Revista Cubana de Ingeniería Vol.XV(3) e395 (2024) ISSN: 2223-1781

Artículo Original

Proceso de Integración Continua para el sistema AsiXmec

Continuous Integration Process for the AsiXmec system

Yanays Fernández Miranda[1*], Héctor Jesús Unzueta Lazo[2], Ángel Alberto Vázquez Sánchez[3]

[1]Universidad de las Ciencias Informáticas. Carretera a San Antonio de los Baños Km 21/2, Rpto. Torrens, La Habana.
[2]Universidad de las Ciencias Informáticas. Carretera a San Antonio de los Baños Km 21/2, Rpto. Torrens, La Habana.
[3]Universidad de las Ciencias Informáticas. Carretera a San Antonio de los Baños Km 21/2, Rpto. Torrens, La Habana.
*Autor de correspondencia: yfernandez@uci.cu

Resumen

La Integración Continua es una práctica que aporta mejora de productividad y calidad en un proyecto de software si se aplica adecuadamente. Este método permite que todo el equipo que produce código pueda recibir retroalimentación de manera más rápida, y a su vez resolver inmediatamente los problemas introducidos en el repositorio de código del proyecto. AsiXmec es un Sistema de Diseño Asistido por Computadora, que se encuentra siendo desarrollado en la Universidad de Ciencias Informáticas. Este proyecto es de gran envergadura y cuenta con diferentes desarrolladores, pero no tiene incorporada una Integración Continua idónea para este contexto, lo cual está afectando a la calidad del producto. Debido a esta problemática, el objetivo de este trabajo es desarrollar un proceso de Integración Continua para AsiXmec que contribuya al aseguramiento de la calidad del sistema. Se propuso utilizar el marco de pruebas unitarias y de interfaz de usuario Qt test, la herramienta de análisis estático de código CPPCheck, Git para el control de versiones y Jenkins para automatizar la integración de todo el desarrollo. Para la solución, se diseñaron fases de la implementación del proceso en el sistema, incluyendo desde la preparación del equipo y del entorno, hasta la planificación y ejecución de pruebas, la evaluación de los resultados y la retroalimentación del equipo. Para validar este proceso se implementó el método experimental validación comparativa, evaluando el impacto de la Integración Continua en la calidad del proceso de desarrollo del proyecto.

Palabras clave: AsiXmec, calidad de software, Integración Continua.

Abstract

Continuous Integration is a practice that enhances productivity and quality in a software project when applied correctly. This method allows the entire team that produces code to receive feedback more quickly, and in turn, resolve issues introduced in the project's code repository immediately. AsiXmec is a Computer-Aided Design System currently being developed at the University of Computer Sciences. This project is substantial and involves various developers, but it lacks an appropriate Continuous Integration framework for this context, which is affecting the product's quality. Due to this issue, the objective of this work is to develop a Continuous Integration process for AsiXmec that contributes to ensuring the quality of the system. It was proposed to use the Qt test framework for unit and user interface testing, the static code analysis tool CPPCheck, Git for version control, and Jenkins to automate the integration of the entire development. For the solution, phases were designed for implementing the process in the system, ranging from preparing the team

and the environment to the planning and execution of tests, evaluating results, and providing feedback to the team. To validate this process, a comparative validation experimental method was implemented to assess the impact of Continuous Integration on the quality of the project's development process.

Keywords: AsiXmec, Continuous Integration, software quality.

1. Introducción

El Diseño Asistido por Computadora (CAD, por sus siglas en inglés) se usa en casi todas las industrias, en proyectos tan variados como el diseño de paisajes, la construcción de puentes, el diseño de edificios de oficinas, la animación en películas y el diseño de piezas para la industria [1]. Según [2], con los programas de diseño CAD 2D o 3D, se pueden realizar diferentes tareas, como crear un modelo 3D de un diseño, aplicar materiales y efectos de iluminación, documentar el diseño de cotas y otras anotaciones. La presente investigación se enmarca en software CAD empleados para modelar piezas de la industria para ser impresas en 2D y 3D. En [3] se plantea que internacionalmente estos softwares se dividen por nivel de experiencia en principiante, nivel intermedio y profesional, siendo estos últimos extremadamente costosos con precios que oscilan entre los 3000 y 10000 euros por licencia anual.

