PROPUESTA DE AUTOMATIZACIÓN DE CASOS DE PRUEBA PARA ASEGURAMIENTO DE CALIDAD EN EL DESARROLLO DE SOFTWARE

Trabajo de investigación realizado por:

ANDRÉS ALBERTO RUEDA PATIÑO

HÉCTOR FABIO CRUZ MOSQUERA

JAIRO ALEJANDRO LONDOÑO ROJAS

Para optar por el título de Especialista En Gerencia de Proyectos

Directora de Investigación:

SANDRA MARIA QUINTERO CORREA

Corporación Universitaria Minuto De Dios

Facultad De Educación

Especialización En Gerencia De Proyectos

Bello – Antioquía

2016

Contenido

Índice de tablas

Índice de figuras

Índice de anexos

Introducción

La automatización de pruebas de software es una de las opciones más fascinantes para enfrentar un equilibrio habitual de entregas al cliente sin comprometer la calidad del producto software; esto se hace más necesario cuando ya se tiene un producto en un ambiente de producción y con una gran cantidad de usuarios que pueden verse afectados por un fallo. Existen diversos tipos y niveles de pruebas, entre ellos las pruebas de aceptación las cuales tienen especial importancia de cara a la conformidad del comportamiento externo del producto según las expectativas del cliente.

Los analistas de pruebas en repetidas ocasiones pueden llegar a convertirse en cuellos de botella durante la etapa de pruebas de software y a pesar de esta situación se les demanda probar más y más código en menos tiempo, ya sea por la liberación de diferentes versiones del código fuente o por las nuevas funcionalidades que se deben probar en conjunto con las ya existentes, implicando que el ciclo de pruebas se vuelva cada vez más lento e ineficiente.

Lo que se pretende con esta investigación no es dejar a los analistas de prueba sin trabajo sino enfocarlos en la realización de pruebas más complejas que sean más difíciles de automatizar. Se pretende mostrar los beneficios que se puede obtener con la implementación de la ejecución automática de los casos de prueba sobre el software desarrollado, con ayuda de herramientas libres que se puedan acoplar a la propuesta a plantear. Esto último se apoyará en el análisis que se realice de la documentación sobre el proceso de pruebas actual y la documentación generada por la ejecución de las pruebas que han realizado sobre el software desarrollado.

El análisis se realizó en una empresa desarrolladora de software que cuenta con procesos definidos y documentados para el desarrollo de software y la ejecución de pruebas de calidad sobre dicho software. Esa ejecución de pruebas de calidad es un proceso que se ejecuta manualmente haciendo que se tome mayor tiempo y que exista una mayor posibilidad de que se cometan errores durante su ejecución. Además del análisis realizado, para el desarrollo de la propuesta de ejecución de casos de pruebas automáticas, se realizó una simulación de ejecución automática de casos reales buscando involucrar las herramientas que fueron identificadas inicialmente.

En la realización de pruebas un tema crucial es el tiempo y con las pruebas automatizadas se puede lograr una disminución de éste al automatizar tareas que permitan la liberación de carga a los analistas de prueba y con esta investigación se busca ayudar en ese proceso de automatización.

La automatización puede prometer grandes beneficios siempre y cuando recordemos que “pruebas” no solo significa probar lo mismo una y otra vez hasta el agotamiento: esto conlleva a evaluar los resultados con respecto a las diferentes entradas para así ver que pruebas dar valor agregado y saber que herramienta a emplear ya que esta nunca harán el trabajo solas sin un buen entendimiento.

1. Planteamiento del Problema

1.1. Descripción del problema

En la industria del desarrollo de software, la calidad es un tema que tiene mucha relevancia; desde el punto de vista del cliente, es importante cumplir con los niveles mínimos de aceptación que un producto debe tener para cumplir sus necesidades, así mismo, la calidad tiene una gran influencia en el grado de aceptación que tiene un desarrollo, ya sea página web, aplicación móvil, aplicación de escritorio u otro.

Desde el punto de vista contractual, la calidad de un software es tan importante que podría llevar a una gran pérdida económica por parte de la casa desarrolladora, en el caso de firmar contratos donde existen cláusulas de calidad mínima. En este sentido, un número determinado de fallas puede desembocar en una multa e incluso en la pérdida total del contrato, que puede generar implicaciones negativas para la empresa, como mala reputación, problemas legales, entre otros.

Ante el crecimiento de pymes enfocadas en el desarrollo del software y en la búsqueda de un fortalecimiento de este mercado en el país, en el año 2007 nació la Unión Temporal Red Colombiana de Calidad del Software (UT_RCCS), que surgió como respuesta a la convocatoria del programa "Apoyo al Fortalecimiento de la Capacidad Nacional del Software" y que buscaba impulsar las exportaciones de software en el país, incentivando a las empresas de este sector para aumentar la productividad y competitividad frente al mercado mundial.

A partir de esta investigación, realizada por la UT_RCCS, se evidencia que las pyme tenían en su momento cierta resistencia a implementar cambios, en este caso, un modelo de

madurez de sus procesos; sin embargo, el panorama en la actualidad ha cambiado, lo que es evidente desde lo planteado por Restrepo (2015), Presidenta Ejecutiva de Fedesoft: “La industria de software de Colombia ha sido uno de esos sectores altamente atractivos para los compradores e inversionistas extranjeros, los cuales cada vez buscan más a nuestro país como potencial tecnológico, guiados por la calidad de los desarrollos y productos, el servicio al cliente y una demanda para software que ha crecido exponencialmente en los últimos años”; igualmente cuando afirma que “Estamos de moda, demostrándole al mundo que Colombia no es solo café y banano, que somos un país innovador, con talento humano capaz de entregar soluciones de alta calidad y que actuamos de acuerdo a los estándares de modelos de calidad internacionales como CMMI (CapabilityMaturityModelIntegration), TSP (Team Software Process), PSP (Personal Software Process), ISO 9001, ISO 27001, entre otros”.

Teniendo en cuenta lo planteado por Restrepo (2015), Colombia no es solo exportador de productos agrícolas y mineros, gracias a la implementación de modelos y certificaciones de calidad, junto con el talento humano en el desarrollo de software, le han dado un posicionamiento global para ser tenido en cuenta por parte de inversionistas extranjeros. De acuerdo con el informe del Software Engineering Institute (SEI) Colombia es el primer país en número de empresas con valoración CMMI entre los niveles III y V, por encima de países como Brasil, Chile y Ecuador.

Sin embargo, tener un modelo o tener certificaciones ISO no garantiza un software de calidad, estos modelos y/o certificaciones establecen qué se debe hacer, pero no el cómo; parten del principio: que al tener un control sobre las actividades realizadas en el proceso se puede obtener un producto de calidad; por ejemplo, es común ver empresas, como las grandes casas de desarrollo con aplicativos propios o pequeñas empresas que prestan el servicio de desarrollo a la medida, que teniendo certificaciones y/o modelos internacionales han estado

poco preocupadas por la calidad de sus desarrollos o su aseguramiento de calidad del producto no es el mejor, pues en algunas ocasiones se conforman con las pruebas unitarias básicas que el desarrollador haya realizado. En este caso, se está cumpliendo con el modelo al tener unas actividades de control y una trazabilidad del proceso, pero no se puede garantizar que con un conjunto de pruebas unitarias se tenga un alto grado de confianza del desarrollo. Otro caso que se presenta es la realización de pruebas de forma manual, que aunque no significa que este mal podría ser optimizado al tener un proceso que ejecute algunos de los casos de prueba de manera automática. Esto podría convertirse en un ahorro en el mediano y largo plazo en el caso de tener desarrollos que sufren cambios en su implementación pero no en su funcionalidad.

Afortunadamente, con el uso masivo de la internet y la generación de nuevas técnicas, metodologías, herramientas e ingeniería esa situación ha cambiado un poco, ya que actualmente ha tomado fuerza la implementación de prácticas y metodologías de desarrollo que hacen hincapié en la calidad del producto de software en desarrollo. Algunas de esas metodologías empezaron a afianzarse en el mercado desde apenas unos 5 años atrás, aunque la mayoría fueron creadas en la década de los 90 y que en algunos países ya tienen una amplia trayectoria.

Dentro de las prácticas y metodologías tenemos: TDD, se enfoca en la programación de los casos de prueba antes del desarrollo de la funcionalidad requerida; XP, se caracteriza por el trabajo en parejas para una codificación más limpia; SCRUM, establece el trabajo en equipos multidisciplinario y se ayuda con prácticas de TDD y algunas de XP; RUP, establece el trabajo por fases donde una de estas es Pruebas; esta última metodología ha tenido una mayor trascendencia en el ámbito Colombiano. Adicional a estas metodologías, se cuenta con herramientas que ayudan a la codificación de casos de pruebas, así como su ejecución de

forma programada para lograr tener una automatización que permita el ahorro en tiempo y dinero a largo plazo.

1.2. Formulación del problema

Actualmente la empresa Axede S.A. tiene una madurez CMMI Nivel 3 en desarrollo, presta el servicio de outsourcing de desarrollo de software al cliente EPM. Dentro de su proceso de desarrollo de software existe una etapa de pruebas la cual es ejecutada por personal de Aseguramiento de Calidad (Quality Assurance, QA por sus siglas en inglés) quienes realizan las pruebas de forma manual a cada artefacto de software que es liberado por los diferentes ingenieros de desarrollo, invirtiendo más tiempo y costos que podrían ser optimizados mediante la automatización de algunos casos de prueba que se podrían reutilizar en los reprocesos o en las adiciones de funcionalidad sobre el software al cual se le da soporte.

Por todo lo anterior, la presente investigación pretende dar respuesta a la pregunta: ¿Cómo diseñar una propuesta para la ejecución de aseguramiento y control de calidad de forma automatizada sobre las piezas de código fuente obtenidas durante el desarrollo de software, con el fin de reducir tiempo en su ejecución y logrando una mejor rentabilidad en la empresa Axede S.A?

2. Justificación

Con el objetivo de buscar mejoras al proceso de desarrollo de software realizado en Axede S.A. y teniendo en cuenta que en la actualidad no se tiene un procedimiento de análisis, diseño y selección de casos de prueba para ser automatizados, la presente investigación busca definir un esquema de trabajo enfocado en el aseguramiento y control de calidad, con el fin de reducir el tiempo en la detección de incidentes en los productos obtenidos durante el desarrollo del software, por medio de la automatización de casos de prueba.

En palabras del consultor Johnson (2011), la automatización de pruebas ha sido una alternativa atractiva para las pruebas manuales costosas que consumen mucho tiempo y en ocasiones son inconsistentes; esta situación es evidente cuando se asume que el objetivo de los casos de prueba automatizados es minimizar tanto como sea posible el esfuerzo dedicado a garantizar el aseguramiento de calidad del software, lo cual se logra con la creación de scripts que sean capaces de ejecutar la mayoría de los casos de prueba, con la posibilidad de su reutilización.

Desde lo anterior, es importante tener presente: estadísticas de cada una de las ejecuciones, un reporte de éstas y una comparación de los resultados entre ejecuciones. Para conseguir las estadísticas es importante apoyarse en frameworks y/o herramientas que ayuden a ese propósito; lo que es posible si se cuenta con una variedad de ellas, ya sean comerciales o libres.

A pesar de las grandes ventajas que puede suministrar la literatura sobre las pruebas automáticas, estas no siempre solucionan los problemas que se tienen en la detección de no conformidades en una etapa temprana. Sin embargo, con una implementación apropiada, se pueden obtener diferentes beneficios, como son: mejor organización de las pruebas, realización de un mayor número de pruebas, mejoras en la comunicación con el equipo de desarrollo, estabilización temprana del código, habilitación de pruebas de regresión, mayor confiabilidad en los resultados y reutilización, reflejada en ahorro de tiempo.

La automatización de pruebas se debe considerar como un proyecto de desarrollo más y como proyecto tiene unas etapas que deben ser implementadas, requiriendo un conjunto de recursos; por lo cual es importante tener en cuenta la estrategia que tiene Axede S.A. frente a la calidad de los desarrollos que se ejecutan en la empresa, porque a pesar de llegar a tener un

resultado positivo en el proyecto que se está emprendiendo, esto no garantiza que se vuelva una política de la empresa el tener la automatización de pruebas. En este punto, otro objetivo de la investigación es hacer un análisis del proceso que se tiene actualmente con el fin de identificar cada una de las partes que intervienen, las actividades realizadas y los resultados obtenidos y a partir de este análisis se podrán rescatar esas acciones que generan valor al aseguramiento de la calidad y así establecer posibles mejoras que ayuden al fortalecimiento del proceso.

Finalmente, dentro de los objetivos se considera hacer una simulación en la que se ponga a prueba el proceso obtenido y validar las posibles ventajas que éste podría generar para el aseguramiento de las pruebas y su rentabilidad. Para el cumplimiento de este objetivo se contará con el uso de herramientas libres, las cuales deberán ser analizadas para tener un buen conocimiento de algunas de ellas y así seleccionar las que mejor se adapten al ambiente de desarrollo que se utiliza. Las herramientas que al final se seleccionen deben ayudar en la creación, ejecución y obtención de estadísticas de las ejecuciones de los casos. A partir de lo anterior, se generará un banco de conocimiento que puede ser utilizado tanto en ejecuciones futuras como en la instrucción de nuevos colaboradores de la compañía.

3. Objetivos del Estudio

3.1. Objetivo General

Diseñar una propuesta para la ejecución de aseguramiento y control de calidad de forma automatizada sobre las piezas de código fuente obtenidas durante el desarrollo de software, con el fin de reducir tiempo en su ejecución y logrando una mejor rentabilidad en la empresa Axede S.A.

3.2. Objetivos Específicos

Analizar el proceso de aseguramiento y control de calidad del desarrollo de software que tiene actualmente la empresa Axede S.A. para el hallazgo de posibles mejoras mediante la automatización de casos de prueba.

Identificar algunas herramientas útiles para la automatización de casos de prueba sobre el desarrollo de software en Axede S.A.

Plantear una propuesta de ejecución de casos de prueba manuales y automatizados en los desarrollos de software de Axede SA para que sea más eficiente el proceso de control de calidad.

4. Marco Referencial

4.1. Antecedentes

Hoy en día existen empresas de desarrollo de software donde se ha aplicado diferentes maneras de automatizar los procesos de aseguramiento y control de calidad sobre los desarrollos que ellos mismos realizan o desarrollos de otras empresas. Estas empresas han evidenciado que la automatización trae grandes beneficios al disminuir los tiempos de ejecución y por consiguiente los costos que conlleva este proceso.

Lopez-Mancisidor (2003) de IBM nos plantea que es de 100 a 1000 veces más costoso encontrar y reparar los problemas del software después del desarrollo completo. Es por esto que recomienda la automatización de pruebas para la identificación de posibles inconsistencias en una etapa temprana del desarrollo y la cual se convierte en una inversión a mediano-largo plazo produciendo mejores resultados. En su presentación nos expone sobre las ventajas de utilizar Rational XDE Tester, que aunque está enfocada en un producto propio, deja ver casos exitosos que se tuvieron por la automatización.

En la tabla 1 se visualiza el beneficio en término de retorno de inversión para varias empresas

Otro caso exitoso de la aplicación de la automatización de casos de prueba se ve en la implementación realizada en la empresa BASALT, esa implementación estuvo enfocada en pruebas de GUI (Graphical User Interfaces, en español Interfaz Gráfica de Usuario) sobre el sistema SWECCIS. BASALT es una empresa que realiza el desarrollo de sistemas que son usados para el mando y control dentro de las fuerzas armadas suecas.

En BASALT, las pruebas realizadas para los desarrollos son ejecutados de forma manual y deben garantizar que esos desarrollos sean de buena calidad debido al compromiso que se tiene con la seguridad nacional, es por esto que el proceso de pruebas es crucial. Para ese proceso se ejecutan muchos casos de pruebas que consumen gran cantidad de tiempo, el cual es utilizado en la planeación, la preparación de los ambientes de prueba, la ejecución de los casos de prueba y documentación. Debido a esto se planteó la opción de realizar la

automatización que tenía como meta mejorar la eficiencia, la calidad y el costo durante el proceso de pruebas.

Tabla 1: *Beneficios de pruebas automatizadas por empresas*

Empresa	**Beneficio**
Merrill Lynch	300% en incremento de la productividad
Ericsson	80% menos errores; 100% incremento de la productividad
Lockheed Martin Canada	US$409.000 Beneficio Neto; 222% ROI
Credence Systems Corporation	1200% incremento en la productividad del desarrollador
	90% Reducción en errores en el Backlog
Covarity	25% de reducción en el ciclo de desarrollo
Choice Hotels International	1440% ROI; 9.5M Beneficio Neto
Information Builders, In	96% en incremento de la productividad
Alltel	66% reducción del tiempo del ciclo de desarrollo
	Ejecución de pruebas manuales de 2 meses reducido a 2 días

Fuente: https://www.ati.es/IMG/pdf/IBM.pdf

Después de ejecutar el proceso para 26 casos de prueba que se tuvieron en cuenta para la investigación, llegaron a la conclusión que la implementación inicial de los casos de prueba de forma automática de gastaba alrededor de 30 horas mientras que una ejecución manual completa toma entre 7 y 9 horas; sin embargo en la 3ª y 4ª ejecución de los casos de prueba tanto manual como automático ya se tenía un retorno de inversión del 90% en términos de tiempo ahorrado. Cabe mencionar que una ejecución automatizada se puede realizar muchas veces de la misma manera, sin excluir algún paso por error, como podría suceder con una ejecución manual.

En este caso fue muy bueno la implementación de la automatización de los casos de pruebas. Como trabajo a futuro, se plantea la posible ejecución en paralelo, en lugar de hacer cada uno de los casos de forma secuencial.

En el ámbito nacional no se encontró información sobre casos de implementación de automatización de casos de prueba sobre desarrollos de software.

