

Pilares de CI/CD

Índice

- 01 [Integración continua y testing](#)
- 02 [Test de integración continua](#) 03
- [Jenkins, pipelines y jobs](#)



01

Integración continua y testing

Integración continua

CI y CD son términos comúnmente utilizados por los equipos de desarrollo.

Cuando hablamos de **integración continua (CI)** se trata de un proceso automatizado para integrar continuamente los cambios que se producen durante el desarrollo del software (construcción, prueba y validación del código). Con la ayuda de las funciones de CI se aceleran los ciclos de liberación del código evitando los conflictos de unión.

La IC requiere un sistema de control de versiones que pueda rastrear los cambios y las versiones del código del software. En este contexto, se suele utilizar Git. Así, los desarrolladores trabajan primero en su máquina

La implementación de CI puede ser el primer paso para producir código de alta calidad en sus equipos, debido a las validaciones. Hay muchas herramientas de integración continua disponibles, entre las más populares está Jenkins.

El producto final de la integración continua es una versión candidata, la forma final de un artefacto para ser desplegado. Pueden darse pasos de calidad para producir el artefacto, como la búsqueda de errores y la identificación de su solución. El empaquetado, la distribución y la configuración entran en una versión candidata. De esta manera, la IC alivia a los equipos de la carga de las construcciones manuales y también hace que las construcciones sean más repetibles, consistentes y disponibles.

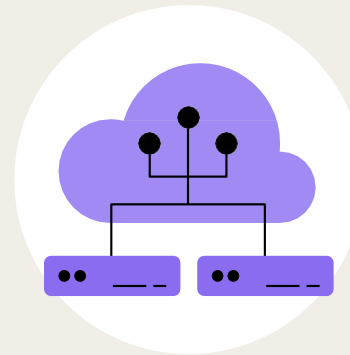
Pilares de la CI

Las Pruebas de Integración Continua sirven como puerta de entrada a la calidad durante cada uno de los tres pilares de CI, construir, empaquetar y publicar artefactos.



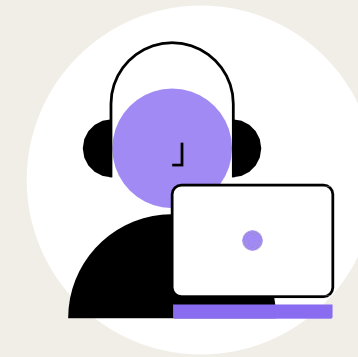
Construir

- Ejecutar herramientas de compilación dependientes del lenguaje, como en el ecosistema JAVA, como Maven/Gradle, o en el ecosistema NodeJS, como NPM. Su compilación combina su código escrito a medida con dependencias de terceros.



Probar

- Como el artefacto aún no ha sido desplegado, las pruebas unitarias son excelentes para ejecutar en el proceso de CI. A medida que se construye el artefacto/unidad desplegable, las comprobaciones centradas en el artefacto, como las de código abierto y de dependencias, son esenciales para encontrar vulnerabilidades y posibles riesgos de licencia.



Publicar

- La publicación en un repositorio/registro de artefactos se produce después de la creación de la unidad de artefacto/implante.

02

Test de integración continua

Test de integración continua

Los tests de integración continua son aquellos enfocados y ejecutados durante el proceso de CI y orquestados por las herramientas de CI, las cuales, son responsables de construir, empaquetar y publicar artefactos.

Las pruebas en el proceso de IC **permiten una rápida retroalimentación** y, por diseño, **detienen la progresión del artefacto si no se cumple la calidad mínima**. Normalmente, las pruebas de CI se centran en el artefacto antes del despliegue en el primer entorno de integración.

Cuando el artefacto pasa las pruebas de integración continua, significa que se ha alcanzado un nivel de calidad antes de la liberación, no sólo que tiene la capacidad de ser construido de forma consistente.

Es importante señalar que una compilación fallida es mucho menos grave que un despliegue fallido y que es perfectamente normal que se encuentren errores y correcciones en las pruebas de CI.

Test de IC

Estas pruebas se orquestan en una cadena y se dividen en dos categorías: pruebas dentro y fuera del proceso.

