

Índice

Tabla de contenido

Índice.....	1
Resumen.....	2
Introducción	3
Desarrollo	4
¿Qué es la Integración Continua?	4
¿En qué consiste?	4
¿Qué gano con implementarlo?	5
¿Qué necesito?	5
Integración continua, ¿más que un proceso?..... Siii	5
Jenkins	6
Conclusión	8
Bibliografía	9



Resumen

En un momento de la historia, desarrollar un software empezaba a ser cada vez más dificultoso, por el simple hecho de que era muy difícil controlarlo cada vez que escalaba (crecía), para lo cual surgió la idea (práctica) de realizar integraciones constantes (continuas).

Dicho informe busca tocar temas a tener en cuenta para implementar dicha práctica, como así también, posibles beneficios.



Introducción

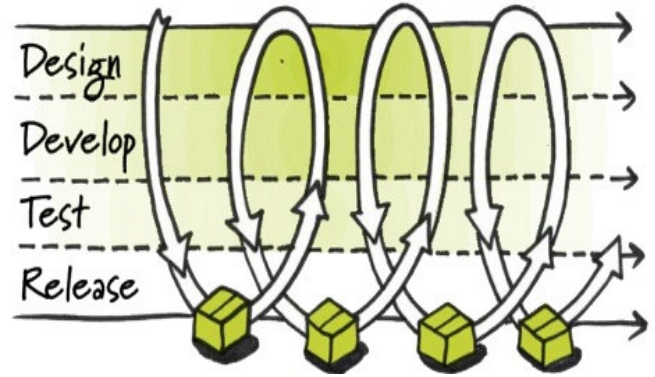
En el momento del desarrollo de un software, el equipo de trabajo se enfrentan a distintos problemas, de los cuales, uno de ellos (no significa que sea el más impórtate ni el menos) es como mantener “organizado” dicho desarrollo, es ahí cuando aparece la Integración Continua y nos propone diversas prácticas, responsabilidades y como poder hacerlo.

Poder implementar dicha práctica no es de manera sencilla, para lo cual, se comentará en unas de las herramientas más utilizada denominada Jenkins.

Desarrollo

¿Qué es la Integración Continua?

La integración continua es una práctica de desarrollo de software donde los miembros de un equipo integran su trabajo con frecuencia, por lo general, cada persona se integra al menos diariamente, lo que lleva a múltiples integraciones por día. Cada integración se verifica mediante una compilación automatizada (incluida la prueba) para detectar errores de integración lo más rápido posible. Muchos equipos encuentran que este enfoque conduce a problemas de integración significativamente reducidos y permite que un equipo desarrolle software cohesivo más rápidamente. Martin Fowler¹

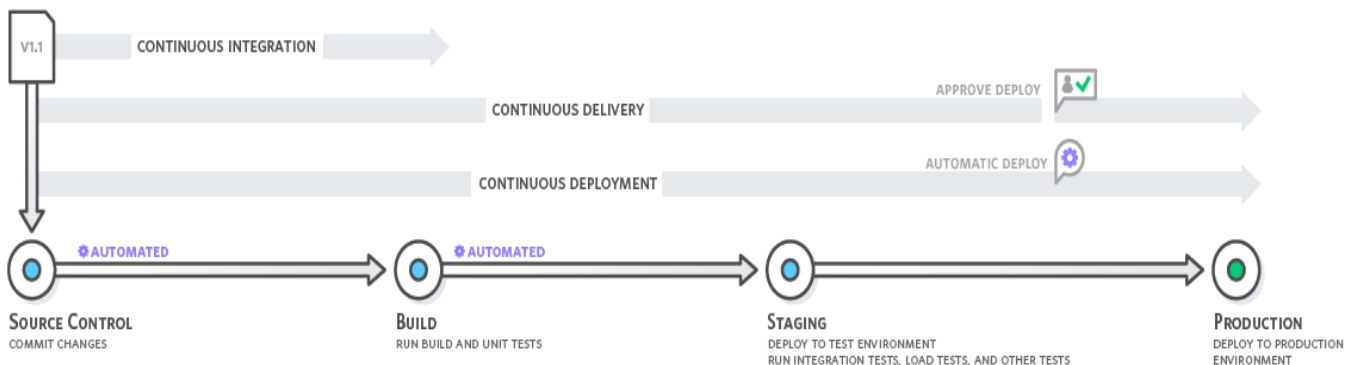


Es decir, que las personas involucradas en un proyecto junten su trabajo de manera automática y sistematizada para poder garantizar que la aplicación se tiene siempre funcional en cualquier momento del desarrollo.

