con múltiples funcionalidades.

• Pieza a pieza: Permite construir de acuerdo con las funciones requeridas.

• Microframework: Symfony se puede utilizar para desarrollar una funcionalidad específica. Sin tener que reconstruir todo y sin necesidad de instalar todo el framework, sólo la pieza concreta requerida.

• Soporte: Proporcionado por Sensio, adicionalmente la comunidad (listas de correo, IRC, entre otras) y las empresas de servicios que han invertido en este framework.

Por último, también es con miras a un desarrollo sostenible que Symfony se distribuye bajo licencia Open Source MIT, que no impone restricciones y permite el desarrollo de código abierto, así como aplicaciones propietarias.

Ampliable: Desde la más pequeña pieza a la base completa en sí, todo lo que se presenta como un "paquete" (o plug-in en el lenguaje Symfony) en Symfony. Cada paquete está destinado para añadir funcionalidad al framework, por supuesto, y cada paquete también puede ser reutilizada en otro proyecto o compartida con el resto de la comunidad. En cualquier caso, el sistema permite todo cambio dentro de Symfony, incluyendo el propio núcleo. Uso de contratos de la interfaz del sistema entre las piezas, el comportamiento del framework así puede ser cambiado a voluntad, sin necesidad de reconfiguración completa.

Estable y sostenible: Desarrollado por los Laboratorios Sensio, las principales versiones de Symfony son soportados por 3 años por la empresa. Para mayor estabilidad, las versiones menores del contrato Symfony y la interfaz también están garantizadas y la compatibilidad entre todas las versiones secundarias se llevará a cabo en el API definido por las interfaces públicas.

La alegría de desarrollo: En un entorno altamente funcional, Symfony también garantiza un cierto nivel de comodidad para los desarrolladores. Al cuidar de una serie de tareas desagradables (desarrollo de funcionalidades de menor importancia, por ejemplo), Symfony permite a los desarrolladores centrarse en los aspectos más destacados reales de una aplicación y para ambos completamente validar su papel y mejorar su productividad. Entre las herramientas de Symfony diseñados para que el vida de un desarrollador mucho más fácil, está la barra de herramientas de depuración web legendario, así como soporte nativo para entornos de desarrollo, páginas de error detallados o incluso de seguridad nativa.

Facilidad de uso: Completamente flexible para satisfacer las necesidades de los profesionales y usuarios avanzados por igual, Symfony es también muy accesible. Abundante documentación, la comunidad y el apoyo profesional, y "incrustados" mejores prácticas dentro del marco (las mejores prácticas que se aplican de forma nativa sin necesidad de ser conscientes de ellos o entenderlos) permiten que un principiante se sienta muy rápidamente a gusto con Symfony.

Características

Symfony fue diseñado para ajustarse a los siguientes requisitos:

- Fácil de instalar y configurar en la mayoría de plataformas (y con la garantía de que funciona correctamente en los sistemas Windows y Unix-like estándares).
- Independiente del sistema gestor de bases de datos. Su capa de abstracción y el uso de ORM (Doctrine 2, Propel), permiten cambiar con facilidad de SGBD en cualquier fase del proyecto.
- Utiliza programación orientada a objetos y características como los espacios de nombres, de ahí que sea imprescindible PHP 5.3.
- Sencillo de usar en la mayoría de casos, aunque es preferible para el desarrollo de grandes aplicaciones Web que para pequeños proyectos.
- Aunque utiliza MVC (Modelo Vista Controlador), tiene su propia forma de trabajo en este punto, con variantes del MVC clásico como la capa de abstracción de base de datos, el controlador frontal y las acciones.
- Basado en la premisa de “convenir en vez de configurar”, en la que el desarrollador sólo debe configurar aquello que no es convencional.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- Preparado para aplicaciones empresariales y adaptable a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.
- Fácil de extender, lo que permite su integración con las bibliotecas de otros fabricantes.
- Una potente línea de comandos que facilitan generación de código, lo cual contribuye a ahorrar tiempo de trabajo.

Características para el desarrollo automatizado de proyectos web

Las características más comunes para el desarrollo de proyectos web están automatizadas en symfony, tales como:

- Permite la internacionalización para la traducción del texto de la interfaz, los datos y el contenido de localización.
- La presentación usa *templates* y *layouts* que pueden ser construidos por diseñadores de HTML que no posean conocimientos del *framework*.
- Los formularios soportan la validación automática, lo cual asegura mejor calidad de los datos en las bases de datos y una mejor experiencia para el usuario.
- El manejo de cache reduce el uso de banda ancha y la carga del servidor.
- La facilidad de soportar autenticación y credenciales facilita la creación de áreas restringidas y manejo de seguridad de los usuarios.
- El enrutamiento y las URLs inteligentes hacen amigable las direcciones de las páginas de la aplicación.
- Las listas son más amigables, ya que permite la paginación, clasificación y filtraje automáticos.
- Los plugins proveen un alto nivel de extensibilidad.
- La interacción con AJAX es mucho más sencilla.

Tendencias tecnológicas a considerar

Arquitectura cliente-servidor: esta arquitectura se divide en dos partes claramente diferenciadas, la primera es la parte del servidor y la segunda la de un conjunto de clientes. Normalmente el servidor es una máquina bastante potente que actúa de depósito de datos y funciona como un Sistema Gestor de Base de Datos (SGBD). Por otro lado, los clientes suelen ser estaciones de trabajo que solicitan varios servicios al servidor. Ambas partes deben estar conectadas entre sí mediante una red (Brox, 2009).

Modelo-Vista-Controlador: Es un patrón de arquitectura de software que separa los datos y la lógica del negocio de una aplicación, de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones. Para ello MVC propone la construcción de tres componentes distintos que son el modelo, la vista y el controlador, es decir, por un lado, define componentes para la representación de la información, y por otro lado para la interacción del usuario.

Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento (Epping & Lott, 1994)

De manera genérica, los componentes de MVC se podrían definir como sigue:

1. **Modelo:** Es la representación de la información con la cual el sistema opera. Gestiona todos los accesos a dicha información, tanto consultas como actualizaciones. Implementa también los privilegios de acceso que se hayan descrito en las especificaciones de la aplicación (lógica de negocio). Envía a la vista aquella parte de la información que en cada momento se le solicita para que sea mostrada (típicamente a un usuario). Las peticiones de acceso o manipulación de información llegan al modelo a través del controlador.
2. **Controlador:** Responde a eventos (usualmente acciones del usuario) e invoca peticiones al modelo cuando se hace alguna solicitud sobre la información (por ejemplo, editar un documento o un registro en una base de datos). Se podría decir que el controlador hace de intermediario entre la vista y el modelo.
3. **Vista:** Presenta el modelo (información y lógica de negocio) en un formato adecuado para interactuar (usualmente la interfaz de usuario) por tanto requiere de dicho modelo la información que debe representar como salida (Epping & Lott, 1994)

IDE

Integrated development environment o entorno de desarrollo integrado es una aplicación de software, que proporciona servicios integrales para facilitarle al programador de computadora el desarrollo de softaware. Normalmente, un IDE consiste de un editor de código fuente, herramientas de construcción automáticas y un depurador. La mayoría de los IDEs tienen auto-completado inteligente de código, algunos contienen un compilador, un iterprete, o ambos, también cuentan con un navegador de clases, un buscador de objetos y un diagrama de jerarquía de clases, para su uso con el desarrollo de software orientado a objetos

NetBeans: Data del año 2010, después de que Sun Microsystems, la lanzara en código abierto. Cuenta con una de las mayores comunidades de desarrolladores que trabajan en un entorno de desarrollo integrado de código abierto. Se puede utilizar con frameworks como Zend, Doctrine, Smarty y Symfony2. Algunas de las características principales son las plantillas de código, el autocompletado inteligente, sugerencias, arreglos rápidos y refactorización. Tiene soporte para idimas como el inglés, el japonés, el ruso, el portugués brasileño y el chno simplificado.

PHPStorm: Desarrollado por JetBrains, PHPStorm es apto y funciona muy bien con todos los frameworks exitentes, este IDE ofrece edición en directo con tecnologías como CSS, HTML5, JavaScript o TypeScript. Grandes marcas com Expedia, Yahoo, Cisco, Salesforce y Wikipedia utilizan PHPStorm como su editor de código.

Sublime Text 3: Este IDE soporta una enorme cantidad de lenguajes y tiene gran capacidad de adaptación a cada uno de ellos. Es ligero, cuenta con multitud de funciones y es compatible con Windows, OSX y Linux. Su principal poder reside en la potencia de sus plugins y paquetes, actualmente cuenta con mucos paquetes de PHP deponibles.

Selección para el desarrollo del trabajo y justificación de la misma
PHP Storm

Depuración y prueba

Depurador visual de fácil configuración para la inspección de variables locales relevantes al contexto y relojes definidos por el usuario, incluyendo matrices y objetos complejos, y edición de valores sobre la marcha. Los scripts se pueden perfilar directamente desde PhpStorm con XDebug o Zend Debugger. Se encuentra disponible un informe agregado, y el usuario puede pasar de las estadísticas de ejecución directamente a la función en código PHP. Las pruebas de PHPUnit pueden desarrollarse en PhpStorm y ejecutarse al instante desde un directorio, archivo o clase utilizando las opciones del menú contextual con cobertura de código.

Funciones de JavaScript, CSS y HTML

Finalización de código para JavaScript, HTML y CSS (para etiquetas, palabras clave, etiquetas, variables, parámetros y funciones).

- Soporte HTML5
- Edición en vivo: los cambios en el código se pueden ver inmediatamente en el navegador sin tener que volver a cargar la página.

