

Integración y entrega continua en el proceso de pruebas a aplicaciones móviles de MCDAI

Continuous Integration and Delivery in the Testing Process for Mobile Applications in MCDAI

Yaneisy Onelia Massó Agramonte^{1*} <https://orcid.org/0000-0003-0023-0079>

Yoandy Lazo Alvarado² <https://orcid.org/0000-0002-8285-2180>

¹ DESOFT, Calle 24, No. 408 entre 23 y 25. Plaza de la revolución. La Habana, Cuba.

² Ministerio de Comunicaciones, avenida Independencia, No. 2, entre 19 de Mayo y Aranguren, Plaza de la Revolución, La Habana, Cuba

*Autor para la correspondencia. (yaneisyonelia@gmail.com)

RESUMEN

La tecnología de internet móvil ha penetrado en todos los mercados de las industrias, por lo que es muy importante las pruebas de estas aplicaciones para mejorar la calidad de los productos. Las pruebas a aplicaciones móviles son diferentes a las pruebas que se realizan a otras aplicaciones por lo que requiere un enfoque diferente. Las empresas de aplicaciones adoptan modelos y normas internacionales que contribuyen a mejorar los indicadores de rendimiento de los procesos de software. En Cuba existe el modelo de la

calidad para el desarrollo de aplicaciones informáticas. Como consecuencia se investigaron las prácticas comunes y las tendencias actuales de las pruebas a aplicaciones móviles y a partir de esto se planteó como objetivo contribuir al aumento de la calidad de los productos de la industria cubana del software. Para lograr este objetivo se analizaron las metodologías de investigación teórico y empírico para conocer la evolución y realizar un estudio de las aplicaciones móviles y para analizar los documentos relacionados con el tema. El resultado obtenido en la investigación confirma la necesidad de un enfoque adecuado para pruebas a aplicaciones móviles, destacando la importancia de la integración y entrega continua en el proceso de pruebas a aplicaciones móviles dentro del Modelo de Calidad para el Desarrollo de Aplicaciones Informáticas.

Palabras clave: aplicaciones móviles; pruebas; integración continua; entrega continua; modelo de calidad.

ABSTRACT

Mobile internet technology has penetrated all industrial markets, making testing these applications very important for improving product quality. Mobile application testing is different from testing other applications and therefore requires a different approach. Application companies adopt international models and standards that contribute to improving the performance indicators of software processes. Cuba has a quality model for the development of computer applications. As a result, common practices and current trends in mobile application testing were investigated, and based on this, the objective was to contribute to increasing the quality of products in the Cuban software industry. To achieve this objective, theoretical and empirical research methodologies were analyzed to understand the evolution and conduct a study of mobile applications and to analyze related documents. The results of the research confirm the need for a suitable approach to mobile application testing, highlighting the importance of continuous integration and delivery in the mobile application testing process within the Quality Model for the Development of Computer Applications.

Keywords: mobile applications; testing; continuous integration; continuous delivery; quality model.

Recibido: 25/01/2025

Aceptado: 24/09/2025

Publicado: 01/10/2025

Introducción

Vivimos en un mundo conectado. Cada día aumenta el número de dispositivos de todo tipo que proporcionan acceso a Internet. Las cosas u objetos que permiten y van a permitir estos accesos irán en aumento con el tiempo. Miles de millones de dispositivos están siendo conectados entre sí a través de distintas redes de comunicación. El mundo se está convirtiendo en un campo de información global, y la cantidad de datos que circula por las redes está creciendo exponencialmente (Aguilar, 2021). Estos dispositivos se han convertido en elementos esenciales constituyéndose en los principales impulsores de cambios significativos en diversas esferas sociales y personales, por tanto, los dispositivos móviles son motores clave del cambio en la sociedad actual, brindando a las personas acceso instantáneo a la información y a los medios de comunicación (Mezones & Vaca-Cárdenas, 2021).

Con la llegada del teléfono móvil, el desarrollo de aplicaciones móviles se hizo responsable de producir programas relativamente cortos que desempeñaban funciones importantes en la comunicación en línea de usuario a usuario. Hoy en día, las aplicaciones móviles juegan un papel importante en la adaptación de los usuarios a la era de la digitalización, donde gran parte de los procesos diarios se pueden realizar con un Smartphone (Puetate & Ibarra, 2020). Las aplicaciones para dispositivos móviles han alcanzado una gran importancia en la sociedad actual siendo unos de los dispositivos más utilizados. Debido a esta situación se están creando más aplicaciones móviles para la realización de múltiples funciones que ayudan a las personas en sus tareas diarias.

El desarrollo de aplicaciones móviles tiene sus propios requisitos que difieren del desarrollo de aplicaciones tradicionales, estos son: interacción potencial con otras aplicaciones, gestión de sensores, aplicaciones nativas e híbridas, gestión de la seguridad, interfaz de usuario para pantallas pequeñas y consumo de energía

(Thomas et al., 2022). El desarrollo de aplicaciones móviles debe cumplir una serie de requisitos y condiciones especiales, lo que lo convierte en un proceso más complejo. El desarrollo de aplicaciones móviles tiene algunas diferencias con el desarrollo de otros tipos de aplicaciones por lo que sus pruebas también lo son. ISTQB afirma que todo el mundo en algún momento de su vida, experimenta una falla de software, desde un error en un dispositivo móvil hasta una transacción bancaria fallida (Graham et al., 2006).

La llegada de las metodologías ágiles, la verificación de software incluye tipos y niveles de pruebas que no solo se ejecutan con el software en ejecución, sino también durante su integración y construcción (Mascheroni, 2021). Existen entre otras una práctica del desarrollo de software, donde miembros de un equipo integran su trabajo con frecuencia. Usualmente, cada persona realiza al menos una integración en el día, lo que conduce a múltiples integraciones diarias. Cada integración es verificada por un proceso de construcción automatizado (que incluye pruebas), para detectar errores de integración tan pronto como sea posible. Cada cambio introducido debe someterse a un proceso de pruebas. Así, cualquier introducción, modificación o borrado de una funcionalidad, es lanzada a producción cuando se lo desee buscando minimizar la cantidad de errores introducidos (Mitasiunas et al., 2014).

Razón por la cual son muy importante las pruebas de software para mejorar la calidad de los productos. Es importante tener en cuenta el entorno en que se realizan las pruebas a aplicaciones móviles. Para aumentar la fiabilidad de las pruebas a veces es necesario simular los procesos de la aplicación con la mayor precisión posible creando un entorno de pruebas a medida que se adapta a la aplicación. Se deben realizar pruebas en distintos sistemas operativos para que todos los usuarios obtengan la experiencia para la que se ha diseñado la aplicación.

