Software para generar reportes de pruebas automatizadas usando RPA en aplicaciones web

Fernando R. ARÉVALO
Escuela de Ingeniería de Sistemas y Computación, Universidad Peruana de Ciencias Aplicadas
Lima, Perú

Manuel A. ALVARADO
Escuela de Ingeniería de Sistemas y Computación, Universidad Peruana de Ciencias Aplicadas
Lima, Perú

Alfredo BARRIENTOS
Escuela de Ingeniería de Sistemas y Computación, Universidad Peruana de Ciencias Aplicadas
Lima, Perú

RESUMEN

Muchas empresas han adoptado la metodología Agile para la implementación de proyectos de software, donde se utiliza un enfoque iterativo e incremental. Esto introduce cambios para el ciclo de vida de las pruebas de software, ya que las funcionalidades desarrolladas se tienen que validar en cada iteración. De esta manera, para evitar las pruebas manuales y el error humano surge la automatización de pruebas como técnica para la ejecución de pruebas funcionales. Sin embargo, automatizar las pruebas y analizar sus resultados en entornos ágiles puede resultar un proceso complejo, porque se incrementan los requisitos constantemente. Debido a ello, este estudio se centra en el desarrollo de un software para la generación de reportes de pruebas automatizadas, donde se presenta los indicadores clave de rendimiento que permiten medir la eficiencia y eficacia del proceso de pruebas. Asimismo, se utilizó la tecnología de RPA para la fase de automatización y ejecución de pruebas funcionales en aplicaciones web. Por último, se validó el nivel de satisfacción del software desarrollado con un grupo de usuarios, obteniendo un 46.72% de mejora en la percepción del usuario sobre la dificultad de realizar las tareas con la solución propuesta comparado con el contexto actual en sus centros laborales.

Palabras Claves: Agile, Key Driven Testing, Robotic Process Automation, Pruebas Automatizadas

1. INTRODUCCIÓN

La etapa de pruebas de software es un aspecto importante del ciclo de vida del desarrollo del software, ya que valida su funcionamiento y características para comprobar si cumple los estándares de calidad. El reporte y análisis de las pruebas se considera una parte esencial en este proceso, ya que ayuda a una organización a tomar mejores decisiones a futuro. Con el apoyo de los reportes los desarrolladores, testers y administradores de proyecto pueden medir y analizar la calidad de la definición, planificación, automatización y ejecución de las pruebas. De esta manera pueden identificar el origen del problema y en qué etapa surgió [1].

En la actualidad, muchas empresas están adoptando metodologías ágiles para el proceso de desarrollo del software, ya que permite mejorar la productividad y reducir los tiempos de desarrollo y lanzamiento del software. Estos marcos ágiles poseen como característica en común la entrega continua y el incremento de los requisitos en cada iteración [2]. En consecuencia, se introducen nuevos retos para la fase de pruebas en los entornos ágiles, ya que el ciclo de pruebas se debe repetir para cada incremento de producto de tal forma que se valide la funcionalidad desarrollada en cada Sprint. Este proceso iterativo aumenta la complejidad de las pruebas y reduce la cobertura de requisitos [3]. Además, se debe mantener una buena trazabilidad entre historias de usuario y casos de prueba. Según una encuesta realizada por [4], los testers consideran a las historias de usuario como la principal fuente de información para el proceso de pruebas. De esta manera, la falta de trazabilidad de requisitos y la complejidad de las pruebas dificulta la generación de reporte y el análisis de los resultados de las pruebas, ya que se ve afectado la precisión de las métricas obtenidas en los informes.

La presente investigación tiene como objetivo el desarrollo de un software para generar reportes de pruebas automatizadas, con el cual se busca facilitar la generación de reportes en marcos ágiles y reducir la complejidad en la ejecución de las pruebas a través del uso de una herramienta de RPA.

2. ESTADO DEL ARTE

A continuación, se presentan algunos trabajos los cuales se categorizaron en métricas para reportes de pruebas y procesos o frameworks para la fase de testing.Se realizaron investigaciones sobre el proceso y enfoques de pruebas en entornos de desarrollo ágil. Asimismo, se examinaron otros trabajos relacionados a las métricas del proceso de pruebas y desarrollo ágil.

