Modelo de Pruebas de Regresión Automatizadas en Procesos de Integración Continua en Sistemas Web

Oscar Wile Mamani Alanoca
Postgrado en Informática
Universidad Mayor de San Andrés
La Paz - Bolivia
oswhile@gmail.com

***Resumen*—Este proyecto se enfoca en presentar un modelo de pruebas de regresión automatizadas y su implementación en un sistema web como prototipo, tomando en cuenta herramientas ya creadas para automatizar el proceso de ejecución de las pruebas funcionales durante la fase de desarrollo y evaluación de la calidad del software, para obtener eficiencia en la detección oportuna de defectos en el software de sistemas web. Se evaluaron herramientas y metodologías para automatizar las pruebas de regresión y procedimientos de Integración Continua, para posteriormente implementarlos y finalmente analizar los resultados.**

Palabras clave—Pruebas de software, casos de prueba, automatización de pruebas, Integración Continua

I. INTRODUCCIÓN

En cualquier tipo de proyecto es inevitable la existencia de los riesgos, más aún en proyectos de aplicaciones informáticas, algunos de los más frecuentes pueden ser controlados mediante integración continua y las pruebas funcionales [1].

La integración continua, funciona realizando comprobaciones interconectadas entre diferentes herramientas que miden o controlan un proceso dentro del desarrollo de Software, para conseguir un producto de mejor calidad [1]. En este contexto, la integración continua también nos permite incrementar la satisfacción del cliente y agregar valor al negocio [2]. Por lo tanto, el modelo de Integración Continua es una metodología que ejecuta tareas automáticamente, Fig.1.

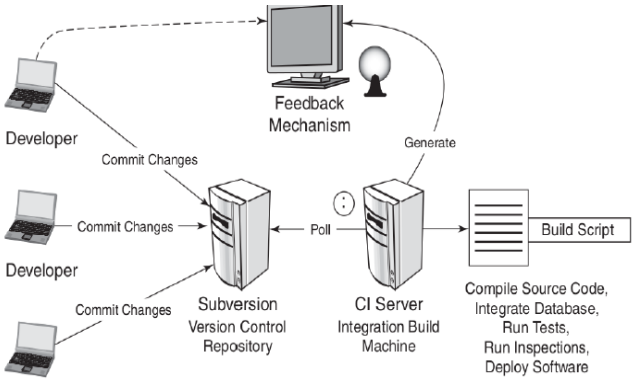

Fig. 1. Modelo de Integración Continua [2]

Las pruebas funcionales nos permiten garantizar en cierta medida la ausencia de fallos en el software, aunque no en un 100% [3], además de garantizar con los requerimientos del usuario final. Para conseguir un producto de software de mejor calidad y con menos recursos, es con la automatización de pruebas funcionales.

La automatización de pruebas funcionales reduce significativamente el esfuerzo dedicado a las pruebas de regresión en productos que se encuentran en continuo mantenimiento [4]. Una buena técnica es preparar un conjunto de pruebas que nos permitirá detectar oportunamente los posibles fallos a su debido tiempo.

En tanto, las pruebas de regresión verifican que el software desarrollado y testeada previamente, sigue trabajando de la forma esperada después de ser actualizado o modificado. A veces la modificación de una parte del software, afecta a otra parte del programa que no ha sido modificada, pero su funcionamiento se puede ver afectado [5].

Muchos de los softwares utilizados por empresas, están construidos como aplicaciones web [6]. Por lo que, las empresas ofrecen servicios a través de Internet utilizando: redes sociales, como Facebook y Twitter. Se estima que, en el año 2017 las ventas del comercio electrónico a nivel mundial supusieron un total 2,3 billones de euros [7], en base al informe de: "Global Ecommerce" [8].

Según estudio realizado por la "Compañía de Pruebas Unitarias" [9] que fue publicado en su portal web, indica lo siguiente: "¿Se podría evitar más de un tercio del costo de los errores de software (más de $ 25 mil millones solo en los Estados Unidos) Si se realizaran mejores pruebas de software en las etapas iniciales de desarrollo? El costo por la incorrecta de pruebas funcionales en los Estados Unidos fue de $ 59.6 mil millones, alrededor del 0.6 por ciento del Producto Interno Bruto de Estados Unidos, según un informe del Instituto Nacional de Estándares y Tecnología". Publicado en 2012 sobre la base de los estudios realizados por los autores Gregory Tassey (2002), Schiller Todd y Michael Ernst (2010).

