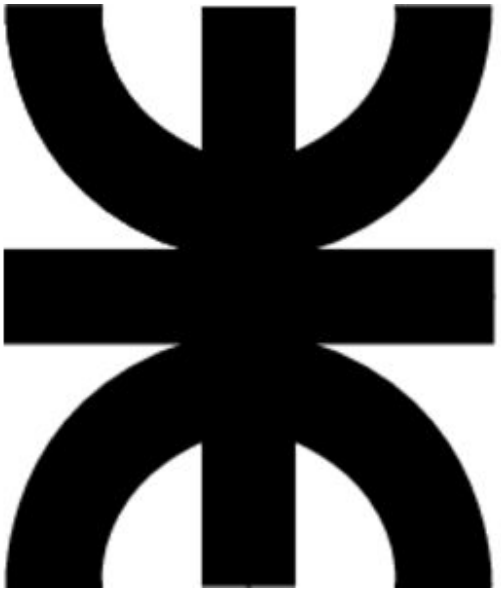
Ingeniería de software

Informe técnico



DOCENTES
Meles, Silvia Judith
Massano, Maria Cecilia
Robles, Joaquin Leonel

GRUPO Nº 4
Arguello, Santiago - 69966
Urbano Moreno, Barbara Romina - 69604
Garcia, Romina - 52889
Britos, Anabel - 63468
Alfonso,Paola - 66685

Fecha de entrega: 30/08/2019

Universidad Tecnológica Nacional – Facultad Regional Córdoba
Ingeniería en Sistemas en Información
Grupo 4

Informe técnico - Continuous Integration

1. Resumen

En el siguiente trabajo buscaremos indagar sobre el concepto de Continuous Integration (Integración Continua) con el fin de comprender esta metodología y su importancia a la hora de aplicarla en un proyecto de desarrollo de software. Para ello, explicaremos qué es la Integración Continua, qué debemos hacer para implementarla en un proyecto de software y qué herramientas podemos encontrar en el mercado actual para llevar a cabo esta metodología.

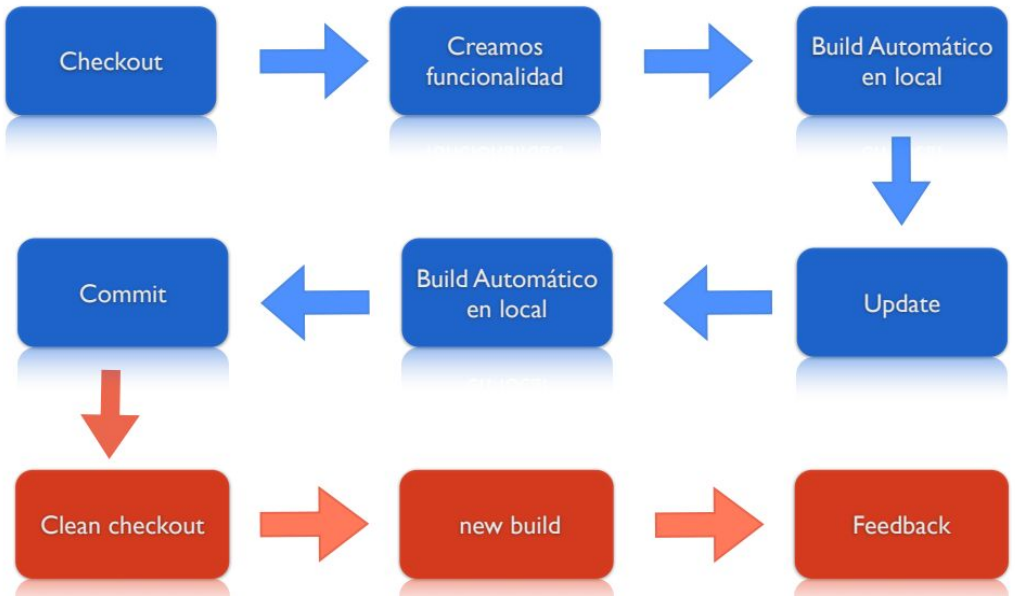
2. Introducción

La integración continua es una práctica de desarrollo de software ágil en ingeniería de software, donde se fusiona toda la base de código de trabajo local de los desarrolladores para compartirla en un repositorio común durante el desarrollo del producto.
En el presente informe conoceremos los beneficios e inconvenientes de aplicar esta práctica.

