DE LAS PRUEBAS MANUALES A LAS AUTOMÁTICAS DESDE ASD EN EL ÁREA DE TECNOLOGÍA DE LA UNIVERSIDAD MARIANA

GEANCARLO DÍAZ MARÍN

UNIVERSIDAD MARIANA
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA DE SISTEMAS
SAN JUAN DE PASTO
2024

DE LAS PRUEBAS MANUALES A LAS AUTOMÁTICAS DESDE ASD EN EL ÁREA DE TECNOLOGÍA DE LA UNIVERSIDAD MARIANA

GEANCARLO DÍAZ MARÍN

Trabajo de grado como requisito para obtener el título de Ingeniero de Sistemas

Asesor
Giovanni Albeiro Hernández Pantoja

UNIVERSIDAD MARIANA
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA DE SISTEMAS
SAN JUAN DE PASTO
2024

ARTÍCULO 71
REGLAMENTO DE INVESTIGACIONES
UNIVERSIDAD MARIANA

"Los conceptos, afirmaciones y opiniones emitidos en el Trabajo de Grado son responsabilidad única y exclusiva del (los) Educando (s). "(1)

NOTA DE ACEPTACIÓN:

_____

_____

_____

_____

_____

_____

_____

Firma Presidente del jurado

_____

Firma del Jurado

_____

Firma del Jurado

San Juan de Pasto, 06 de Junio de 2024.

DEDICATORIA

A mis padres, quienes siempre creyeron en mí y sacrificaron tanto para brindarme las oportunidades que hoy tengo frente a mí. A mis hermanos, por su aliento y comprensión en los momentos más difíciles. A mis amigos, quienes han estado compartiendo este viaje conmigo. Este logro también es suyo, y les estoy eternamente agradecido.

A todos aquellos profesores cuyas enseñanzas han iluminado mi camino y cuyo ejemplo ha sido mi guía en la búsqueda del conocimiento. Agradezco profundamente su dedicación y sabiduría, la cual ha enriquecido este trabajo y mi vida en general.

A mi yo del pasado, aquel que dudaba de sus propias capacidades y temía enfrentarse a los desafíos. Agradezco cada obstáculo que me has ayudado a superar, cada fracaso que me ha enseñado una lección invaluable.

Geancarlo Díaz Marín

AGRADECIMIENTOS

En primer lugar, deseo expresar mi sincero agradecimiento a mis padres, cuyo apoyo incondicional ha sido fundamental para la consecución de mis metas personales y académicas. Su constante respaldo emocional y material ha sido un pilar en mi trayectoria, permitiéndome concentrarme en mis estudios y perseverar frente a las adversidades.

De otra parte, mi agradecimiento se extiende a mi tutor, cuya dedicación y paciencia han sido invaluables en mi proceso de aprendizaje. Sus orientaciones precisas y consejos perdurarán como una guía invaluable en mi futuro profesional.

CONTENIDO

pág.

LISTAS DE TABLAS

LISTAS DE FIGURAS

LISTAS DE ANEXOS

INTRODUCCIÓN

La transición de las pruebas manuales a las pruebas automáticas en el marco del desarrollo ágil de software representa un cambio significativo que tiene un impacto directo en la eficiencia, la calidad y la velocidad de entrega en proyectos de desarrollo de software. Este proceso implica una revisión profunda de las prácticas existentes y la adopción de nuevas estrategias que permitan integrar las pruebas de forma temprana y continua en el ciclo de desarrollo. Al valorar antecedentes y experiencias previas en este ámbito, se evidencia la importancia de comprender cómo diferentes enfoques y metodologías impactan en la efectividad de las pruebas en entornos ágiles. Esto no solo implica considerar aspectos técnicos como la automatización de pruebas unitarias, sino también aspectos relacionados con la gestión de la calidad del software, la curva de aprendizaje del equipo de pruebas y la satisfacción del cliente. Por tanto, esta investigación busca diseñar un proceso específico que permita no solo mejorar la calidad del software, sino también optimizar los recursos y tiempos de entrega, contribuyendo así a la excelencia en el desarrollo de software en entornos ágiles.

La calidad del software es un componente esencial en el ámbito de la informática, siendo las pruebas de software una práctica clave en el proceso de desarrollo. Estas pruebas tienen como objetivo principal demostrar que un programa funciona según lo planeado y detectar posibles defectos antes de su puesta en marcha. Mediante el uso de datos simulados, se verifica el comportamiento del software para garantizar su correcto funcionamiento y la ausencia de errores y fallas. En consonancia con las metodologías ágiles, las pruebas de software se integran de manera temprana y continua en el ciclo de desarrollo, lo que permite identificar y corregir problemas de forma proactiva, mejorando así la calidad del producto final, ya sea personalizado o genérico.

En el contexto de la creciente complejidad en el desarrollo de software debido a la transformación digital y el uso extendido de tecnología, las organizaciones enfrentan el desafío de garantizar la calidad de sus productos a través de pruebas efectivas. La Universidad Mariana se encuentra en una situación donde la realización de pruebas manuales consume recursos significativos, tanto en tiempo como en costos, lo que afecta la calidad y la entrega oportuna de nuevas funcionalidades. Ante esta problemática, surge la necesidad de implementar un enfoque sistemático que permita pasar de pruebas manuales a pruebas automáticas, integrándolas de

manera efectiva en el proceso de desarrollo, operación y mantenimiento ágil de software, con el fin de mejorar la eficiencia, la cobertura y la confiabilidad del software entregado, aspectos fundamentales para mantener la alta calidad requerida por la institución.

La presente investigación tiene como objetivo principal proponer un proceso para el desarrollo y ejecución de pruebas automáticas de unidad desde el desarrollo ágil de software para el Área de Tecnología de la Universidad Mariana. Esta iniciativa surge de la necesidad de optimizar el proceso de construcción, soporte y mantenimiento de software, buscando ahorrar tiempo, mejorar la calidad del software y agilizar su despliegue en producción. Además, se busca validar este proceso en un entorno experimental para proporcionar pautas claras a los profesionales del departamento de tecnología, contribuyendo así a la adopción de prácticas efectivas y al aumento de la calidad del software en la universidad.

La investigación en el Área de Tecnología de la Universidad Mariana revela un panorama detallado sobre el proceso de pruebas de software y la gestión de no conformidades, basado en la aplicación de una encuesta a los profesionales encargados en el equipo del desarrollo, soporte y mantenimiento de software. Se destaca la importancia de pruebas unitarias, de componentes, de aceptación y de Interfaces Gráficas de Usuario (GUI), así como la existencia de dos caminos en el desarrollo de pruebas para cambios y mejoras en el software, y para identificar y solucionar problemas en productos existentes. El uso predominante de herramientas como Mantis y Excel resalta la necesidad de explorar nuevas tecnologías para optimizar la eficiencia de las pruebas. Además, se propone un modelo de proceso de negocio basado en BPMN para la gestión de no conformidades.

En el documento se encuentra estructurado en cuatro partes. La primera parte corresponde con los elementos del proceso investigativo. La segunda parte muestra los resultados obtenidos en el desarrollo de cada uno de los objetivos específicos. La tercera parte presenta las conclusiones obtenidas en el estudio. Finalmente, se presentan un conjunto de recomendaciones donde se visualizan acciones concretas por realizar y trabajos futuros.

1. ELEMENTOS DEL PROCESO INVESTIGATIVO

1.1. ANTECEDENTES Y ESTADO DEL CONOCIMIENTO

En esta sección se desea hacer una valoración de hechos, fenómenos y experiencias anteriores en relación con la forma de transitar de las pruebas manuales a las pruebas automáticas desde el desarrollo ágil de software.

Los trabajos revisados relacionados con el proyecto fueron recuperados de los repositorios ACM Digital Library, IEEExplorer Digital Library y Springer. En total se consultaron quince (15) estudios de los cuales se seleccionaron cinco (5). De cada antecedente se extrajo objetivos y conclusiones, para posteriormente establecer similitudes y diferencias con la investigación propuesta. Finalmente, se identifica la oportunidad de investigación, como se muestra a continuación.

En el trabajo realizado por Kettunen et al (2010), se analiza las diferencias en actividades de pruebas entre empresas que desarrollan software que aplican métodos ágiles y empresas que presentan un enfoque orientado al plan de manera tradicional. Las conclusiones permiten observar que, los métodos ágiles les permiten a las organizaciones involucrar las pruebas de forma temprana y tiene una influencia positiva sobre la satisfacción del producto final. La principal similitud con este antecedente obedece a que se realiza una caracterización de la forma como se hacen las pruebas. Las principales diferencias con el antecedente se presentan en cuanto a que, en esta investigación, se diseñará un proceso para desarrollo y ejecución de pruebas unitarias automáticas desde el desarrollo ágil de software, con base en la caracterización. Además, se validará el proceso para desarrollo y ejecución de pruebas unitarias automáticas, en un ambiente experimental.

En el trabajo realizado por Klammer y Ramler (2017) se tiene como objetivo acelerar los ciclos de ejecución de pruebas y reducir el esfuerzo involucrado en ejecutar pruebas manualmente. Las principales conclusiones permiten observar que, la propuesta para el desarrollo ágil enfatiza la unidad de pruebas como la base de una sólida estrategia de automatización de pruebas y desalienta las pruebas automatizadas de GUI, muchos de los errores críticos solo se

encontraron con auto - pruebas de GUI acopladas, las cuales no pudieron ser reemplazadas por pruebas unitarias o de laboratorio, la acción permite crear una gran cantidad de pruebas que pueden ejecutarse automáticamente. Sin embargo, para detectar errores, las pruebas requieren un oráculo de prueba que define cuáles son los resultados correctos esperados, así como también, la generación de pruebas para aplicaciones GUI que hereda desafíos de ambos: generación de pruebas y pruebas de GUI. Las pruebas resultantes tienden a ser duraderos y difíciles de entender, y como ultima conclusión el límite de generación de prueba está definido por la capacidad disponible para analizar los resultados de pruebas. Estas se alcanzan cuando los resultados no se analizan a tiempo antes de la próxima prueba.

La principal similitud con este antecedente obedece a que se diseñará un proceso para el desarrollo y la ejecución de pruebas automáticas (pruebas unitarias) desde el desarrollo ágil de software. Las principales diferencias con este antecedente se presentan en cuanto a, caracterizar la forma como se realizan las pruebas en el área de tecnología de la Universidad Mariana. Además, se validará el proceso para desarrollo y ejecución de pruebas automáticas (pruebas unitarias) diseñando, en un ambiente experimental. (Qué variables se pueden evaluar: Calidad del software, curva de aprendizaje del tester).

En el trabajo realizado por Maqbool et al (2018), el estudio se centra en las cinco partes clave de las pruebas de software, explícitamente métodos de prueba de software, métricas de prueba de software, prácticas y técnicas, estándares de prueba, herramientas de prueba automatizadas y pruebas de educación y formación. Las principales conclusiones permiten observar que, el trabajo de investigación describió y examinó los resultados de pruebas de prácticas y procedimientos, también, que en una encuesta hecha a muchas empresas se exploraron las tendencias de las prácticas de prueba de software que son seguidos en la industria de TI de Pakistán. La necesidad de esto es el trabajo de investigación es establecer la conexión ideal entre calidad y prueba de software; garantizando que las estrategias de prueba en los sistemas de software producirán el software que tenga la mejor calidad, y como ultima conclusión las organizaciones de software en Pakistán necesitan hacer mejoras en sus programas de prueba de software para que puedan obtener una competencia. La principal similitud con este antecedente obedece a que se validará el proceso para el desarrollo y ejecución de pruebas automáticas (pruebas unitarias) diseñando, en un ambiente experimental. Las principales diferencias con este

antecedente se presentan en cuanto a, caracterizar la forma como se realizan las pruebas en el área de tecnología de la Universidad Mariana, y en diseñar un proceso para desarrollo y ejecución de pruebas automáticas (pruebas unitarias) desde el desarrollo ágil de software, con base en la caracterización.

En el trabajo realizado por Tyagi, S., Sibal, R., & Suri, B. (2017), el objetivo del estudio es crear una comprensión de los diferentes desafíos enfrentados por profesionales ágiles al adoptar la automatización de pruebas en proyectos ágiles y presentar algunas estrategias posibles para superar esos desafíos. Las principales conclusiones permiten observar que, descubren los diversos desafíos y estrategias adoptadas por los equipos ágiles mientras se establece una buena automatización de pruebas prácticas en sus proyectos, y como ultima conclusión se tiene comprender el desafío clave al adoptar la automatización de pruebas en proyectos ágiles y proporciona algunas estrategias ampliamente utilizadas para superar esos desafíos. La principal similitud con este antecedente obedece a diseñar un proceso para desarrollo y ejecución de pruebas automáticas (pruebas unitarias) desde el desarrollo ágil de software, con base en la caracterización. Las principales diferencias con este antecedente se presentan en cuanto a que, en esta investigación, se diseñará un proceso para desarrollo y ejecución de pruebas automáticas (pruebas unitarias) desde el desarrollo ágil de software, con base en la caracterización. Además, se validará el proceso para desarrollo y ejecución de pruebas automáticas (pruebas unitarias) diseñando, en un ambiente experimental.

En el trabajo realizado por Collins, E. F., & de Lucena, V. F. (2012), se tiene como objetivo prevenir defectos durante el proceso de desarrollo. La principal conclusión permite observar que es factible adaptar prácticas ágiles para atender la meta del proyecto, trabajar para resolver problemas, busque herramientas integradas de prueba de código abierto mientras se mantiene la gestión y organización de unas pruebas de software. La principal similitud con este antecedente obedece a diseñar un proceso para desarrollo y ejecución de pruebas automáticas (pruebas unitarias) desde el desarrollo ágil de software, con base en la caracterización. La principal diferencia con este antecedente, se presentan en cuanto a validar el proceso para el desarrollo y ejecución de pruebas automáticas (pruebas unitarias) diseñando, en un ambiente experimental.

1.2. TITULO

Tránsito de las pruebas manuales a las pruebas automáticas desde el desarrollo ágil de software en el Área de Tecnología de la Universidad Mariana.

1.3. PROBLEMA DE INVESTIGACIÓN

1.3.1. Descripción del problema

El incremento en el uso de tecnología como efecto de la transformación digital en los procesos de negocio de las organizaciones; ha generado que la complejidad para desarrollar software se incremente. En este sentido, cobra relevancia asegurar la calidad de los productos a través del desarrollo de pruebas.

Existen múltiples factores que inciden e incrementan la complejidad al momento de realizar pruebas de software y que posibiliten evidenciar fallos en los sistemas. Entre los principales factores, se encuentran: entradas de usuario, dispositivos/pantallas, ambientes heterogéneos, condiciones de ejecución, contexto, complejidad de dominio, los diferentes ambientes de despliegue, restricciones de aplicación del software y presión continua para entregas frecuentes Linares-v et al (2017).

Actualmente, si se desea desarrollar una aplicación para un dispositivo móvil, los casos de prueba que se presentan debido al número de modelos, versiones y tamaños en el mercado, corresponden con combinaciones, entre estos elementos, tal es el caso de la línea de teléfonos inteligentes marca Samsung que tiene vigentes en el año 2022 diez modelos, cada uno posee de una a cuatro versiones. Además, por cada modelo existen dos diferentes tamaños de dispositivo Samsung Electronics (2022). Realizando la combinación de modelos, versión y tamaño, se obtiene un total de 19 dispositivos. Lo anteriormente descrito deja ver el grado de complejidad que implica realizar pruebas a un producto software en este tipo de dispositivos. El esfuerzo de aplicar pruebas manuales para los casos mencionados representaría un incremento considerable en los costos, causando que el presupuesto destinado al desarrollo se invierta solo en esta fase, y que contradictoriamente dentro de los proyectos siempre es el más bajo.

El área de tecnología de la Universidad Mariana administra, desarrolla y brinda mantenimiento a todo el sistema de información, mantiene permanentemente todos los procesos automatizados que operan en cada una de las unidades administrativas, financieras, académicas y operacionales, así como garantizar el buen funcionamiento de estos procesos. También administra la operación de las bases de datos para garantizar la integridad de los mismos, bajo una adecuada definición, diseño, mantenimiento y seguridad de la información compartida en el sistema de Base de Datos de la Institución.

Según el director del Departamento de Tecnología de la Universidad Mariana (2021) las pruebas a nuevo software se realiza dependiendo de las funcionalidades, a medida que se van dando los casos o los procedimientos se hacen pruebas, piden capacitaciones o asesorías. (Ver anexo 1). En relación con la forma como están desarrollando las pruebas, el director manifiesta que es manual y se hacen al funcionamiento del software, la dificultad que se presenta es la pérdida de tiempo en el desarrollo de pruebas de manera manual y también al ejecutar pruebas de esta misma manera se tiene que hacer matrices en Excel, tienen que documentar todos los errores, la cantidad de información no les ha posibilitado plantear una solución para automatizar el proceso de pruebas, ya que requiere personal.

Lo que el director quiere es no recurrir a procesos de ayuda que les aumenta costos, manifiesta que le gustaría que se automaticen y esas pruebas quieren también emplearlas a requerimientos no funcionales, concurrencia y escalabilidad (Ver anexo 1)

Las razones por las cuales se vienen presentado las situaciones relacionadas anteriormente con el desarrollo de pruebas obedecen a que el equipo de ingenieros es reducido y tienen una alta carga de trabajo. Además, las pruebas de software se realizan de forma manual o en muchos casos no se hacen. Otro aspecto corresponde con que a veces hacen pruebas y fallan, entonces se pierde tiempo, piden asesorías, se busca con el proveedor, los acompañan y van encontrando errores. También, los asesores no siempre están disponibles y se tiene que esperar el turno en la mesa de ayuda, en los tickets, para que puedan asignar uno y poder continuar una estrategia o hacer la revisión de los fallos que tienen en las pruebas.

