RUTA HACIA LA COMPRENSIÓN DE LA CALIDAD DE LAS HERRAMIENTAS DE PRUEBAS HABILITADAS CON INTELIGENCIA ARTIFICIAL

PATH TOWARDS UNDERSTANDING THE QUALITY OF ARTIFICIAL INTELLIGENCE ENABLED TESTING TOOLS

Giulianna Bortone[1], **Isabela Espinoza**[1], **María Pérez**[1], **Dinarle Ortega**[2]
[1]Universidad Metropolitana, Caracas- Venezuela
[2]Universidad Católica Andrés Bello.Caracas- Venezuela
Email: b.giulianna@correo.unimet.edu.ve

Recibido: 12-09-2024
Aceptado: 19-10-2024

RESUMEN

En el contexto del desarrollo de productos de software, se considera al enfoque DevSecOps como una evolución de DevOps, el cual enfatiza la importancia de incorporar la seguridad en todas las etapas del ciclo de vida del software, haciendo énfasis en la automatización e integración de este proceso. Sin embargo, la complejidad de este enfoque exige herramientas de pruebas que no solo garanticen la calidad del software, sino que también se adapten a las necesidades de seguridad y eficiencia. Este trabajo, que es una investigación en progreso, propone un Modelo de Dominio utilizando la notación UML para comprender y establecer una base sólida, requerida para la posterior formulación de un modelo de estimación de la calidad robusto y operacionalizado, que apoye la selección de las herramientas de pruebas, en un contexto DevSecOps. Por lo tanto, se revisaron los estándares ISO 25010 y 25059, el enfoque DevSecOps y lo relacionado con la etapa de pruebas y sus herramientas de soporte habilitadas con IA. Este trabajo subraya la necesidad de una terminología común que facilite la colaboración y comunicación entre las partes interesadas.

Palabras clave: Herramientas de pruebas, DevSecOps, Inteligencia Artificial, ISO 25010, ISO 25059.

ABSTRACT

In the context of software product development, the DevSecOps approach is considered an evolution of DevOps, emphasizing the importance of incorporating security at all stages of the software lifecycle, with a focus on automation and integration of this process. However, the complexity of this approach requires testing tools that not only ensure software quality but also meet security and efficiency needs. This work, which is an ongoing investigation, proposes a Domain Model using UML notation to understand and establish a solid foundation necessary for the subsequent formulation of a robust and operationalized quality estimation model, aimed at supporting the selection of testing tools in a DevSecOps context. Therefore, ISO 25010 and 25059 standards, the DevSecOps approach, and aspects related to the testing stage and its AI-enabled support tools were reviewed. This work highlights the need for a common terminology to facilitate collaboration and communication among stakeholders.

Key words: Testing tools, DevSecOps, Artificial Intelligence, ISO 25010, ISO 25059.

Giulianna Bortone: Estudiante de Ingeniería de Sistemas, Universidad Metropolitana Caracas- Venezuela. Desarrolladora web Grupo Mon. Miranda, Venezuela. b.giulianna@correo.unimet.edu.ve

Isabela Espinoza: Estudiante de Ingeniería de Sistemas, Universidad Metropolitana Caracas- Venezuela.Desarrolladora web Gioseg Miranda, Venezuela. isabela.espinoza@correo.unimet.edu.ve

María Pérez: Doctorado en Ciencias de la Computación, Profesora Titular,Escuela de Ingeniería de Sistemas, Universidad Metropolitana, Caracas- Venezuela. maperez@unimet.edu.ve

Dinarle Ortega: Doctorado en Ciencias de la Computación. Profesora Titular, Escuela de Informática, Universidad Católica Andrés Bello.Caracas. dortega@ucab.edu.ve

Introducción

DevSecOps ofrece un enfoque de desarrollo que enfatiza la colaboración entre los equipos de desarrollo, operaciones y seguridad a lo largo del desarrollo del producto de software. En particular, la selección y evaluación de herramientas de pruebas para entornos DevSecOps es un área que ha recibido poca atención en las investigaciones realizadas, ellas permiten identificar y mitigar vulnerabilidades de manera temprana.[1]

Este artículo se considera el resultado de una iteración inicial de una investigación en progreso relacionada con los criterios de calidad de las herramientas de pruebas de software más utilizadas en DevSecOps.

En particular, aquí el objetivo es presentar un modelo de dominio expresado en la notación de UML, donde se muestran los conceptos y relaciones involucrados en el contexto de la selección de herramientas de pruebas de Funcionalidad y pruebas de Seguridad, con el fin de sentar las bases para la formulación de un modelo de calidad robusto y operacionalizado para estimar la calidad de estas herramientas. De esta manera, se logra una terminología común entre todos los involucrados, facilitando así la comunicación y una mejor comprensión del problema tratado.

Contexto de la investigación.

