Serie Científica de la Universidad de las CienciasInformáticas
Vol. 12, No. 9, Mes Septiembre, 2019
ISSN: 2306-2495| RNPS: 2343
Pág. 112-127
http://publicaciones.uci.cu

Tipo de artículo: Artículo original
Temática: Ingeniería y Gestión de Software
Recibido: 15/06/2019 | Aceptado: 20/09/2019 | Publicado: 22/09/2019

Propuesta de Automatización de Pruebas Funcionales Durante el Ciclo de Vida del Software en Desoft

Functional Testing Automation Proposal During the Software Lifecycle in Desoft

Yadira Ramos Marén [1*], Zoila Esther Morales Tabares [2], Yaimí Trujillo Casañola [3]

[1] Empresa Cubana del Software Desoft. Carretera de la Gran Piedra km1 El Sapo. yadira.ramos@scu.desoft.cu
[2] Universidad de Ciencias Informáticas (UCI). Universidad de Ciencias Informáticas. zemorales@uci.cu
[3] Universidad de Ciencias Informáticas (UCI). Universidad de Ciencias Informáticas. yaimi@uci.cu

[*] Autor para correspondencia: yadira.ramos@scu.desoft.cu

Resumen
Durante el proceso de gestión y administración de proyectos, la calidad del software es un factor importante a tener en cuenta. Contribuir con la calidad se ha convertido en una necesidad prioritaria para las organizaciones en aras de mantener la rentabilidad y un nivel de prestigio aceptable en el mercado. Un requisito indispensable para ello lo constituye la realización de pruebas funcionales durante la construcción de un producto, ya que figuran como un instrumento para evaluar la calidad al detectar los defectos existentes en el software.
Este manuscrito se enfoca en realizar una propuesta de pruebas funcionales automatizadas para las aplicaciones que se desarrollan dentro de la Empresa de Aplicaciones Informáticas (DESOFT) de la provincia Santiago de Cuba. Muestra la información referente a las pruebas funcionales (técnicas, métodos, niveles y tipos) dentro de la calidad del desarrollo de software. Se plasman métricas de pruebas funcionales, criterios de éxito y culminación para las pruebas y un conjunto de buenas prácticas para el proceso de prueba de las aplicaciones. Además, se muestran los resultados experimentales de la investigación.
Palabras clave: Calidad del software, pruebas funcionales automatizadas, métricas.

Abstract
During the process of project management and management, the quality of the software is an important factor to consider. Contributing to quality has become a priority need for organizations in order to maintain profitability and an acceptable level of prestige in the market. An essential requirement for this is the performance of functional tests during the construction of a product, as it is an instrument to evaluate the quality when detecting the defects in the software.
This document focuses on making a proposal of automated functional tests for the applications that are developed within the Enterprise of software Applications (Desoft) of Santiago of Cuba province. It shows the information

Serie Científica de la Universidad de las CienciasInformáticas
Vol. 12, No. 9, Mes Septiembre, 2019
ISSN: 2306-2495| RNPS: 2343
Pág. 112-127
http://publicaciones.uci.cu

regarding functional tests (techniques, methods, levels and types) within the quality of software development. Metrics of functional tests, success criteria and culmination of tests and a set of good practices for the test process of the applications are expressed. The research results are also shown. The proposal contributes to reduce costs by reducing the time of execution of the tests and increasing the quality of the process in the company.
Keywords: *Software quality, automated functional tests, metrics.*

Introducción

En estos tiempos de tanta globalización y competencia se hace necesario lograr un mejor desempeño de las organizaciones en aras de alcanzar una mayor profundización y agilidad en el proceso de toma de decisiones. Las exigencias del mercado imponen que las instituciones tengan en cuenta un grupo de elementos que les permita maximizar su productividad e insertarse con éxito en el mismo.

La industria del software, a pesar de ser una de las últimas en llegar al mercado, ha alcanzado grandes niveles de desarrollo, lo que constituye un reto para cualquier empresa desarrolladora de sistemas informáticos. No solo deben tener en cuenta el alto nivel de competencia que existe en este mundo, sino además el crecimiento del interés de los clientes y usuarios por la calidad a medida que se vuelven más selectivos y comienzan a rechazar productos poco fiables o que no dan respuesta a sus necesidades. Sin embargo, no existe una forma de organización global definida que rija el desarrollo o forma de implementación para la industria del software. Según sus características y posibilidades cada país adopta la manera que considera más favorable. No obstante, la calidad del software sigue siendo una prioridad, que teniendo en cuenta la percepción del cliente y de los usuarios, se analiza principalmente por los fallos que se encuentran en el producto y por la influencia que tienen para el correcto funcionamiento del negocio del cliente. [Scalone, 2006, Hernández, 2009].