CREO es uno de los softwares líder del mercado de diseño de productos, desarrollado por la empresa Parametric Technology Corporation. Además, está el software SolidWorks, desarrollado por la empresa francesa Dassaul Systemes, el cual es conocido por ser práctico y detallado en las piezas industriales que diseña [3]. Otra de las herramientas profesionales ideales para el diseño, simulación, visualización y documentación de productos es Autodesk Inventor, desarrollado en California, Estados Unidos. Las versiones de Autodesk Inventor tienen un costo de 2190 USD por suscripción anual, comercializadas solo para países con convenio [4]. Países de América Latina como México, Brasil y Argentina utilizan Autodesk. Debido al alto costo de estos productos, Cuba necesita su propia herramienta de diseño paramétrico para facilitar la confección de piezas para la industria.

En la Universidad de Ciencias Informáticas (UCI), se desarrolla el sistema CAD nombrado AsiXmec como parte del proyecto de Investigación y Desarrollo “Plataforma para el modelado paramétrico en ingeniería basado en tecnologías de código abierto”. Este proyecto, aprobado por el Programa Nacional de Automática, Robótica e Inteligencia Artificial, incorpora un modelador geométrico y paramétrico, basado en el paradigma de diseño mediante características e historial de operaciones. AsiXmec tiene el propósito de acortar el ciclo de diseño, en un área caracterizada por una elevada competitividad. Para lograr esto, según [5], se captura tanto la geometría del diseño CAD como la intención del diseñador, utilizando conceptos conocidos como modelado paramétrico y diseño basado en restricciones geométricas.

El proyecto AsiXmec es un sistema multiplataforma en el que colabora un equipo de diferentes desarrolladores, lo que implica un ritmo de trabajo adaptable y receptivo a cambios. Esto requiere la capacidad de adaptarse rápidamente a las modificaciones del proyecto. Con la evolución y la complejidad del desarrollo del sistema, se vuelve más desafiante garantizar la calidad del producto final. En AsiXmec se han logrado avances significativos en el desarrollo de funcionalidades esenciales para el diseño y visualización de prototipos virtuales de piezas mecánicas. Sin embargo, existen limitaciones que obstaculizan su progreso. Entre ellas se encuentran:

- No hay un proceso que permita el seguimiento de los cambios en el código: no se rastrea de forma sistemática los cambios realizados en el código, lo que compromete la integridad del

código de toda la aplicación, ya que es difícil mantener un registro de quién realizó cambios y cuándo.

- No están definidas las pruebas que permitan que se detecten los errores lo antes posible: debido a que, no se han establecido pruebas automatizadas que permitan detectar los errores lo más temprano posible. Esto aumenta el riesgo del proyecto, ya que los errores pueden pasar desapercibidos hasta fases avanzadas del desarrollo.
- Incertidumbre sobre qué funciona y qué no: no se tiene total conocimiento de qué partes del sistema están funcionando correctamente y cuáles presentan fallas. Esto genera incertidumbre y afecta la toma de decisiones en el proyecto.
- Afectación del mantenimiento del software, por la integración de partes del sistema desconociendo las fallas en sus unidades: existe el riesgo de que partes del sistema con fallas o errores se integren sin ser detectados. Esto puede afectar el mantenimiento del software a largo plazo, ya que dichas fallas pueden pasar desapercibidas y ocasionar problemas en etapas posteriores.

La Integración Continua (CI, por sus siglas en inglés) es una práctica de desarrollo de software en la cual los miembros de un equipo integran su trabajo frecuentemente. Este proceso tiene el propósito de mejorar la calidad de la aplicación y evitar posibles errores o anomalías generados en el desarrollo del sistema [6]. La CI incluye prácticas recomendadas como son las pruebas de software y el control de versiones lo que permite identificar errores en el sistema en etapas tempranas [7]. Ante las problemáticas anteriormente abordadas se deduce que en AsiXmec no se encuentra implementado el proceso CI de la forma correcta y esto afecta a la calidad del software.

Debido a la problemática existente en el proyecto AsiXmec, se elabora la presente investigación con el objetivo de desarrollar un proceso de Integración Continua para el sistema AsiXmec, que permita contribuir a la calidad del software mediante la detección de errores, el monitoreo de cambios en el código y la prevención de integración de unidades con fallas.