Un enfoque utilizado para este proceso es DevOPs; una mejora de este se le conoce como DevSecOps y es clave para un desarrollo de calidad.[2,3,4]

Pressman[5] y las Normas ISO 25010,[6] señalan la importancia de cumplir los requisitos explícitos e implícitos y satisfacer las necesidades del cliente. Los estándares ISO 25010 y el 25059, ofrecen modelos y guías para garantizar la calidad.

Las herramientas de desarrollo de software ayudan a garantizar la calidad durante todo el ciclo de vida del software,[7,4,8] porque apoyan al desarrollador en la realización de sus tareas de una manera automatizada, minimizando el riesgo de equivocaciones. Una de las actividades del proceso de desarrollo, son las pruebas de software, esta actividad busca prevenir los errores, reducir costos y mejorar el rendimiento del desarrollo de software.[9,10,11] Epitech[12] divide los tipos de pruebas en funcionales y no funcionales.

En otro orden de ideas, la aplicación de la Inteligencia Artificial (IA) se utiliza para automatizar y reducir la cantidad de tareas en el desarrollo y en particular, en la actividad de pruebas.[13]

Sin embargo, no está claro cómo estimar la calidad de este tipo de herramientas. Esto lleva a la siguiente pregunta, ¿Cómo se puede estimar la calidad de las herramientas de prueba habilitadas con IA?.

El foco es proponer un modelo de dominio que represente los conceptos y sus respectivas relaciones que cubren lo relacionado con la calidad del software, el proceso de desarrollo, el enfoque DevSecOps y los estándares de ISO 25010 25059, usando herramientas habilitadas por IA para realizar pruebas funcionales. La meta es contar con una herramienta que facilite la colaboración entre todas las partes interesadas, garantizando un entendimiento compartido del dominio de la investigación en curso.

Modelo de dominio

Su desarrollo será por subdominios y utilizando la notación de UML para Diagramas de Clases. Cada clase representa un concepto definido. Los subdominios son: DevSecOps, Herramientas de Pruebas y Modelo de Calidad.

DevSecOps

Es un tipo de enfoque para el desarrollo de software es DevOps, según Kim[2] DevOps consiste en un conjunto de prácticas que automatizan, integran y aceleran los procesos de desarrollo, prueba e implementación, permitiendo la entrega rápida y confiable de software de alta calidad. Microsoft[14] resalta la colaboración entre los equipos de

Revista de Ingeniería y Tecnología Educativa (RITE) Vol 7 N 2 Julio - Diciembre 2024, ISSN: 2665-0339
Ruta Hacia la Comprensión de la Calidad... pg. 94 - 104
Giulianna Bortone, et al.

desarrollo y operaciones.

Este enfoque evolucionó con como DevSecOps. Para Software Engineering Institute[3] y AWS,[4] este enfoque, integra la seguridad en todos los aspectos del proceso. El objetivo es solucionar los problemas de seguridad desde el principio del proyecto.

Para Veritis,[15] DevSecOps se compone de ocho etapas principales. En la fase de planificación, desarrollo, construcción, prueba, liberación, despliegue, operación, y monitoreo La Figura 1 muestra este subdominio.

La Figura 1 identifica cómo DevSecOps integra la seguridad en todas las etapas del desarrollo de software, desde la planificación hasta la operación y el monitoreo.

Nótese la relación estrecha entre DevOps y DevSecOps, donde el primero se enfoca en la automatización, integración y aceleración de procesos, y el segundo añade el elemento de seguridad en todas sus etapas.

El diagrama también resalta la participación de las herramientas de software en el proceso de desarrollo, que toman además un rol importante en el enfoque DevSecOps, pues tiene entre sus principios la integración y automatización del proceso.

Herramientas de Prueba

Las organizaciones exitosas en DevSecOps, según AWS,[4] tienen una cultura de seguridad sólida integrada en sus procesos de desarrollo desde las etapas iniciales y no se limita únicamente a herramientas y tecnología, sino que también implica un cambio cultural y de procesos. Durante estas etapas se emplean prácticas fundamentadas en la automatización, la cual exige el uso de herramientas de desarrollo de software.

El Instituto San Ignacio de Loyola (ISIL),[7] Amazon Web Services (AWS) 4 y Margaret Rouse,[8] coinciden en que las herramientas de desarrollo de software son programas o utilidades que asisten a los desarrolladores en sus tareas, facilitando y optimizando el proceso de desarrollo.
Entre ellas resaltan las herramientas de prueba IBM,[9] la Universidad Internacional de La Rioja,[10] Jain,[11] y Epitech[12] coinciden en que estas son esenciales para garantizar la funcionalidad y calidad del software.

Estos autores definen las pruebas de software como el proceso de evaluar y verificar que un producto o aplicación de software cumple con su propósito, destacando su papel en la prevención de errores, la reducción de costos y la mejora del rendimiento del desarrollo del software.