Por todo lo anterior, se puede afirmar que a pesar de que el aseguramiento de la calidad es importante en las nuevas funcionalidades que desarrolla el equipo del área de Tecnología de la

Universidad Mariana, el personal capacitado reducido, el alto nivel de requerimientos para el área, la premura con que se debe entregar las funcionalidades y el desarrollo de actividades manuales impiden que se pueda probar el software con un nivel de cobertura alto.

De continuar presentándose esta problemática el software tendría un bajo nivel de pruebas o sin probar acción que se puede ver reflejada en brindar información inconsistente, desarrollar tareas con no conformidades, retraso en la entrega de nuevas funcionalidades; fundamentales ahora que la Universidad Mariana se encuentra acreditada en alta calidad por el Ministerio de Educación.

Por todo lo anterior nace la motivación por plantear un camino sistemático que le permita al departamento de tecnología de manera sistemática transitar de las pruebas manuales a las pruebas automáticas de unidad desde el desarrollo ágil de software.

1.3.2. Formulación del problema

¿Cómo transitar de las pruebas manuales a las pruebas automáticas de unidad desde el desarrollo ágil de software en el área de tecnología de la Universidad Mariana?

1.4. OBJETIVOS

1.4.1. Objetivo general

Proponer un proceso para el desarrollo y ejecución de pruebas automáticas de unidad desde el desarrollo ágil de software para el Área de Tecnología de la Universidad Mariana.

1.4.2. Objetivos Específicos

-Caracterizar la forma como se realizan las pruebas de unidad en el área de tecnología de la Universidad Mariana.

-Diseñar un proceso para el desarrollo y ejecución de pruebas automáticas de unidad desde el desarrollo ágil de software, con base en la caracterización.

-Validar el proceso para desarrollo y ejecución de pruebas automáticas de unidad diseñando, en un ambiente experimental.

1.5. **JUSTIFICACIÓN**

Este estudio es interesante porque al caracterizar la forma como se realizan las pruebas de unidad en el área de tecnología de la Universidad Mariana, se recopilará información relacionada con el proceso de pruebas que posteriormente será analizada con el fin de comprender en profundidad la forma como se está realizando, quienes intervienen, los controles que se realizan y con qué herramientas se están apoyando. Además, al diseñar un proceso para el desarrollo y ejecución de pruebas automáticas de unidad basados en la caracterización, se formula un camino a seguir para automatizar el proceso de construcción, soporte y mantenimiento de software en el área de tecnología. Al validar el proceso diseñando, en un ambiente experimental, se busca poner a prueba, presentar pautas para que los integrantes del equipo del departamento de tecnología lo adopten.

La utilidad de este trabajo investigativo radica en que caracterizar la forma como se realizan las pruebas en el área de tecnología de la Universidad Mariana permite identificar las actividades, recursos, actores y herramientas al momento de realizar pruebas en el departamento de TI de la Universidad Mariana. Además, al diseñar un proceso para desarrollo y ejecución de pruebas automáticas desde el desarrollo ágil de software, con base en la caracterización, permite ahorrar tiempo en el desarrollo de pruebas, posibilita que la calidad del software aumente y se despliegue más rápido en producción. Finalmente, al validar el proceso para desarrollo y ejecución de pruebas automáticas diseñando, en un ambiente experimental, es útil porque a través de la experimentación permite llevar la propuesta hecha de la teoría a la práctica y realizar ajustes necesarios para el mejoramiento de la misma. Los principales beneficiarios por la realización del trabajado investigativo serán los profesionales del área de tecnología dedicados al desarrollo, soporte y mantenimiento de software en la Universidad Mariana. Además, el director del área de tecnología en cierto grado podrá asegurar la calidad del software que soporta los procesos académicos, administrativos y financieros de la universidad, aumentando la cobertura con las pruebas automáticas.

Esta investigación es novedosa porque, al caracterizar la forma como se realizan las pruebas en el área de tecnología de la Universidad Mariana se utilizará un lenguaje de modelado visual para representar el proceso como realizan las pruebas. Igualmente, diseñar un proceso para desarrollo y ejecución de pruebas automáticas es novedosa, debido a que, se realizará a partir de las prácticas propuestas por los valores y principios descritos en el agilísimo; además, la propuesta se realizará con base en la caracterización. Otro aspecto de novedad radica en que, al validar el proceso para desarrollo y ejecución de pruebas automáticas diseñando, se busca integrar, a través de la experimentación continua, la etapa de pruebas con el proceso de desarrollo, soporte y mantenimiento de software, para que el departamento de TI de la Universidad Mariana pueda transitar de las pruebas manuales, a las automáticas, además el cual no se han presentado estudios similares en la Universidad.

1.6. MARCOS DE REFERENCIA

A continuación, se presentan los marcos de acuerdo con la naturaleza de la investigación.

1.6.1. Marco Teórico - Conceptual

1.6.1.1. Pruebas de software

A partir del trabajo de Sommerville (2011), las pruebas de software intentan demostrar que un programa hace lo que se intenta que haga, así como descubrir defectos en el programa antes de usarlo. Al probar el software, se ejecuta un programa con datos artificiales. Hay que verificar los resultados de la prueba que se opera para buscar errores, anomalías o información de atributos no funcionales del programa.

Actualmente, se quiere lograr entregar y desarrollar el software en iteraciones más pequeñas, dando más tiempo para ejecutar tareas de prueba dentro de cada una de las iteraciones y permitiendo un inicio más temprano del trabajo de prueba. Como las pruebas de software se pueden comenzar temprano y se completan simultáneamente con el trabajo de desarrollo, también puede enfocarse en realizar las pruebas para el software pieza por pieza y elaborar las partes ya existentes del producto final, en lugar de ser simplemente la última fase del proyecto antes de la entrega Kettunen, Kasurinen, Taipale, & Smolander (2010).

La prueba de software es una práctica de ejecutar aplicaciones o programas para encontrar defectos en ella Myers (2012), para una ingeniería de software efectiva, se requiere la transferencia de software de gran calidad. Las pruebas de software identifican problemas en el software y aseguran que el software entregado esté libre de errores y fallas Jovanović (2017).

A partir del trabajo de Sommerville (2011), el propósito de las pruebas de software busca demostrar al desarrollador y al cliente que el software cumple con los requerimientos. Para el software personalizado, esto significa que en el documento de requerimientos debe haber, por lo menos, una prueba por cada requerimiento. Para los productos de software genérico, esto quiere decir que tiene que haber pruebas para todas las características del sistema, junto con combinaciones de dichas características que se incorporarán en la liberación del producto y encontrar situaciones donde el comportamiento del software sea incorrecto, indeseable o no esté de acuerdo con su especificación. Tales situaciones son consecuencia de defectos del software. La prueba de defectos tiene la finalidad de erradicar el comportamiento indeseable del sistema, como caídas, interacciones no deseadas con otros sistemas, cálculos incorrectos y corrupción de datos.

Una propuesta para el desarrollo de pruebas es el planteado por Cohn (2009). (Ver Figura 1):

Figura 1. Pirámide de Cohn
Fuente: Una adaptación de Cohn (2009)

Las pruebas unitarias deben escribirse durante el sprint, tal vez incluso de una manera basada en pruebas. La experiencia muestra que es raro que un programador vuelva más tarde al código de trabajo y agregue pruebas unitarias. Por lo general, es la historia pasada con pruebas automatizadas de estilo de captura y reproducción lo que se lleva a pensar que solo se puede automatizar una vez que se completa una función.

Las pruebas en todos los niveles de la pirámide de automatización de pruebas lograron las siguientes reducciones:

- Redujo la cantidad de personal involucrado en sus nueve implementaciones de la aplicación en un 65% (15 personas)

- Se redujo la cantidad de tiempo dedicado a las pruebas finales de puesta en marcha: dos o tres horas de esfuerzo manual se convirtieron en diez minutos de pruebas automatizadas.

- Se redujo la cantidad de tiempo en las pruebas de cordura posteriores al lanzamiento: de tres a cuatro horas de pruebas manuales se convirtieron en 45 minutos de ejecución de más de 200 pruebas acopladas automáticamente

- Redujo la cantidad de personas involucradas en la liberación de parches en casi un 80% (a aproximadamente cinco personas)

- Ahorró más de 300 horas de trabajo por versión principal y cientos más por todas las versiones de parches. Estos resultados no son inusuales para una organización que se toma en serio las pruebas automatizadas.

Las pruebas de interfaz de usuario automatizadas se ubican en la parte superior de la pirámide de automatización de pruebas porque se quiere hacer lo menos posible. Se quiere esto porque las pruebas de la interfaz de usuario a menudo tienen los siguientes atributos negativos:

- Quebradizo. Un pequeño cambio en la interfaz de usuario puede romper muchas pruebas. Cuando esto se repite muchas veces en el transcurso de un proyecto, los equipos simplemente se rinden y dejan de corregir las pruebas cada vez que cambia la interfaz de usuario.

- Es caro de escribir. Un enfoque rápido de captura y reproducción para grabar pruebas de interfaz de usuario puede funcionar, pero las pruebas grabadas de esta manera suelen ser las más frágiles. Escribir una buena prueba de interfaz de usuario que siga siendo útil y válida lleva tiempo.

- Consume mucho tiempo. Las pruebas que se ejecutan a través de la interfaz de usuario suelen tardar bastante en ejecutarse. Ha visto numerosos equipos con impresionantes conjuntos de pruebas de interfaz de usuario automatizadas que tardan tanto en ejecutarse que no pueden ejecutarse todas las noches, y menos varias veces al día.

La capa de servicio es algo que la aplicación hace en respuesta a alguna entrada o conjunto de entradas. La prueba de nivel de servicio consiste en probar los servicios de una aplicación por separado de su interfaz de usuario. Entonces, en lugar de ejecutar una docena de casos de prueba de multiplicación a través de la interfaz de usuario de la calculadora, realizamos esas pruebas a nivel de servicio. Para ver cómo podría funcionar esto, suponga que creamos una hoja de cálculo como la tabla, donde cada fila representa un caso de prueba. Las dos primeras columnas representan los números que se van a multiplicar, la tercera columna es el resultado esperado y la

cuarta columna contiene notas explicativas que no serán utilizadas por la prueba pero que hacen que las pruebas sean más legibles.

1.6.1.2. Pruebas de software manuales

A partir del trabajo de Paul Ammann y Jeff Offutt (2013), en las pruebas manuales, un evaluador ejecuta el programa con algunos datos de prueba y compara los resultados con sus expectativas. Anotan y notifican las discrepancias a los desarrolladores del programa.

Según Gloriana Araya Solís, Geovanny Méndez Marín y Ronald Jiménez (2014), asegura las pruebas manuales son propensos a errores y los procesos de ejecución son lentos. Se recomienda enfocar la automatización para pruebas unitarias, pruebas funcionales y pruebas de interfaz de usuario. Estas pruebas son realizadas sobre aplicaciones dirigidas por eventos (Event-Driven) que son un subconjunto de los Sistemas Reactivos. Dichas aplicaciones reciben eventos tanto por parte del usuario, como de la plataforma, y sus sensores. Los componentes que conforman la estructura de la GUI de este tipo de aplicaciones, aceptan secuencias de eventos de usuario, que alteran el estado de la aplicación. Esta alteración puede o no, incluir un cambio que altera a la GUI. Las pruebas para el software con GUI, consisten en la ejecución de eventos pertenecientes a esos componentes y el monitoreo del resultado al cambiar el estado del programa.

Los casos de prueba para aplicaciones con interfaces gráficas de usuario tienen secuencias de eventos tales como inputs y alguna indicación del estado del programa (Estado de la GUI, Estado de la Memoria, Log de errores, u otros indicadores en tiempo de ejecución del estado de la aplicación) como el output esperado. En muchos casos (a menos que una aplicación incluya dos tipos de interfaces, con GUI y sin GUI) las pruebas de GUI (el GUI Testing) es la única forma posible de realizar pruebas a nivel de sistema para este tipo de aplicaciones. Esto hace que el testing de este tipo sea una parte crítica de las pruebas para cualquier software GUI. El tamaño y complejidad de las GUI modernas, en términos de los componentes GUI, y los eventos que deben ser ejecutados en ellos, exceden los límites prácticos de la exhaustiva y analítica aproximación para el testing. El número de posibles casos de test para una GUI incrementa exponencialmente

dependiendo del número de eventos por caso de prueba. Desafortunadamente, los testers no pueden ignorar secuencias de eventos largos y complejos, debido a que esas secuencias frecuentemente revelan bugs de estados específicos.

1.6.1.3. Pruebas en el desarrollo ágil de software

Se puede usar el desarrollo ágil en vez de la creación de prototipos, de manera que se diseñen en conjunto los requerimientos y la implementación del sistema. Algunas personas consideran la ingeniería de requerimientos como el proceso de aplicar un método de análisis estructurado, tal como el análisis orientado a objetos Larman (2002).

Esto implica analizar el sistema y desarrollar un conjunto de modelos gráficos del sistema, como los modelos de caso de uso, que luego sirven como especificación del sistema. El conjunto de modelos describe el comportamiento del sistema y se anota con información adicional que describe, por ejemplo, el rendimiento o la fiabilidad requeridos del sistema.

El desarrollo ágil tiene que administrarse de tal modo que se busque el mejor uso del tiempo y de los recursos disponibles para el equipo. Esto requiere un enfoque diferente a la administración del proyecto, que se adapte al desarrollo incremental y a las fortalezas particulares de los métodos ágiles. Aunque el enfoque de Scrum Schwaber (2004), es un método ágil general, su enfoque está en la administración iterativa del desarrollo, y no en enfoques técnicos específicos para la ingeniería de software ágil.

Los métodos de desarrollo ágiles argumentan que los requerimientos cambian tan rápidamente que un documento de requerimientos se vuelve obsoleto tan pronto como se escribe, así que el esfuerzo se desperdicia en gran medida.

Los procesos de desarrollo ágil, como la programación extrema, se diseñaron para enfrentar los requerimientos que cambian durante el proceso de desarrollo. En dichos procesos, cuando un usuario propone un cambio de requerimientos, éste no pasa por un proceso de administración del cambio formal. En vez de ello, el usuario tiene que priorizar dicho cambio y, si es de alta prioridad, decidir qué características del sistema planeadas para la siguiente iteración pueden eliminarse.

Hay nodos en el sistema que entregan funcionalidad con frecuencia son sistemas independientes sin una sola autoridad encargada de ellos. La red que conecta dichos nodos es un sistema de gestión independiente. Es un sistema complejo por derecho propio y no puede controlarse por los propietarios de los sistemas que usan la red. Por lo tanto, existe una imprevisibilidad inherente en la operación de los sistemas distribuidos que debe considerar el diseñador del sistema. Algunos de los conflictos de diseño más importantes que deben considerarse en la ingeniería de sistemas distribuidos son:

- Transparencia ¿En qué medida el sistema distribuido debe aparecer al usuario como un solo sistema? ¿Cuándo es útil para los usuarios entender que el sistema es distribuido?
- Apertura ¿Un sistema debe diseñarse usando protocolos estándar que soporta interoperabilidad o deben usarse protocolos más especializados que restrinjan la libertad del diseñador?
- Escalabilidad ¿Cómo puede construirse el sistema para que sea escalable? Esto es, ¿cómo podría diseñarse un sistema global para que su capacidad se pueda aumentar en respuesta a demandas crecientes hechas sobre el sistema?
- Seguridad ¿Cómo pueden definirse e implementarse políticas de seguridad útiles que se apliquen a través de un conjunto de sistemas administrados de manera independiente?
- Calidad de servicio ¿Cómo debe especificarse la calidad del servicio que se entrega a los usuarios del sistema y cómo debe implementarse el sistema para entregar una calidad de servicio aceptable para todos los usuarios?
- Gestión de fallas ¿Cómo pueden detectarse las fallas del sistema, contenerse (de modo que tengan efectos mínimos sobre otros componentes del sistema) y repararse?

Sin embargo, las prácticas y los procedimientos organizacionales adecuados pueden controlar la deseconomía de escala que surge como consecuencia de aumentar la complejidad.

Como plantean Boehm y Turner (2003), cada enfoque es adecuado para diferentes tipos de software, se necesita encontrar un equilibrio entre procesos dirigidos por un plan y procesos ágiles. No hay un proceso de software “ideal”, sin embargo en muchas organizaciones sí existe un ámbito para mejorar el proceso de software. Los procesos quizás incluyan técnicas anticuadas

o tal vez no aprovechen las mejores prácticas en la industria de la ingeniería de software. Por consecuencia, muchas organizaciones aún no sacan ventaja de los métodos de la ingeniería de software en su desarrollo. Los procesos de software pueden mejorarse con la estandarización de los procesos, esto conduce a mejorar la comunicación, a reducir el tiempo de capacitación, y a que el soporte de los procesos automatizados sea más económico. La estandarización también es importante tanto en la introducción de nuevos métodos y técnicas de ingeniería de software, como en sus buenas prácticas.

Se han planteado varias técnicas como posibles enfoques para la descomposición o el análisis del peligro, las cuales resume Storey (1996). Incluyen revisiones y listas de verificación, técnicas formales como el análisis de red de Petri Peterson (1981), lógica formal Jahanian y Mok (1986) y análisis de árbol de fallas Leveson y Stolzy, (1987); Storey (1996).