Proceso y enfoques de las pruebas

Un estudio realizado por [5], nos define a las pruebas de software como una fase dentro del ciclo de vida del software, la cual cuenta con su propio ciclo de vida. Esta etapa de pruebas requiere mucho esfuerzo y es un proceso intensivo, por ello el autor expone diversas técnicas y metodologías para agilizar esta actividad. En primer lugar, el autor menciona que la automatización de prueba es útil para reducir tiempos en la ejecución de casos de prueba. Los principales tipos de prueba que se automatizan suelen ser pruebas de regresión, ya que estas requieren de mucho tiempo cuando se realizan manualmente. Luego, como parte de la mejora del proceso de pruebas se

ISBN: 978-1-950492-75-6
ISSN: 2831-7270

https://doi.org/10.54808/CISCI2023.01.26

tienen las métricas de las pruebas. Estas métricas sirven para medir la eficacia del proceso y aseguran que el software producirá un resultado de alta calidad. Las métricas que se utilizan son: La cobertura de prueba, la eficiencia de corrección de defectos, el índice de volatilidad de requisitos, la cantidad de casos de prueba fallidos, ejecutados y superados. Por otra parte, se tiene la Matriz de Trazabilidad de Requisitos (RTM), la cual permite un proceso de prueba mejorado al asignar cada caso de prueba con un requisito específico.

Los autores en [6] presentan dos enfoques para la fase de pruebas que se lleva a cabo en el proceso de Scrum y entornos ágiles. El enfoque convencional es Test Last, el cual consiste en realizar las pruebas al final de cada Sprint. Por otra parte, se tiene el enfoque Test Driven Development (TDD) o Test First, que es una práctica ágil en la que los casos de prueba se crean antes de codificar. En este estudio se compara la calidad del código y la productividad de ambos enfoques. Para la validación, realizaron un experimento formal entre estudiantes de pregrado sin experiencia como programadores y testers. El proceso de desarrollo utilizado fue Scrum. A partir de los resultados, los autores concluyeron que el enfoque de Test First presenta ventajas como la reducción en el tiempo dedicado al desarrollo, sin embargo; los desarrolladores y testers están más familiarizados con la práctica de Test Last.

En [7] los autores describen la importancia de las pruebas de regresión en la adopción de metodologías de desarrollo ágil. Este tipo de pruebas impacta en el costo total del desarrollo y el mantenimiento del software. En este artículo se propuso un enfoque para automatización de pruebas de regresión basado en la Técnica de Priorización de Casos de Prueba de Sprint Ponderado (WSTP) y Técnica de Selección de Casos de Prueba de Lanzamiento Basada en Clústeres (CRTS). Como resultados obtuvieron que su enfoque mejora las capacidades de detección de fallas en un 78%. Luego, señalan la importancia de priorizar y seleccionar los casos de prueba en proyectos ágiles y que a pesar de que existen diversos estudios sobre pruebas de regresión no se consideran muy a menudo el uso de parámetros definidos para proyectos ágiles. Además, mencionan que se debe considerar el nivel de severidad y prioridad de los problemas encontrados para brindar una priorización y selección más precisa.

Métricas de las pruebas

En [8], los autores presentan una métrica para medir la calidad del proceso de prueba en un marco de desarrollo ágil de Scrum. Además, se plantean aspectos de calidad de requisito como, ambigüedad de requisitos, completitud, índice de madurez. Por otro lado, para evaluar el éxito del proceso de prueba en términos de defectos se tienen en cuenta, aspectos como defectos encontrados, errores omitidos por casos de prueba, corrección de errores. Finalmente, mediante el promedio ponderado de los elementos mencionados anteriormente se obtiene la métrica que nos permite medir la calidad del proceso de prueba en Scrum.

Las pruebas de software son un proceso continuo durante las etapas de desarrollo de software para garantizar productos de software de calidad. Es por ello por lo que [9] propone un enfoque para pruebas basadas en contexto (CDT). Este enfoque ayuda al tester a seleccionar y priorizar sus casos de prueba en función a contextos específicos. Primero, se definen las historias de usuario con sus criterios de aceptación, luego se realiza el análisis del contexto para priorizar los escenarios dependiendo de su prioridad. Finalmente, se reportan los resultados con datos como número de defectos, cantidad de defectos solucionados, calidad, costo y tiempo. Los resultados obtenidos durante la comparación entre dos sprints fueron que la identificación de defectos fue mucho más rápida en la cual se encontraron un 16% más en cantidad de defectos y además se obtuvo una cobertura del 89%.