En la sociedad actual para las empresas desarrolladoras de software ser rentables y mantener un cierto nivel de prestigio, necesitan asegurar que sus productos tengan una calidad aceptable antes de su instalación en el ambiente del cliente. Para conseguir esta calidad en un software es tan importante la especificación como la evaluación del mismo.

Una de las empresas líderes en la producción y comercialización de software en Cuba es la Empresa de Aplicaciones Informáticas (DESOFT). La misma fue sometida a un proceso de revisión de sus procesos en el año 2013 en el cual se detectó que particularmente en el proceso de pruebas existen un grupo de factores que atentan contra la calidad del desarrollo de sistemas lo que definitivamente influye en la satisfacción del cliente, daña la imagen de la empresa y disminuye los niveles de productividad. Al revisar las condiciones que son necesarias para la realización exitosa de

las actividades competentes a las tareas de pruebas de software, se identificó que no hay completitud en la evaluación de la calidad, puesto que esta se circunscribe a la opinión que pueda suministrar el cliente. Los criterios de análisis difieren según el evaluador, pues son basados en la experiencia y no en un proceso acertado y bien definido de evaluación. Los indicadores para evaluar la calidad no siempre son objetivos y existe descontrol de la gerencia sobre el trabajo realizado durante el proceso. De manera general se carece de mecanismos que posibiliten garantizar la calidad del desarrollo de sistemas.

Aunque esta situación ha mejorado considerablemente las pruebas son realizadas en el Departamento de Calidad por los especialistas probadores de forma manual, lo que trae consigo demoras en el tiempo de ejecución de las mismas y en ocasiones productos liberados con errores no encontrados debido al nivel de familiarización que alcanzan los probadores con el sistema.

Para una empresa con las características de Desoft que trabaja de cara a un cliente no es conveniente que esto ocurra, ya que la demora trae consigo aumento de los costos de producción y uso indiscriminado de los recursos. Además, liberar un producto con errores a un cliente que en la mayaría de los casos no conoce suficientemente las características del desarrollo de un software y subestima el trabajo de un equipo de desarrollo atenta considerablemente contra el prestigio de la empresa y de los especialistas que la integran.

Teniendo en cuenta lo antes expuesto surge la necesidad de crear un mecanismo de apoyo a las pruebas funcionales que permita reducir los tiempos de ejecución de las mismas, disminuya los errores humanos introducidos en el proceso y contribuya a la liberación de productos con la menor cantidad de errores posibles.

Se realiza la presente investigación, para obtener una propuesta de pruebas funcionales automatizadas durante el ciclo de vida del software en Desoft y se capacite a los especialistas probadores del Departamento de Calidad.

Muchos han sido los autores que han brindado sus aportes en los temas relacionados con las pruebas de software. A continuación, se explican brevemente algunas definiciones:

“Las pruebas de software son un elemento crítico para la garantía de calidad del software y representa una revisión final de las especificaciones, del diseño y la codificación”. [Pressman, 2002].

“La prueba de software es una actividad en la cual un sistema o componente es ejecutado bajo condiciones específicas, se observan o almacenan los resultados y se realiza una evaluación de algún aspecto del sistema o componente”. [IEEE, 1990]

“Conjunto de propiedades y de características de un producto o servicio, que le confieren aptitud para satisfacer unas necesidades explícitas o implícitas”. [ISO 8402:1995]

Materiales y métodos o Metodología computacional

Cada vez es más crítico ir a la par de los exigentes tiempos de entrega y cambios demandados por las áreas de negocio. Una forma de lograr esto es automatizando las pruebas con la finalidad de que se puedan ejecutar de forma rápida y repetitiva. Las herramientas de automatización de las pruebas permiten que una máquina se encargue de realizar una prueba predefinida, haciendo que el proceso de ejecución de las pruebas sea más rápido, preciso y pueda repetirse cuando se desee. Frente a los nuevos riesgos que presenta la creciente complejidad del software, las herramientas de prueba son probablemente la única forma práctica para que el cliente confíe en que obtuvo el valor justo de lo que pagó.