- Soporte CSS / SASS / SCSS / LESS (finalización de código, resaltado de error, validación, entre otras).
- Codificación Zen.
- Navegar por código y búsqueda de usos (Ir a declaración / símbolo, Buscar usos).
- ECMAScript Harmony Support.
- Refactorización de JavaScript (Renombrar, Extraer Variable / Función, Variable / Función en línea, Mover / Copiar, Eliminar de forma segura, Extraer script incrustado en un archivo).
- Depurador de JavaScript y pruebas de unidad.
- Soporte de IntelliJ IDEA PHP

Política de licencias y actualizaciones

En noviembre de 2015, JetBrains cambió a la licencia de PhpStorm por suscripción. Las suscripciones anuales o suscripciones mensuales mantenidas durante 12 meses consecutivos también reciben una "licencia de reserva perpetua" para la versión principal disponible en el momento de la compra. PhpStorm está disponible para desarrolladores individuales, empresas y organizaciones, con Licencias adicionales con descuento y complementarias están disponibles para startups, estudiantes y profesores, y proyectos de código abierto no comerciales. Estas licencias complementarias requieren aprobación y no incluyen la cláusula de exclusión perpetua, es decir, caducan.

Servidor Web

Un servidor Web es un programa que utiliza el protocolo de transferencia de hiper texto, HTTP (Hypertext Transfer Protocol), para servir los archivos que forman páginas Web a los usuarios, en respuesta a sus solicitudes, que son reenviados por los clientes HTTP de sus computadoras. Las computadoras y los dispositivos dedicados también pueden denominarse servidores Web

Apache: Este es el más común y más utilizado en todo el mundo. Además, es gratuito (cómo no), y de código abierto, así que podríamos decir que corre sobre cualquier plataforma.

Microsoft IIS: Sólo funciona sobre sistemas Windows, como ya habréis imaginado.

Sun Java System Web Server: Este producto pertenece a la casa Sun, y suele empalarse sobre entorno de este sistema.
Sin embargo, como Apache, es multiplataforma, y recientemente Sun ha decidido distribuirlo con licencias de código abierto (BSD concretamente).

Ngnix: Este es un servidor Web muy ligero y corre sobre sistemas Unix y Windows. Se ha convertido en el 4º servidor HTTP más popular de la red y también se distribuye bajo licencia BSD.

Lighttp: Este servidor Web es otro de los más ligeros que hay en el mercado. Está especialmente pensado para hacer cargas pesadas sin perder balance, utilizando poca RAM y poca de CPU. Algunas páginas populares que lo usan son Youtube, Wikipedia y otras que soportan gran tráfico diariamente. También es gratuito y se distribuye bajo licencia BSD.

Selección para el desarrollo del trabajo y justificación de la misma
Apache:

Costo

Una segunda ventaja relacionada con el diseño de código abierto de Apache es su costo. El servidor web Apache es completamente gratuito y puede ser descargado por cualquier persona en el mundo. Por el contrario, la competencia, como la tecnología de servidor web de Windows Server 2008 puede tener un costo mínimo de US$470, con un precio de más de 1,000 dólares para las instalaciones más avanzadas.

Excepto en los casos en que ha sido una aplicación específicamente diseñada para competir con el servidor de Windows, utilizar el código abierto Apache Web Server crea un ahorro sustancial. Esto es particularmente valioso para las pequeñas empresas que están lanzando nuevos programas de tecnología, y no tienen grandes presupuestos para el servidor.

Funcionalidades

A pesar de su costo mínimo, Apache Web Server tiene un gran conjunto de funcionalidades de gran alcance. Estas características principales, junto con las extensiones creadas por programadores de todo el mundo, ayudan a que la plataforma Apache sea competitiva incluso frente a rivales de alto precio.

Apache ha incorporado en su soporte a una amplia gama de lenguajes de programación web, como Perl, PHP y Python. Estos lenguajes son fáciles de aprender y se pueden utilizar para crear potentes aplicaciones en línea. Apache también incluye soporte "SSL" y "TLS". Estos son los protocolos para enviar datos encriptados a través de Internet, y son importantes en el desarrollo de tiendas seguras en línea y otras aplicaciones que requieren privacidad.

Soporte

Apache Web Server cuenta con una gran comunidad de usuarios de soporte. A diferencia de muchas compañías de software que se encargan de todo el soporte a los programas desde un solo lugar, el soporte técnico de Apache se extiende a lo largo de múltiples localizaciones, empresas, y foros. Este modelo de distribución del soporte permite a los usuarios obtener respuestas a preguntas técnicas casi las 24 horas al día, no importa dónde se encuentren.

Al ser de código abierto, Apache está conectado a muchos usuarios que son capaces de crear parches y correcciones de errores técnicos muy rápidamente. Tan pronto como se encuentra un problema, los usuarios de todo el mundo comunican y aportan soluciones. El resultado de este soporte de la comunidad es software que es muy estable y bien mantenido.

Portabilidad

Apache Web Server es muy portable. Esto significa que se puede instalar en una amplia variedad de servidores y sistemas operativos. Apache es capaz de ejecutarse en todas las versiones del sistema operativo UNIX. Linux es compatible, así como los sistemas operativos Windows NT y MacOS. En comparación, el propio servidor de Microsoft Windows normalmente sólo se puede instalar en sistemas operativos Windows.

Desde un punto de vista de hardware, Apache puede ser utilizado en cualquier servidor con procesador de la serie Intel 80x86 cuando se combina con Windows. Si Apache está siendo utilizado en un sistema operativo Unix o Linux, casi cualquier tipo de procesador es compatible. En general, Apache es uno de los sistemas de servidores más adaptables disponibles en la actualidad, y se ejecutará en una amplia gama de entornos técnicos.

Sistema gestor de base de datos(SGBD)

Es un sistema de software que permite la definición de bases de datos; así como la elección de las estructuras de datos necesarios para el almacenamiento y búsqueda de los datos, ya sea de forma interactiva o a través de un lenguaje de programación. Un SGBD relacional es un modelo de datos que facilita a los usuarios describir los datos que serán almacenados en la base de datos junto con un grupo de operaciones para manejar los datos.

PostgreSQL: Brinda un control de concurrencia multi-versión (MVCC por sus siglas en inglés) que permite trabajar con grandes volúmenes de datos; soporta gran parte de la sintaxis SQL y cuenta con un extenso grupo de enlaces con lenguajes de programación. Posee características significativas del motor de datos, entre las que se pueden incluir las subconsultas, los valores por defecto, las restricciones a valores en los campos (constraints) y los disparadores (triggers). Ofrece funcionalidades en línea con el estándar SQL92, incluyendo claves primarias, identificadores entrecomillados, conversión de tipos y entrada de enteros binarios y hexadecimales.

MySQL: Es rápido, sólido y flexible. Es idóneo para la creación de bases de datos con acceso desde páginas web dinámicas, así como para la creación de cualquier otra solución que implique el almacenamiento de datos, posibilitando realizar múltiples y rápidas consultas. Está desarrollado en C y C++, facilitando su integración en otras aplicaciones desarrolladas también en esos lenguajes.

Es un sistema cliente/servidor, por lo que permite trabajar como servidor multiusuario y de subprocesamiento múltiple, o sea, cada vez que se crea una conexión con el servidor, el programa servidor establece un proceso para manejar la solicitud del cliente, controlando así el acceso simultáneo de un gran número de usuarios a los datos y asegurando el acceso a usuarios autorizados solamente.

Microsoft SQL Server: Es un sistema gestor de base de datos relacionales producido por Microsoft. Es un sistema cliente/servidor que funciona como una extensión natural del sistema operativo Windows. Entre otras características proporciona integridad de datos, optimización de consultas, control de concurrencia y backup y recuperación.

Es relativamente fácil de administrar a través de la utilización de un entorno gráfico para casi todas las tareas de sistema y administración de bases de datos. Utiliza servicios del sistema operativo Windows para ofrecer nuevas capacidades o ampliar la base de datos, tales como enviar y recibir mensajes y gestionar la seguridad de la conexión. Es fácil de usar y proporciona funciones de almacenamiento de datos que sólo estaban disponibles en Oracle y otros sistemas gestores de bases de datos.

Conclusiones del capítulo

Se determina **Scrum** como metodología de desarrollo debido a las características que tiene el proyecto y el equipo de desarrollo lo cual ya determina los artefactos y procesos a realizar en la ingeniería.

Se asume como framework de desarrollo a **symphony** por las facilidades de desarrollo que provee y la amplia comunidad de desarrolladores que brindan información accesible.

Las herramientas utilizadas son **PHP**, **MySql**, **Apache** por la simplicidad e las soluciones que proponen a tono con el sistema a desarrollar además de una amplia comunidad de desarrolladores a consultar para las problematizas asociadas con su utilización durante el proyecto.

CAPÍTULO 3: Descripción de la solución

En este capítulo se explicitan los artefactos y de realizan las acciones previstas en la metodología asumida con anterioridad. Para ello se explicitan cada uno de los spring realizados, así como la pila del producto, cuestiones esenciales de scrum.

Descripción de los artefactos de scrum

Pila del producto:

La pila de producto es, básicamente, una lista priorizada de requisitos, o historias, o funcionalidades. Cosas que el cliente quiere, descritas usando la terminología del cliente a esto *historias*, o a veces simplemente *elementos de la pila.*

Las historias incluyen los siguientes campos:

ID – un identificador único, simplemente un número auto-incremental.

Esto nos permite no perder la pista a las historias cuando cambiamos su nombre.

Nombre – una descripción corta de la historia. Por ejemplo, “Ver tu historial de transacciones”. Suficientemente claro como para que el Dueño de Producto comprenda aproximadamente cada funcionalidad, y suficientemente clara como para distinguirla de las otras historias. Normalmente, 2 a 10 palabras.