Otras métricas utilizadas para llevar un seguimiento y control dentro del proceso de desarrollo de software ágil se enfocan en la corrección de defectos, elaboración de casos de pruebas y sus estados. En [10] se plantea un conjunto de métricas para cobertura de defectos, tiempo promedio de ejecución de las pruebas, porcentaje de pruebas sin ejecutar, entre otras. Para la validación de estas métricas, se realizó una prueba piloto entre dos equipos de trabajo en la cual los resultados obtenidos fueron que las métricas más útiles estaban enfocadas en la identificación de defectos y tiempo promedio de corrección de defectos y además mediante las métricas aplicadas lograron una mejora en tiempos de corrección de defectos y esfuerzo planificado.

3. DESARROLLO

Análisis de indicadores clave de rendimiento de las pruebas

A partir de la investigación realizada se definieron las métricas que permitieran a los usuarios realizar una correcta gestión y evitar consecuencias en costos debido a demoras en tiempos de entrega, estimaciones no precisas, cambios a última hora y falta de capacitación por parte de los desarrolladores [11]. En [12] los autores presentan un conjunto de métricas e indicadores claves de rendimiento para los proyectos de prueba de software, de esta manera poder controlar el estado de los objetivos estratégicos y metas fijadas por la organización. A partir de la revisión de la literatura, se ha seleccionado una lista de KPI's y se clasificaron en tres categorías como se muestra en la Tabla 1.

Tabla 1. Listado de KPI'S por categoria

N°	Kpi's by category	
	Category	***Metric***
1	Coverage	Test Design Coverage
2		Test Execution Coverage
3		Requirement Coverage
4	Test Tracking	Test Cases Percentage by Status
5		Test Runs per Period
6		Test Cases Percentage by Priority
7	Defects Tracking	Defects Find Rate Percentage
8		Defects Percentage by Status
9		Defects Percentage by Priority

Los indicadores de cobertura permiten conocer cuántos de los requisitos han sido desarrollados por los testers. De esta manera, un informe de cobertura puede ahorrarles tiempo a los líderes de proyecto al mostrar rápidamente la información que necesita, en lugar de ponerse a revisar uno por uno los requisitos que fueron cubiertos con la ejecución de las pruebas [13]. En la Tabla 2 se puede apreciar las fórmulas empleadas para cada indicador dentro de esta categoría.

Tabla 2. KPI'S para medidas de cobertura

N°	Coverage Metrics	
	Kpi	***Formula***
1	Requirement Coverage	(*Total number of covered requirements* / *Total number of requirements*) × 100%
2	Test Execution Coverage	(*Total number of executed tests* / *Total number of tests*) × 100%
3	Test Design Coverage	(*Total number of tests mapped to requirements* / *Total number of tests*) × 100%

Las métricas de seguimiento de pruebas permiten mostrar el estado de las ejecuciones, de esta forma se puede determinar la productividad de las pruebas realizadas y cuál es el esfuerzo real invertido por los testers. A partir de los resultados obtenidos se pueden realizar mejores estimaciones en futuros proyectos de una organización. En la Tabla 3 se expone cada uno de estos indicadores.

Tabla 3. KPI'S para el seguimiento de pruebas

N°	Test Tracking Metrics	
	Kpi	***Formula***
1	Test Cases Percentage by Status	(*Total number of passed, failed or skipped tests* / *Total number of tests*) × 100%
2	Test Runs per Period	*Total number of executed tests* / *Total time*
3	Test Cases Percentage by Priority	(*Total number of high, medium or low* priority *tests* / *Total number of tests*) × 100%

Por último, los indicadores de seguimiento de defectos permiten determinar la efectividad de los casos de prueba ejecutados. También, posibilitan la medición de la calidad en las correcciones de defectos y el trabajo realizado por el equipo de desarrollo. Estas métricas sirven para que los analistas y líderes de proyecto puedan analizar la estabilidad en la construcción de un software y tomar mejores decisiones antes del lanzamiento de un producto de software. En la Tabla 4 se presentan la lista de métricas definidas.

Tabla 4. KPI'S para el seguimiento de defectos

N°	Defects Tracking Metrics	
	Kpi	***Formula***
1	Defects Find Rate Percentage	(*Total number of defects found* / *Total number of executed tests*) × 100%
2	Defects Percentage by Status	(*Total number of new, accepted, fixed or rejected defects* / *Total number of defects*) × 100%
3	Defects Percentage by Severity	(*Total number of* critical, normal *or* trivial defects / *Total number of* defects) × 100%

Análisis de herramienta de RPA

Dentro de entornos ágiles de desarrollo, se requieren realizar pruebas de regresión de manera manual y repetirlas durante cada iteración lo cual conlleva un gran esfuerzo. Ante esto, se plantea la tecnología Robotic Process Automation (RPA), la cual nos permite la creación de robots de software para la automatización de las pruebas como lo plantean en [14], obteniendo resultados favorables en cuanto a tiempo en elaboración y ejecución de las pruebas, y además en costos por presupuestos. Por ello, para la presente investigación se ha realizado un benchmarking de selección para la herramienta RPA. Además, se definieron criterios de selección como se visualiza en la Tabla 5.