Las herramientas de prueba soportan todas las fases de un proyecto, desde la reunión de los requisitos iniciales hasta el soporte del sistema después del despliegue. Se pueden utilizar para diseñar, codificar y, por supuesto, evaluar un sistema. Muchas han sido las clasificaciones que se le han dado a las herramientas que apoyan los diversos aspectos relacionados con las pruebas:

- **Administración de las pruebas y el proceso de pruebas:** herramientas para la administración de las pruebas, para el seguimiento de incidentes, para la gestión de la configuración y para la administración de requerimientos.
- **Pruebas estáticas:** herramientas para apoyar el proceso de revisión, herramientas para el análisis estático y herramientas de modelado.
- **Especificación de las pruebas:** herramientas para el diseño de las pruebas y para la preparación de datos de prueba.
- **Ejecución de las pruebas:** herramientas de ejecución de casos de prueba, herramientas de pruebas unitarias, comparadores, herramientas de medición del cubrimiento, herramientas de seguridad.
- **Desempeño y monitorización:** herramientas de análisis dinámico, herramientas de desempeño, de carga y de estrés, herramientas de monitorización.

Para la automatización de las pruebas funcionales se recomienda utilizar las herramientas de captura y reproducción. Permiten a los especialistas probadores capturar y grabar pruebas con el objetivo de editarlas, modificarlas y reproducirlas en distintos entornos. Durante la grabación son capturadas las acciones realizadas por el probador, creando así, de manera automática un script en algún lenguaje de alto nivel. El script es modificado por el probador para crear una prueba reusable y mantenible. Este script se vuelve la línea base y luego es reproducido en una nueva versión, contra la cual es comparado. En general estas herramientas vienen acompañadas de un comparador, que compara automáticamente la salida en el momento de ejecutar el script con la salida grabada.

Serie Científica de la Universidad de las CienciasInformáticas
Vol. 12, No. 9, Mes Septiembre, 2019
ISSN: 2306-2495| RNPS: 2343
Pág. 112-127
http://publicaciones.uci.cu

A continuación, se realiza una breve explicación de las herramientas utilizadas:

JMeter

El Apache JMeter es un software de código abierto, una aplicación diseñada en JAVA para medir el rendimiento y comportamiento de servidores mediante pruebas. Originalmente se diseñó para pro-bar aplicaciones Web, pero se ha ampliado desde entonces a otras funciones. Se utiliza para probar el rendimiento tanto de los recursos estáticos y dinámicos (archivos, Servlets, scripts de Perl, objetos Java, bases de datos - consultas, servidores FTP y mucho más). Se puede utilizar para simular una carga pesada en un servidor, la red o un objeto para poner a prueba su resistencia o para analizar el rendimiento global en diferentes tipos de carga. Puede usarlo para hacer un análisis gráfico de rendimiento o para probar el comportamiento de diferentes elementos con un gran volumen de carga y concurrencia. [Martínez, 2009]. Algunos de los tipos de Servidor que se pueden probar son:

- Web HTTP y HTTPS.
- SOAP.
- Base de datos a través de JDBC.
- LDAP.

Selenium IDE

Es una extensión para el navegador Web Firefox. Permite grabar clicks, tipeo y otras acciones para realizar test. Estos scripts grabados, luego pueden ser exportados en distintos lenguajes (PHP, JAVA, Ruby, C, etc.) para su posterior adaptación y utilización.

Es un entorno de pruebas de software para aplicaciones basadas en la web. Selenium provee una herramienta de grabar/reproducir para crear pruebas sin usar un lenguaje de scripting para pruebas. Incluye también un lenguaje específico de dominio para escribir pruebas en un amplio número de lenguajes de programación incluyendo Java, C#, Ruby, Groovy, Perl, Php y Python. Las pruebas pueden ejecutarse entonces usando la mayoría de los navegadores web modernos en diferentes sistemas operativos como Windows, Linux y OSX. [https://www.seleniumhq.org/docs/02_selenium_ide.jsp].

BadBoy

Es una herramienta de gran alcance diseñada para ayudar en la prueba y en el desarrollo de aplicaciones. Permite efectuar el testeo de la Web con docenas de características incluyendo una interfaz simple, fácil e intuitiva, mediante

los métodos de captura y repetición, siendo una gran ayuda para la prueba de carga de gran alcance, informes detallados, gráficos, etc. Permite grabar y luego reproducir las acciones realizadas por los usuarios, luego este script puede ser utilizado en otras herramientas, como JMeter. Se puede integrar al navegador Web Internet Explorer.

Sin embargo, para que la automatización de las pruebas tenga éxito es necesario elegir correctamente los casos de prueba para la automatización y no olvidar que los scripts son un apoyo a las pruebas manuales y no la sustitución de ellas. Como sea, para hacer una correcta elección de los casos de prueba uno de los factores fundamentales a tener en cuenta es la complejidad de los mismos. [Martínez, 2009].