3. Contenido

¿Que es la integración continua y por qué deberías usarla?

Es la práctica de desarrollo software donde los miembros del equipo integran su trabajo periódicamente. Cada integración se verifica con un build automático (que incluye la ejecución de pruebas) para detectar y corregir errores de integración con mayor rapidez mejorando así la calidad del software. Esto significa probar todo, desde las clases y el funcionamiento hasta los distintos módulos que conforman toda la aplicación. Si una prueba automática detecta un conflicto entre el código nuevo y el existente, la integraciòn continua facilita la resolución de esos errores con frecuencia y rapidez.
El objetivo principal de la Integración Continua es evitar que los desarrolladores se crucen entre sí y eliminar problemas de integración.
Se debe usar con el objetivo de implementar el proyecto en cualquier momento ya que permite un mecanismo de retroalimentación rápida sobre los resultados de la compilación.



Elementos para llevar a cabo integracion continua

Repositorio de código:

Es una herramienta fundamental que nos permite como equipo trabajar de forma sincronizada. Esto significa que cada persona puede avanzar en sus tareas sin retrasar a otras ni ser retrasado por tareas no realizadas hasta el momento. Sin un repositorio de código es prácticamente imposible realizar integración continua.

Sistema de construcciòn automatizado:

Utilizar un entorno complejo de desarrollo para implementar y compilar proyectos puede resultar contraproducente a la hora de construir el proyecto en diferentes máquinas de trabajo, ya que no todas las máquinas en las que vayamos a instalar nuestro proyecto tendrán el mismo entorno. Existen muchas herramientas que nos permiten construir nuestro proyecto de manera simple, automática y abstrayendo la dependencia con el entorno en el que nuestro proyecto es construido. Estas herramientas también suelen tener funciones que automatizan no solo la construcción de nuestros proyectos, sino también el testing asociado a esta actividad.

Commits diarios:

Una vez que tenemos todos los elementos necesarios, debemos recordar que el almacenamiento de los cambios debe ser diario, ya que sí tenemos un repositorio de código pero cada desarrollador sube sus cambios cada periodos muy largos, la integración del proyecto se retrasara durante todo ese tiempo, perdiendo la opción de avanzar continuamente. Además, esto genera más conflictos en los componentes modificados por cada desarrollador, ya que aumenta la posibilidad de que más de un integrante del equipo trabaje en un mismo componente, lo que genera los conflictos mencionados. Por otro lado, no subir diariamente los cambios locales al repositorio limita la posibilidad de comprobar que los cambios realizados funcionan correctamente y no han desestabilizado el proyecto.

Pruebas unitarias:

Para poder comprobar que el proyecto funciona de forma automática necesitamos pruebas unitarias. Existen muchas maneras de incluir tests en el proyecto y dependerá del equipo y su expertis la manera de implementarlos.

Servidor de integración:

Muchos creen que un servidor de integración no es necesario, pero es un paso muy sencillo y barato de automatizar que se podría considerar como necesario. Esta herramienta monitorea el repositorio y detecta los cambios automáticamente, además de integrar el sistema y correr las pruebas unitarias y de integración.

Ventajas de la integración continua

1- Mejora la calidad del Código:

La integración continua contribuye a minimizar los problemas en los sistemas por errores de código. Esto se debe a la prueba constante y el análisis de código al momento de integrar el código de los desarrolladores.

2- Detección de errores más rápida y fácil:

Al integrar continuamente, es mucho más fácil detectar errores lo que permite resolverlos de forma rápida y fácil. Esto no significa que la integración continua resuelva errores, sino que permite detectarlos y corregirlos más rápidamente.

3- Reduce tareas repetitivas y manuales:

Los procesos manuales repetitivos son lentos y propensos a cometer más errores por lo que reducirlos implica un gran beneficio para estar seguro de que los procesos se realizarán aplicando los mismos estándares cada vez.

4- Puede crear versiones de prueba en cualquier momento:

Al estar constantemente integrando el código es posible liberar software en cualquier momento, y sin los errores que ya previamente se detectaron y solucionaron.

5- Completa visibilidad del proyecto:

Se tiene información concreta del avance del proyecto y métricas sobre la calidad del código que se está desarrollando. Ésto permite tomar mejores decisiones a la hora de querer hacer cambios o modificaciones.

