

Jenkins: CI/CD pour DevOps

ULRICH MONJI

Plan

Présentation du formateur

Introduction au DevOps et au CI/CD

Jenkins

Installation et Découverte

Build

Test d'acceptance

Artefact

Déploiement

Test de fonctionnement

Shared Library

Sécurité

Mini-projet

Plan

Présentation du formateur

Introduction au DevOps et au CI/CD

Jenkins

Installation et Découverte

Build

Test d'acceptance

Artefact

Déploiement

Test de fonctionnement

Shared Library

Sécurité

Mini-projet

Ulrich MONJI - Ingénieur en Systèmes, Réseaux et Telecommunications - UTT



- Atos-Worldline - Ingénieur Système
 - Build et Run de plateforme Cloud
 - Virtualisation - Stockage - Automatisation
 - Comptes clients: Carrefour, Auchan, ARJEL, SAMU
- Adneom - Consultant IT
- Groupe SII - Consultant IT (Cloud/Devops)
 - Consultant chez Orange France
 - Exploitant sur le PaaS Erable (Offre PaaS d'Orange)
 - Migration d'une application monolithique en microservice
- Formateur chez Eazytraining

Plan

Présentation du formateur

Introduction au DevOps et au CI/CD

Jenkins

Installation et Découverte

Build

Test d'acceptance

Artefact

Déploiement

Test de fonctionnement

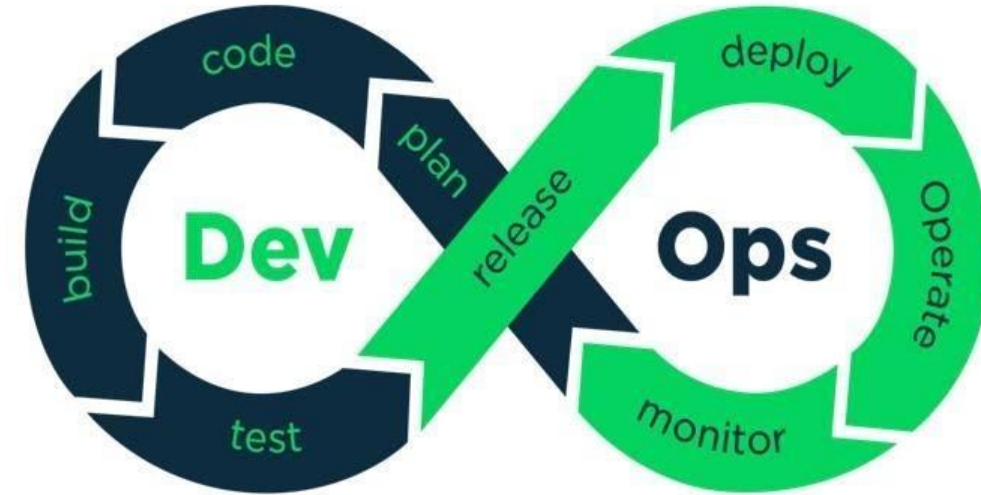
Shared Library

Sécurité

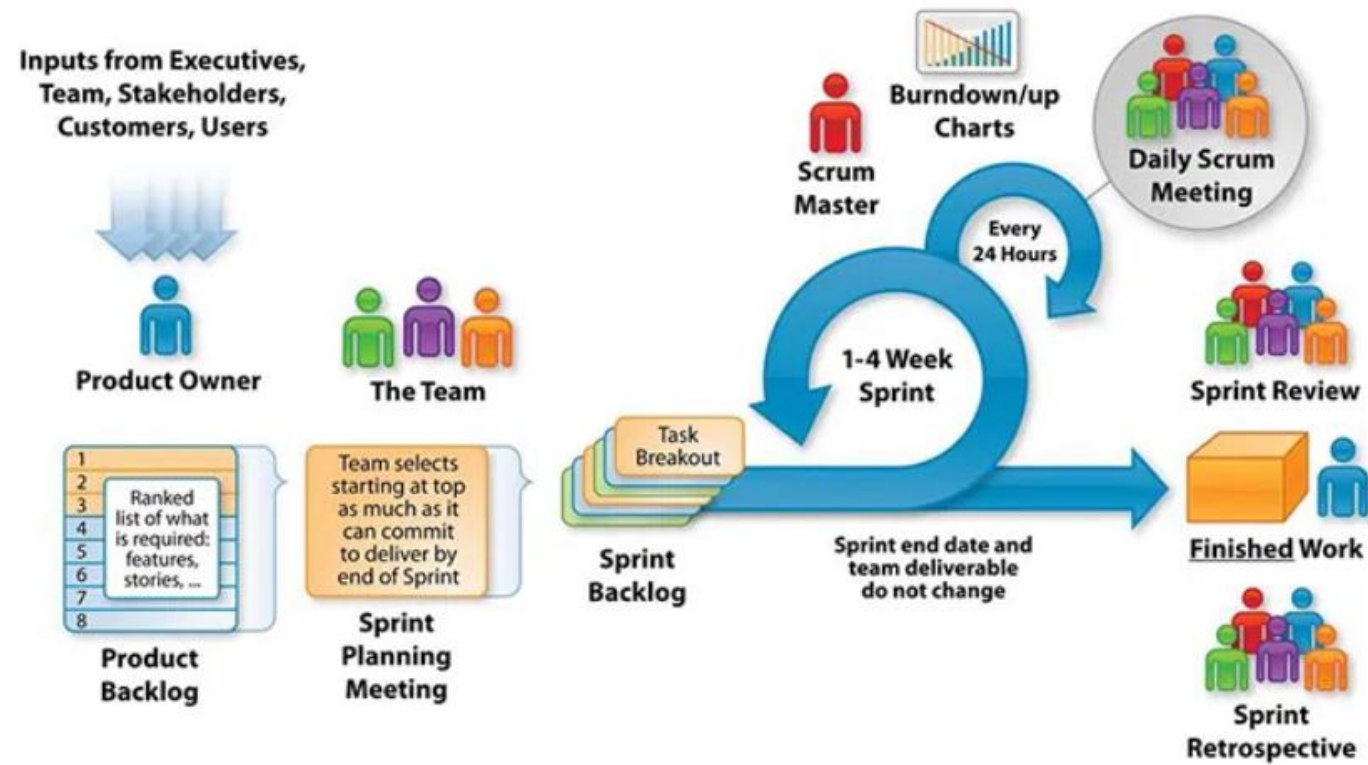
Mini-projet

Intro au DevOps et à la CI/CD : Le DevOps

Agile vs. DevOps



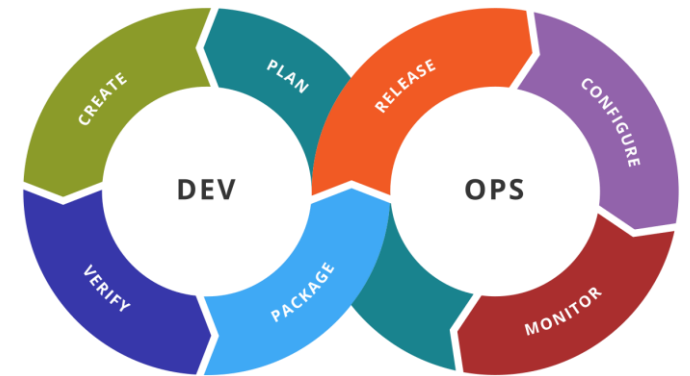
- Agile: méthode de développement
- DevOps: agilité entre Dev et Ops = CI + CD