2. Materiales y Métodos

Para el desarrollo de la presente investigación se realizó un análisis bibliográfico sobre el proceso de Integración Continua, con énfasis en las buenas prácticas de esta actividad. Se utilizaron métodos empíricos como la observación, lo cual facilitó la determinación de la problemática analizada mediante su percepción directa. Se tuvo en cuenta que los desarrolladores están familiarizados con la metodología ágil XP, lo que facilita la implementación exitosa de la Integración Continua al proporcionar un marco de trabajo para el desarrollo del proyecto. Se determinó utilizar la herramienta Qt test versión 5.0.2 para las pruebas unitarias y de interfaz de usuario, debido a que el sistema AsiXmec está desarrollado en el entorno multiplataforma QtCreator, empleando lenguaje Qt y C++. Para el análisis estático de código se empleó CPPCHECK en su versión 2.6.1. El repositorio de código del proyecto se encuentra en Gitlab, por lo cual se empleó Git en su versión 2.35.1 para el control de versiones.

La selección de herramientas para pruebas de software, como el análisis estático de código y las pruebas unitarias, se realiza en función del lenguaje de programación y el entorno de desarrollo empleados en la implementación del sistema. No obstante, la elección del servidor de Integración Continua (CI) se basa en otras características específicas del proyecto. Por esta razón, se evaluaron diversas herramientas de automatización CI según las necesidades particulares del proyecto AsiXmec. Como resultado, se determinó que Jenkins en su versión 2.387.3 es la herramienta gratuita más adecuada para establecer el entorno CI necesario en AsiXmec (Tabla 1 y Tabla 2). Para llevar a cabo esta evaluación y facilitar la selección formal del servidor CI, se utilizó la norma ISO 25010 (Figura 1).

De las ocho características presentadas en la Figura 1, se consideraron en esta investigación la adecuación funcional, la usabilidad y la mantenibilidad. En función de los requerimientos del proyecto AsiXmec, se seleccionaron las siguientes subcaracterísticas [8]:

Completitud funcional: Grado en el cual el conjunto de funcionalidades cubre todas las tareas y los objetivos del usuario especificados. Para evaluar el servidor de automatización se divide en:

- Conexión de repositorios.
- Conexión con servidores remotos por ssh.
- Plugins de integración con otras herramientas
- Pipelines y Jobs.

Capacidad de aprendizaje: Capacidad del producto que permite al usuario aprender su aplicación. Este criterio se aborda para el servidor CI de la siguiente manera:

- Documentación.
- Comunidad
- Soporte

Reusabilidad: Capacidad de un activo que permite que sea utilizado en más de un sistema de software o en la creación de otros activos.

Figura 1: Características de calidad evaluadas por la ISO/IEC 25010. Fuente: [8].

Tabla 1: Ponderaciones Jenkins CI. Fuente: Elaboración propia

Características	**Sub características**	**Valoración**
Completitud funcional	Conexión con repositorios	9
	Conexión con servidores por ssh	9
	Plugins de integración con otras herramientas	10
	Pipelines y Jobs	10
Capacidad de aprendizaje	Documentación	10
	Comunidad	10
	Soporte	9
Reusabilidad	Capacidad de reusabilidad	10

Tabla 2: Ponderaciones porcentuales Jenkins CI. Fuente: Elaboración propia

Características	Sub características	Valor porcentual
Completitud funcional	Conexión con repositorios	4.5%
	Conexión con servidores por ssh	13,50%
	Plugins de integración con otras herramientas	10%
	Pipelines y Jobs	15%
Capacidad de aprendizaje	Documentación	15%
	Comunidad	15%
	Soporte	13,50%
Reusabilidad	Capacidad de reusabilidad	10%
Total		96.5%

En el script de Jenkins se incluyeron los pasos necesarios para ejecutar las pruebas, haciendo uso de herramientas como CMake y QMake, de QtCreator para gestionar las dependencias y el sistema de construcción, respectivamente. Además, se empleó el método experimental de validación comparativa, lo que permitió comparar el proceso de desarrollo de software en el proyecto AsiXmec antes y después del Proceso de Integración Continua propuesto.

3. Resultados y Discusión

Para llevar a cabo el proceso de Integración Continua (CI) en el proyecto AsiXmec, se elaboró un plan de implementación que sirve como un marco sólido para garantizar el correcto funcionamiento del proceso y promover su mejora continua. Con este propósito, se propuso dividir el proceso de CI en las siguientes fases:

- **Fase 1: Preparación**

Preparación del equipo de trabajo: Se aseguró de que todos los miembros comprendieran sus responsabilidades en el proceso de Integración Continua para lograr una efectiva integración del equipo.
Preparación del entorno: Se gestionó la descripción del entorno seleccionado y la instalación del ambiente necesario.