La mejora de procesos se ha vuelto cada vez más importante a lo largo de los años, con muchas organizaciones tratando de reducir sus costos de producción, mejorando la eficiencia de sus procesos de desarrollo (Socarrás Ramírez et al., 2022). Por ello se adoptan modelos y normas internacionales que contribuyen a mejorar los indicadores de rendimiento de los procesos de software. Cuba no es ajena al desarrollo de las Tecnologías de la Información y las Comunicaciones, la Industria Cubana del Software trabaja en la búsqueda de nuevas alternativas y caminos con el objetivo de generar mayores ingresos para contribuir al desarrollo de la economía. En Cuba existe el modelo de la calidad para el desarrollo de

aplicaciones informáticas (MCDAI). La producción de aplicaciones para dispositivos móviles se ha incrementado exponencialmente en la industria del software en Cuba incluyendo empresas estatales y privadas. Las empresas productoras de software deben tener siempre presente la calidad de los productos y la satisfacción de los clientes. Son elementos esenciales para tener resultados satisfactorios en el panorama actual debido a la gran competencia.

Apoyándose en los métodos empíricos se realizó un diagnóstico a personas involucradas en el proceso de pruebas que permitió realizar una caracterización del proceso de pruebas a aplicaciones móviles en las empresas cubanas. Permitted conocer que el proceso de pruebas que se sigue, en muchos casos no se toman en cuenta actividades propias de las pruebas para aplicaciones móviles.

El objetivo de esta investigación es analizar las mejores prácticas de integración continua y entrega continua en el proceso de pruebas a aplicaciones móviles en el modelo de calidad para el desarrollo de aplicaciones informáticas.

Métodos o Metodología Computacional

El proceso de investigación sobre la integración y entrega continua en el proceso de pruebas a aplicaciones móviles estuvo guiado por los métodos de investigación científica que se dividen en teóricos y empíricos.

Entre los teóricos que se utilizan se encuentran:

Histórico-Lógico que brindan la posibilidad de conocer cómo se ha ido desarrollando el proceso de pruebas en los modelos de calidad en el mundo hasta la actualidad, y utilizarlos como punto de referencia y comparación de resultados alcanzados.

Analítico-Sintético que permite desarrollar un estudio de las pruebas a aplicaciones móviles y se extrajeron elementos relevantes que dieron la posibilidad de analizar, caracterizar y definir las fases del proceso de pruebas y sus aspectos específicos, para obtener una propuesta correcta.

Los métodos empíricos utilizados fueron:

Análisis documental para la identificación, recopilación y transformación de los documentos utilizados para enunciar las teorías que sustentan el estudio de las pruebas de software y la integración y entrega continua.

Observación que permitió conocer la realidad mediante percepción directa, impulsó el desarrollo de nuevos elementos y su refinación.

Calidad del Software

Desde el surgimiento de la ingeniería de software la definición de calidad se ha intentado formalizar dentro del contexto del desarrollo de software. Muchos autores están de acuerdo con la importancia que tiene la calidad del software para obtener aplicaciones con un rendimiento óptimo que cumplan con todos los requisitos establecidos por el cliente.

La IEEE, Std. 610-1990, plantea que “La calidad del software es el grado con el que un sistema, componente o proceso cumple los requerimientos especificados y las necesidades o expectativas del cliente o usuario”. También es definida por ISO/IEC 25000:2014, es conocida con el nombre de SQuaRE (Requisitos y Evaluación de Calidad de Productos de Software), como la capacidad de un producto de software de satisfacer las necesidades establecidas e implícitas cuando se utiliza bajo las condiciones específicas. Una de las definiciones más concretas de la calidad del software es de (Sommerville & Sawyer, 2005), que la define como un proceso de software efectivo que se aplica de manera que crea un producto útil que provee un valor medible a quienes lo producen y a quienes lo utilizan.

Analizando las definiciones anteriores se coincide con la IEEE, Std. 610-1990 respecto al concepto de calidad de software y se llega a la conclusión que es un proceso efectivo que se aplica al desarrollo de un producto, de manera que este cumpla con los requerimientos, necesidades, y expectativas del usuario final.

Características de calidad a evaluar en las pruebas de software

La característica de la calidad del software es la categoría de atributos de la calidad del software que influye en la calidad del software. Las características de la calidad del software pueden refinarse en múltiples niveles de subcaracterísticas y finalmente en los atributos de la calidad del software.

La norma NC-ISO 25010:2016 de calidad de software, define 8 características principales de calidad y sus características secundarias.

CALIDAD DEL PRODUCTO SOFTWARE								
ADECUACIÓN FUNCIONAL	EFICIENCIA DE DESEMPEÑO	COMPATIBILIDAD	CAPACIDAD DE INTERACCIÓN	FIABILIDAD	SEGURIDAD	MANTENIBILIDAD	FLEXIBILIDAD	PROTECCIÓN
COMPLETITUD FUNCIONAL	COMPORTAMIENTO TEMPORAL	COEXISTENCIA	RECONOCIBILIDAD DE ADECUACIÓN	AUSENCIA DE FALLOS	CONFIDENCIALIDAD	MODULARIDAD	ADAPTABILIDAD	RESTRICCIÓN OPERATIVA
CORRECCIÓN FUNCIONAL	UTILIZACIÓN DE RECURSOS	INTEROPERABILIDAD	APRENDIZABILIDAD	DISPONIBILIDAD	INTEGRIDAD	REUSABILIDAD	ESCALABILIDAD	IDENTIFICACIÓN DE RIESGOS
PERTINENCIA FUNCIONAL	CAPACIDAD		OPERABILIDAD	TOLERANCIA A FALLOS	NO-REPUDIO	ANALIZABILIDAD	INSTALABILIDAD	PROTECCIÓN ANTE FALLOS
			PROTECCIÓN FRENTE A ERRORES DE USUARIO	RECUPERABILIDAD	RESPONSABILIDAD	CAPACIDAD DE SER MODIFICADO	REEMPLAZABILIDAD	ADVERTENCIA DE PELIGRO
			INVOLUCRACIÓN DEL USUARIO		AUTENTICIDAD	CAPACIDAD DE SER PROBADO		INTEGRACIÓN SEGURA
			INCLUSIVIDAD		RESISTENCIA			
			ASISTENCIA AL USUARIO					
			AUTO-DESCRIPTIVIDAD					

Fig. 1 – Atributos de Calidad del producto del software [ISO/IEC 25010:2023]

Modelos de calidad

Los modelos de calidad integran la mayoría de las buenas prácticas, proponiendo temas de administración en los que las organizaciones deben hacer énfasis, además integran varias prácticas dirigidas a los procesos importantes para medir los avances de calidad. Existen varias definiciones sobre ello.