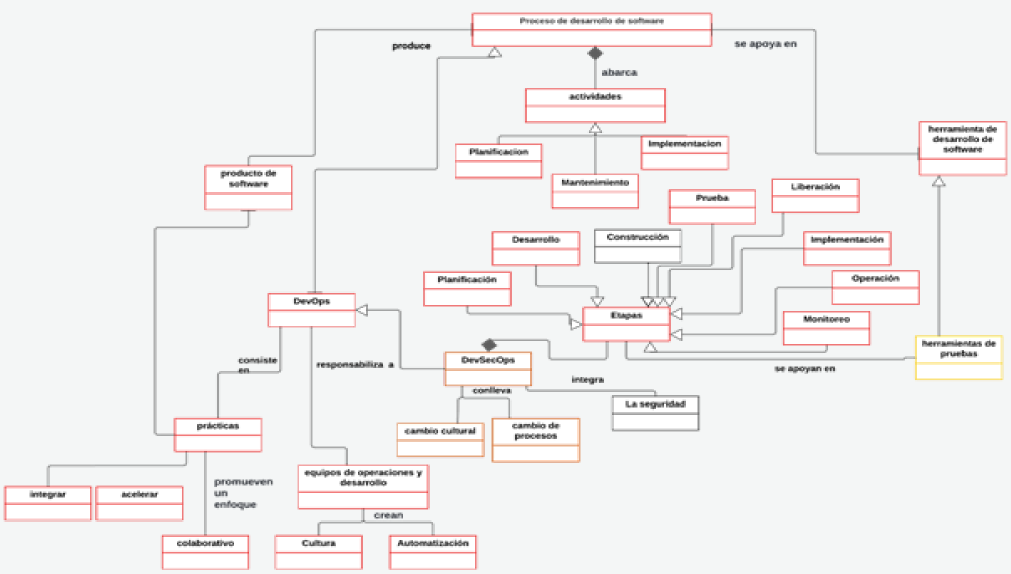

Figura 1. Subdominio DevSecOps del modelo de calidad.
Fuente: Elaboración propia.

Revista de Ingeniería y Tecnología Educativa (RITE) Vol 7 N 2 Julio - Diciembre 2024, ISSN: 2665-0339

Estas herramientas implementan diferentes tipos de pruebas. En las pruebas funcionales destacan: las de integración, de humo, de regresión y de aceptación. Mientras que las pruebas no funcionales evalúan aspectos como el rendimiento, la seguridad y la escalabilidad. Entre sus tipos de pruebas, se destacan las de estrés, usabilidad, pruebas de rendimiento, carga, y seguridad.

El objetivo de las pruebas de seguridad es identificar brechas o ataques inesperados al sistema que podrían resultar en la pérdida de datos para la organización o compañía.[16]

Iberdrola[17] y Katalon[18] enfatizan el impulso y el constante cambio y optimización de las herramientas de software, especialmente con la incorporación de versiones habilitadas con Inteligencia Artificial (IA).

La IA ha mejorado drásticamente estas técnicas utilizadas por las herramientas de prueba; con la IA, estas herramientas no solo ejecutan pruebas automatizadas, sino que también analizan datos inteligentemente, identifican patrones complejos y predicen posibles problemas, permitiendo a los probadores centrarse en tareas más estratégicas.[18] La Figura 2 muestra este subdominio desarrollado con UML, según la sintaxis del Diagrama de clases.

En este subdominio aparece de nuevo el concepto herramientas de pruebas software. Igualmente se incorpora el concepto de IA, ahora cada vez más presente. Dentro de los tipos de pruebas representados se hace énfasis en los de Seguridad, dada su estrecha vinculación con DevSecOps.

Modelo de Calidad

Pressman[5] se enfoca en la importancia de cumplir con los requisitos establecidos y satisfacer las necesidades del usuario. Un proceso de software eficaz se caracteriza por crear un producto que no solo sea útil, sino que también proporcione un valor medible tanto para los productores como para los usuarios.

Este enfoque pone énfasis en la importancia de abordar tanto los requisitos explícitos como los implícitos para garantizar que el software sea realmente valioso y efectivo.

Una guía para estimar la calidad de un producto de software es los estándares específicos, tal como SQuaRE (Requisitos y Evaluación de la Calidad del Software y los Sistemas), en particular, el ISO 25010 y el 25059;[7] el ISO 25010 cubre nueve características de calidad a cumplir por un producto software.

Natale[19] explica que el estándar ISO/IEC 25059[20] es una extensión del ISO/IEC 25010,[6] ofreciendo características de calidad específicas para sistemas de IA.

El ISO 25010, se centra en características: adecuación funcional, eficiencia del desempeño, compatibilidad, usabilidad, capacidad de interacción, fiabilidad, seguridad, mantenibilidad, flexibilidad y protección.

Por su parte, el ISO/IEC 25059 incorpora las características específicas: robustez, transparencia, adaptabilidad funcional, controlabilidad del usuario e interverbilidad.

La figura 3 muestra este subdominio. Para este subdominio la integración entre los subdominios se logra a través del concepto IA, el cual se contempla en la extensión del ISO25010. Otro concepto con un rol integrador es el de calidad del software, que enlaza el subdominio DevSecOps con los estándares.