4.2. Marco Teórico

4.2.1. Metodologías de Desarrollo de Software.

4.2.1.1. XP: Extreme Programing (Programación extrema).

La programación extrema es una metodología ágil dentro de la ingeniería del software que empezó a ser utilizado a finales de la década de 1980. Fue formulado por el ingeniero de software estadounidense Kent Beck quien escribió el primer libro sobre la materia "Extreme Programming Explained: Embrace Change" publicado en 1999.

XP se centra en potenciar las relaciones interpersonales dentro del equipo desarrollador siendo esto la clave del éxito del desarrollo de software; promueve el trabajo en equipo, generando aprendizaje continuo y propiciando un buen clima de trabajo; busca una comunicación fluida entre todos los participantes del equipo de desarrollo y el cliente teniendo retroalimentación continua; y es adecuada para proyectos con requisitos poco definidos y muy cambiantes, y donde existe un riesgo técnico muy alto.

XP utiliza un enfoque orientado a objetos como su paradigma de desarrollo preferido y abarca un conjunto de reglas y prácticas que ocurren en el contexto de cuatro actividades del marco de trabajo: planeación, diseño, codificación y pruebas:

Planeación: esta actividad comienza creando una serie de historias o también llamadas historias de usuario que describen las características y funcionalidades requeridas para el software que se construirá, las cuales son redactadas por el cliente en tarjetas indexadas asignándoles un valor correspondiente a la prioridad basándose en los valores generales del negocio.

Los miembros del equipo XP evalúan cada historia y le asigna un costo que se mide en semanas de desarrollo (1 a 3 semanas); si la historia requiere más de tres semanas de desarrollo, se le solicita al cliente que sea dividida en historias menores asignándoles un nuevo valor y costo.

Los clientes y el equipo XP trabajan juntos para determinar cómo agrupar las historias hacia el próximo lanzamiento para que el equipo XP las desarrolle. Una vez acordado el compromiso primordial de que historias de usuario se incluirán y fecha de entrega, el equipo XP ordena las historias que serán desarrolladas teniendo en cuenta la prioridad, el costo y/o el riesgo que tienen dentro del proyecto.

Acorde avanza el trabajo de desarrollo, el cliente puede adicionar historias, cambiar el valor de la historia existente, dividir historias o eliminarlas, haciendo que el equipo XP examine de nuevo los lanzamientos restantes y ajuste sus planes de acuerdo a ello.

Diseño: en esta actividad XP enmarca de manera precisa el principio MS (Mantenerlo Simple). Continuamente se prefiere un diseño simple respecto de una presentación compleja. Además el diseño ofrece una orientación de la implementación de una historia como está escrita, ni más ni menos. Se desaprueba el diseño de funcionalidad extra.

XP patrocina la refactorización de código (Refactoring) una técnica de construcción que Fowler (2006) describe de esta manera: “El refactoring es el proceso de cambiar un sistema de software de tal manera que no altere el comportamiento externo del código y que mejore la estructura interna. Es una manera disciplinada de limpiar el código y modificar/simplificar el diseño interno. Lo que minimiza las oportunidades de introducir errores. En esencia el refactoring es la mejora de diseño del código después de que se ha escrito.”

Codificación: XP sugiere que después de diseñar las historias y realizar el trabajo de diseño preliminar el equipo no debe dirigirse hacia la codificación, sino que debe crear una serie de pruebas de unidad que ejerciten cada una de las historias que vayan a incorporar en un lanzamiento.

Una consideración clave durante la actividad de codificación es la programación en pareja. XP aconseja que dos personas trabajen juntos en un mismo equipo de cómputo para crear el código de una historia, esto facilita un mecanismo para la solución de problemas en tiempo real (dos cabezas piensan mejor que una) y el aseguramiento de la calidad en las mismas condiciones. También estimula a los desarrolladores para que se mantengan concentrados en el problema que se tiene a la mano. En la práctica cada persona tiene un papel sutilmente distinto, una persona puede pensar en los detalles de la codificación en un bloque particular del diseño, mientras que la otra garantiza que se sigan los estándares de codificación.

Pruebas: las pruebas de unidad se deben crear bajo un marco de trabajo que permita ser automatizadas para que puedan ejecutarse de manera más fácil y numerosas ocasiones. Esto favorece a la estrategia de prueba de regresión.

Cuando las unidades de prueba se estructuran en un conjunto universal de pruebas, las pruebas de validación e integración del sistema se pueden realizar a diario, proporcionando al equipo de XP un aviso continuo del progreso y así mismo prendiendo alarmas de emergencia anticipadas si las cosas salen mal, Wells menciona “Resolver problemas pequeños cada pocas horas toma menos tiempo que resolver problemas enormes justo antes de la fecha límite”.

Las ventajas de esta metodología radican en que es una programación organizada, menor tasa de errores y satisfacción del desarrollador.

“XP es una disciplina de desarrollo de software que se basa en los valores de la simplicidad, comunicación, retroalimentación y audacia” R. Jeffries.

4.2.1.2. TDD: Test Driven Development (Desarrollo guiado por pruebas).

Es una práctica de ingeniería de software que involucra otras dos prácticas: escribir las pruebas primero (Test First Development) y refactorización (Refactoring). Para escribir las pruebas generalmente se utilizan las pruebas unitarias (unit test en inglés).

TDD se rige bajo dos reglas básicas: se codifica una nueva funcionalidad solamente cuando un caso de prueba automático haya fallado y se debe eliminar cualquier código repetido que se encuentre. Esto quiere decir que se debe hacer un caso de prueba y validar que falle, realizar la mínima cantidad de código para que el caso de prueba anterior pase y finalmente eliminar cualquier código repetido.

Las dos reglas anteriores llevan a reglas más complejas, como son: se hace un desarrollo orgánico, es decir, la codificación de las funcionalidades cada vez va creciendo como si fuera un proceso natural; es mejor escribir tus propios casos de prueba y así no esperar un tiempo a que otra persona lo realice; los ambientes de desarrollo deben responder de forma rápida a pequeños cambios; y los diseños serán altamente cohesivos, eliminando componentes duplicados para hacer más fáciles las pruebas y el correspondiente mantenimiento.

Al implementar TDD es necesario aprender a construir casos de prueba unitarios que sean efectivos y así no tome mucho tiempo el desarrollo de la funcionalidad por la repetición de los pasos de TDD.

De acuerdo con el libro “Diseño Ágil con TDD”, se establecen las siguientes ventajas de TDD:

- La calidad del software aumenta
- Conseguimos código altamente reutilizable.
- El trabajo en equipo se hace más fácil, une a las personas.
- Nos permite confiar en nuestros compañeros aunque tengan menos experiencia.
- Multiplica la comunicación entre los miembros del equipo.
- Las personas encargadas de la garantía de calidad adquieren un rol más inteligente e interesante.
- Escribir el ejemplo (test) antes que el código nos obliga a escribir el mínimo de funcionalidad necesaria, evitando sobrediseñar.

- Cuando revisamos un proyecto desarrollado mediante TDD, nos damos cuenta de que los casos de prueba son la mejor documentación técnica que podemos consultar a la hora de entender qué misión cumple cada pieza del puzzle.

4.2.1.3. SCRUM.

Scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos.

En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, 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.

Scrum también se utiliza para resolver situaciones en que no se está entregando al cliente lo que necesita, cuando las entregas se alargan demasiado, los costes se disparan o la calidad no es aceptable, cuando se necesita capacidad de reacción ante la competencia, cuando la moral de los equipos es baja y la rotación alta, cuando es necesario identificar y solucionar ineficiencias sistemáticamente o cuando se quiere trabajar utilizando un proceso especializado en el desarrollo de producto.

En Scrum se definen unos roles y un conjunto de ceremonias. A continuación se listan cada uno de ellos:

Roles

Scrum Team (El equipo Scrum)

En Scrum, el equipo se focaliza en construir software de calidad. La gestión de un proyecto Scrum se centra en definir cuáles son las características que debe tener el producto a construir (qué construir, qué no y en qué orden) y en vencer cualquier obstáculo que pudiera entorpecer la tarea del equipo de desarrollo. El equipo Scrum está formado por los siguientes roles:

Product Owner: Representante del cliente que usa el software.

Product Owner Proxy: Es el PO del lado de la empresa desarrolladora que apoya las actividades del PO del cliente, teniendo en cuenta que las responsabilidades del producto y las decisiones de importancia estratégica siempre recaen en el PO del cliente.

Scrum Master: Líder facilitador, mentor y coach que acompaña al equipo de trabajo en su día a día y garantiza que la organización, los stakeholders y el equipo Scrum, utilicen Scrum de forma correcta ajustándose a la teoría, prácticas y reglas de Scrum.

Development Team: Equipo de desarrollo conformado de 3 a 9 profesionales con los conocimientos técnicos necesarios y que desarrollan el proyecto de manera conjunta llevando a cabo las historias a las que se comprometen al inicio de cada sprint para entregar un producto de valor.

El equipo de desarrollo es multidisciplinario, es decir, no existen especialistas exclusivos sino individuos generalistas con capacidades especiales, lo que se espera de un miembro del equipo de desarrollo es que no solo realice las tareas en las cuales se especializa sino también todo lo que esté a su alcance para colaborar con el éxito del equipo.

El equipo de desarrollo es auto-organizado. Esto significa que no existe un líder externo que asigne las tareas ni que determine la forma en la que serán resueltos los problemas. Es el mismo equipo quien determina la forma en que realizará el trabajo y cómo resolverá cada problemática que se presente. La contención de esta auto-organización está dada por los

objetivos a cumplir: transformar las funcionalidades comprometidas en software funcionando y con calidad productiva, o en otras palabras, producir un incremento funcional potencialmente entregable.

En los procesos ágiles existen ceremonias o eventos predefinidos con el fin de crear regularidad y minimizar la necesidad de reuniones no definidas. Todas las ceremonias son bloques de tiempo (time-boxes), de tal modo que todos tienen una duración máxima. Una vez que comienza un Sprint, su duración es fija y no puede acortarse o alargarse. Las demás ceremonias pueden terminar siempre que se alcance el objetivo de la ceremonia, asegurando que se emplee una cantidad apropiada de tiempo sin permitir desperdicio en el proceso.

Además del propio Sprint, que es un contenedor del resto de las ceremonias, cada una de las ceremonias constituye una oportunidad formal para la inspección y adaptación de algún aspecto. Estas ceremonias están diseñadas específicamente para habilitar las vitales de transparencia e inspección. La falta de alguno de estos eventos da como resultado una reducción de la transparencia y constituye una oportunidad perdida para inspeccionar y adaptarse.

Ceremonias

Planning Meeting Sprint (Reunión de Planificación del Sprint): Pretende identificar el trabajo a realizar durante el Sprint, este plan se crea mediante el trabajo colaborativo del Team Scrum completo. Tiene por objetivo que todo el equipo conozca las razones y los detalles con el nivel necesario para estimar el trabajo necesario.

Daily Scrum (Scrum Diario): Es una reunión con un bloque de tiempo de 15 minutos para que el Development Team sincronice sus actividades y cree un plan para las siguientes 24 horas. Esto se lleva a cabo inspeccionando el trabajo avanzado desde el último Daily Scrum y haciendo una proyección acerca del trabajo que podría completarse antes del siguiente.

Sprint Review (Revisión de Sprint): Se realiza al final del Sprint para inspeccionar el incremento y adaptar el Product Backlog si fuese necesario. Durante el Sprint Review el

Development Team presenta al Product Owner, clientes, usuarios, gestores, y otros lo que se hizo durante el Sprint. Basándose en esto y en cualquier cambio en el Product Backlog durante el Sprint, los asistentes colaboran para determinar las siguientes cosas que podrían hacerse para optimizar el valor. Se trata de una reunión informal, no una reunión de seguimiento, y la presentación del incremento tiene como objetivo facilitar la retroalimentación de información y fomentar la colaboración.

Sprint Retrospective (Retrospectiva del Sprint): Es la oportunidad para el Scrum Team de inspeccionarse a sí mismo y crear un plan de mejoras que sean abordadas durante el siguiente Sprint. Al igual que los modelos de procesos incorporan prácticas de "ingeniería de procesos" para conseguir una mejora continua de su capacidad, en agilidad también van surgiendo prácticas para lo que es el equivalente de mejora continua de la agilidad de la organización.

Esta ceremonia tiene lugar después del Sprint Review y antes del siguiente Planning Meeting Sprint. El Sprint Retrospective se centra en "CÓMO" lo estamos construyendo, "CÓMO" estamos trabajando, con el objetivo de analizar problemas y aspectos mejorables.

4.2.2. Automatización de Casos de Prueba.

4.2.2.1. Mejores Prácticas Para La Automatización De Pruebas.

Como mejores prácticas Smartbear (2016) nos propone:

- Selección de casos de prueba: los casos de prueba deben cumplir ciertos criterios para que el esfuerzo utilizado en estos sea bastante útil.
- Probar lo más pronto posible y a menudo: en el ciclo de vida del desarrollo de software es importante empezar la codificación de los casos de prueba lo más pronto posible para empezar a validar los entregables.
- Seleccionar la herramienta adecuada para la automatización: existen muchas herramientas en el mercado que ayudan a la automatización de los casos de prueba,

es importante saber escoger la que más se adapta a las necesidades propias de la empresa. Se debe tener en cuenta:

  - Soporte a plataformas y tecnologías.
  - Flexibilidad de acuerdo a las competencias de los Tester.
  - Fácil de utilizar.
  - Fácil de mantener los casos de prueba automatizados.

- Dividir los esfuerzos de la automatización: es importante identificar las habilidades y competencias de las personas que intervienen en la automatización para que de acuerdo a ellas se le asignen tareas. Por ejemplo: algunos pueden tener habilidades para la codificación de las pruebas y otros pueden tener mejores capacidades para el diseño de los casos de prueba.
- Crear buenos datos de prueba: los datos de prueba permiten realizar pruebas para diferentes escenarios, es por esto que se debe definir un conjunto de datos de prueba que permita cubrir varios escenarios con la menor cantidad de ejecuciones
- Crear casos de prueba que son resistentes a cambios de Interfaz Gráfica: dentro de los desarrollos de software las interfaces de usuario es lo que más cambia, y es por esto que muchas veces los casos de prueba de este tipo tienden a quedar obsoletos entre versiones; sin embargo, si se tienen ciertas prácticas como proveer nombres únicos a los controles, esto puede ayudar a que esos casos de prueba automatizados resistan más ante los cambios, también ayuda para que los casos no sean dependientes de coordenadas de la pantalla para encontrar un control en la pantalla.

Por otra parte, Groder (2015) nos plantea las siguientes mejores prácticas para el desarrollo de la automatización de pruebas:

- Planear los casos de pruebas lo más pronto posible: los casos de prueba y su ejecución requieren recursos, tanto de hardware como software, que no son necesarios para la ejecución manual. Es necesario contar con estos recursos lo más pronto para poder empezar a identificar posibles fallas en el desarrollo.
- Planear casos de prueba para requerimientos funcionales y no funcionales: se debe tener casos de pruebas de ambos tipos porque los funcionales ayudan en la identificación de fallas sobre el aplicativo mientras que los no funcionales ayudan

a identificar configuraciones y características para un mejor desempeño de la aplicación.

- Establecer una arquitectura: seleccionar una buena arquitectura es fundamental para construir casos de prueba que sean robustos, mantenibles y escalables.
- Tener casos de prueba fuera de la ruta crítica: los casos de prueba a implementar no deben tomar mucho tiempo en desarrollar y probar para que sean eficientes.
- Diseñar los casos de prueba en compañía de ingenieros de automatización de casos de prueba: los ingenieros pueden ayudar a desarrollar casos de prueba que funcionen de acuerdo a las entradas definidas y no serían casos de prueba con datos fijos.
- Usar técnicas de diseño de casos de prueba basado en Keywords (Palabras claves): tener casos de prueba basado en keywords ayuda en la modularidad de los casos de prueba, ayuda a identificar fácilmente el objetivo de la función, los casos de prueba son más fáciles de mantener.
- Proveer la capacidad de resetear los datos: cuando se realizan pruebas sobre bases de datos y esas pruebas cambian la información de la base de datos, se debe tener la posibilidad de reiniciar fácilmente la información, con el fin de poder reutilizar los casos de prueba.
- Usar principios del agilismo: la automatización de los casos de prueba siempre va a verse afectado por cambio de requerimientos, cambios en los cronogramas, cambios en las prioridades, es por esto que adoptar principios de las metodologías ágiles puede ayudar a minimizar el impacto por ese tipo de cambios.

4.2.3. Criterios de Selección de Casos de Prueba.

Cuando una empresa decide iniciar el proceso de automatización debe tener claro que no es posible automatizar cualquier caso de prueba, por tanto es importante definir de un conjunto de casos de prueba cuáles deben ser implementados. La idea principal de un caso automatizado es que se pueda reutilizar la mayor cantidad de veces.

De acuerdo con Smartbear (2016) los criterios que se deben tener en cuenta para obtener mejores beneficios son:

- Casos de pruebas repetitivos que se puedan ejecutar múltiples veces.
- Casos de prueba que tienden a causar error humano (al ejecutarse de forma manual).
- Casos de prueba que requieren múltiples conjuntos de datos.
- Funcionalidades que son usadas frecuentemente y que presentan condiciones de alto riesgo.
- Casos de prueba que son imposibles o difíciles de ejecutar manualmente.
- Casos de prueba que se deben ejecutar en múltiples plataformas (Hardware, Software).
- Casos de prueba que requieren de esfuerzo y tiempo al ser ejecutadas manualmente

Por su parte TestObject propone los siguientes criterios:

- Casos de prueba usados con más frecuencia.
- Casos de prueba más fáciles de automatizar.
- Casos de prueba que tienen resultados predecibles.
- Casos de prueba más tediosos de realizar manualmente.
- Casos de prueba que son imposibles o difíciles de ejecutar manualmente.
- Casos de prueba que se deben ejecutar en múltiples plataformas (Hardware, Software).
- Funcionalidad que es usada frecuentemente.