Selección de los casos de prueba a automatizar

La (IEEE, 1990), define caso de prueba (CP) como:

Un conjunto de entradas, condiciones de ejecución y resultados esperados diseñados para un objetivo particular.

Para llevar a cabo un proceso de pruebas se pueden identificar un conjunto de acciones, dentro de las cuales podemos citar las siguientes [Aristegui, 2010]:

a. Preparar CP.
b. Ejecutar los CP.
c. Suspender la prueba.
d. Evaluar resultados obtenidos.
e. Dar un criterio de evaluación.

De las acciones anteriormente presentadas surgen cuestionamientos, como los que se muestran a continuación:

a) ¿Cómo seleccionar casos de prueba representativos?
b) ¿Cuántas pruebas realizar?
c) ¿Cómo decidir si es o no de calidad el producto evaluado?

Los cuales permiten entender que cada una de estas acciones requiere especial atención. Mostrando especial interés en responder la primera interrogante y atendiendo a la complejidad, podemos encontrar casos de pruebas simples, medios y complejos, esta complejidad la da el número de pasos (acciones) y verificaciones que tenga cada uno de ellos. Para entender con mayor claridad se puede hacer una tabla como la que se muestra a continuación, donde se recoja la información necesaria de cada caso de prueba y a partir de ahí realizar entonces una selección.

	Complejidad del	**Número de**	**Número de**	**Buen candidato (Número**

Serie Científica de la Universidad de las CienciasInformáticas
Vol. 12, No. 9, Mes Septiembre, 2019
ISSN: 2306-2495| RNPS: 2343
Pág. 112-127
http://publicaciones.uci.cu

	Caso de Prueba	**acciones**	**verificaciones**	**de ejecuciones)**
1	Simple	<5	< 5	> 15 ejecuciones
2	Medio	>5 y <15	> 5 - < 10	> 8 - 10 ejecuciones
3	Complejo	> 15 - < 25	> 10 - < 15	> 5 - 8 ejecuciones

Tabla 1 Descripción de los CP según la complejidad

Se debe tener en cuenta que los scripts comprenden tanto los pasos de las acciones como los puntos de verificación o los resultados esperados. Las acciones suelen ser métodos directos o llamados a funciones del lenguaje de la herramienta de automatización, sin embargo, los puntos de verificación no son de ese tipo y son más complejos.

El error más común en los intentos de automatización es intentar automatizar todo, así que la recomendación aquí es ***no intente automatizar todos los casos de prueba***. Teniendo una visión del negocio y también de la estructura interna del sistema, para lo cual se necesita involucrar a los desarrolladores se debe decidir cuales casos de prueba automatizar. Algunas de los factores a tener en cuenta son:

a) ¿cuáles casos de prueba se necesitarán correr más veces durante el proyecto? Típicamente aquellos casos de prueba asociados al núcleo de la aplicación se pueden construir al principio del proyecto y luego sirven (adaptándolos cuando sea necesario) para el resto de la vida del sistema.
b) ¿cuáles casos de prueba conllevan mucha dedicación humana y son repetitivos? Hay veces que un caso de prueba requiere de un trabajo tedioso para configurar el ambiente de prueba y luego ejecutar el caso de prueba. Estos casos de prueba es bueno automatizarlos para liberar el recurso humano y que pueda dedicarse a otros casos más desafiantes.
c) ¿qué funcionalidades son críticas para el cliente/usuario? Aquellas funcionalidades que tienen que andar si o si en cada entrega generalmente es interesante incluirlas en los casos automatizados.
d) ¿qué costo tiene asociado la automatización del caso de prueba? Hay casos de prueba que son muy difíciles de automatizar, ya sea porque hay que realizar validaciones visuales complejas o por otras razones. Por este motivo si el costo de su automatización es grande hay veces que es mejor dejarlo para ser ejecutado manualmente

Teniendo en cuenta estas características se puede ponderar cada caso de prueba en base a cada una de ellas para luego hacer un ranking o priorizar los casos de prueba a automatizar. En el caso de que se vaya a automatizar en un proyecto de mantenimiento en el cual ya se están corriendo pruebas manuales, muchas veces la fuente de información

más importante que tenemos para elegir los casos de prueba a automatizar es el historial de incidentes. Con este instrumento, más las consideraciones vistas previamente, se puede tener una idea clara de cuáles son aquellos casos de prueba o funcionalidades a automatizar.