Tabla 5. Criterios de selección para la herramienta RPA

N°	Selection Criteria	
	Criteria	***Weight***
1	Features and functionalities	20%
2	Community Support	10%
3	Price	25%
4	Integration with other tools	10%
5	Relevant data for KPI's	35%

Estos criterios nos permitieron enfocarnos en la característica más importante la cual fue la información relevante para los KPI's, además los criterios fueron propuestos bajo la característica clave de relevancia como se define en [15]. De esta manera, seleccionar la herramienta más idónea que brinde los datos requeridos para el cálculo de los KPI's. Para el benchmarking realizado se compararon las herramientas de RPA, UIPath, Robocorp, Power Automate, RocketBot.

Tabla 6. Selección de herramienta RPA

C	W	UiPath		Robocorp		Power Automate		Rocketbot	
		P	***We***	***P***	***We***	***P***	***We***	***P***	***We***
C1	20%	9	1,80	9	1,80	7	1,40	9	1,80
C2	10%	7	0,70	7	0,70	6	0,60	7	0,70
C3	25%	4	1,00	10	2,50	4	1,00	7	1,75
C4	10%	7	0,70	7	0,70	5	0,50	0	0,00
C5	35%	6	2,10	10	3,50	6	2,10	7	2,45
Results			6,30		9,20		5,60		6,70

La herramienta seleccionada fue la de Robocorp con un puntaje ponderado de 9,6. Obtuvo un gran puntaje debido a que en los resultados de las pruebas exporta datos relevantes para el cálculo de los KPI's definidos. Por otra parte, utiliza el enfoque Keyword Driven Testing, el cual es un tipo de automatización de pruebas funcionales basado palabras clave. En comparación a otras prácticas, este enfoque es más adecuado para un tester principiante ya que no requiere conocimientos en programación. Además, las palabras tienen un alto grado de reusabilidad en los scripts de prueba. En consecuencia, las pruebas requieren menos mantenimiento a largo plazo ya que al actualizar una palabra clave todas las pruebas son actualizadas de forma automática [16].

Implementación

La aplicación desarrollada lleva por nombre Prova Report como se muestra en la Figura 1. Esta aplicación web tiene como objetivo la generación de reportes de pruebas funcionales sobre sitios web, las cuales se ejecutan a través de la herramienta Robocorp Lab.

Figura 1. Login de la aplicación

Antes de desarrollar la aplicación de Prova Report, hemos definido un flujo de procesos basado en el proceso de pruebas ISTQB. Este tiene como objetivo la obtención de datos en cada fase de las pruebas. De esta forma, al recopilar los datos obtenidos en la ejecución y en la gestión de defectos se puede calcular los KPIs definidos y mostrar la información en los reportes. A continuación, se muestra el proceso definido en la Figura 2.

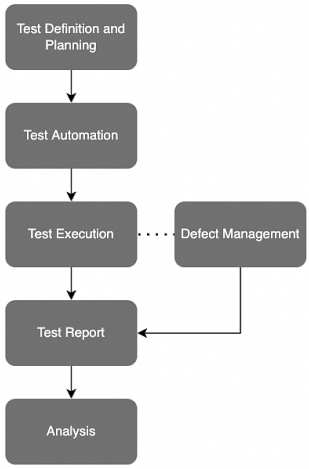

Figura 2. Proceso de generación y análisis de reportes

La fase de definición de las pruebas comienza cuando el usuario registra los suites y casos de prueba de un proyecto.

Para cada prueba debe especificar el nivel de prioridad y severidad. Esto permite poder clasificar los resultados de las pruebas según los niveles de urgencia y el impacto que tienen sobre el software bajo prueba. Por otro lado, se tienen las historias de usuario que se deben registrar con sus criterios de aceptación, cada criterio se asocia con un caso de prueba para poder mantener una trazabilidad. Tanto para historias de usuario como para los suite y casos de prueba, la aplicación permite realizar una carga masiva a través de archivos con valores separados por coma.

En la etapa de planificación, el usuario puede asignar un responsable para la ejecución de cada caso de prueba. Estos colaboradores son registrados y asignados a distintos proyectos. Cada vez que un responsable es asignado a un caso de prueba, la aplicación le notifica a través de correo electrónico que ha sido asignado a una prueba.