6- Mayor confianza y seguridad del equipo de trabajo:

La integración continua permite obtener un software probado y funcional en todo momento, lo que contribuye a la confianza del equipo de trabajo pudiendo ir viendo y probando los avances del software. No se requiere esperar al final del proyecto para verificar que todo se ha realizado de la forma correcta.

Desventajas que se pueden presentar al aplicar integración continua

Cambio de procesos habituales:

Los equipos deben adaptar su forma de trabajo a una nueva metodología, cosa que lleva tiempo.

Precisa servidores y entornos adicionales:

Implementar la integración continua requiere de una inversión inicial que servirá para obtener los servidores de integración y las herramientas necesarias en caso de ser pagas.

Elaboración de un plan de pruebas propio:

Se debe planificar cómo se automatizan las pruebas que el la herramienta implementada realizará.

Puede derivar en demoras cuando varios desarrolladores intentan integrar sus códigos al mismo tiempo:

Al implementar integración continua, tenemos la ventaja de poder trabajar varias personas al mismo tiempo en el mismo proyecto, pero de esto puede derivar un conflicto de código, el cual se debe resolver en el momento de integración, y el cual puede llevar mucho tiempo.

Selección de herramientas de integración continua

En principio, la integración continua se puede aplicar sin necesidad de utilizar herramientas específicas, ya que todas las fases pueden llevarse a cabo de forma manual, pero esto requeriría mucho tiempo y disciplina. Con ayuda de las herramientas justas, se puede facilitar el trabajo. Estas herramientas suelen proveer un servidor y ayudan en la compilación del proyecto y en el control de versiones. Algunas de las herramientas que se pueden utilizar son las siguiente:

Jenkins: este popular programa escrito en Java es una bifurcación de Hudson, que ha dejado de actualizarse. El software de código abierto funciona con distintas herramientas de compilación y sistemas de control de versiones.

Travis CI: esta herramienta de integración continua es especialmente apreciada por su compatibilidad con GitHub, que informa a Travis de los cambios realizados en el código. Existe una versión gratuita del software para proyectos de código abierto y cuenta también con una versión de pago.

Bamboo: con el servidor Bamboo, los desarrolladores podrán llevar a cabo la integración, el despliegue y la gestión del lanzamiento continuo. Pertenece a la empresa Atlassian y cuenta con interfaz web y tecnología Java. Bamboo supone una ayuda a los desarrolladores mediante la automatización de procesos y funciona con diferentes herramientas de compilación. Existe una versión gratuita para proyectos de código abierto.

Gitlab CI: GitLab ofrece un programa propio de integración continua que funciona con la conocida herramienta de control de versiones. Los pipelines pueden configurarse y adaptarse así a los requisitos de cada proyecto. Además, es compatible con GitLab CI Docker.

CircleCI: de este software existen dos versiones distintas para la integración continua. Las versiones permiten trabajar directamente en la nube o bien en un servidor local propio.

