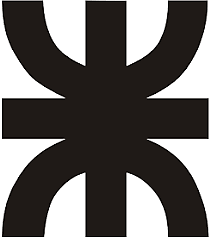
**Universidad tecnológica nacional**

Facultad Regional de Córdoba



Ingeniería en Sistemas de Información

Ingeniería de Software

***Docentes***

ING COVARO, Laura Inés (Adjunto);

ING. MASSANO, María Cecilia (JTP)

ING CRESPO, Mickaela (Ayudante 1ra)

***Integrantes***

ANGLADA, Martín Legajo: 58159 martinanglada@gmail.com;

CHECA, Nicolás Legajo: 58351;

MARRO, Yanina Legajo: 47594 yani\_marro@hotmail.com;

PICOSSI, Matias Legajo: 54941;

TOLEDO, Rodrigo Legajo: 75441.

**Grupo N°**: 7

***Curso:*** 4K3

2020

ÍNDICE

[Propósito 2](#_Toc51326421)

[Integración Continua 3](#_Toc51326422)

[Introducción 3](#_Toc51326423)

[El método tradicional y su relación con la integración continua 4](#_Toc51326424)

[Explicación de CI: ¿Qué es la integración continua? 4](#_Toc51326425)

[El origen de la integración continua 5](#_Toc51326426)

[Los pros y los contras de la integración continua 6](#_Toc51326427)

[¿Cómo es el proceso de implementación continua? 7](#_Toc51326428)

[Elegir las herramientas de integración continua adecuadas 8](#_Toc51326429)

[Estructura del poster 10](#_Toc51326430)

[Herramientas y tecnologías utilizadas 10](#_Toc51326431)

[Referencias Bibliográficas 11](#_Toc51326432)

Propósito

El documento presentado a continuación detalla un resumen realizado respecto a lo investigado y analizado de toda la bibliografía e información web consultada.

Este documento ha sido utilizado de base para construir el poster científico. Toda la información referida puede ser consultado en la sección referencias bibliográficas al final del documento.

Integración Continua

Introducción

La Integración Continua es una práctica de software expuesta por Martin Fowler, que permite entre otras bondades la reducción de riesgos y la realización de tareas repetitivas, automatizando al máximo los procesos involucrados en el desarrollo de software y permitiendo la generación de software listo para desplegar e incremento de su calidad. Cuando se cumplen los principios que rigen la Integración Continua se obtiene una herramienta eficaz y útil que ayuda a la sinergia del desarrollo de los proyectos de software.

Según Martin Fowler, precursor del concepto de Integración Continua, define esta técnica como una práctica de software donde los miembros del equipo integran su trabajo frecuentemente, usualmente al menos una vez al día, dando lugar a múltiples integraciones por día; cada integración es verificada por un autómata para detectar errores de integración lo más pronto posible (1). La Integración Continua se ha convertido en una práctica necesaria en el desarrollo del software por los múltiples beneficios que ofrece, entre los que se destacan: reducción de riesgos y tareas repetitivas, generación de software listo para desplegar e incremento de la calidad. Esta técnica permite el monitoreo de Sistemas de Control de Versiones debido a que cuando un cambio es detectado, es posible que una herramienta compile y pruebe automáticamente los cambios detectados, notificando de inmediato a los involucrados de las posibles fallas detectadas.

Existen algunos requisitos vitales para lograr los objetivos en los que se enfoca la Integración Continua. Entre éstos se destacan:

• Un equipo motivado en la tarea que realizan.

• Disponibilidad de un sistema de Control de Versiones (Subversion, Git, etc.).

• Un servidor de Integración Continua (Jenkins, Cruise Control, Travis CI, etc.).

• Automatización al máximo de los procesos.

• Sistema de notificación (correo, SMS, etc.).

