¿Que es la integración continua?

Para poder hablar de sus ventajas, lo primero es conocer en qué consiste. La integración continua es el nombre que se le da a la automatización de las labores de compilación, test y análisis estático del código. Esto se puede corregir de muchas maneras y podemos llamar integración continua a todo lo que hay entre un script que periódicamente ejecuta el trabajo y un servicio online que lo haga, es decir, consiste en detectar errores más rápido mediante pruebas, mejorar la calidad del software y reducir el tiempo el actualizar y publicar versiones.

¿Cómo funciona?

Independientemente del sistema que utilicemos para iniciarlo, el proceso como mínimo seguirá el siguiente camino :

Codigo fuente --> Control de Versiones --> Build Automatico --> Pruebas Automatizadas --> Resultado y Reportes--> Código...

1.Descarga el código fuente desde el repositorio de control de versiones
2.Compilara el código según sea necesario
3.Realizará las pruebas unitarias y/o de integración
4.Publicará los resultados de modo que sean accesibles

(Dependiendo de lo integrado que esté el servicio con el repositorio de código, podemos verlo directamente en el historial de commits. Por ejemplo en el caso de GitHub)

¿Cuando debería ejecutarla?

Lo ideal, es ejecutarla siempre que se añadan cambios al repositorio principal. Pero eso tiene un coste. Compilar cada cambio que hacemos nos da una cierta seguridad, pero puede hacer que nuestro sistema de CI trabaje mas de la cuenta y suponga un coste adicional. En cambio, aplicara a todos los pull request nos garantiza que todos los cambios sobre la linea principal de trabajo van a funcionar bien. Aqui se trata de encontrar un equilibrio entre coste y beneficio.

Si nuestro proyecto es Open Source, existen varios servicios de integracion continua online gratuitos como (Travis CI, App
Veyor, Azure Pipelines, Cicle CI)

¿Y que me aporta?

·Detectar rapidamente los posibles errores de compilacion de nuestro codigo
·Detectar funcionamientos anomalos
·Mejorar la calidad
·Nos permite compilar/testear nuestro código en diferentes plataformas

En conclusión :

Con el gran abanico de posibilidades que existen hoy en día para poder añadir integracion continua a nuestro código, no hacerlo no es una opción. Da igual si es un proyecto grande o pequeño, da igual si eres un único desarrollador o un equipo.
Existen incluso algunas opciones gratuitas que nos permiten hacerlo.

¿Quién no ha oído nunca "en mi máquina funcionaba"? La integración continua es una solución muy eficaz para evitar este tipo de problemas y poder aportar valor con nuestro software sin tener que perder las horas y horas en compilar nuestro proyecto para varias plataformas, ejecutar pruebas, analizadores de código.

Aplicación a utilizar : Jenkins (Mejor Valorada)

Aplicaciones similares a conocer : CircleCI, Travis CI, Bitrise, Visual Studio App Center, TeamCity, GoCD, Bamboo, GitLab CI

Jenkins : Es un servidor de Integración Continua, open-source, escrito en java, el cual nos permite automatizar todos los procesos de integración y entrega mediante tareas,tareas que son extremadamente fáciles de crear y programar. Con Las tareas podemos monitorear cambios sobre un control de versiones, compilar código, ejecutar pruebas, vaya, podemos realizar todos los pasos necesarios para que el software esté listo para su puesta en producción.

Algo interesante a mencionar, es que si en algún momento alguna tarea falla, Jenkins podrá notificar al equipo de desarrollo al product Manager o cualquier responsable del error, de tal forma que se tomen cartas en el asunto y se llegue a una solución
lo más pronto posible.

Con Jenkins podemos obtener métricas sobre el software, cantidad de pruebas exitosas, cobertura de código documentación, tiempo de compilación , etc.

==========
==========
==========
==========

Contenido Extra

Integración continua es un tema el cual se ha vuelto muy popular en los últimos años, principalmente, por el impulso que ha tenido por parte de grandes empresas, tales como Netflix, Uber, entre otros; Las cuales basan su desarrollo en un enfoque

agilista.

Al nosotros hablar de integración continua comunmente, tambien hablaremos de entrega continua y despliegue continuo.

En esta ocasión estaremos explicando cada uno de estos temas y veremos las ventajas de su implementación.

En muchas ocasiones como desarrolladores estaremos haciendo deploy de nuestras aplicaciones de forma manual; Por ejemplo,
si hablamos de un sitio web, el primer paso es subir todos los cambios a nuestra rama principal, posteriormente ejecutar
todas las pruebas unitarias, si las pruebas son exitosas se procede ha compilar el proyecto y subir todos los cambios ha
produccion, ejecutar las migraciones y listo, los clientes ya podrán hacer uso de los nuevos features en el sitio web, claro
siempre y cuando ninguno de los pasos anteriores hayan fallado.

Todos estos pasos a menudo son tediosos e involucran una cierta cantidad de tiempo, tiempo en el que regularmente el
desarrollador no tiene mucho que hacer. Para evitar estos problemas surge la integración, entrega y despliegue continuo.

·Integración Continua
·Entrega Continua
·Despliegue Continuo

Entrega Continua : no es mas que una extension de integracion continua. Al realizar un commit sobre la rama principal se
ejecutaran (de manera automatica)todos los pasos necesarios para hacer que el proyecto se libere a produccion. Se compila
el codigo, se ejecutan pruebas unitarias, se ejecutan migraciones. Todo esto en un ambiente que no es produccion, es decir
entrega continua permite que nuestro proyecto este listo para que el usuario final haga uso de el, sin embargo, despliegue
a produccion no se realiza. Este pequeño y ultimo paso (realizar el deploy)sera ejecutado de forma manual, esto con la
finalidad de tener un control sobre cuando liberar el producto. Habra ocasiones en las cuales sera necesario postergar la
liberacion del software, ya sea por estrategias de marketing, peticion del cliente, etc...

La gran ventaja de Entrega Continua es que nos permite reducir la cantidad de errores al momento de realizar el deploy

de la aplicacion, puesto que ya todos los pasos para la liberacion fueron ejecutados anteriormente.

Podemos concluir que el objetivo de entrega continua no es colocar el producto en produccion, sino mas bien que el software
este displonible para su puesta en produccion en cualquier momento.

Despliegue Continuo : Si en entrega Continua nos quedamos a un solo paso para que el usuario final utilice el producto, con
despliegue continuo todo el ciclo de entrega se completa.

Utilizando despliegue continuo al realizar un commit sobre la rama principal, se ejecutan todos los pasos necesarios
(de forma automatica) para el despliegue del proyecto a produccion. Esto se traduce en que el usuario final estara utilizando
un producto en constante actualizacion.

Para implementarlo sera necesario haber implementado de forma correcta Integracion Continua.