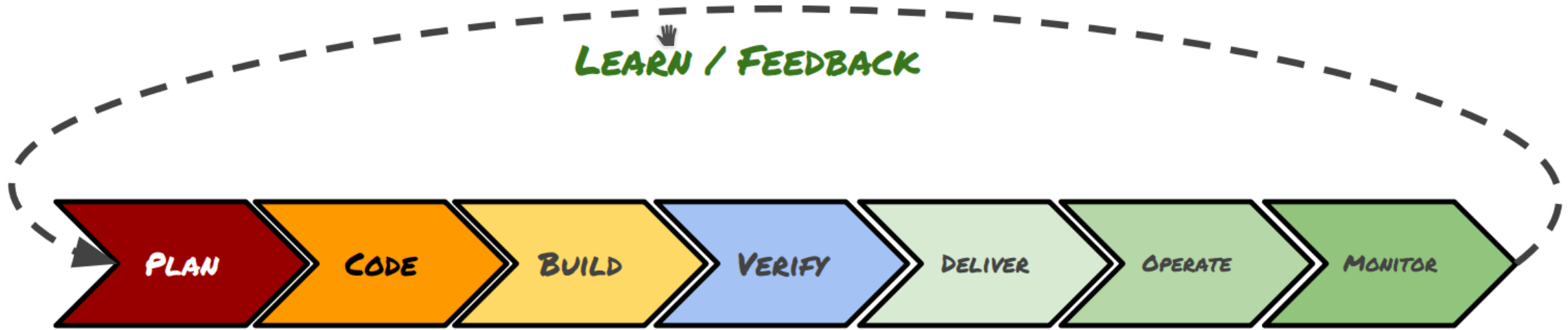


METHODOLOGIE SCRUM

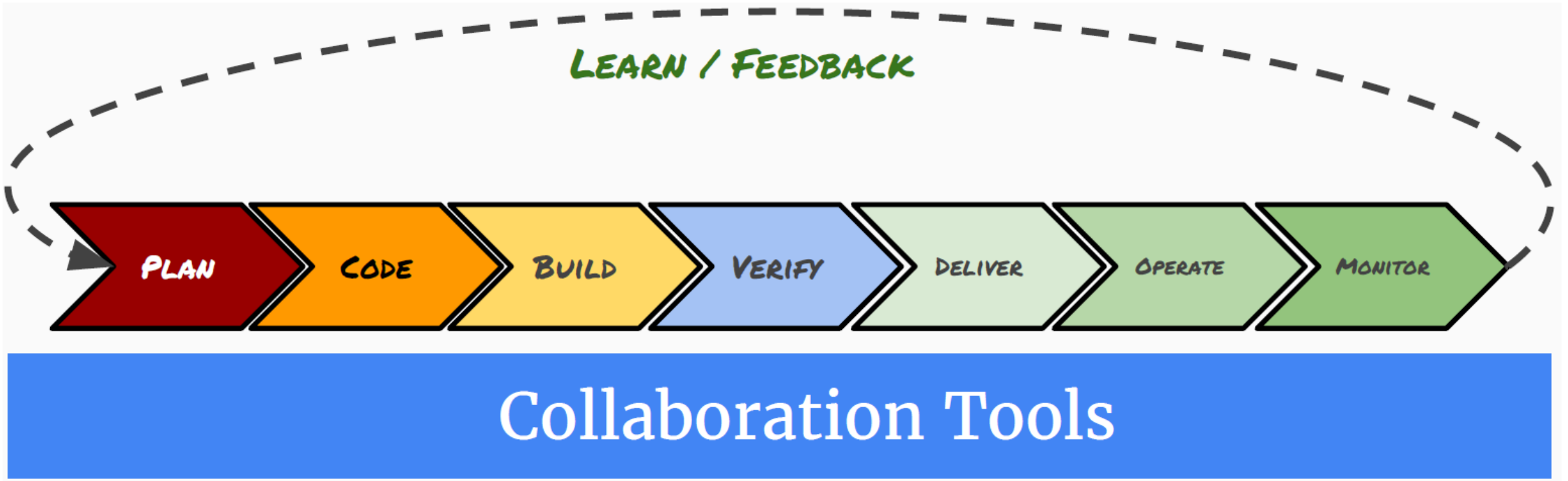
DevOps : qu'est-ce que c'est ?

- C'est un mouvement, une pratique technique visant à l'alignement des développeurs (**Dev**) et des opérationnels (**Ops**) vers des objectifs communs.
- Les **Ops** désignent de façon générale les administrateurs système, base de données, réseaux, stockage, les exploitants, etc ...
- Le DevOps est une **manière de travailler**, et les outils disponibles viennent promouvoir cette manière de travailler.
- Parlant de DevOps, les vocabulaires suivants sont incontournables : l'automatisation à travers l'**Infrastructure as Code**, les chaînes d'intégration continue et de déploiement continue (**CI/CD**), les **tests** unitaires, les méthodes **agiles**, etc ...

