ANSIBLE PARA DEV+OPS

DÍA 3 – PARTE I

QUE VEREMOS HOY

- Integración continua / Despliegue contínuo (CI / CD)
- Interacción con Jenkins
- Labs:
  - TDD + integración continua

INTEGRACIÓN CONTINUA

- Integración Continua (CI)
- Despliegue Continuo (CD)
- La suma de todo

INTEGRACIÓN CONTINUA

- Es el proceso de hacer merge a la rama de producción varias veces al día.
- Se basa en introducir pequeños cambios de forma rápida para evitar el "integration hell"
- Estos cambios deben estar probados de antemano para asegurar que no rompen nada

INTEGRACIÓN CONTINUA

1. El desarrollador toma una copia del código en el que quiere hacer cambios
2. Crea una rama específica para su cambio
3. Implementa el cambio
4. Prueba su cambio en local
5. Sube su cambio a la rama de integración
6. El sistema de CI valida que funciona (mediante tests automáticos) y hace merge a la rama de Producción

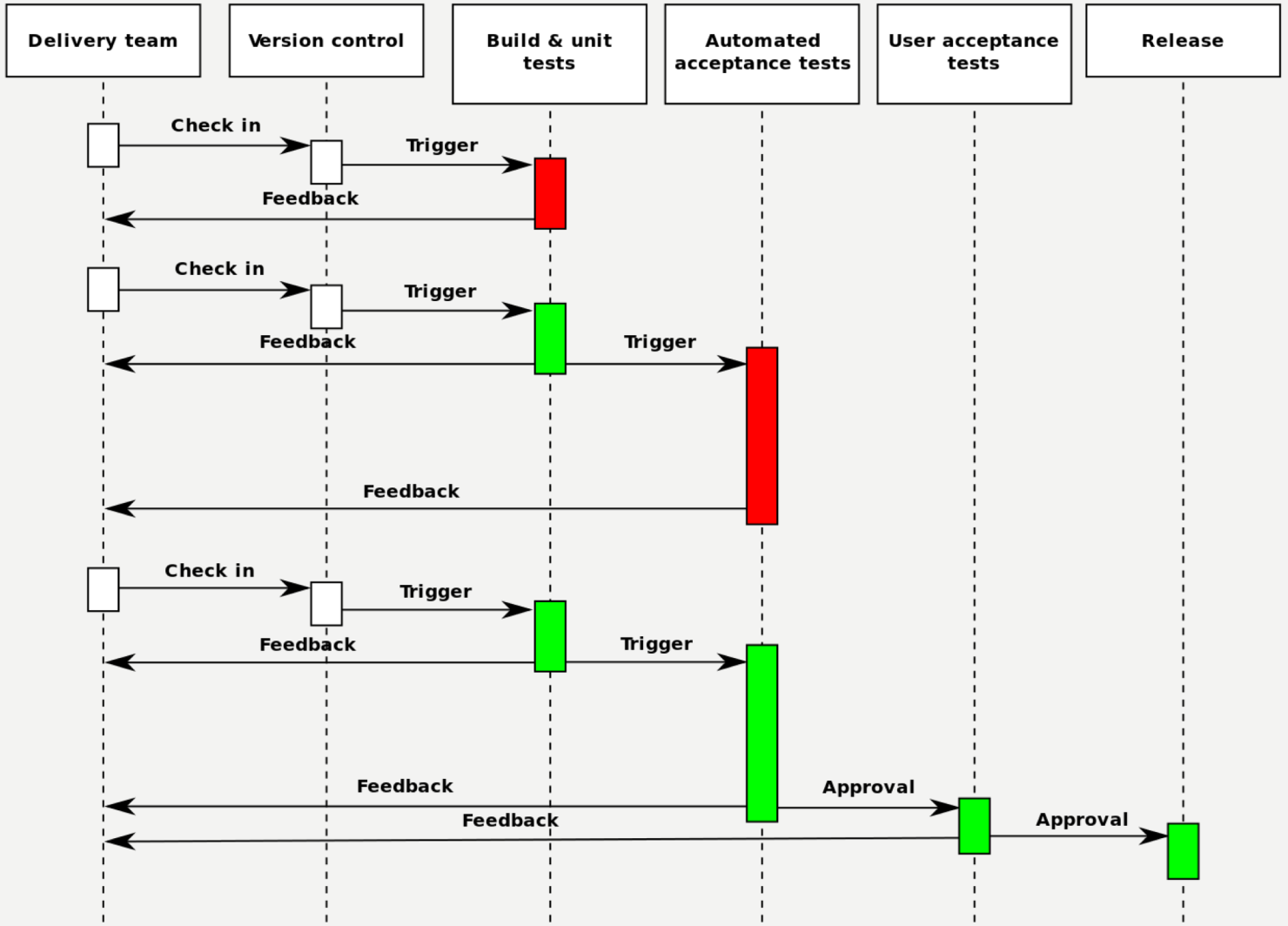
INTEGRACIÓN CONTINUA

- Es extremadamente importante que el tiempo entre 1 y 5 sea lo más corto posible, de lo contrario pueden haber aparecido nuevos cambios en integración / producción que hagan conflicto con el desarrollo que se quiere implementar

DESPLIEGUE CONTINUO

- Es la capacidad de poder desplegar nuevas versiones de un desarrollo en cualquier momento
- Requiere un proceso de despliegue repetible y confiable para ser implementado

DESPLIEGUE CONTINUO



+ Y – DE CI/CD

- +
  - Se reduce el tiempo de salida al mercado
  - Se construye el producto que se necesita (cada funcionalidad es una petición y cada petición se traduce en una parte de código individual)
  - Ahorro de tiempo gracias a la automatización
  - Mejora de calidad de los entregables (al usar procesos repetitivos validados)

+ Y – DE CI/CD

- -