Importancia – la ratio de importancia que el Dueño de Producto da a esta historia. Por ejemplo, 10. O 150. Más alto = más importante. Suelo evitar el término “prioridad” porque típicamente “1” se considera la “máxima prioridad, lo que es muy incómodo si posteriormente decides que algo es más importante. ¿Qué prioridad le daríamos a ese nuevo elemento? ¿Prioridad 0? ¿Prioridad -1?

Como probarlo –Se trata, esencialmente, de una simple especificación de un test.

Pila del producto

ID	Nombre	Import.	Estim.	Como Probarlo	Notas
1	Diseñar y crear la Base de Datos	20	3	Se verifica la inserción de datos en la base de datos y las validaciones de la misma	Realizada a partir de un diagrama de entidades
2	Autentificarse	28	3	Verificar que el usuario ingresado accedió al sistema	Si el usuario no está registrado no va a ser posible la entrada al sistema.
3	Gestionar usuarios	25	4	Verificar si puede acceder al sistema	Solo el administrador puede gestionar los usuarios y asignar roles (Administrador, Cliente, Gestor)
4	Generar trazas	20	3	Verificar si se registra cada acción que realiza el usuario con su fecha correspondiente	Solo el administrador del sistema puede consultar las trazas
5	Gestionar softwares	20	4	Verificar que se insertó el manual de usuario correspondiente al software.	Solo el cliente registrado en el sistema puede gestionar la información del software

6	Gestionar peticiones	28	5	Verificar si en la lista de peticiones se encuentra registrada , ver las fechas	Se registra toda la información correspondiente a la petición de modificación que realiza el cliente, por ejemplo: motivo, solicitante, fecha creación y modificación. Es realizado por el cliente
7	Aprobar petición de mantenimiento	28	1	Si se encuentra marcado la opción de aprobado, entonces se considera que la petición fue aprobada para aplicarle el mantenimiento	Si la petición no es aprobada, no se muestra en otro lugar del sistema queda almacenada como no aprobada. Solo el gestor puede aprobar las peticiones

8	Gestionar atributos	25	4	Verificar si en la lista de atributos se encuentra alguno	Estos atributos son los que se van a controlar de cada tipo de mantenimiento. Por ejemplo: del mantenimiento CORRECTIVO voy a controlar los atributos: nombre, observación, breve descripción entre otros
9	Gestionar tipos de mantenimiento	25	4	Verificar si en la lista de los tipos de mantenimientos se encuentra alguno	Son los tipos de mantenimiento que se van a aplicar en el proceso de mantenimiento, solo el gestor puede adicionar nuevos tipos, modificar o eliminarlos.
10	Gestionar asignar atributos que corresponden a cada mantenimiento	28	3	Verificar si en la lista se encuentran los atributos asigados a un tipo de mantenimiento	Si no se ha insertado algún atributo o tipo de mantenimiento no va a ser posible la asignación ya que no existe ninguno, solo el gestor puede realizar esta acción

11	Gestionar causas	25	3	Verificar si en la lista de las causas se encuentra alguna	Son las causas que generan el problema en el software, solo el gestor puede realizar esta acción
12	Gestionar asignar causa que corresponden a cada aplicación de mantenimiento	28	3	Verificar si en la lista de Aplicar mantenimiento - Causa se encuentra alguna causa asignada a uno de ellos	Cuando aplicas un mantenimiento es por una o varias causas, aquí asigno las causas que generaron la aplicación del mantenimiento, por ejemplo: se aplicó el mantenimiento correctivo urgente por un error crítico y por error en la portada del software
13	Gestionar aplicar mantenimiento	25	4	Verificar si en la lista de aplicar mantenimiento se encuentra alguna aplicación verificar las fechas	El nombre es el que se le da a modo de identificar la aplicación del mantenimiento, por ejemplo: ¨mant software marino escuela José martí¨ se registra el encargado, solo el gestor puede realizar esta acción

14	Gestionar asignar petición que se le aplica el mantenimiento	28	3	Verificar si en la lista de Aplicar mantenimiento – Petición se encuentra alguna petición asignada a uno de ellos	El mantenimiento se aplica a una o varias peticiones, el gestor es el único que puede gestionar esta información
15	Gestionar categorías	25	4	Verificar si en la lista categorías se encuentra alguna	Son los tipos de prueba que se realizarán, ej. Prueba de integración, caja negra, caja blanca
16	Gestionar resultados	20	3	Verificar si en la lista resultados se encuentra alguno	Son los diferentes clasificaciones que le les dan a las pruebas, ej. Satisfactorio, Bueno, Malo, Regular
17	Gestionar aplicar pruebas	25	4	Verificar si en la lista de Pruebas se encuentra algún registro	El nombre es el que se le da a modo de identificar la prueba, por ejemplo: ¨prueb software marino 1¨ se registra el encargado, solo el gestor puede realizar esta acción
18	Gestionar asignar petición que se le aplica la prueba	28	3	Verificar si en la lista de Prueba – Petición se encuentra alguna petición asignada	Las pruebas se aplican a una o varias peticiones, el gestor es el único que puede

					gestionar esta información
19	Generar reportes	28	4	Verificar que se exporte como PDF la información solicitada	es posible exportar la información, imprimirla, mandarla por correo, entre otras opciones

Estimación del software utilizando puntos de función

EI: 15x6=90

EO: 17*7=119

EQ: 17*4=56

ILF: 16*5=80

EIF: 0*5=0

PFSA = PFTe + PFTo + PFTq + PFTif + PFTef

PFSA = 90+119+56+80+0

PFSA =345

PFA=345*[0.65+[0.01*ACT]]

FCT: Factor de Complejidad Técnica

1. Comunicación de datos 2
2. Proceso Distribuido de Datos 3
3. Desempeño 2
4. Configuración 3
5. Volumen de Transacciones 4
6. Captura de Datos en Línea 2
7. Eficiencia al usuario final 3

8. actualización de Datos en Línea	4
9. Complejidad	3
10. Reusabilidad	3
11. Facilidad de Instalación	4
12. Facilidad de operación	3
13. Instalación Múltiple	2
14. Facilidad de cambio	2
Total	40

Puntos de Función Ajustados (PFA)

PFA= PFSA*[0,65+ (0,01* ACT)]

PFA=345* [0,65+ (0,01* 40]

PFA= 345*(0,65+0,4)

PFA= 345*1,05

PFA=362,25

Líneas de código (LC)

LC= PFA *(Líneas * PF)

LC=362,25*20

LC= 7,245

Esfuerzo hora/persona

E=PFA/ (1/8 persona/hora)

1 persona trabaja 8h

E= 2898 horas/persona

Tomando 24 días laborables en el mes y 8 horas productivas al día, obtenemos 192 horas laborables al mes.

Duración del proyecto en meses

2898 horas/persona/1 persona = 889,2 horas

DM=2898 horas /192 horas/mes

DM= 15,09 aproximadamente 15 meses y 1 semana por el margen de error de la estimación por puntos de función y 1mes para la fase de prueba

Costo total del proyecto

CT= sueldo de 1 persona/ cant de personas*DM

CT= 400*1*16

CT=6400

Sprint

El Sprint es una carrera con una fecha de inicio y una fecha de entrega y un tiempo de trabajo de una a cuatro semanas. Se muestra a continuación uno de los sprint más importantes del software.

No.13	Nombre: Gestionar aplicar mantenimiento.		
Encargado: Mario Sergio Santana Rodríguez			
Fecha Inicial	Fecha Final	Tiempo de Trabajo	Tiempo de Descanso
16/11/2017	18/11/2017	10:30-5:00	00:10
Descripción: Se Implementa la gestión de la aplicación del mantenimiento, cuando se aplica el mantenimiento se almacena una observación, fecha de creación y modificación, entre otros. Se realizarán las siguientes acciones: -Insertar, Buscar, Mostrar, Cancelar, Eliminar, Editar			

Plan de pruebas

Al desarrollar el plan de pruebas, se puede obtener información sobre los errores, defectos o fallas que tiene el prototipo, así se realizan las correcciones pertinentes, según el caso y se asegura la calidad del producto que se está entregando al cliente.

ID	Nombre	Pruebas a realizar
1	Diseñar y crear la base de datos	• Test base de datos.
2	Autenticarse.	• Test de autenticarse.
3	Gestionar usuarios.	• Test de listar usuario. • Test de insertar usuario. • Test de editar usuario. • Test de eliminar usuario.
4	Generar trazas.	• Test de listar traza. • Test de buscar traza.
5	Gestionar softwares.	• Test de listar los softwares. • Test de insertar los softwares. • Test de editar los softwares. • Test de eliminar los softwares.
6	Gestionar peticiones.	• Test de listar las peticiones. • Test de insertar las peticiones. • Test de editar las peticiones. • Test de eliminar las peticiones.
7	Aprobar petición de mantenimiento	• Test de aprobar petición.
8	Gestionar atributos.	• Test de listar los atributos. • Test de insertar los atributos. • Test de editar los atributos. • Test de eliminar los atributos.
9	Gestionar tipos de mantenimiento	• Test de listar los tipos de mantenimiento.

		• Test de insertar los tipos de mantenimiento. • Test de editar los tipos de mantenimiento. • Test de eliminar los tipos de mantenimiento. • Test de buscar los tipos de mantenimiento.
10	Gestionar asignar atributos que corresponden a cada mantenimiento.	• Test de listar los atributos que corresponden a cada mantenimiento. • Test de insertar los atributos que corresponden a cada mantenimiento. • Test de editar los atributos que corresponden a cada mantenimiento. • Test de eliminar los atributos que corresponden a cada mantenimiento. • Test de buscar los atributos que corresponden a cada mantenimiento.
11	Gestionar causas.	• Test de listar las causas. • Test de insertar las causas. • Test de editar las causas. • Test de eliminar las causas.