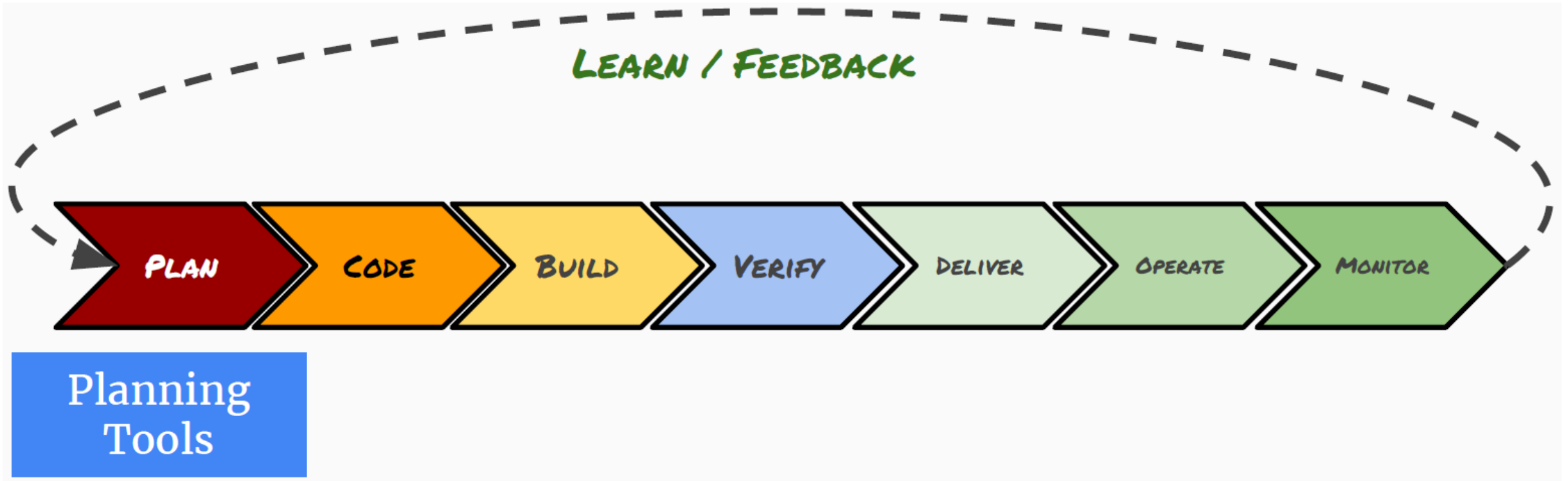




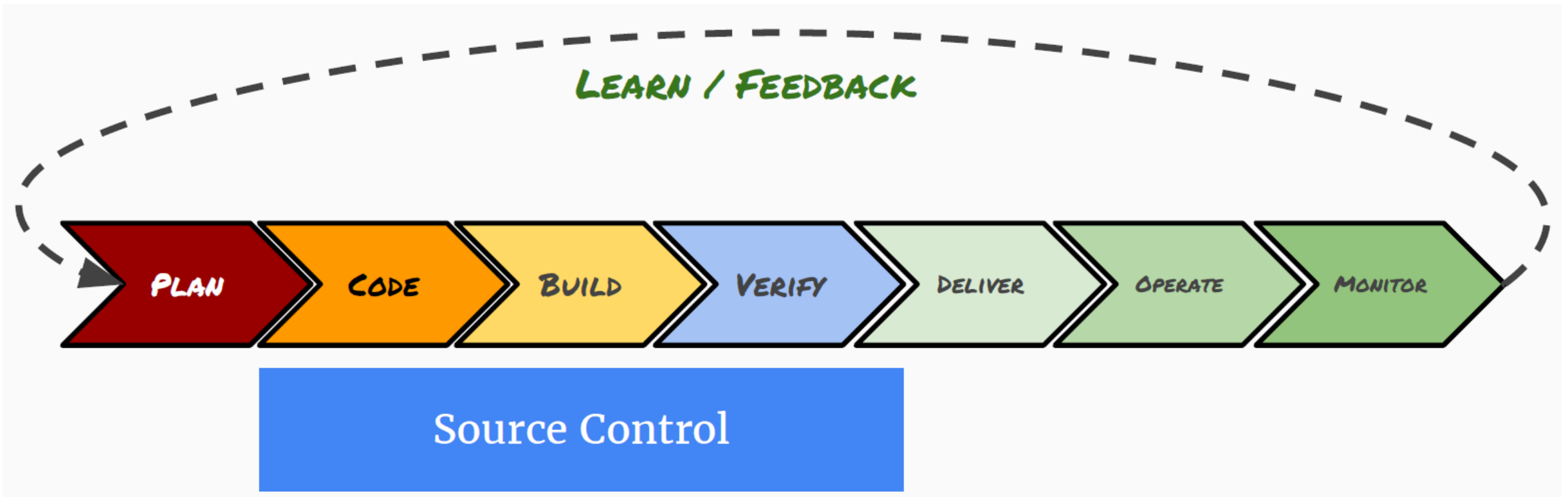
