INTEGRACIÓN CONTINUA Y ENTREGA CONTINUA

Claudia Iturrizaga Goñi

38991@sistemas.frc.utn.edu.ar

Pablo Jornet

62901@sistemas.frc.utn.edu.ar

Santiago Ponce

70083@sistemas.frc.utn.edu.ar

Maximiliano Yacuzzi

66276@sistemas.frc.utn.edu.ar

Ignacion Zanotti

69860@sistemas.frc.utn.edu.ar

Genaro Franceschelli

67099@sistemas.frc.utn.edu.ar

**RESUMEN:** *En el presente paper describimos la definición de lo que es entrega continua y integración continua, plantemos cuales son las ventajas y desventajas de ambas técnicas y sus formas de implementación.*

**PALABRAS CLAVE:** Deployment Pipeline, Entrega continua, Filosofía lean, Integración continua, Manifiesto ágil

**1 INTRODUCCIÓN**

Las metodologías ágiles son aquellas metodologías de gestión que permiten adaptar la forma de trabajo al contexto y naturaleza de un proyecto, basándose en la flexibilidad y la inmediatez, y teniendo en cuenta las exigencias del mercado y los clientes. Los pilares fundamentales de las metodologías ágiles son el trabajo colaborativo y en equipo.[ 1]

En 2001 se fecha el nacimiento oficial de Agile. Entonces se trataba de una evolución de metodologías de desarrollo como scrum y sentó las bases de la filosofía lean. Poco a poco, pasó de un entorno exclusivo de startups a ser aplicada en grandes empresas. Agile busca eliminar barreras entre departamentos y burocracias innecesarias, flexibilizar el proceso de desarrollo y dividirlo en tareas pequeñas alcanzables en periodos cortos de tiempo, reducir el tiempo de ciclo y alinear el trabajo con las necesidades de los clientes.

La Entrega Continua no sale de la nada, sino que es a su vez una evolución del método de Integración Continua, popularizado por Martin Fowler en los últimos años. La Integración Continua se basa en compilar y probar el software de forma automática lo más frecuentemente posible durante el desarrollo. La Entrega Continua busca que dicho software no solo se pruebe, sino que esté listo para pasar a producción en cualquier momento. [2]

**2 Desarrollo**

**2.1 Integración continua**

Una de las funciones principales de la CI o Integración Continua, es facilitar el trabajo a los desarrolladores. Cuando hablamos de integración contínua, nos referimos a la creación de código entre diferentes desarrolladores y bajo una minuciosa comprobación del mismo. Para hacer que esto sea posible y fácil, no esperaremos a que el código esté finalizado, como se hacía en los métodos convencionales o de cascada, sino que dividiremos el trabajo en pequeñas tareas ,también denominadas sprints, que no suponen un gran esfuerzo ni necesitan una gran inversión de tiempo, de forma que se puedan ir comprobando dichas tareas frecuentemente. Estas comprobaciones se deben llevar a cabo por lo menos una vez al día y suelen ser realizadas al finalizar cada jornada laboral. En base a los resultados, podemos seguir creando o entramos en la fase de modificación, donde una vez detectado el error, el grupo decidirá cómo se puede superar y para ello, que deben modificar.[ 3]

A continuación mencionaremos las ventajas y desventajas de la integración continua.

Desventajas

1. Feature Branching se encuentra en una espiral descendente sin un final bonito. El cambio de nombre de una función es un caso relativamente obvio de un conflicto semántico. En la práctica pueden ser mucho más sutiles. Las pruebas son la clave para descubrirlas, pero cuanto más código haya para fusionar, más probable será que tenga conflictos y más difícil será solucionarlos. Es el riesgo de conflictos, particularmente conflictos semánticos, lo que hace que las grandes fusiones den miedo.[4]
2. Costes de hardware. Es necesario un conjunto de máquinas que albergarán las diversas herramientas para la integración contínua: repositorio, automatización, entornos,[5]
3. Costes de personal de QA, además de las máquinas habrá personal que se encargue de la configuración, seguimiento, realización y generación de informes de las pruebas. La ventaja es que cualquiera puede ver el resultado de las pruebas con lo que los propios desarrolladores pueden corregir aquellas partes del código que no sean correctas, o los gestores de proyecto controlar la calidad del producto construido

Ventajas

1. Como es fácilmente deducible, el nombre que adquiere esta práctica es descriptivo y literal. Nos basamos en la comprobación continua del código para poder integrar poco a poco las mejoras o ir actualizando a diario para obtener un resultado mucho más fiable y en periodo menor de tiempo. Lo que nos permite ajustarnos mejor a los timings propuestos y evitar las presiones derivadas de última hora por la presión de las entregas.
2. Otra de las ventajas que tiene la CI es a nivel personal. Puesto que estas prácticas se llevan a cabo en proyectos conjuntos, son los propios desarrolladores los que van a ir analizando el código creado, apoyándose los unos en los otros. Ellos mismos van a tener que aprender diferentes métodos de integración y verse obligados a superar día tras día diferentes errores o fallos encontrados en el código creado. Esto fomenta la comunicación entre el equipo, y hace que sea mucho más enriquecedor tanto a nivel individual como a nivel de grupo o equipo.
3. En todo momento se tendrá una versión para pruebas, o una primera fase para poder contrastar la evolución del código de forma que haga posible detectar los errores a tiempo y poder corregirlos.
4. A su vez, en todo momento cada miembro del equipo tiene acceso a la versión final.
5. La integración continua garantiza unos resultados de calidad y un funcionamiento correcto del proyecto gracias a su continua supervisión y a la reducción de errores.
6. Evidentemente todo este proceso de revisiones y correcciones lo más frecuentes posibles, forman parte de una automatización que se desarrolla para facilitar la comunicación entre el equipo y la evolución del proyecto.
7. Otro factor importante en la integración contínua, es la monitorización de las métricas más relevantes de nuestro proyecto, de forma que podamos tener presente en todo momento la calidad del mismo. Esto a la larga, nos llevará a un buen código sin necesidad de modificaciones una vez llevado a la última fase, puesto que lo vamos optimizando a medida que lo construimos.[ 3]