12	Gestionar asignar causas que corresponden a cada aplicación de mantenimiento	• Test de listar las causas que corresponden a cada aplicación de mantenimiento. • Test de insertar las causas que corresponden a cada aplicación de mantenimiento. • Test de editar las causas que corresponden a cada aplicación de mantenimiento. • Test de eliminar las causas que corresponden a cada aplicación de mantenimiento. • Test de buscar las causas que corresponden a cada aplicación de mantenimiento.
13	Gestionar aplicar mantenimiento.	• Test de listar la aplicación del mantenimiento. • Test de insertar la aplicación del mantenimiento. • Test de editar la aplicación del mantenimiento. • Test de eliminar la aplicación del mantenimiento.
14	Gestionar asignar petición que se le aplica el mantenimiento.	• Test de listar las peticiones a las que se le aplica el mantenimiento. • Test de insertar las peticiones a las que se le aplica el mantenimiento.

		• Test de editar las peticiones a las que se le aplica el mantenimiento. • Test de eliminar las peticiones a las que se le aplica el mantenimiento.
15	Gestionar categorías	• Test de listar las categorías. • Test de insertar las categorías. • Test de editar las categorías. • Test de eliminar las categorías.
16	Gestionar resultados.	• Test de listar los resultados. • Test de insertar los resultados. • Test de editar los resultados. • Test de eliminar los resultados.
17	Gestionar aplicar pruebas	• Test de listar la aplicación de las pruebas. • Test de insertar la aplicación de las pruebas. • Test de editar la aplicación de las pruebas. • Test de eliminar la aplicación de las pruebas.
18	Gestionar petición que se le aplica la prueba	• Test de listar la petición que se le aplica la prueba. • Test de insertar la petición que se le aplica la prueba. • Test de editar la petición que se le aplica la prueba. • Test de eliminar la petición que se le aplica la prueba.

19	Generar reportes.	• Test de generar los reportes.

Pruebas de aceptación

Las Pruebas de Aceptación (PA) son las realizadas por el cliente y usuarios finales de la aplicación. En estas serán probadas las funcionalidades definidas por el cliente y descritas en las historias de usuario, además de los aspectos de seguridad requeridos. Luego de haber superado las pruebas de aceptación podrá considerarse que la aplicación es apta para el uso. A continuación, se muestran algunas de estas pruebas:

Prueba de aceptación al módulo "Gestionar aplicar mantenimiento"

Pruebas de Aceptación
ID: 13
Responsable: Mario Santana Rodríguez
Nombre de caso de prueba: Test de listar la aplicación del mantenimiento
Descripción: Verificar que se permita listar todas las categorías de los riesgos
Condiciones de ejecución Los datos de la aplicación del mantenimiento solo pueden ser creados, modificados o eliminados, por los usuarios que tenga como rol: Gestor y Superadmin, estos son los únicos que tienen permisos y accesos para esta funcionalidad en el sistema. Estar conectado a la Base de Datos.
Entrada/Pasos de ejecución: Acceder a la vista donde se muestra el listado de todas las categorías de los riesgos.
Resultado esperado: El sistema debe mostrar en una tabla todas las categorías de los riesgos en la base de datos.
Evaluación de la prueba: Satisfactoria.

Prueba de aceptación al módulo "Gestionar aplicar mantenimiento"

Pruebas de Aceptación
ID: 13
Responsable: Mario Santana Rodríguez
Nombre de caso de prueba: Test de insertar la aplicación del mantenimiento
Descripción: Se insertan los datos necesarios para la aplicación del mantenimiento. Se insertarán de forma incorrecta los datos, dejando campos en blanco, se tendrá en cuenta los datos no existan en el sistema. Luego se insertarán de manera correcta para comprobar que los datos sean almacenados. Se modifican los datos de la aplicación del mantenimiento. Se modificarán de forma incorrecta, dejando campos en blanco para verificar la validación, luego se modificarán de manera correcta para comprobar que los datos sean almacenados y cargados. Se eliminará la aplicación del mantenimiento aceptando el mensaje de certeza de la acción a realizar.
Condiciones de ejecución: Los datos de los resultados solo pueden ser creados, modificados o eliminados, por los usuarios que tenga como rol: Gestor y Superadmin, estos son los únicos que tienen permisos y accesos para esta funcionalidad en el sistema. Estar conectado a la base de datos
Entrada/Pasos de ejecución: Dejar campos en blanco. Insertar los datos correctamente. Verificar que se muestren los nuevos datos. Insertar datos de manera incorrecta.
Resultado esperado: El sistema debe alertar al usuario cuando se inserten datos erróneos. Cuando se inserten los datos correctamente, el sistema debe almacenarlos en la base de datos y mostrarlos. Cuando se modifiquen los datos correctamente, el sistema debe almacenarlos en la base de datos y mostrar los nuevos.
Evaluación de la prueba: Satisfactoria.

Prueba de aceptación al módulo "Gestionar aplicar mantenimiento"

Pruebas de Aceptación
ID: 13
Responsable: Mario Santana Rodríguez
Nombre de caso de prueba: Test de editar la aplicación del mantenimiento
Descripción: Verificar que se permita editar todos los datos de la aplicación del mantenimiento. Se modifican correctamente los datos para comprobar que los datos son almacenados en la base de datos. También se modificarán con datos incorrectos y campos en blancos teniendo en cuenta que no se almacenarán en la base de datos.
Condiciones de ejecución: Los datos de los resultados solo pueden ser creados, modificados o eliminados, por los usuarios que tenga como rol: Gestor y Superadmin, estos son los únicos que tienen permisos y accesos para esta funcionalidad en el sistema. Estar conectado a la base de datos
Entrada/Pasos de ejecución: Modificar los datos dejando campos en blanco. Modificar los datos de forma correcta. Verificar que se muestren los nuevos datos. Modificar datos de manera incorrecta.
Resultado esperado: El sistema debe alertar al usuario cuando se editen datos erróneos. Cuando se modifiquen los datos correctamente, el sistema debe almacenarlos en la base de datos y mostrar los nuevos.
Evaluación de la prueba: Satisfactoria.

Prueba de aceptación al módulo "Gestionar aplicar mantenimiento"

Pruebas de Aceptación
ID: 13
Responsable: Mario Santana Rodríguez
Nombre de caso de prueba: Test de eliminar la aplicación del mantenimiento

Descripción: Verificar que se permita eliminar de la tabla el registro deseado
Condiciones de ejecución: Los datos de los resultados solo pueden ser creados, modificados o eliminados, por los usuarios que tenga como rol: Gestor y Superadmin, estos son los únicos que tienen permisos y accesos para esta funcionalidad en el sistema. Estar conectado a la base de datos
Entrada/Pasos de ejecución: Se eliminará la aplicación del mantenimiento dando aceptar en el mensaje que se muestra.
Resultado esperado: Se elimina la aplicación del mantenimiento del sistema.
Evaluación de la prueba: Satisfactoria.

Pruebas Funcionales

Insertar la aplicación del mantenimiento

Atributos:

- nombre
- observación

1-Clases de equivalencia por cada atributo:

Atributo	Válida	Representante	Inválida	Representante
nombre	1-Cualquier combinación de letras y número que tenga al menos 3 caracteres.	Pedagógico	2-Que este vacía	NULL
			3- Cadena que contenga menos de 3 caracteres	as
			5-Cadena que contenga	+`+{¨¨

			caracteres extraños.	
observación	6- Cualquier combinación de letras y números de al menos 10 caracteres.	El grado de incertidumbre de la preparación de los profesores para trabajar con el software, el nivel de desarrollo de los estudiantes, el nivel educacional y la institución donde se trabajará con el software ya terminado y el nivel de desarrollo de la tecnología que allí posean	7-Que este vacía	NULL
			8-Cadena que contenga menos de 10caracteres	El grado

2-Definir casos de Pruebas:

No.	Clase Equiv.	nombre	observación	Result. Esperado
1	1,6	Pedagógico	El grado de incertidumbre de la preparación de los profesores para trabajar con el software, el nivel de desarrollo de los estudiantes, el nivel educacional y la institución donde se trabajará con el software ya terminado y el nivel de desarrollo de la tecnología que allí posean	Resultado esperado
2	1,7	Pedagógico	NULL	Observación no puede estar vacía.
3	1,8	Pedagógico	El grado	Observación es muy breve, tiene que contener al menos 10 caracteres.

4	2,6	NULL	El grado de incertidumbre de la preparación de los profesores para trabajar con el software, el nivel de desarrollo de los estudiantes, el nivel educacional y la institución donde se trabajará con el software ya terminado y el nivel de desarrollo de la tecnología que allí posean	El nombre no puede estar vacío.
5	3,6	as	El grado de incertidumbre de la preparación de los profesores para trabajar con el software, el nivel de desarrollo de los estudiantes, el nivel educacional y la institución donde se trabajará con el software ya terminado y el nivel de desarrollo de la tecnología que allí posean	Nombre está muy corto, debe tener al menos 3 caracteres.

6	5,6	+`+{¨¨	El grado de incertidumbre de la preparación de los profesores para trabajar con el software, el nivel de desarrollo de los estudiantes, el nivel educacional y la institución donde se trabajará con el software ya terminado y el nivel de desarrollo de la tecnología que allí posean	El nombre no puede contener caracteres extraños ni números.