Una propuesta para el desarrollo de pruebas ágiles es el planteado por Lisa Crispin (2008). (Ver Figura 2):

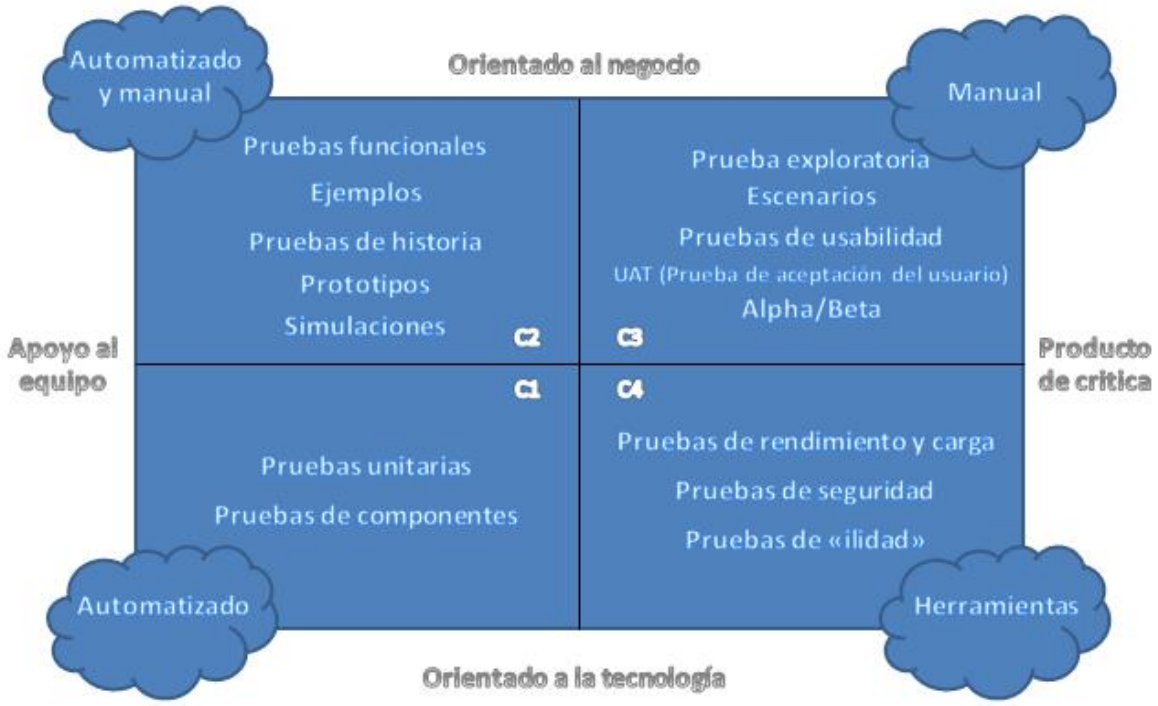


Figura 2. Cuadrante de pruebas ágiles

Fuente: Una adaptación de Lisa Crispin (2008)

Para Crispin (2008), el cuadrante C1 (Ver figura 2, inferior izquierdo) representa el desarrollo impulsado por pruebas, que es una práctica de desarrollo ágil central. Las pruebas unitarias verifican la funcionalidad de un pequeño subconjunto del sistema, como un objeto o método. Las pruebas de componentes verifican el comportamiento de una gran parte del sistema, como un grupo de clases que brindan algún servicio Meszaros (2007). Ambos tipos de pruebas generalmente se automatizan con un miembro de la familia de herramientas de automatización de pruebas xUnit. A estas se las reconoce como pruebas de programador, pruebas de cara al desarrollador o pruebas de cara a la tecnología. Permiten a los programadores medir lo que Kent Beck ha llamado la calidad interna de su código Beck (1999).

Para Crispin (2008), las pruebas del cuadrante C2 (Ver figura 2, superior izquierdo) también respaldan el trabajo del equipo de desarrollo, pero a un nivel superior. Estas pruebas de cara al negocio, también llamadas pruebas de cara al cliente y pruebas de cliente definen la calidad externa y las características que los clientes desean.

Al igual que las pruebas del cuadrante C1, también impulsan el desarrollo, pero a un nivel superior. Con un desarrollo ágil, estas pruebas se derivan de ejemplos proporcionados por el equipo del cliente, Crispin (2008). Describen los detalles de cada historia. Las pruebas orientadas a la empresa se ejecutan a nivel funcional y cada una verifica una condición de satisfacción empresarial. Están redactadas de manera que los expertos empresariales puedan comprenderlos fácilmente mediante el lenguaje del dominio empresarial. De hecho, los expertos en negocios utilizan estas pruebas para definir la calidad externa del producto y, por lo general, ayudan a redactarlas. Es posible que este cuadrante pueda duplicar algunas de las pruebas que se realizaron a nivel de unidad; sin embargo, las pruebas del cuadrante 2 están orientadas a ilustrar y confirmar el comportamiento deseado del sistema a un nivel superior.

Para Crispin (2008), el cuadrante C3 (Ver figura 2, inferior derecho) es donde entran en juego las pruebas para criticar el producto en el tercer y cuarto cuadrantes. El cuadrante 3 clasifica las pruebas orientadas a los negocios que ejercitan el software en funcionamiento para ver si no cumple con las expectativas o no se enfrenta a la competencia Cripsin (2008). Cuando se hacen pruebas comerciales para criticar el producto, se intenta emular la forma en que un usuario real trabajaría con la aplicación. Esta es una prueba manual que solo un humano puede hacer. Se podría usar algunos scripts automatizados para ayudar a configurar los datos que se

necesitan, pero se tiene que usar los sentidos, cerebro e intuición para verificar si el equipo de desarrollo ha entregado el valor comercial requerido por los clientes.

Para Crispin (2008), el cuadrante C4 (Ver figura 2, superior derecho) posee los tipos de pruebas que son tan críticos para el desarrollo ágil como para cualquier tipo de desarrollo de software. Estas pruebas están orientadas a la tecnología y se discuten en términos técnicos más que comerciales. Las pruebas de tecnología en el cuadrante 4 tienen como objetivo criticar las características del producto, como el rendimiento, la solidez, la seguridad y las "ilidades" (Confiabilidad, Operabilidad, entre otras). Por ejemplo, los programadores podrían aprovechar las pruebas unitarias en pruebas de rendimiento con un motor de subprocesos múltiples. Sin embargo, la creación y ejecución de estas pruebas puede requerir el uso de herramientas especializadas y experiencia adicional.

1.6.2. Marco contextual

La Universidad Mariana es una Institución de Educación Superior de carácter católico y privado que tiene como misión formar profesionales integrales, humana y académicamente competentes, con responsabilidad social, espíritu crítico y sentido ético (Universidad Mariana, 2020). En este propósito, se ha caracterizado por estar a la vanguardia en la adquisición y uso de tecnologías de la información necesarias para que la comunidad universitaria cumpla con las actividades que contribuyan al mejoramiento de la calidad de los procesos educativos y administrativos en la institución.

La infraestructura tecnológica de la Universidad Mariana es coordinada por el Área de Tecnología y centra sus esfuerzos en cuatro frentes principales que relacionan el tipo de servicios y procesos que tienen que ver con la administración de la tecnología. Estos frentes son: educación virtual, aulas de informática, sistemas de información, y redes de comunicaciones.

Los sistemas de información se han convertido en un puente primordial para articular la tecnología con los procesos de la Universidad, y en este sentido, cobra relevancia que los elementos nuevos, cambios y modificaciones que se les realice, cuenten con pruebas que permitan garantizar la calidad y confiabilidad de las tareas y datos. Por esta razón, para esta investigación el contexto corresponde con los ingenieros que hacen parte de la oficina para la administración de los sistemas de información.

1.7. METODOLOGÍA

1.7.1. Paradigma, enfoque y tipo de investigación

Paradigma: Esta investigación es de corte cuantitativo, porque según Hernández Sampieri, Fernández y Baptista (2011) representa un conjunto de procesos secuenciales y probatorios, por lo tanto no se pueden saltar los pasos debido a que el orden es riguroso que parte de una idea, se deriva en objetivos y preguntas de investigación, de las preguntas se establecen hipótesis y se determinan variables y la validez de los resultados se soportará a través de un proceso estadístico descriptivo debido a que se realizan mediciones, se utilizan técnicas de recolección y se analizan los datos procediendo de manera inductiva donde se obtendrán conclusiones empíricas.

Enfoque: El enfoque para esta investigación es Empírico-analítico Hernández Sampieri, Fernández, & Baptista (2011), debido a que se basa en las experiencias propias para establecer un camino sistemático que permita transitar de las pruebas manuales a las pruebas automáticas desde el desarrollo ágil de software en el área de tecnología de la Universidad Mariana.

Tipo de investigación: El tipo de investigación a utilizar en el presente proyecto es descriptivo-propositivo Hernández Sampieri, Fernández, & Baptista (2011), este tipo de estudio busca describir situaciones o acontecimientos en cuanto caracterizar la forma como se realizan las pruebas en el área de tecnología de la Universidad Mariana. Además, es propositiva, porque después de la elaboración de las descripciones, se diseñará un proceso para el desarrollo y ejecución de pruebas automáticas, que será evaluado en un ambiente experimental.

1.7.2. Línea y Áreas Temáticas de investigación

Línea de investigación: Ingeniería, Informática y computación.

Áreas Temáticas de investigación: Innovación, modelamiento y desarrollo de software.

1.7.3. Población y muestra

Para esta investigación la población corresponde con los ingenieros que hacen parte de la oficina para la administración de los sistemas de información. En el segundo semestre del año 2022, se cuenta con tres (3) ingenieros y dos (2) practicantes en el área de soporte y desarrollo de los

sistemas de información y el director del área de tecnología. Para este caso, no se realizará ningún tipo de muestreo, se trabajará con los seis (6) profesionales en tecnología, ya que la muestra será la misma población.

1.7.4. Proceso de investigación

Descripción del proceso que se pretende realizar y los medios para lograrlo.
Diligenciar el siguiente formato.

Tabla 1. Operacionalización de objetivos

Objetivos específicos	Fuente	Técnica de recolección	Instrumento	Técnica de Procesamiento	Resultado
Caracterizar la forma como se realizan las pruebas en el área de tecnología de la Universidad Mariana	Director y cinco (5) profesional del área de tecnología de la Universidad Mariana	Encuesta	Cuestionario	Estadística descriptiva	Documento con la caracterización de la forma como se realizan las pruebas.
Validar el proceso para desarrollo y	Director y cinco (5)	Encuesta	Cuestionario	Estadística descriptiva	Documento con los resultados de la Validación

Objetivos específicos	Fuente	Técnica de recolección	Instrumento	Técnica de Procesamiento	Resultado
ejecución de pruebas automáticas diseñando, en un ambiente experimental	profesional del área de tecnología de la Universidad Mariana				del proceso para desarrollo y ejecución de pruebas automáticas.

1.7.5. Variables e Hipótesis:

Hipótesis: Es posible transitar de las pruebas manuales a las pruebas automáticas de unidad desde el desarrollo ágil de software en el área de tecnología de la Universidad Mariana.

Variables: las variables a trabajar en esta investigación se presentan en la Tabla 2.

Tabla 2. Variables de la investigación

Variable	Descripción	Tipo de Variable	Objetivo específico	Indicador	Naturaleza	Fuente	Tr*	Ta**
Proceso de pruebas	La forma como se realizan las pruebas en el área de	Independiente	Caracterizar la forma como se realizan las pruebas en el área de	Sub-proceso	Cualitativa	Director y cinco (5) profesional del	Encuesta	Estadística descripti
				Actividad	Cualitativa			
				Actor	Cualitativa			
				Flujo de	Cualitati			

Variable	Descripción	Tipo de Variable	Objetivo específico	Indicador	Naturaleza	Fuente	Tr*	Ta**
	tecnología de la Universidad Mariana		tecnología de la Universidad Mariana	control	va	área de tecnología de la Universidad Mariana		va
Incidencia	Factor que determina el nivel de incidencia del proceso propuesto en el trabajo del área de tecnología de la Universidad Mariana	Independiente	Validar el proceso para desarrollo y ejecución de pruebas automáticas diseñando, en un ambiente experimental	Favorable	Cuantitativa	Director y cinco (5) profesional del área de tecnología de la Universidad Mariana	Encuesta	Estadística descriptiva
				Desfavorable	Cuantitativa			

* Técnica de recolección

**Técnica de análisis (Ubicar en página horizontal)

1.8. **PRESUPUESTO**

Tabla 3. Presupuesto global del proyecto

RUBROS	TOTAL ($)
PERSONAL	3.894.000
MATERIALES Y SUMINISTROS	2.340.000
TOTAL	6.234.000

Tabla 4. Descripción de la Inversión en personal.

NOMBRE	FORMACIÓN ACADÉMICA	INVESTIGADOR / ASESOR	DEDICACIÓN Horas/semana	VALOR
Geancarlo Díaz Marín	PROFESIONAL	INVESTIGADOR	10	3.024.000
Giovanni Hernández P.	MAGISTER	ASESOR	2	870.000
TOTAL				**3.894.000**

Tabla 5. Descripción de Materiales y suministros.

MATERIALES	VALOR
Conexión a internet (Mensual)	100.000
Computador numero 1	2.240.000
TOTAL	**2.340.000**

1.9. **CRONOGRAMA**

A continuación, se presenta las actividades a realizar por cada objetivo específico y el tiempo estimado para cada una de ellas.

Actividades	Tiempo/Semanas																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Caracterizar la forma como se realizan las pruebas en el área de tecnología de la Universidad Mariana	X	X																		
Elaborar instrumentos de recopilación de datos.		X	X	X																
			X	X																
				X	X	X														
						X	X													
Validar los instrumentos.								X												
Realizar una prueba piloto.																				
Aplicar los instrumentos.																				

Actividades	Tiempo/Semanas																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Analizar la información recopilada. Elaborar el capítulo de informe.																				
Diseñar un proceso para el desarrollo y ejecución de pruebas automáticas (pruebas unitarias) desde el desarrollo ágil de software, con base en la caracterización.								X	X	X										
Definir referentes teóricos para la									X	X	X									

Actividades	Tiempo/Semanas																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
representación del proceso.											X	X	X							
Definir los elementos de la propuesta.												X	X							
Elaborar el modelo del proceso.												X	X	X						
Sintetizar los resultados del diseño del proceso.													X	X						
Elaborar el capítulo del informe final.																				
Validar el proceso para desarrollo y ejecución de pruebas														X	X					

Actividades	Tiempo/Semanas																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
automáticas (pruebas unitarias) diseñando, en un ambiente experimental. (Qué variables se pueden evaluar: Calidad del software, curva de aprendizaje del tester)																				
Planear un taller de capacitación para el desarrollo y ejecución de pruebas automáticas. Organizar los recursos para el															X	X	X			
																X	X	X		
																		X	X	
																			X	X

Actividades	Tiempo/Semanas																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
taller de capacitación. Desarrollar el taller de capacitación. Evaluar las percepciones de los participantes. Sintetizar los resultados de la evaluación. Elaborar el capítulo del informe final.																				X

1.10. **CONDICIONES DE ENTREGA**

Como resultado de esta investigación se elaborará un informe técnico con la caracterización de la forma como se realizan las pruebas en el Área de Tecnología de la Universidad Mariana y el diseño del proceso para el desarrollo y ejecución de pruebas automáticas desde el desarrollo ágil de software, con base en la caracterización.

1.11. **PRODUCTOS ESPERADOS**

Como resultado del desarrollo de esta investigación se esperan los siguientes productos:

- Documento de informe final.
- Un artículo en proceso de revisión.
- El certificado de la participación en un evento.

2. RESULTADOS

En esta sección se presentan los resultados obtenidos por el desarrollo de cada objetivo específico.

2.1. Caracterización de la forma como se realizan las pruebas de unidad y componentes

En esta sección se presenta los resultados de la caracterización de la forma como se realizan las pruebas de unidad y componentes en el área de tecnología de la Universidad Mariana. Para realizar la caracterización, se utilizó como técnica la encuesta construyendo un instrumento para recopilar información siguiendo el protocolo propuesto por (Guerrero-Calvache & Hernandez, 2023). En la Figura 1, se pueden observar las fases propuestas en el protocolo.

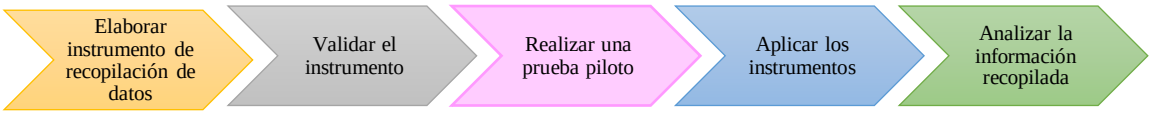


Figura 3. Proceso de la caracterización

2.1.1. Elaboración del instrumento de recopilación de datos

Como el propósito de la caracterización fue elaborar una representación del proceso para realizar pruebas de unidad y componentes en el área de tecnología de la Universidad Mariana, como indicadores se utilizó los elementos planteados por (White & Miers, 2008) para la elaboración de un modelo de proceso de negocio considerando: actividades, actores y flujos de control.

Lo anterior contribuyó a la formulación de las preguntas que harán parte del cuestionario (Guerrero-Calvache & Hernandez, 2023). Las preguntas fueron de tipo cerrado y para profundizar la respuesta, se establecen preguntas abiertas. Para el caso de las preguntas cerradas se introdujo respuestas de selección múltiple.

El cuestionario evalúo 15 ítems y el tiempo estimado para su diligenciamiento fue de 10 minutos. El instrumento se organizó a través de seis secciones. La primera corresponde con las generalidades donde se presenta indicaciones para el diligenciamiento del instrumento y la aceptación del consentimiento informado para el tratamiento de los datos. La segunda hace referencia a la información sociodemográfica del informante. La tercera indaga por los roles que

desempeñan los profesionales en el proceso de elaboración de pruebas. La cuarta corresponde con las actividades necesarias para desarrollar el proceso de pruebas. La quinta hace referencia a las herramientas que soportan el proceso de pruebas. Finalmente, la sexta presenta preguntas asociadas con los flujos de control que se realizan en el proceso de pruebas.