Modelo de calidad se define como herramientas que guían a las Organizaciones a la Mejora Continua y la Competitividad brindando especificaciones sobre qué tipo de requerimientos debe implementarse para poder brindar productos y servicios de alto nivel (Pesado et al., 2006). Un Análisis comparativo de modelos de calidad orientado al desarrollo de software define el modelo de calidad como un conjunto de buenas prácticas vinculadas a los procesos de gestión y desarrollo de proyectos (Llaneza et al., 2013). Este modelo supone una planificación para alcanzar un impacto estratégico, cumpliendo con los objetivos fijados en lo referente a la calidad del producto o servicio.

Dentro de las diferentes definiciones de modelos de calidad se coincide con la NC-ISO/IEC 25040:2016 dado que se concuerda en que son un conjunto de buenas prácticas o características que sirven como guía para obtener la calidad requerida del producto.

Resultados y discusión

Modelo de calidad de desarrollo de aplicaciones informáticas: MCDAI

Para los estándares y modelos de calidad son muy comunes los principios y criterios y a su vez existen algunas diferencias respecto a contenidos dentro de los criterios, pero todos tienen como objetivo principal la innovación y la mejora continua.

MCDAI, se distingue por su facilidad de uso y se plantea el propósito de servir de base para alcanzar evaluaciones futuras en otros modelos de referencia internacional (Oro et al., 2019). MCDAI se basa en los estándares y modelos internacionales referentes al desarrollo de software CMMI-DEV v1.3, MoProSoft (Modelo de Procesos para la Industria del Software), MPS.Br (Mejora de Proceso de Software Brasileño), COMPETISOFT (Mejora de Procesos para Fomentar la Competitividad de la Pequeña y Mediana Industria del Software de Iberoamérica), PMBOK, ISO 9001:2015 y la ISO 12207:2008 Procesos del Ciclo de Vida del Software. El MCDAI incluye como entrada las regulaciones nacionales y las buenas prácticas de los modelos y estándares de referencias mencionados anteriormente. Propone una estructura que soporta los principales procesos de la organización (Díaz Patterson & Silega Martínez, 2021).

Tabla 1 - Buenas prácticas seleccionadas de modelos y estándares de calidad.

Buenas prácticas	CMM I-DEV	TMM I	MPS.Br	MoProSoft	COMPETI SOFT	ISTQ B	ISO/IEC/IEEE		NC ISO/IEC 25040: 2016	MCDAI
							12207	29119		
Elaborar estrategia de pruebas.	x	x	x			x	x	x		x
Determinar la cobertura de las pruebas.	x	x				x		x		x
Seleccionar medidas de la calidad.	x	x				x	x	x	x	x
Automatizar la ejecución de las pruebas.	x	x				x		x		x
Analizar y	x	x	x	x	x	x	x	x	x	x

diseñar las pruebas.											
Determinar los elementos de cobertura.	x	x				x	x	x			x
Analizar y diseñar las pruebas para la reutilización.	x	x				x	x	x			x
Configurar entorno de pruebas.	x	x	x	x	x	x	x	x			x
Ejecutar las pruebas.	x	x	x	x	x	x	x	x	x		x
Analizar los resultados de las pruebas.	x	x	x	x	x	x	x	x	x		x
Identificar las causas de los defectos.	x	x									x
Evaluar las características de calidad.	x	x					x	x	x		x
Realizar seguimiento y control del proyecto de prueba.		x	x	x	x	x	x				x
Finalizar proyecto de prueba.	x	x	x	x	x	x	x	x			x

Fuente. Proceso de pruebas de software para un modelo de calidad en Cuba.(Figueredo, 2021)

Los modelos de calidad de software tratan las buenas prácticas de pruebas de software de manera general.

Proceso de pruebas del MCDAI

El proceso de pruebas para el MCDAI plantea una serie de requisitos que garantizan que las aplicaciones tengan la calidad adecuada.

Requisitos del proceso de Pruebas de MCDAI:

- Elaborar y mantener actualizada la política y estrategia de prueba organizacional
- Elaborar estrategia de pruebas
- Automatizar la ejecución de las pruebas
- Analizar y diseñar las pruebas
- Configurar el entorno de pruebas
- Ejecutar pruebas.
- Analizar los resultados de las pruebas
- Evaluar las características de la calidad
- Realizar seguimiento y control del proyecto de prueba
- Finalizar el proyecto de prueba

Aplicaciones móviles

En el caso particular de las aplicaciones móviles difiere del desarrollo de aplicaciones tradicional en muchos aspectos, lo que provoca que las metodologías usadas para estos entornos también difieran de las de las aplicaciones tradicionales; esto es, porque las aplicaciones para dispositivos móviles tienen que satisfacer una serie de requerimientos y condicionantes especiales tales como: canal, movilidad, portabilidad, capacidades específicas de las terminales, las cuales hacen de su desarrollo un proceso más complejo (Lara & Figueroa, 2020).

Debido a la cantidad de fabricantes existe una numerosa oferta de dispositivos móviles. Actualmente se clasifican los dispositivos en tres grandes grupos: teléfonos inteligentes, tabletas, dispositivos incorporados (“wearable”) y algunos dispositivos de internet de las cosas (IoT) por sus siglas en inglés. Esta clasificación se fundamenta en el uso que se le da a cada grupo. Otro elemento que debe tener presente relacionado con los dispositivos móviles son los componentes del mismo, con el objeto de estructurar los diferentes casos de prueba relacionados con las funcionalidades de la aplicación. Casos de prueba que pueden estar asociados a

conexiones wifi, GPS, bluetooth, tecnología NFC (Near Field Communication), batería, tarjeta SIM, memoria, sensor de movimiento, sensor de luz, sensor de localización, entre otros.

Existen tres tipos de aplicaciones móviles: nativas, híbridas y web. A diferencia de las aplicaciones web, las aplicaciones nativas e híbridas están instaladas físicamente en el dispositivo y, por lo tanto, siempre están disponibles para el usuario. Dependiendo del tipo de aplicación, el probador establece qué tipos de casos debe incorporar. Dependiendo de esto y del volumen de usuarios esperado para la aplicación, se pueden incluir casos de prueba de desempeño (Velásquez et al., 2019).

Las pruebas en aplicaciones móviles, como cualquier otro tipo de pruebas de aplicaciones, se basan en la verificación y validación de las métricas priorizadas de acuerdo con la experiencia del usuario, la población a la cual va dirigida, el tipo de aplicación móvil, el tipo de dispositivo, la plataforma tecnológica empleada para su desarrollo y ejecución; que el probador establece a través del plan de pruebas y el diseño de casos de prueba, para los cuales debe aplicar técnicas de estimación y diseños (caja negra y caja blanca).

(Rasool & Ali, 2020) Mencionan que “el éxito de aplicaciones móviles depende de su calidad, fiabilidad, corrección, rendimiento y satisfacción general de los usuarios.”