DevOps : Cycle de vie et CICD



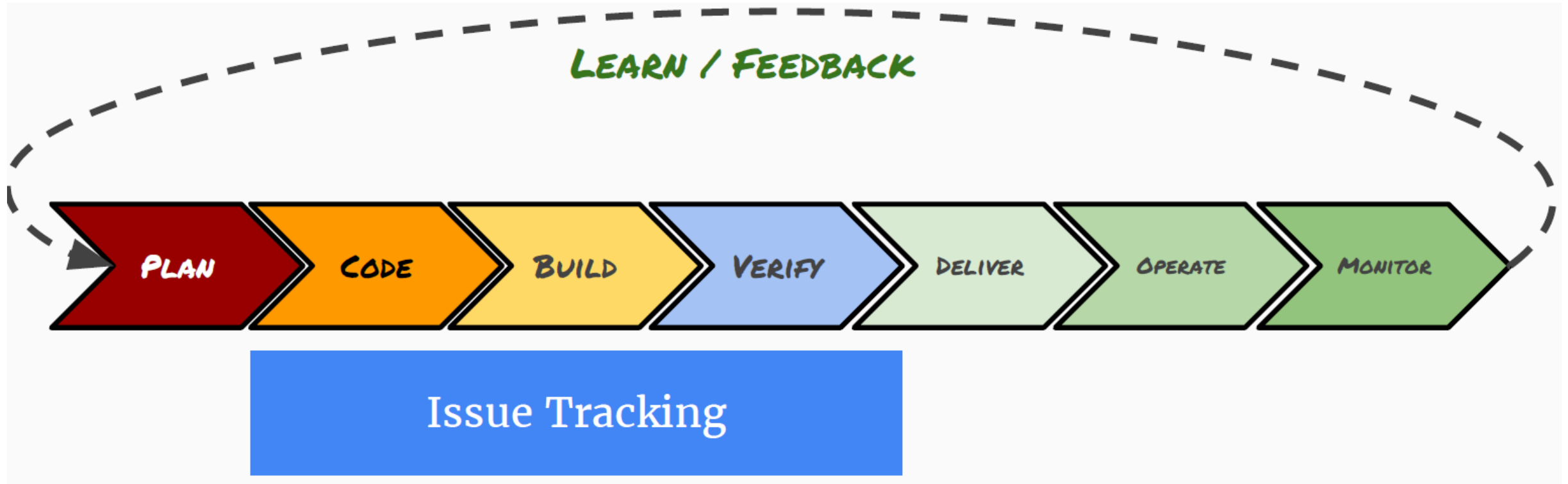
Teams, Slack, Mattermost etc ...