Reflexión sobre el Impacto del Modelo en el Contexto de la Investigación

Herramientas de pruebas funcionales y de seguridad

En esta investigación además se exploró el panorama de las herramientas de soporte para pruebas funcionales y de seguridad en el software.

Ante la abundancia de opciones en el mercado, fue oportuna una revisión sistemática de las mismas, con el fin de identificar algunas de las características resaltantes para identificar tendencia de

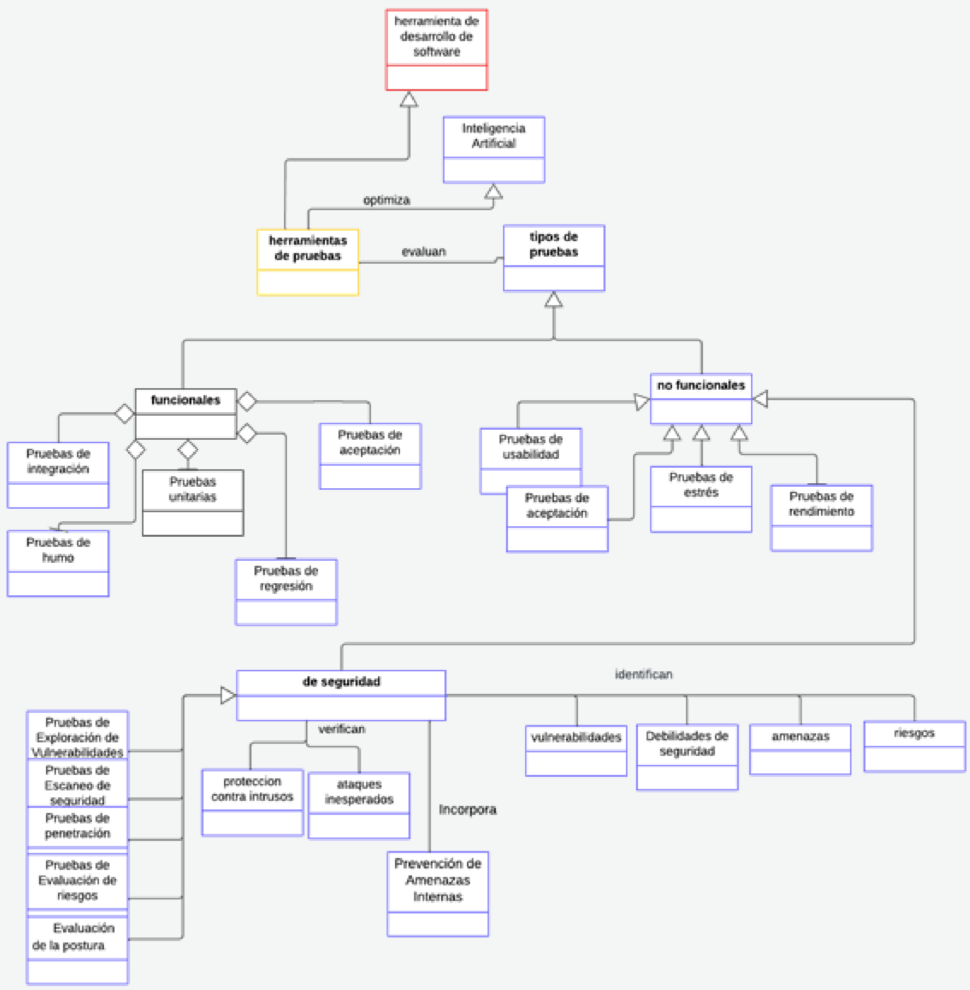

Figura 2. Subdominio Herramientas de Prueba del modelo de calidad.
Fuente: Elaboración propia.

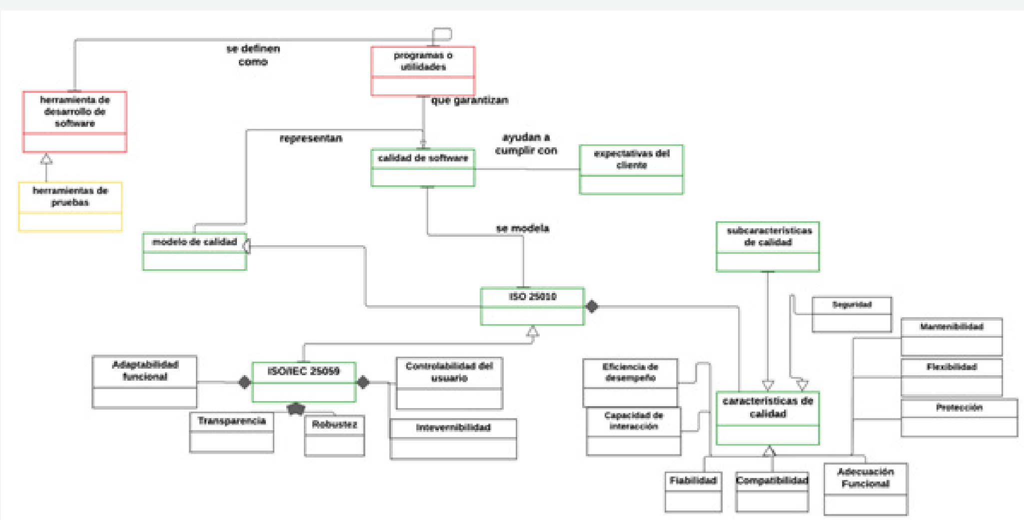

Figura 3. Subdominio Modelo de calidad.
Fuente: Elaboración propia.