3-Tabla de Prueba por Caso de Prueba:

No.	2
Requerimiento	Insertar aplicación del mantenimiento
Objetivo	Probar la acción de insertar la aplicación del mantenimiento (clases equivalentes 1,7)
Tipo de Prueba	Funcional
Hardware	Un procesador Core i5 , disco duro de 750GB, memoria RAM de 6GB
Software	Sistema Operativo Windows 7 o superior, Base de Datos Xampp, Navegador Google Chrome 60.0.3 o superior
Personal	Ingeniero de Pruebas
Casos de Prueba	

Datos de Entrada	Nombre: Pedagógico Observación: NULL		
Resultados Esperados	Mensaje: ¨Rellene este campo ¨		
Resultados Obtenidos	Si(x) No()		
Casos de Excepción:		**Comentarios:**	
Aprobado por: Mario Santana Rodríguez	**Cargo:** Líder de pruebas	**Fecha:**	

No.	1
Requerimiento	Insertar aplicación del mantenimiento
Objetivo	Probar la acción de insertar la aplicación del mantenimiento (clases equivalentes 1,6)
Tipo de Prueba	Funcional
Hardware	Un procesador Core i5 , disco duro de 750GB, memoria RAM de 6GB
Software	Sistema Operativo Windows 7 o superior, Base de Datos Xampp, Navegador Google Chrome 60.0.3 o superior
Personal	Ingeniero de Pruebas
Casos de Prueba	
Datos de Entrada	Nombre: Pedagógico Observación: El grado de incertidumbre de la preparación de los profesores para trabajar con el software, el nivel de desarrollo de los estudiantes, el nivel educacional y la

	institución donde se trabajará con el software ya terminado y el nivel de desarrollo de la tecnología que allí posean.	
Resultados Esperados	Mensaje: ¨Se agregó con éxito la aplicación del mantenimiento:Pedagógico ¨	
Resultados Obtenidos	Si(x) No()	
Casos de Excepción:	**Comentarios:**	
Aprobado por: Mario Santana Rodríguez	**Cargo:** Líder de pruebas	**Fecha:**

No.	4
Requerimiento	Insertar aplicación del mantenimiento
Objetivo	Probar la acción de insertar la aplicación del mantenimiento (clases equivalentes 2,6)
Tipo de Prueba	Funcional
Hardware	Un procesador Core i5 , disco duro de 750GB, memoria RAM de 6GB
Software	Sistema Operativo Windows 7 o superior, Base de Datos Xampp, Navegador Google Chrome 60.0.3 o superior
Personal	Ingeniero de Pruebas
Casos de Prueba	
Datos de Entrada	Nombre: NULL Observación: El grado de incertidumbre de la preparación de los profesores para trabajar con el software, el nivel de desarrollo de los estudiantes, el nivel educacional y la

	institución donde se trabajará con el software ya terminado y el nivel de desarrollo de la tecnología que allí posean.
Resultados Esperados	Mensaje: ¨Rellene este campo ¨
Resultados Obtenidos	Si(x) No()

Casos de Excepción:	**Comentarios:**

Aprobado por: Mario Santana Rodríguez	**Cargo:** Líder de pruebas	**Fecha:**

Editar la aplicación del mantenimiento

Atributos:

- nombre
- observación

1-Clases de equivalencia por cada atributo:

Atributo	Válida	Representante	Inválida	Representante
nombre	1-Que se encuentre en la base de datos.	Pedagógico	2-Que no esté en la base de datos	NULL
observación	3- Que se encuentre en la base de datos.	El grado de incertidumbre de la preparación de los profesores para trabajar con el software, el nivel de	4- Que no esté en la base de datos	NULL.

		desarrollo de los estudiantes, el nivel educacional y la institución donde se trabajará con el software ya terminado y el nivel de desarrollo de la tecnología que allí posean		

2-Definir casos de Pruebas:

No.	Clase Equiv.	Nombre	observación	Result. Esperado
1	1,3	Pedagógico	El grado de incertidumbre de la preparación de los profesores para trabajar con el software, el nivel de desarrollo de los estudiantes, el nivel educacional y la institución donde se trabajará con el software ya terminado y el nivel de desarrollo	Resultado esperado

			de la tecnología que allí posean	
2	1,4	Pedagógico	NULL	Observación no se encuentra en la base de datos.
3	2,3	NULL	El grado de incertidumbre de la preparación de los profesores para trabajar con el software, el nivel de desarrollo de los estudiantes, el nivel educacional y la institución donde se trabajará con el software ya terminado y el nivel de desarrollo de la tecnología que allí posean	El nombre no se encuentra en la base de datos..
4	2,4	NULL	NULL	El nombre y la observación no se encuentra en la base de datos. .

3-Tabla de Prueba por Caso de Prueba:

No.	1		
Requerimiento	Editar la aplicación del mantenimiento		
Objetivo	Probar la acción de editar la aplicación del mantenimiento (clases equivalentes 1,3)		
Tipo de Prueba	Funcional		
Hardware	Un procesador Core i5 , disco duro de 750GB, memoria RAM de 6GB		
Software	Sistema Operativo Windows 7 o superior, Base de Datos Xampp, Navegador Google Chrome 60.0.3 o superior		
Personal	Ingeniero de Pruebas		
Casos de Prueba			
Datos de Entrada	Nombre: Pedagógico Observación: El grado de incertidumbre de la preparación de los profesores para trabajar con el software, el nivel de desarrollo de los estudiantes, el nivel educacional y la institución donde se trabajará con el software ya terminado y el nivel de desarrollo de la tecnología que allí posean		
Resultados Esperados	Mensaje: ¨Se editó con éxito la aplicación del mantenimiento: Pedagógico ¨		
Resultados Obtenidos	Si(x) No()		
Casos de Excepción:		**Comentarios:**	
Aprobado por: Mario Santana Rodríguez	**Cargo:** Líder de pruebas		**Fecha:**

Eliminar la aplicación del mantenimiento

Atributos:

-Nombre

1- Clases de equivalencia por cada atributo:

Atributo	**Válida**	**Representante**	**Inválida**	**Representante**
Nombre	1-Que la selección exista en la Base de Datos	Costo	2-No seleccionar ninguno	NULL

2- Definir casos de Pruebas:

No.	Clase Equiv.	Nombre	Result. Esperado
1	1	Costo	Operación exitosa
2	2	Null	No se seleccionó ningún nombre

3- Tabla de Prueba por Caso de Prueba:

No.	1
Requerimiento	Eliminar la aplicación del mantenimiento
Objetivo	Probar la acción de eliminar la aplicación del mantenimiento(clases equivalentes 1)
Tipo de Prueba	Funcional
Hardware	Un procesador Core i5 , disco duro de 750GB, memoria RAM de 6GB
Software	Sistema Operativo Windows 7 o superior, Base de Datos Xampp, Navegador Google Chrome 60.0.3 o superior
Personal	Ingeniero de Pruebas
Casos de Prueba	
Datos de Entrada	Nombre: Coste
Resultados Esperados	Mensaje: ¨Se eliminó con éxito la aplicación del mantenimiento: Coste ¨
Resultados Obtenidos	Si(x) No()

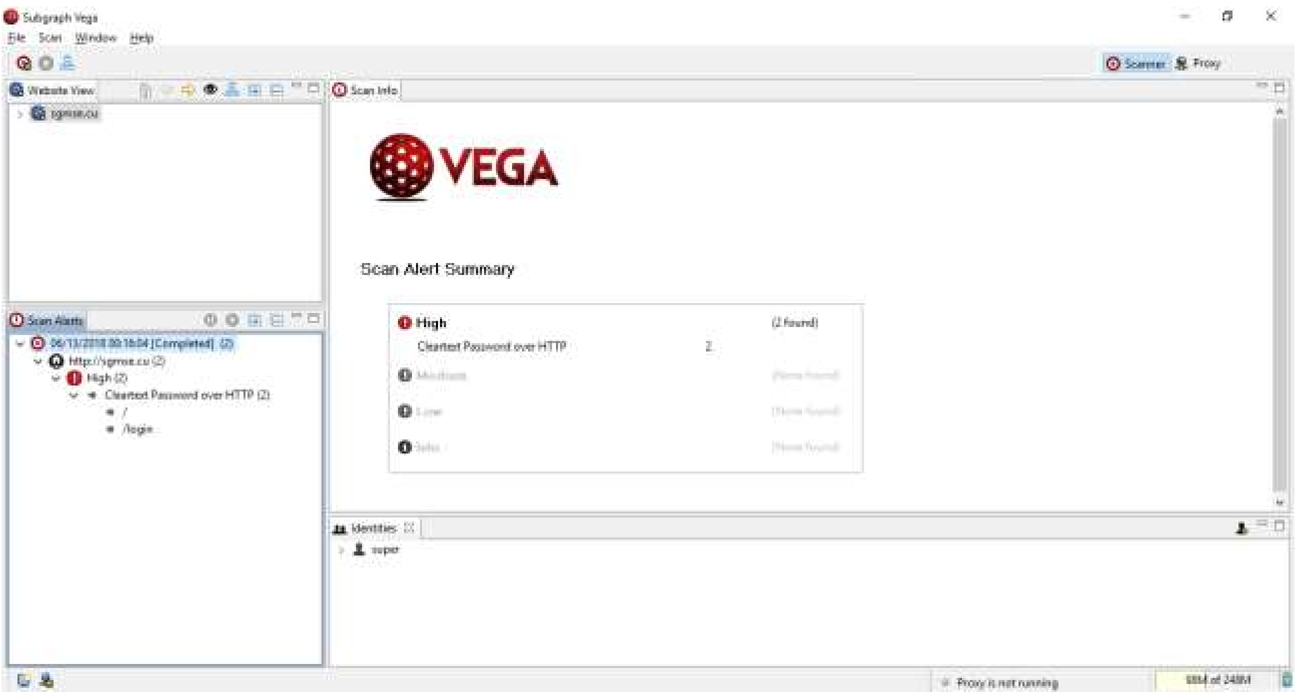
Casos de Excepción:	**Comentarios:**	
Aprobado por: Mario Santana Rodríguez	**Cargo:** Líder de pruebas	**Fecha:**