- **Fase 2: Planificación, Configuración y Ejecución de Pruebas**

En esta fase, se elaboró el proceso de planificación de pruebas, que incluyó diagramas de flujo para las pruebas a realizar, así como su implementación y ejecución en el sistema. Las pruebas consideradas primordiales para AsiXmec son: pruebas de análisis estático de código, pruebas unitarias y pruebas de interfaz de usuario (UI).

Al planificar las pruebas, se estableció como objetivo evaluar la calidad y el funcionamiento del software AsiXmec en un entorno controlado. Se definieron como criterios de entrada la documentación del sistema (que incluye requisitos funcionales, historias de usuario y diseños arquitectónicos) y el código fuente disponible en el repositorio de GitLab. Con respecto a los criterios de salida, se generó un informe detallado de los resultados de las pruebas automatizadas, que incluye errores y vulnerabilidades detectadas, un listado de las pruebas realizadas con sus respectivos resultados, además de un análisis sobre la eficacia y eficiencia del proceso de pruebas automatizadas y su integración en el desarrollo continuo.

- **Fase 3 Evaluación de resultados**

En esta fase, se evaluaron las métricas de calidad cumplidas utilizando la herramienta Jenkins. Se analizaron los resultados obtenidos en las pruebas, evaluando el cumplimiento de los criterios de calidad establecidos.

- **Fase 4 Retroalimentación del equipo.**

En esta fase, se realizó una retroalimentación del equipo en la que se identificaron lecciones aprendidas durante el proceso de implementación de Integración Continua. Se analizaron los puntos fuertes y áreas de mejora con el objetivo de optimizar los procesos futuros y mantener un ciclo de mejora continua.

En la fase 1 se llevó a cabo un estudio sobre los roles establecidos en un proceso de Integración Continua en las investigaciones de [9] [10]. Tras analizar los resultados, se propuso asignar en AsiXmec los siguientes roles y responsabilidades al equipo de desarrollo del proyecto, conforme a las necesidades específicas del mismo, con el fin de implementar el proceso de Integración Continua:

- Desarrollador: es responsable de escribir y mantener el código del software, asegurándose de que el código sea funcional, limpio y esté bien documentado.

- Integrador: es el encargado de coordinar la integración del código nuevo en el repositorio principal, asegurándose que el código se mezcle correctamente y verificando su integridad.

- Automatizador: se encarga del diseño, desarrollo e implementación de soluciones automatizadas, asegurándose de automatizar tareas y procesos repetitivos.

- Tester: avala la calidad del software a través de la ejecución de pruebas, asegurándose de que se cubran todos los casos de prueba necesarios y de reportar cualquier error o problema encontrado.

- Administrador de infraestructura: es responsable de mantener y gestionar los servidores, entornos y herramientas utilizadas en el proceso, asegurándose de que los sistemas estén configurados y disponibles para el equipo en todo momento.

- Gestor de configuración: es el encargado de gestionar el control de versiones del código fuente y garantizar que la configuración del sistema esté debidamente registrada y controlada.

- Agile Coach: se encarga de ayudar en la gestión de cambio y en la adopción de nuevas formas de trabajo ágiles en el proyecto basadas en la metodología XP. Monitorea y mide el rendimiento del equipo utilizando métricas ágiles y proporciona retroalimentación para impulsar la mejora continua.

- Administrador de la Calidad: Se encarga de garantizar que todo el proceso y resultados obtenidos cumplan con los estándares de calidad establecidos para el proyecto. Tiene la responsabilidad de establecer métricas y KPIs (Indicadores Clave de Rendimiento) para medir la calidad del software y del proceso; identificar áreas de mejora y proponer acciones correctivas o preventivas; y realizar auditorías periódicas para verificar el cumplimiento de los estándares de calidad establecidos.