A lo largo de este tiempo han ido apareciendo metodologías y técnicas que permiten superar las deficiencias en los desarrollos de software. Una de las técnicas del desarrollo de software es la Integración Continua [2]. Y la otra técnica a ser estudiada es la de pruebas de regresión automatizada. Según [10], las pruebas de regresión es un tipo de prueba para repetir una prueba de funcionalidad que ha sido verificada previamente. Es posible que un cambio hecho en una parte del código, ya sea por corrección u otro tipo de cambio, pueda afectar

Para referenciar este artículo (IEEE):
[N] O. Mamani, «Modelo de Pruebas de Regresión Automatizadas en Procesos de Integración Continua en Sistemas Web», *Revista PGI. Investigación, Ciencia y Tecnología en Informática,* nº 8, pp. 143-148, 2020.

accidentalmente el comportamiento de otras partes del código. Las pruebas de regresión se deben realizar en los siguientes niveles de pruebas relacionadas con el Modelo en V, que representa los niveles de desarrollo y su correspondiente etapa de prueba, reflejado en la Fig.2.

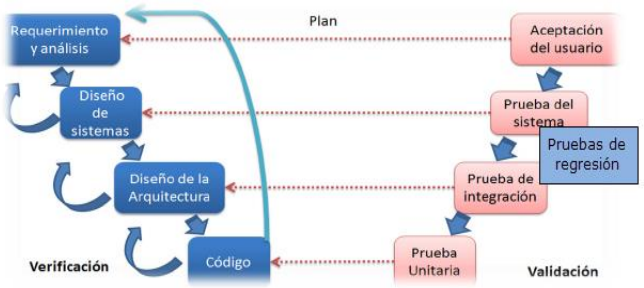

Fig. 2. Modelo en V aplicando pruebas de regresión
Fuente: Elaboración propia en base a [11]

Para el caso de estudio de este proyecto se realizará las pruebas de regresión en los niveles de pruebas de Integración y de Sistemas.

A. *Planteamiento del problema*

Por lo expuesto en los puntos anteriores, existe una carencia de modelos para Pruebas de Regresión. Por lo que, se propone implementar un modelo de Pruebas de Regresión automatizadas en los procesos de Integración Continua donde se utilizan metodologías de desarrollo de software ágil, ya sea SCRUM donde la IC es aplicado en incremento del producto o Programación Extrema donde la IC es aplicado en codificación-pruebas, que permitirá reducir considerablemente el tiempo dedicado a comprobar si las nuevas funcionalidades y los errores solucionados estén funcionando correctamente. En la Fig.3, se refleja el modelo de *Regression Tests* a ser implementado, la misma forma parte de *Run Tests* del modelo de Integración Continua.

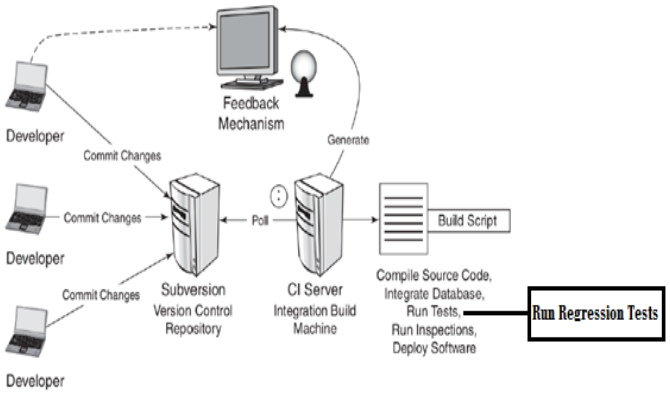

Fig. 3. Modelo de Integración Continua planteado
Fuente: Elaboración propia a partir del modelo original

B. *Formulación del problema*

¿Sera posible implementar un modelo de pruebas de regresión automatizada en procesos de Integración Continua para mejorar la eficiencia en la detección oportuna de defectos en el software de sistemas web?