Se dice que el acto de juntar el trabajo es a lo que se define como integración y es continua porque se trata que los miembros del equipo junten su trabajo continuamente, por ejemplo cada vez que termine de arreglar un bug (error o fallo), implementar una característica, un módulo de la aplicación, etc.

¿En qué consiste?

Los desarrolladores envían los cambios de forma periódica a un repositorio compartido con un sistema de control de versiones como Git. Antes de cada envío, los desarrolladores pueden elegir ejecutar pruebas de unidad local en el código como medida de verificación adicional antes de la integración. Un servicio de integración continua crea y ejecuta automáticamente pruebas de unidad en los nuevos cambios realizados en el código para identificar inmediatamente cualquier error.





¿Qué gano con implementarlo?

- **Mejore la productividad de desarrollo:** La integración continua mejora la productividad del equipo al liberar a los desarrolladores de las tareas manuales y fomentar comportamientos que ayudan a reducir la cantidad de errores y bugs enviados a los clientes.
- **Encuentre y arregle los errores con mayor rapidez:** Gracias a la realización de pruebas más frecuentes, el equipo puede descubrir y arreglar los errores antes de que se conviertan en problemas más graves.
- **Entregue las actualizaciones con mayor rapidez:** La integración continua le permite a su equipo entregar actualizaciones a los clientes con mayor rapidez y frecuencia.

¿Qué necesito?

En el momento de implementar esta buena práctica, es necesario tener y saber manipular ciertas cosas como las aquí mencionadas:

- **Sistema de control de versiones (VCS):** Proporciona un método confiable para centralizar y preservar los cambios realizados en su proyecto a lo largo del tiempo.
- **Máquina virtual:** Para soluciones en el sitio, debe tener un servidor libre o al menos una máquina virtual. Para una máquina limpia para construir su sistema es de vital importancia.
- **Soluciones de herramientas de CI alojadas:** para evitar servidores o máquinas virtuales, puede buscar soluciones de herramientas de CI alojadas que ayuden en el mantenimiento de todo el proceso y ofrezcan una escalabilidad más sencilla.
- **Herramientas:** Si opta por la variante auto hospedada, deberá instalar una de las muchas herramientas de integración continua disponibles como Jenkins, TeamCity, Bamboo, etc.

Integración continua, ¿más que un proceso?..... Siii

La integración continua está respaldada por varios principios y prácticas importantes.

Las practicas

- Mantener un único repositorio de origen
- Automatiza la construcción
- Haga que su autocomprobación de compilación
- Cada commit debe basarse en una máquina de integración
- Mantén la construcción rápida
- Prueba en un clon del entorno de producción.
- Facilite a cualquiera obtener la última versión ejecutable
- Todos pueden ver lo que pasa
- Automatizar la implementación

Cómo hacerlo



- Los desarrolladores revisan el código en sus espacios de trabajo privados
- Cuando termine, confirme los cambios en el repositorio
- El servidor CI monitorea el repositorio y verifica los cambios cuando ocurren
- El servidor CI construye el sistema y ejecuta pruebas unitarias y de integración.
- El servidor CI lanza artefactos desplegados para pruebas
- El servidor CI asigna una etiqueta de compilación a la versión del código que acaba de compilar
- El servidor CI informa al equipo de la compilación exitosa
- Si la compilación o las pruebas fallan, el servidor de CI alerta al equipo
- El equipo soluciona el problema lo antes posible
- Continuar integrando y probando continuamente durante todo el proyecto

Responsabilidades del equipo

- Registrarse frecuentemente
- No verifique el código roto
- No verifique el código no probado
- No te registres cuando la construcción esté rota
- No se vaya a casa después de registrarse hasta que el sistema se construya

Muchos equipos desarrollan rituales en torno a estas políticas para puedan administrarse de manera efectiva y autónomas (sin que estén esperando directivas o políticas de superiores)

Jenkins

Jenkins es un servidor de integración continua, gratuito, open-source y actualmente uno de los más empleados para esta función. Además es muy fácil de utilizar.