(Maia et al., 2019) Plantean la necesidad de considerar otros factores propios de los dispositivos que ejecutan este tipo de aplicaciones, tales como memoria, red, almacenamiento y pantalla debido a que influyen en el funcionamiento de las mismas.

Definir la calidad de un software móvil requiere el análisis de diversas métricas que estén acorde a su contexto, debido a que este tipo de software se desarrolla teniendo presente el campo de aplicación y las limitaciones de los dispositivos donde se utilizarán, como batería, capacidad de procesamiento, memoria, dimensiones de pantalla, red, entre otras (Carrión et al., 2021).

Tabla 2 - Métricas consideradas para aplicaciones móviles

Investigaciones	Métricas consideradas
(Syer et al., 2015)	Dependencia de Plataforma Móvil (SO)
(Noei et al., 2017)	Interfaz –Rendimiento- Tamaño
(Mendonça et al., 2019)	Energía- Tiempo de ejecución- Disponibilidad -Conexión -Rendimiento
(Pandey et al., 2019)	Funcionalidad- Interfaz -Rendimiento –Compatibilidad- Conexión- Tiempo de respuesta, Energía -Seguridad

(Maia et al., 2019)	Energía- Memoria- Almacenamiento –Rendimiento- Pantalla- Dependencia de red- Usabilidad- Mantenibilidad -Portabilidad -Eficiencia -Compatibilidad -Satisfacción
(Xiang et al., 2020)	Rejuvenecimiento del software- Confiabilidad- Disponibilidad
(Soui et al., 2020)	Interfaz de Usuario Móvil (MUI) -Interacción -Usabilidad -Eficacia
(Rodrigues et al., 2020)	Funcionalidad –Estética- Entretenimiento -Interactividad
(Biørn-Hansen et al., 2020)	Multiplataforma- Rendimiento
(Davalbhakta et al., 2020)	Funcionabilidad- Amigable- Interactiva -Accesibilidad

Tendencias de las pruebas a aplicaciones móviles

La seguridad, como atributo de calidad, es la capacidad de protección de la información y los datos de manera que personas o sistemas no autorizados no puedan leerlos o modificarlos (Redondo & Cárdenas, 2022). El aumento de la seguridad en las aplicaciones móviles es muy importante debido a que la información de los usuarios y de las empresas es vulnerable a las amenazas. El aumento de los dispositivos IoT aumenta las posibilidades de ataque, por lo que las organizaciones lo deben tener en cuenta y reforzar la ciberseguridad. Algunas de las prácticas más comunes son las pruebas de penetración que permiten detectar vulnerabilidades que pueden dar lugar a filtraciones de datos. Las pruebas de seguridad garantizan que los sistemas y servidores internos sean protegidos adecuadamente.

Las pruebas automatizadas eliminan la necesidad del esfuerzo manual por lo que los equipos de desarrollo pueden responder más rápido a las demandas del mercado con mayor eficiencia, velocidad, reducción de riesgos y bucles de retroalimentación continua. Así las pruebas de aplicaciones móviles son un elemento importante del proceso de desarrollo, llevando así mayor fiabilidad del producto al cliente.

Usar pruebas de automatización sin código es más simple y rápido. Sin tener conocimientos de programación los probadores pueden crear y automatizar casos de prueba. Incorporarlas en la estrategia de pruebas puede aumentar la cobertura de pruebas. Las plataformas de automatización de pruebas móviles, reducen el tiempo y el dinero invertidos en crear y probar aplicaciones para dispositivos móviles.

Las pruebas móviles en la nube son cada vez más frecuentes (Ali et al., 2022). Los equipos pueden probar las aplicaciones en diversas plataformas y hardware para hacer un seguimiento de las posibles configuraciones de hardware y dispositivos de prueba. Las pruebas en la nube son escalables y adaptables y la realización de estas pruebas responde mejor a los constantes cambios que están surgiendo en torno a los dispositivos móviles, por lo que se incorporan pruebas automatizadas y manuales en la nube a la estrategia

de pruebas. Cuando se realizan pruebas en la nube de dispositivos reales permite que las aplicaciones sean más robustas y tengan menos errores que otras aplicaciones. Con este enfoque se pueden realizar pruebas manuales y automatizadas, y en estas últimas se pueden llevar a cabo a gran escala.

En casi todos los aspectos del ciclo de desarrollo, el uso de la Inteligencia Artificial (IA) y el aprendizaje automático ayuda a mejorar entre otras cosas las estrategias de prueba y la detección de errores. Se puede optimizar la cobertura de pruebas, las pruebas que se deben priorizar y demás. El aprendizaje automático se vuelve más potente cuando se recopilan datos porque puede realizar revisiones y realizar una exploración más profunda y así encontrar problemas que las pruebas manuales no tendrían en cuenta.

El desarrollo de Interfaz de Programación de Aplicaciones por sus siglas en inglés API, ha aumentado, muchos sitios utilizan funcionalidades para ser usadas a través de las API para desarrollar aplicaciones móviles más rápido. Debido a que las API disminuyen los requisitos de codificación y permiten un rápido despliegue, hace que su integración y mantenimiento sea más sencillo y ha aumentado el desarrollo del lado cliente-servidor por lo que son necesarias las pruebas de las API, no solo se debe probar la funcionalidad de la API, sino también el rendimiento, la seguridad y el trabajo conjunto de todos los componentes. Los equipos de prueba se tendrán que centrar en la comunicación entre aplicaciones y las API. Aumentarán las pruebas para asegurar la seguridad en las API y su automatización para ayudar a que las API sean más eficientes.

Dado que cada vez se comercializan más los wearables, las aplicaciones de IoT son cada vez más populares debido a que cada vez más hay aplicaciones que interactúan con estos dispositivos. Ya que los dispositivos IoT se conectan a los otros dispositivos móviles, el proceso de pruebas se hace más complejo. Se debe tener en cuenta las pruebas de seguridad y procesamiento de datos, consumo de energía, problemas de conexión y fallos de comunicación. Todo ello con dispositivos reales y con enfoque en la nube donde debido a la gran variedad y diversidad de dispositivos el enfoque de pruebas en la nube hace menos difíciles estas pruebas.