C. *Planteamiento del objetivo*

Objetivo General. Implementar un modelo de pruebas de regresión automatizadas en procesos de Integración Continua en sistemas web, para mejorar la eficiencia en la detección oportuna en el software.

Objetivos específicos

- Determinar herramientas libres existentes para implementar el modelo de pruebas de regresión automatizadas en procesos de Integración Continua para sistemas web.
- Desarrollar el modelo para Pruebas de Regresión automatizadas para tareas de Test Funcional.
- Analizar los resultados obtenidos de la implementación del modelo de pruebas de regresión automatizadas.

D. *Planteamiento de Hipótesis*

La implementación de un modelo de pruebas de regresión automatizadas en procesos de Integración Continua, mejorará la eficiencia en la detección oportuna de defectos en el software de sistemas web.

E. *Variables*

Dependiente: Pruebas de regresión automatizadas

Independiente: Eficiencia en la detección oportuna de defectos en el software

Operacionalización de variables: Para determinar el porcentaje de afectación a otros módulos se podrá medir con la siguiente fórmula planteada:

$$PMA = MA / TM * 100$$

Donde:
MA = Módulos Afectados
TM = Total Módulos
PMA = Porcentaje (%) Módulos Afectados

Para determinar el porcentaje de defectos encontrados en el software se podrá medir con siguiente fórmula planteada:

$$PDS = DE / CP * 100$$

Donde:
DE = Defectos Encontrados
CP = Total de Casos de Prueba
PDS = Porcentaje (%) de Defectos del Software

La tabla 1 presenta la operacionalización de las variables desde su definición, dimensión, indicador, instrumento de medición para una mayor comprensión de los mismos.

TABLA I. OPERACIONALIZACIÓN DE VARIABLES

Variable	Dimensiones	Indicadores	Instrumento
Variable independiente Pruebas de regresión automatizadas	Pruebas de regresión	Afectación a otros módulos	Observación
Variable dependiente Eficiencia en la detección oportuna de defectos en el software	Defectos en el software	Defectos en el software; Origen del defecto	Observación

Fuente: Elaboración propia

F. *Marco Teórico*

1) *Estado del Arte*

Se han realizado investigaciones relacionadas al presente tema de investigación que fueron expuestos y publicados en años pasados. A continuación, se mencionan los más relevantes:

a) Incorporación de pruebas automáticas sobre transformaciones de modelos en el entorno de integración continua de MOSKitt [12].

Una de las funcionalidades que ofrece la herramienta *MOSKitt* es la transformación de modelos, una parte clave en la Ingeniería Dirigida por Modelos.

La integración continua es un concepto que surge a partir de la idea de realización de construcciones de software diarias. El modelo ideal permite que la construcción y ejecución de las pruebas sea realizada cada vez que el código cambia o es enviado al software de control de versiones.

Así pues, la idea principal de este proyecto es la incorporación de una serie de pruebas automáticas en el proceso de integración continua de la herramienta *MOSKitt*, de tal modo que se asegure que se está construyendo de un modo fiable y funcionalmente correcto.

b) Implementación de herramientas de integración continua para gestión de código fuente en pandora technologies [13].

En la empresa se ha optado por seleccionar las herramientas: Git, SonarQube, Jenkins y Bitbucket por ser herramientas que se adecuan a las necesidades

La implantación ha sido realizada luego de la fase de pruebas en la que se ha terminado de configurar, ajustar y comprobar todo lo necesario de manera que se determinó la adecuación completa a las necesidades del cliente.

Luego de instaladas las herramientas, quedó listo para su uso y accesible desde cualquier equipo de los desarrolladores y analistas de la organización.

Se puede afirmar que se ha tenido un cierto impacto sobre los colaboradores o usuarios de la solución ya que adaptarse a una nueva herramienta no es fácil ni rápido, por ello se realizaron las capacitaciones obteniendo buenos resultados.

Luego de algunos meses de utilización de las herramientas y del nuevo flujo de actualización de código fuente propuesto se van viendo las mejoras en el proceso de gestión de código y por ende en la calidad del producto final

c) La integración continua y su aporte al aseguramiento de la calidad en el ciclo de vida del desarrollo de software [1].