En la siguiente fase de automatización de las pruebas, el tester que ha sido asignado como responsable de un conjunto de pruebas debe dirigirse a la aplicación y revisar las especificaciones definidas por el administrador del proyecto. De esta manera, con ayuda de la herramienta de Robocorp se desarrollan los scripts de las pruebas asignadas. Estos scripts se codifican utilizando lenguaje Gherkin como se observa en la Figura 4.

```
*** Test Cases ***
TC-01 Listar marcas satisfactoriamente
    [Documentation] Listar marcas satisfactoriamente
    [Tags] regression
    Given Ha iniciado sesion ${URL} ${USER} ${PWD}
    When Haga clic en el enlace de marcas
    Then La aplicacion mostrara el listado de marcas
```

Figura 4. Automatización de un caso de prueba con Robocorp

En la fase de ejecución el tester debe ejecutar las pruebas y exportar los resultados obtenidos con la herramienta de Robocorp. Luego de ejecutar las pruebas el usuario carga los resultados en Prova Report. La aplicación se encarga de leer los datos del XML exportado y registrar los logs de la ejecución.

Si el tester encuentra defectos en las ejecuciones fallidas, puede reportar los errores asignándole un nivel de prioridad y severidad como se observa en la Figura 5.

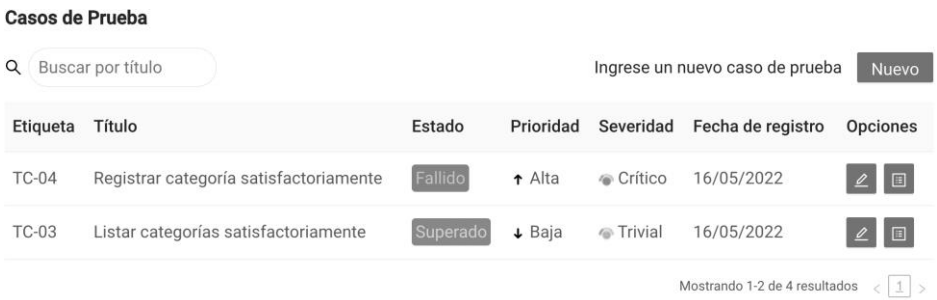

Figura 3. Listado de casos de prueba

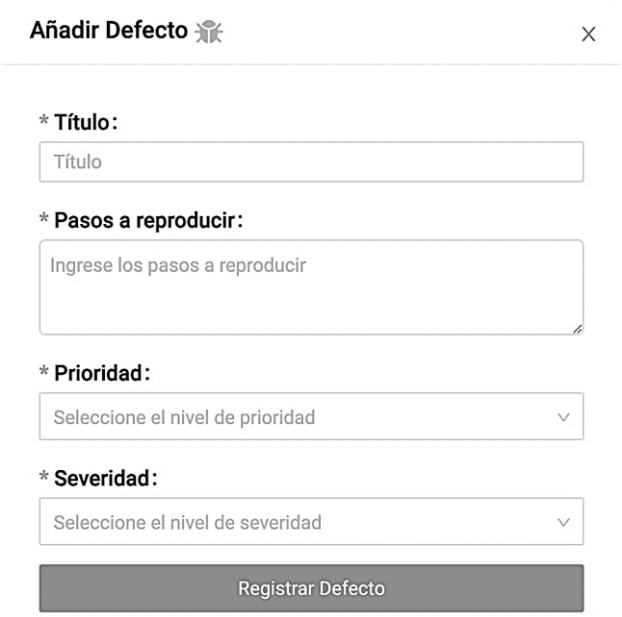

Figura 5. Registro de defectos

En la etapa de reporte de las pruebas. La aplicación muestra un dashboard con el resumen de los resultados obtenidos durante el diseño, ejecución de las pruebas y la gestión de defectos realizada. Estos resultados se pueden filtrar en un período de tiempo determinado y en un proyecto en particular.