Esta herramienta, proviene de otra similar llamada Hudson, ideada por Kohsuke Kawaguchi, que trabajaba en Sun. Unos años después de que Oracle comprara Sun, la comunidad de Hudson decidió renombrar el proyecto a Jenkins, migrar el código a Github y continuar el trabajo desde ahí. No obstante, Oracle ha seguido desde entonces manteniendo y trabajando en Hudson.



Jenkins

























Una pregunta clave... ¿Para qué sirve Jenkins dentro del proceso de Integración Continua?

Su base son las tareas, donde se indica que es lo que hay que hacer en un build. Por ejemplo, que compruebe el repositorio de control de versiones cada cierto tiempo, cada vez que se suba un código del control de versiones, este se compile y se ejecuten las pruebas.

Si se produce alguna falla (resultado no esperado o hay algún error), Jenkins notificara al desarrollador, al equipo de QA, por algún medio (como por email) para que lo solucionen. Si es correcto el build, podemos indicar a Jenkins que lo integre y lo suba al repositorio.

No solamente sirve para mantener un control del código, sino, como se indicó anteriormente, sirve para enlazar todo el proceso de desarrollo, entre ellas, muestre métricas de calidad, ver resultados de test, generar y visualizar la documentación del proyecto incluso pasar una versión estable del software al entorno de QA para ser probado, a pre-producción o producción.

Por ejemplo, en esta imagen puedes ver una zona del cuadro de mando de Jenkins, donde aparecen las distintas tareas que se han llevado a cabo, la duración, cuándo se produjo el último fallo...

All +						
S	W	Name ↓	Last Success	Last Failure	Last Duration	
		apicavh-ff-firefox36-staging	45 min (#1)	N/A	5.2 sec	
		apicavh-ff-update	2 hr 49 min (#5)	8 days 5 hr (#5)	3.7 sec	
		bargaatch-ie-staging	8 hr 40 min (#2)	20 days (#11)	41 sec	
		buildss-update	43 min (#20)	N/A	3 sec	
		hg_plaround-staging	24 days (#8)	N/A	4.8 sec	
		ienotifr-ie-staging	8 hr 42 min (#15)	N/A	27 sec	
		iesetuper-ie-staging	N/A	16 days (#5)	2.4 sec	
		libbhoer-ie-staging	8 hr 42 min (#9)	20 days (#2)	39 sec	

Para poder utilizar Jenkins, es imprescindible tener un repositorio de control de versiones (mercurial, git, svn, plastic, etc). Todo lo necesario para realizar el build debe estar allí (código, scripts de test, librerías de terceros...).



Conclusión

Para ir finalizando dicho informe, podemos decir que para poder entregar Software de calidad es necesario poder contar con distintas prácticas como es la Integración Continua.

En ella nos propone ir actualizando en un repositorio el código de nuestra aplicación, para poder mantener siempre la última versión (y porque no, versiones anteriores) funcional (reléase) del software, pero para ello, debe superar distintas pruebas (tests) que pueden ser de manera local (en el mismo lugar de desarrollo) o mediante distintas herramientas (como el mencionado Jenkins).

Poder poner en práctica dicho dilema, incluye tener un equipo dispuesto a ser disciplinado y comprometido en su labor. Es un gran esfuerzo por parte del equipo, pero que a la corta, se empieza a ver sus beneficios.

Bibliografía

- <https://ocw.unican.es/pluginfile.php/1408/course/section/1805/tema10-comoEstructurarUnInformeTecnico.pdf>
- <https://www.youtube.com/watch?v=58SjUKY6sKk>
- <https://www.martinfowler.com/articles/continuousIntegration.html#IntroducingContinuousIntegration>
- <https://www.thoughtworks.com/continuous-integration>
- <https://www.edureka.co/blog/continuous-integration/#WhatIsContinuousIntegration?>
- <https://aws.amazon.com/es/devops/continuous-integration/>
- <https://www.javiergarzas.com/2014/08/implantar-integracion-continua.html>
- <https://codigofacilito.com/articulos/integracion-continua>
- <https://www.javiergarzas.com/2014/05/jenkins.html>