La automatización del proceso de aseguramiento de la calidad es una parte importante del desarrollo, a pesar de que siempre habrá algún tipo de control manual. DevOps (Desarrollo y Operaciones), es un enfoque basado en principios ágiles en que los interesados y los departamentos de desarrollo, operaciones y control de calidad colaboran para entregar de manera continua, lo que permite a la empresa aprovechar rápidamente las oportunidades del mercado y reducir la carga de trabajo. Integración continua(IC), fusiona

el proceso de desarrollo de muchos desarrolladores a través de un repositorio de código compartido. CI es una práctica del desarrollo de software, en donde los desarrolladores integran el código fuente con frecuencia y cada integración es verificada por un sistema que construye el código y lo prueba automáticamente. Entrega continua(EC), automatiza el proceso de implementación de una nueva construcción en producción, contempla realizar pruebas automatizadas en cada nueva compilación; gestiona el aprovisionamiento y la configuración automática de los entornos de implementación, las pruebas para comprobar la estabilidad, el rendimiento y el cumplimiento de la seguridad. Integra las fases de prueba y lanzamiento. Teniendo en cuenta esto, el enfoque ágil, la automatización de pruebas sin código y la IC y EC, son prácticas adecuadas que cuando se aplican se puede automatizar más el proceso de control de la calidad.

El desarrollo ágil sigue siendo fundamental en la industria móvil. La IC y EC, forman parte del desarrollo de aplicaciones. A través del uso de pruebas automatizadas la IC y EC se adaptan a cada fase del ciclo de vida de desarrollo. Con la IC y la EC se realizan cambios en el código más rápido y fiable; a menudo el código se añade a un repositorio compartido y se utilizan herramientas de compilación automatizadas para verificar el código y su estabilidad por lo que los problemas se detectan en una fase temprana del proceso y se corrigen casi al instante.

Apoyándose la documentación estudiada, el análisis de las tendencias y el diagnóstico realizado, se obtienen resultados que demuestran que el proceso de pruebas para aplicaciones en MCDAI no es suficientemente específico para los requerimientos de las aplicaciones para dispositivos móviles. Los anteriores aspectos apoyan la necesidad de definir una instanciación del proceso de pruebas para aplicaciones de dispositivos móviles en MCDAI con enfoque ágil que permitirá asegurar la calidad del producto durante todo su ciclo de vida.

Beneficios de IC/EC en pruebas a aplicaciones de Dispositivos Móviles

En el desarrollo de aplicaciones móviles existe una tendencia por el uso de las metodologías con principios ágiles por los beneficios que brinda durante el ciclo de vida del software. Sin embargo, no se pueden considerar todos los marcos de trabajo con base en los mismos principios como metodologías ideales para el desarrollo de aplicaciones móviles (Molina Ríos et al., 2021).

La prueba de aplicaciones móviles es un proceso multifacético que implica diversos tipos de pruebas para garantizar la calidad y confiabilidad de las aplicaciones móviles. Cada tipo de prueba cumple un propósito específico en la evaluación de diferentes aspectos de la funcionalidad, rendimiento, usabilidad, compatibilidad y seguridad de una app.

En Integración Continua, se recomienda automatizar las pruebas unitarias, de integración y de rendimiento e incluirlas en los scripts de construcción. Las herramientas de IC ejecutan automáticamente pruebas unitarias, de integración y de aceptación en emuladores/simuladores cada vez que se realiza un cambio. Y mediante la detección temprana de errores se proporciona retroalimentación rápida a los desarrolladores sobre la calidad de la nueva integración. A su vez, como la EC se basa en la IC con las implementaciones frecuentes permite a los desarrolladores lanzar nuevas funcionalidades y correcciones más rápidamente. Además del feedback rápido que se obtiene a medida que las pruebas se ejecutan automáticamente para conocer de inmediato sobre el rendimiento en entornos simulados. Cuando se automatizan las pruebas para diversas configuraciones de dispositivos se reduce el tiempo de las pruebas. También se asegura que el código se prueba en tantos escenarios como sea posible aumentando la cobertura de pruebas. Y así se identifican y corrigen errores continua y tempranamente en el ciclo de desarrollo, mejorando de esta forma la calidad del producto.

En el ámbito de las aplicaciones móviles, la IC es particularmente importante debido a la diversidad de plataformas, dispositivos, resoluciones de pantalla y versiones de sistema operativo que deben soportarse. La ejecución automatizada de pruebas sobre múltiples entornos simulados permite identificar fallos de compatibilidad y garantizar una experiencia de usuario coherente y estable. Así mismo, con la entrega continua, los equipos de desarrollo pueden responder rápidamente a incidentes críticos, introducir mejoras incrementales y adaptarse de manera ágil a cambios en los requisitos del mercado o del entorno tecnológico. Esto es crucial en las aplicaciones móviles, donde las expectativas del usuario son altas y la competencia exige una capacidad de actualización constante y sin interrupciones.

En la actualidad, existen muchos tipos y niveles de pruebas, pero no todos pueden abordarse del mismo modo en los ambientes ágiles. Inclusive, las pruebas deben cumplir ciertos requerimientos para ser implementadas en los entornos continuos. Uno de estos requerimientos es la rapidez en sus tiempos de ejecución.

La IC/EC garantiza mucha ventaja en las pruebas de aplicaciones de Dispositivos Móviles permitiendo que sea mejor la calidad del producto ya que la automatización garantiza que la aplicación esté libre de defectos significativos al momento del despliegue. Las aplicaciones pueden ser actualizadas constantemente mediante despliegues rápidos y frecuentes. Esto garantiza que los equipos de desarrollo se enfoquen en mejorar la funcionalidad de la aplicación. De esta forma la capacidad de respuesta es más rápida y se aseguran las actualizaciones frecuentes que mejoran la experiencia general del usuario.

La validación de la IC/EC es una necesidad para asegurar un desarrollo ágil. La IC permite incorporar cambios de código en un repositorio compartido varias veces al día junto con la ejecución automática de pruebas. La EC la frecuencia de actualizaciones es alta y las fallas pueden afectar negativamente la experiencia de usuario o derivar en la pérdida de datos. Un proceso de IC/EC validado garantiza que el código se integre de forma segura, que la aplicación funcione adecuadamente en múltiples configuraciones. Permite reducir el tiempo de respuesta ante fallos. Humble y Farley afirman que se deben automatizar todos los niveles de pruebas posibles (Humble & Farley, 2010) y Shahin et al. realizaron un estudio que demuestra que la automatización de pruebas es una pieza clave en Entrega Continua (Shahin et al., 2017). Sin embargo, las pruebas exploratorias manuales son importantes en EC y no pueden ser automatizadas ya que requieren de la experiencia de un equipo especialista en pruebas manuales calificado.

La aplicación de CI/CD en el proceso de pruebas de aplicaciones móviles ofrece múltiples beneficios:

- Reducción de errores en producción mediante la detección temprana.
- Mayor frecuencia de actualizaciones sin comprometer la estabilidad.
- Aumento en la cobertura de pruebas, incluso en escenarios complejos.
- Reducción del tiempo de desarrollo y despliegue, lo que favorece la competitividad del producto.
- Mayor confianza del equipo de desarrollo y del usuario final en el producto entregado.