La implementación de Integración Continua representa agilidad en los procesos de validación y verificación del modelo en V para el ciclo de vida del desarrollo de software, ya que permite automatizar acciones que se realizan en muchas compañías de manera manual o por el mismo tiempo que implica en la ejecución del proyecto son obviadas.

Implementar un servidor de integración continua puede resultar un poco complicado al principio. Si se desea montar el modelo completo, existen muchas herramientas y opciones de configuración, lo cual hace difícil tomar decisiones de adquisición, sin embargo, antes de realizar cualquier acción se debe estructurar el proyecto para identificar que partes del proyecto deben priorizarse para la automatización.

d) Suite Open Source de Automatización de Pruebas de Regresión para Entorno Web [5]

A medida que la aplicación se hace más grande y crece en funcionalidades, la tarea de probar la aplicación manualmente se hace más larga y tediosa con cada iteración, llegando a un punto que ya no resulta viable. Sin embargo, este proceso se puede automatizar usando software de automatización.

Por lo tanto, el programa final ha resultado bastante útil y ya está siendo aprovechado actualmente al menos por un equipo de testing (QA) con lo que se puede decir que su usabilidad ha quedado demostrada.

2) Referencia Teórica o Conceptual

a) Gestión de pruebas

Para gestión de pruebas se requiere el conocimiento de planificación, generación de casos de uso, seguimiento de los errores, manejar las configuraciones del sistema. A estas tareas se denomina gestión de pruebas. Para el caso de este trabajo se utilizarán las tareas de gestión de errores, generación de casos de uso, planificación de las pruebas y el manejar configuraciones para establecer que una herramienta automatizada da apoyo en la gestión de pruebas de software [11].

b) Pruebas funcionales

Se denominan pruebas funcionales, a las pruebas de software que tienen por objetivo validar el comportamiento del software cumple o no con sus especificaciones. La prueba funcional toma el punto de vista del usuario. Las funciones son probadas ingresando las entradas y examinando las salidas. La estructura interna del programa raramente es considerada. A este tipo de pruebas se les denomina también pruebas de comportamiento o pruebas de caja negra [14].

c) Automatización de las pruebas funcionales

Es la automatización de las pruebas funcionales sin la intervención del individuo, así de esta manera reducir significativamente el esfuerzo dedicado a las pruebas de regresión en productos que se encuentran en continuo mantenimiento [14].

d) Pruebas de regresión

El objetivo de las pruebas de regresión es la de verificar que el software que ya ha sido desarrollado y testeada previamente, sigue trabajando de la forma esperada después de ser modificado. A veces la modificación de una parte del software, afecta a otra parte del programa que no ha sido modificada, pero su funcionamiento se puede ver afectado o corrompido. La forma más común de realizar las pruebas de regresión es realizando una serie de pasos y comprobando que el resultado es el mismo como anteriormente [5].

e) Integración Continua

La integración continua es una práctica en la cual los miembros de un grupo de desarrollo integran los distintos componentes de un proyecto con una frecuencia especificada. Cada integración se realiza de forma automática con el fin de detectar los errores de integración lo antes posible. Muchos equipos de desarrollo han encontrado que este enfoque reduce significativamente los problemas de integración y permite que los equipos desarrollen software cohesivo más rápido [15], tal como se muestra en la Fig. 1.

f) Modelo en V

El área de aseguramiento de calidad tiene participación desde la concepción del proyecto hasta el cierre del mismo, esta participación se puede observar en un reconocido modelo de desarrollo de software expuesto en 1986 por Paul Rook (Rook, 1986); el Modelo en V que presenta los diferentes niveles del

desarrollo de software y su correspondiente etapa de pruebas [1], tal como muestra la Fig. 2.

g) Niveles de prueba

Son grupos de actividades de prueba que se organizan y gestionan conjuntamente. Los niveles de pruebas están relacionados con otras actividades dentro del ciclo de desarrollo de software [10]:

- *Prueba de Componente*: Que se centra en los componentes que se pueden probar por separado.
- *Prueba de Integración*: que se centra en la interacción entre componentes o sistema.
- *Prueba de Sistema*: se centra en el comportamiento y las capacidades de todo un sistema, así como comportamientos no funcionales.
- *Prueba de Aceptación*: se centra normalmente en el comportamiento y las capacidades de todo un sistema.