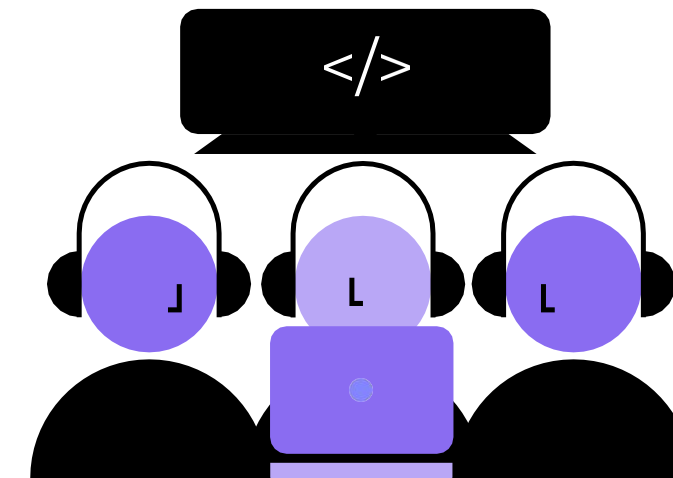
- **Test de calidad del código.** Las herramientas de calidad del código, como SonarQube y Checkmarx, se centran en el análisis estático del código y son prudentes durante los cambios de código.
- **Pruebas unitarias.** Las pruebas unitarias se centran en bloques/métodos de código que han cambiado, están diseñadas para ser granulares y se ejecutan juntas. Por lo general, se crean objetos simulados y se comprueban las afirmaciones.
- **Prueba de integración.** En el caso de las pruebas de integración en el contexto de la integración continua, se centrará en probar módulos cruzados de la aplicación. Las herramientas modernas de pruebas unitarias y de integración se solapan mucho. JUnit puede utilizarse para realizar pruebas unitarias y de integración, por ejemplo.
- **Prueba de seguridad/licencia.** El objetivo es encontrar la exposición y el riesgo de utilizar determinados paquetes. Las herramientas de análisis de seguridad/licencias como Blackduck, Snyk y StackHawk tienen diferentes métodos de introspección: algunos requieren que se ejecute el artefacto terminado,

03 Jenkins, pipelines y jobs

Jenkins

Jenkins es una **solución de código abierto** que cuenta con el apoyo de proveedores, principalmente Cloudbees.

Jenkins es un servidor de Integración Continua de código abierto escrito en Java para orquestar una cadena de acciones para lograr el proceso de Integración Continua de forma automatizada, soportando el ciclo de vida completo del desarrollo de software.



Pipeline

Un pipeline es una serie de pasos orquestados con la capacidad de llevar el código fuente a producción. Las etapas incluyen:

- Construcción
- Envasado
- Pruebas
- Validación
- Verificación de la estructura
- Despliegue

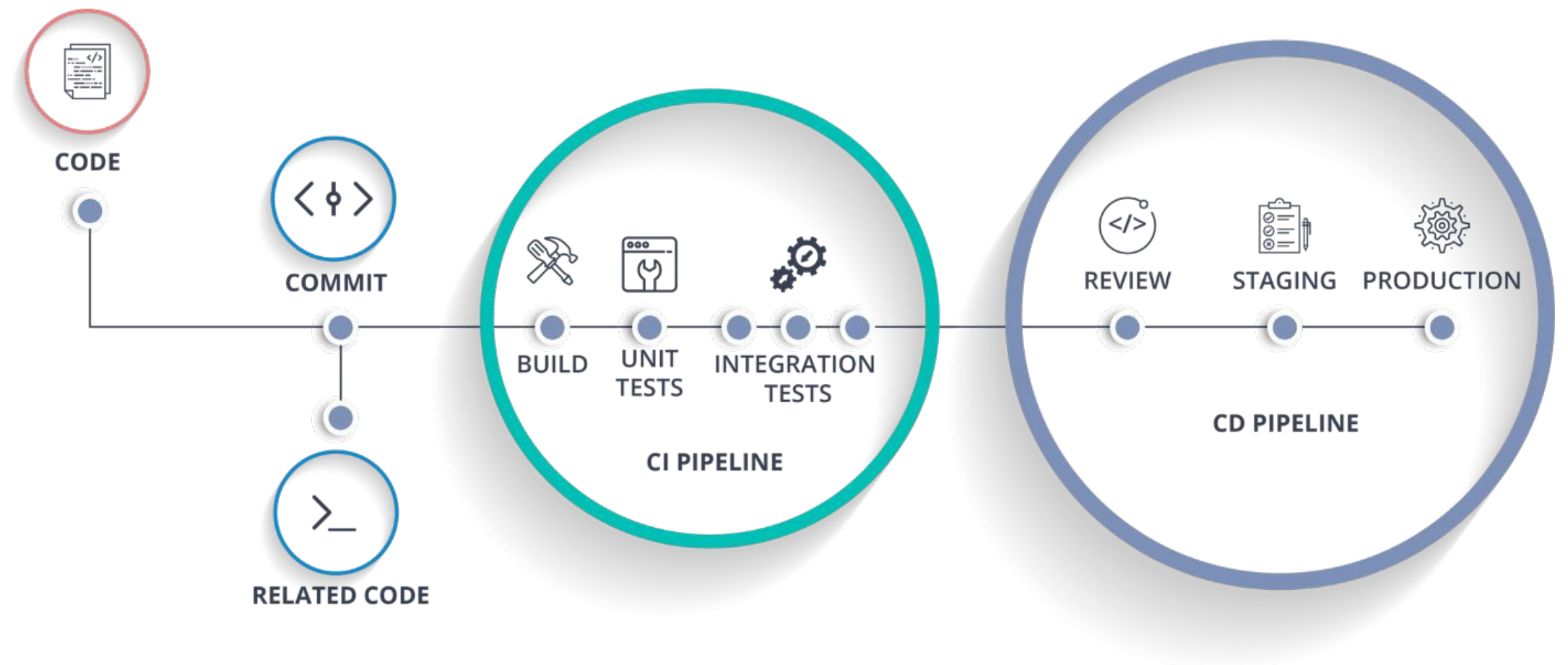
Dependiendo de las estructuras organizativas y de los equipos, se requiere el uso de varios conductos para lograr este objetivo.