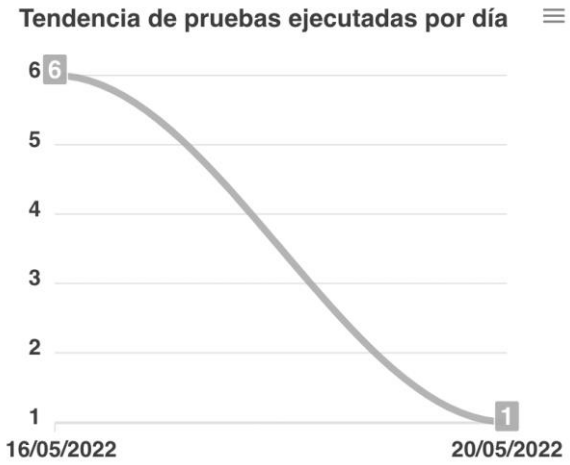

Figura 6. Tendencia de pruebas ejecutadas en un día

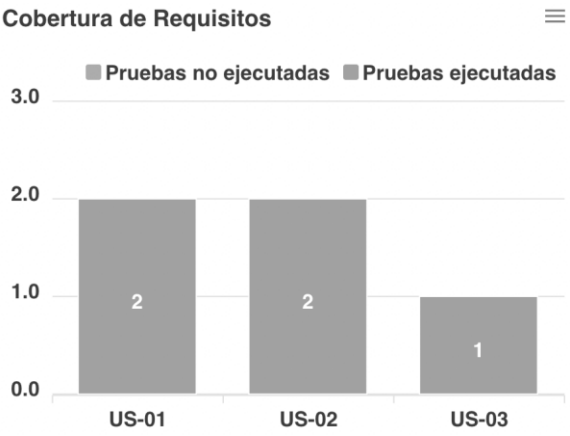

Figura 7. Cobertura de requisitos

Como se observa en la figura 6 y figura 7, tenemos la tendencia de pruebas ejecutadas por día, y la cantidad de casos de prueba ejecutados por historias de usuario.

Figura 8. Porcentaje de pruebas ejecutadas por estado

Luego, se visualizan las métricas de defectos reportados.

Figura 9. Porcentaje de defectos reportados por estado

Finalmente, para la fase de análisis la aplicación de Prova Report permite exportar un informe de cierre de las pruebas ejecutadas. En este reporte se consolidan las métricas de cobertura, así como la matriz de trazabilidad de requisitos, lo cual da un soporte a los usuarios para la toma de decisiones en su organización.

4. CASO DE ESTUDIO

Perfil del usuario

La validación se llevó a cabo con el apoyo de 30 personas, las cuales fueron egresados y estudiantes de los últimos ciclos de las carreras de Ingeniería de Software, Ingeniería de Sistemas de la Información y Ciencias de la Computación. Estos participantes desempeñaban roles como QA Testers, administradores de proyecto y desarrolladores de software y trabajan con marcos ágiles en sus organizaciones.

Implementación

Escenario de prueba: Para poder validar la solución propuesta se ha diseñado un escenario de prueba sobre una aplicación web. Se definió un conjunto de historias de usuario y casos de prueba para que cada participante realice el proceso de generación de reportes con pruebas automatizadas. Como resultado se elaboraron dos archivos con extensión “CSV”.

Automatización con RPA: Se automatizaron los scripts de las pruebas funcionales a través de la herramienta de Robocorp Lab. Luego, se ejecutaron las pruebas automatizadas en la aplicación web bajo prueba, obteniendo como resultado un archivo con extensión "XML" que contenía los datos de la ejecución.

Cuestionario: Se elaboró un cuestionario para medir el nivel de satisfacción de cada tarea del escenario de prueba basado en Single Ease Question (SEQ) tomando como referencia la validación hecha por [17]. Este cuestionario se dividió en dos secciones, una para valorizar el contexto actual de cada participante en su organización donde trabaja, y otra para valorizar las tareas realizadas con la solución propuesta. Cada pregunta utiliza una escala de Likert de siete puntos, donde "1" es muy difícil y "7" es muy fácil.

Tabla 7. Cuestionario de nivel de satisfacción de tareas

N°	Questionary	
	Task	***Context***
1	Define test cases and user stories	Current context
2	Execute tests	Current context
3	Report defects found in execution	Current context
4	Prepare report with test execution results	Current context
5	Define test cases and user stories	Proposed solution
6	Execute tests	Proposed solution
7	Report defects found in execution	Proposed solution
8	Prepare report with test execution results	Proposed solution

Inducción al usuario sobre el escenario de prueba: Se presentó a cada participante un conjunto de diapositivas para explicar la solución propuesta, la problemática y las tareas del escenario de prueba. Además, se le brindó los archivos generados en los pasos previos y el enlace a la aplicación de Prova Report para que puedan desarrollar el escenario.