Para ello existen herramientas comunes para la IC/EC en aplicaciones móviles.

Appium, herramienta de automatización de pruebas multiplataforma de código abierto para aplicaciones web, escritorio, híbridas y móviles que permite crear pruebas en diversas plataformas utilizando la misma API lo que permite la reutilización de código entre varios casos de prueba definidos en cualquier plataforma, de tal manera que es considerado un ecosistema de software relacionado, diseñado para facilitar la automatización de la interfaz de usuario multiplataforma.

Selendroid es una herramienta de automatización de pruebas para aplicaciones nativas e híbridas de Android, es un complemento de Selenium WebDriver. Elimina la interfaz de usuario y utiliza la API del cliente Selenium 2 para escribir y ejecutar pruebas.

Jenkins es una aplicación que ayuda a los desarrolladores a realizar Integración Continua de proyectos de software y ayuda detectar y corregir errores de integración de forma continua. Fue creado del desarrollo original Hudson. Esta herramienta está implementada en lenguaje de programación Java por lo siguiente para poder utilizarlo es necesario una máquina virtual.

Travis CI se basa en su proceso de desarrollo donde crea y prueba automáticamente los cambios de código, proporcionando información inmediata sobre el éxito del cambio. Travis CI también puede administrar otros procesos del desarrollo administrativo, como los despliegues de nuestros proyectos en algún entorno específico y también provee notificaciones, lo cual parece bien para mantenernos al tanto de lo ocurrido.

Travis CI: es una plataforma en la nube que funciona de manera gratuita para repositorios en GitHub públicos y BitBucket, esta herramienta ofrece opciones de integración continua automatizadas que eliminan la necesidad de un servidor dedicado, lo que permite realizar pruebas en diferentes entornos, en varias máquinas, y corriendo en diferentes sistemas operativos.

CircleCI es una herramienta basada en la nube que automatiza el proceso de integración e implementación. También se enfoca en probar cada cambio de código antes de su implementación, utilizando métodos como pruebas unitarias, pruebas de integración y pruebas funcionales. Se integra perfectamente con el sistema de control de versiones actual.

GitLab CI/CD es un servicio que viene incluido en GitLab, el cual construye y prueba el software cada vez que un desarrollador actualiza el repositorio. Dentro de la Suite de GitLab también existe GitLab CD, que se encarga de la entrega continua, después de que se completa la integración por medio de GitLab CI, se puede ejecutar la implementación continua para actualizar los cambios dentro del producto que ya está en producción.

Testim es una herramienta de prueba de software automatizada inteligente que utiliza el aprendizaje automático para acelerar el diseño, la ejecución y el mantenimiento de casos de prueba automatizados. Los casos de prueba se pueden ejecutar en múltiples plataformas, incluidos dispositivos móviles. Testim usa anotaciones para encontrar inconsistencias y errores en el sistema. Los errores que se registran se pueden

reproducir automáticamente simplemente haciendo clic en la prueba nuevamente. El rastreador de errores Testim se usa para compartir capturas de pantalla anotadas y ver detalles de errores.

Appium Se trata de una herramienta que realiza pruebas móviles automatizadas para Android e iOS. Appium Essentials es una guía práctica que le ayudará a realizar pruebas de automatización móvil y a comprender bien los conceptos de automatización móvil.

Bitrise. La entrega continua cuenta también con varias aplicaciones web que permiten reflejar el concepto de automatizar pasos (pipeline) en el desarrollo de software. Dado que se está trabajando con aplicaciones móviles, Bitrise es una de las herramientas que dan opción para compilar aplicaciones híbridas (iOS y Android) en sus servidores de manera gratuita. Esto quiere decir que tendremos la posibilidad de incluir un paso de compilación para iOS (utilizando un servidor con macOS que es el sistema operativo para poder compilar aplicaciones para iOS) y esto no es brindado de manera gratuita por casi ninguna herramienta de pipelines. En Bitrise se reflejará todos los pasos que harán que el proceso de creación de las apps sea automatizado, rápido y fiable.

IC/EC en el Proceso de pruebas a aplicaciones móviles de MCDAI

En MCDAI los requisitos específicos del proceso base (PB) están agrupado por niveles. En la Figura 2 se listan los requisitos específicos del PB, indicando cuáles son necesarios cumplir para alcanzar cada nivel. A continuación, se explican se indican los requisitos y notas específicas para el proceso de pruebas a aplicaciones móviles.

PS 2 Elaborar estrategia de pruebas

Nota 2.1: Además de las estrategias de pruebas mencionadas en la nota dos, la estrategia de pruebas en el caso de las pruebas para aplicaciones móviles puede incluir la metodología ágil de Desarrollo impulsado por pruebas de aceptación (ATDD). Es una práctica de diseño de software en donde se obtiene el producto a partir de las pruebas de aceptación, se basa en la misma notación de las pruebas TDD con la diferencia de que los programadores que escriben pruebas de unidad también hacen las pruebas de aceptación de usuarios. La idea es tomar cada requerimiento, en la forma de una Historia de usuario, luego construir varias pruebas de aceptación del usuario de las cuales se van a construir las pruebas automáticas de aceptación para finalmente escribir el código.