4.2.4. Integración Continua.

栖Fowler (2006) “Integración continua es una práctica de desarrollo de software en la cual

Fowler (2006) “Integración continua es una práctica de desarrollo de software en la cual los miembros de un equipo integran su trabajo con frecuencia, por lo general cada persona integra su trabajo por lo menos una vez al día dando lugar a múltiples integraciones por día. Cada integración es verificada por una build automática (incluyendo pruebas) para detectar errores de integración lo más rápidamente posible. Muchos equipos encuentran que este enfoque conduce a la reducción de manera significativa de los problemas derivados de integración y permite a los equipos desarrollar software con mayor rapidez”.

Fowler (2006), a pesar de no ser la persona que creó el concepto de Integración Continua, se considera una de las personas que hizo que se hiciera popular, varios años después de su creación. Las prácticas que se desarrollan en la integración continua son:

Mantener un repositorio único: Debido a la gran cantidad de archivos que se tienen en un desarrollo de software, es importante contar una herramienta que ayude en el control de las versiones de los archivos. En este repositorio deben estar todos los archivos necesarios para el correcto funcionamiento del proyecto: scripts, código fuente, esquemas, archivos de configuración, entre otros.

Una vez se tenga el repositorio, todas las personas interesadas en el proyecto deben tener el conocimiento del lugar donde éste está.

Automatizar la compilación/construcción de una revisión: Debe ser posible contar con la compilación automática del código fuente que se encuentra en el repositorio para garantizar que el código fuente está consistente y sin errores, cuando menos de compilación.

Pruebas automáticas para las revisiones: Una vez se ha compilado se deben ejecutar un conjunto de pruebas automatizadas que permitan garantizar que el funcionamiento de lo entregado es correcto.

Todos deben subir los cambios a diario: Realizar la entrega en el repositorio de forma frecuente permite encontrar y resolver de forma más fácil los errores por conflictos en el código fuente; en comparación con los conflictos que se podrían presentar en entregas de más tiempo.

Toda entrega al repositorio debe compilar la línea principal en un ambiente de integración: Debido a la falta disciplina en la actualización del repositorio local antes de hacer la compilación en el ambiente del desarrollador o por las diferencias que se presentan entre los ambientes de cada desarrollador del proyecto, es pertinente realizar una compilación de la entrega que realiza el desarrollador junto con el código que se encuentra en la línea/rama principal del repositorio del proyecto. Esto garantiza que los cambios que se suben al repositorio son correctos.

Corregir revisiones con problemas de inmediato: En el caso que una revisión (build) falle por alguna razón, la prioridad debe ser corregir esa revisión de inmediato. No deben dejarse errores en la línea/rama principal del proyecto.

Pruebas en un ambiente similar al de producción: Las pruebas realizadas al aplicativo se deben ejecutar en un ambiente similar al de producción o cuando menos lo más parecido. En algunos casos no es posible tener ambientes iguales debido a licencias de software o características especiales del hardware. En este punto se busca disminuir los posibles riesgos que se presenten por diferencias entre ambientes.

Conseguir el último ejecutable de forma fácil: Cualquier persona interesada en el proyecto debe tener la posibilidad de contar con la versión más reciente del ejecutable de forma fácil, ya sea para demostraciones, pruebas o visualizar cambios.

Cualquiera puede ver que está pasando: El proceso de la integración continua debe ser visible para cualquier persona interesada.

Puesta en los ambientes de forma automática: Debido a los diferentes ambientes con los que se cuenta en algunos proyectos, la integración debe tener la posibilidad de desplegar en los ambientes que sea posible y necesario de forma automática. Para el caso de los ambientes de producción, se debe tener la posibilidad de deshacer los cambios realizados por una entrega, teniendo en cuenta lo delicado de ese ambiente y que un error siempre puede pasar.

4.3. Marco Conceptual

En la presente investigación, se tendrán en cuenta algunos conceptos que son claves para desarrollar la propuesta de automatización de casos de prueba para aseguramiento de calidad en el desarrollo de software.

4.3.1. Calidad de Software.

La preferencia de la calidad se ha empleado desde los años cuarenta con el interés de obtener una alta calidad del producto y ahorro de costos. Para ello se basó en la terminología Total Quality Management (Gestión de Calidad Total) que se constituye en cuatro pasos: los dos primeros pasos se centran en la mejora del proceso, el tercer paso se enfoca en el usuario del producto y finalmente el cuarto paso se centra en la uso del producto en el mercado.

Pressman (2010) define la calidad de software como: "Concordancia con los requisitos funcionales y el rendimiento explícitamente establecidos, con los estándares de desarrollo explícitamente documentados, y con las características implícitas que se espera de todo software desarrollado profesionalmente."

Existen las mediciones de las características de un programa. Dichas propiedades incluyen complejidad ciclomática, cohesión, número de puntos de fusión y muchas otras. Cuando se examina un elemento con base a sus características mensurable se pueden encontrar dos tipos de calidad: calidad de diseño y calidad de concordancia: la calidad de diseño se refiere a las características que los diseñadores especifican para un elemento y la calidad de concordancia es el grado en el que las especificaciones de diseño se aplican durante la fabricación.

¿Cómo se logra el control de calidad? este involucra una serie de inspecciones, revisiones y pruebas empleadas a lo largo del proceso de software para garantizar que cada producto de trabajo satisfaga los requisitos que se han asignado. El control de calidad incluye un ciclo de retroalimentación con el proceso que creó el producto de trabajo. La combinación de medición y retroalimentación permiten afinar el proceso cuando los productos de trabajo creados fracasen en cuanto a satisfacer sus especificaciones.

4.3.2. Herramientas.

4.3.2.1. TestLink.

Es una herramienta web que permite la gestión de casos de pruebas. Dentro de las posibilidades que brinda TestLink está la administración de los casos de prueba, suite de pruebas, planes de prueba, proyectos de pruebas y gestión de usuarios. A través de su uso es posible obtener reportes de los casos de prueba ejecutados dentro de un plan de pruebas, así como la cobertura que tuvo en el desarrollo del mismo.

De acuerdo con la empresa QAustral, TestLink es una de las herramientas Open Source más usadas actualmente por los equipo de pruebas a nivel mundial.

4.3.2.2. Robot Framework.

Es un framework genérico para la automatización de pruebas. Permite realizar pruebas de diferentes tipos, de acuerdo al enfoque que se le quiera dar a las pruebas automatizadas.

El método de programación de las pruebas es por medio de keywords (palabras claves) que se definen en el desarrollo de las pruebas. Robot Framework cuenta con librerías internas con sus respectivas keywords para la ayuda en la codificación de las pruebas y permite la incorporación de nuevas librerías externas al proyecto.

4.3.2.3. Jenkins

Es un servidor de integración continua de código abierto y actualmente es uno de los más utilizados para esta función. En este servidor es posible configurar y agendar la ejecución de tareas con cada una de las partes que conforman el ciclo de vida de un proyecto.

Posee una larga lista de plugins los cuales le permiten definir tareas que puedan integrarse con herramientas de control de versiones y ejecutar proyectos basados en Apache Maven, Apache Ant, Microsoft MSBuild, scripts en shell y scripts batch. Además permite ejecutar tareas adicionales previas y posteriores a la compilación como preparar el entorno, preparar un emulador, realizar un despliegue o compactar y subir binarios a un FTP.

5. Diseño Metodológico

5.1. Enfoque

De acuerdo con Bernal (2010) el enfoque cualitativo está enfocado en la profundización del caso de estudio y no en generalizar, lo que se busca es un entendimiento de la situación analizada. No está enfocado en medir, por el contrario, lo que busca es cualificar y describir a partir de características determinantes de acuerdo a como sean percibidos por los elementos que están dentro de la situación a estudiar.

La presente investigación tendrá un enfoque cualitativo, ya que tiene como finalidad analizar y describir el proceso de control de calidad del desarrollo de software en Axede el cual se realiza de manera manual; tomando como base la documentación de la definición del proceso y los documentos generados como resultado de la ejecución. Finalmente, se realizará una propuesta en la implementación de pruebas automáticas que permitan una optimización del control de la calidad.

5.2. Tipo de Investigación

De acuerdo con Hernández (2010) la investigación descriptiva pretende especificar las propiedades y características del fenómeno que se somete a un análisis. Con este tipo de investigación se pretende recoger información sobre las variables del caso de estudio pero no pretende indicar cómo se relacionan entre ellas.

Para el caso concreto de la investigación a realizar, se busca hacer una descripción de la situación actual del proceso de aseguramiento de calidad para el desarrollo de software en la empresa Axede S.A.; así mismo, hacer una descripción de posibles mejoras al proceso, las cuales se verán reflejadas en una propuesta que permita a Axede tener un control de calidad más eficiente.

5.3. Contexto

La presente investigación se realizará en la empresa Axede S.A. que tiene una trayectoria de más de 40 años en el mercado Colombiano como integradores de comunicaciones y con más de 15 años en el desarrollo de software a la medida. Ofrece soluciones integrales en el campo de las Tecnologías de Información y Comunicaciones, convirtiéndose en un aliado para entregar respuestas más inteligentes a sus clientes.

Cuenta con un equipo de más de 250 profesionales altamente calificados y capacitados, orientados en desarrollar soluciones que hagan más competitivos a sus clientes. Actualmente tienen más de 100 ingenieros certificados, distribuidos en las regionales de Bogotá, Medellín, Cali y Barranquilla.

La satisfacción del cliente es su cultura y gracias a las cualidades profesionales y humanas de cada miembro del equipo, son un componente esencial del éxito de sus Clientes.

Misión

Garantizar a nuestros clientes la satisfacción de sus necesidades de soluciones de tecnologías de información y comunicaciones con productos y servicios innovadores que aseguren su gestión.

Visión

Consolidarnos como líderes en soluciones de tecnologías de información y comunicaciones antes del 2015.

Unidad de Software

Son una compañía que cuenta con más de 20 años de experiencia en venta de soluciones y servicios de software. Actualmente, manejan dos líneas de negocio: Outsourcing y Producto

Propio. Su alto nivel de madurez y conocimiento en las principales tecnologías del mercado aseguran un éxito indiscutible en todos sus proyectos tecnológicos.

Metodología y madurez del servicio (Unidad de Software)

Cuentan con una metodología de planeación, desarrollo e implementación de proyectos acorde a los más altos estándares mundiales. La combinación de certificaciones y alianzas así como la adopción de las mejores prácticas en sus procesos, son un sinónimo de calidad y garantía en sus servicios.

- CMMI DEV nivel 3 ver 1.3
- CMMI SVC nivel 3 ver 1.3
- PMP: PMI
- RUP
- Metodologías Ágiles: SCRUM (Master y Developer)

Modelos de Gestión Adoptados

CMMI: Capability Maturity Model Integration

Es un modelo de mejores prácticas que ayuda a las organizaciones a mejorar sus procesos y evalúa la madurez de los mismos en 5 niveles.

Consiste en aplicar las mejores prácticas dirigidas al mantenimiento y desarrollo de actividades que cubren el ciclo de vida de producto/servicio desde su concepción, entrega y mantenimiento. También proporciona orientación a las organizaciones proveedoras de servicios en el establecimiento, administración y prestación del servicio.

La valoración en CMMI permite a las empresas, optimizar sus procesos y garantizar su calidad; disminuir los márgenes de error y costos, administrar el riesgo, mejorar los tiempos de respuesta, incrementar la productividad, generar confiabilidad y mayor satisfacción de los clientes.

1. CMMI para el Desarrollo (CMMI-DEV): En él se tratan procesos de desarrollo de productos y servicios. Axede S.A. se encuentra valorada en Nivel 3 desde 2007 y en Marzo de 2013 paso a la versión 1.3 del modelo, manteniendo su nivel de madurez.
2. CMMI para servicios (CMMI-SVC): Está diseñado para cubrir todas las actividades que requieren gestionar, establecer y entregar Servicios. Axede S.A. obtuvo valoración en Nivel 3 en Marzo de 2015.

ISO 9001:2008

La norma ISO 9001, establece la estructura de un Sistema de Gestión de la Calidad y proporciona a la organización las bases fundamentales para controlar las operaciones de producción y de servicio dentro del marco de un Sistema de Gestión de la Calidad, y mejora la orientación hacia el cliente y el incremento en la competitividad.

La certificación ISO 9001, demuestra una base sólida del Sistema de Gestión implementado en Axede S.A, en cuanto al cumplimiento satisfactorio de los requisitos de nuestros clientes y la excelencia en el desempeño

Axede S.A. desde el año 2007 está certificada en ISO 9001 en la versión 2008, teniendo como alcance el Diseño, venta, implementación, soporte y mantenimiento de soluciones de Tecnologías de Información y Comunicaciones.

OHSAS 18001

Es una norma que establece los requisitos mínimos de las mejores prácticas en gestión de seguridad y salud en el trabajo con el objetivo de proteger a la organización y a sus colaboradores.

La certificación OHSAS 18001 vigente desde el año 2014, demuestra la solidez del Sistema de Gestión implementado en Axede S.A, en cuanto a:

- Crear las mejores condiciones de trabajo posibles en toda su organización
- Identificar los riesgos y establecer controles para gestionarlos

- Reducir el número de accidentes laborales y bajas por enfermedad para disminuir los costos y tiempos de inactividad ligados a ellos
- Comprometer y motivar al personal con unas condiciones laborales mejores y más seguras
- Demostrar la conformidad a clientes y proveedores

5.4. Instrumentos de recolección de información

Para la presente propuesta se implementarán instrumentos de recolección de información como son: Análisis Documental, entrevistas semi-estructuradas y simulación.

A continuación se hará la descripción de cada uno de ellos, así como también la intención de su aplicación en la presente investigación.

5.4.1. Análisis documental

De acuerdo con Corbin y Strauss (2008), la técnica de análisis documental es un procedimiento sistemático para revisar o evaluar documentos impresos y materiales electrónicos. Al igual que otros métodos analíticos en la investigación cualitativa, el análisis documental requiere que los datos sean interpretados y examinados a fin de obtener significado, ganar la comprensión y desarrollar el conocimiento empírico.

El análisis documental a realizar estará enfocado en la revisión de:

- Proceso de desarrollo de software: con este análisis se busca conocer a detalle el proceso que se tiene actualmente en la empresa el cual está registrado en la documentación del área de procesos.
- Ejecución de casos de prueba: con este análisis se busca identificar cuáles de los casos de prueba existentes actualmente son relevantes para una posible automatización y de ellos tomar una muestra para una simulación a realizar en el marco de esta de investigación.

5.4.2. Entrevista semi-estructurada

La entrevista es una técnica que aporta un gran beneficio en la investigación cualitativa para recolectar datos; se define como un diálogo que se presenta con un fin establecido diferente al sencillo hecho de conversar. Canales (2006) la define como "la comunicación interpersonal establecida entre el investigador y el sujeto de estudio, a fin de obtener respuestas verbales a las interrogantes planteadas sobre el problema propuesto".

De acuerdo con Hernández (2010), la entrevista semiestructurada se basa en una guía de preguntas estructuradas a formular al entrevistado y adicional a esto el entrevistador tiene la libertad de introducir preguntas adicionales para precisar conceptos u obtener mayor información sobre los temas a investigar.

Esta entrevista se utilizará en la investigación para averiguar sobre la historia del proceso de desarrollo de software en Axede S.A, cómo se tiene implementado, aspectos que a la fecha no se tienen en cuenta en el proceso y posibles mejoras que se hayan intentado realizar pero que no fueron exitosas.

5.4.3. Simulación

De acuerdo con Shanon y Johaness (1976) la simulación puede definirse como un proceso en el cual se establece un delineamiento, cuyo objetivo es plantear la representación de un sistema real y con la única finalidad de entender el funcionamiento del sistema o la evaluación de nuevas estrategias.

Ahora bien, la simulación es relevante en esta investigación porque permite analizar y comparar una construcción y ejecución manual contra la codificación y ejecución de los casos de prueba previamente codificados. La comparación puede ayudar a revelar que determinados casos de prueba sería más conveniente manejarlos de forma codificada que tener un procedimiento que se debe ejecutar en diferentes ocasiones de forma manual.

5.5. Validez de los instrumentos

Con respecto a la validez de los instrumentos, Hernández y otros (2003), establecen que "se refiere al grado en que un instrumento mide la variable que pretende medir" (p.346). De acuerdo con esto, se pretende que la validez de los instrumentos de recolección de información utilizados en la presente investigación, se relacionen con la problemática frente a la ejecución de casos de prueba de forma manual sobre los desarrollos de software, los objetivos propuestos, así como las variables e indicadores de este estudio.

Para validar los instrumentos se solicitó la revisión de expertos para garantizar la obtención de información hacia el logro de los objetivos propuestos.

6. Resultado

6.1. Análisis de los resultados

6.1.1. Codificación de las producciones.

Los instrumentos utilizados para la recolección de la información fueron:

- Análisis documental: para conocer a detalle el proceso de ejecución de casos de prueba de desarrollo de software que se tiene actualmente en la empresa Axede S.A.
- Entrevista semi-estructurada: para indagar sobre el proceso que se tiene implementado, aspectos que a la fecha no se tienen en cuenta en el proceso y posibles mejoras que se hayan intentado realizar pero que no fueron exitosas.
- Simulación: para analizar y comparar una construcción y ejecución manual contra la codificación y ejecución de los casos de prueba previamente codificados.