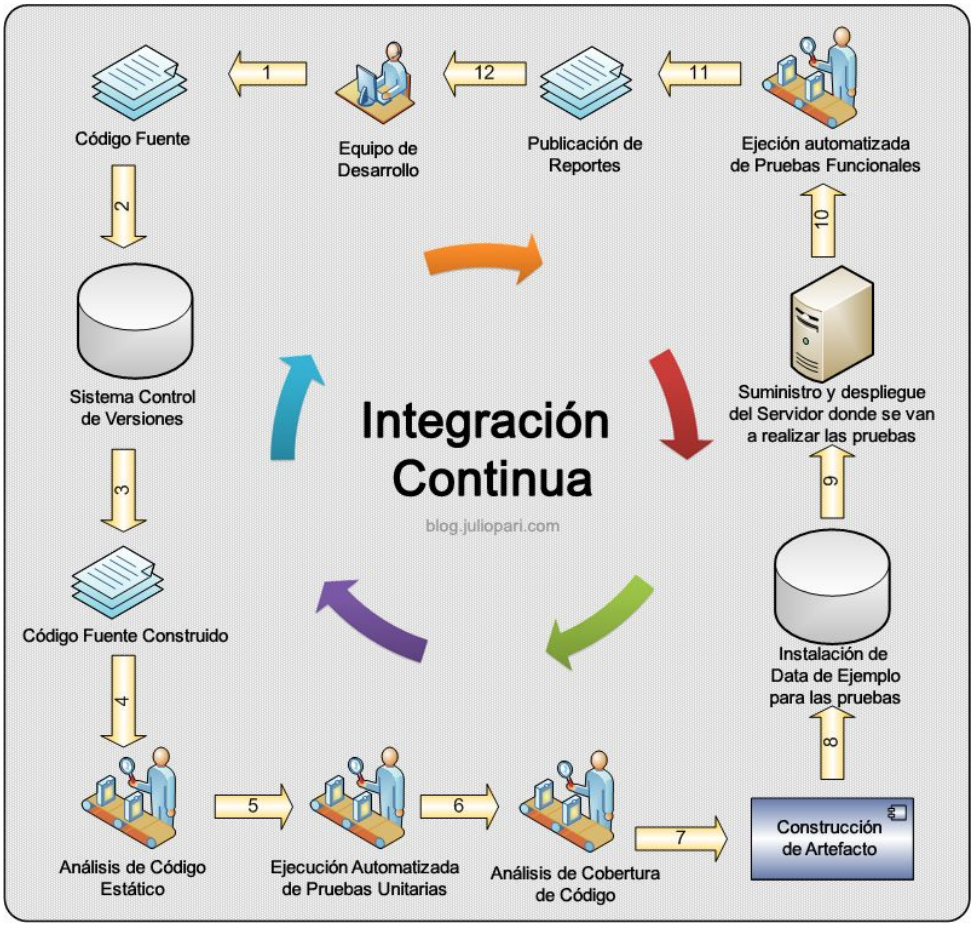
CruiseControl: aunque fue desarrollado por ThoughtWorks (empresa relacionada con Martin Fowlers), CruiseControl ha pasado a ser un proyecto independiente. Este software libre está basado en Java y es compatible con cualquier plataforma. Entre otros, CruiseControls ofrece a los desarrolladores un panel de control (una página web propia) en el que se puede consultar el estado de los builds.

Codeship: Codeship pretende ofrecer a los desarrolladores una opción sencilla para la integración continua. Basándose en la tecnología de contenedores, se pueden crear con él automatismos fácilmente. Para esta tarea, la compañía ofrece dos versiones diferentes: Basic y Pro.

TeamCity: este software comercial pone mucho énfasis en la interoperabilidad con otras herramientas. Su versión estándar es compatible con numerosos programas y el espectro puede ampliarse mediante plugins. Una característica de este software son los Pre-tested commits. TeamCity comprueba el código nuevo antes de integrarlo en la línea principal e informa en caso de errores.

4. Conclusión

Con la investigación realizada para dicho informe y la experiencia de varios integrantes de este grupo que utilizan esta metodología a diario en sus trabajos, pudimos debatir y poder tener una visión más clara sobre las ventajas que nos ofrece la utilización de esta metodología y aprendiendo a tener en cuenta también, sus puntos débiles. De esta manera, podemos concluir junto al resto de integrantes que aún no trabajan o no han utilizado esta metodología, que la integración continua es una metodología importante a considerar para el trabajo en equipo, ya que permite disminuir el riesgo y a la detención y solución temprana de problemas otros beneficios ya mencionados.
Además, nos brinda visibilidad del proyecto en todo momento lo cual permite una rápida retroalimentación del mismo



Referencias

https://medium.com/@realsure10/the-art-of-continuous-integration-ci-in-agile-software-process-2f908ed5e921
https://www.exoscale.com/syslog/what-is-continuous-integration/
https://www.javiergarzas.com/2014/08/implantar-integracion-continua.html
http://www.ambysoft.com/essays/agileTesting.html#ContinuousIntegration
https://sg.com.mx/revista/54/integraci-n-continua
https://www.smartnodus.cl/integracion-continua/
http://blog.juliopari.com/integracion-continua-en-proyectos-agiles-de-software/
https://emanchado.github.io/camino-mejor-programador/html/ch08.html
https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/integracion-continua/
https://sg.com.mx/revista/54/integraci-n-continua

Modelo de Informe técnico utilizado:
https://ocw.unican.es/pluginfile.php/1408/course/section/1805/tema10-comoEstructurarUnInformeTecnico.pdf