Un aspecto importante en la implantación de un sistema de Integración Continua lo constituye la selección del servidor encargado de realizar las integraciones. Existen muchas alternativas en este sentido, pero una solución completa lo constituye Jenkins (2), el cual es un servidor de Integración Continua Open Source escrito en Java. Jenkins es fácil de usar, su interfaz gráfica es simple, intuitiva, su curva de aprendizaje es baja, cuenta con buena documentación, gran comunidad y un excelente sistema de plugins, lo que permite integrar múltiples sistemas y que la interacción con el usuario se facilite (3).

Los aportes de la Integración Continua al desarrollo del software son significativos. Sin embargo, este proceso conlleva a la adopción de algunas prácticas de los desarrolladores; una de ellas es la realización de commits frecuentes y que se realicen sin errores para no interrumpir la integración con el trabajo del resto de los implicados. Es por ello que se hace necesaria la concientización en todos los involucrados de la importancia que tiene cumplir con las normas establecidas para el proceso de Integración Continua.

El método tradicional y su relación con la integración continua

Los métodos de desarrollo tradicionales no tienen un requisito específico sobre la frecuencia con la que se integrará el código fuente de un proyecto. Los desarrolladores pueden trabajar en el mismo proyecto por separado durante días o incluso semanas en el código fuente y nunca se dan cuenta de cuántos conflictos se están creando. Estos procesos a menudo dan como resultado largas sesiones de depuración que pueden retrasar gravemente la implementación o pueden provocar problemas importantes de software una vez implementados. Cuantos más desarrolladores compartan un código, más problemas pueden surgir a largo plazo. Existe una integración continua para minimizar estos conflictos y facilitar a los equipos la identificación de errores y problemas antes de la implementación.

Durante el proceso de desarrollo ágil, el código debe actualizarse constantemente, pero sin un sistema eficaz, estos cambios constantes pueden llevar a confirmaciones poco frecuentes y pruebas tardías. La integración continua brinda a los equipos la capacidad de crear un proceso verdaderamente ágil mientras se minimizan los errores de software y los posibles conflictos.

Las empresas de software a menudo encuentran que un proceso de integración continuo es mucho más eficiente para sus negocios, pero para que un sistema de CI sea efectivo, debe implementarse correctamente. En la siguiente publicación, exploraremos todo lo que necesita saber sobre la integración continua y le mostraremos cómo CI puede ayudar a agilizar los procesos de desarrollo de su negocio.

Explicación de CI: ¿Qué es la integración continua?

La integración continua, o CI, es una práctica de desarrollo en la que los desarrolladores integran el código en un repositorio compartido varias veces al día. Cada una de estas integraciones se puede confirmar mediante compilaciones y pruebas automatizadas. En lugar de realizar grandes integraciones cada poco día (o más), se introducen pequeñas iteraciones con frecuencia. Por lo tanto, si se detecta un defecto, se puede identificar rápidamente el cambio específico que lo causó.

Martin Fowler, científico jefe de ThoughtWorks, dijo: "Muchos equipos encuentran que este enfoque [de integración continua] conduce a problemas de integración significativamente reducidos y permite que un equipo desarrolle software cohesivo más rápidamente".

Los equipos de desarrollo que no utilizan la integración continua son más propensos a enfrentarse a desafíos de integración. Dado que su integración solo ocurre ocasionalmente, existe un alto potencial de error: los cambios de un desarrollador pueden entrar en conflicto fácilmente con las integraciones de otro desarrollador. Cuanto más tiempo tarde en que se lleve a cabo una integración, es probable que aparezcan más problemas, y más difícil será rastrear el origen de estos problemas.