Durante el análisis de los resultados dentro del componente cualitativo, se tuvieron en cuenta las producciones del Gerente de Proyectos y el Analista de Pruebas, para los cuales se reservó sus nombres y se utilizaron seudónimos a fin de referir las intervenciones realizadas por cada uno de ellos. Específicamente, en las entrevistas, se identificaron de la siguiente manera:

- **Entrevista Semi-estructura N°1**

  Entrevistador: E
  Entrevistado: P1

- **Entrevista Semi-estructura N°2**

  Entrevistador: E
  Entrevistado: P2

6.2. Análisis cualitativo

Siguiendo las recomendaciones de autores como Hernández, Fernández y Baptista (2006), el procedimiento para la organización y análisis de los datos fue el siguiente:

- Se realizó la lectura y estudio detallado de todos los instrumentos aplicados en la investigación como: análisis documental, entrevistas y simulación.
- Se realizaron las transcripciones literales de las entrevistas.
- Se organizó la información de acuerdo a las categorías iniciales, subcategorías e indicadores específicos definidos de acuerdo al marco teórico.
- Se identificaron las categorías, subcategorías e indicadores, antes mencionados y que están involucrados en las respuestas de las entrevistas.
- Posteriormente y a partir de la organización de los datos, se realizó el análisis descriptivo de cada categoría en relación con las preguntas de la entrevista, para obtener un análisis de los hallazgos encontrados.

A continuación se presentan las categorías, subcategorías e indicadores que permitieron abordar el análisis de los datos.

Tabla 2: *Categorías, subcategorías e indicadores*

Categorías	Subcategorías	Indicadores
Proceso de ingeniería de software	Desarrollo de software	Metodología de desarrollo
	Aseguramiento y control de calidad	Prácticas para la automatización de pruebas Herramientas para ejecución automática de casos de pruebas Integración continua para los casos de pruebas automáticas

Fuente: Elaboración propia

Categoría: Proceso de ingeniería de software

Subcategoría: Desarrollo de software

Indicador: Metodología de desarrollo

Para la empresa Axede.S.A, en el área de software, existen dos clientes importantes: UNE y EPM, ambos de la ciudad de Medellín. Para la ejecución de los desarrollos tanto de nuevas funcionalidades (denominado Mantenimiento Preventivo), como soporte (denominado Mantenimiento Correctivo), se siguen un conjunto de etapas bajo el lineamiento de la metodología RUP, lo cual se ve reflejado en el documento Proceso de Desarrollo de Software, así como en la respuesta a la pregunta "*¿Cuál es la metodología que se maneja actualmente en el proceso de desarrollo de software de Axede S.A?"* data por P1 *"Actualmente Axede S.A. ejecuta su proceso de desarrollo bajo una metodología propia, definida a nivel de los servicios que presta de Outsourcing, Desarrollo a la medida, y Adaptativo Ágil; bajo las guías de buenas prácticas de Modelos de Madurez de procesos como CMMI DEV – SVC, y estándares PMI. Adicionalmente a ello, la metodología establecida para Desarrollo a la medida ha adoptado algunos lineamientos RUP, y la de Marcos Ágiles lineamientos SCRUM."*

Por otro lado, de acuerdo con la página oficial de Axede S.A. y su intranet, existe la intención de implementar una metodología ágil dentro de la oferta de la empresa como parte de su plan estratégico y para esto está en desarrollo la definición del marco de trabajo bajo la metodología ágil SCRUM. Así mismo, EPM está modificando sus procesos para que los servicios contratados con terceros se ejecuten bajo la misma metodología que está implementando Axede S.A.

Teniendo en cuenta que el mercado está en un proceso de evolución hacia metodologías ágiles, como LEAN, KANBAN, XP o SCRUM; adicional que uno de sus clientes más grandes está en proceso de migración de su forma de trabajo a SCRUM y que Axede S.A. en su continua evolución y adaptación al mercado, junto con la migración de EPM, es conveniente que Axede S.A. continúe fortaleciendo la implementación de la metodología SCRUM. Además, cabe resaltar que actualmente gran parte de sus colaboradores han

aprovechado las convocatorias implementadas por MINTIC y cuentan con la certificación oficial de la entidad Scrum Alliance en los niveles Scrum Master y Scrum Developer.

Al trabajar con clientes que tienen sistemas de información de ruta crítica los cuales deben responder a normas regulatorias de manera eficaz, SCRUM es un aliado ya que busca finalizar entregas lo más rápido posible, el cual es un indicador operante muy importante para el cliente. SCRUM incrementa la productividad ya que implica trabajar alrededor de necesidades o requisitos del negocio implicando un grupo de trabajo con conocimiento multidisciplinario, donde cada persona entra en un proceso de autogestión realizando tareas lo más simple y rápidamente posible.

No se debe olvidar que dentro de las metodologías existentes se encuentra el proceso de pruebas, que actualmente en Axede S.A. trabaja bajo la metodología RUP, tiene este proceso después de la etapa de construcción lo cual se puede evidenciar en el documento Proceso de Desarrollo de Software; mientras que en SCRUM, el proceso de pruebas se da inicio desde el análisis de las historias de usuario hasta que se realiza el despliegue de lo construido, es decir, la calidad va incorporada en todas las etapas de la metodología, en donde el equipo de pruebas es parte integral del equipo. La necesidad de que el proceso de pruebas se encuentre en todas la etapas de desarrollo de software, se refleja en la respuesta de P2 a la pregunta "*En la metodología de desarrollo que tiene actualmente Axede ¿Cree que deberían aplicarse algún cambio en el desarrollo de sus actividades? Si es así ¿Cuáles cambios?*: *Dentro de la metodología de desarrollo actualmente definida por Axede, considero que la empresa no le da la importancia que realmente se merece al proceso de pruebas. Por lo tanto, considero que se podrían adoptar los siguientes cambios: Se debería tener en cuenta al analista de pruebas desde el inicio de la construcción de un producto; capacitaciones sobre el negocio; y capacitación en las diferentes herramientas o aplicativos."*

En el marco teórico de la presente investigación se menciona las metodologías XP y TDD, las cuales no se recomienda aplicarlas en Axede por las siguientes razones:

XP

- XP se caracteriza por el trabajo en pares y aplicar esta metodología con la cantidad de empleados que se tienen en el área de software no permitiría cubrir las necesidades de los clientes.

- Incursionar en esta metodología de desarrollo implicaría una curva de aprendizaje que no sería conveniente
- Inversión en capacitación

TDD

- Al igual que XP es necesario una curva de aprendizaje
- Es necesario la inversión en capacitación de todo el personal
- La utilización de TDD puede Implicar demasiados costos que difícilmente el cliente los reconozca.

Categoría: Proceso de ingeniería de software

Subcategoría: Aseguramiento y control de calidad

Indicador: Prácticas para la automatización de pruebas

Las mejores prácticas para la automatización de pruebas enunciadas por Smartbear y por Groder tienen similitud en varios de sus puntos, los que no son similares sirven como complemento a los puntos del otro autor. A continuación se listan las mejores prácticas de cada uno de los autores, al frente de cada una están las prácticas que a criterio de la presente investigación son similares y los que no son similares tienen una casilla en blanco al frente.

Tabla 3: *Mejores prácticas para automatización de pruebas de Smartbear y Groder*

Smartbear	**Groder**
- Selección de casos de prueba.	- Planear casos de prueba para requerimientos funcionales y no funcionales. - Diseñar los casos de prueba en compañía de ingenieros de automatización de casos de prueba.
- Probar lo más pronto posible y a menudo.	- Planear los casos de pruebas lo más pronto posible.

- Seleccionar la herramienta adecuada para la automatización.	- Establecer una arquitectura.
- Dividir los esfuerzos de la automatización.	- Usar principios del agilísmo.
- Crear buenos datos de prueba.	- Tener casos de prueba fuera de la ruta crítica - Usar técnicas de diseño de casos de prueba basado en Keywords (Palabras claves).
- Crear casos de prueba que son resistentes a cambios de Interfaz Gráfica.	
	- Proveer la capacidad de resetear los datos.

Fuente: Elaboración propia

- La práctica "*Selección de casos de prueba*" de Smartbear se puede relacionar con las prácticas "*Planear casos de prueba para requerimientos funcionales y no funcionales*", "*Diseñar los casos de prueba en compañía de ingenieros de automatización de casos de prueba*" de Groder.

  Estas prácticas hacen hincapié en la generación de casos de prueba útiles que le den valor y permitan ver de manera rápida los beneficios y utilidad que puede tener la automatización de los casos de prueba. En este punto es importante resaltar que no se debe partir de la idea de que todos los casos de prueba se deben automatizar, por tanto es esencial determinar los casos de prueba a automatizar. Los beneficios de los casos de prueba automatizados están relacionado con la cantidad de veces que el caso de prueba puede ser aprovechado, si el caso de prueba será ejecutado pocas veces, es mejor dejarlo para que sea implementado de manera manual. Los criterios que se deben tener en cuenta son:

  - ✓ Casos de prueba repetitivos.
  - ✓ Casos de prueba que puedan generar error por causas humanas.
  - ✓ Casos de prueba que requieran muchas variaciones en su conjunto de datos de prueba.
  - ✓ Casos de prueba que sean difíciles de ejecutar manualmente.

- ✓ Casos de prueba que se ejecutan con diferentes configuraciones o en diferentes plataformas.

Además de tener presente los criterios mencionados, tener la posibilidad de contar con colaboradores que sean ingenieros de automatización de casos de prueba o que cuenten con experiencia sería de gran ayuda tanto para la selección de los casos de prueba como en el momento de la implementación.

Como se ha dicho en el desarrollo de la presente documentación, Axede S.A. no cuenta en este momento con un proceso de automatización de casos de prueba en ninguno de sus proyectos como lo confirma P1 frente a la pregunta "De las mejores prácticas existentes para la automatización de pruebas, ¿Conoce alguna que se está aplicando actualmente en las pruebas manuales?: Actualmente en Axede S.A. solo se están realizando pruebas manuales. La compañía ha realizado algunos pilotos enfocados a la automatización de las pruebas manuales, pero estos pilotos no se han implementado en los servicios, ni se han adoptado de manera oficial en el proceso de calidad."; y la selección de los casos de prueba se hace de acuerdo con base en la experiencia, esto último se pudo ver reflejado en el análisis documental de las matrices de casos de prueba existentes a la fecha. En cuanto a la experticia de P2, de acuerdo con la respuesta a la pregunta "¿En su vida profesional en alguna ocasión ha implementado casos de prueba automatizados?: Hasta el momento la mayoría de aplicativos a los cuales les he realizado calidad no han requerido de este tipo de pruebas, de otro lado, los aplicativos que lo han requerido, la empresa no ha estado dispuesto a invertir en dicho propósito.", se puede deducir que no tiene; esto lleva a considerar la posibilidad de capacitación del colaborador por medio de un tercero y/o consultar a otros colaboradores sobre el conocimiento en actividades de automatización de casos de prueba. Sobre la capacitación por medio de un tercero, se obtuvo información que en este momento el analista de pruebas fue aceptado para participar en la convocatoria de competencias transversales desarrollada por el MINTIC y su capacitación se realizará en la certificación Certified Professional - Basic Agile Testing donde se cubrirán temas de automatización de casos de prueba.

- La práctica "*Probar lo más pronto posible y a menudo*" de Smartbear se puede relacionar con la práctica "*Planear los casos de pruebas lo más pronto posible*" de Groder.

  La planeación de los casos de prueba así como la ejecución de estos, ya sea manual o automática, se debe realizar lo más pronto posible dentro del ciclo de vida del proyecto; con esto se busca encontrar posibles inconvenientes en una etapa temprana.

  Para la planeación de los casos de prueba automáticos se deben tener en cuenta los recursos no sólo humanos sino también recursos en máquinas y software que permitan un desempeño óptimo en el proceso de implementación del caso de prueba de forma automática y en la ejecución del caso de prueba. El uso de las máquinas y del software implica un gasto que debe ser tenido en cuenta al momento del desarrollo del proyecto

  En el caso específico de Axede S.A., el análisis documental deja ver que en la actualidad no se cuenta con software para el diligenciamiento de los casos de prueba, todo está registrado en archivos en EXCEL. Al momento de hacer la planeación no se tiene en cuenta la reutilización de casos de prueba previamente ejecutados, por el contrario lo que se hace es volver a diligenciar los casos de prueba en un nuevo archivo en EXCEL que una vez terminado la etapa de prueba será almacenado en el repositorio de archivos y no se tendrá en cuenta para una próxima ejecución. Con el objetivo de contrarrestar esta situación se realizará una propuesta de aplicativos que pueden ayudar al momento de la planeación con la reutilización de los casos de prueba.

  Otra evidencia que permitió ver el análisis documental es la intervención del analista de calidad en el proyecto, el cual entra a realizar sus labores una vez los desarrolladores han realizado sus entregas, es decir, el analista de calidad no es tenido en cuenta desde el inicio de las actividades de desarrollo. P2 deja ver esta situación al responder "*Se debería tener en cuenta al analista de pruebas desde el inicio de la construcción de un producto.*" ante la pregunta "*En la metodología de desarrollo que tiene actualmente Axede ¿Cree que deberían aplicarse algún cambio en el desarrollo de sus actividades? Si es así ¿Cuáles cambios?*".

- La práctica "*Seleccionar la herramienta adecuada para la automatización*" de Smartbear se puede relacionar con "*Establecer una arquitectura*" de Groder.

  La selección de las herramientas a utilizar es esencial en la automatización de los casos de prueba debido a diferentes factores como lo son la inversión en licencias, la facilidad en el uso de la herramienta o las características que estas poseen. Adicional a la selección de las herramientas a utilizar se debe definir una arquitectura que las involucre para su interacción y que defina la disposición de los archivos generados por la codificación de los casos de prueba sin demasiadas complicaciones; para la definición de la arquitectura, es necesario tener claro que los casos de prueba automatizados van a tener cambios en el tiempo y que esas modificaciones deben ser lo menos traumáticas como sea posible.

  Para el caso de Axede S.A., dado que no cuenta con una automatización de casos de prueba, no existe ni el uso de herramientas ni se tiene definida una arquitectura. Con el fin de minimizar la inversión necesaria por parte de Axede S.A. para iniciar la automatización de sus casos de prueba, la presente investigación hará la propuesta de algunas herramientas libres que tienen una amplia trayectoria y que tienen un gran respaldo en la comunidad. Este punto sobre las herramientas será consolidado más adelante al igual que una posible arquitectura.

- La práctica "*Dividir los esfuerzos de la automatización*" de Smartbear y "*Usar principios del agilismo*" de Groder.

  Tener un conjunto de casos de prueba automatizados es equivalente a tener un proyecto de desarrollo más, en este proyecto se van a presentar diferentes modificaciones y desarrollo de nuevos casos; y como cualquier proyecto de desarrollo, se deben utilizar los recursos de forma eficiente. Es aquí donde cabe conocer las habilidades, conocimiento y/o experiencia de los colaboradores que van a participar en el desarrollo de la implementación de los casos para dividir el esfuerzo de la mejor manera y así no tener una implementación que en lugar de generar beneficios se convierta en un dolor de cabeza. Ejemplos de tareas que se

tienen en la implementación son el diseño de los casos de prueba y la codificación de esos casos.

En ese desarrollo del proyecto de la automatización, utilizar algunas actividades del agilismo puede ayudar a que el proyecto no se convierta en un problema. No se trata de implementar todas las actividades de metodologías ágiles, sino utilizar algunas actividades. Ejemplo de esas actividades son:

- ✓ Tener una lista con cada una de las tareas a realizar.
- ✓ Mantener constante el trabajo programado en el sprint.
- ✓ Planear las tareas del sprint al inicio del sprint.
- ✓ Asegurar que hay un seguimiento al progreso que se tiene del proyecto con revisiones diarias.
- ✓ Desarrollar gráficas que muestren el progreso del proyecto.
- ✓ El equipo realice la asignación de acuerdo a la experticia.

Para el caso de Axede S.A., al no tener casos de prueba automatizados esto no se realiza. En cuanto a las habilidades, conocimiento y experticia, como se dijo anteriormente, los analistas de calidad están próximos a capacitarse haciendo uso de la convocatoria realizada por el MINTIC, lo cual es de gran ayuda para el desarrollo de la implementación de los casos de prueba.

- La práctica "*Crear buenos datos de prueba*" de Smartbear se puede relacionar con las prácticas "*Tener casos de prueba fuera de la ruta crítica*" y "*Usar técnicas de diseño de casos de prueba basado en Keywords (Palabras claves)*" de Groder

Al momento de la ejecución de los casos de prueba es inherente contar con buenos datos de prueba buscando cubrir diferentes escenarios con la menor cantidad de ejecuciones. Esos datos de prueba pueden ser generados de forma individual asociado a un caso específico o se puede utilizar la técnica de casos de prueba basado en Keywords que ayuda a crear funciones donde se puede identificar fácilmente el objetivo de la función y los casos de prueba son más fáciles de mantener; una vez se tiene la función, ésta tomará los datos de la prueba desde una fuente que contenga todos los datos para ese caso de prueba.

Para el caso de Axede S.A., al no tener casos de prueba automatizados, esto no se evidencia de ninguna forma, sin embargo las herramientas a utilizar y la arquitectura que se propondrá más adelante permitirá aplicar esta práctica. Es necesario aclarar que el uso de esta práctica depende mucho del conocimiento y la experticia del colaborador que está implementando el caso de prueba.

- "*Crear casos de prueba que son resistentes a cambios de Interfaz Gráfica*" de Smartbear.

  Las interfaces de usuario son de los artefactos que más presenta modificaciones dentro de los desarrollos de software y a causa de esto los casos de prueba de este tipo tienden a quedar obsoletos entre versiones; sin embargo, la aplicación de ciertas prácticas como proveer nombres únicos a los controles ayuda a que esos casos de prueba automatizados resistan más ante los cambios. También ayuda para que los casos no sean dependientes de coordenadas de la pantalla para encontrar un control dentro de esta.