Revista de Ingeniería y Tecnología Educativa (RITE) Vol 7 N 2 Julio - Diciembre 2024, ISSN: 2665-0339

las funcionalidades presentes en ellas. Esto propicia los objetivos de la investigación.[21]

Esta revisión se enfoca particularmente en herramientas que incorporan IA. Por ello se propone una tabla comparativa. Ver Tabla 1 Por cada herramienta de pruebas de software se precisa:

Herramienta: Nombre de la herramienta de prueba; Tipo de pruebas: Funcionales: Buscan proporcionar evidencia convincente de que el sistema es apto para su propósito.[22]

No funcionales, tales como:

De rendimiento: Prueba que el software funcione use eficientemente sus recursos.[22]

De carga: Prueban que cubra la demanda esperada.[23]

De usabilidad: Miden la facilidad de uso, efectividad y satisfacción.[24]

De estrés: Someter a la aplicación a una carga mucho mayor.[23]

De API: Analiza la interfaz de programa de aplicación (API), para verificar que cumple con lo esperado.[25]

Pentesting (Penetración): Se intenta rastrear e identificar las vulnerabilidades.[26]

End to end: prueba flujos de ejecución de comienzo a fin.[27]

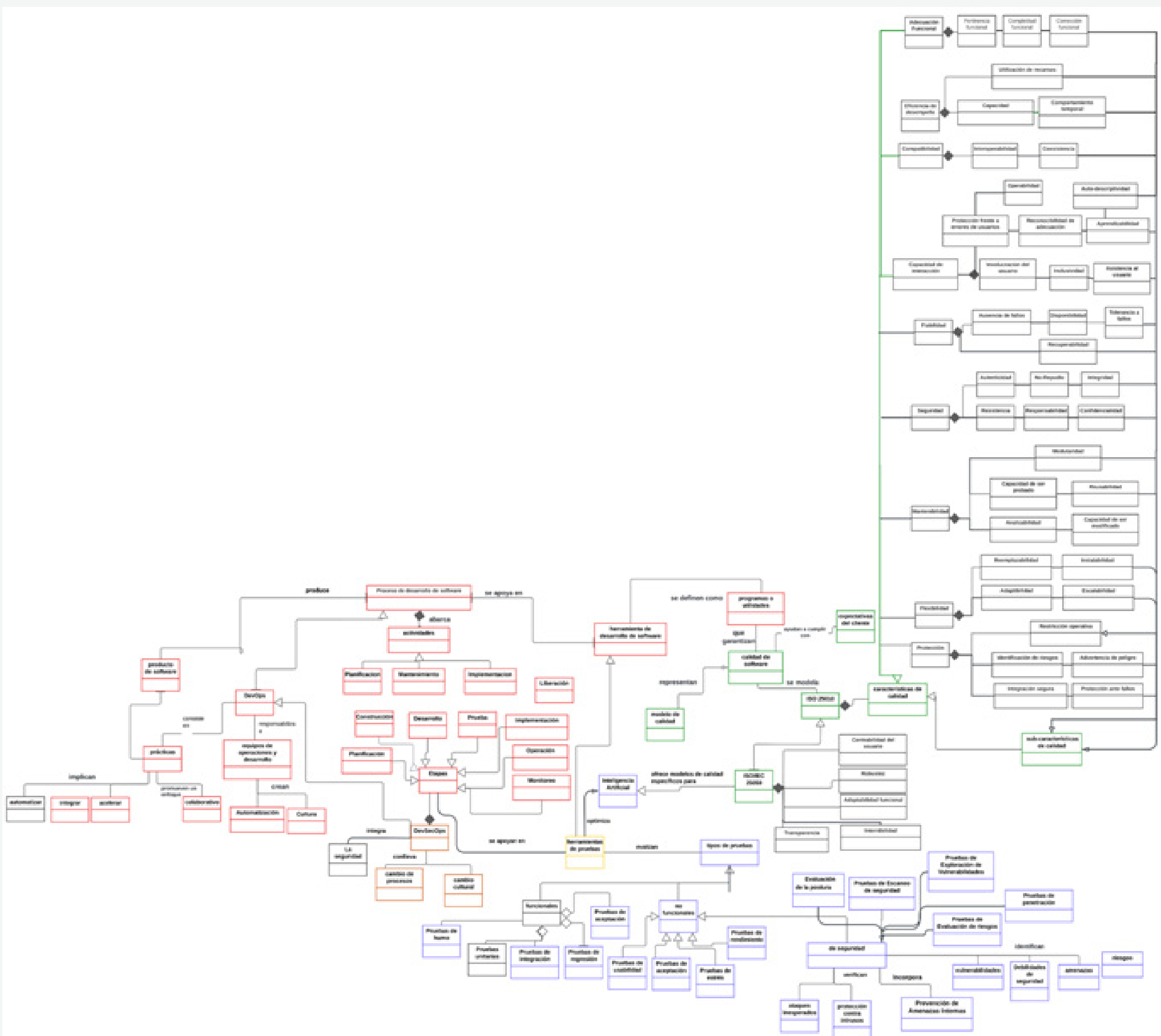

Figura 4. Modelo de dominio.
Fuente: Elaboración propia.

Revista de Ingeniería y Tecnología Educativa (RITE) Vol 7 N 2 Julio - Diciembre 2024, ISSN: 2665-0339

De seguridad: Buscan vulnerabilidades.[22]

Fuzzing: De técnica de caja negra, que busca errores de implementación mediante la inyección de datos malformados/semi-malformados.[28]

Visuales: Comparan capturas de pantalla para detectar cambios no deseados en la interfaz de usuario.

Esta tabla es base para los próximos pasos a seguir. En ella se nota una presencia importante de herramientas de pruebas de código abierto, con comunidades significativas (grandes o muy grandes), y bastante activas, los lenguajes predominantes son Java y Python. La tabla permitirá la inclusión de nuevos criterios de comparación en la medida que la investigación avance, así como agregar nuevas herramientas. Finalmente, será el punto de partida para una reflexión sobre sus características y las propuestas por ISO 25010 y/o ISO 25059.

El modelo de dominio propuesto, permite visualizar todos los aspectos (conceptos representados por clases) a ser contemplados en la investigación para una clara comprensión de su complejidad. Ver figura 4; aprovechando las bondades de la sintaxis se pudo precisar lo siguiente: el concepto herramienta de pruebas, es el concepto integrador de los tres subdominios, obsérvese que se identifican cinco relaciones entre ella y otros conceptos: etapas (de DevOps), herramientas de desarrollo de software, tipos de pruebas, IA y seguridad.