Automatización de las pruebas funcionales en el ciclo de vida del software en Desoft

La Empresa Cubana del Software, Desoft se encarga de brindar servicios informáticos a partir de las necesidades de los clientes. Esta empresa posee una variedad de líneas de negocios encaminadas a brindar a sus clientes soluciones informáticas integrales, aplicadas a sus necesidades mediante una línea de trabajo que incluye una amplia gama de servicios. En los últimos años ha aumentado considerablemente el desarrollo de productos software. Las divisiones territoriales con las que cuenta en todo el país, atendiendo a las necesidades de los clientes brindan los servicios de desarrollo, despliegue y soporte de soluciones informáticas, por mencionar algunos, y una de ellas es Desoft Santiago de Cuba.

La metodología de desarrollo de software que se utiliza es propia de la empresa e incluye las buenas prácticas de las metodologías ágiles Scrum, Xp, Iconix y OpenERP. Esta metodología proporciona la guía de actividades y los flujos de trabajo que organiza el proceso de desarrollo de software. Para manejar el enorme esfuerzo necesario para ejecutar un proyecto con esta metodología es necesario dividir el mismo en iteraciones. Cada iteración del proceso (entregable realizado) toma como entrada el producto resultado de la iteración anterior y genera como salida un producto incrementado, que deberá ir verificando y validando cada iteración con el área de calidad y el cliente. [Varios Autores, 2017].

Según la metodología de Desoft en su versión 3.0 el ciclo de vida del software se define de la siguiente manera:

1. Requerimientos (incluye las etapas de Diagnóstico y Especificación de requisitos).
2. Desarrollo (incluye las etapas de Diseño e Implementación).
3. Pruebas de liberación
4. Despliegue y Capacitación (incluye las etapas de Migración y Capacitación).

En la etapa de pruebas se corrobora que el sistema y los artefactos del proyecto cumplen con la calidad requerida, tanto funcionalmente como por las normas y estándares utilizados. Tiene como objetivos principales: validar la adecuación de cada entregable de cada actividad a los estándares de la empresa, verificar la funcionalidad del sistema contra los requerimientos del cliente, verificar el código fuente para detectar errores y validar la adecuación a los estándares de codificación y validar el producto respecto a nuestros atributos de calidad.

Serie Científica de la Universidad de las CienciasInformáticas
Vol. 12, No. 9, Mes Septiembre, 2019
ISSN: 2306-2495| RNPS: 2343
Pág. 112-127
http://publicaciones.uci.cu

Esta disciplina se comienza a trabajar desde el inicio del proyecto, al tener identificados los requisitos del software, para poder probar la arquitectura seleccionada. Durante esta etapa se generan los siguientes artefactos:

- Aplicación y herramientas que la soportan.
- Manual de Usuario y/o Ayuda.
- Manual de Instalación y Configuración.
- Acta de liberación

Las pruebas que se realizan son manuales y aunque detectan muchos errores pueden mejorar en cuanto a tiempo de ejecución y calidad en el proceso. Por tal motivo se propone automatizar las pruebas con el objetivo de disminuir su tiempo de ejecución, aumentar la calidad del proceso en cuanto a cantidad de errores encontrados y aumentar la profesionalidad de los especialistas probadores con la implementación de un plan de capacitación sobre la automatización de las pruebas funcionales.

Con el objetivo de incidir directamente en los problemas que atentan contra la calidad del proceso de pruebas se realiza un diagrama de campo de fuerzas para encontrar los factores que apoyan o van en contra de la solución.

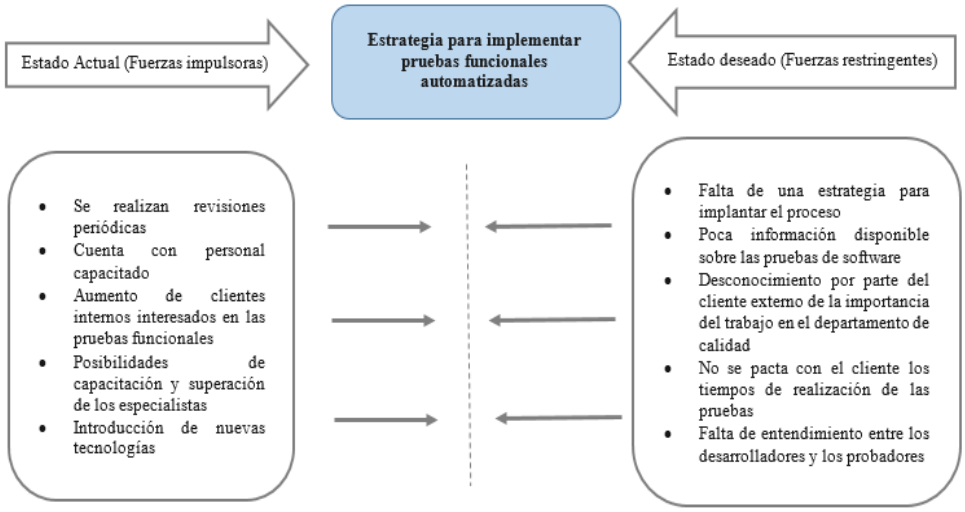

Figura 1Fuerzas impulsoras y fuerzas restringentes para analizar las insuficiencias detectadas en el proceso de realización de pruebas de software en Desoft