  En el desarrollo de la presente investigación no se considera la aplicación de casos de prueba de interfaz gráfica debido a la complejidad de uso en las herramientas que permiten la generación de esos casos de prueba. Adicional a esto, tomando la mejor práctica "*Selección de casos de prueba*" y de acuerdo con el análisis documental realizado a los casos de pruebas ejecutados, las interfaces gráficas presentan pocas modificaciones y no sería rentable invertir tiempo en el desarrollo de casos de prueba que no van a ser utilizados en diferentes ocasiones.

- "*Proveer la capacidad de resetear los datos*" de Groder.

  En la ejecución de casos de pruebas sobre bases de datos u otro medio de almacenamiento donde se presenta que esa ejecución realiza cambios significativos sobre la información almacenada, se hace necesario contar con la posibilidad de reiniciar de manera fácil el medio de almacenamiento buscando que los casos de prueba sean reutilizables.

En casos ideales, la mejor opción es contar con un ambiente especial el cual se pueda reiniciar por completo una vez se ha ejecutado un ciclo de pruebas automáticas. El reinicio se puede ejecutar por medio de recuperación de respaldos (Backups) que contenían la versión inicial del medio de almacenamiento. Sin embargo, en muchas ocasiones la opción ideal de contar con un ambiente especial se hace muy difícil debido al licenciamiento o a los requisitos técnicos del sistema que son requeridos.

En el caso de Axede, en la revisión documental se encontró que aplica el caso en el que no es posible contar con un ambiente especial debido a las exigencias técnicas y de licenciamiento que tienen sus clientes más grandes. Como opción se plantea la alternativa de recurrir a la característica Teardown en la ejecución de los casos de prueba la cual consiste en la eliminación de los datos de prueba almacenados al momento de ejecutar el caso de prueba. Algunos datos se podrán eliminar de forma masiva, otros será necesario realizar el borrado de forma individual.

Categoría: Proceso de ingeniería de software

Subcategoría: Aseguramiento y control de calidad

Indicador: Herramientas para ejecución automática de casos de pruebas

Actualmente en Axede S.A. no existe una herramienta en la cual se almacene los casos de pruebas que se construyen a las piezas de código fuente de los diferentes proyectos que son atendidos. El proceso de aseguramiento de calidad se lleva a cabo diligenciando plantillas construidas en EXCEL en la cual se plasma la información de los diferentes casos de prueba, iteraciones realizadas y tiempos que se lleva la ejecución de estos.

Para mitigar esta situación y darle visibilidad a los casos de prueba, se propone la utilización de la herramienta TestLink que es código libre y no tiene costo de licenciamiento. Esta herramienta permite el registro de los diferentes casos de prueba para cada uno de los proyectos de software y así mantener una mejor calidad, una trazabilidad de todo lo que se está probando y una documentación optimizada.

Una de las bondades de TestLink es la creación de plantillas las cuales pueden ser importadas desde un archivo de EXCEL lo cual sería un beneficio para Axede S.A ya que es posible importar información de casos de pruebas que ya se encuentran diligenciados. Otra ventaja de la utilización de TestLink es la posible integración con el bug tracker que tiene actualmente la compañía (Mantis-bugtracker), de esta manera se puede lograr tener sistemas de información integrados y llevar una trazabilidad en conjunto tanto de los errores reportados con sus respectivas pruebas realizadas.

Otra herramienta propuesta es Robot Framework el cual es un framework genérico para la automatización de pruebas. La forma como trabaja es por la definición de los casos de prueba empleando palabras clave, permitiendo ser entendido de una forma más fácil. Es una herramienta de automatización a nivel de pruebas de aceptación muy fácil de usar, potente, flexible y se puede extender su funcionalidad a través de módulos de Python y Java, no tiene costo de licenciamiento y no necesita mucha experiencia en programación para escribir los casos de pruebas.

Simulación

- **Selección de los casos de prueba**

  Uno de los objetivos de la presente investigación es la realización de una simulación donde se implemente la codificación de algunos de los casos de prueba que se tienen en Axede S.A. Para la realización de esta simulación se utilizaron 9 casos de prueba de funcionalidades que son importantes en el funcionamiento de uno de los clientes de Axede S.A, todos son de operación crítica y algunos de ellos son transversales a muchas funcionalidades del aplicativo. Los casos de prueba a implementar son los siguientes:

  - Estandarización de dirección
  - Creación de Cliente
  - Creación de Contrato
  - Creación de Producto
  - Creación de Elementos de Medida
  - Cambio de Dirección de Producto
  - Cambio de Dirección de Contrato
  - Cambio de Dirección del Elemento

- Cambio de Ciclo

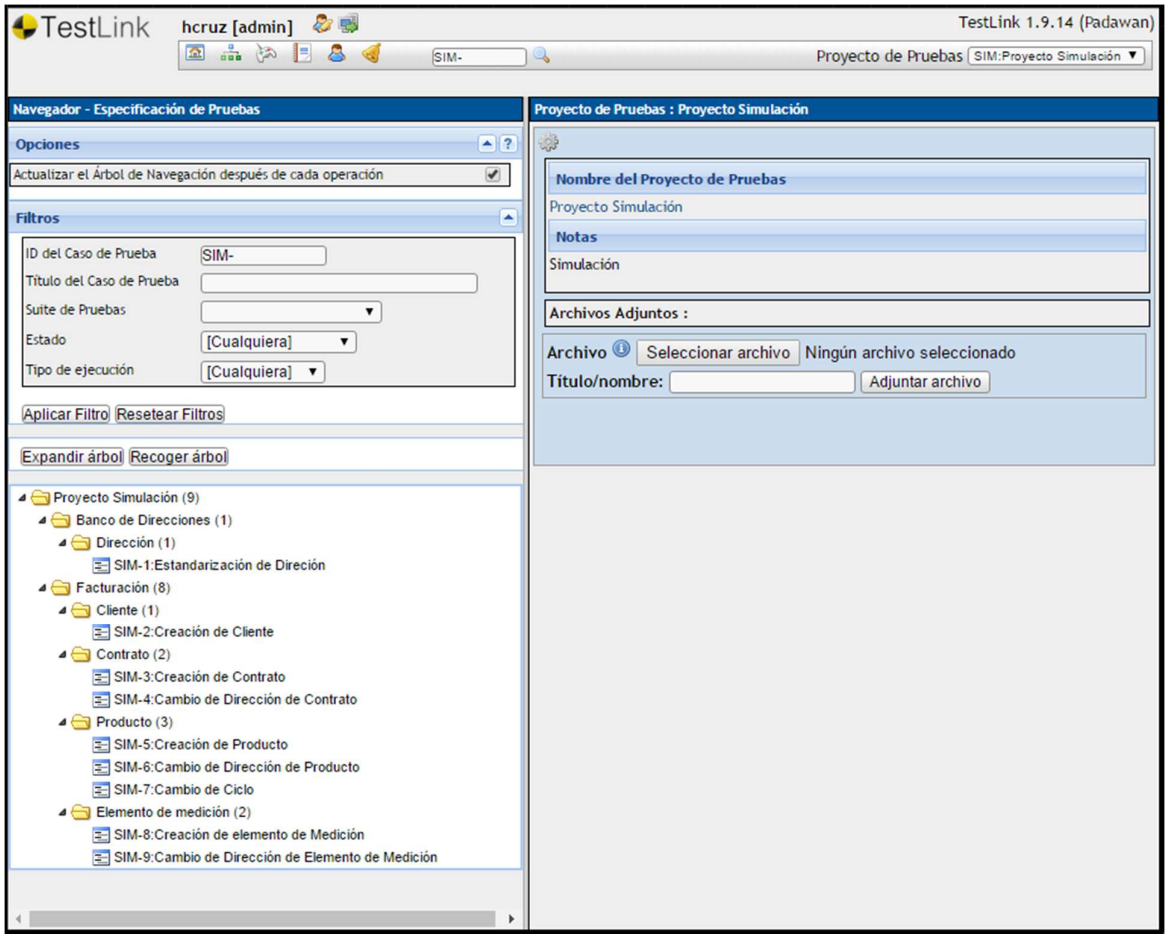
Por motivos de confidencialidad no es posible detallar estos casos de prueba en la presente investigación, por tanto no se mostrarán los registros generados ni la codificación realizada en las herramientas. Para cada uno de los casos de prueba se implementará el camino óptimo, es decir, se van a ignorar los casos donde se presentan excepción por validación de atributos ingresados.

- **Registro en TestLink**

Los registros en TestLink quedan de la siguiente forma:

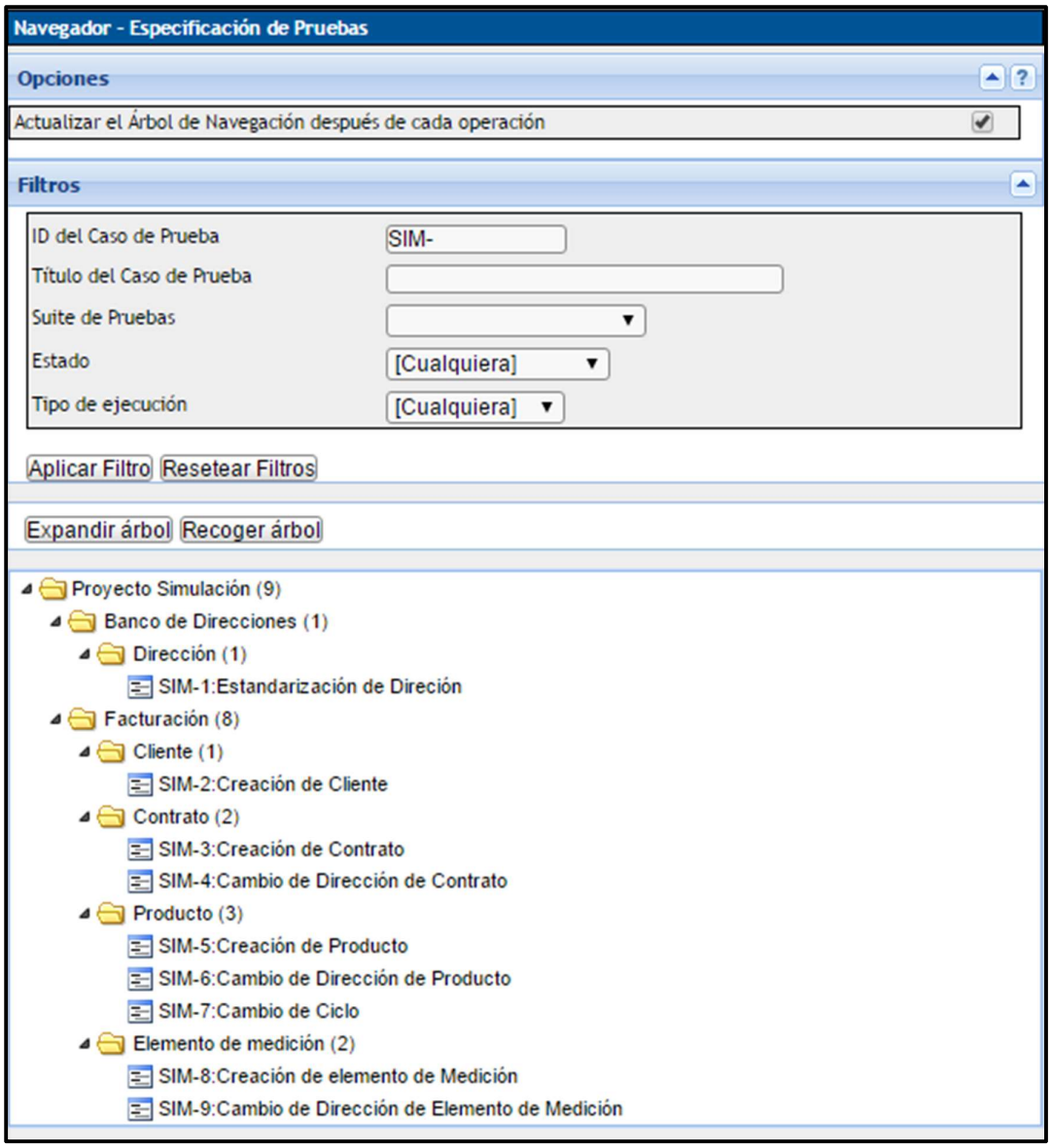
- Proyecto Simulación
  - Banco de Direcciones
    - Dirección
      - Estandarización de Dirección
  - Facturación
    - Clientes
      - Creación de Cliente
    - Contrato
      - Creación de Contrato
      - Cambio de Dirección de Contrato
    - Producto
      - Creación de Producto
      - Cambio de Dirección de Producto
      - Cambio de Ciclo
    - Elemento de medición
      - Creación de elemento de Medición
      - Cambio de Dirección de Elemento de Medición

Figura 1: *Estructura de registros de casos de prueba*



Fuente: Imágenes de la ejecución de la simulación

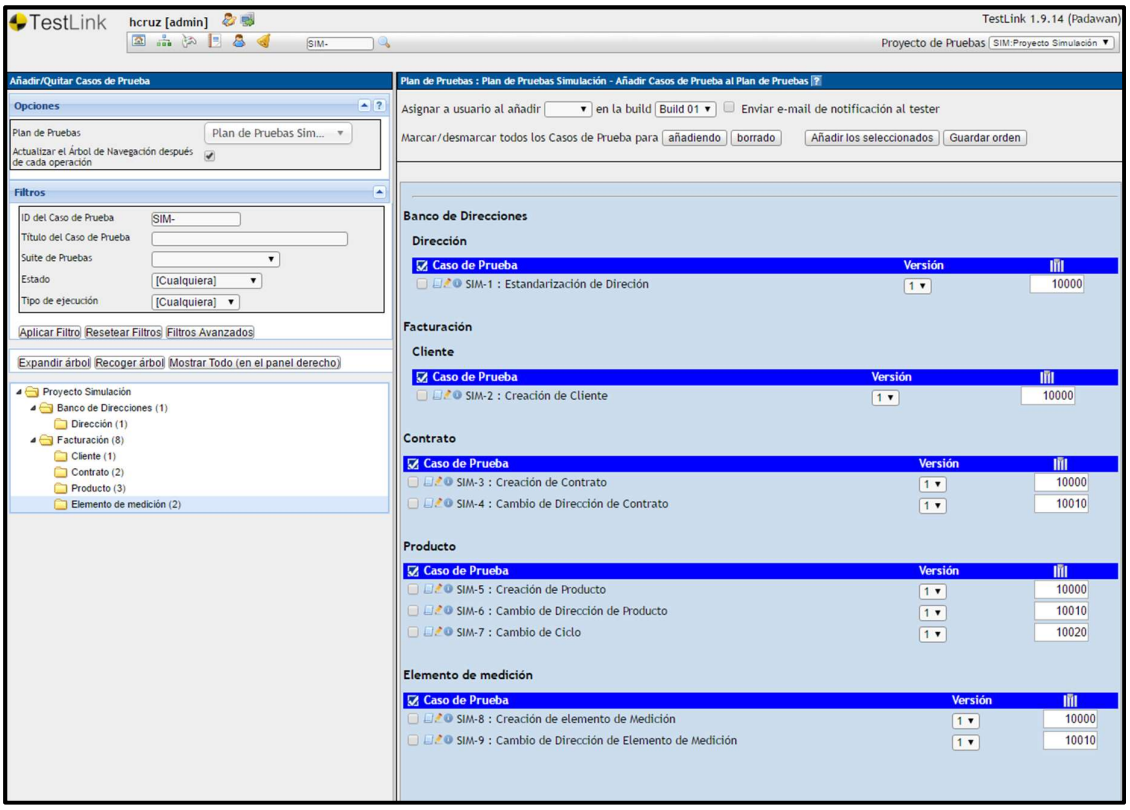
Figura 2*: Estructura de registros de casos de prueba – Especificación de Pruebas*



Fuente: Imágenes de la ejecución de la simulación

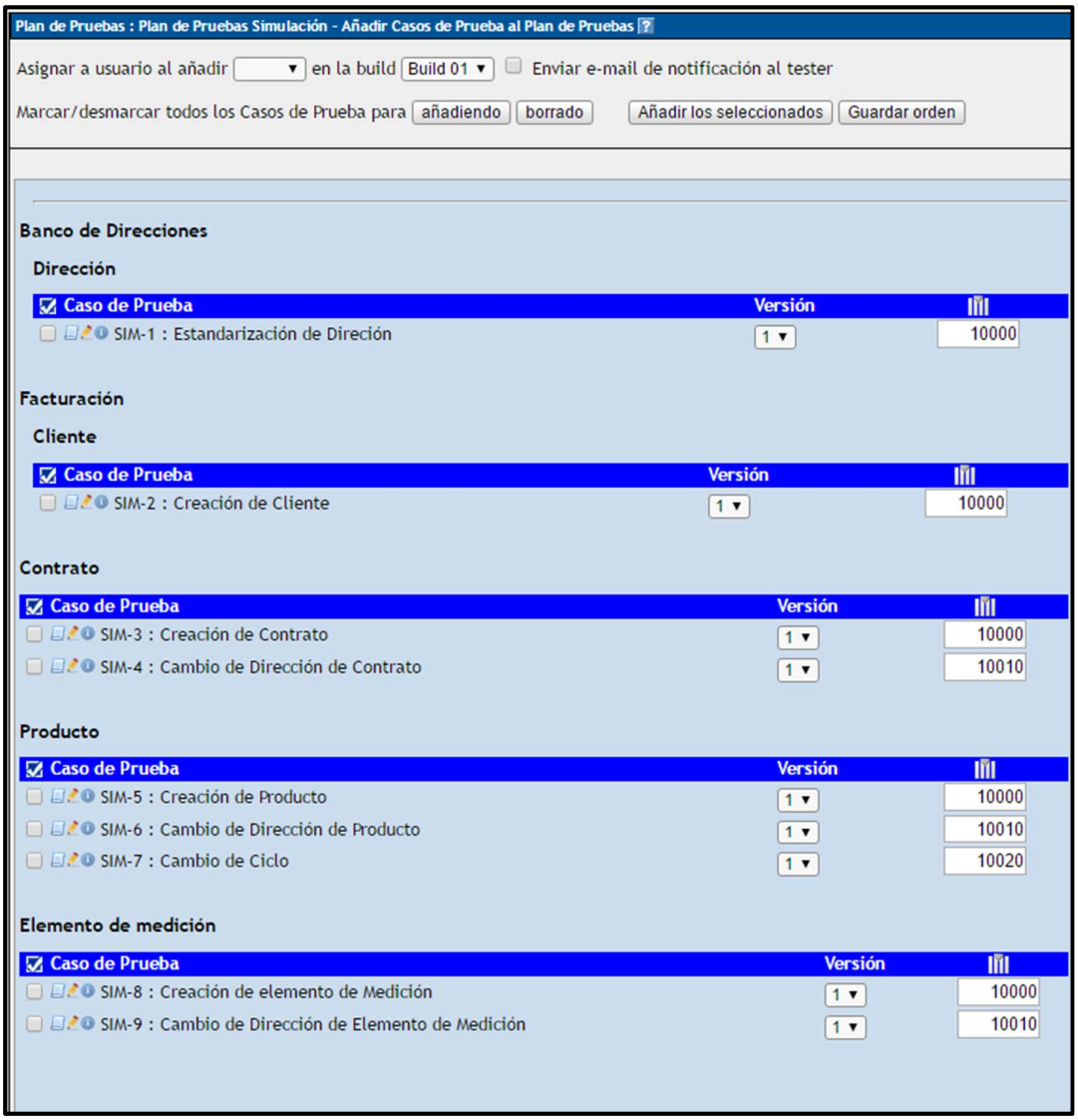
Una vez se han diligenciado los casos de prueba bajo un proyecto en TestLink, se procede a asociar los casos de prueba a un plan de pruebas (Ver figura 3 y 4). En este punto se asocian los casos de prueba al colaborador que vaya a ejecutar el caso.