**2.2 Entrega continua**

La entrega continua es un proceso de punta a punta que abarca el ciclo entero de construcción, prueba, despliegue y liberación. Para que la integración continua sea una realidad, debemos automatizar estas actividades. Esto habilita a las organizaciones para innovar, reducir costos y diferenciarse en mercados competitivos, por medio de un proceso eficiente de entrega de software.

El manifiesto ágil define 12 principios, y el primero de estos dice “... satisfacer al cliente por medio de la entrega temprana y continua de software valioso”.

Es decir, las metodologías ágiles nos enseñaron a agregar valor a los productos de software una historia a la vez, de manera que aceleremos los ciclos de retroalimentación y continuamente estemos alineando los productos que construimos con las necesidades del mercado. Estas metodologías promueven la integración continua, y esto provocó un flujo de liberaciones más pequeñas y frecuentes, que llegaron hacia los equipos de calidad y operación de TI.

La prueba clave de si has logrado o no la entrega continua es que el usuario de negocio pueda solicitar en cualquier momento que la versión actual de desarrollo pueda ser liberada a producción, y el equipo no sienta temor por esto.[6]

La Entrega Continua consiste en lo siguiente:

* Lograr que el software se pueda implementar en cualquier momento de su ciclo de vida
* El equipo de desarrollo prioriza tener un producto funcional en todo momento ante trabajar en nuevas funcionalidades
* Cualquiera en la empresa puede conocer, de forma rápida, el estado del producto
* El producto está siempre listo para implementarse en pocos clics si fuese necesario [2]

A continuación, abordamos ventajas e implementacion de las entregas continuas.

I.Ventajas

Las organizaciones que implementen la Entrega Continua obtendrán cuatro grandes ventajas:

1. Un tiempo de comercialización más rápido. Los clientes tienen acceso antes a las nuevas funciones y los desarrollos reciben el feedback del cliente con mayor agilidad.
2. Mayor capacidad de experimentación. Al agilizar el proceso de lanzamiento, las nuevas funciones pueden probarse, activarse o desactivarse con diferentes audiencias o públicos y con mayor flexibilidad.
3. Aumento de la capacidad de reacción ante errores. Poder probar el producto de forma continua facilita la identificación de errores y su subsanación.
4. Refuerzo de la productividad de ingeniería. La motivación en el equipo de desarrollo aumenta, ya que se puede ver de forma inmediata el resultado del trabajo y cómo este aporta valor a la empresa.[ 2]

II. Cómo se implementa

La clave para implementar un proceso exitoso de entrega continua pasa por integrar continuamente el software realizado por el equipo de desarrollo, la creación automática de versiones instalables y la ejecución de estas pruebas automatizadas para detectar errores, todo a través de un flujo de tareas automatizado que se conoce como Deployment Pipeline.

Así, la implementación de la Entrega Continua se daría en tres etapas:

1. La automatización de la compilación y la integración continua. A medida que se desarrollan e implementan nuevas funciones, estas se compilan y prueban de forma automática.
2. La automatización de pruebas. Las nuevas versiones de la aplicación se testean de forma automática para comprobar que son estables y funcionales.
3. La automatización de la implementación. Por último, se implementa, si es necesario, la nueva versión en el entorno de producción.[ 2]

**3 Conclusión**

1. Se puede decir que el objetivo principal de las metodologías ágiles es maximizar el valor entregado al cliente de la manera más rápida posible. Para ello, se apoyan en las directrices establecidas en el manifiesto ágil y en los doce principios que se derivan del mismo.
2. La IC está enfocada a disminuir el riesgo. Permite detectar y solucionar los problemas de manera temprana.
3. La IC nos brindará información en todo momento.
4. Permite una rápida retroalimentación de nuestro proyecto.
5. La IC reduce de manera increíble el número de errores en el producto final. En fases ya avanzadas de la integración continua se suelen lanzar inspecciones continuas de código, análisis periódicos para detectar problemas de calidad en él.
6. Son usadas por muchas grandes empresas y se han utilizado en grandes sistemas, lo que hace prever que estas metodologías han llegado para quedarse.
7. La entrega continua abarca más que el proceso de pruebas e incluye al proceso de entrega. Las nuevas características y las modificaciones realizadas en el código llegan automáticamente al usuario final.
8. Las pruebas tienen que ser realmente eficaces en el CD porque los resultados se entregan directamente al usuario final.

**4 Referencias**

[1]https://blog.kezmo.com/qu%C3%A9-son-las-metodolog%C3%ADas-%C3%A1giles-y-por-qu%C3%A9-debes-implementarlas-en-tu-organizaci%C3%B3n-484a510e5b0

[2]https://hablemosdeempresas.com/empresa/entrega-continua-agile/

[3]https://apiumhub.com/es/tech-blog-barcelona/integracion-continua/

[4] FeatureBranch - Martin Fowler https://martinfowler.com/bliki/FeatureBranch.html

[5]Desarrollo Ágil Desventajas http://agilizate.blogspot.com/2012/04/ventajas-y-desventajas-de-la.html

[6] https://sg.com.mx/revista/54/entrega-continua