A continuación, se explican las etapas de la propuesta de Automatización (Figura 2), así como el plan de capacitación.

Etapa I: Inicio

Pasos:

1.1 Gestión del procedimiento de pruebas de software en Desoft Santiago de Cuba.

Actividades:

1. Analizar las características del proceso de pruebas en Desoft Santiago.
2. Definir los roles involucrados dentro del proceso.
3. Determinar la interacción de los involucrados con el proceso.
4. Esbozar un esquema con los componentes del proceso y la interacción de los involucrados con el mismo.

Resultados:

Se obtienen las características del proceso de pruebas funcionales, así como el nivel de interacción de los involucrados. De esta manera se cumple con el objetivo de conocer el entorno de pruebas actual, y se define la arquitectura que puede ser implantada.

1.2 Gestión del conocimiento

Actividades:

1. Realizar un test para evaluar a los involucrados sobre la utilización de herramientas automatizadas en la realización de pruebas funcionales.
2. Definir mediante pequeños grupos el nivel de conocimiento de los involucrados.
3. Determinar aquellos involucrados que necesitan capacitación con respecto a las herramientas, a partir del resultado del test.
4. Planificar actividades de capacitación teniendo en cuenta el resultado del test y los grupos que se crearon.

Resultados:

Se capacita a los involucrados en la realización de pruebas funcionales automatizadas, para garantizar un nivel de conocimiento adecuado en la interacción con las herramientas.

Etapa II: Planificación

Pasos:

2.1: Gestión de la planificación

Actividades:

1. Planear la planeación de las pruebas por cada proyecto de desarrollo.

2. Planear las pruebas atendiendo a las características de cada proyecto.
3. Definir teniendo en cuenta la metodología de desarrollo cuáles son los artefactos que entran en las pruebas.
4. Definir cuáles son los artefactos de salida.

Resultados:

Se obtiene una planificación confiable de la realización de las pruebas funcionales automatizadas para cada proyecto, así como los artefactos de entrada y salida para cada iteración realizada.

2.2: Gestión de las herramientas

Actividades:

1. Aplicar buenas prácticas para determinar cuáles son los criterios que evaluarán las pruebas en cada etapa.
2. Preparar las herramientas que se van a utilizar, teniendo en cuenta lo que buscan las pruebas por etapas y las herramientas predefinidas.

Resultados:

Se obtiene por cada etapa de realización de las pruebas cuáles son las herramientas que serán utilizadas.

Etapa III: Ejecución

Pasos:

3.1: Utilización de herramientas automatizadas OpenSource en la realización de las pruebas funcionales automatizadas.

Actividades:

1. Introducir los casos de prueba en las herramientas que se utilizarán.
2. Traducir al lenguaje común los scripts generados, mediante la extracción de las no conformidades.
3. Archivar los scripts generados para futuras iteraciones de las pruebas.
4. Documentar las pruebas realizadas.

Resultados:

1. Se obtienen y se archivan los scripts de pruebas generados para futuras iteraciones de las pruebas.
2. Se logra un entendimiento común entre los involucrados al estar debidamente documentado el proceso de pruebas funcionales.

3.2: Estandarización del proceso de realización de pruebas funcionales automatizadas.

Actividades:

1. Identificar fuentes de conocimiento sobre herramientas para automatizar las pruebas funcionales.
2. Documentar casos de prueba representativos.
3. Aplicar buenas prácticas en las técnicas y enfoques metodológicos de las pruebas funcionales.
4. Documentar las no conformidades encontradas.
5. Realizar un repositorio de lecciones aprendidas, con vista a identificar y eliminar errores comunes.
6. Realizar un seguimiento a los scripts de prueba hasta tanto esté iterando el proyecto, y una vez concluidas las iteraciones archivar como evidencia representativa, como mínimo tres meses.
7. Involucrar al cliente en el proceso de realización de las pruebas como un miembro más del equipo de involucrados.