Figura 3: *Asociación de casos de prueba a un plan de pruebas*



Fuente: Imágenes de la ejecución de la simulación

Figura 4: *Asociación de casos de prueba a un plan de pruebas - Detalle*



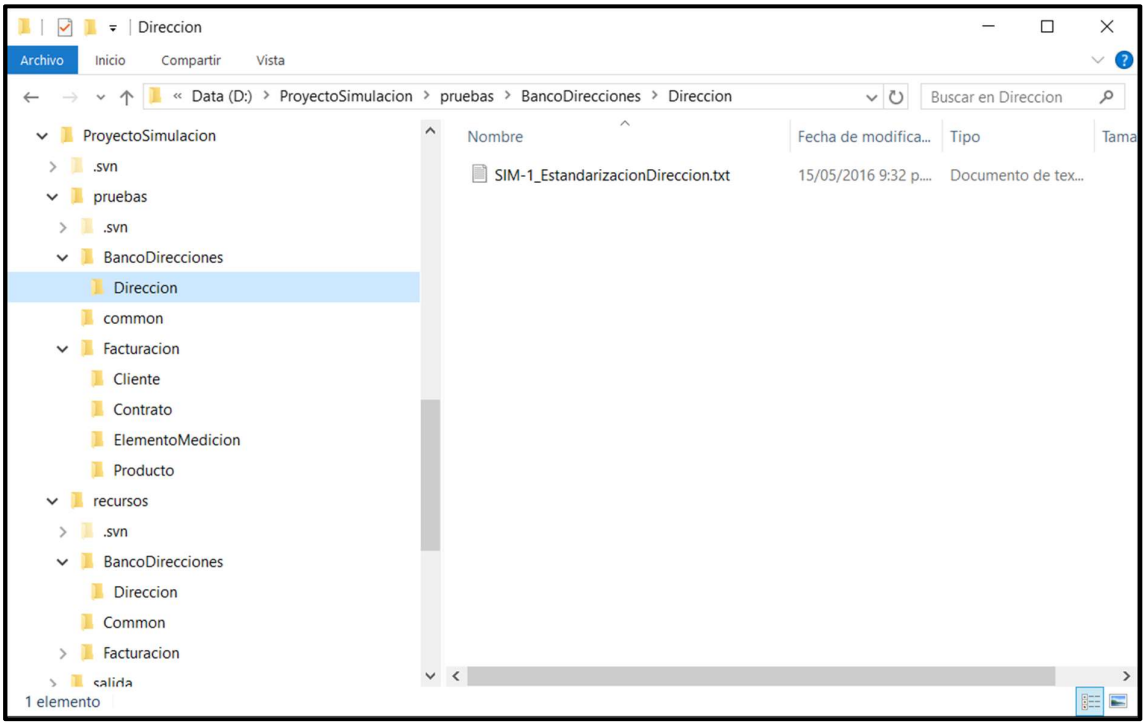
Fuente: Imágenes de la ejecución de la simulación

El ingreso de los casos de prueba en la herramienta TestLink no depende de si la ejecución se realizará forma manual o de forma automática, siempre se debe realizar el registro para tener un banco de casos de prueba y una trazabilidad de las ejecuciones realizadas con cada uno de ellos.

- **Codificación y ejecución de los casos de prueba**

Una vez se ha realizado el registro en TestLink, se procede a realizar la codificación de los casos.

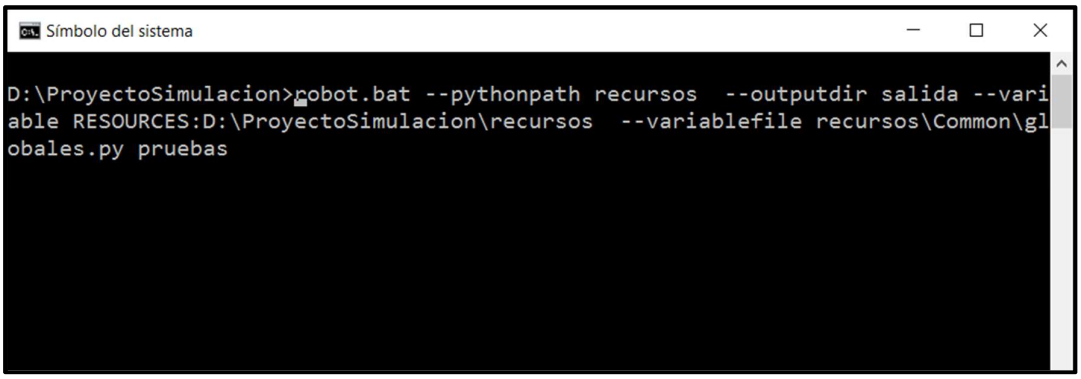
Figura 5: *Estructura de directorios para Robot Framework*



Fuente: Imágenes de la ejecución de la simulación

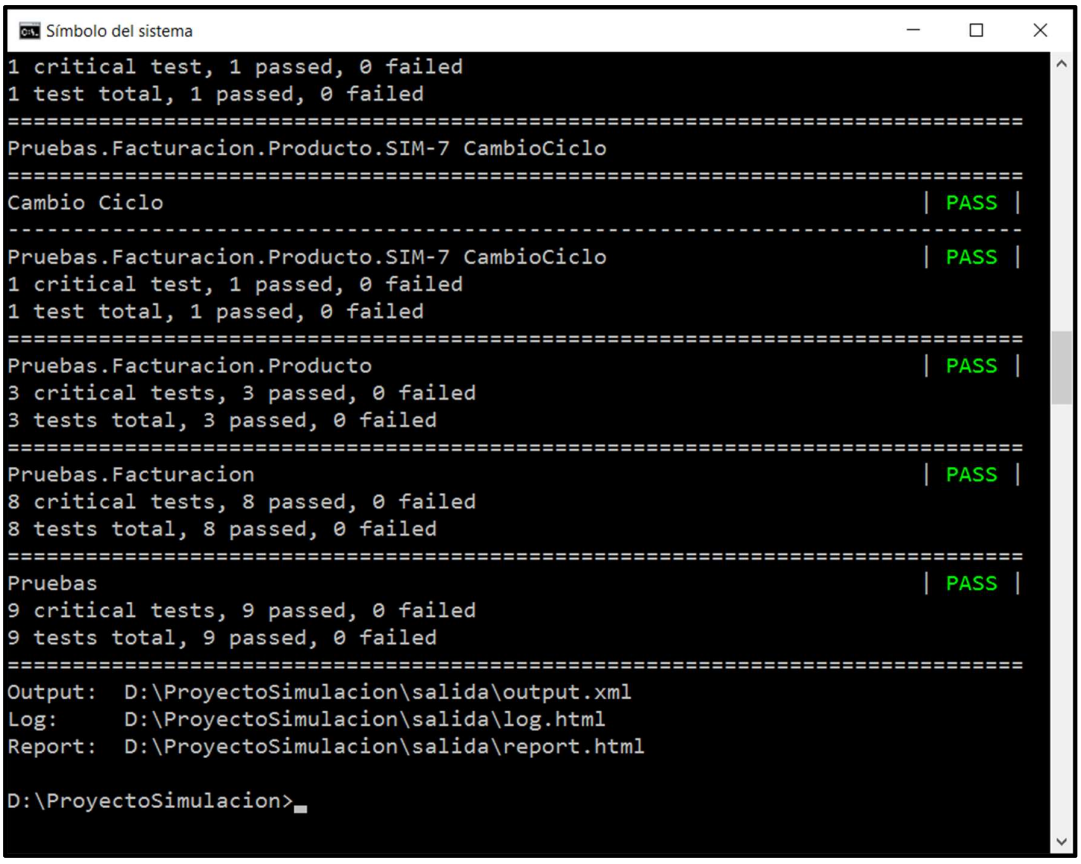
Al terminar de codificar cada uno de los casos que se tenía planeado hacer, se hace una ejecución de todos ellos, esto se realiza utilizando una Consola y ejecutando el comando con los parámetros necesarios.

Figura 6: *Ejecución por consola de Robot Framework*



Fuente: Imágenes de la ejecución de la simulación

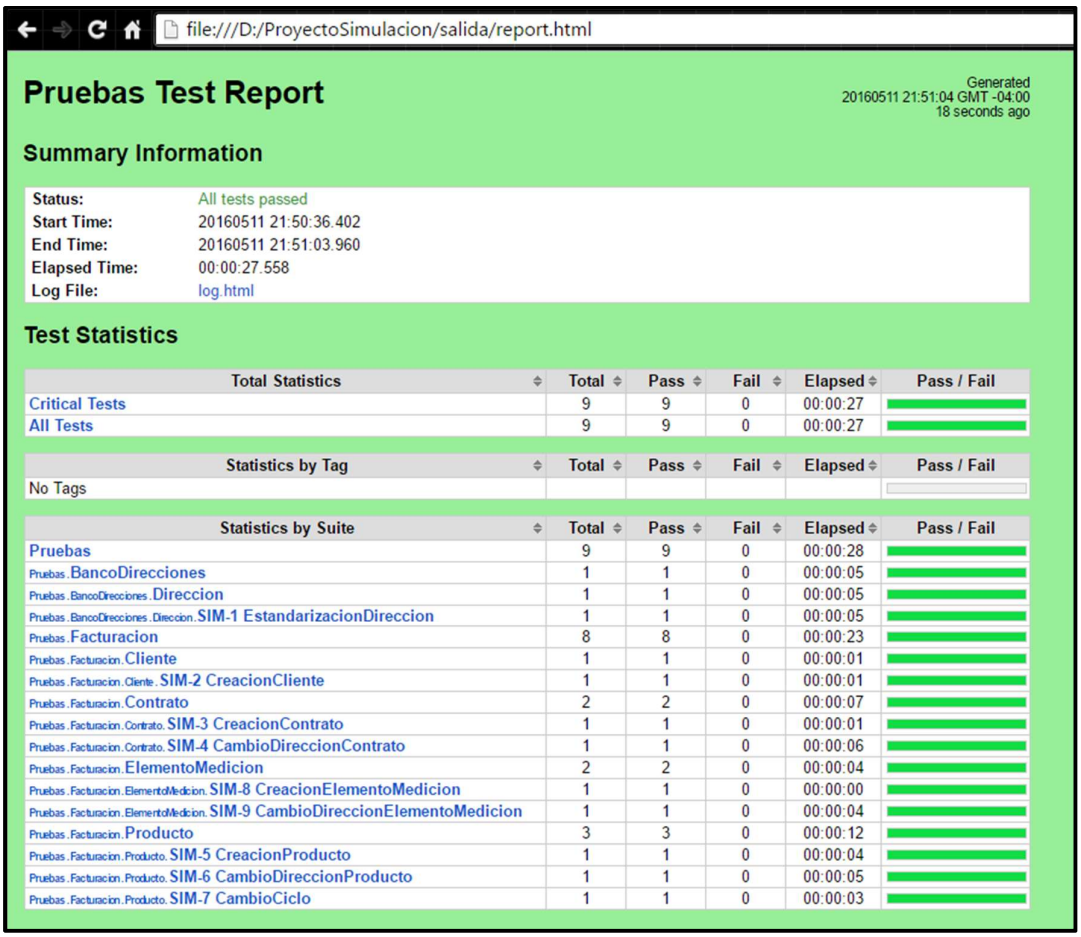
Figura 7: *Resultado de ejecución por consola de Robot Framework*



Fuente: Imágenes de la ejecución de la simulación

Una vez se ha terminado la ejecución se puede abrir el archivo de reporte y de log.

Figura 8: *Reporte de ejecución de las pruebas*



file:///D:/ProyectoSimulacion/salida/report.html

Pruebas Test Report

Generated
20160511 21:51:04 GMT -04:00
18 seconds ago

Summary Information

Status:	All tests passed
Start Time:	20160511 21:50:36.402
End Time:	20160511 21:51:03.960
Elapsed Time:	00:00:27.558
Log File:	log.html

Test Statistics

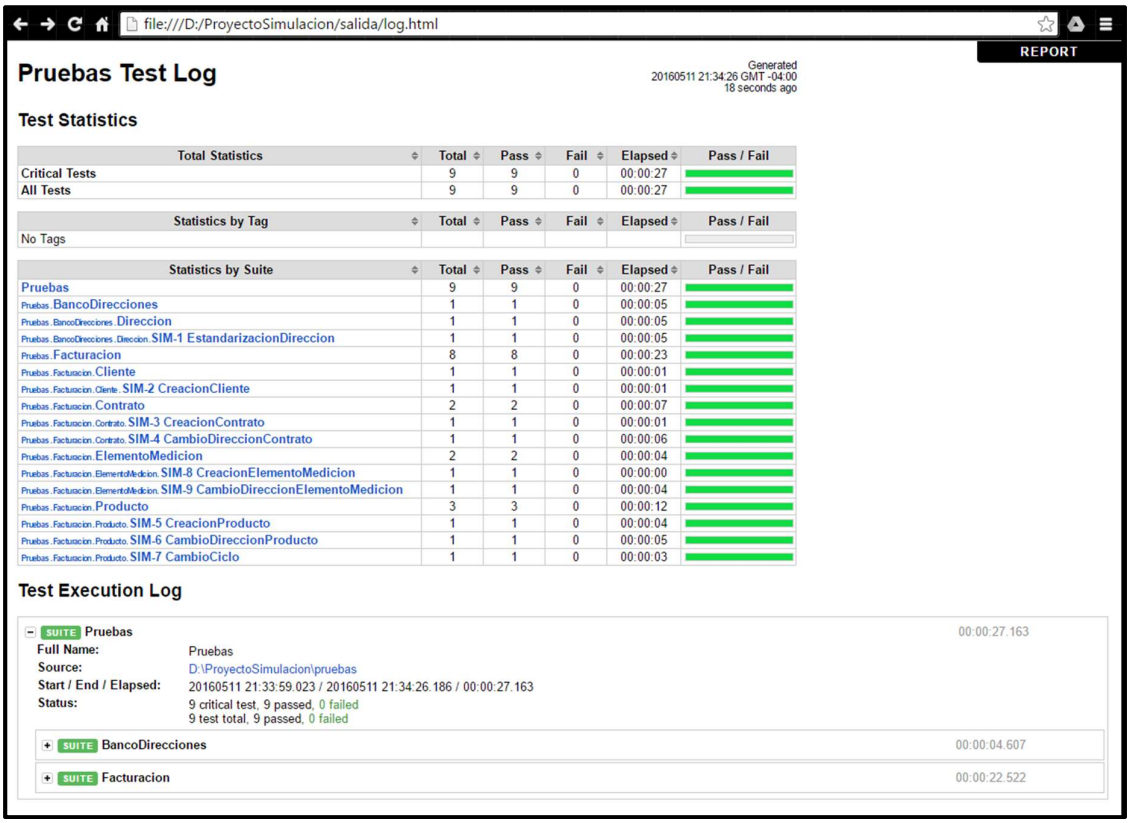
Total Statistics	Total	Pass	Fail	Elapsed	Pass / Fail
Critical Tests	9	9	0	00:00:27	
All Tests	9	9	0	00:00:27	