Entrevista: Cada entrevista fue grabada con el consentimiento de los participantes, en esta etapa. Al final del escenario de prueba, se le compartió un cuestionario al participante para que valorice las tareas realizadas en el contexto actual de su organización y, por otra parte; utilizando la solución propuesta.

Resultados

Se compararon los resultados obtenidos sobre el nivel de satisfacción de los usuarios en ambos contextos, se han tomado los valores medios de cada valorización.

Tabla 8. Promedio de calificación de las tareas y mejora

Context	Task 1	Task 2	Task 3	Task 4	Median
Current	4.17	4.27	4.40	4.10	4.23
Proposed	5.70	5.93	6.33	6.20	6.04
% Improvement	36.80%	39.06%	43.94%	51.22%	42.76%

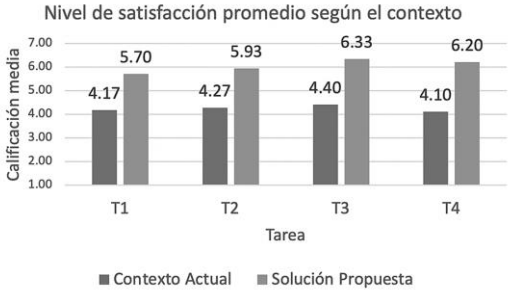

Figura 10. Nivel de satisfacción promedio de las tareas según contexto

Los resultados muestran una mejora considerable en la facilidad para realizar las tareas con la solución propuesta. De acuerdo con la tabla 8, se tuvo una mejora del 42.76% en la dificultad para la realización de tareas del proceso de generación de reportes. Según el gráfico mostrado en la Figura 10, en ambos contextos hay semejanza en la calificación de todas las tareas. Las tareas más fáciles de realizar con la solución propuesta fueron reportar los defectos encontrados (Task 3) y elaborar reporte con los resultados de las pruebas (Task 4), según las entrevistas realizadas estas acciones han sido fáciles de identificar dentro de la aplicación de Prova Report. Por otra parte, en el contexto actual los participantes consideraron como la tarea más difícil la elaboración de reportes con los resultados de ejecución de las pruebas (Task 4) y la definición de casos de prueba e historias de usuario (Task 1), según los entrevistados estas tareas pueden resultar complejas cuando se tiene un gran número de requisitos y escenarios de prueba.

5. CONCLUSIONES Y TRABAJO FUTURO

El propósito de esta investigación fue desarrollar un software para generar reportes de pruebas automatizadas usando RPA en aplicaciones web para facilitar el proceso de elaboración de reportes en entornos ágiles. El estudio fue realizado con 30 participantes que trabajan en el área de TI de distintas empresas peruanas y utilizan metodologías ágiles.

Los resultados muestran que la solución propuesta tuvo un alto grado de satisfacción sobre las tareas realizadas. Esto indica que el software desarrollado junto a la tecnología de RPA fueron más fáciles de utilizar con una mejora del 42.76% sobre la percepción de los participantes. Además, durante las entrevistas los participantes resaltaron que la aplicación de Prova Report fue muy intuitiva y resulta bastante útil para la certificación de proyectos ágiles.

Por otro lado, a pesar de que muchos participantes consideraron los KPIs como la parte más útil de la aplicación, han sugerido variar los tipos de gráfico y añadir mayor personalización al dashboard, ya que dentro de una organización existen distintas perspectivas de acuerdo al rol de los trabajadores.

Finalmente, en este estudio no se han considerado pruebas de bajo nivel como pruebas de integración y pruebas unitarias. Asimismo, no se abarcaron aplicaciones de escritorio y aplicaciones móviles para las pruebas. Estudios futuros deberían considerar el uso de distintos tipos de plataforma y tipos de prueba para investigar más sobre los KPIs y tecnologías de automatización de pruebas.

6. REFERENCIAS

[1] "Test Reporting and its significance in Continuous Testing | pCloudy," pCloudy, May 11, 2022. https://www.pcloudy.com/blogs/test-reporting-and-its-significance-in-continuous-testing/ (accessed Jun. 12, 2022).

[2] Andy Canales, Aldair Ore, Alfredo Barrientos and Miguel Cuadros. 2022. Modelo para la Adopción de Frameworks de Escalamiento Ágiles en las Empresas del Sector Bancario de Perú. CICIC (Mar. 2022), 119-123. DOI: https://doi.org/10.54808/CICIC2022.01.119

[3] Alejandra Ponce and Raul Naranjo. 2018. Implementación de un Software para la ejecución de pruebas en aplicaciones web extendiendo la herramienta Selenium. Retrieved May 26, 2022 from http://repositorio.espe.edu.ec/handle/21000/14134

[4] "The 2022 State of TestingTM Report," Practitest.com, 2022. https://www.practitest.com/state-of-testing/ (accessed Jun. 12, 2022).