  - Algunas organizaciones no pueden tolerar cambios constantes (especialmente en algunas épocas del año)
  - Algunos sectores requieren de validaciones muy burocráticas que pueden entorpecer el proceso
  - La falta de automatización en los tests impide la existencia de CI/CD
  - Diferencia entre entornos: CI/CD requiere que todos los entornos sean iguales para evitar que problemas no detectados acaben en producción
  - Tests no automatizables: si existen tests no automatizables habrá que integrar validación humana en el pipeline , lo que retrasará el proceso

INTERACCIÓN CON JENKINS

- Introducción a Jenkins
- Tipos de Jobs
- Pipelines de Jenkins

INTRODUCCIÓN A JENKINS

- Sistema de CI/CD escrito en Java
- Muy extendido
- Con gran cantidad de plugins
- Funciona definiendo jobs que pueden realizar tareas variadas
- Las tareas se ejecutan en Build Servers, que ejecutan un agente de Jenkins

TIPOS DE JOB

- Freestyle
- Maven Project
- Pipeline
- External job
- Multi conf project
- Multi branch pipeline

PIPELINES DE JENKINS

- Jobs definibles programáticamente con una DSL en Groovy
  https://jenkins.io/doc/pipeline/steps/
- Todos los pasos que se pueden hacer en jobs freestyle de Jenkins se pueden implementar en pipelines
- Ciertos plugins pueden extender la DSL
- Permiten definir procesos de CI / CD de forma programática
- Permiten pasos en paralelo

LABS

- Instalación de Jenkins
- Integración con github
- Preparación de Jenkins para poder ejecutar nuestros comandos de test
- Desarrollo del pipeline de CI/CD para nuestros playbooks
- Prueba

No subestimes este lab ;)

POR SI SOBRA TIEMPO…

- Alternativas a Jenkins
  - TravisCI
  - GitlabCI
  - AWS CodePipeline
- Alternativas a Github
  - Gitlab
  - gogs.io
- El CI/CD en el mundo del IaaS