Statistics by Tag	Total	Pass	Fail	Elapsed	Pass / Fail
No Tags					

Statistics by Suite	Total	Pass	Fail	Elapsed	Pass / Fail
Pruebas	9	9	0	00:00:28	
Pruebas.BancoDirecciones	1	1	0	00:00:05	
Pruebas.BancoDirecciones.Direccion	1	1	0	00:00:05	
Pruebas.BancoDirecciones.Direccion.SIM-1 EstandarizacionDireccion	1	1	0	00:00:05	
Pruebas.Facturacion	8	8	0	00:00:23	
Pruebas.Facturacion.Cliente	1	1	0	00:00:01	
Pruebas.Facturacion.Cliente.SIM-2 CreacionCliente	1	1	0	00:00:01	
Pruebas.Facturacion.Contrato	2	2	0	00:00:07	
Pruebas.Facturacion.Contrato.SIM-3 CreacionContrato	1	1	0	00:00:01	
Pruebas.Facturacion.Contrato.SIM-4 CambioDireccionContrato	1	1	0	00:00:06	
Pruebas.Facturacion.ElementoMedicion	2	2	0	00:00:04	
Pruebas.Facturacion.ElementoMedicion.SIM-8 CreacionElementoMedicion	1	1	0	00:00:00	
Pruebas.Facturacion.ElementoMedicion.SIM-9 CambioDireccionElementoMedicion	1	1	0	00:00:04	
Pruebas.Facturacion.Producto	3	3	0	00:00:12	
Pruebas.Facturacion.Producto.SIM-5 CreacionProducto	1	1	0	00:00:04	
Pruebas.Facturacion.Producto.SIM-6 CambioDireccionProducto	1	1	0	00:00:05	
Pruebas.Facturacion.Producto.SIM-7 CambioCiclo	1	1	0	00:00:03	

Fuente: Imágenes de la ejecución de la simulación

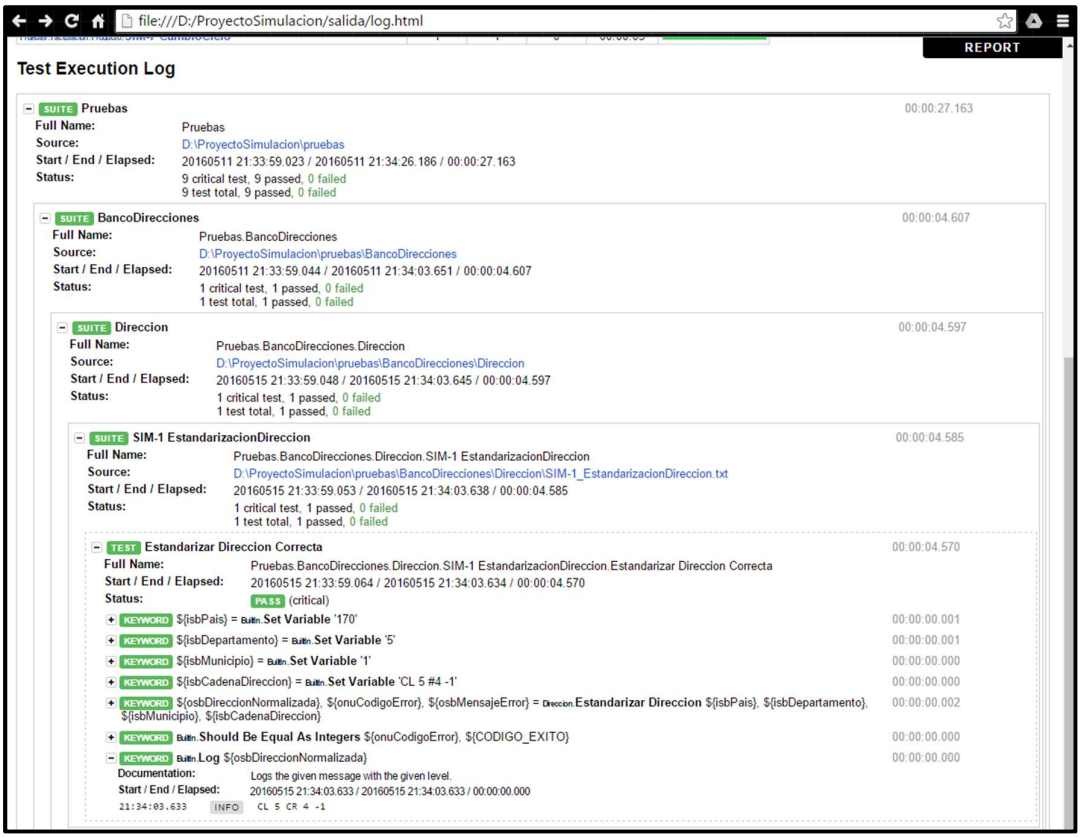
Figura 9: *Log de ejecución de las pruebas*



Fuente: Imágenes de la ejecución de la simulación

A continuación se muestra el detalle de cada uno de los pasos ejecutados para un caso de prueba

Figura 10: *Detalle de log de ejecución de las pruebas*



Fuente: Imágenes de la ejecución de la simulación

En la tabla 4 que se encuentra a continuación se ven cada uno de los datos recolectados dentro de la simulación. Aquí cabe aclarar:

- El registro en TestLink se debe realizar sin importar si el caso de prueba se va a ejecutar de forma automática o manual.
- El tiempo de ejecución manual corresponde al tiempo que le toma a un colaborador realizar la ejecución del caso de prueba de forma manual. Para ese tiempo no se tiene en cuenta tiempos de documentación de la prueba, sólo tiempo de preparación y ejecución. El tiempo fue establecido por el analista basado en la experiencia.
- El tiempo de codificación es el tiempo que tomó la construcción del caso de prueba en Robot Framework hasta llegar a la ejecución exitosa, esta construcción fue

realizada por parte de un ingeniero de desarrollo con poca experiencia en codificación de casos de prueba utilizando esta herramienta.

- El tiempo de ejecución automática es el tiempo que le tomó a Robot Framework ejecutar la codificación del caso de prueba. En el total se presenta el tiempo redondeado por Robot Framework, sin embargo, al final se muestra el total tanto redondeado como real.

Tabla 4: *Tabla de comparación ejecución manual vs automático*

Caso de Prueba	Registro en TestLink	Tiempo Ejecución Manual	Tiempo Codificación	Tiempo Ejecución Automática
SIM-1_Estandarización de dirección	20 Minutos	5 Minutos	30 Minutos	5 Segundos
SIM-2_CreaciónCliente	20 Minutos	15 Minutos	40 Minutos	1 Segundo
SIM-3_CreaciónContrato	20 Minutos	15 Minutos	40 Minutos	1 Segundo
SIM-4_CambioDirecciónContrato	20 Minutos	15 Minutos	30 Minutos	6 Segundos
SIM-5_CreaciónProducto	20 Minutos	15 Minutos	40 Minutos	4 Segundos
SIM-6_CambioDirecciónProducto	20 Minutos	15 Minutos	20 Minutos	5 Segundos
SIM-7_CambioCiclo	20 Minutos	15 Minutos	20 Minutos	3 Segundos
SIM-8_CreaciónElementoMedición	20 Minutos	15 Minutos	40 Minutos	1 Segundo
SIM-9_CambioDirecciónElemento	20 Minutos	15 Minutos	40 Minutos	4 Segundos
TOTAL	**180**	**125 Minutos**	**300 Minutos**	**30 Segundos (Redondeado)** **27 Segundos (Real)**

Fuente: Elaboración propia

Categoría: Proceso de ingeniería de software

Subcategoría: Aseguramiento y control de calidad

Indicador: Integración continua para los casos de pruebas automáticas

La integración continua dentro de un proyecto en desarrollo es muy importante ya que permite detectar fallos del software cuanto antes, para que así se puedan resolver de forma inmediata y tener versiones listas de forma continua para realización de las pruebas.

Teniendo en cuenta la información documentada de los Proceso de Desarrollo de Software que tiene Axede, se observa que no realiza integración continua a los desarrollos y es debido a que estos son realizados sobre aplicaciones propias del cliente, lo que genera que no se tenga control total sobre las versiones de los diferentes código fuente que se van ajustando o creando, imposibilitando la realización de las pruebas de forma continua. Con lo que sí cuenta Axede S.A. es con el proceso de creación y liberación de línea base, que consiste en tener un conjunto de especificaciones o producto de trabajo que son funcionales, los cuales sirven como base para un desarrollo posterior y que sólo pueden ser modificadas por medio de los procedimientos previamente establecidos; es decir, solo se controlan las versiones de los códigos fuentes de los objetos del cliente que se van modificando. Para este proceso de creación y liberación de línea base se utiliza la aplicación cliente TortoiseSVN.

En la entrevista realizada a P2, manifestó "*Considero que si es posible aplicar algo de la integración continua en las siguientes etapas del proceso de pruebas: Diseño de los casos de pruebas, con la utilización de la herramienta Testlink se podría contar un grupo de casos de prueba reutilizables para todo el proceso de pruebas; Ejecución de los casos de pruebas. Con la utilización de alguna herramienta de automatización, se podría obtener mejores resultados y así mismo tener mejor control de dichas pruebas."* ante la pregunta "*¿Considera que es posible aplicar alguna práctica de la integración continua en las actividades que realizas actualmente? Si es así ¿Cuáles?*". Teniendo en cuenta la respuesta de P2 se considera la utilización de Jenkins como software de código libre que permite realizar integración continua, y así tener control sobre los casos de pruebas desarrollados. Así mismo, en combinación con TestLink, se programaría las ejecuciones automáticas de los casos de

prueba cada vez que el desarrollador libere un entregable, posterior a su despliegue que se realice automáticamente.

7. Propuesta de ejecución de casos de prueba manuales y automatizados en los desarrollos de software de Axede S.A.

En Axede, como se ha manifestado en la presente investigación, no cuenta con un esquema de trabajo o metodología, ni herramientas que permitan la ejecución de casos de prueba de manera automatizada sobre los desarrollos de software. Todos los casos de prueba son ejecutados de forma manual en la última etapa del desarrollo, haciendo que el analista de prueba sólo conozca las soluciones a las necesidades de los clientes a último momento y no tenga una contextualización completa para poder desarrollar los casos de prueba de forma eficiente. Por tal motivo, a continuación se realiza una propuesta para la implementación de un esquema de trabajo para la ejecución automática de casos de prueba:

- Se propone que el analista de pruebas esté involucrado en todas las etapas del desarrollo de software para que así tenga una visión más integral de las necesidades del cliente. De esta manera, el analista de pruebas podría elaborar los casos de prueba abarcando la mayor cantidad de caminos posibles y no dejar por fuera opciones que puedan generar un error en el futuro. Esto implica realizar la modificación al proceso de pruebas que tiene Axede S.A. en la actualidad.
- Se debe disponer de un servidor con sistema operativo Linux para realizar la instalación y configuración de las herramientas TestLink, Robot Framework y Jenkins, los cuales permitirían registrar casos de prueba y ser ejecutados de manera automática, aplicando la integración continua.
- Realizar la sensibilización y capacitación sobre la herramienta TestLink a los ingenieros de desarrollo y a los analistas de pruebas para que tengan una visión detallada en cuanto a su configuración, utilización y beneficios, buscando que no se presente la resistencia al cambio.
- Seleccionar una población de los casos de prueba que existen actualmente para definirles el proyecto, módulo y funcionalidad; al igual que identificar cuáles son los casos de prueba más repetitivos, los que presentan pocas modificaciones y los que son casos de prueba básicos que no tienen una gran complejidad. Con todo lo anterior se busca tener unos criterios al momento de definir los casos que harán parte de la automatización inicial y para esa labor será de gran apoyo los analistas

de pruebas puesto que estos tienen un amplio conocimiento sobre los casos de prueba.

- Realizar la definición de estructura para el registro de los casos de prueba en la herramienta TestLink, teniendo en cuenta la matriz de ítems de la configuración de los proyectos existentes en Axede. S.A, ya que esto puede impactar los lineamiento del proceso de maduración CMMI. Estos posibles impactos pueden ser detectados o mitigados por los colaboradores del área de procesos de la compañía los cuales están velando por su cumplimiento y mejoras en el proceso.

  La estructura propuesta en TestLink para la organización de los casos de prueba al momento de registrarlos es:

  <Proyecto>
  - <Modulo_01>
    - <Funcionalidad_01>
      - <Casos de Prueba_01>
      - <Casos de Prueba_02>
    - <Funcionalidad_02>
      - <Casos de Prueba_01>
  - <Modulo_02>
    - <Funcionalidad_01>
      - <Casos de Prueba_01>
  - <Modulo_N>
    - <Funcionalidad_01>
      - <Casos de Prueba_01>
      - <Casos de Prueba_N>

  Donde:

  - ✓ <Proyecto> es el proyecto al que pertenece el caso de prueba que se va a crear o modificar.
  - ✓ <Modulo_N> es el módulo al que pertenece el caso de prueba.
  - ✓ <Funcionalidad_N> es la funcionalidad a la que pertenece el caso de prueba.
  - ✓ <Caso de Prueba_N> es el caso de prueba a crear.

  Al momento de utilizar los casos de prueba dentro de un ciclo, se propone la siguiente organización:

Plan de Pruebas
- Caso de prueba

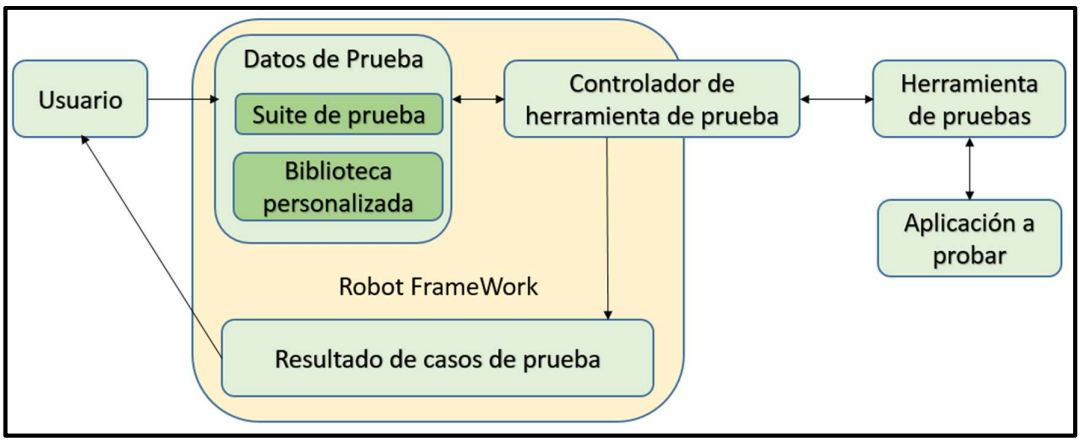
Donde:

- ✓ <Plan de Pruebas> es la agrupación que hace TestLink a un conjunto de casos de prueba a ejecutar.
- ✓ <Caso de Prueba> es el caso de prueba a ejecutar.

- Se debe realizar reuniones con las personas involucradas en el proceso de aseguramiento de calidad y procesos corporativos, para validar si se puede llegar a una unificación de una plantilla en Excel para el registro de casos de prueba que posteriormente serán importados a TestLink. El objetivo de las reuniones busca que esta plantilla aplique para todos los proyectos que maneja la compañía y no debe afectar los procesos institucionales establecidos.
- Realizar capacitación en la construcción de las reglas de automatización para el Robot Framework en la cual se recomienda impartir el conocimiento a los ingenieros de desarrollo y los analistas de prueba, permitiendo un trabajo en conjunto de los dos roles. De esta manera los ingenieros de desarrollo pueden crear reglas de automatización básicas y entregarlas al analista de pruebas para que éste las enriquezca adicionando o eliminando condiciones.
- Definir un estándar para la construcción de las reglas de automatización en Robot Framework para permitir hablar un mismo lenguaje al momento que cualquier colaborador, ingeniero de desarrollo o analista de pruebas, deba hacer alguna modificación en dichas reglas; así mismo, en el momento que ingrese una nueva persona a la compañía pueda entender y estar más familiarizado con las reglas de automatización existentes.
- Establecer una medida de complejidad (baja, media, alta) de los casos de pruebas para así identificar cuáles no se pueden automatizar y es necesario continuar su ejecución de manera manual. Se debe tener en cuenta que todos los casos no se puede automatizar ya que pueden surgir algunos muy complejos donde es determinante la intervención humana, por ejemplo la impresión de un reporte que se envía a una impresora.

- Realizar la integración de Mantis-bugtraker, el sistema de incidencias de la compañía, con la nueva herramienta Testlink para permitir una mayor visibilidad entre la relación de errores reportados con los casos de prueba registrados en TestLink y así poder realizar seguimientos más detallados.
- Para la utilización de Robot Framework se propone la siguiente arquitectura:

Figura 11: *Propuesta de arquitectura de Robot Framework*



Fuente: Elaboración propia

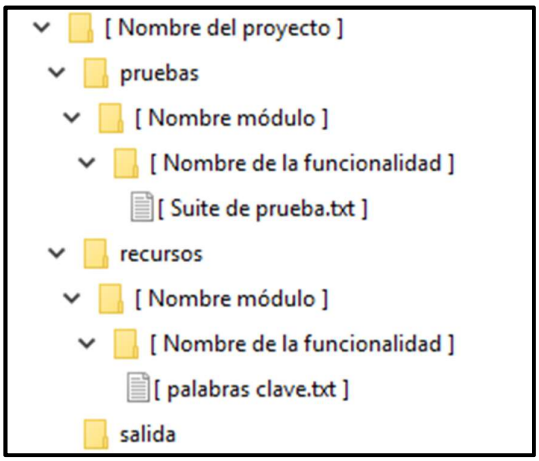
Donde:

- ✓ **Datos de prueba:** Es la configuración de las pruebas, el más cercano a lo que la mayoría de los analistas de prueba. Se compone de archivos de prueba, datos y carpetas, así como los contenidos de los datos que se utilizan para la ejecución de la prueba.
- ✓ **Resultados de casos de prueba:** son las respuestas finales de las pruebas y permiten determinar los resultados de las pruebas, así como, para evaluar las diversas partes de la prueba.
- ✓ **Robot Framework:** Esta es la herramienta básica que realiza el trabajo pesado al momento de ejecutar las pruebas.
- ✓ **Controlador de herramienta de prueba:** Esto proporciona la comunicación entre el framework y la herramienta a probar. Este se hace a la medida de acuerdo a las necesidades específicas.

- ✓ **Herramienta de pruebas:** Este es el software utiliza para realizar las pruebas de aceptación.
- ✓ **Aplicación a probar (sistema bajo prueba):** Este es el software puesto a prueba para su aceptación por el cliente o el usuario final.