Con el objetivo de participar de manera activa en el proceso, se tomó como medida la capacitación y formación adecuada del equipo en control de versiones con Git y en la automatización de pruebas. Para ello, se propuso la asistencia a cursos de posgrado especializados en este tema, así como a capacitaciones internas. Para fomentar la colaboración e integración del equipo, se propuso organizar reuniones de seguimiento cada dos semanas, donde cada miembro tiene la oportunidad de expresar sus ideas, sugerencias y preocupaciones. Estas reuniones son un espacio seguro para el intercambio de opiniones y el fomento de la comunicación abierta entre todos los miembros del equipo.

El proceso se desarrolló en una máquina Ubuntu 20.04 con QtCreator 5.0.2 como IDE de desarrollo, el entorno CI integrable con Git, Qt test y CppCheck. Inicialmente se desplegó a un servidor local para realizar las pruebas pertinentes en el entorno de desarrollo. El entorno CI de AsiXmec posee el flujo de trabajo representado en la Figura 2.

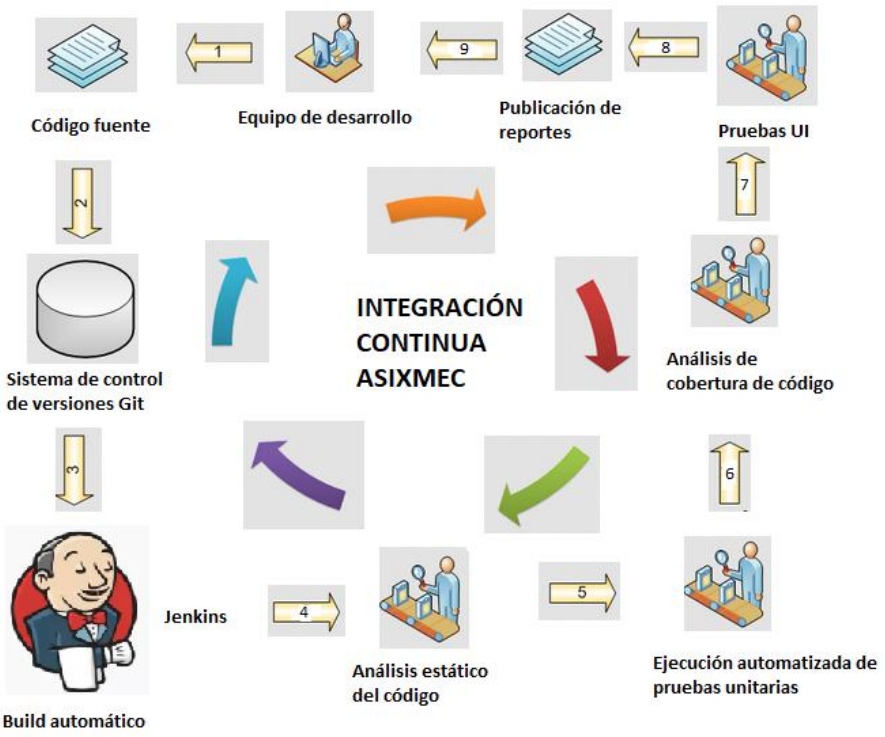

Figura 2: CI AsiXmec. Fuente: Elaboración propia.

Al incorporar el proceso Integración Continua en el desarrollo del sistema, durante la fase 2 y fase 3, Jenkins como servidor CI generó un historial de ejecuciones que permitió visualizar los cambios y el estado del proyecto, logrando una gestión efectiva y eficiente del proceso de desarrollo (Figura 3). Además, se obtuvo la tendencia del tiempo de cada una de las ejecuciones realizadas y se evaluó los resultados obtenidos en estas (Figura 4). De esta manera, se mostró en distintos colores el resultado de cada ejecución. En verde si se ejecuta con éxito, en gris cuando no se ejecuta y en rojo cuando se producen errores.

Con el seguimiento regular de los resultados obtenidos en Jenkins, y en las herramientas de pruebas y control de versiones utilizadas, se evalúan métricas en el proyecto como el tiempo de compilación, tiempo de detección de errores, número de errores encontrados, frecuencia con la que se envían cambios al repositorio, entre otras. Estas métricas son indicadores clave de rendimiento del proyecto (KPIs) y se evalúan en correspondencia con los objetivos establecidos en la fase inicial de

preparación. En la Figura 5 se muestra el análisis realizado de 3 de los indicadores en el proyecto AsiXmec. Este análisis reveló que existen oportunidades para realizar ajustes adicionales e implementar nuevas mejoras en el proyecto, con el objetivo de reducir el tiempo de compilación y disminuir el número de errores. La implementación de las mejoras fue contemplada en la fase de evaluación de resultados del proceso de mejora continua (CI) de AsiXmec. A través de esta evaluación fue posible diseñar planes de acción para mejorar el proceso.