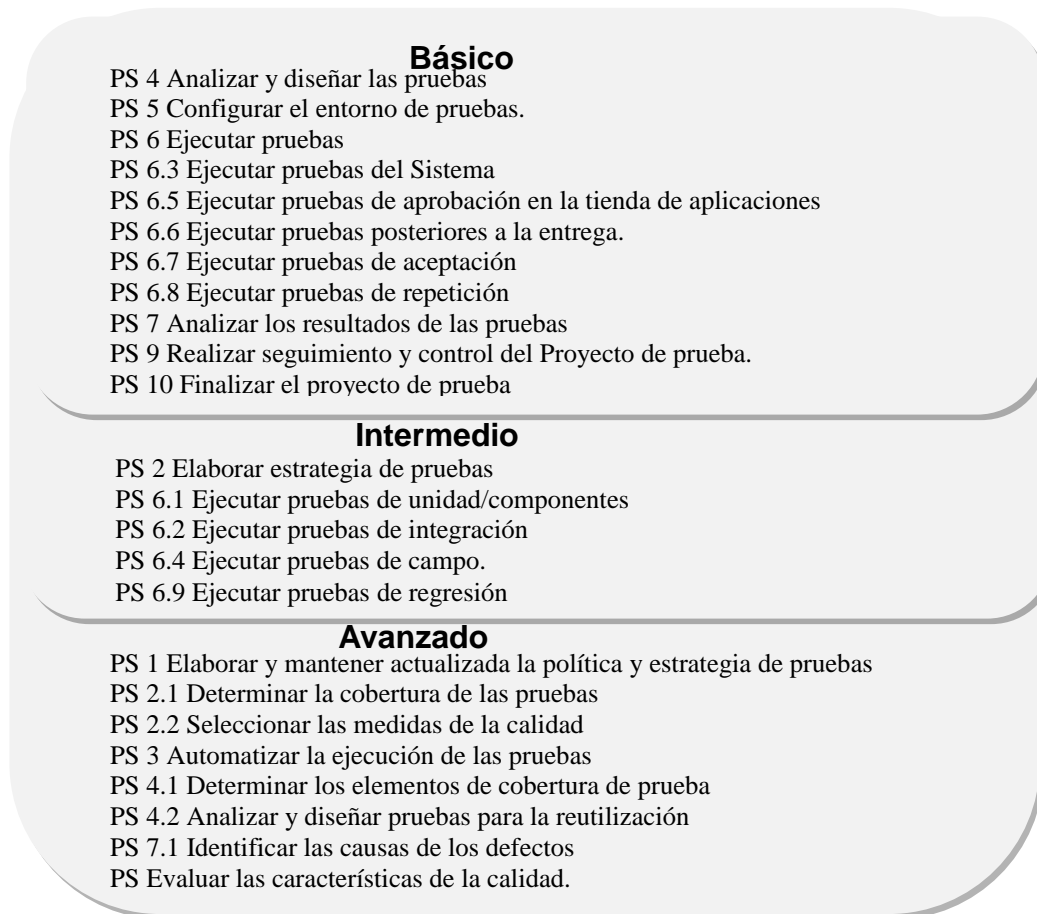


Fig. 2 – Requisitos específicos del PB Pruebas de Software Móviles a cumplir por cada nivel.
(Elaboración propia)

Nota 5.1: Además de los cuatro niveles de prueba enunciados en la nota 5, en las pruebas para aplicaciones móviles se necesitan tres nuevos niveles: pruebas de campo, pruebas para la Aprobación del Almacén de Aplicaciones y Pruebas posterior a la entrega.

Nota 8.1: Durante la definición de los requisitos del entorno de pruebas deben tenerse en cuenta el uso de emuladores y simuladores móviles. Y la combinación de dispositivos que necesitan probarse.

PS 3 Automatizar la ejecución de las pruebas

Nota 2: Se escribirán scripts de pruebas para ejecutar casos de prueba desde la primera fase del proyecto. Se automatizarán las pruebas diseñadas utilizando las herramientas especificadas. Se automatizarán las pruebas

de los defectos encontrados y se reutilizarán estos scripts en nuevas versiones de la aplicación para mejorar eficiencia en la ejecución de la prueba.

Nota 3: Identificar en las historias de usuario que funcionalidades se deben priorizar para automatizar.

Nota 4: Las herramientas de automatización seleccionadas deben ser compatibles con los dispositivos móviles y ser compatibles con las herramientas de integración continua utilizadas en el proceso de desarrollo. Se pueden utilizar herramientas de automatización específicas para dispositivos móviles, como: Appium, Espresso y XCUITest.

Nota 5: Los scripts integrados en la integración y entrega continua (IC/EC) deben garantizar que se ejecuten las pruebas automáticamente cada vez que se realice algún cambio en el código.

PS 4 Analizar y diseñar las pruebas

Nota 3: Se diseñarán los casos de prueba que pueden ser automatizados incluyendo los de la integración continúa.

Nota 4: Tener en cuenta el diseño de casos de pruebas para los niveles de prueba de campo y el nivel de prueba en la tienda de aplicaciones y posterior a la entrega.

Nota 5: Durante el diseño de la prueba de instalabilidad se debe tener en cuenta como condición de prueba instalar, reinstalar, desinstalar, actualizar y cancelar la instalación de la aplicación. Todo esto mediante wifi o datos móviles o desde el cable de datos. Es importante verificar el comportamiento si el usuario niega el permiso, si se interrumpe el proceso por causa de cancelación, interrupción por apagar el dispositivo o desconectándolo de internet.

Nota 6: para el diseño de los casos de pruebas para probar la característica eficiencia del desempeño, deben diseñarse para ejecutarse en el dispositivo. Es importante tener en cuenta el rendimiento percibido por el usuario.

PS 5 Configurar el entorno de pruebas

Nota 4: En el entorno de pruebas para aplicaciones móviles, se deben configurar herramientas de automatización de pruebas. Herramientas para configurar el entorno de integración continua que incluya las pruebas automatizadas para validar cada cambio de código de la aplicación móvil.

Nota 4.1: En el caso de las herramientas de CI se recomiendan Jenkins, Travis CI, CircleCI o GitLab CI para automatizar el proceso de integración y prueba.

Nota 4.2: para la configuración del Repositorio ejemplo como Git, para almacenar el código fuente y las pruebas automatizadas.

Nota 5: Se debe configurar una infraestructura de pruebas que incluya dispositivos reales además de los emuladores y simuladores para móviles, para cubrir una amplia gama de dispositivos y sistemas operativos.

PS 6 Ejecutar pruebas

Nota 3: Las pruebas deben ejecutarse automáticamente cada vez que se realicen cambios en el código, que permite detectar y corregir errores de forma temprana.

Conclusiones

El proceso de pruebas para aplicaciones móviles tiene diferencias en cuanto a las pruebas que se realizan a otros tipos de aplicaciones. Estas pruebas son esenciales para el desarrollo de las aplicaciones móviles garantizando la calidad y la satisfacción del usuario, identificando y resolviendo los problemas antes de que sea utilizada por este. Constantemente surgen nuevas tendencias en las pruebas a aplicaciones móviles que influyen en las buenas prácticas que deben aplicarse en los procesos de pruebas. La investigación propone un enfoque basado en integración y entrega continua para el proceso de pruebas a aplicaciones móviles en MCDAI donde se pueden actualizar constantemente mediante despliegues rápidos y frecuentes, garantizando así, la capacidad de respuesta rápida y se aseguran las actualizaciones frecuentes aportando al aumento de la calidad del producto y mejorando la experiencia del usuario.