2.1.2. Validación del instrumento

Para la validación, se utilizó un panel de dos (2) expertos para juzgar el valor del contenido de cada pregunta. A partir de las recomendaciones hechas, se realizó los ajustes correspondientes.

El cuestionario fue puesto en marcha de manera digital a través de la herramienta de Google Forms. En el link https://forms.gle/NdQxJ3tbDipMxPUo6 se puede consultar el instrumento de recopilación de información.

2.1.3. Prueba piloto

La prueba piloto se realizó con dos (2) profesionales con el fin de identificar problemas en el diligenciamiento del instrumento. Con base en las recomendaciones elaboradas en la prueba, se efectuaron los cambios que garanticen calidad en el instrumento.

2.1.4. Aplicación de instrumentos

El instrumento fue aplicado a los 4 profesionales del área de tecnología de la Universidad Mariana que hacen parte del equipo de desarrollo, soporte y mantenimiento de software, en el periodo comprendido entre agosto y septiembre de 2023. Para el diligenciamiento se solicitó el permiso respectivo a la directora del Área de Tecnología de la Universidad, el cual se puede observar en el Anexo 2.

2.1.5. Análisis de la información

En esta sección, se presentan los resultados obtenidos a partir de la aplicación del cuestionario a los 4 integrantes del equipo del Área de Tecnología que trabajan en desarrollo, operación y mantenimiento de software.

- Información sociodemográfica

De la población de profesionales que trabajan en desarrollo, operación y mantenimiento de software en el Área de Tecnología de la Universidad Mariana, el 75% corresponde con el género masculino como se puede observar en la Figura 4.

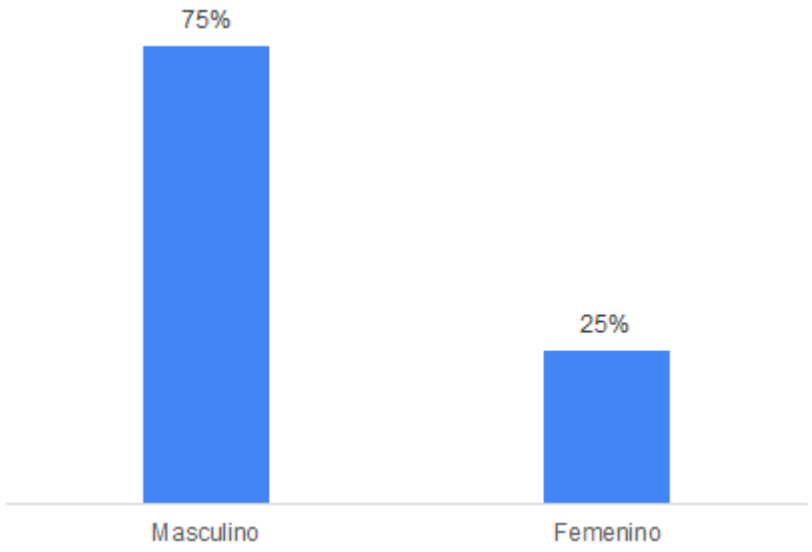


Figura 4. Genero de la población

La edad promedio de los profesionales es de 28 años, aspecto que indica que el equipo de trabajo es joven y se encuentran ejerciendo la profesión en promedio hace 5 años.

En cuanto el nivel de formación, el 75% de los miembros son profesionales y el 25% además del título profesional posee el nivel de maestría como se observa en la figura 5.

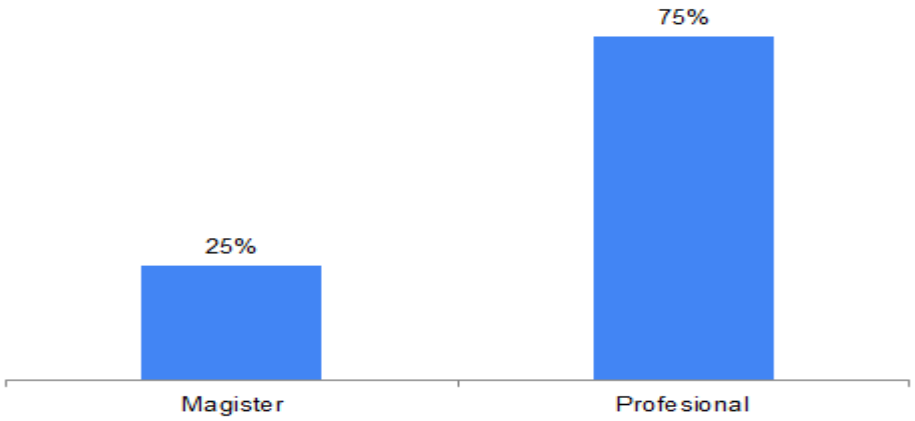


Figura 5. Nivel de formación de la población

La experiencia laboral de la población objeto de estudio se encuentra en un 100% en el rango de 1 a 3 años con una media de 1.8 años. El 75% de la población tiene un año de experiencia desempeñando el cargo actual. El 25% lleva dos años de experiencia en el cargo que desempeña en la actualidad (Ver figura 6).

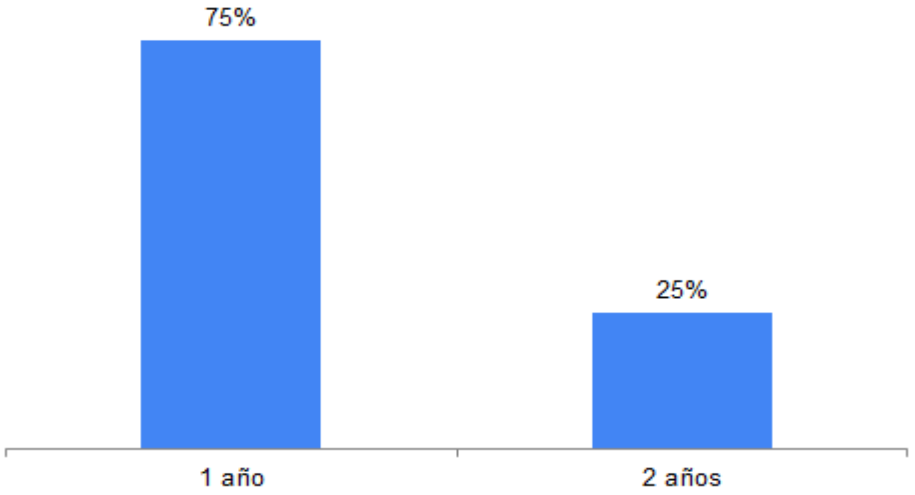


Figura 6. Tiempo de trabajo en el cargo actual de la población

A continuación, se presentan un análisis de la forma como se realizan las pruebas de unidad y componentes en el área de tecnología de la Universidad Mariana.

- Roles

El equipo de desarrollo, soporte y mantenimiento de software del Área de Tecnología manifiestan que realizan los siguientes tipos de pruebas: unitarias y componentes, de aceptación y de Interfaces Graficas de Usuario (GUI por sus siglas en ingles). Los roles encargados de realizar los diferentes tipos de pruebas son: Analista de Sistemas, Directora del Área de Tecnología y Soporte Técnico.

- Actividades

Los integrantes del equipo plantean que existen dos caminos para el desarrollo de pruebas. El primero, para cambios o mejoras en el software, se procede a identificar el tipo de pruebas a realizar, posteriormente se elaboran los casos de prueba a partir de los requerimientos o necesidades solicitadas, los cuales, se ejecutan en cada ciclo. Para productos software desarrollados por terceros, se procede a enviar los requerimientos o necesidades a la casa productora a través de la mesa de ayuda del proveedor.

El segundo camino inicia cuando se identifica un problema o error en los productos software a cargo del área. Entonces, se procede en primer lugar, a documentar el error. Posteriormente, se le asigna una prioridad al problema. Finalmente, se direcciona el error identificado a la persona encargada de solventar el problema o se registra en la mesa de ayuda del proveedor si el software fue desarrollado por terceros.

- Herramientas

Para realizar las pruebas de software los profesionales en el área de tecnología de la universidad usan las siguientes herramientas:

- Mantis: herramienta software que se utiliza para registrar y hacer un seguimiento del progreso de los errores y problemas en el software durante el desarrollo.
- Postman: herramienta que se utiliza para probar APIs (por sus siglas en inglés, Application Programming Interface) web. Permite a los desarrolladores compartir, probar y documentar APIs de una manera fácil y eficiente.
- Apache JMater: herramienta que se utiliza para analizar y medir el rendimiento de una variedad de servicios, enfocada principalmente aplicaciones web. JMeter permite simular un gran número de usuarios concurrentes, solicitando páginas web u otros servicios y analizando el rendimiento general del servidor bajo diferentes cargas.

- Excel: herramienta que sirve para documentar información, errores y problemas. Puedes usar Excel para crear documentos que contengan datos importantes, listas, gráficos y fórmulas.

Cuando se identifican errores en el software se usan las herramientas Mantis y Excel, siendo esta última, la más usada con un 75% como se observa en la Figura 7.

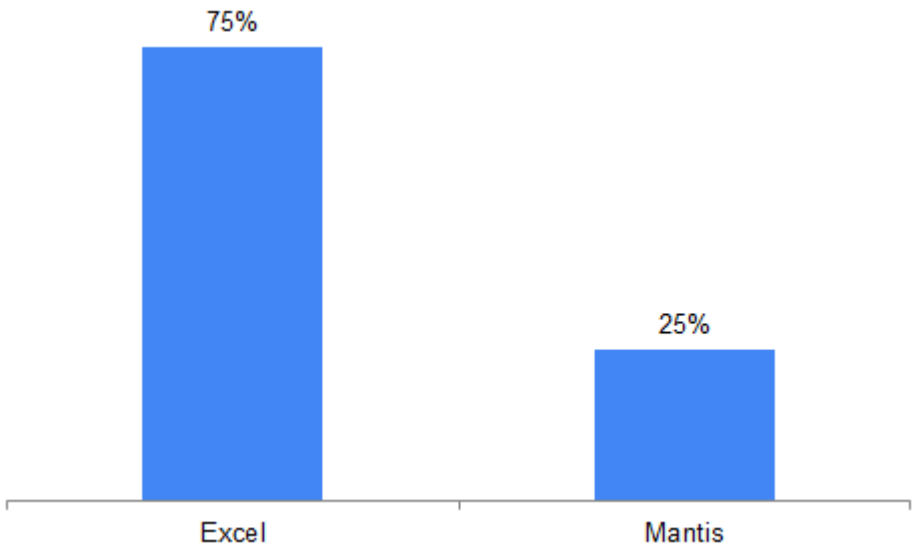


Figura 7. Herramientas utilizadas en la población

- Flujos de control

Los integrantes del equipo están encargados de identificar, diagnosticar y resolver problemas o errores tanto en funcionalidades existentes como en nuevas. Este proceso implica un diagnóstico detallado, resolución efectiva del problema o error, documentación exhaustiva y corrección necesaria. Además, cuando se requiere modificar el software en respuesta a un problema o necesidad funcional identificada, se lleva a cabo un ajuste en la funcionalidad seguido de la implementación de la modificación en el software. Asimismo, al solicitar un informe detallado de errores o problemas en el software, se realiza la generación y presentación del informe correspondiente, el cual se documenta y culmina con la entrega del plan de pruebas.

2.1.6. Síntesis de la caracterización

El instrumento (encuesta) fue aplicado a los 4 profesionales del área de tecnología de la Universidad Mariana, el 75% de los miembros tienen un título profesional y en su experiencia laboral en promedio, han estado ejerciendo la profesión durante 5 años.

Las pruebas que se realizan son: pruebas unitarias, pruebas de componentes, pruebas de aceptación y pruebas de Interfaces Gráficas de Usuario (GUI). Existen dos caminos para el desarrollo de pruebas: uno para cambios o mejoras en el software y otro para identificar y solucionar problemas o errores en los productos de software existentes.

Los roles encargados de realizar las pruebas son el Analista de Sistemas, la Directora del Área de Tecnología y el Soporte Técnico. Y los integrantes utilizan varias herramientas para las pruebas de software, incluyendo Mantis para el seguimiento de errores y problemas, Postman para probar APIs web, Apache JMeter para analizar y medir el rendimiento de aplicaciones web y Excel para documentar información, errores y problemas. Excel es la herramienta más utilizada, siendo utilizada en el 75% de los casos, seguida por Mantis.

2.2. Proceso para el desarrollo y ejecución de pruebas automáticas de unidad desde el desarrollo ágil de software

En esta sección, se presentan la forma como se diseñó un proceso para el desarrollo y ejecución de pruebas automáticas de unidad desde el desarrollo ágil de software, con base en la caracterización. Para proponer el diseño, se realizaron las actividades que se presentan en la Figura 8.

Figura 8. Actividades para elaborar el diseño de la propuesta

2.2.1. Referentes teóricos para la representación del proceso

Dentro del sector empresarial u organizacional, la presentación o modelado de un proceso facilita la comprensión del funcionamiento de una empresa o área; dado que permite la visualización de los objetivos, actividades y participantes que interactúan, la forma en que se desarrollan o ejecutan las acciones y brindan la oportunidad de identificar errores para corregirlos.

En 2001, se desarrolló un lenguaje XML para presentar procesos comerciales y su ejecución (Boiko B., 2001).

Actualmente existe una notación para representar los procesos comerciales de una organización llamada notación de gestión de procesos de negocios, BPMN por sus siglas en inglés. La orquestación es un modelo central, es decir, se basa en la presentación de los procesos comerciales que lleva a cabo una empresa. Según Debevoise Neilson (2008), la coreografía es la presentación del comportamiento que los participantes deben tener en un proceso comercial, y la cooperación es la comunicación entre los participantes de un proceso comercial que puede incluirse en una coreografía.

La representación de un proceso puede ser realizada por diferentes modelos que se clasifican de acuerdo con la descripción o los detalles; estos incluyen: tarjetas de proceso que son diagramas simples de actividades; descripción de los procesos que muestran más información sobre un proceso y modelos de proceso que representan gráficos con un mayor nivel de detalle para analizar y simular el proceso. Los elementos principales que BPMN utiliza para presentar un proceso son: actividades, eventos, puertas de enlace y flujos de secuencia (Briol P., 2008).

Para Briol (2008), las actividades son las acciones que se llevan a cabo en un proceso. Se pueden clasificar de acuerdo con los detalles que representan, por ejemplo: atómicas y compuestas (Ver Fig. 9).

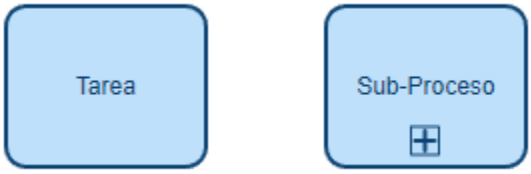


Figura 9. Representación de actividad atómica y compuesta

Los eventos son casos que pueden ocurrir en el flujo de un proceso de negocio para que puedan comenzar, retrasar, interrumpir o terminar (Briol P., 2008). Generalmente tiene un resultado o una razón para iniciarlo (ver Fig. 10).

Figura 10. Representación de evento de inicio y fin

Para Briol (2008), las puertas de enlace o Gateway son elementos que controlan la división o unión de las calles presentadas en el flujo del proceso por un rombo que puede variar según la acción (ver Fig. 11).

Figura 11. Representación de puertas de enlace o Gateway

Para Briol (2008), el flujo de secuencia es responsable de la conexión de los elementos involucrados en la presentación del proceso de negocio, y muestra las carreteras registradas, su forma es una flecha (ver Fig. 12).

Figura 12. Representación del flujo de secuencia

El modelado de un proceso de negocio se puede llevar a cabo utilizando herramientas que faciliten la visualización, incluidos: Bizagi Modeler, Lucichart, Bonita, entre otros.

A manera de ejemplo, se realiza la representación del modelo de proceso para validar un instrumento de recopilación de información. En primer lugar, se definen las actividades. Por cada actividad se establece el elemento de entrada, el producto de trabajo y el rol que lo desempeña.

En la Tabla 6, se pueden observar las actividades que tiene el proceso.

Tabla 6. Actividades del proceso

Actividad	**Elemento de entrada**	**Producto de trabajo**	**Rol**
Elaborar instrumento de recopilación de datos.	Indicadores de medición	Cuestionario de recopilación de datos	Investigador
Enviar cuestionario.	Cuestionario de recopilación de datos	e-mail del envío redactado	Investigador
Revisar cuestionario.	Indicadores de medición	Cuestionario de recopilación de datos	Asesor

Actividad	Elemento de entrada	Producto de trabajo	Rol
Realizar ajustes al cuestionario.	Indicadores de medición	Cuestionario de recopilación de datos	Investigador
Enviar cuestionario para prueba piloto.	Cuestionario de recopilación de datos	e-mail del envió redactado	Investigador
Diligenciar encuesta.	Indicadores de medición	Cuestionario de recopilación de datos	Funcionario Área de tecnología
Enviar novedades al investigador.	Cuestionario de recopilación de datos	e-mail del envió redactado	Funcionario Área de tecnología
Analizar información recopilada.	Indicadores de medición	Cuestionario de recopilación de datos	Investigador
Aplicar cuestionario a la población.	Cuestionario de recopilación de datos	e-mail del envió redactado	Investigador

Con base en las actividades identificadas para el proceso de negocio descrito en la Tabla 6, se diseñó y representó mediante la herramienta *Bizagi Modeler* el proceso de negocio del ejemplo (Ver Figura 13).