S	Ejecución	Tiempo desde ↑	Estado
✓	AsiXmec » Test unity AsiXmec » Test #24	7 Min 54 Seg	estable
✓	AsiXmec » Test unity AsiXmec » Test #23	19 Min	estable
✓	AsiXmec » Test unity AsiXmec » Test #22	19 Min	estable
✓	AsiXmec » Test unity AsiXmec » Test #21	19 Min	estable
✓	AsiXmec » Test unity AsiXmec » Test #20	19 Min	estable
✓	AsiXmec » Test unity AsiXmec » Test #19	19 Min	estable
✓	AsiXmec » Test unity AsiXmec » Test #18	15 días	estable
✓	AsiXmec » Test unity AsiXmec » Test #17	15 días	estable
✓	AsiXmec » Test unity AsiXmec » Test #16	15 días	volvió a normal
✗	AsiXmec » Test unity AsiXmec » Test #15	15 días	fallido desde la ejecución #10
✗	AsiXmec » Test unity AsiXmec » Test #14	15 días	fallido desde la ejecución #10
✗	AsiXmec » Test unity AsiXmec » Test #13	15 días	fallido desde la ejecución #10
✗	AsiXmec » Test unity AsiXmec » Test #12	15 días	fallido desde la ejecución #10

Figura 3: Historial de ejecuciones en Jenkins. Fuente: Elaboración propia

Tendencia del tiempo de ejecución

Ejecución ↑	Duración
#24	3 Min 2 Seg
#23	32 Seg
#22	34 Seg
#21	37 Seg
#20	1 Min
#19	1 Min 6 Seg
#18	35 Seg
#17	29 Seg
#16	39 Seg
#15	53 Ms
#14	59 Ms
#13	5 Seg
#12	5 Min 5 Seg
#11	4 Min 4 Seg
#10	6 Min 4 Seg
#9	41 Seg
#8	41 Seg
#7	31 Seg
#6	38 Seg
#5	32 Seg
#4	46 Seg
#3	32 Seg
#2	1 Min 20 Seg

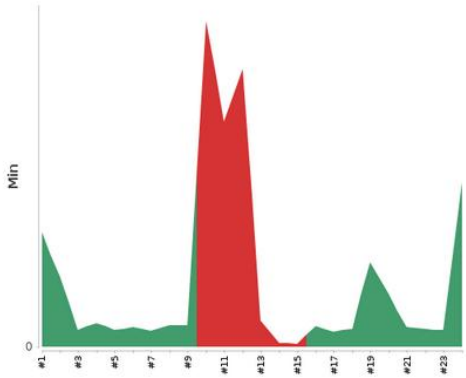

Figura 4: Tendencia de las ejecuciones realizadas. Fuente: Elaboración propia.