Resultados:

1. Se obtiene un procedimiento estándar para la realización de las pruebas funcionales.
2. Se logra un entendimiento común entre los miembros del proyecto, los probadores y el cliente en cuanto a las pruebas funcionales.

Etapa IV: Cierre

Pasos:

Actividades:

1. Aplicar entrevistas y encuestas a los involucrados en el proceso.
2. Recolectar y medir los resultados obtenidos.
3. Tabulación de resultados para autoevaluar la implantación de la estrategia en el proceso.

Resultados:

Un informe de los resultados obtenidos con la implantación de la estrategia en el proceso de realización de pruebas funcionales.

Plan de capacitación:

A partir de los resultados obtenidos en la etapa IV de la propuesta, se implementa un plan de capacitación para los involucrados en la etapa de pruebas de software en Desoft. Los involucrados son: Especialista Principal Área de Desarrollo, Especialista Principal de Calidad, Jefe de Proyecto, Especialista Probador. La capacitación incluye los temas relacionados con las herramientas utilizadas en cuanto a:

1. Instalación.

Serie Científica de la Universidad de las CienciasInformáticas
Vol. 12, No. 9, Mes Septiembre, 2019
ISSN: 2306-2495| RNPS: 2343
Pág. 112-127
http://publicaciones.uci.cu

2. Manual de usuario.
3. Práctica a partir de un caso de estudio guiado por video.
4. Preguntas de participación grupal como buenas prácticas.
5. Práctica de un caso de estudio a partir de equipos conformados teniendo en cuenta los resultados obtenidos en las preguntas de participación.

Resultados:

1. Estandarización de los conocimientos relacionados con las herramientas de pruebas funcionales automatizadas.
2. Se obtiene un material de apoyo para generalizar los conocimientos sobre las pruebas funcionales.

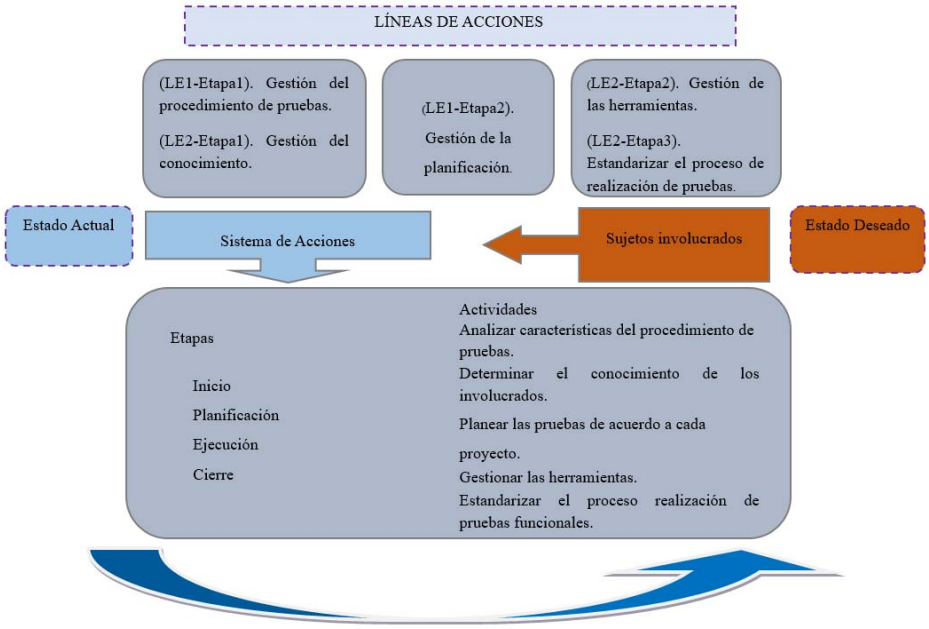

Figura 2 Diseño de la propuesta de automatización de pruebas

Resultados y discusión

A continuación, se muestra una comparabilidad de la situación presentada antes de ser implantada la propuesta de solución y los resultados obtenidos después de haber sido implantada la propuesta de automatización de pruebas funcionales.

Serie Científica de la Universidad de las CienciasInformáticas
Vol. 12, No. 9, Mes Septiembre, 2019
ISSN: 2306-2495| RNPS: 2343
Pág. 112-127
http://publicaciones.uci.cu

Análisis del proceso de obtención de requisitos en los portales web	
Situación ANTES de la implantación de la propuesta de automatización.	**Situación DESPUÉS de la implantación de la propuesta de automatización.**
1. Escaso conocimiento sobre el proceso.	1. Definición de los objetivos y metas a alcanzar.
2. Inexistencia de control de cambios en las pruebas funcionales.	2. Gestión de la planificación de los recursos y el tiempo.
3. No se documentan las pruebas realizadas.	3. Estandarización del proceso realización de pruebas. 4. Repositorio de lecciones aprendidas.