II. MÉTODOS

a) Diseño Metodológico

Para el desarrollo del presente trabajo es una investigación científica tecnológica en las ciencias de la computación. El diseño de investigación será Longitudinales de tendencia por tratarse de un comportamiento cambiante de variables en un tiempo determinado.

b) Tipo de Investigación

El presente trabajo de investigación será no experimental ya que el mismo se enfocará el estudio de las realidades de los sistemas desarrollados para el sector financiero en Bolivia.

c) Método de Investigación

Se plantea una hipótesis que se puede analizar deductivamente y posteriormente comprobar experimentalmente. Se utilizará métodos correlaciónales, ya que existirá una relación entre variables.

d) Fases Metodológicas

Fase 1: Recopilación y Análisis de Información

Fase 2: Desarrollo del modelo de Pruebas de Regresión

Fase 3: Ejecutar los casos planteado y realizar un diagnostico

Fase 4: Análisis y Conclusión

e) Técnicas de investigación

Observación. En base la experiencia de trabajo, se podrá realizar las observaciones de los flujos o actividades o procesos para la implementación de TICs en la institución.

f) Universo o Población de Referencia

El presente trabajo de investigación está dirigido a las instituciones, empresas y profesionales dedicadas a desarrollar sistemas en el sector público y privado en la Ciudad de La Paz.

g) Delimitación Geográfica

El presente trabajo de investigación se llevará a cabo, en la Ciudad de La Paz.

h) Delimitación Temporal

El periodo de la información a ser utilizada corresponde al periodo julio a diciembre de 2020.

III. MODELO DE PRUEBAS DE REGRESIÓN

A. Modelo de pruebas de regresión automatizada

Se crea un nuevo modelo de pruebas de regresión automatizadas, donde se describirá paso a paso las actividades a realizar durante el proceso de desarrollo de las pruebas de regresión, en donde se utilizará la herramienta Katalon Studio para aprovechar las funcionales que éste contiene [16].

En la Figura 4, se refleja un modelo de pruebas de regresión automatizada con los siguientes roles y sus actividades que se describen en la Tabla 3.

TABLA II. ROLES Y ACTIVIDADES

Rol	Actividades
Jefe de proyecto	Define requerimientos
Analista Programador	Desarrolla de acuerdo a los nuevos requerimientos; Analiza y levanta los errores; Corrige las observaciones encontradas; Commit al repositorio
Analista de Pruebas	Elabora plan de pruebas; Re - elabora los casos de prueba; Preparación del entorno para las pruebas; Elabora sub - proceso de automatización en Katalon Studio y sub – proceso de creación de proyecto Jenkins
Integración Continua	Versionamiento en GIT; *Run Regresión Test* los casos de pruebas tomadas desde Jenkin; Reporte de errores; Generar Reporte de Katalon Analitic; Subida a Producción
Jefe de Control de Calidad	Evalúa plan de pruebas; Valida plan de pruebas y se procede a pedir ejecución; Conformidad de Control de Calidad

Fuente: Elaboración propia

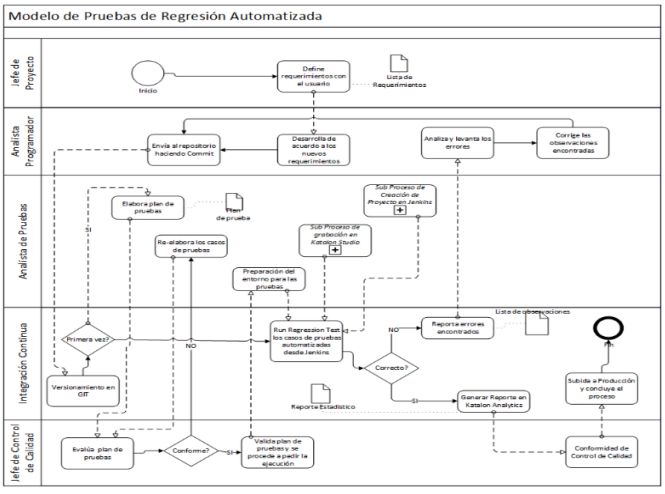

Fig. 4. Modelo de pruebas de regresión
Fuente: Elaboración propia, sobre la base de [16]

B. Subproceso para automatización de pruebas de regresión

La automatización consiste en la grabación de las pruebas funcionales durante la Generación de las pruebas. Con esto se consigue obtener un conjunto de pruebas automatizadas que podrán ser utilizadas cuando sea necesario para repetir las pruebas, es decir guarda un histórico de las pruebas para una reutilización de ser necesaria, ver tabla 4 y figura 5.