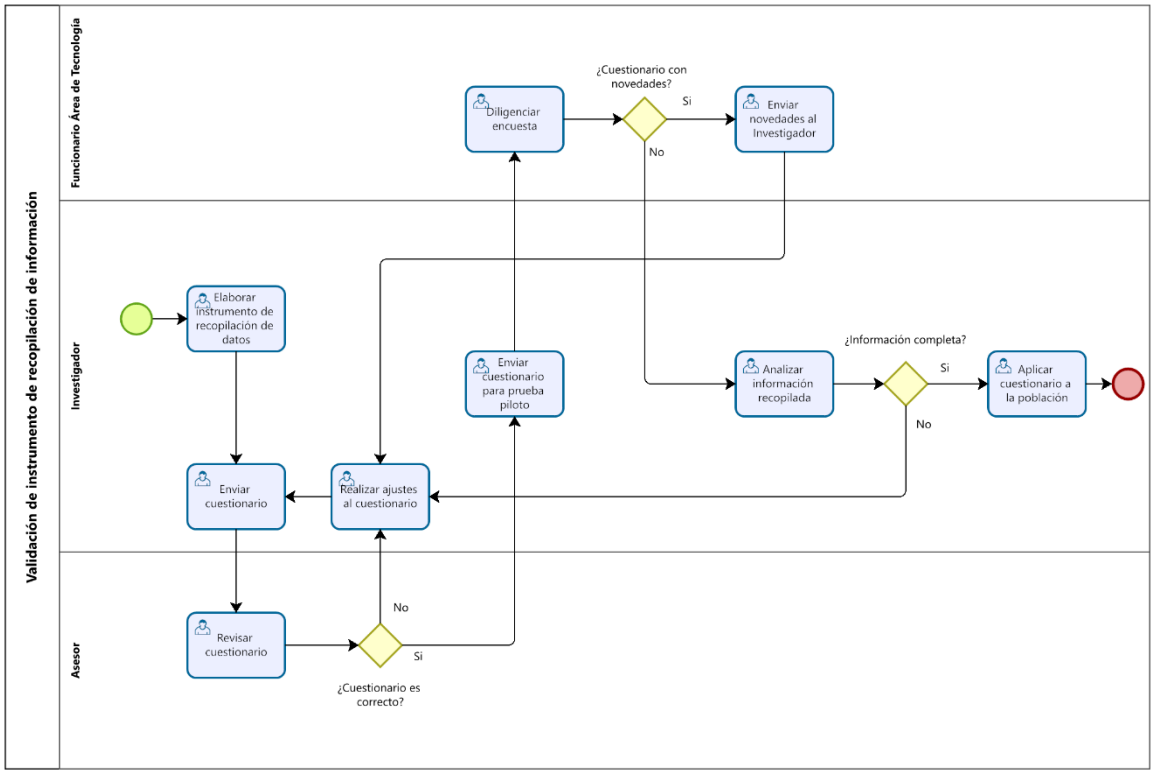


Figura 13. Representación del modelo de proceso con BPMN

2.2.2. Modelo del proceso actual

El equipo de desarrollo, soporte y mantenimiento de software del Área de Tecnología manifiestan que los roles encargados de realizar los diferentes tipos de pruebas son: Analista de Sistemas, Directora del Área de Tecnología y Soporte Técnico.

Los integrantes del equipo plantean que existen dos caminos para el desarrollo de pruebas, es decir, dos procesos. La primera ruta corresponde con cambios o mejoras en el software. En este sentido, se procede a identificar el tipo de pruebas a realizar, posteriormente se elaboran los casos de prueba a partir de los requerimientos o necesidades solicitadas, los cuales, se ejecutan en cada ciclo.

En la Tabla 7, se pueden observar las actividades con elementos de entrada, productos de trabajo y el rol de quien ejecuta la actividad.

Tabla 7. Actividades del proceso con cambios o mejora en el software

Actividad	**Elemento de entrada**	**Producto de trabajo**	**Rol**
Identificar el tipo de prueba a realizar.	Documentos que describen las funcionalidades del software que se está realizando	Software probado	Analista de Sistemas
Elaborar los casos de prueba.	Software probado	Informe, registro y casos de pruebas	Analista de Sistemas
Ejecutar los casos de prueba en cada ciclo.	Informe, registros y casos de pruebas actualizado	Software probado	Analista de Sistemas

Para productos software desarrollados por terceros, se procede a enviar los requerimientos o necesidades a la casa productora a través de la mesa de ayuda del proveedor, razón por la cual no se realizan pruebas por parte del equipo del área.

El segundo camino inicia cuando se identifica un problema o error en los productos software a cargo del área. Entonces, se procede en primer lugar, a documentar el error. Posteriormente, se le asigna una prioridad al problema. Finalmente, se direcciona el error identificado a la persona encargada de solventar el problema o se registra en la mesa de ayuda del proveedor si el software fue desarrollado por terceros.

En la Tabla 8, se pueden observar las actividades con elementos de entrada, productos de trabajo y el rol de quien ejecuta la actividad.

Tabla 8. Actividades del proceso para errores identificados en el software

Actividad	**Elemento de entrada**	**Producto de trabajo**	**Rol**

Actividad	**Elemento de entrada**	**Producto de trabajo**	**Rol**
Documentar el error	Descripción del error	Formato de registro del error	Analista de Sistemas
Almacenar el error	Formato de registro del error	Registro del error	Herramienta Mantis
Asignar una prioridad al problema	Registro del error	Registro del error con prioridad	Analista de Sistemas
Registrar la prioridad del error	Registro del error con prioridad	Registro del error actualizado	Herramienta Mantis
Direccionar el error identificado	Registro del error con prioridad	Registro de asignación del error al desarrollador	Analista de Sistemas
Almacenar el registro de la asignación del error al desarrollador	Registro de asignación del error al desarrollador	Registro del error actualizado	Herramienta Mantis
Direccionar el error identificado en la mesa de ayuda del proveedor	Registro del error con prioridad	Registro de asignación del error en la mesa de ayuda del proveedor	Analista de Sistemas
Almacenar error	Registro del error actualizado	Registro del error almacenado en la mesa de ayuda	Software mesa de ayuda del proveedor

Para realizar las pruebas de software los profesionales en el área de tecnología de la universidad usan las siguientes herramientas como: Mantis, Postman, Apache JMater y Excel. Cuando se identifican errores en el software se usan las herramientas Mantis y Excel.

En la Figura 14, se presenta el modelo de proceso utilizando notación BPMN para las pruebas cuando se presentan cambios o mejoras en el software a cargo del área de Tecnología.

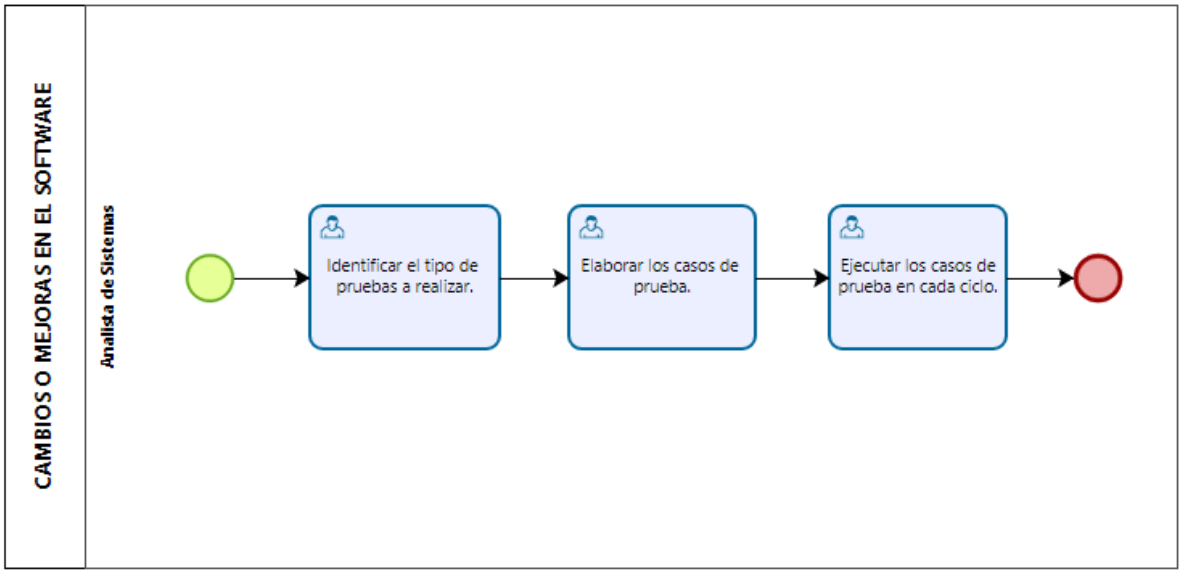


Figura 14. Representación del modelo de proceso con BPMN de cambios en el software

En la Figura 15, se presenta el modelo de proceso utilizando notación BPMN para las pruebas cuando se presentan errores en el software a cargo del área de Tecnología.

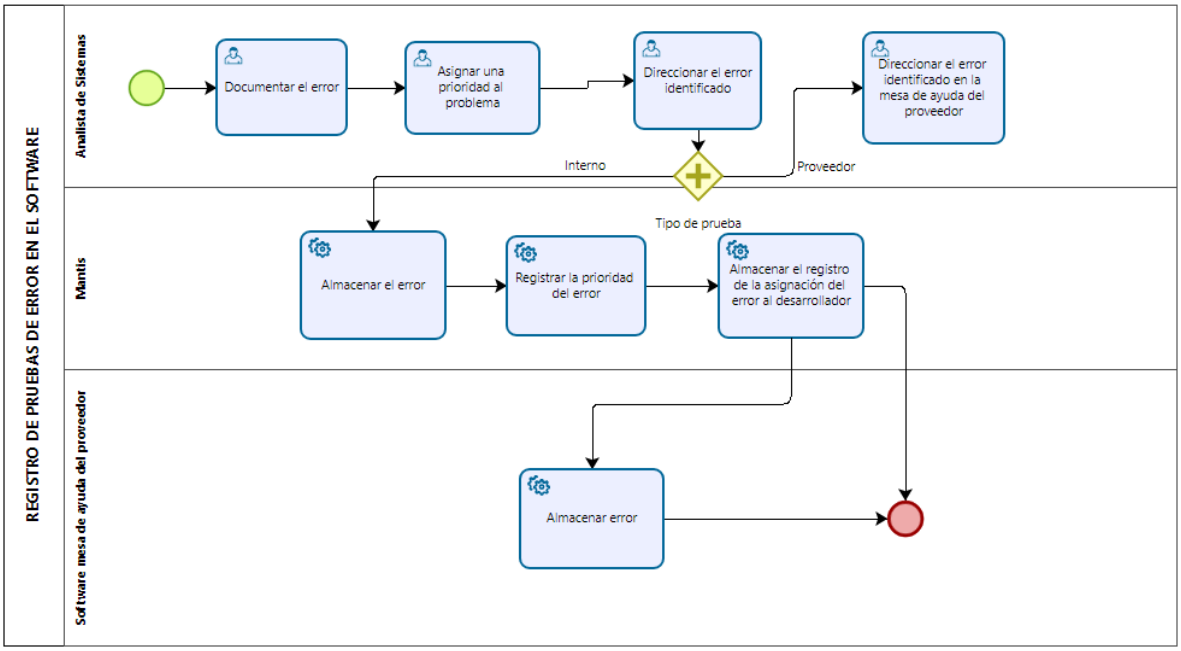


Figura 15. Representación del modelo de proceso con BPMN de errores en el software

2.2.3. Elementos de la propuesta

En el estudio llevado a cabo por Calpa y Garzón (2022), se propone la implementación del proceso "Success Method UCT" para realizar pruebas automáticas de unidad y componentes en el contexto del desarrollo ágil de software. Este enfoque tiene como objetivo facilitar la transición de las pruebas manuales a las pruebas automáticas, basándose en los modelos presentados en AGILUS, así como en las recomendaciones de aplicaciones de Pruebas Para Scrum (PPS) y la Propuestas para Desarrollo Incremental Iterativo (PIID por sus siglas en inglés).

De acuerdo con Calpa y Garzón (2022), en la Figura 16 se ilustra los elementos tomados de varios métodos, destacados en gris, junto con los métodos específicamente diseñados para este enfoque, resaltados en verde, que constituyen la propuesta Success Method UCT.

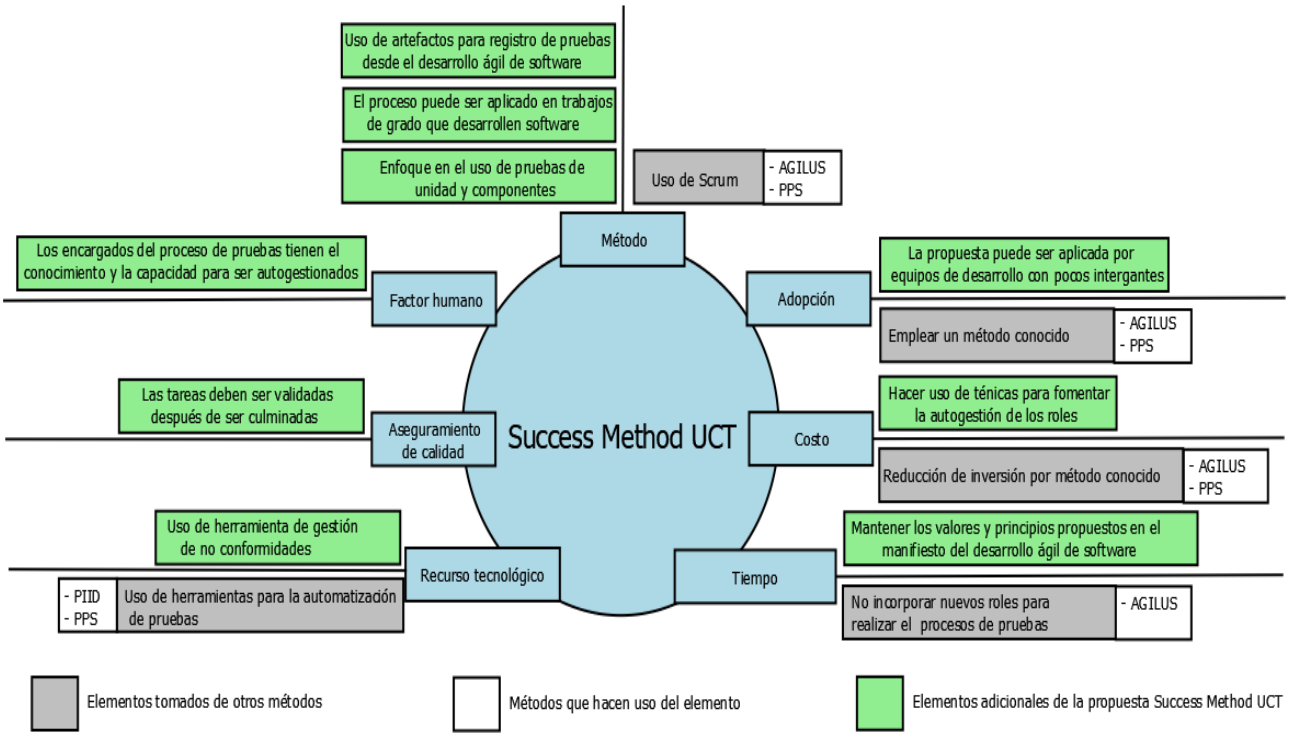


Figura 16. Esquema de la propuesta Succes Method UCT

Fuente: Tomado de Calpa y Garzón (2022)

Para garantizar que los principios del desarrollo ágil no se vean afectados, Calpa y Garzón (2022) proponen una solución que respeta los roles definidos en Scrum. En lugar de cambiar los roles existentes, la idea es equipar a estos roles con las habilidades y conocimientos necesarios para llevar a cabo las pruebas automáticas de unidad. Esta estrategia también fomenta prácticas de autogestión, ya que cada rol adquirirá nuevas habilidades.

Esta propuesta presentada por Calpa y Garzón (2022) muestra varias ventajas, incluida la capacidad de implementarse en equipos de desarrollo pequeños y su aplicabilidad en proyectos, ya que se centra en las prácticas de pruebas en lugar de los roles que las desempeñan. Esto permite que cualquier miembro del equipo realice las pruebas de acuerdo con sus necesidades, lo que garantiza la calidad del software al verificar cada actividad una vez que se haya completado.

En la Figura 17, se puede apreciar el modelo de proceso utilizando notacion BPMN sobre la propuesta *Succes Method UCT* de Calpa y Garzón (2022).