Este es el foco central de la investigación en progreso que busca estimar la calidad de este tipo de herramientas de prueba.

Tabla 1. Herramientas de Pruebas de Software.

Herramienta	Tipo de pruebas	Código	IA	Comunidad	Soporte técnico	Lenguaje
Apache JMeter	Carga, rendimiento	Abierto	-	Muy grande	Comunidad activa	Java
Appium	Automatización móvil (Funcional)	Abierto	-	Grande	Comunidad activa	Java, JavaScript, Python, Ruby
Selenium	Funcionales, UI	Abierto	-	Muy grande	Comunidad activa	Java, Python, CSharp, Ruby, JavaScript, Kotlin
Insomnia	API	Abierto	-	Grande	Bueno	No aplica
Jasmine	Funcionales	Abierto	-	Grande	Comunidad activa	JavaScript
Jest	Funcionales	Abierto	-	Grande	Bueno	JavaScript
Cypress	End-to-end	Abierto	-	Grande	Bueno	JavaScript
Enzyme	Testing de componentes de React (Funcionales)	Abierto	-	Grande	Bueno	JavaScript
JUnit 5	Funcionales	Abierto	-	Muy grande	Comunidad activa	Java
Karma	Test runner (Funcionales)	Abierto	-	Grande	Comunidad activa	JavaScript
Katalon Studio	Funcionales, API, móviles	Gratuito	-	Grande	Bueno	JavaScript

LoadNinja	Carga, rendimiento	Privativo	-	Pequeña	Bueno	Java
Mockito	Mocking (Funcionales)	Abierto	-	Grande	Comunidad activa	Java
Functionize	Automatización visual (Funcionales)	Privativo	Sí	Pequeña	Bueno	No aplica
Perfecto	Automatización móvil, web (Funcionales)	Privativo	-	Pequeña	Bueno	Java C# (iOS/Android), Python, Ruby JavaScript Integración con Selenium o Appium
Eggplant	Funcionales, UI	Privativo	-	Pequeña	Bueno	No aplica
Testim	Automatización visual (Funcionales)	Privativo	Sí	Pequeña	Bueno	JavaScript, TypeScript
Leapwork	Automatización visual, sin código (Funcionales)	Privativo	-	Pequeña	Bueno	No aplica
Mabl	End-to-end	Privativo	Sí	Pequeña	Bueno	No aplica
Netsparker	Pentesting web	Privativo	-	Pequeña	Bueno	No aplica
OpenVAS	Pentesting	Abierto	-	Grande	Comunidad activa	Python, C
Nessus	Pentesting	Privativo	-	Pequeña	Bueno	No aplica
Acunetix	Pentesting web	Privativo	-	Pequeña	Bueno	No aplica
Retina	Pentesting	Privativo	-	Pequeña	Bueno	No aplica
Probely	Pentesting web	Privativo	-	Pequeña	Bueno	No aplica
ZAP	Pentesting web	Abierto	-	Grande	Comunidad activa	Java
Wfuzz	Fuzzing	Abierto	-	Grande	Comunidad activa	Python
Sqlmap	Inyección SQL	Abierto	-	Grande	Comunidad activa	Python
Metasploit	Pentesting	Abierto	-	Muy grande	Comunidad activa	Ruby
Applitools	Visuales	Privativo	Sí	Pequeña	Bueno	JavaScript, Java, C#, Python, Rub