El origen de la integración continua

La “integración continua” no es un término nuevo; y aunque en realidad es anterior a lo que ahora se conoce como procesos ágiles, el proceso no se definió realmente hasta el año 2000. La historia de la integración continua incluye:

* En 1996, Steve McConnell presentó un concepto que Microsoft utilizó para Windows NT 3.0 en la década de 1990. Este proceso se conoció como la técnica de “Prueba de humo y construcción diaria” y se centró principalmente en la frecuencia de integración en lugar de la automatización.
* En 1998, la integración continua se convirtió oficialmente en una de las prácticas centrales de Extreme Programming (XP): un enfoque para el desarrollo de software que se centra en los resultados comerciales primero y adopta un enfoque incremental para el desarrollo de productos con pruebas y revisiones continuas.
* En 2000, Martin Fowler publicó un artículo que finalmente definió el proceso de "integración continua".
* En 2001, se publicó el primer “servidor de integración continua” bajo una licencia de código abierto, con el nombre *CruiseControl* .

En la actualidad, algunas de las empresas de tecnología más grandes del mundo utilizan la integración continua y otros procesos continuos. Si bien establecer un gran proceso de integración continua puede ser difícil para los equipos de desarrollo más pequeños, el resultado de implementar uno suele ser el mismo: una implementación más rápida con menos errores.

Los pros y los contras de la integración continua

La implementación de la integración continua conlleva inmensos beneficios para cualquier organización. Sin embargo, también existen varios inconvenientes potenciales que las organizaciones deben considerar antes de decidir si la integración continua es la opción correcta para su proyecto en particular.

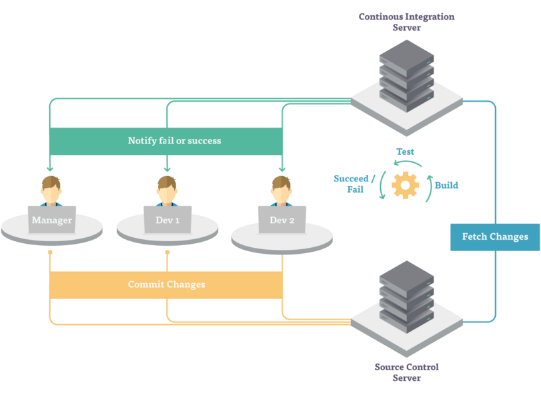
Algunos de los beneficios más atractivos de la integración continua incluyen:

1. **Riesgo reducido** : en la mayoría de los casos, cuanta más gente trabaje en un proyecto, más riesgoso es integrar código nuevo. Cuando el código se integra con poca frecuencia y en grandes cantidades, es probable que haya problemas de codificación. Depurar y resolver estos problemas puede resultar caro y llevar mucho tiempo; y podría significar realizar cambios importantes en el código. Al integrarse con frecuencia y en pequeñas iteraciones, los errores se pueden identificar rápidamente y solucionar sin una depuración extensa.
2. **Mayor calidad de código:** con menos tiempo dedicado a depurar el software, se puede dedicar más tiempo a la creación de nuevas funciones y al desarrollo adicional del producto. Con cada pequeño cambio que se prueba automáticamente durante la integración, la práctica de CI conduce a una mayor calidad de código en todo el proyecto.
3. **Menos estrés para los desarrolladores**: el estrés de depurar miles de líneas de código puede ser intenso. Las sesiones de depuración pueden estar llenas de presión, ya que los administradores de la organización se apoyan en gran medida en los programadores para solucionar rápidamente los problemas que enfrentan los clientes. La integración continua elimina una gran cantidad de estrés de la etapa de depuración, lo que permite a los desarrolladores resolver problemas más rápido, con menos esfuerzo y con un tiempo de respuesta más corto.

Sin embargo, antes de implementar un sistema de integración continua, las organizaciones deben ser conscientes de los siguientes inconvenientes:

1. **Nuevos procesos de equipo** : implementar procesos continuos puede parecer intenso para un equipo que está acostumbrado a integrarse con un enfoque diferente. Algunos miembros del equipo pueden no estar de acuerdo en que es necesario, y no todos pueden estar convencidos del inmenso beneficio que ofrece CI. Una vez que se ha implementado un sistema de CI, los miembros del equipo deben recibir capacitación en todos los procesos nuevos para garantizar que sea adoptado por todo el equipo de desarrollo.
2. **Costos asociados** : puede haber algunos costos adicionales asociados con el establecimiento y el mantenimiento de un sistema de integración continua eficaz, y es posible que la administración no comprenda por qué estos costos son necesarios. Sin embargo, a largo plazo, los procesos continuos tienden a amortizarse rápidamente, ya que el tiempo y los costos de depuración y resolución de problemas se reducen considerablemente.
3. **Proyectos existentes**: la integración continua es extremadamente valiosa en cualquier etapa de un proyecto, pero es mucho más fácil establecer un sistema de CI al desarrollar un nuevo proyecto. Los proyectos existentes pueden requerir algunos pasos más para configurarse de manera efectiva. Sin embargo, con las herramientas adecuadas, un proceso de CI se puede configurar de forma bastante sencilla; incluso con proyectos existentes.

¿Cómo es el proceso de implementación continua?



*Figura 1: el proceso de integración continua*

Un sistema de integración continua exitoso consta de varios procesos automatizados para probar, aprobar e integrar nuevos cambios de codificación. Como se muestra arriba, un sistema de CI efectivo incluye los siguientes pasos:

1. Primero, el desarrollador escribe código nuevo y lo envía a un repositorio central.
2. El repositorio es monitoreado por el servidor de CI e identifica los cambios cuando ocurren.
3. El servidor crea el sistema y ejecuta pruebas automatizadas.
4. El equipo de desarrollo recibe información automática si la prueba falla, lo que les permite solucionar el problema de inmediato.
5. Este ciclo se repite de forma continua, integrándose y probando a lo largo de todo el proyecto.

Elegir las herramientas de integración continua adecuadas

Una herramienta de integración continua (CI) es una solución de software que le ayuda a automatizar un paso más del proceso de CI; por ejemplo, puede ejecutar pruebas automatizadas cada vez que envía una nueva confirmación e informa los resultados a una solicitud de extracción. Cuando se combina con herramientas de implementación y entrega continua como DeployBot, los desarrolladores pueden probar su código en múltiples configuraciones, ejecutar pruebas de rendimiento adicionales y crear un proceso automatizado para cada paso hasta la producción.

En la actualidad, existen muchas herramientas de integración continua disponibles en el mercado. Muchas de estas herramientas se pueden integrar fácilmente con administradores de repositorios de git como BitBucket o GitHub, y le permiten automatizar de manera efectiva las compilaciones y las pruebas. Al elegir qué herramienta de integración continua es mejor para sus organizaciones, se deben considerar varios factores:

1. *Soporte:* Dependiendo de la solución que elija, el soporte puede estar incluido o no. Si algo sale mal durante el proceso de configuración, integración u operación, es importante saber dónde puede acudir para obtener ayuda. La duración entre una consulta y una respuesta del soporte podría causar grandes retrasos en su proyecto. También es importante que su solución tenga documentación accesible para que pueda encontrar respuestas a sus problemas específicos, cuando sea necesario.
2. *Modelo de precios:* también es importante que comprenda la estructura de precios de cualquier solución que elija. Algunos software de código abierto pueden ser gratuitos en teoría, pero pueden existir tarifas asociadas por soporte o costos generales adicionales en los que se incurrirá a largo plazo. Por otro lado, las soluciones pagas pueden tener un costo fijo, pero a menudo son más inclusivas en sus características y ofrecen un servicio al cliente más accesible. Sin embargo, las diferentes soluciones de pago suelen tener diferentes modelos de precios. Encuentre una solución y un paquete que se adapte a los requisitos de su organización y asegúrese de comprender qué costos asociados pueden ser necesarios en el futuro.
3. *Integración:* al elegir una solución de CI, busque una que se integre fácilmente con sus herramientas actuales. Tener que cambiar todas sus herramientas para respaldar su nuevo proceso de CI puede resultar extremadamente costoso tanto en tiempo como en dinero. Muchas de las principales soluciones de CI están diseñadas para admitir una amplia gama de herramientas, o pueden ampliarse para admitir otras herramientas mediante complementos; pero algunas soluciones no son tan avanzadas y pueden no ser compatibles con su software existente.
4. *Características del producto:* antes de elegir una solución, identifique exactamente qué características son críticas para su organización. Aunque algunas herramientas de integración continua son totalmente inclusivas y proporcionan una amplia variedad de características diferentes, algunas se centran específicamente en solo unas pocas características importantes. Una solución puede proporcionar una larga lista de funciones, pero es posible que esas funciones no sean las más importantes para su negocio; mientras que otra solución puede proporcionar las herramientas exactas que necesita, pero no *todas* las herramientas que necesita. Evalúe qué características necesita su organización para crear el proceso perfecto y asegúrese de que el software que elija ofrezca esas características.
5. *Facilidad de uso:* implementar una nueva solución puede ser un proceso intensivo, especialmente si tiene un gran equipo de desarrollo. Cada miembro del equipo deberá aprender y adaptarse al nuevo proceso y al nuevo software. Algunas soluciones son muy avanzadas, lo que dificulta la configuración y la formación de su equipo para utilizarlas correctamente. La solución que elija debe ser fácil de implementar y simple de entender para los miembros de su equipo sin una curva de aprendizaje extrema.