La estructura de los proyectos que se sugiere al trabajar con Robot Framework es:

Figura 12: *Estructura de proyectos en Robot Framework*



Fuente: Elaboración propia

Donde:

- ✓ "[Nombre del proyecto]" es la carpeta con el nombre del proyecto al cual pertenece la implementación de los casos de prueba.
- ✓ "[Nombre módulo]" es la carpeta con el nombre del módulo al cual pertenece la implementación de los casos de prueba
- ✓ "[Nombre de la funcionalidad]" es la carpeta con el nombre de la funcionalidad al cual pertenece la implementación de los casos de prueba.
- ✓ "[Suite de prueba.txt]" es el archivo con el suite de pruebas con los casos de prueba automatizados agrupados por características similares.
- ✓ "[palabras clave.txt]" es el archivo con las palabras clave definidas por el proyecto agrupados por características similares.
- ✓ "salida" es la carpeta que va a contener los resultados de la ejecución de las pruebas.

- Con la utilización de Jenkins es posible lograr una integración continua en los proyectos en donde se pueda definir dicha práctica lo cual permitiría mejorar la calidad del producto y de las personas que están trabajando como equipo. Jenkis realizará las posibles tareas de ejecutor de pruebas o compilador de código fuente, esto se puede definir según la necesidad de cada proyecto.
- Es importante generar un indicador que calcules el valor de la automatización, especialmente si es la primera vez que un proyecto comienza a utilizar la automatización de pruebas. Los posibles indicadores para esta propuesta son:

El Porcentaje Automatizable o Índice de Automatización:

$$PA(\%) = \frac{ATC}{TC}$$

- ✓ PA = Porcentaje Automatizable
- ✓ ATC = Número de casos de prueba automatizables
- ✓ TC = Número total de casos de prueba

En la automatización de casos de prueba se debe considerar qué se puede automatizar y qué no, de esta manera se puede determinar el porcentaje de automatización de casos de prueba de un proyecto.

El Progreso de Automatización:

$$AP(\%) = \frac{AA}{ATC}$$

- ✓ AP = Progreso de automatización
- ✓ AA = Número de casos de prueba automatizado
- ✓ ATC = Número de casos de prueba automatizables

El progreso de automatización se refiere al número de pruebas que se han automatizado como un porcentaje de todos los casos de pruebas automatizados,

esto significa que también se están realizando las reglas de automatización para los casos de pruebas que se quieren automatizar cuyo objetivo final es 100% de casos automatizables. Esto se puede realizar en varias fases y es importante establecer una meta la cual debe determinar los plazos que tardaran esa automatización.

El Progreso en casos de prueba:

$$TP = \frac{TC}{T}$$

- ✓ TP = Progreso en casos de prueba
- ✓ TC = Número de casos de prueba (intento o completado)
- ✓ T = Alguna unidad de tiempo (días / semanas / meses, etc)

El Progreso en el caso de la prueba simplemente se puede definir como el número de casos de prueba intentados o terminados con el tiempo.

8. Conclusiones

En la realización de pruebas la principal razón es el tiempo, con las pruebas automatizadas se puede disminuir el tiempo de las pruebas. Además, al automatizar las tareas comunes que no requieren de inteligencia humana, los analistas de pruebas pueden dedicar mayor tiempo a pruebas más críticas y rutas más elaboradas dejando las rutas básicas a las pruebas automatizadas.

Siempre hay que analizar, cuál es el costo y cual el beneficio de la manera que enfrentemos las pruebas que vamos a realizar (manual o automatizado) teniendo en cuenta el conocimiento de la herramienta a utilizar, conocimientos en programación, tiempo de las pruebas, cantidad de ciclos de prueba para así garantizar si un proyecto de automatización puede ser rentable para una compañía.

La realización de pruebas debe ser efectivo, localizando la mayor cantidad de errores de forma eficiente, ejecutando mayor cantidad de casos de prueba en el menor tiempo y con menos costos. Esto se podría ver reflejado mediante la automatización, el cual consiste en aplicar técnicas y múltiples herramientas para ejecutar pruebas sin intervención humana o con la ayuda mínima. Es por ello que las tareas de automatización de pruebas empiezan ser significativas dentro de las áreas de los sistemas.

9. Recomendaciones

Se debe tener en cuenta la creación de una campaña de capacitación y sensibilización para la utilización de esta herramienta al interior de la compañía.

Para la implementación de la automatización como mínimo se deben tomar roles de ingenieros de desarrollo y analista de pruebas que tenga un perfil Senior ya que estos tienen una mayor experiencia y visión de los proyectos que actualmente maneja Axede S.A pudiendo tomar decisiones más acertadas a la hora de seleccionar y construir reglas de automatización y selección de casos de prueba.

En la elección del servidor a utilizar para el despliegue de las herramientas de automatización se debe tener en cuenta que este servidor debe ser utilizado por todos los proyectos existentes en la compañía, lo que podría entrar en conflicto al prestar servicios directos en clientes que tengan restricciones por políticas de seguridad y que impidan la conexión de servidores que no estén en su red. Por tanto, se debe buscar una alternativa para poder realizar pruebas automáticas a clientes específicos donde se pueda utilizar los servicios que tiene la compañía Axede S.A con la plataforma Cloud Azure de Microsoft.

Referencias

Albaladejo, X. (s.f) *Qué es SCRUM.* Proyectos agiles.org. Recuperado de https://proyectosagiles.org/que-es-scrum/

Ambler, S. (2013). *Introduction to Test Driven Development (TDD).* Agiledata.org. Recuperado de http://agiledata.org/essays/tdd.html

Bernal, C. A. (2010), *Metodología de la Investigación.* Bogotá, Colombia. Pearson.

Ble, C. y Beas, J. M. (2010). *Diseño Ágil con TDD.* Recuperado de http://www.carlosble.com/downloads/disenoAgilConTdd_ebook.pdf

Canales Cerón M. Metodologías de la investigación social. Santiago: LOM Ediciones; 2006. p. 163-165.

Corbin, J. & Strauss, A. (2008). *Basics of qualitative research: Techniques and procedures for devel- oping grounded theory* (3rd ed.). Thousand Oaks, CA: Sage.

Fowler Martin. (2006). *Continuous Integration*. Recuperado de http://martinfowler.com/articles/continuousIntegration.html

Gonzalez, L. D. (2015). *Jenkins CI – Qué es y porqué usarlo* [Web log post]. Recuperado de http://davidg-tech.blogspot.com.co/2013/01/jenkins-ci-que-es-y-porque-usarlo.html

Groder, K. (2015). *Improving Software Quality: Nine Best-Practices for Test Automation.* Recuperado de http://www.intervise.com/wp-content/uploads/2015/01/Best_Practices_Test_Automation.12.14.pdf

Hernández R., Fernández, C. y Baptista P. (2010). *Metodología de la Investigación. México D.F., México*. McGraw-Hill.

Johnson, D. (2011 enero). TechTarget. *Test automation: When, how and how much*. Recuperado de http://searchsoftwarequality.techtarget.com/tip/Test-automation-When-how-and-how-much

Laurén, R. (2015 octubre). *Investigating GUI test automation ROI: An industrial case study*. Recuperado de http://www.idt.mdh.se/utbildning/exjobb/files/TR1779.pdf

Llamosa, R. y Estrada, L. Y. (2010). *Evaluación de Pymes de Software Colombianas bajo el modelo CMMI - Dev En El Marco Del Proyecto RCCS*. Recuperado de http://www.uelbosque.edu.co/sites/default/files/publicaciones/revistas/cuadernos_latinoamericanos_administracion/volumenVI_numero11_2010/evaluacion_pymes_software_colombianas_modelo_cmmi_dev_proyecto_rccs.pdf

Lopez-Mancisidor, A. (2003 noviembre). *¿Por qué invertir en la automatización de pruebas Software?. Rational XDE Tester*. Recuperado de https://www.ati.es/IMG/pdf/IBM.pdf

Mintic (2015 marzo). MINTIC. *Colombia líder en la región en la producción de software de calidad*. Recuperado de http://www.mintic.gov.co/portal/604/w3-article-8571.html

Pressman, R. S. (2010). Ingeniería de software. Un enfoque práctico. México D.F., México: McGraw-Hill

Restrepo, P. (2015 marzo). El potencial del software colombiano en el mundo. *La Republica*. Recuperado de http://www.larepublica.co/el-potencial-del-software-colombiano-en-el-mundo_228816

Robot Framework (2016). Robot Framework. Recuperado de http://robotframework.org/

QAustral S.A. (2006). *Manual de usuario de Testlink*. Recuperado de http://qaustral.com.ar/sitio/manual-de-usuario-de-testlink/

Shanon y Johaness (1976). *Systems Simulation: The Art and Science Recuperado de http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4309432*

Smartbear (2016) Automated Testing Best Practices and Tips. Recuperado de: https://smartbear.com/learn/automated-testing/best-practices-for-automation/

TestObject (2015). *Test Automation: Best Practices*. Recuperado de https://testobject.com/wp-content/uploads/Test_Automation-Best_Practices.pdf

Anexos

Anexo N° 1. Entrevista N°1

CORPORACIÓN UNIVERSITARIA MINUTO DE DIOS
FACULTAD DE EDUCACIÓN
ESPECIALIZACIÓN EN GERENCIA DE PROYECTOS

Proyecto:

Propuesta de automatización de casos de prueba para aseguramiento de calidad en el desarrollo de software

Entrevista

1. ¿Cuál es la metodología que se maneja actualmente en el proceso de desarrollo de software de Axede S.A?

Actualmente Axede S.A. ejecuta su proceso de desarrollo bajo una metodología propia, definida a nivel de los servicios que presta de Outsourcing, Desarrollo a la medida, y Adaptativo Ágil; bajo las guías de buenas prácticas de Modelos de Madurez de procesos como CMMI DEV – SVC, y estándares PMI.

Adicionalmente a ello, la metodología establecida para Desarrollo a la medida ha adoptado algunos lineamientos RUP, y la de Marcos Ágiles lineamientos SCRUM.

2. ¿Se ha considerado el cambio o la definición de otra metodología de desarrollo?

A lo largo del desarrollo de la Unidad de Software y de las variaciones que a nivel de negocio se han presentado con respecto a las tendencias del mercado, y a los contratos que se han ejecutado, Axede S.A. ha definido y redefinido sus procesos de desarrollo y la aplicación de estos, modificando y adaptándolos a las necesidades que a nivel de servicios debemos

cumplir, bajo las guías de buenas prácticas de Modelos de Madurez de procesos como CMMI DEV – SVC, y estándares PMI.

3. ¿Qué opinión tiene sobre la etapa de control de calidad en el proceso de desarrollo de software actual?

Axede S.A. tiene definido su proceso de calidad a nivel de verificación y validación a lo largo del ciclo de vida del Software, bajo las guías de buenas prácticas de Modelos de Madurez de procesos como CMMI DEV – SVC y estándares PMI, el cual se implementa en los servicios de Desarrollo a la medida Adaptativo Ágil, y Outsourcing (con las adaptaciones necesarias debido a la naturaleza de dicho servicio).

Actualmente la compañía cuenta en una gran proporción con servicios Insourcing y Outsourcing, para los cuales no se aplica este proceso de calidad o se aplica con varias adaptaciones (dependiendo del tipo de requerimiento que se atienda en el servicio y de las condiciones contractuales que se presenten en los procesos licitatorios); respectivamente. Esta situación, conlleva a que tanto las actividades de planeación, aseguramiento y control de calidad, sean más complejas en el momento de definirse e implementarse.

Debido a la complejidad descrita en el párrafo anterior, el proceso de control de calidad para los servicios de Outsourcing en especial, requiere de un alto nivel de adaptaciones, para que sus mediciones y la evaluación de estas permitan identificar y aplicar mejoras continuas al proceso.

Como todo proceso, la aplicación del proceso de calidad definido en Axede S.A. puede tener algunas mejoras enfocadas a la utilización de herramientas que permitan sistematizar su planeación, diseño, ejecución, cierre, gestión, reutilización y automatización, lo cual permite facilitar y optimizar las actividades propias del control de calidad.

4. De las mejores prácticas existentes para la automatización de pruebas, ¿Conoce alguna que se está aplicando actualmente en las pruebas manuales?

Actualmente en Axede S.A. solo se están realizando pruebas manuales. La compañía ha realizado algunos pilotos enfocados a la automatización de las pruebas manuales, pero estos

pilotos no se han implementado en los servicios, ni se han adoptado de manera oficial en el proceso de calidad.

5. ¿Tiene conocimiento si la empresa Axede S.A ha considerado la implementación de un control de calidad ayudado por casos de prueba automatizados? De ser así ¿Por qué no se ha implementado?

Axede S.A. lo ha considerado en varias ocasiones, pero no se realizado aún ninguna implementación al respecto, debido a que no se ha dado al proceso de calidad este foco a nivel de procesos.

6. ¿Cree que la utilización de herramientas de código libre pueden ser una buena alternativa para dar inicio a la automatización de pruebas?

Personalmente considero que sí, pero la decisión de adoptarlas o no dentro de la compañía, debe ir acompañada de una investigación y evaluación de estas, de manera tal que permita realizar una toma estructurada de decisiones sobre sí estas herramientas tienen correspondencia con la estrategia organizacional.

7. Dentro del conocimiento que tiene sobre automatización de casos de prueba ¿Considera que su implementación traería beneficios para Axede S.A?

Personalmente considero que sí, ya que esto nos ayudaría a ser más efectivos y productivos en el momento de controlar la ejecución de pruebas, la comparación entre los resultados obtenidos y los resultados esperados, y la reutilización de estos casos de pruebas en la ejecución de nuevos escenarios de pruebas.

8. ¿Considera que Axede está dispuesta a realizar una inversión para la realización de pruebas automatizadas, teniendo en cuenta que esta práctica no refleja beneficios en el corto plazo?

Personalmente considero que sí.

9. De las herramientas existentes en el mercado ¿Cuáles conoce para la automatización de pruebas?

Algunas de las que tengo referencia las listo a continuación:

- IBM Rational Functional Tester
- IBM Rational Performance Tester
- HP Unified Functional Testing
- Selenium
- Testlink

10. ¿Considera que es posible tener una integración continua para los proyectos actualmente manejados por Axede?

Para algunos que actualmente conozco sí.

Anexo N°2. Entrevista N°2

CORPORACIÓN UNIVERSITARIA MINUTO DE DIOS
FACULTAD DE EDUCACIÓN
ESPECIALIZACIÓN EN GERENCIA DE PROYECTOS

Proyecto:

Propuesta de automatización de casos de prueba para aseguramiento de calidad en el desarrollo de software

Entrevista

1. En la metodología de desarrollo que tiene actualmente Axede ¿Cree que deberían aplicarse algún cambio en el desarrollo de sus actividades? Si es así ¿Cuáles cambios?

Dentro de la metodología de desarrollo actualmente definida por Axede, considero que la empresa no le da la importancia que realmente se merece al proceso de pruebas. Por lo tanto, considero que se podrían adoptar los siguientes cambios.

Se debería tener en cuenta al analista de pruebas desde el inicio de la construcción de un producto.

- Capacitaciones sobre el negocio.
- Capacitación en las diferentes herramientas o aplicativos.

2. Dentro de los casos existentes de pruebas en Axede S.A, ¿Cree que todos se pueden llevar a la automatización?

Considero que no en todos los casos las pruebas se pueden llevar a Automatización, ya que en algunos casos por el tipo de negocio e información, se requiere de la malicia del tester y de su objetividad para lograr la completitud del producto y la calidad del mismo.

3. ¿En su vida profesional en alguna ocasión ha implementado casos de prueba automatizados?

Hasta el momento la mayoría de aplicativos a los cuales les he realizado calidad no han requerido de este tipo de pruebas, de otro lado, los aplicativos que lo han requerido, la empresa no ha estado dispuesto a invertir en dicho propósito.

4. ¿Considera que la automatización de las pruebas es beneficioso dentro del proceso de desarrollo de software? Si es así, ¿Qué beneficios podrían generar?

Personalmente considero que es un beneficio importante para la empresa implementare la automatización de las pruebas, ya que en algunos casos hay aplicativos que por su tipo de negocio y flujo de información lo requieren.

Beneficios:

- Permite mejorar considerablemente las pruebas.
- Permite realizar un mayor número de pruebas.
- Proporciona mayor confiabilidad en las pruebas y sus resultados
- Permitiría aplicar pruebas complicadas.
- Mayor facilidad y agilidad en las pruebas de regresión.

5. De las herramientas existentes en el mercado ¿Cuáles conoce para la automatización de pruebas?
   - Selenium
   - Jmeter

6. ¿Cree que la utilización de herramientas de código libre pueden ser una buena alternativa para dar inicio a la automatización de pruebas?

Si considero que una buena alternativa, ya que en la mayoría de las empresas, éstas no invierten sus recursos en el proceso de pruebas y si contratan el servicio con un tercero para

ejecutarlas, casi siempre no están dispuestos a invertir en la automatización de dichas pruebas, es decir, en la compra de licencias de herramientas de automatización.

7. ¿Utiliza alguna técnica en el diseño de los casos de prueba?
    - Técnica de Caja Blanca
    - Técnica de Caja Negra
    - Técnica basada en la experiencia.

8. ¿Considera que es posible aplicar alguna práctica de la integración continua en las actividades que realizas actualmente? Si es así ¿Cuáles?

Considero que si es posible aplicar algo de la integración continua en las siguientes etapas del proceso de pruebas.

- Diseño de los casos de pruebas, con la utilización de la herramienta Testlink se podría contar un grupo de casos de prueba reutilizables para todo el proceso de pruebas.
- Ejecución de los casos de pruebas. Con la utilización de alguna herramienta de automatización, se podría obtener mejores resultados y así mismo tener mejor control de dichas pruebas.