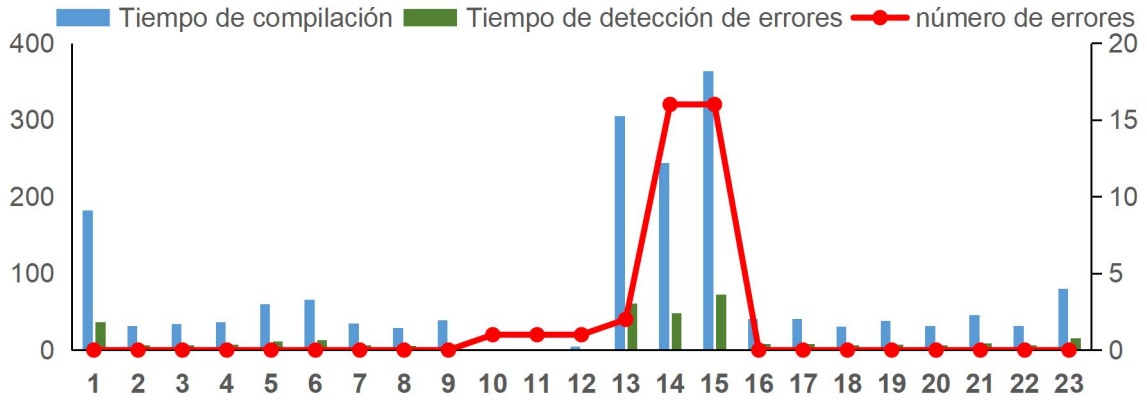

Figura 5: Análisis de KPIs en el proyecto AsiXmec. Fuente: Elaboración propia

En la fase 4 del proceso CI, se lleva a cabo la utilización de la matriz de análisis DAFO para representar el proceso de retroalimentación del equipo. En la matriz DAFO del proyecto se identifican puntos fuertes (Fortalezas) y áreas de mejora (Debilidades), se destacan los aspectos positivos del proceso (Oportunidades), y se analizan las amenazas o desafíos que se han presentado durante el proceso de Integración Continua en AsiXmec (Figura 6).

Figura 6: Análisis DAFO del proyecto. Fuente: Elaboración propia

Después de analizar de manera clara y concisa los aspectos relevantes en la matriz DAFO para la retroalimentación del equipo, se identifican las siguientes lecciones aprendidas:

- La importancia de definir pruebas adecuadas desde el inicio del proceso para detectar y corregir errores de forma temprana.
- La necesidad de documentar de forma clara y detallada los procesos y procedimientos para facilitar la comprensión y la trazabilidad de las acciones realizadas.
- La importancia de buscar capacitación y formación continua en herramientas y tecnologías utilizadas en la Integración Continua.

- La necesidad de mantener un compromiso constante con la mejora continua y la optimización de los procesos para garantizar la calidad del software y la eficiencia en el desarrollo.

Validación Comparativa

En la tabla 3 se muestra la comparativa realizada del proyecto antes y después del proceso CI, lo cual representa el impacto de la Integración Continua en AsiXmec y valida el proceso aplicado como solución a la problemática planteada.

Tabla 3: Comparativa del proyecto AsiXmec antes y después del proceso CI. Fuente: Elaboración propia.

Parámetro de CI	Antes de CI	Después de CI
Tiempo de detección de errores	Manual, Varios días	Automático, Minutos
Cobertura de pruebas	Baja	Alta
Tiempo de compilación	Horas	Minutos
Feedback del equipo	Lento, poco claro	Rápido, detallado
Colaboración del equipo	Baja colaboración, trabajos aislados	Mayor colaboración, trabajo en equipo, mayor comunicación
Identificación de responsables de errores	Difícil de identificar responsables	Más fácil de identificar quién produjo el error.
Cumplimiento de estándares de código	Cumplimiento irregular de estándares	Mejora en el cumplimiento de estándares de código
Automatización de revisiones de código	Revisiones manuales	Automatización de revisiones de código implementada
Retroalimentación sobre calidad del código	Retroalimentación limitada y ocasional	Retroalimentación continua y enfocada en la calidad del código.

A partir del trabajo realizado, se obtuvieron las siguientes recomendaciones a tener en cuenta para trabajos futuros con el fin de extender la investigación y contribuir en la mejora del proceso de desarrollo de software:

- Elaborar un plan de crecimiento para escalar el proceso de Integración Continua y futuros procesos como el Despliegue Continuo, a medida que el proyecto crezca, considerando la posibilidad de incorporar nuevas herramientas y pruebas que ayuden a gestionar un aumento en la complejidad del software.
- Crear un repositorio para las lecciones aprendidas que se actualice después de cada fase de evaluación, esto permite capitalizar conocimiento adquirido, evitar errores pasados y maximizar el aprendizaje organizacional.
- Elaborar guías y tutoriales sobre el uso de herramientas y metodologías adoptadas, así como mejores prácticas para los diferentes roles.

4. Conclusiones

Se realizó un proceso de Integración Continua para el sistema AsiXmec que abarca herramientas de software libre, permitiendo en el proyecto implementar una opción a bajo costo, con una retroalimentación suficiente para afirmar que se cumple con un proceso de Integración Continua. Las fases definidas en el plan de implementación fueron cruciales, comenzando con la preparación del equipo y el entorno, hasta la planificación y ejecución de pruebas y la retroalimentación del equipo. Cada etapa se diseñó para asegurar una integración adecuada del código, facilitando la colaboración efectiva entre los miembros del proyecto. La especificación de pruebas de software, que incluye análisis estático de código, pruebas unitarias y pruebas de interfaz de usuario, resultó fundamental, no solo en la mejora de la calidad del proyecto, sino también en la detección temprana y corrección de errores, lo que contribuye a minimizar riesgos y costos en las fases posteriores del desarrollo.