Fuente: Propia.

Revista de Ingeniería y Tecnología Educativa (RITE) Vol 7 N 2 Julio - Diciembre 2024, ISSN: 2665-0339

Conclusiones

Las herramientas de prueba aseguran la calidad, funcionalidad y no funcional de las aplicaciones. En este trabajo, se presenta un modelo de dominio utilizando UML, que representa las relaciones claves entre conceptos como DevSecOps, herramientas de prueba y estándares de calidad, proporcionando una comprensión compartida de la complejidad del tema tratado; y una base estructurada la investigación en progreso de la formulación de un modelo de calidad para estas herramientas.

El modelo se centra en tres subdominios: DevSecOps, herramientas de prueba y modelos de calidad. La integración de la inteligencia artificial en estas herramientas representa un avance significativo, pero también plantea nuevos desafíos en la medición de su calidad.

Este trabajo sienta las bases para la creación de un lenguaje común entre los profesionales involucrados, facilitando la colaboración y la comprensión mutua entre los involucrados.

Recomendaciones

La investigación futura debe centrarse en la validación y refinamiento del modelo de dominio propuesto, hasta refinarlo en un modelo de calidad; este modelo se debe operacionalizar con métricas específicas que permitan su uso para estimar la calidad de las herramientas de pruebas habilitadas con IA

Referencias

1.- QAlified. Las 10 Mejores Herramientas de Testing de Software del 2024. QAlified [Internet]. 2024 [citado 12-07-2024]. Disponible en: https://qalified.com/es/blog/software-qa-testing-herramientas/.

2.- Kim G, Behr K, Spafford G. The Phoenix Project. [Internet]. 2013 [citado 13-07-2024]. Disponible en: https://www.haio.ir/app/uploads/2021/12/The-Phoenix-Project-A-Novel-about-IT-DevOps-and-Helping-Your-Business-Win-by-Gene-Kim-George-Spafford-Kevin-Behr-z-lib.org_.pdf.

3.- Software Engineering Institute. DevSecOps | Software Engineering Institute. Software Engineering Institute [Internet]. 2024 [citado 14-07-2024]. Disponible en: https://www.sei.cmu.edu/our-work/devsecops/.

4.- Amazon Web Services. ¿Qué son las herramientas para desarrolladores?: Explicación sobre las herramientas para desarrolladores: AWS. Amazon AWS [Internet]. s.f. [citado 15-07-2024]. Disponible en: https://aws.amazon.com/es/what-is/developer-tools/.

5.- Pressman R. Ingeniería del software. Un enfoque práctico. 9th ed. McGrawHill [Internet]. 2020 [citado 16-07-2024].

6.- Normas ISO Org. ISO 25010: Mejora calidad y satisfacción del usuario en software. Normas ISO [Internet]. 2024 [citado 17-07-2024]. Disponible en: https://normasiso.org/norma-iso-25010/.

7.- Instituto San Ignacio de Loyola. Inicio Tecnología Las Mejores Herramientas en el Desarrollo de Software. ISIL [Internet]. s.f. [citado 18-07-2024]. Disponible en: https://isil.pe/blog/tecnologia/mejores-herramientas-desarrollo-software/.

Revista de Ingeniería y Tecnología Educativa (RITE) Vol 7 N 2 Julio - Diciembre 2024, ISSN: 2665-0339

8.- Rouse M. What is a Programming Tool? - Definition from Techopedia. Techopedia [Internet]. 2020 [citado 19-07-2024]. Disponible en: https://www.techopedia.com/definition/8996/programming-tool.

9.- IBM. ¿Qué es DevSecOps? IBM [Internet]. s.f. [citado 20-07-2024]. Disponible en: https://www.ibm.com/es-es/topics/devsecops.

10.- Universidad Internacional de La Rioja. Pruebas de software: Tipos e importancia. UNIR México [Internet]. 2023 [citado 21-07-2024]. Disponible en: https://mexico.unir.net/noticias/ingenieria/pruebas-software/.

11.- Jain S. Software Testing Tools. GeeksforGeeks [Internet]. 2024 [citado 22-07-2024]. Disponible en: https://www.geeksforgeeks.org/software-testing-tools/.