[5] Arun Kumar Arumugam, "Software Testing Techniques New Trends," International Journal of Engineering Research and, vol. V8, no. 12, Jan. 2020, doi: 10.17577/ijertv8is120318.

[6] Wantana Sisomboon. 2021. Test First vs Test last: A Study of Software Quality in Action. ICIC Express Letters 15, 6 (June 2021), 545-552. DOI: https://doi.org/10.24507/icicel.15.06.545

[7] Passant Kandil, Sherin Moussa and Nagwa Badr. 2016. Cluster-based Test Cases Prioritization and Selection Technique for Agile Regression Testing. Journal of Software: Evolution and Process 29, 6 (July 2016). DOI: https://doi.org/10.1002/SMR.1794

[8] Imrul Kayes, Mithun Sarker, and Jacob Chakareski. 2016. Product backlog rating: a case study on measuring test quality in scrum. Innov. Syst. Softw. Eng. 12, 4 (December 2016), 303–317. https://doi.org/10.1007/s11334-016-0271-0

[9] Huynh Quyet, Thang & Pham, Le-Trinh & Ha, Nhu-Hang & Nguyen Duc, Man. (2020). An Effective Approach for Context Driven Testing in Practice — A Case Study. International Journal of Software Engineering and Knowledge Engineering. 30. 1245-1262. DOI: http://doi.org/10.1142/s0218194020500333

[10] M. Choraś et al., Measuring and Improving Agile Processes in a Small-Size Software Development Company (2020). IEEE Access, 8. 78452-78466, DOI: https://doi.org/10.1109/ACCESS.2020.2990117.

[11] Brian Hambling, Peter Morgan, Angelina Samaroo, Geoff Thompson and Peter Williams. July 2019. The Fundamentals of Testing. BCS Learning & Development Limited, OReilly. Retrieved May 26, 2022 from https://learning.oreilly.com/library/view/software-testing/9781780174921/

[12] Gregory Jose and Jusha Joseph. 2014. TEST METRICS AND KPI'S. Retrieved May 26, 2022, from https://docplayer.net/8701977-Test-metrics-and-kpi-s.html

[13] Marcin Żurawiecki. 2018. Why requirements test coverage is key to success? Retrieved May 26, 2022 from https://deviniti.com/blog/application-lifecycle-management/why-requirements-test-coverage-is-key-to-success/

[14] Murgueytio, F. M., Galarza, P. J., Barrientos, A. (2022). Proceso de Automatización de Pruebas de Aplicaciones Web Desarrolladas con React, Angular, Ant y Laravel. En N. Callaos, J. Horne, B. Sánchez, A. Tremante (Eds.), *Memorias de la Vigésima Primera Conferencia Iberoamericana en Sistemas, Cibernética e Informática: CISCI 2022*, pp. 192-197. International Institute of Informatics and Cybernetics. https://doi.org/10.54808/CISCI2022.01.192

[15] Jóakim v. Kistowski, Jeremy A. Arnold, Karl Huppler, Klaus-Dieter Lange, John L. Henning, and Paul Cao. 2015. How to Build a Benchmark. In Proceedings of the 6th ACM/SPEC International Conference on Performance Engineering (ICPE '15). Association for Computing Machinery, New York, NY, USA, 333–336.DOI: https://doi.org/10.1145/2668930.2688819

[16] Microfocus. 2022. Advantages of Keyword-Driven Testing. Retrieved May 26, 2022 from https://www.microfocus.com/documentation/silk-test/195/en/silktestclassic-195-help-en/GUID-43E2F89C-061B-4EE0-8A94-C949D599D56F.html

[17] R. Zahiriddin, J. Rustamov, T. Chuan, H. Kaur, A. Singh, and A. Sin, "A USABILITY EVALUATION OF TARCAPP MOBILE APPLICATION." Accessed: Jun. 11, 2022. [Online]. Available: https://www.tarc.edu.my/files/icdxa/8F6B1B05-25B7-43D9-8878-81F00E2B3BFA.pdf

[18] Morgan, P., Samaroo, A., Geoff, T., & Peter Williams. (2019). Software Testing - An ISTQB-BCS Certified Tester Foundation guide 4th edition (B. Hambling (ed.)). BCS Learning & Development Limited. https://learning.oreilly.com/library/view/software-testing/9781780174921/