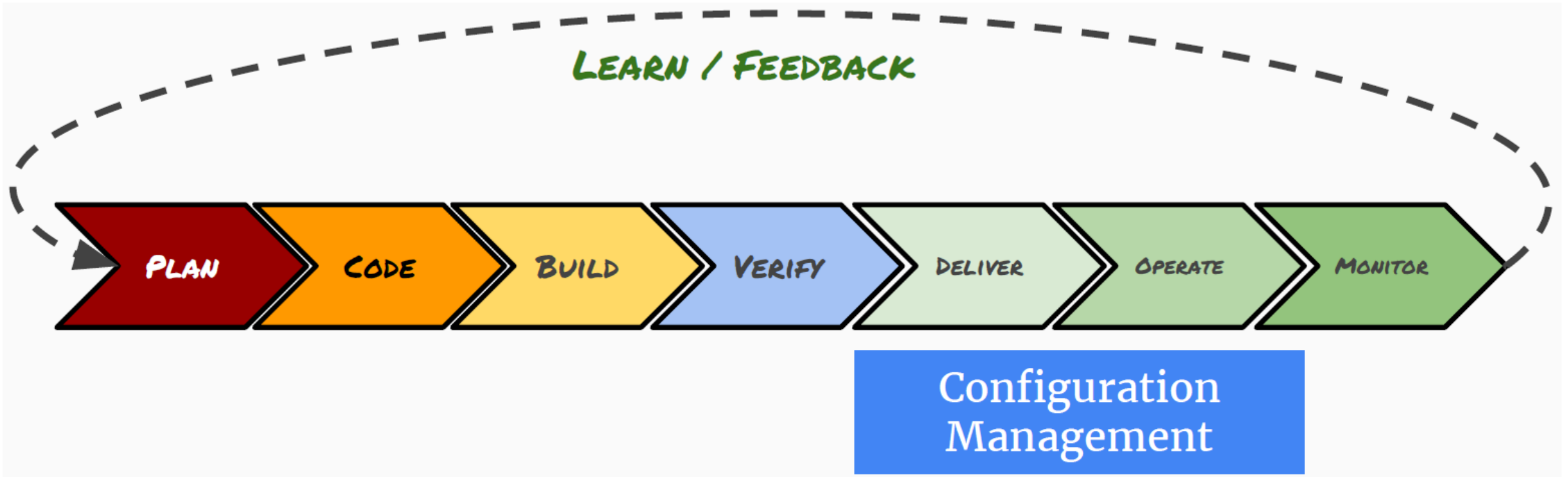
Jira, Confluence, Taiga etc ...



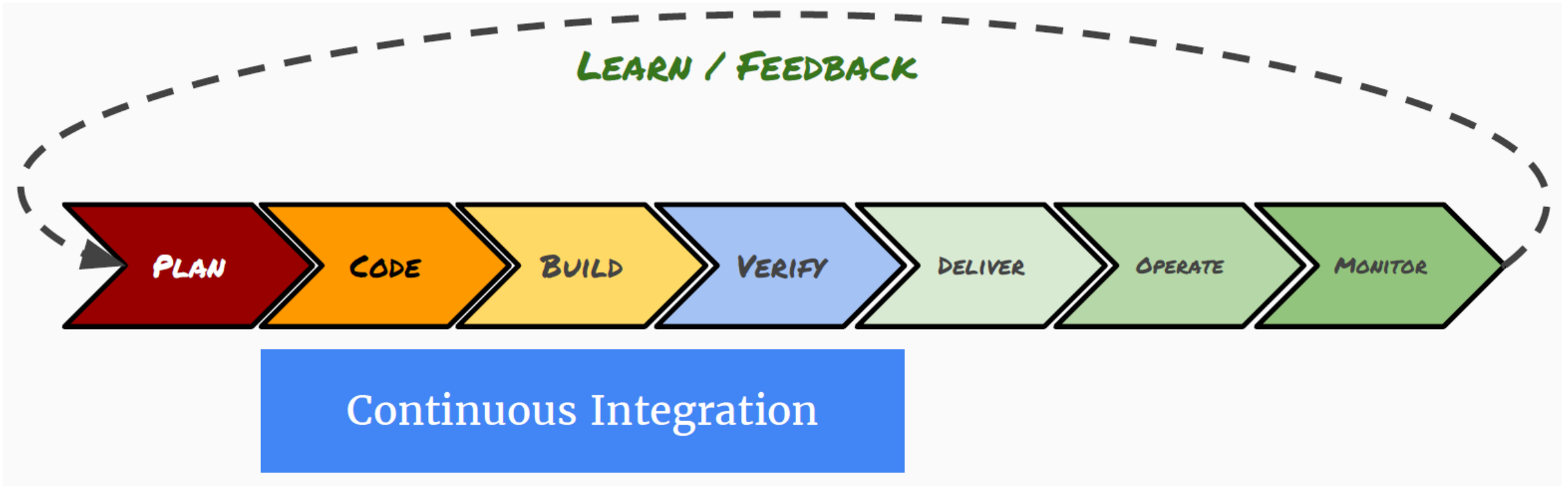
Git, Github, Gitlab etc ...