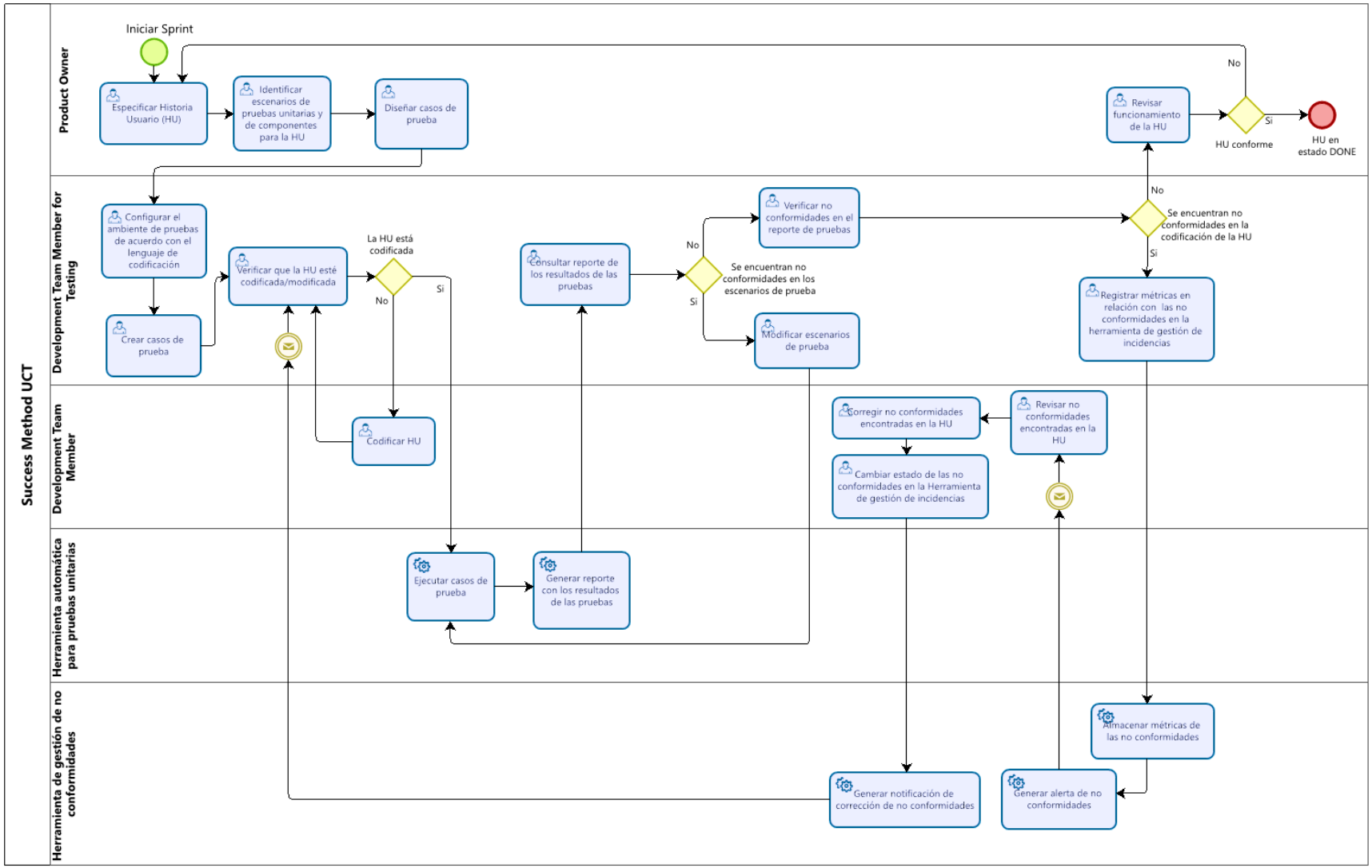


Figura 17. Diagrama BPMN de la propuesta Success Method UCT

Fuente: Tomado de Calpa y Garzón (2022)

Teniendo como base la propuesta "Success Method UCT" de Calpa y Garzón (2022), y las actividades que desarrollan los responsables del desarrollo y mantenimiento de software en el Área de Tecnología de la Universidad Mariana; se procede a establecer los elementos que constituirán la propuesta con el fin de diseñar un modelo de proceso de negocio utilizando notación BPMN. Los elementos son: roles, actividades y flujos de control.

- Roles

Los roles encargados de realizar tipos de pruebas con el modelo del proceso propuesto son: Analista de sistemas, herramienta de gestión de no conformidades, desarrollador, software de la mesa de ayuda del proveedor.

- Actividades

Las actividades que se encargan de hacer cada rol con el modelo del proceso propuesto son: La identificación del tipo de prueba a realizar, seguido por la creación de casos de prueba. Posteriormente, se ejecutan los casos de prueba y, en caso de identificar no conformidades, se registran con prioridad. Estas no conformidades se almacenan y, simultáneamente, se registran en la ayuda de mesa del proveedor. Se procede a enviar mensajes de notificación sobre las no conformidades identificadas, y al recibir los mensajes, se realiza la recepción de la notificación. La siguiente fase implica la actualización del estado de la notificación y el almacenamiento correspondiente de la no conformidad. Y finalmente se actualiza el estado general de la no conformidad, asegurando así un completo seguimiento y este mismo se almacena cerrando así el proceso.

- Flujos de control

Los encargados de las actividades deben identificar, registrar y almacenar las no conformidades. Este proceso implica un resultado y registro detallado de la no conformidad. Cuando no se identifica una no conformidad los casos de prueba finalizan sin embargo si se identifican entonces pasarían a almacenarse, enviarse y actualizarse en sus debidas áreas. Además, cuando hay una no conformidad en el área de TI, se envía una notificación al desarrollador para recepcionar y actualizar el estado de la no conformidad para almacenarlo y finalizar los casos de prueba.

En la Tabla 9, se pueden observar las actividades con elementos de entrada, productos de trabajo y el rol que desempeña cada actividad.

Tabla 9. Actividades del proceso propuesto

Id	**Actividad**	**Elemento de entrada**	**Producto de trabajo**	**Rol**
Act1	Identificar el tipo de prueba a realizar	Documentos que describen las funcionalidades del software que se está realizando	Software probado	Analista de Sistemas
Act2	Crear casos de prueba	Software probado	Informe, registro y casos de pruebas	Analista de Sistemas
Act3	Ejecutar caso de prueba	Informe, registros y casos de pruebas actualizado	Software probado	Analista de Sistemas
Act4	Registrar no conformidad identificada con prioridad	Registro de la no conformidad con prioridad	Registro de la no conformidad actualizado	Analista de Sistemas
	Almacenar no conformidad	Formato de registro de la no conformidad	Registro de la no conformidad con prioridad	Herramienta de gestión de no conformidades
Act5	Registrar no conformidad identificada en	Registro de la no conformidad con prioridad	Registro de asignación de la no	Analista de Sistemas

Id	Actividad	Elemento de entrada	Producto de trabajo	Rol
	la ayuda de mesa del proveedor		conformidad en la mesa de ayuda del proveedor	
	Enviar mensaje de notificación	Registros de las no conformidades	e-mail del envió redactado	Herramienta de gestión de no conformidades
Act6	Recepcionar mensaje de no conformidad	Registros de las no conformidades	Registros de las no conformidades	Desarrollador
Act7	Registrar la resolución de la no conformidad	Registros de las no conformidades	Registros de las no conformidades	Desarrollador
Act8	Actualizar estado de la notificación	Registro del estado actualizado	Registro del estado actualizado	Desarrollador
	Almacenar no conformidad	Formato de registro de la no conformidad	Registro de la no conformidad con prioridad	Software mesa de ayuda del proveedor
Act9	Actualizar estado de la no conformidad	Formato de registro de la no conformidad	Registro de la no conformidad con prioridad	Analista de Sistemas
	Almacenar estado de la no	Registro de la no conformidad con	Registro de la no	Herramienta de gestión de

Id	Actividad	Elemento de entrada	Producto de trabajo	Rol
	conformidad	prioridad	conformidad con prioridad	no conformidades

2.2.4. Modelo de proceso propuesto

De acuerdo con los elementos establecidos para la propuesta, en la Figura 18, se muestra el modelo de proceso de negocio elaborado utilizando la notación BPMN para el desarrollo de pruebas en el Área de Tecnología de la Universidad Mariana.

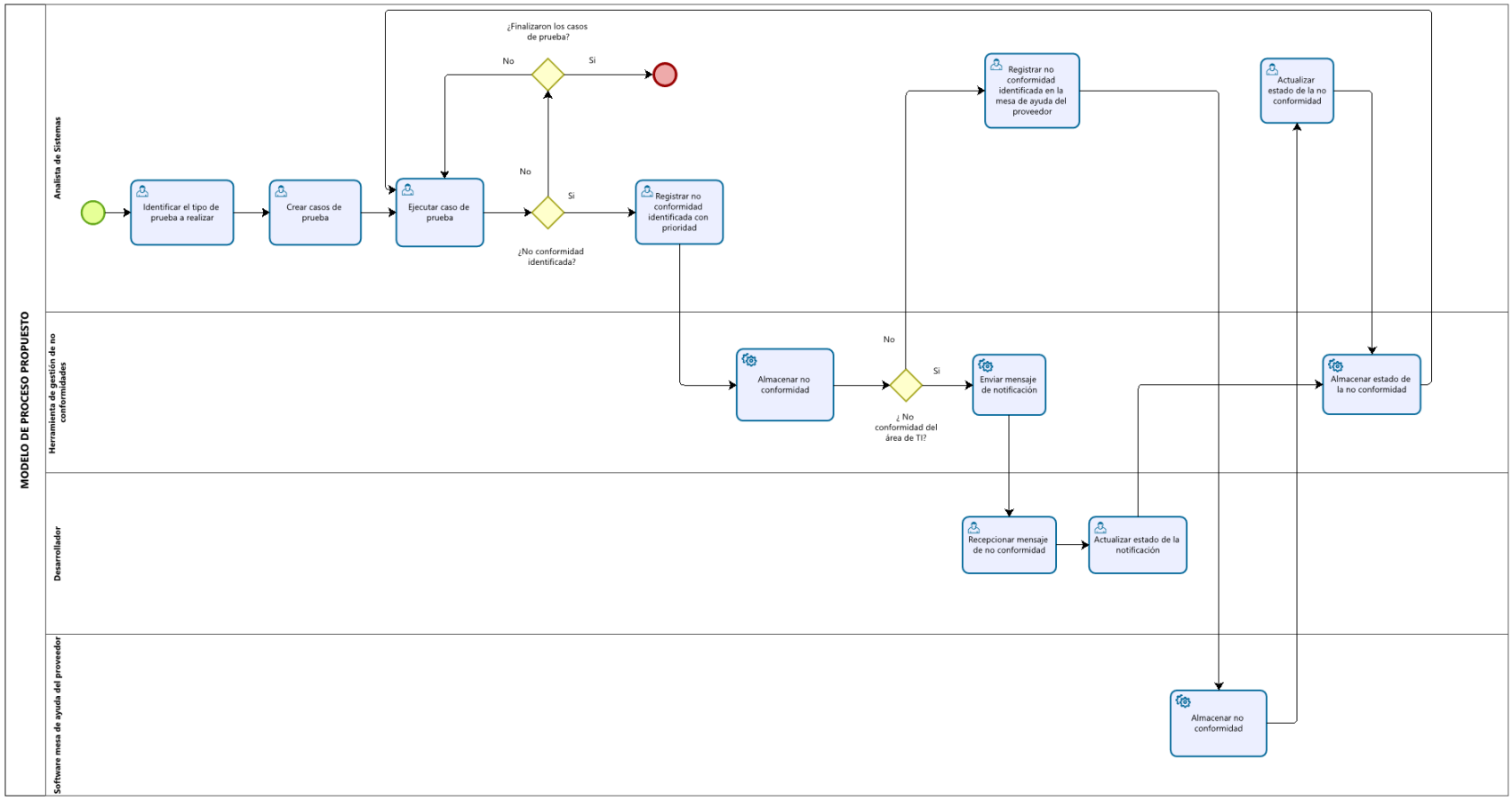


Figura 18. Diagrama BPMN del modelo propuesto.

2.2.5. Síntesis

En el ámbito empresarial u organizacional, la representación o modelado de procesos es fundamental para comprender el funcionamiento de una empresa o área. Esto implica visualizar objetivos, actividades y participantes, así como entender la ejecución de acciones, identificando oportunidades para corregir errores. En 2001, se introdujo un lenguaje XML para representar procesos comerciales. Actualmente, la BPMN (Notación de Gestión de Procesos de Negocios) proporciona una notación estándar para representar procesos empresariales, donde la orquestación destaca la ejecución de procesos, y la coreografía describe el comportamiento de los participantes. La cooperación implica la comunicación entre ellos. La representación de procesos puede realizarse mediante diversos modelos, como tarjetas de proceso, descripciones detalladas y modelos gráficos con elementos clave como actividades, eventos, puertas de enlace y flujos de secuencia, según BPMN. Para facilitar el modelado de procesos, existen herramientas como Bizagi Modeler, Lucichart, Bonita, entre otras.

El equipo de desarrollo, soporte y mantenimiento de software del Área de Tecnología identifica roles clave, como Analista de Sistemas, Directora del Área de Tecnología y Soporte Técnico, responsables de realizar pruebas en dos procesos: cambios o mejoras en el software. Según Calpa y Garzón (2022), proponen el "Success Method UCT" para pruebas automáticas en desarrollo ágil, respetando los roles de Scrum y promoviendo habilidades autogestionadas. La propuesta destaca elementos específicos y ventajas, aplicándose en equipos pequeños y proyectos, centrándose en prácticas de prueba más que en roles.

Basándose en "Success Method UCT," el Área de Tecnología de la Universidad Mariana establece elementos para diseñar un modelo de proceso de negocio con notación BPMN. Los roles involucrados son Analista de Sistemas, herramienta de gestión de no conformidades, desarrollador y software de mesa de ayuda. Las actividades incluyen identificación del tipo de prueba, creación y ejecución de casos de prueba, registro y notificación de no conformidades. Los flujos de control aseguran la identificación y registro detallado de no conformidades, con procedimientos específicos para conformidades y acciones correspondientes en caso de no conformidades, garantizando un seguimiento completo y almacenamiento del proceso al cerrarse.

2.3. Validación del proceso para desarrollo y ejecución de pruebas automáticas de unidad propuesto

En esta sección, se describe la forma como se realizó la validación del proceso para desarrollo y ejecución de pruebas automáticas de unidad propuesto en el numeral 2.2. En la Figura 19, se puede observar el proceso realizado.

Figura 19. Actividades para la validación del proceso

2.3.1. Construcción del instrumento de validación

Para la construcción del instrumento se desarrolló las 3 primeras etapas propuestas por Kitchenham y Pfleeger (2008) y Guerrero-Calvache y Hernández (2023) que corresponde con: definición de objetivos, diseño del instrumento y elaboración del instrumento.

- Definición de objetivos

Según los autores Kitchenham y Pfleeger (2008) la primera etapa para la construcción del instrumento de recolección de información es la definición de objetivos. Por lo anterior, el objetivo de la encuesta es validar el proceso para desarrollo y ejecución de pruebas automáticas de unidad diseñando en el numeral 2.2.

- Diseño del instrumento

Según Kitchenham y Pfleeger (2008), la planificación de una encuesta puede clasificarse en dos categorías: transversal y longitudinal. Este análisis en particular se clasifica como transversal, ya que la herramienta se administró a profesionales de la industria del software que forman parte de equipos que aplican métodos ágiles en un momento específico, en este caso, durante el periodo comprendido entre septiembre y octubre de 2022.

- Elaboración del instrumento

La creación del instrumento abarca la especificación de los tipos de preguntas y respuestas a incorporar, así como el formato, la disposición y la longitud del cuestionario. En la elaboración del instrumento, se tuvieron en cuenta los resultados obtenidos en la ejecución previa de un mapeo sistemático de la literatura, considerando las concepciones identificadas sobre la productividad del equipo en ASD y los factores que intervienen en su medición. Este enfoque contribuyó al diseño de las preguntas que formarán parte del cuestionario. Se optó por preguntas cerradas, complementadas con preguntas abiertas para profundizar en las respuestas. Las preguntas cerradas incluyeron opciones de respuesta múltiple mediante una escala Likert como método de evaluación subjetiva.

El cuestionario abarcó la evaluación de 12 ítems y se estimó que tomaría aproximadamente 10 minutos completarlo. El instrumento se estructuró en cuatro secciones: Generalidades, información sociodemográfica, actividades y flujos de control.

2.3.2. Socialización del proceso propuesto

La socialización del modelo el cual se puede ver en la Figura 18 fue propuesta a los 4 profesionales del área de tecnología de la Universidad Mariana que hacen parte del equipo de desarrollo, soporte y mantenimiento de software, en el periodo comprendido entre agosto y septiembre de 2023.

2.3.3. Aplicación del instrumento de validación

Después de la socialización del modelo propuesto, el cuestionario fue aplicado a los 4 profesionales del área de tecnología de la Universidad Mariana, en el periodo comprendido entre agosto y septiembre de 2023. En el link https://forms.gle/2868x6e968kdDH3b9 se puede consultar el instrumento de validación.

2.3.4. Análisis al proceso propuesto

- Aspectos positivos

Los integrantes del equipo plantean que lo mejor del proceso propuesto es el uso de herramientas para automatizar las pruebas de software y se hace uso de recursos tecnológicos para registrar las no conformidades durante las pruebas. Sin embargo, mencionan que las

mayores dificultades para implementar este proceso son: más herramientas para automatizar las pruebas, los roles que plantea el modelo y falta de personal con el perfil de Analista de Pruebas.

Teniendo en cuenta lo anterior los integrantes adoptarían esta propuesta con un nivel de aceptación muy alto y alto en un 50% como se puede ver en la Figura 20.

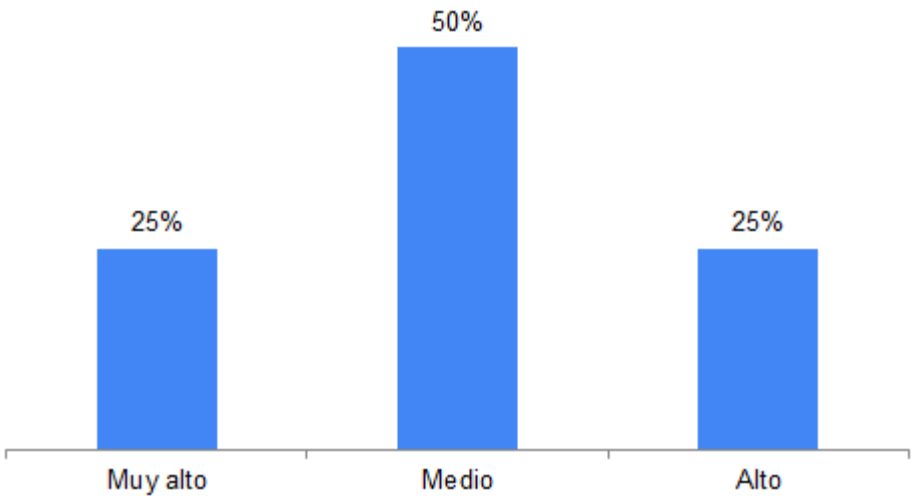


Figura 20. Nivel de aceptación de la propuesta.