Pipeline predica la continuidad, es decir, **incluye la integración continua, la entrega/despliegue continuo (CI/CD), el feedback continuo y las operaciones continuas**. Así, en lugar de pruebas puntuales o despliegues programados, cada función se produce continuamente.

Jobs

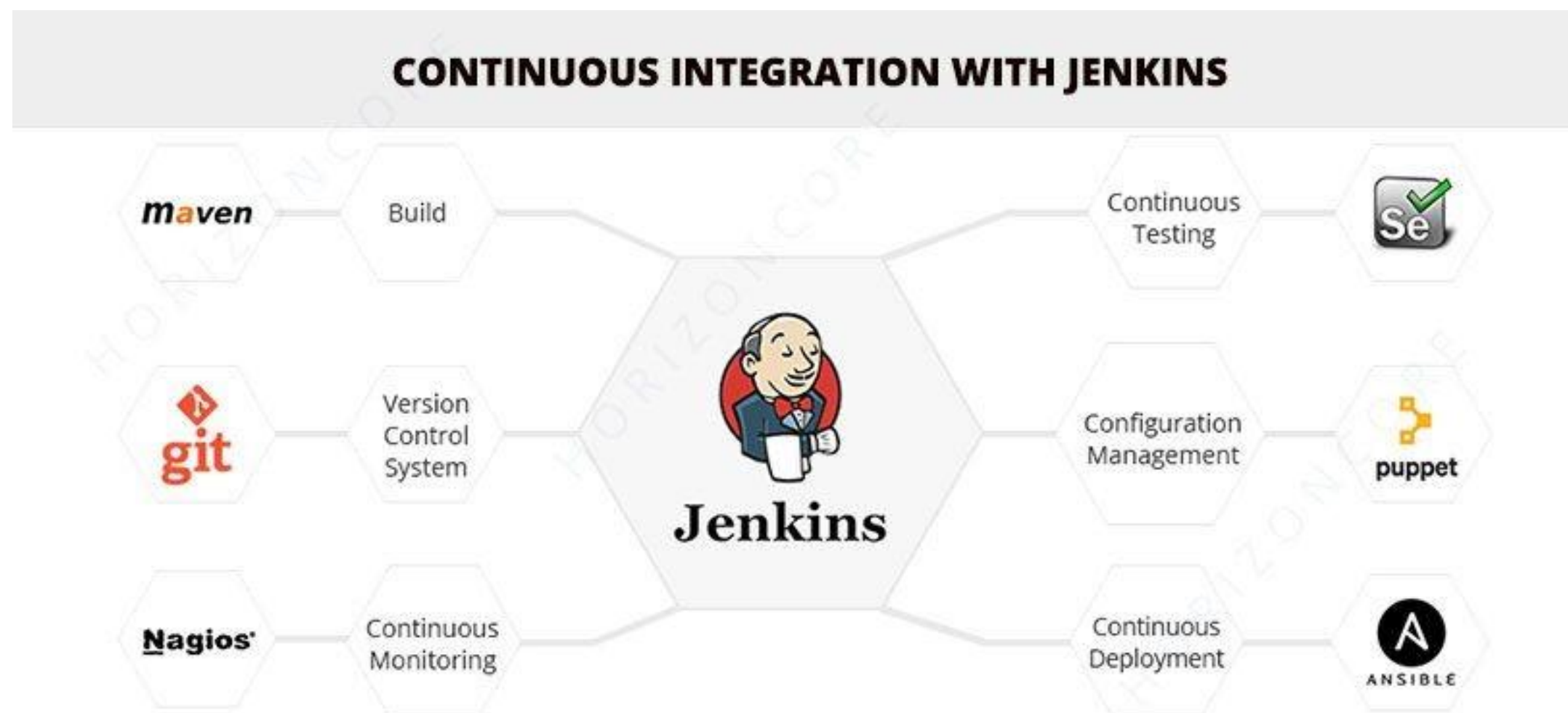
El trabajo se refiere a cada tarea dentro de la tubería, estas tareas pueden ser interdependientes y hacer llamadas cuando se empuja, a través de la programación o manualmente. Construir es cada una de las ejecuciones de Job.

El CD de flujo completo de CI:



Jobs

Ver la relación entre las herramientas de CI y las actividades:



¡Muchas
gracias!