TABLA III. ROLES Y ACTIVIDADES DEL SUBPROCESO

Rol	Actividades
Analista de Pruebas	Seleccionar el aplicativo; Prepara los casos de pruebas; Recepción las observaciones; Inicializar la grabación; Verificar la grabación; Repetir el flujo de los casos; Prepara Test Suite (varios hilos); Commit, repositorio GIT
Jefe de Control de Calidad	Valida los casos de prueba; Verificar el motivo del error

Fuente: Elaboración propia

Fig. 5. Subprocesos automatización de pruebas de regresión.
Fuente: Elaboración propia, sobre la base de [16]

IV. RESULTADOS

Una vez elaborado e implementado el Modelo de Pruebas de Regresión Automatizada, se ejecutó y se validó, dando el siguiente resultado para analizar:

TABLA IV. RESULTADO DEL MODELO

Corrida	Test Suite (Modulo)	Cantidad	Estado	Observación
1	Administrador	1	Éxito=1 Error=0	
	Cliente	14	Éxito=0 Error=14	Error de código
2	Administrador	1	Éxito=1 Error=0	
	Cliente	14	Éxito=8 Error=6	Error de validación de fecha
3	Administrador	1	Éxito =1 Error=0	
	Cliente	14	Éxito=9 Error=5	Error de validación de fecha
4	Administrador	1	Éxito=1 Error=0	
	Cliente	14	Éxito=10 Error=4	Error de validación de fecha
5	Administrador	1	Éxito=1 Error=0	
	Cliente	14	Éxito=11 Error=3	Error de validación de fecha
6	Administrador	1	Éxito=1 Error=0	
	Cliente	14	Éxito=12 Error=2	Error de validación de fecha
7	Administrador	1	Éxito=1 Error=0	
	Cliente	14	Éxito=13 Error=1	Error de validación de fecha
8	Administrador	1	Éxito=1 Error=0	
	Cliente	14	Éxito=14 Error=0	
Sub Total	Administrador	8	Éxito=8 Error=0	
	Cliente	112	Éxito=77 Error=35	
Total	**120**	**Éxito=85 Error=35**		

Fuente: Elaboración propia

En base a los resultados mencionados en el punto anterior, se aplica a la formula propuesto en la operacionalización de variables, ver tabla 6 y 7.

TABLA V. PORCENTAJE DE MÓDULOS AFECTADO

TM	MA	Variable Independiente	Interpretación
8	0	**PMA=MA/TM*100 = 0 / 8 =0%** Donde, MA=Módulos Afectados TM=Total Módulos PMA= (%) Módulos Afectados	No hubo afectación a otros módulos

Fuente: Elaboración propia

TABLA VI. PORCENTAJE DE DEFECTOS ENCONTRADOS

TM	MA	Variable Dependiente	Interpretación
120	35	PDS=DE/CP*100 = 35/120*100=29.16% Donde, DS=Defectos Encontrados CP=Total Casos de Prueba PDS= (%) de Defectos del Software	Porcentaje de defectos encontrados en un 29.16%

Fuente: Elaboración propia

Por lo tanto, con la implementación de un modelo de pruebas de regresión automatizadas en procesos de Integración Continua en el desarrollo de sistemas web que utilizan metodologías ágiles, mejoró en un 29.16% en la detección oportuna de defectos en el software. Aplicado a un Sistema Web Agenda para Entidades Financieras, donde, es una plataforma web que está orientado a la gestión clientes: registrar o recuperar datos del cliente, ver su crédito, ver su flujo de caja y su rendimiento.

V. CONCLUSIONES

Con la implementación del Modelo de Pruebas de Regresión Automatizadas en procesos de Integración Continua, mejoró en un 29.16% la eficiencia en la detección oportuna de defectos, así mismo se garantiza la disminución del tiempo de pruebas de regresión y procesos manuales con la automatización, así como la optimización de los recursos a utilizar.