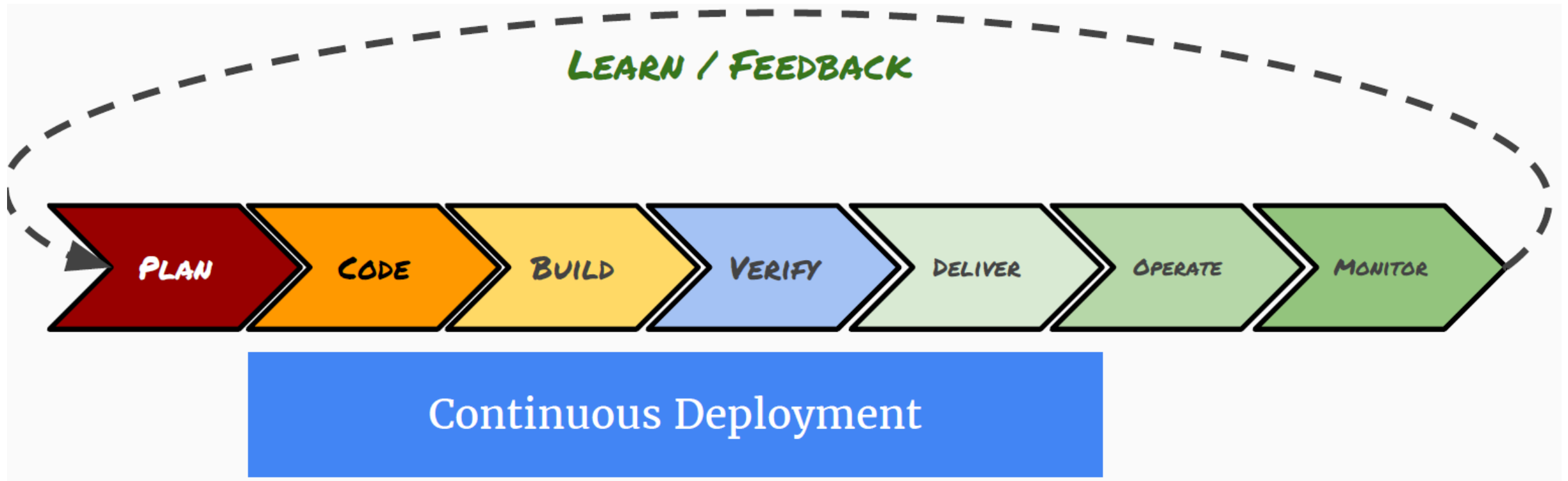
Jira, Taiga, Zendesk etc ...



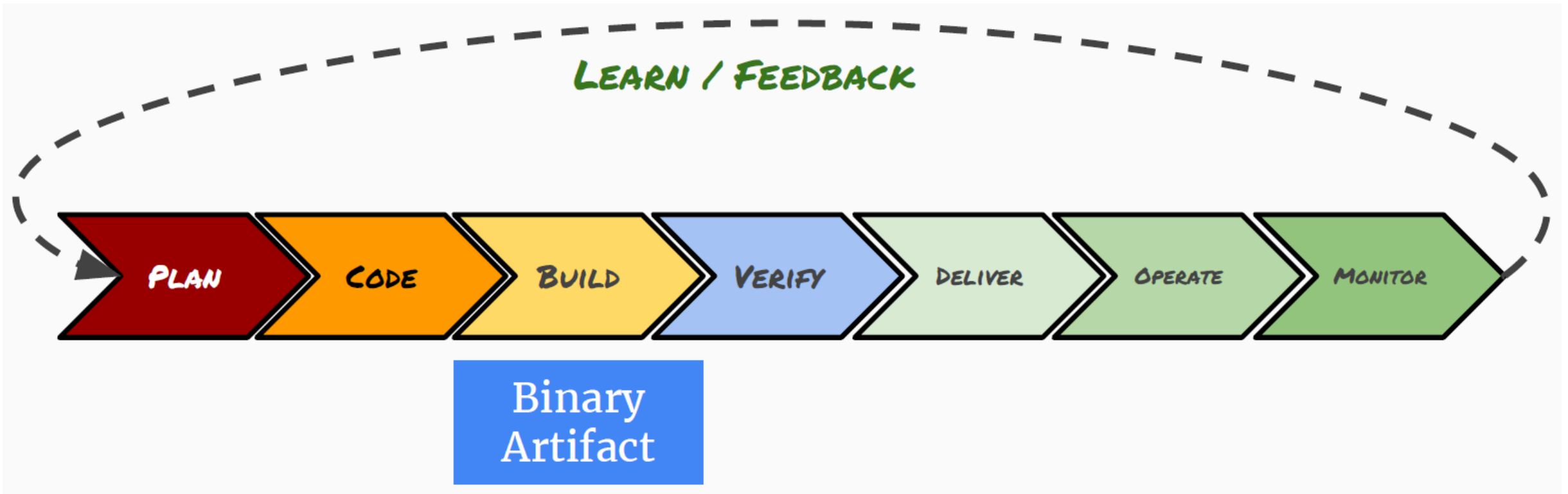
Ansible, Puppet, Chef etc ...