Estructura del poster

1. Título1 – (Explicativo y acorde con el contenido de la ponencia. Extensión máxima de 10 palabras, de ser necesario utilizar subtítulo)
2. Información de los investigadores: a. Nombre completo del autor | correo b. Nombre completo del autor | correo c. Universidad del autor(es) d. Semillero
3. Introducción: Problema, objetivos, justificación
4. Metodología: cúal fue el proceso con respecto al método y metodología que se llevó a cabo para obtener los datos y procesamiento de la información de la investigación. (un máximo de 100 palabras)
5. Marco conceptual: cuáles fueron los principales conceptos que se utilizan para el análisis de la investigación. (un máximo de 200 palabras)
6. Resultados: cuáles fueron los hallazgos más importantes del proyecto. Cómo se resuelve. (un máximo de 200 palabras)
7. Conclusiones: cuáles son los principales aportes de la investigación. Qué otras preguntas arrojan. (un máximo de 100 palabras)
8. Bibliografía: principales fuentes de referencia. Citar en formato APA.

El material bibliográfico leído esta disponible de ser consultado en el apartado referencias.

Herramientas y tecnologías utilizadas

Se trabajó con CANVAS y Adobe Color.

Referencias Bibliográficas

* Sander Rossel. (2017). *Continuous Integration Delivery and Deployment* (Packt 2017 ed.).
* Material referenciado de cátedra.
* DeployBotBlog. The Expert Guide To Continuous Integration: Disponible en <https://deploybot.com/blog/the-expert-guide-to-continuous-integration>
* Jorge Turrado. Integración continua: qué es y por qué deberías aprender a utilizarla cuanto antes. Disponible en: <https://www.campusmvp.es/recursos/post/integracion-continua-que-es-y-por-que-deberias-aprender-a-utilizarla-cuanto-antes.aspx>
* Edureka. What is Continuous Integration?. Disponible en: <https://www.edureka.co/blog/continuous-integration/>
* MUHAMMAD ALI BABAR. Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices. Disponible en: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7884954>
* Global Continuous Integration Tools Market: Industry Analysis and Forecast (2018-2026) –by Organization Size, Deployment Mode, Vertical and Region. Disponible en: <https://www.maximizemarketresearch.com/market-report/global-continuous-integration-tools-market/28030/>
* La integración continua en el desarrollo de software. Disponible en: <https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/integracion-continua/>
* Poster: Como diseñé mi poster de la A a la Z. Disponible en: <https://neoscientia.com/como-hacer-un-poster-cientifico/>
* Poster: El poster científico: pasos. Disponible en: <https://files.sld.cu/bmn/files/2018/04/EL-POSTER-CIENTIFICO.-PASOS.pdf>