- Aspectos por mejorar

Los integrantes del equipo mencionan que los aspectos que mejorarían serian: agregar las demás herramientas con las que trabajan para realizar las pruebas, también que en el rol de Desarrollador se podría añadir el proceso de generación de un informe de la solución de la inconformidad, se debería evaluar que hay aplicaciones que son directamente del proveedor las cuales no se las puede actualizar a nivel técnico por parte de la universidad, lo que conlleva a que el encargado de reparar o solucionar los problemas es el proveedor. También mencionan que hay desarrollos propios de la universidad y el proceso es diferente al modelo presentado. Por último, el rol esperado para el Analista de Sistemas es un Analista de pruebas, pero actualmente no se cuenta con dicho perfil.

- Aspectos por incluir

Los informantes, de manera general manifiestan que complementarían el modelo de proceso propuesto con la elaboración de plantillas específicas para llevar a cabo las pruebas.

2.3.5. Ajustes a la propuesta

Con base en los aspectos por mejorar e incluir en el modelo de proceso propuesto, se hace una síntesis de los requerimientos y posteriormente se plasman en una nueva versión de la representación.

Tabla 10. Requerimientos sobre aspectos por mejorar e incluir a la propuesta

RQ1	Incluir en el rol de Desarrollador una actividad para el registro de la solución de una inconformidad.
RQ2	Cambiar el rol de Analista de Sistemas por Analista de pruebas.
RQ3	Proponer un conjunto de plantillas para llevar a cabo las pruebas.

Con base en los requerimientos anteriormente planteados se modifica el modelo de proceso de la propuesta, la cual se puede observar en la Figura 21.

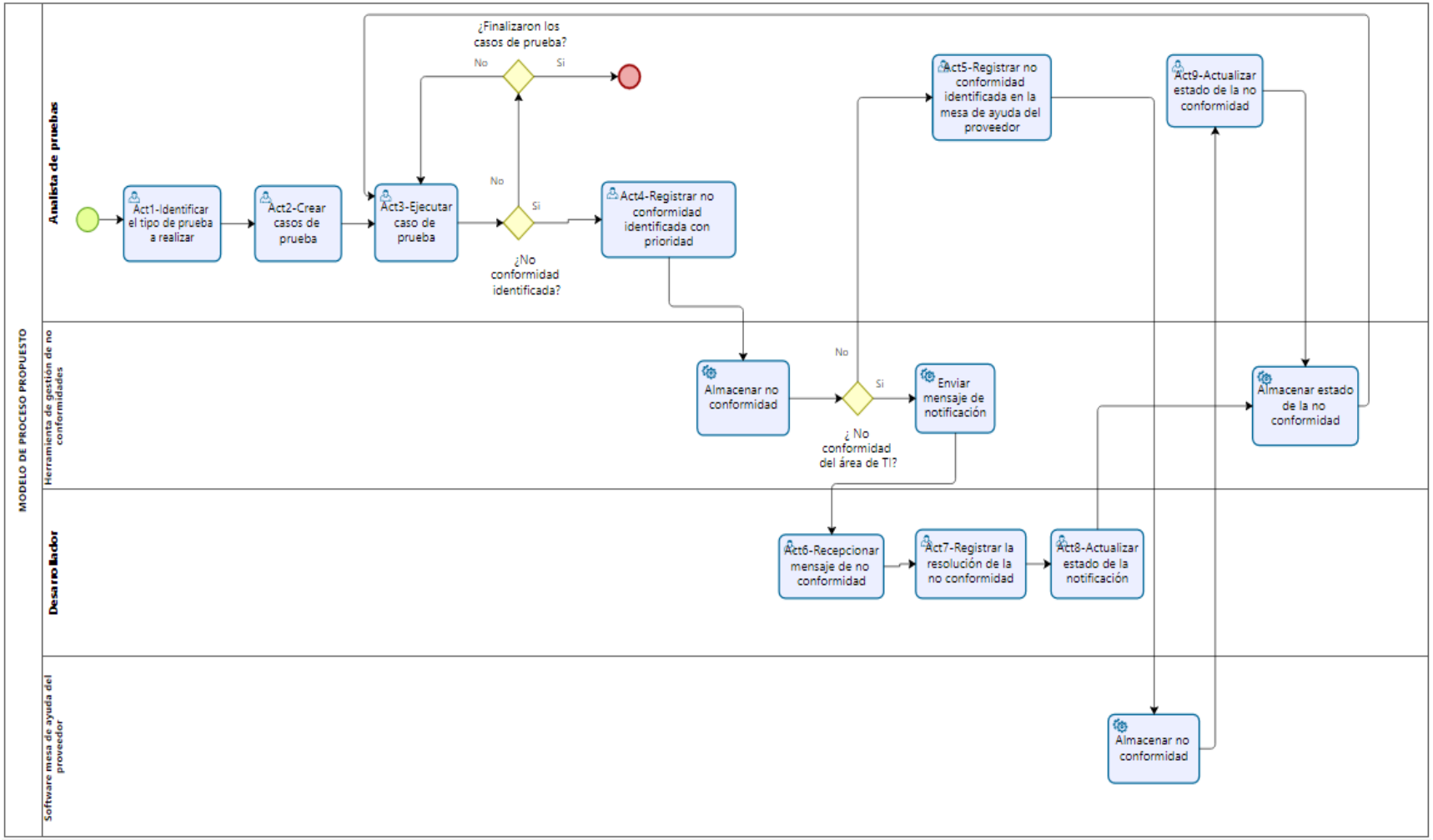


Figura 21. Modelo del proceso de la propuesta modificado.

Con base en el modelo de proceso actualizado (Ver Figura 21) se procede a elaborar el formato para la actividad **ACT-1 Identificar el tipo de prueba a realizar.** En la Tabla 11 se puede observar el formato establecido para determinar el tipo de prueba que se realizará. El **Id de la prueba** corresponde con un identificador único para diferenciar la prueba dentro de un conjunto de pruebas. El **Nombre de la prueba** corresponde con una descripción corta de la prueba que se realizará. La **Descripción** hace referencia a una explicación ampliada de lo que se pretende con el desarrollo de la prueba. El **Tipo de prueba** se refiere a seleccionar una de las posibles pruebas que se pueden realizar entre las que se encuentran: [Funcional][Unitaria][Interfaz UI]. El **Usuario** corresponde con el nombre y rol de la persona que identifica el tipo de prueba. La **Fecha de creación** corresponde con el día, mes y año de cuando se realiza la identificación de la prueba. La **Fecha de actualización** corresponde con el día, mes y año de cuando se actualizan los datos de la identificación de la prueba. El **Resultado** es una breve descripción de lo que se espera obtener con la realización de la prueba.

Tabla 11. Formato para identificar el tipo de prueba

Id de la prueba:	
Nombre de la prueba:	
Descripción:	
Tipo de prueba:	[Funcional][Unitaria][Interfaz UI]
Usuario:	
Fecha de creación:	
Fecha de actualización:	
Resultado:	

A continuación, se presenta un ejemplo de un tipo de prueba que se identifica. En la Tabla 12, se muestra el formato diligenciado para un tipo de prueba que se realiza en el área.

Tabla 12. Ejemplo del formato de tipo de prueba

Id de la prueba:	PR-1

Nombre de la prueba:	Buscar un estudiante
Descripción:	Se requiere matricular a un estudiante que no se encuentra inscrito en ningún programa de la Universidad.
Tipo de prueba:	[Unitaria]
Usuario:	Geancarlo Díaz Marín – Analista de pruebas
Fecha de creación:	14-2-2024
Fecha de actualización:	
Resultado:	Se informe que el estudiante no se encuentra inscrito en la universidad.

Continuando con el modelo actualizado se procede a elaborar el formato para la actividad **ACT-2 Crear casos de prueba.** Con base en el trabajo realizado por Calpa y Garzón (2022) en la Tabla 11. se puede observar el formato establecido para la creación de casos de prueba. El **CPId** corresponde con un valor único que identifica el caso de prueba. El **Nombre** corresponde con un nombre descriptivo del caso de prueba, se recomienda usar el prefijo "test". La **Clase** hace referencia a un objeto al que pertenecen los métodos a validar. El **Método** se refiere al método al que aplican los casos de prueba. El **RQ** corresponde a un valor único que identifica a un requerimiento de otros dentro de las solicitudes hechas al área. El **Escenario** corresponde a un valor único que identifica el escenario de prueba de acuerdo con el requerimiento. Los **Valores de entrada** corresponden a datos que se ingresan para ejecutar el caso de prueba. El **Resultado esperado** es el resultado que se debe obtener al ejecutar el caso de prueba, según el escenario al que corresponda.

Tabla 13. Formato para crear casos de prueba

CPId	**Nombre**	**Clase**	**Método**	**RQ**	**Escenario**	**Valores de entrada**	**Resultado esperado**

Fuente: Una adaptación de Calpa y Garzón (2022)

A continuación, se presenta un ejemplo de un caso de prueba. En la tabla 14, se muestra el formato diligenciado para un caso de prueba que se realiza en el área.

Tabla 14. Ejemplo del formato de casos de prueba

CPId	**Nombre**	**Clase**	**Método**	**RQ**	**Escenario**	**Valores de entrada**	**Resultado esperado**
1	testBuscarEstudiante1	Universidad	buscarEstudiante	RQ-1	1	Identificacion= “001”	Falso. El estudiante no existe

Para establecer el caso de prueba, según Calpa y Garzón (2022) se debe incluir el formato para el escenario de prueba. En la Tabla 15, se puede visualizar el formato. En la tabla 15 se puede observar el formato establecido para la creación de escenarios de prueba. El **RQ** corresponde con un identificador único de la historia de usuario para la cual se elabora el escenario. El **No** corresponde con un valor único que identifica el escenario de prueba de acuerdo con la historia de usuario. La **Descripción** corresponde a la descripción del escenario de prueba. Los **Datos** corresponden con valores del estado inicial de el/los objeto(s) para los cuales conforman el escenario de prueba.

Tabla 15. Formato para crear escenario de prueba

RQ:	
No:	
Descripción:	
Datos:	

Fuente: Una adaptación de Calpa y Garzón (2022)

A continuación, se presenta un ejemplo de un escenario de prueba. En la tabla 16, se muestra el formato diligenciado para un escenario de prueba que se realiza en el área.

Tabla 16. Ejemplo del formato de escenarios de prueba

RQ:	RQ-1
No:	1
Descripción:	Programa de Ingeniería de Sistemas con 3 estudiantes
Datos:	Estudiante1 = {id:"100", nombre:"Geancarlo", programa:"Ingeniería de Sistemas"} Estudiante2 = {id: "101", nombre:"Rodolfo", programa:"Ingeniería de Sistemas"} Estudiante3 = {id: "102", nombre:"Mara", programa:"Ingeniería de Sistemas"}

Para resolver una no conformidad se propone el formato que se presenta en la Tabla 17. El **NoCId** corresponde con un valor único que identifica la no conformidad. El **CPId** corresponde con un valor único que identifica el caso de prueba. El **Resultado esperado** corresponde al resultado que se debe obtener al ejecutar el caso de prueba, según el escenario al que corresponda. El **Resultado obtenido** corresponde al resultado que se obtuvo al ejecutar el caso de prueba, según el escenario al que corresponda. Las **Observaciones** corresponden a la evaluación del proceso que se obtuvo y documentarlo como evidencia.

Tabla 17. Formato para el registro de una no conformidad

NoCId:	
CPId:	
Resultado esperado:	
Resultado obtenido:	
Observaciones:	

A continuación, se presenta un ejemplo de un registro de una no conformidad. En la tabla 18, se muestra el formato diligenciado para un escenario de prueba que se realiza en el área.

Tabla 18. Ejemplo del formato de un registro de una no conformidad

NoCId:	NC-1
CPId:	1
Resultado esperado:	Falso. El estudiante no existe.
Resultado obtenido:	Verdadero. El estudiante está registrado.
Observaciones:	De acuerdo con el escenario de prueba se esperaba que el estudiante no se encuentre en la búsqueda, no obstante, al realizarla, se obtuvo como respuesta un valor verdadero en la búsqueda.

Por último se procede a elaborar el formato para la actividad **ACT-7 Registrar la resolución de la no conformidad.** En la Tabla 19 se puede observar el formato establecido para el registro de la no conformidad. El **RNoCId** corresponde con un valor único que identifica la resolución de la no conformidad. El **NoCId** corresponde con un valor único que identifica la no conformidad. La **Prioridad** se refiere a dar preferencia a la no conformidad dependiendo del riesgo las cuales pueden ser: [1][2][3]. El **Riesgo** corresponde al nivel de daño de la no conformidad entre las cuales se encuentran: [Alto][Medio][Bajo]. El **Usuario** corresponde con el nombre y rol de la persona que identificó y dio una resolución a la no conformidad. La **Fecha de realización** corresponde con el día, mes y año de cuando se dio la resolución a la no conformidad. El **Resultado obtenido** corresponde al resultado que se obtuvo al ejecutar el caso de prueba después de su resolución.

Tabla 19. Formato para registrar la resolución de la no conformidad

RNoCId:	
NoCId	
Prioridad:	[1][2][3]
Riesgo:	[Alto][Medio][Bajo]
Usuario:	
Fecha de realización:	
Acción realizada:	
Resultado obtenido:	

A continuación, se presenta un ejemplo de un registro de la resolución de la no conformidad. En la Tabla 20, se muestra el formato diligenciado para un registro de la resolución de la no conformidad en el área.

Tabla 20. Ejemplo de la resolución de la no conformidad

RNoCId	RNC-1
NoCId	NC-1
Prioridad:	2
Riesgo:	Medio
Usuario:	Geancarlo Díaz Marín – Analista de pruebas
Fecha de realización:	22-2-2024
Acción realizada	Se inspeccionó el método que realiza la búsqueda y se identificó que había un error en el recorrido parcial, donde la expresión para la búsqueda hacía que la variable bandera se devolviera siempre en verdadero. Por lo tanto, se procedió a modificar el recorrido y cambiar la expresión de comparación en la búsqueda.
Resultado obtenido:	Falso. El estudiante no existe.

2.3.6. Síntesis

Los integrantes del equipo que desarrollan pruebas en el Área de Tecnología de la Universidad Mariana destacan la eficacia del proceso propuesto por su enfoque en la automatización de pruebas de software y el registro de no conformidades. Sin embargo, identifican obstáculos como la necesidad de disponer de más herramientas, los roles ambiguos en el modelo y la carencia de personal con habilidades específicas de Analista de Pruebas. Aunque muestran un nivel de aceptación muy alto y alto con un 50%, proponen mejoras como la inclusión de herramientas adicionales, la integración del proceso de informes en el rol de Desarrollador, y la consideración de limitaciones técnicas en ciertas aplicaciones proveedoras. Además, señalan diferencias entre los desarrollos internos y el modelo propuesto, subrayando la falta de personal con el perfil de Analista de Pruebas. Sugieren la incorporación de plantillas específicas para el desarrollo de las pruebas.

3. CONCLUSIONES

Al caracterizar la forma como se realizan las pruebas de unidad y componentes en el Área de Tecnología de la Universidad Mariana, se logra identificar que, las pruebas desarrolladas son: pruebas unitarias, pruebas de componentes, pruebas de aceptación y pruebas de Interfaces Gráficas de Usuario (GUI). Además, existen dos caminos para el desarrollo de pruebas: uno para cambios o mejoras en el software y otro para identificar y solucionar problemas o errores en los productos de software existentes. Los roles encargados de realizar las pruebas son: Analista de Sistemas, Directora del Área de Tecnología y Soporte Técnico. De igual manera, los integrantes utilizan herramientas para gestionar las pruebas de software, incluyendo Mantis para el seguimiento de errores y problemas, Postman para probar APIs web, Apache JMeter para analizar y medir el rendimiento de aplicaciones web y Excel para documentar información, errores y problemas.

Se logra diseñar un proceso para el desarrollo y ejecución de pruebas automáticas de unidad desde el desarrollo ágil de software, que se representa a través de la notación BPMN. Las actividades del proceso incluyen identificación del tipo de prueba, creación y ejecución de casos de prueba, registro y notificación de no conformidades. Además, los flujos de control aseguran la identificación y registro detallado de no conformidades. Los roles propuestos en el proceso son: Analista de sistemas, herramienta de gestión de no conformidades, desarrollador, software de la mesa de ayuda del proveedor.

Al validar el proceso propuesto para el desarrollo y ejecución de pruebas automáticas de unidad con los integrantes del equipo que pruebas del Área de Tecnología de la Universidad Mariana, ellos destacan su eficacia por proporcionar un enfoque en la automatización de pruebas de software y el registro de no conformidades. Además, proponen mejoras como la inclusión de herramientas adicionales, la integración del proceso de informes en el rol de Desarrollador, y la consideración de limitaciones técnicas en ciertas aplicaciones proveedoras. Además, subrayan la falta de personal con el perfil de Analista de Pruebas. Sugieren la incorporación de plantillas específicas para el desarrollo de algunas actividades del proceso.

4. RECOMENDACIONES

Debido a que, se logró validar el proceso propuesto para el desarrollo y ejecución de pruebas automáticas de unidad con los integrantes del equipo que pruebas del Área de Tecnología de la Universidad Mariana, se sugiere que se desarrolle un proyecto donde se aborde el proceso de validación con desarrolladores que hagan uso de métodos ágiles con el fin de enriquecerlo y perfeccionarlo. Además, se podría explorar la posibilidad de realizar evaluaciones periódicas o seguimientos continuos del proceso de pruebas, utilizando indicadores de desempeño o resultados obtenidos a partir de la aplicación de las pruebas. Esto permitiría no solo medir el rendimiento del proceso, sino también identificar oportunidades de optimización, ajustes necesarios en las herramientas utilizadas o en las actividades desarrolladas, y evaluar el impacto real de las mejoras implementadas a lo largo del tiempo.