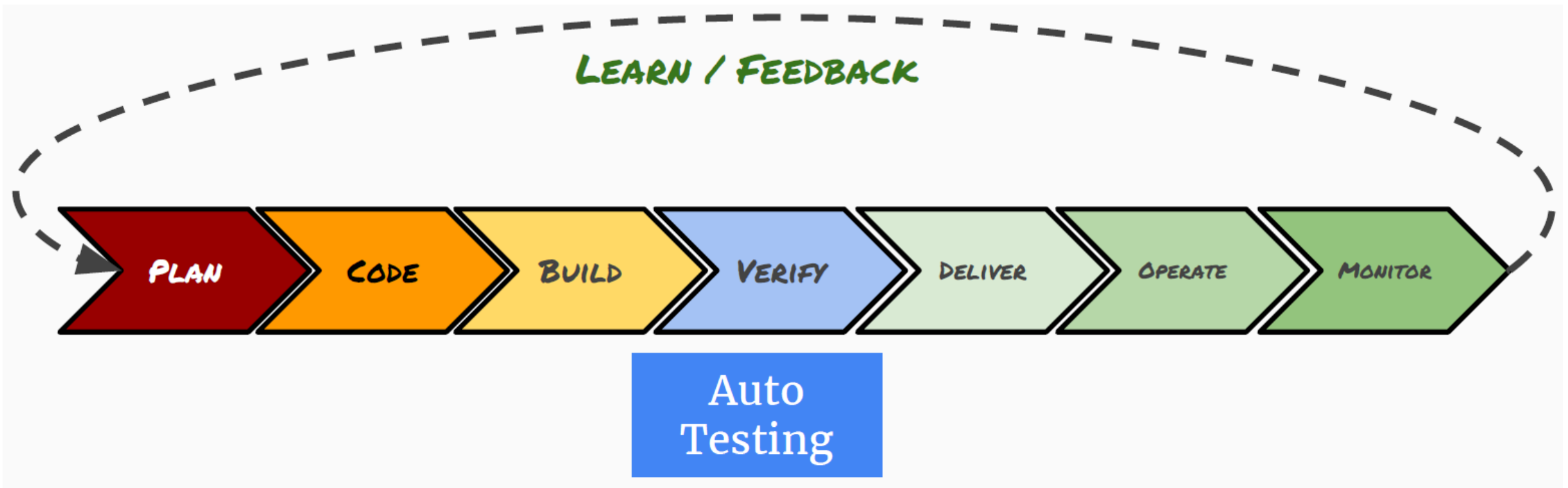
Jenkins, Travis, Gitlab-CI, etc ...



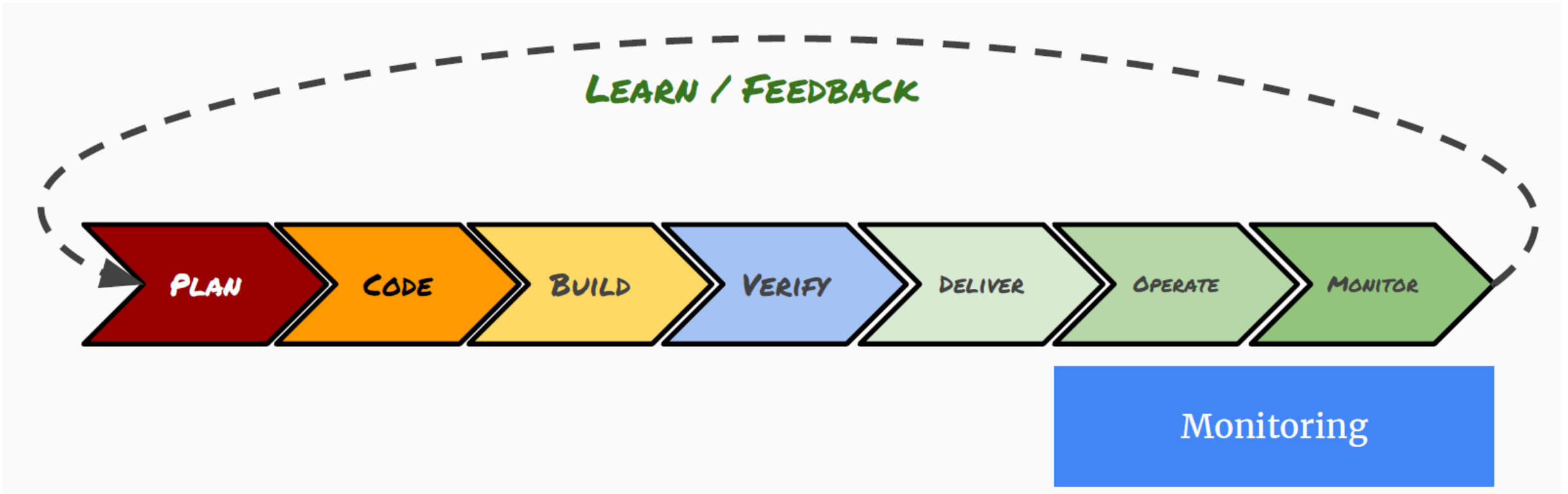
Jenkins, Gitlab-CI, Rundeck, etc ...