<table>
<tr><td>No.</td><td colspan="3">2</td></tr>
<tr><td>Requerimiento</td><td colspan="3">Eliminar la aplicación del mantenimiento</td></tr>
<tr><td>Objetivo</td><td colspan="3">Probar la acción de eliminar la aplicación del mantenimiento(clases equivalentes 2)</td></tr>
<tr><td>Tipo de Prueba</td><td colspan="3">Funcional</td></tr>
<tr><td>Hardware</td><td colspan="3">Un procesador Core i5 , disco duro de 750GB, memoria RAM de 6GB</td></tr>
<tr><td>Software</td><td colspan="3">Sistema Operativo Windows 7 o superior, Base de Datos Xampp, Navegador Google Chrome 60.0.3 o superior</td></tr>
<tr><td>Personal</td><td colspan="3">Ingeniero de Pruebas</td></tr>
<tr><td colspan="4">Casos de Prueba</td></tr>
<tr><td>Datos de Entrada</td><td colspan="3">Nombre: NULL</td></tr>
<tr><td>Resultados Esperados</td><td colspan="3">Mensaje: ¨ No se seleccionó ningún nombre ¨</td></tr>
<tr><td>Resultados Obtenidos</td><td colspan="3">Si(x) No()</td></tr>
<tr><td colspan="2">Casos de Excepción:</td><td colspan="2">Comentarios:</td></tr>
<tr><td>Aprobado por: Mario Santana Rodríguez</td><td colspan="2">Cargo: Líder de pruebas</td><td>Fecha:</td></tr>
</table>

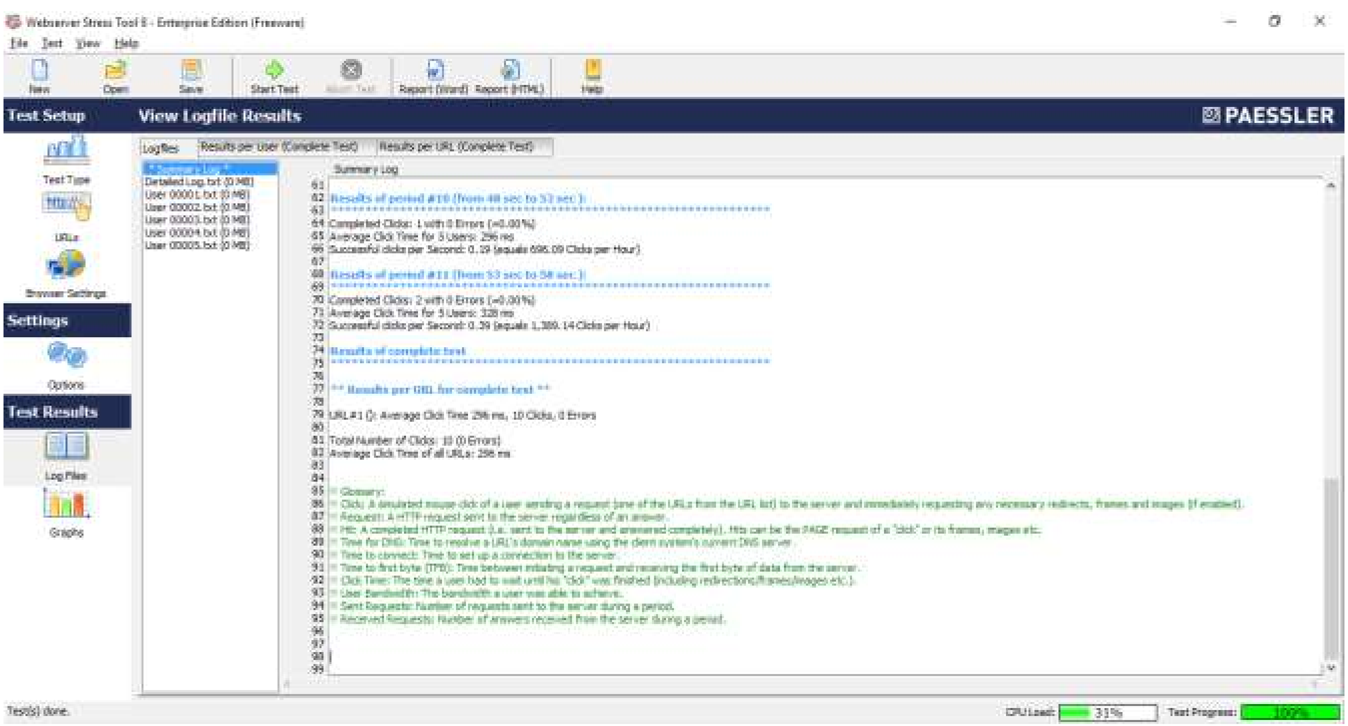
Herramientas de Pruebas.

También se emplearon herramientas para hacer pruebas tal es el caso del software Subgraph Vega, Webstress Tool y Xenu obteniendo los siguientes resultados:

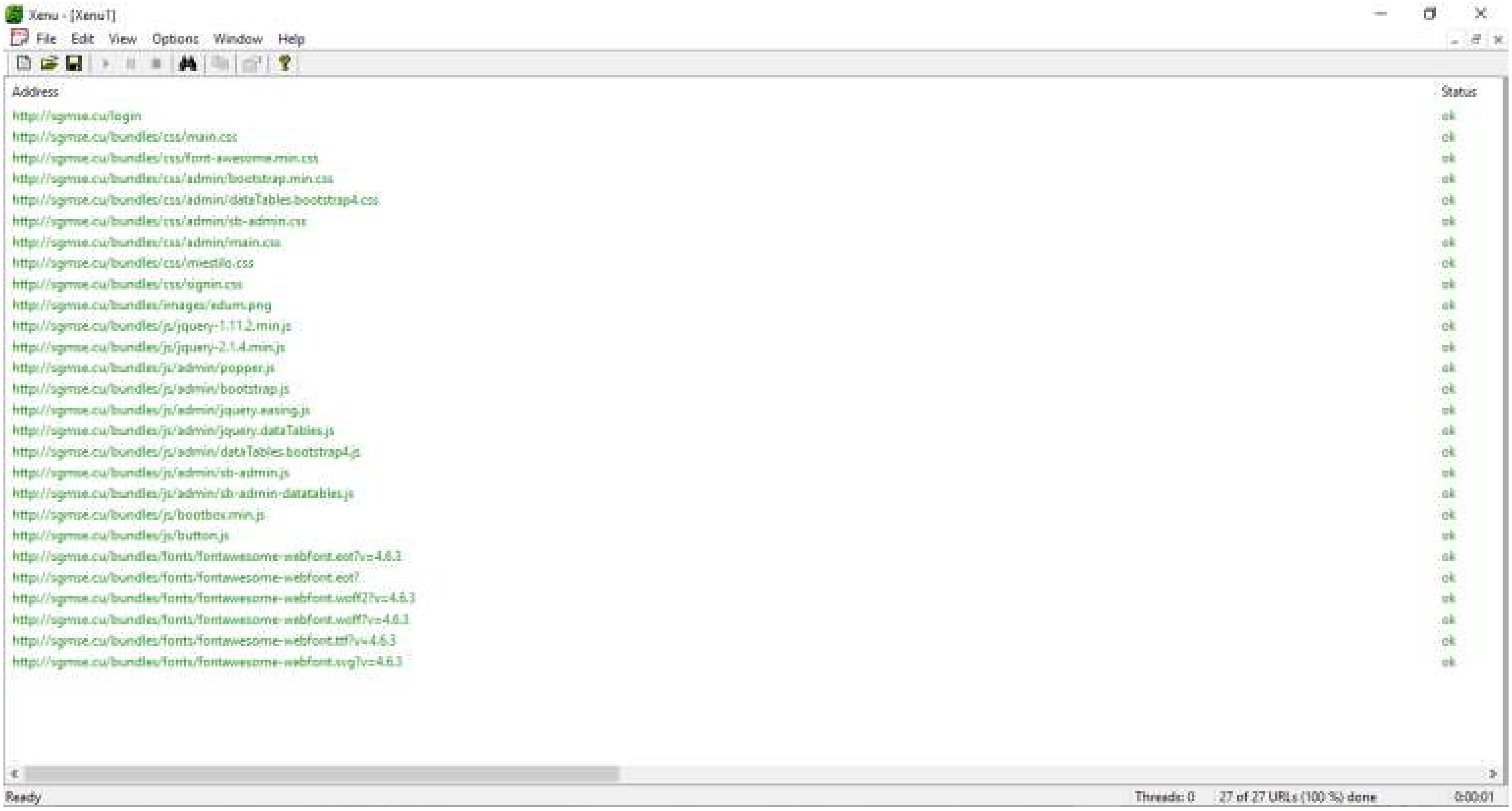
Subgraph Vega



Webstress Tool



Xenu



Análisis de los resultados obtenidos.

Al concluir el desarrollo del proceso de pruebas se lograron resultados satisfactorios en el progreso del sistema con un nivel medio de sencillez. Las pruebas fueron desplegadas por cada una de las historias de usuario. Los elementos de pruebas abordados, permitieron validar el funcionamiento de la aplicación y los resultados satisfactorios de dichas pruebas.