Para llegar a este punto del proyecto, se realizó el cumplimiento de los objetivos planteados: como la realización de un análisis de las diferentes herramientas libres para implementar la Integración Continua, elaboración del modelo de pruebas de regresión automatizadas, para luego implementarlos y finalmente analizar los resultados, donde se logró cumplir con los objetivos y demostrar la hipótesis planteado.

Recomendaciones

- Tener en cuenta que el proyecto presentado está orientado a pequeñas y grandes empresas o instituciones privadas o públicas dedicadas desarrollo de softwares.
- La presente tesis está orientada a los proyectos de nuevos desarrollos, pero, así también en las etapas de mantenimientos del software.

REFERENCIAS

[1] A. M. García Orozco, «La integración continua y su aporte al aseguramiento de la calidad en el ciclo de vida del desarrollo de software,» Bogotá, 2015.

[2] O. M. Berta Hinostroza, «Incorporación de la integración continua en el desarrollo de software: Caso de estudio: Organismo supervisor de la inversión en energía y minería,» Piura, 2011.

[3] S. Fernandez, «Un marco de trabajo para el desarrollo y ejecución de pruebas automatizadas aplicadas al front-end de una aplicación web,» Valencia, 2016.

[4] I. Esmite, M. Farías, N. Farias y B. Pérez, «Automatización y Gestión de las Pruebas Funcionales Usando Herramientas Open Source,» Montevideo, 2007.

[5] R. Koszewski, «suite open source de automatización de pruebas de regresión para entorno web,» Valencia, 2016.

[6] M. Giménez y A. Espinola, «Automatización de Pruebas para Interfaces deAplicaciones Web,» Asuncion, 2017.

[7] A. T. Vega Llobell, «Pruebas funcionales automatizadas para aplicaciones Web: Usando Selenium para aplicar pruebas de regresión automatizadas,» Valencia, 2018.

[8] G. E. S. a. I. G. Trends, «https://www.shopify.com/enterprise/global-ecommerce-statistics,» 15 04 2018. [En línea]. [Último acceso: 15 04 2018].

[9] T. U. T. Company, «https://www.typemock.com/what-is-the-cost-of-avoiding-unit-testing-and-the-cost-of-software-bugs/,» 8 5 2012. [En línea]. [Último acceso: 10 9 2020].

[10] I. S. T. Q. B. ISQTB, «Programa de Estudio Nivel Basico,» 2018.

[11] E. J. Chinarro Morales, «Definición e implementación del proceso de pruebas de software basado en la NTP-ISO/IEC 12207:2016 aplicado a una empresa consultora de software,» Lima, 2019.

[12] J. Badenas Martínez, «Incorporación de pruebas automáticas sobre transformaciones de modelos en el entorno de integración continua de MOSKitt,» Valencia, 2019.

[13] M. L. Guido Saravia, «Implementación de herramientas de integración continua para gestión de código fuente en Pandora Technologies,» Lima, 2018.

[14] D. A. Esteve Ambrosio, «Implantación de un proceso de automatización de pruebas para una aplicación software,» Valencia, 2015.

[15] A. Salamon, P. Maller, A. Boggio, N. Mira, S. Perez y F. Coenda, «La Integración Continua Aplicada en el Desarrollo de Software en el Ámbito Científico – Técnico,» Cordoba.

[16] E. Chinarro Morales, M. E. Ruiz Rivera y E. Ruiz Lizama, «Desarrollo de un modelo de pruebas funcionales de software basada en la herramienta Selenium,» 2017.

Breve CV del autor

Oscar Wile Mamani Alanoca es Licenciado en Informática por la Universidad Mayor de San Simón (Cochabamba, 2009). Actualmente realiza la Maestría en Alta Gerencia en Tecnologías de la Información y las Comunicaciones e Innovación MAG-TIC (2015-2016) en el Postgrado en Informática de la Universidad Mayor de San Andrés.
Ejerce profesionalmente como desarrollador de sistemas informáticos en Banco Central de Bolivia. Anteriormente en el Ministerio de Economía y Finanzas Publicas.
Sus intereses investigativos son los datos abiertos, inteligencia de negocios y automatización de pruebas funcionales.
Email: oswhile@gmail.com.