12.- Epitech. Testing software: Qué son las pruebas de software y cómo funcionan. Epitech [Internet]. 2023 [citado 23-07-2024]. Disponible en: https://www.epitech-it.es/testing-software-pruebas-software/.

13.- Kirilenko Í. Inteligencia artificial en pruebas de software y automatización de API. Parasoft [Internet]. 2022 [citado 24-07-2024]. Disponible en: https://es.parasoft.com/blog/what-is-artificial-intelligence-in-software-testing/.

14.- Microsoft. What Is DevSecOps? Definition and Best Practices. Microsoft [Internet]. 2024 [citado 25-07-2024]. Disponible en: https://www.microsoft.com/en-us/security/business/security-101/what-is-devsecops#devsecops-components.

15.- Veritis. What are the Phases of Devsecops - Challenges & Best Practices. Veritis [Internet]. s.f. [citado 26-07-2024]. Disponible en: https://www.veritis.com/blog/what-are-the-phases-of-devsecops/#03.

16.- Joshi V. Introducción a las pruebas de seguridad. Cynoteck [Internet]. 2020 [citado 27-07-2024]. Disponible en: https://cynoteck.com/es/blog-post/introduction-to-security-testing/.

17.- Iberdrola. Algoritmos de IA. Iberdrola [Internet]. s.f. [citado 28-07-2024]. Disponible en: https://www.iberdrola.com/innovacion/algoritmos-ia.

18.- Katalon. What is AI Testing? The Future of Software Testing. Katalon [Internet]. s.f. [citado 29-07-2024]. Disponible en: https://katalon.com/resources-center/blog/ai-testing.

19.- Natale D. Extensions of ISO/IEC 25000 Quality Models to the Context of Artificial Intelligence. CEUR-WS.org [Internet]. 2022 [citado 30-07-2024]. Disponible en: https://ceur-ws.org/Vol-3356/paper-02.pdf.

20.- ISO/IEC 25059:2023 - Software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Quality model for AI systems. ISO [Internet]. 2023 [citado 31-07-2024]. Disponible en: https://www.iso.org/standard/80655.html.

21.- Zap Chernyak A. Las 30 mejores herramientas para pruebas de software en 2024 (gratuitas + para empresas). zaptest [Internet]. 2024 [citado 01-08-2024]. Disponible en: https://www.zaptest.com/es/herramientas-para-pruebas-de-software-los-30-mejores-productos-para-pruebas-de-software-del-mercado-en-2024.

Revista de Ingeniería y Tecnología Educativa (RITE) Vol 7 N 2 Julio - Diciembre 2024, ISSN: 2665-0339

22.- Sommerville I. Engineering Software Products: An Introduction to Modern Software Engineering. Pearson [Internet]. 2020 [citado 02-08-2024].

23.- Campos Chiu C. LAS PRUEBAS EN EL DESARROLLO DE SOFTWARE. UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO [Internet]. 2015 [citado 03-08-2024]. Disponible en: Las_pruebas_en_el_desarrollo_de_software-libre.pdf?1512259021=&response-content-disposition=inline%3B+filename%3DUNIVERSIDAD_NACIONAL_AUTONOMA_DE_MEXICO.pdf&Expires=1724078456&Signature= gz7bopQ2hKYaMpTUeoeKK2umNvrt~dKf2sl6E4niz20BXdzTjLK1MR4~MxiVtO6weuDMYV

24.- Sánchez Peño JM. Pruebas de Software. Fundamentos y Técnicas. Archivo Digital UPM [Internet]. 2015 [citado 04-08-2024]. Disponible en: https://oa.upm.es/40012/1/PFC_JOSE_MANUEL_SANCHEZ_PENO_3.pdf.

25.- Llanos Falen C. ¿Qué son las pruebas de API? ¿Qué implica? LinkedIn [Internet]. 2023 [citado 05-08-2024]. Disponible en: https://www.linkedin.com/pulse/qu%C3%A9-son-las-pruebas-de-api-implica-carla-llanos-falen/.

26.- Bernal Ontiveros JM, Bailón Estrada M, Flores Regalado A, Benítez Linares Vásquez M, Escobar Velásquez C. PRUEBAS AUTOMÁTICAS DE SOFTWARE. Universidad de los Andes [Internet]. 2020 [citado 06-08-2024]. Disponible en: https://miso-4208-labs.gitlab.io/book/.

27.- Linares Vásquez M, Escobar Velásquez C. PRUEBAS AUTOMÁTICAS DE SOFTWARE. Universidad de los Andes [Internet]. 2020 [citado 08-08-2024]. Disponible en: https://miso-4208-labs.gitlab.io/book/.

28.- OWASP. Fuzzing. OWASP Foundation [Internet]. s.f. [citado 07-08-2024]. Disponible en: https://owasp.org/www-community/Fuzzing

Revista de Ingeniería y Tecnología Educativa (RITE) Vol 7 N 2 Julio - Diciembre 2024, ISSN: 2665-0339