Conclusiones del capítulo

Se establecieron los componentes de los artefactos fundamentales de Scrum para una descripción detallada de los procesos a realizar en el software que permita una comprensión detallada de cada uno de ellos.

Se describe la solución a la problemática planteada en cada artefacto y elemento del desarrollo de la metodología Scrum lo que permitió describir la pila del producto y cada iteración a realizar.

Se detallan las pruebas realizadas para detectar los errores cometidos durante la implementación del sistema que fueron corregidos durante dos meses. Se realizaron pruebas automatizadas para evaluar los niveles de estrés y carga que soporta el sistema, así como los errores de seguridad.

Conclusiones Generales

Se cumplieron los objetivos planteados y de esta forma se determinaron las siguientes conclusiones:

1.Se determinó el marco teórico referencial sobre la gestión de la información generada en el proceso de mantenimiento a softwares educativos que se exponen a continuación:

- Se normalizaron los fundamentos teóricos que permitan la creación de un sistema web que gestione la información asociada a la aplicación de mantenimientos a softwares educativos
- Se determinó la metodología SCRUM para el desarrollo del sistema web debido a sus características.
- Se determinaron como herramientas más viables para la creación del proyecto el gestor de base de datos MySQL 4.5.1, lenguaje PHP 5.6.21, como servidor web Apache 2.4.17, *framework* Symfony 3.3, el Entorno de Desarrollo Integrado (IDE) utilizado fue JetBrains PhpStorm 2016.1

2.Se implementó una aplicación web que facilite la gestión de la información generada en el proceso de mantenimiento a softwares educativos

3.Se aplicó un sistema de pruebas que permitió detectar errores cometidos durante los procesos anteriores del desarrollo del sistema web para la gestión de los riesgos en el desarrollo de los softwares educativos y corregirlos.

Recomendaciones

Desde el punto de vista del alcance del presente trabajo y teniendo en cuenta el momento de desarrollo del mismo, se proponen las siguientes recomendaciones:

1. Agregarle nuevos reportes que sean de interés para los usuarios que interactúen con la aplicación.
2. Implantar el sistema en el repositorio nacional para generalizar su introducción en todos los CES del país.

Bibliografía

Aguilar-Vera, R. A., Aké, I. E., & Ucán-Pech, J. P. (2015). Developing virtual learning environments for software enginnering education: a ludic proposal. *EDULEARN15 Proceedings*.

Armendáriz Barreno, G. A., & Guaraca, M. G. S. (2013). *Adaptación de las metodologias ágiles scrum y extreme game development en una metodologia para desarrollo de videojuegos en android. Caso práctico: Desarrollo de un videojuego.* (Tesis de Grado previa a la Obtencion del Título de: Ingenieros en Sistemas Informáticos), Escuela Superior Politécnica Del Chimborazo, Riobamba – Ecuador.

Assis, M. P. d., & Almeida, M. E. B. d. (2017). Learning Design and Technologies: Creating Collaborative Environments for the Learning Process. *Revista Psicologia da Educação, 44*(1), 47-56. doi: 10.5935/2175-3520.20170005

Bernardinia, S., Porayska-Pomstab, K., & Smith, T. J. (2014). ECHOES: an Intelligent Serious Game for Fostering Social Communication in Children with Autism. *Information Sciences, 264*, 41-60.

Bradford, L. M. (2012). *The Viability of Virtual Worlds in Higher Education: Can Creativity Thrive Outside the Traditional Classroom Environment?* (Doctor of Philosophy), Brigham Young University, Brigham. England.

Brox, C. (2009). A Business Model for the Exchange of E-Learning Courses in an International Network. In M. Stansfield & T. Connolly (Eds.), *Institutional Transformation through Best Practices in Virtual Campus Development: Advancing E-Learning Policies.* . Hershey • New York: University of the West of Scotland, UK.

Bul, K. C. M., Franken, I. H. A., Van der Oord, S., Kato, P. M., Danckaerts, M., Vreeke, L. J., . . . Maras, A. (2015). Development and User Satisfaction of

"Plan-It Commander," a Serious Game for Children with ADHD. *Games for Health Journal, 4*(6), 502-512. doi: 10.1089/g4h.2015.0021

Coomans, S., & Lacerda, G. S. (2015). PETESE, a Pedagogical Ergonomic Tool for Educational Software Evaluation. *Procedia Manufacturing, 3*, 5881-5888. doi: 10.1016/j.promfg.2015.07.895

Choi, C., Seok, M.-G., Choi, S. H., & Kim, T. G. (2015). Military serious game federation development and execution process based on interoperation between game application and constructive simulators. *Simulation and Process Modelling, 10*(2), 103-116.

Epping, A., & Lott, C. (1994). *Does Software Design Complexity Affect Maintenance Effort?* Paper presented at the Proc. 19th Software EngineeringWorkshop, USA.

Jain, R., & Suman, U. (2015). A Systematic Literature Review on Global Software Development Life Cycle. *ACM SIGSOFT Software Engineering Notes, 40*(2), 1-14. doi: 10.1145/2735399.2735408

Pereira dos Santos, R. (2016). *Managing and monitoring software ecosystem to support Demand and solution analysis.* (Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro), Universidade Federal do Rio de Janeiro

Rio de Janeiro, Brasil.

Rabbi, F., & Mannan, K. O. B. (2016). A Short Review for Selecting the Best Tools and Techniques to Perform Software Risk Management. *European Journal of Advances in Engineering and Technology, 3*(6), 1-7.

Rios Herrera, D. A. (2016). *Dirección de proyectos de software desde la metodología PMBOK®.* (Ingeniería de Sistemas y Computación), Universidad Tecnológica de Pereira, Pereira - Risaralda.

Rodríguez, J. S. P. (2017). *Análisis de frameworks php para entornos de la web semántica y su aplicación a un módulo en MOODLE.* (Trabajo de titulación presentado para optar el grado académico de: Ingeniero en Sistemas Informáticos), Escuela Superior Politécnica de Chimborazo. Facultad de Informática y Electrónica, Chimborazo, Ecuador.

Suh, Y. S. (2014). Development of educational software for beam loading analysis using pen-based user interfaces. *Journal of Computational Design and Engineering, 1*(1), 67-77. doi: 10.7315/jcde.2014.007

Xie, T., Tillmann, N., & Halleux, J. d. (2013). *Educational Software Engineering: Where Software Engineering, Education, and Gaming Meet*. Paper presented at the Games and Software Engineering (GAS), New York, USA.

Aguilar-Vera, R. A., et al. (2015). "Developing virtual learning environments for software enginnering education: a ludic proposal." EDULEARN15 Proceedings.

Armendáriz Barreno, G. A. and M. G. S. Guaraca (2013). Adaptación de las metodologias ágiles scrum y extreme game development en una metodologia para desarrollo de videojuegos en android. Caso práctico: Desarrollo de un videojuego. Escuela de Ingeniería en Sistemas, Facultad de Informatica y Electrónica. Riobamba – Ecuador, Escuela Superior Politécnica Del Chimborazo. **Tesis de Grado previa a la Obtencion del Título de: Ingenieros en Sistemas Informáticos:** 332.

Assis, M. P. d. and M. E. B. d. Almeida (2017). "Learning Design and Technologies: Creating Collaborative Environments for the Learning Process." Revista Psicologia da Educação **44**(1): 47-56.

Bernardinia, S., et al. (2014). "ECHOES: an Intelligent Serious Game for Fostering Social Communication in Children with Autism." Information Sciences **264**: 41-60.

Bradford, L. M. (2012). The Viability of Virtual Worlds in Higher Education: Can Creativity Thrive Outside the Traditional Classroom Environment? Department of Instructional Psychology and Technology. Brigham. England, Brigham Young University. **Doctor of Philosophy:** 131.

Brox, C. (2009). A Business Model for the Exchange of E-Learning Courses in an International Network. Institutional Transformation through Best Practices in Virtual Campus Development: Advancing E-Learning Policies. . M. Stansfield and T. Connolly. Hershey • New York, University of the West of Scotland, UK.

Bul, K. C. M., et al. (2015). "Development and User Satisfaction of "Plan-It Commander," a Serious Game for Children with ADHD." Games for Health Journal **4**(6): 502-512.

Coomans, S. and G. S. Lacerda (2015). "PETESE, a Pedagogical Ergonomic Tool for Educational Software Evaluation." Procedia Manufacturing **3**: 5881-5888.

Choi, C., et al. (2015). "Military serious game federation development and execution process based on interoperation between game application and constructive simulators." Simulation and Process Modelling **10**(2): 103-116.

Epping, A. and C. Lott (1994). Does Software Design Complexity Affect Maintenance Effort? Proc. 19th Software EngineeringWorkshop. USA, NASA**:** 1-16.

Jain, R. and U. Suman (2015). "A Systematic Literature Review on Global Software Development Life Cycle." ACM SIGSOFT Software Engineering Notes **40**(2): 1-14.

Pereira dos Santos, R. (2016). Managing and monitoring software ecosystem to support Demand and solution analysis. Engenharia de Sistemas e Computação. Rio de Janeiro, Brasil, Universidade Federal do Rio de Janeiro

Tesi de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro: 246.

Rabbi, F. and K. O. B. Mannan (2016). "A Short Review for Selecting the Best Tools and Techniques to Perform Software Risk Management." European Journal of Advances in Engineering and Technology **3**(6): 1-7.

Rios Herrera, D. A. (2016). Dirección de proyectos de software desde la metodología PMBOK®. Facultad de Ingenierías. Pereira - Risaralda, Universidad Tecnológica de Pereira. **Ingeniería de Sistemas y Computación:** 54.