Debido a que, con este estudio, se logra diseñar un proceso para el desarrollo y ejecución de pruebas automáticas de unidad desde el desarrollo ágil de software, que se representa a través de la notación BPMN, se recomienda a la dirección del Área de Tecnología de la Universidad Mariana iniciar con la elaboración de un plan para la implementación incremental del proceso propuesto. Además, se sugiere continuar con una nueva idea de investigación que aborde otro tipo de pruebas susceptibles de ser automatizadas, como las de Interfaz Gráfica de Usuario que complementa el proceso propuesto.

Como la caracterización de la forma como se realizan las pruebas de unidad y componentes en el Área de Tecnología de la Universidad Mariana, se realizó utilizando como técnica fundamentalmente la encuesta y complementariamente usando preguntas abiertas; se recomienda en futuros proyectos relacionados con este tipo de estudios, realizar de manera complementaria la entrevista semiestructurada con el fin de precisar aspectos propios de la representación de los procesos.

REFERENCIAS BIBLIOGRAFICAS

Araya Solís, G., Méndez Marín, G., & Jiménez Segura, R. (2014). Pruebas de software para dispositivos móviles android. *Congreso de Computación Para El Desarrollo, VII*. http://repositorio.unan.edu.ni/id/eprint/2372

Boiko, Bob (2001) *The Content Management Bible.* New York: Wiley. ISBN 076454862X

Boehm, B., & Turner, R. (2003). People factors in software management: lessons from comparing agile and plan-driven methods. *CrossTalk: The Journal of Defense Software Engineering*, *16*(12), 4–8. http://sunset.usc.edu/csse/TECHRPTS/2003/usccse2003-517/usccse2003-517.pdf

Briol P. (2008). *BPMN, the Business Process Modeling Notation Pocket Handbook*. LuLu. ISBN 978-1-4092-0299-8.

Cohn, M. (2009). *Succeeding with Agile: Software Development Using Scrum.* Addison-Wesley Professional.

Calpa, A., & Garzón, J (2022). De las pruebas manuales a las pruebas automáticas en el desarrollo ágil de software: caso de estudio Universidad de Nariño.

Crispin, L. Gregory, J. (2009). *Agile Testing: A Practical Guide for Testers and Agile Teams.* Addison-Wesley.

Cvetkovic, N., Morača, S., Jovanović, M., Medojević, M., & Lalić, B. (2017). Enhancing the agility and performances of a project with lean manufacturing practices. *Annals of DAAAM and Proceedings of the International DAAAM Symposium*, 661–670. https://doi.org/10.2507/28th.daaam.proceedings.093

Debevoise, Neilson T, et. (2008). *The MicroGuide to Process Modeling in BPMN.* BookSurge Publishing. ISBN 978-1-4196-9310-6.

Faulring, A., Myers, B. A., Oren, Y., & Rotenberg, K. (2012). A case study of using HCI methods to improve tools for programmers. *2012 5th International Workshop on Co-Operative and Human Aspects of Software Engineering, CHASE 2012 - Proceedings*, 37–39. https://doi.org/10.1109/CHASE.2012.6223018

Guerrero-Calvache, M., & Hernández, G. (2023). Un estudio exploratorio de las percepciones de productividad en equipos de software ágil. *TecnoLógicas*, *26*(56), 1-17.

Jahanian, F., & Mok, A. K. L. (1986). Safety Analysis of Timing Properties in Real-Time Systems. *IEEE Transactions on Software Engineering*, *SE-12*(9), 890–904. https://doi.org/10.1109/TSE.1986.6313045

Kettunen, V., Kasurinen, J., Taipale, O., & Smolander, K. (2010). A study on agility and testing processes in software organizations. *ISSTA'10 - Proceedings of the 2010 International Symposium on Software Testing and Analysis*, *May 2016*, 231–240. https://doi.org/10.1145/1831708.1831737

Klammer, C., & Ramler, R. (2017). A Journey from Manual Testing to Automated Test Generation in an Industry Project. *Proceedings - 2017 IEEE International Conference on Software Quality, Reliability and Security Companion, QRS-C 2017*, 591–592. https://doi.org/10.1109/QRS-C.2017.108

Kitchenham, B.A., Pfleeger, S.L. (2008). Personal Opinion Surveys. In: Shull, F., Singer, J., Sjøberg, D.I.K. (eds) Guide to Advanced Empirical Software Engineering. Springer.

Kaminski, G., Ammann, P., & Offutt, J. (2013). Improving logic-based testing. *Journal of Systems and Software*, *86*(8), 2002–2012. https://doi.org/10.1016/j.jss.2012.08.024

Larman, C. (n.d.). *Agile Processes and Modeling*. 2591.

Leveson, N. G., & Stolzy, J. L. (1987). Safety Analysis Using Petri Nets. *IEEE Transactions on Software Engineering*, *SE-13*(3), 386–397. https://doi.org/10.1109/TSE.1987.233170

Maqbool, B., Ur Rehman, F., Abbas, M., & Rehman, S. (2018). Implementation of software testing practices in Pakistan's software industry. *ACM International Conference Proceeding Series*, 147–152. https://doi.org/10.1145/3180374.3181340

Meszaros, G. (2007). *xUnit Test Patterns: Refactoring Test Code.* Pearson Education.

Sommerville, I. (2011). *Ingeniería de Software.* Addison-Wesley Professional.

Schwaber, K. (2004). *Agile Project Management with Scrum.* Microsoft Press.

Storey, M. A. D., & Muller, H. A. (1995). Manipulating and documenting software structures

using SHriMP views. *Conference on Software Maintenance*, 275–285. https://doi.org/10.1109/icsm.1995.526549

White, S. A., & Miers, D. (2008). BPMN. *Modeling and Reference Guide. Understanding and using BPMN. Develop rigorous yet understandable graphical representations of business processes. Future Strategies Inc., Lighthouse Point, Fla*.

ANEXOS

Anexo 1. Entrevista al jefe del área de tecnología de la Universidad Mariana

Según el director del departamento de tecnología de la universidad mariana (2021) manifiesta que para realizar las pruebas de software tienen configurados 3 ambientes, las pruebas las ejecutan en el último ambiente descrito, de acuerdo a la información suministrada se puede ver que la aplican de forma manual, a continuación, la transcripción de la entrevista entre el estudiante y el jefe a cargo del área, Andrés Muñoz Guzmán:

Estudiante: ¿Qué tipos de pruebas realizan?

Jefe: Nosotros realizamos pruebas en nuestro ambiente de pruebas, pero a diferencia del sistema antiguo, es que en este tenemos un ambiente de pruebas, un ambiente de desarrollo y un ambiente de producción. En el ambiente de pruebas hacemos todo lo que nosotros necesitamos o nos vaya surgiendo, entonces tenemos clonada la base de datos con el sistema y vamos haciendo las pruebas ahí, por ejemplo, el sistema de grados entonces probamos, si la cosa funciona bien se va para producción; nomina electrónica también en producción. Pasa siempre y cuando haya pasado todo el proceso, entonces se paramétriza, se hace todo normal en el ambiente clonar y listo, y en cuanto a equipos y a hardware igual, los proveedores a veces nos envían partes y nos envían equipos; probamos, dimensionamos, miramos, verificamos que los ingenieros cumplan las cosas que piden, eso es más o menos el punto, las cosas que hacemos tanto en software como en hardware, en software igual siempre pedimos demostraciones cuando se compra software y ese tipo de cosas pero en los software que tenemos nosotros tenemos esos ambientes y ahí vamos haciendo nuestras pruebas.

Estudiante: ¿Cómo están realizando las pruebas a nuevo software?

Jefe: Las pruebas a nuevo software es dependiendo de las funcionalidades, los software tienen muchas cosas que nosotros no las hemos utilizado, a medida que se nos van dando los casos o los procedimientos vamos haciendo pruebas, pedimos capacitaciones, pedimos asesorías o pedimos acompañamiento con las empresas que vendieron los software y empezamos a trabajar, entonces nos dicen: necesito que este clonada la base de datos, por ejemplo para hacer ceremonia de grados se clona cuando ya solicitaron grados y empezamos a verificar diplomas, entonces todo eso más o menos se lo realiza de esa forma y cuando hay funcionalidades nuevas vamos al

ambiente de desarrollo y empezamos a hacer desarrollos y vamos probando los sub desarrollos de acuerdo a las nuevas funcionalidades que nosotros mismos pidamos, o ellos hacen funcionalidades nuevas que alguna otra universidad les pidió, los aplican y ven si funcionan en ambiente de universidad, no de la Mariana específicamente porque el software es un estándar, no es propio de la universidad, entonces si funciona para todos lo prueban y nos mandan autorizaciones para nosotros subirlas y ya.

Estudiante: ¿Se les ha presentado dificultades realizando esas pruebas?

Jefe: Si, a veces cuando no tenemos clonado los ambientes entonces perdemos tiempo esperando la clonación, nosotros tenemos ventanas de mantenimiento, y en esas ventanas están clonadas las bases de datos pero a veces cuando reportamos casos nos piden clonación, pero la clonación la hacemos periódicamente no siempre, entonces nos demoramos haciendo clonación, esa es una dificultad, tenemos que esperar que se clone; no se demora mucho el procedimiento pero generalmente lo hacemos en la noche las clonaciones para poder trabajar al otro día, otras cosas es que a veces empezamos a hacer pruebas y nos fallan, entonces ahí perdemos tiempo, nos demoramos, se pide asesorías, se busca con el proveedor, nos acompañan y vamos encontrando errores y vamos haciendo ese tipo de cosas, pero hay otra y es que a veces lo asesores no siempre están disponibles y tenemos que esperar como el turno en la mesa de ayuda, en los tickets, para que nos puedan asignar un asesor y poder continuar una estrategia o hacer la revisión de los fallos que tenemos en las pruebas.

Estudiante: ¿Qué herramientas utilizan para automatizar las pruebas?

Jefe: En algunos casos usamos migraciones, plantillas de migración que nos envía un proveedor para llenar datos en bases de datos en archivos planos, en Excel, pero el resto es manual, entonces no utilizamos herramientas como tal sino que subimos los datos y ya a través de una Excel o un archivo plano empezamos a modificar esos datos, pero la modificación de esos datos si las hacemos de manera manual, ya lo que salga de allí son las herramientas que utilizamos, los PDF si quedan bien, si no quedan bien, si se están mandando las cosas al drive para hacer las copias de seguridad, y ese tipo de cosas pues, son como herramientas que utilizaríamos, pero para hacer pruebas así, siempre son pruebas manuales para verificar el correcto funcionamiento, lo único que hacemos automático es generar una plantilla de migración hacer los datos.

Estudiante: ¿En caso de que les ofrezcan una posibilidad de hacerlo automático, aceptarían?

Jefe: Claro, lo que pasa es que las pruebas a veces son con datos, por eso nos piden clonar y empezamos a modificar esos datos, entonces necesariamente tiene que estar el usuario, el operador, modificando las cosas, no hay como utilizar una herramienta, creo que para este caso no, porque son procesos manuales los que se hacen, si hubiera por ejemplo, hacíamos pruebas en infraestructuras de saturación, ese tipo de cosas, como cuando hacemos aplicaciones y el software responde, cuando se usa el software de votaciones, que pasa si entran 100, 200, 500 usuarios, eso se puede hacer automáticamente, pero ese tipo de pruebas generalmente son manuales.

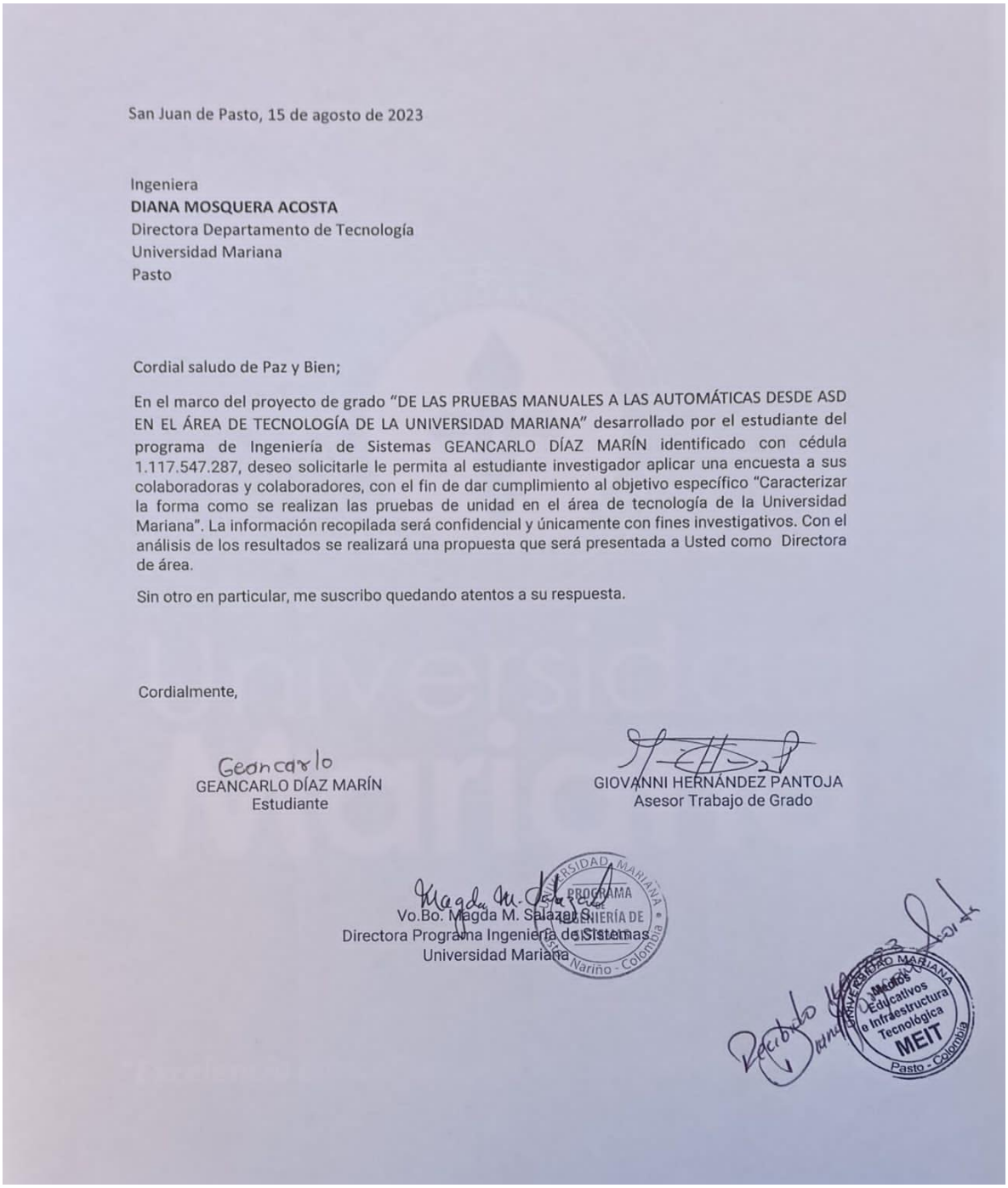
Estudiante: ¿Documentan las pruebas que realizan?

Jefe: Si, todo se documenta porque igual así como funcionan pruebas, tienen que funcionar en producción, entonces ya nos queda una guía, como prueba se va haciendo correcciones, entonces ya nos queda la guía que va para producción y es la guía que entregamos a los usuarios funcionales o es la guía de parametrización para nosotros mismos, entonces eso lo vamos documentando, para que la próxima ves que no toque repetir ya tenemos el manual y se vuelve a hacer lo mismo, y una vez funcione pues solamente lo repetimos.

Estudiante: ¿Documentan todo, hasta los errores y demás?

Jefe: Claro, de hecho, los errores son los que más se documentan, porque muchos de nuestros usuarios cometen el mismo error siempre, entonces se busca la solución, y como ya tenemos documentado el error vamos directamente a la solución, que son errores en parametrización, falta un check por ahí, ese tipo de cosas, entonces los errores son lo que más se documenta.

Anexo 2. Permiso para realizar la encuesta al área de tecnología de la Universidad Mariana



San Juan de Pasto, 15 de agosto de 2023

Ingeniera
DIANA MOSQUERA ACOSTA
Directora Departamento de Tecnología
Universidad Mariana
Pasto

Cordial saludo de Paz y Bien;

En el marco del proyecto de grado "DE LAS PRUEBAS MANUALES A LAS AUTOMÁTICAS DESDE ASD EN EL ÁREA DE TECNOLOGÍA DE LA UNIVERSIDAD MARIANA" desarrollado por el estudiante del programa de Ingeniería de Sistemas GEANCARLO DÍAZ MARÍN identificado con cédula 1.117.547.287, deseo solicitarle le permita al estudiante investigador aplicar una encuesta a sus colaboradoras y colaboradores, con el fin de dar cumplimiento al objetivo específico "Caracterizar la forma como se realizan las pruebas de unidad en el área de tecnología de la Universidad Mariana". La información recopilada será confidencial y únicamente con fines investigativos. Con el análisis de los resultados se realizará una propuesta que será presentada a Usted como Directora de área.

Sin otro en particular, me suscribo quedando atentos a su respuesta.

Cordialmente,

Geancarlo
GEANCARLO DÍAZ MARÍN
Estudiante

GIOVANNI HERNÁNDEZ PANTOJA
Asesor Trabajo de Grado

Vo.Bo. Magda M. Salazar S.
Directora Programa Ingeniería de Sistemas
Universidad Mariana

UNIVERSIDAD MARIANA PROGRAMA INGENIERÍA DE SISTEMAS Nariño - Colombia

UNIVERSIDAD MARIANA Medios Educativos e Infraestructura Tecnológica MEIT Pasto - Colombia