Tabla 2 Comparación entre el antes y el después de la propuesta estratégica

Métricas para pruebas funcionales

Las pruebas al software se realizan con el objetivo de encontrar y documentar los defectos en la calidad del software. Para obtener un software de calidad es necesario medir el proceso (Avances, tamaño, costos, etc.). Las métricas determinan, el avance del software y el cumplimiento de parámetros requeridos. Dentro de las métricas para pruebas de software podemos encontrar las siguientes:

a) Métricas relacionadas con los defectos. Densidad de defectos.
b) Efectividad de las pruebas
c) Métricas en el proceso.

Métricas relacionadas con los defectos

Se concentran en determinar los defectos del software. Brindan soporte para predecir y controlar los defectos esperados, la duración de las pruebas, los recursos dedicados, el tiempo medio entre defectos en distintos momentos de la entrega. Ante la incapacidad de entregar un producto 100% libre de defectos durante el seguimiento del progreso de la fase de pruebas podemos predecir las desviaciones y determinar las acciones correctivas más convenientes para entregar el nivel de calidad tolerado por el cliente en cuanto a cantidad de defectos encontrados en los plazos de tiempos acordados.

La densidad de defectos ofrece una medida sobre la proporción de defectos con respecto a la cantidad de elementos de especificación. Esta métrica permite realizar análisis estadísticos al finalizar las pruebas para valorar la integridad y madurez del software analizado. [Rea, Darío, 2016]

Efectividad de las pruebas

Esta métrica mide la efectividad para encontrar errores. Se dice que un conjunto de pruebas es efectivo si maximiza la cantidad de errores encontrados durante su realización.

Métricas en el proceso

Son métricas relacionadas con el proceso que se determinan a medida que se aplican las pruebas. Pemiten obtener un conjunto de indicadores de proceso que conduzcan a la mejora de los procesos a largo plazo.

Conclusiones

La propuesta de automatización de pruebas funcionales como solución al problema, tiene como característica que parte del presente y se proyecta al estado deseado, se apoya en un conjunto de sistema de acciones que satisfacen las necesidades y exigencias de los objetivos propuestos y que pueden enriquecerse y/o modificarse en la propia práctica a partir de los resultados que se obtengan.

El sistema de acciones propuestas tiene potencialidades para lograr la transformación deseada en cuanto a la ejecución del proceso de realización de pruebas. Para el desarrollo de la propuesta se emplearon buenas prácticas de las técnicas y del enfoque metodológico.

Referencias

Ruiz Tenorio, Roberto. 2010. *Las pruebas de software y su importancia en las organizaciones. Xalapa, Veracruz: Universidad Veracruzana, 2010.*

Pressman, Roger S. (2002)**.** *Ingeniería de Software, un enfoque práctico. 2002.*

IEEE, 1990, Std 610.12.1990. *Revision and redesignation of IEEE Std 792-1983.*

ISO 8402,1995 Gestión de la Calidad y Aseguramiento de la Calidad.

https://www.seleniumhq.org/docs/02_selenium_ide.jsp

Scalone, Fernanda (2006). *Estudio Comparativo de los Modelos y Estándares de Calidad del Software* (Tesis de Maestría). Universidad Tecnológica Nacional Facultad Regional Buenos Aires.

Santos Hernández, Vismar (2009). *La Industria del Software. Estudio a nivel global y América Latina.*

Martínez, Ander (2009). *Apache JMeter. Manual de usuario v1.2.doc.*

Martínez, Ander (2009). *BadBoy. Manual de usuario v1.2.doc.*

Aristegui, José Luis (2010). *Los casos de prueba en la prueba del software.*

Varios Autores (2017). *Metodología de Desarrollo de Software Desoft Versión 3.0.*

https://prezi.com/mvvijozduay5/metricas-para-el-proceso-de-pruebas-de-software/

O'Connor, R., Laporte, C.: *Software Project Manage-ment in Very Small Entities with ISO/IEC 29110, Communica-tions in Computer and Information Science, vol. 301, pp. 330 a 341. Dublin. (2012)*

García, L., Laporte, C., Arteaga, J., Bruggmann, M.: *Im-plementation and Certification of ISO/IEC 29110 in an IT Startup in Peru, Software Quality Professional, vol. 17, no. 2, pp. 16 a 29. Perú. (2015)*

Rea, Darío. Métricas para pruebas funcionales(2016)