La evaluación de métricas de calidad y el uso de herramientas como Jenkins proporcionaron una visibilidad clara del estado del trabajo, lo que favoreció la toma de decisiones informadas. Este monitoreo constante identificó oportunidades de mejora y permitió implementar acciones correctivas, manteniendo así un compromiso con la optimización de los procesos. La retroalimentación del equipo, apoyada en el análisis DAFO, resaltó lecciones aprendidas valiosas para el crecimiento y madurez del grupo de desarrollo. Se enfatizó la necesidad de capacitación continua, documentación clara y la adecuada definición de pruebas como pilares fundamentales para el éxito del proceso de Integración Continua. La CI implementada para AsiXmec no solo tuvo como objetivo mejorar la calidad del software, sino que también fomentar una cultura de colaboración y mejora continua dentro del equipo, creando un entorno propicio para el desarrollo ágil y sostenible del proyecto.

Referencias

1. Erazo-Arteaga, Victor A. *El diseño, la manufactura y análisis asistido por computadora (CAD/CAM/CAE) y otras técnicas de fabricación digital en el desarrollo de productos en América Latina.* Chile : La Serena, Scielo Analytics, Información tecnológica, abril de 2022, Vol. 33, No.2, ISSN 0718-0764. http://dx.doi.org/10.4067/S0718-07642022000200297. Disponible en la web: https://www.scielo.cl/scielo.php?script=sci_arttext&pid=S0718-07642022000200297

2. Cruz-Arcos, Guillermo Mauricio, y otros. *Importancia de los sistemas CAD-CAM para el desarrollo de proyecto de conformado de materiales.* Polo del Conocimiento, Edición núm. 63, noviembre de 2021, Vol. 6, No. 11, pp. 370-382, ISSN: 2550 - 682X.

3. C., Lucía. *Los mejores softwares cad para todos los niveles.* 3Dnatives, febrero de 2024. Disponible en la web: https://www.3dnatives.com/es/mejores-softwares-cad-programa-180320192 .

4. Shih, Randy. *Learning autodesk inventor 2021. Modelin, Assembly and Analysis*. s.l.: SDC Publications, 2021, ISBN: 978-1630573447.

5. Artigas, Midiala Baños y Rodríguez, Marvyn Amado. *Versión 2.0 del módulo Snap de la herramienta AsiXMec.* Serie Científica de la Universidad de las Ciencias Informáticas, 2016, Vol. 9, No. 3 pp. 149-163. ISSN: 2306-2495.

6. Rodriguez, M. R.; Gamboa, Y. M.; López, S. T. La integración continua pilar fundamental en el proceso de desarrollo de software, 2019, vol. 12, pp. 102-116.

7. Istifarulah, Mochamad Hanif Rifa'i y Tiaharyadini, Rizka. *DevOps, Continuous Integration and Continuous Deployment Methods for Software Deployment Automation.* JISA (Jurnal Informatika dan Sains), 2023, Vol. 6, No. 2, pp. 116-123. ISSN: 2614-8404.

8. ISO25010. *ISO/IEC 25010 Modelo de Calidad*, 2011. Recuperado el 21 de Junio de 2023, de http://www.iso25000.com/index.php/normas-iso-25000/iso-25010.

9. Kim, Gene, y otros. *The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations.* s.l. : IT Revolution Press, 2016. pp. 150 -162. ISBN: 978-1942788003.

10. Forsgren, Nicole, Humble, Jez y Kim, Gene. *Accelerate: The Science of Lean Software and DevOps: Building and Scaling High Performing Technology.* s.l. : IT Revolution Press, 2018. pp. 200 -220. ISBN: 978-1942788331.

Contribución de los autores

Yanays Fernández Miranda. ORCID, https://orcid.org/0000-0002-2137-1201
Participó en el diseño, implementación y validación de la investigación, determinación de la viscosidad dinámica y redacción del manuscrito

Héctor Unzueta Lazo. ORCID, https://orcid.org/0000-0003-1227-3631
Participó en la implementación y validación de la investigación

Ángel Alberto Vázquez Sánchez. ORCID, https://orcid.org/0000-0002-3130-7983
Participó en la supervisión y validación de la investigación.