Rodríguez, J. S. P. (2017). Análisis de frameworks php para entornos de la web semántica y su aplicación a un módulo en MOODLE. Escuela de Ingeniería En Sistemas. Chimborazo, Ecuador, Escuela Superior Politécnica de Chimborazo. Facultad de Informática y Electrónica. **Trabajo de titulación presentado para optar el grado académico de: Ingeniero en Sistemas Informáticos:** 153.

Suh, Y. S. (2014). "Development of educational software for beam loading analysis using pen-based user interfaces." Journal of Computational Design and Engineering **1**(1): 67-77.

Xie, T., et al. (2013). Educational Software Engineering: Where Software Engineering, Education, and Gaming Meet. Games and Software Engineering (GAS). New York, USA, IEEE**:** 36-39.

Aguilar-Vera, R. A., et al. (2015). "Developing virtual learning environments for software enginnering education: a ludic proposal." EDULEARN15 Proceedings.

Armendáriz Barreno, G. A. and M. G. S. Guaraca (2013). Adaptación de las metodologias ágiles scrum y extreme game development en una metodologia para desarrollo de videojuegos en android. Caso práctico: Desarrollo de un videojuego. Escuela de Ingeniería en Sistemas, Facultad de Informatica y Electrónica. Riobamba – Ecuador, Escuela Superior Politécnica Del Chimborazo. **Tesis de Grado previa a la Obtencion del Título de: Ingenieros en Sistemas Informáticos:** 332.

Assis, M. P. d. and M. E. B. d. Almeida (2017). "Learning Design and Technologies: Creating Collaborative Environments for the Learning Process." Revista Psicologia da Educação **44**(1): 47-56.

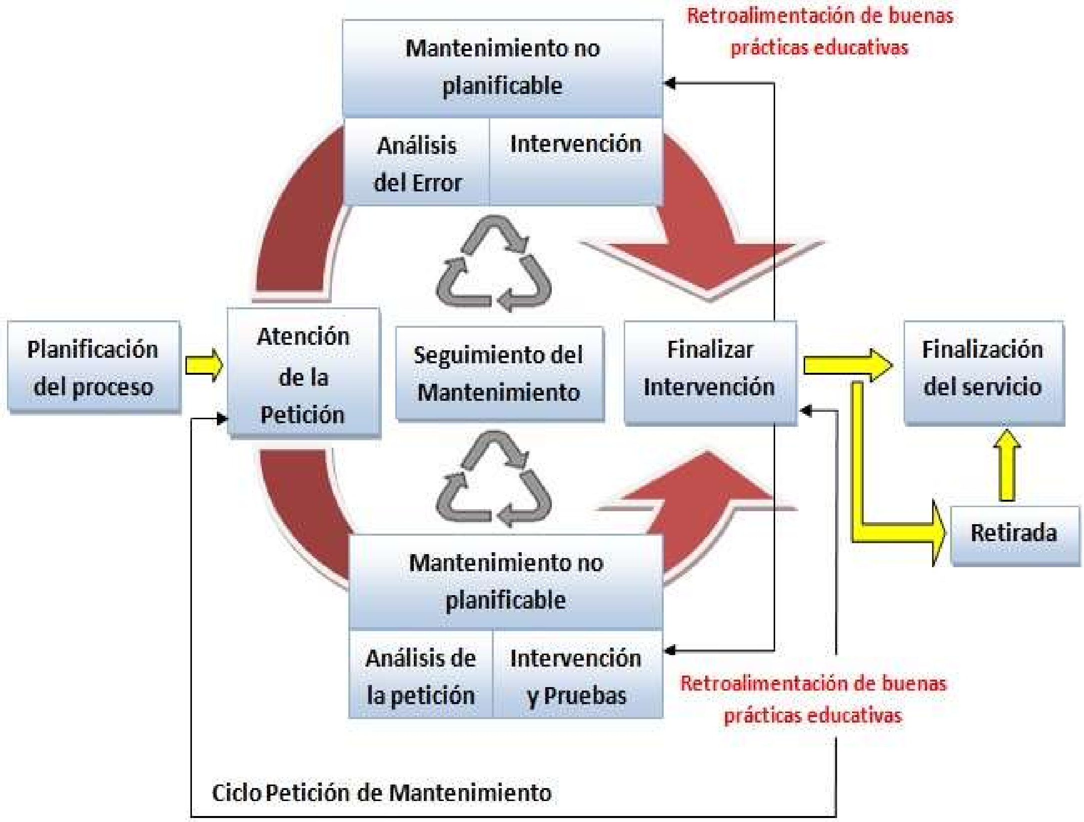
Bernardinia, S., et al. (2014). "ECHOES: an Intelligent Serious Game for Fostering Social Communication in Children with Autism." Information Sciences **264**: 41-60.

Bradford, L. M. (2012). The Viability of Virtual Worlds in Higher Education: Can Creativity Thrive Outside the Traditional Classroom Environment? Department of

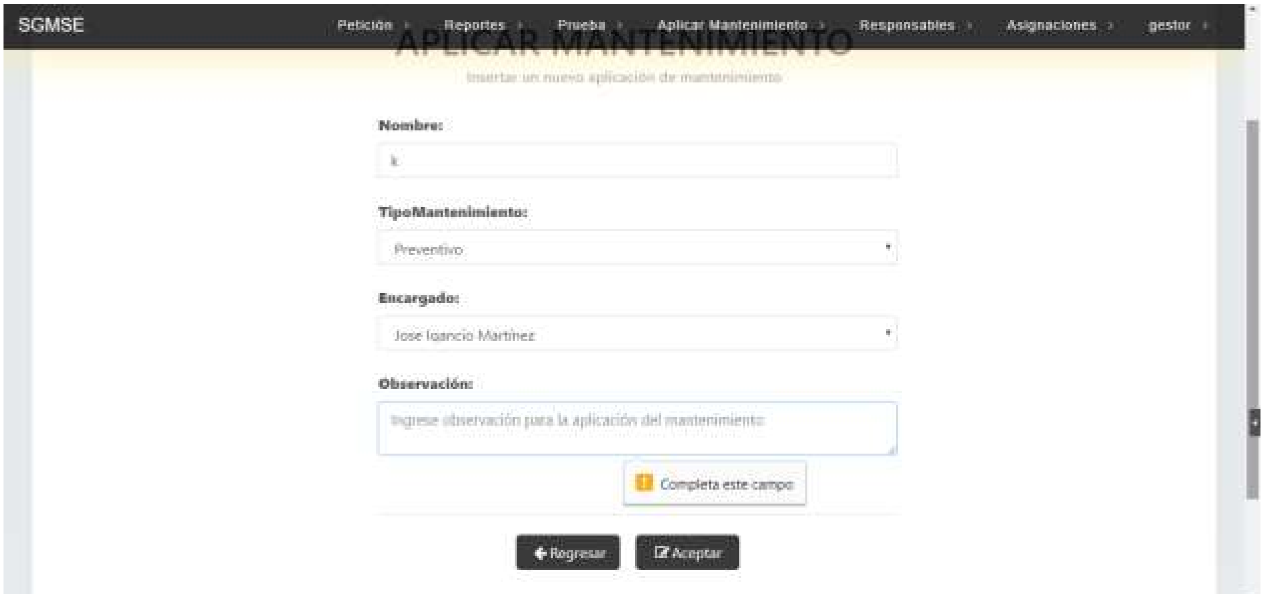
Instructional Psychology and Technology. Brigham. England, Brigham Young University. **Doctor of Philosophy:** 131.

Anexos

Proceso de EDUMANTE.



Imágenes relacionadas con los casos de prueba



Tablas de ponderaciones para EI, EQ, EO, ILF, EIF

CLASIFICACION DE ENTRADAS Y CONSULTAS	1-4 Atributos	5-15 Atributos	Más de 15 Atributos
0 o 1 ficheros accedidos	BAJA 3	BAJA 3	MEDIA 4
2 ficheros accedidos	BAJA 3	MEDIA 4	ALTA 6
Más de 2 ficheros accedidos	MEDIA 4	ALTA 6	ALTA 6

CLASIFICACION DE SALIDAS	1-5 Atributos	6-19 Atributos	Más de 19 Atributos
0 o 1 ficheros accedidos	BAJA 4	BAJA 4	MEDIA 5
2 o 3 ficheros accedidos	BAJA 4	MEDIA 5	ALTA 7
Más de 3 ficheros accedidos	MEDIA 5	ALTA 7	ALTA 7

FICHEROS LÓGICOS INTERNOS	1-19 Atributos	20-50 Atributos	Más de 50 Atributos
1 Entidad o registro lógico	BAJA 7	BAJA 7	MEDIA 10
2 - 5 Entidades o registros lógicos	BAJA 7	MEDIA 10	ALTA 15
Más de 5 Entidades o registros lógicos	MEDIA 10	ALTA 15	ALTA 15

FICHEROS LÓGICOS EXTERNOS	1-19 Atributos	20-50 Atributos	Más de 50 Atributos
1 Entidad o registro lógico	BAJA 5	BAJA 5	MEDIA 7
2 - 5 Entidades o registros lógicos	BAJA 5	MEDIA 7	ALTA 10
Más de 5 Entidades o registros lógicos	MEDIA 7	ALTA 10	ALTA 10

Entorno y Lenguaje	Líneas de Código por PF	Horas por PF	Entorno y Lenguaje
Lenguajes 2GL: Ensamblador, C,…	300	20 a 30	**Lenguajes 2GL: Ensamblador, C,…**
Lenguajes 3GL: Cobol	100	10 a 20	**Lenguajes 3GL: Cobol**
Lenguajes 4GL: VisualXX	20	5 a 10	**Lenguajes 4GL: VisualXX**