Referencias

- Aguilar, L. J. (2021). *Internet de las cosas: Un futuro hiperconectado: 5G, inteligencia artificial, Big Data, Cloud, Blockchain, Ciberseguridad*. Alpha Editorial.
- Ali, A., Maghawry, H. A., & Badr, N. (2022). Performance testing as a service using cloud computing environment: A survey. *Journal of Software: Evolution and Process*, 34(12), e2492. <https://onlinelibrary.wiley.com/doi/abs/10.1002/smr.2492>

- Carrión, J. L. A., Roman, R. M., Castillo, F. F. R., & Apolinario, D. A. T. (2021). Estado del arte: métricas del desarrollo de software móvil. *3c Tecnología: glosas de innovación aplicadas a la pyme*, 10(3), 17-37. <https://dialnet.unirioja.es/descarga/articulo/8097700.pdf>
- Díaz Patterson, D., & Silega Martínez, N. (2021). Enfoque ontológico para el análisis de estándares de calidad del proceso de software. *Revista Cubana de Ciencias Informáticas*, 15(3), 136-152. <http://scielo.sld.cu/pdf/rcci/v15n3/2227-1899-rcci-15-03-136.pdf>
- Figueredo, L. (2021). Proceso de pruebas de software para un modelo de calidad en Cuba. *I+ D Tecnológico*, 17(1), 23-35. <https://revistas.utp.ac.pa/index.php/id-tecnologico/article/download/2914/3617>
- Graham, D., Veenendaal, E. v., Evans, I., & Black, R. (2006). *Foundations of software testing: ISTQB certification*. Cengage Learning.
- Humble, J., & Farley, D. (2010). *Continuous delivery: reliable software releases through build, test, and deployment automation*. Pearson Education. <https://ptgmedia.pearsoncmg.com/images/9780321601919/samplepages/0321601912.pdf>
- Lara, C., & Figueroa, L. M. (2020). Metodología ágil para el desarrollo de aplicaciones móviles educativas. XV Congreso Nacional de Tecnología en Educación y Educación en Tecnología (TE&ET 2020)(Neuquén, 6 y 7 de julio de 2020),
- Llaneza, M., Dapozo, G. N., Greiner, C. L., & Estayno, M. G. (2013). Análisis comparativo de modelos de calidad orientado al desarrollo de software en pymes. XV Workshop de Investigadores en Ciencias de la Computación,
- Maia, V., Gonçalves, T. G., & da Rocha, A. R. C. (2019). Quality characteristics of mobile applications: A survey in Brazilian context. Proceedings of the XVIII Brazilian Symposium on Software Quality,
- Mascheroni, M. A. (2021). *Modelo de mejora para pruebas continuas* Universidad Nacional de La Plata]. https://sedici.unlp.edu.ar/bitstream/handle/10915/122709/Documento_completo.pdf?sequence=1
- Mezones, M. K. B., & Vaca-Cárdenas, M. E. (2021). Dispositivos móviles en los trastornos de conductas de los niños de 0 a 3 años. <https://revistas.utm.edu.ec/index.php/Cognosis/article/download/3206/4672>.

- Mitasiunas, A., Rout, T., O'Connor, R. V., & Dorling, A. (2014). Software process improvement and capability determination. *Communications in Computer and Information Science*, 477. <https://link.springer.com/content/pdf/10.1007/978-3-319-13036-1.pdf>
- Molina Ríos, J. R., Honores Tapia, J. A., Pedreira Souto, N., & Pardo, H. (2021). Comparativa de metodologías de desarrollo de aplicaciones móviles. <https://ruc.udc.es/bitstreams/523936a6-cd0b-422e-a2f7-f5b19aa5dbce/download>
- Oro, L., Alvarado, Y., Felipe, J., & Ramírez Pérez, J. (2019). La gestión de reutilización de software en el Modelo de la Calidad para el Desarrollo de Aplicaciones Informáticas en Cuba.
- Pesado, P. M., Bertone, R. A., Ramón, H. D., Pasini, A. C., Esponda, S., & Alonso, L. M. (2006). Calidad en el desarrollo de Sistemas de Software. VIII Workshop de Investigadores en Ciencias de la Computación,
- Puetate, G., & Ibarra, J. L. (2020). Aplicaciones móviles híbridas. *Centro de publicaciones PUCE*. <https://biblioteca.ismejia.com/files/pam/Aplicaciones-M%D0%B2viles-H%D0%B1bridas-2020.pdf>
- Rasool, G., & Ali, A. (2020). Recovering android bad smells from android applications. *Arabian Journal for Science and Engineering*, 45(4), 3289-3315. <https://link.springer.com/article/10.1007/s13369-020-04365-1>
- Redondo, A. M. F., & Cárdenas, F. d. J. N. (2022). DevOps: un vistazo rápido. *Ciencia Huasteca Boletín Científico de la Escuela Superior de Huejutla*, 10(19), 35-40. <https://repository.uaeh.edu.mx/revistas/index.php/huejutla/article/download/8121/8570/>
- Shahin, M., Babar, M. A., & Zhu, L. (2017). Continuous integration, delivery and deployment: a systematic review on approaches, tools, challenges and practices. *IEEE access*, 5, 3909-3943. <https://ieeexplore.ieee.org/iel7/6287639/6514899/07884954.pdf>
- Socarrás Ramírez, I., Vega Prieto, R., & Trujillo Casañola, Y. (2022). El Proceso de definición y enfoque de los procesos en las organizaciones desarrolladoras de software Haciendo uso de MCDAI. *Revista Cubana de Ciencias Informáticas*, 16(3), 102-117. http://scielo.sld.cu/scielo.php?pid=S2227-18992022000300102&script=sci_arttext
- Sommerville, I., & Sawyer, P. (2005). Ingeniería del software: Un enfoque práctico. *Eddison Wesley, México*.

- Thomas, P. J., Delía, L. N., Corbalán, L. C., Fernández Sosa, J. F., Tesone, F., Aguirre, V., Olsowy, V., & Pesado, P. M. (2022). Enfoques y tendencias en el desarrollo de aplicaciones móviles con resiliencia. XXIV Workshop de Investigadores en Ciencias de la Computación (WICC 2022, Mendoza),
- Velásquez, S. M., Sossa, D. E. M., Zapata, M. E., Adasme, M. E. G., & Ríos, J. P. (2019). Pruebas a aplicaciones móviles: avances y retos. *Lámpsakos (revista descontinuada)*(21), 39-50.
<http://revistas.ucatolicaluisamigo.edu.co/index.php/lampsakos/article/download/2983/2527>

Conflicto de interés

Los autores autorizan la distribución y uso de su artículo.

Contribuciones de los autores

Conceptualización: Yaneisy Onelia Massó Agramonte.

Curación de datos: Yoandy Lazo Alvarado

Análisis formal: Yaneisy Onelia Massó Agramonte

Adquisición de fondos: Yaneisy Onelia Massó Agramonte

Investigación: Yaneisy Onelia Massó Agramonte

Metodología: Yoandy Lazo Alvarado

Administración del proyecto: Yaneisy Onelia Massó Agramonte

Recursos: Yaneisy Onelia Massó Agramonte

Software: Yaneisy Onelia Massó Agramonte

Supervisión: Yoandy Lazo Alvarado

Validación: Yaneisy Onelia Massó Agramonte

Visualización: Yoandy Lazo Alvarado

Redacción – borrador original: Yaneisy Onelia Massó Agramonte

Redacción – revisión y edición: Yoandy Lazo Alvarado