Nexus, Artifactory, Registry, etc ...



Jmeter, RobotFramework, etc ...



Grafana, Kibana, Zabbix, etc ...

EMBED DOWNLOAD ADD

Os	Open Source	SCM	Database Mgmt	Build
Fr	Free	CI	Repo Mgmt	Testing
Fm	Freemium	Deployment	Config / Provisioning	Containerization
Pd	Paid	Cloud / Iaas / Pass	Release Mgmt	Collaboration
En	Enterprise	BI / Monitoring	Logging	Security



91 En Xlr XL Release	92 En Ur UrbanCode Release	93 En Bm BMC Release Process	94 En Hp HP Codar	95 En Au Automic	96 En Pl Plutora Release	97 En Sr Serena Release	98 Pd Tfs Team Foundation	99 Fm Tr Trello	100 Pd Jr Jira	101 Fm Rf HipChat	102 Fm Sl Slack	103 Fm Fn Flowdock	104 Pd Pv Pivotal Tracker	105 En Sn ServiceNow
106 Os Ki Kibana	107 Fm Nr New Relic	108 Os Ni Nagios	109 Os Zb Zabbix	110 En Dd Datalog	111 Os El Elasticsearch	112 Os Ss StackState	113 En Sp Splunk	114 Fm Le Logentries	115 Fm Sl Sumo Logic	116 Os Ls Logstash	117 Os Gr Graylog	118 Os Sn Snort	119 Os Tr Tripwire	120 En Ff Fortify



Intro au DevOps et à la CI/CD : Prérequis

Plan

Présentation du formateur

Introduction au DevOps et au CI/CD

Jenkins

Installation et Découverte

Build

Test d'acceptance

Artefact

Déploiement

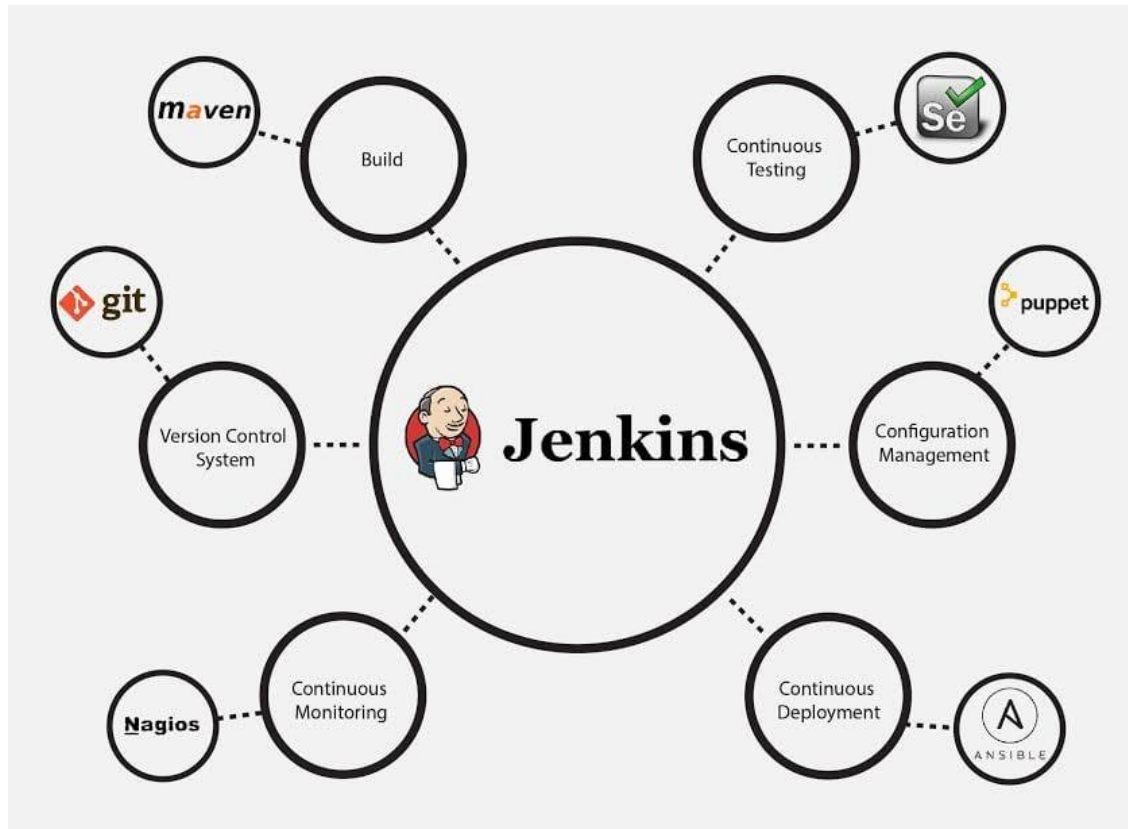
Test de fonctionnement

Shared Library

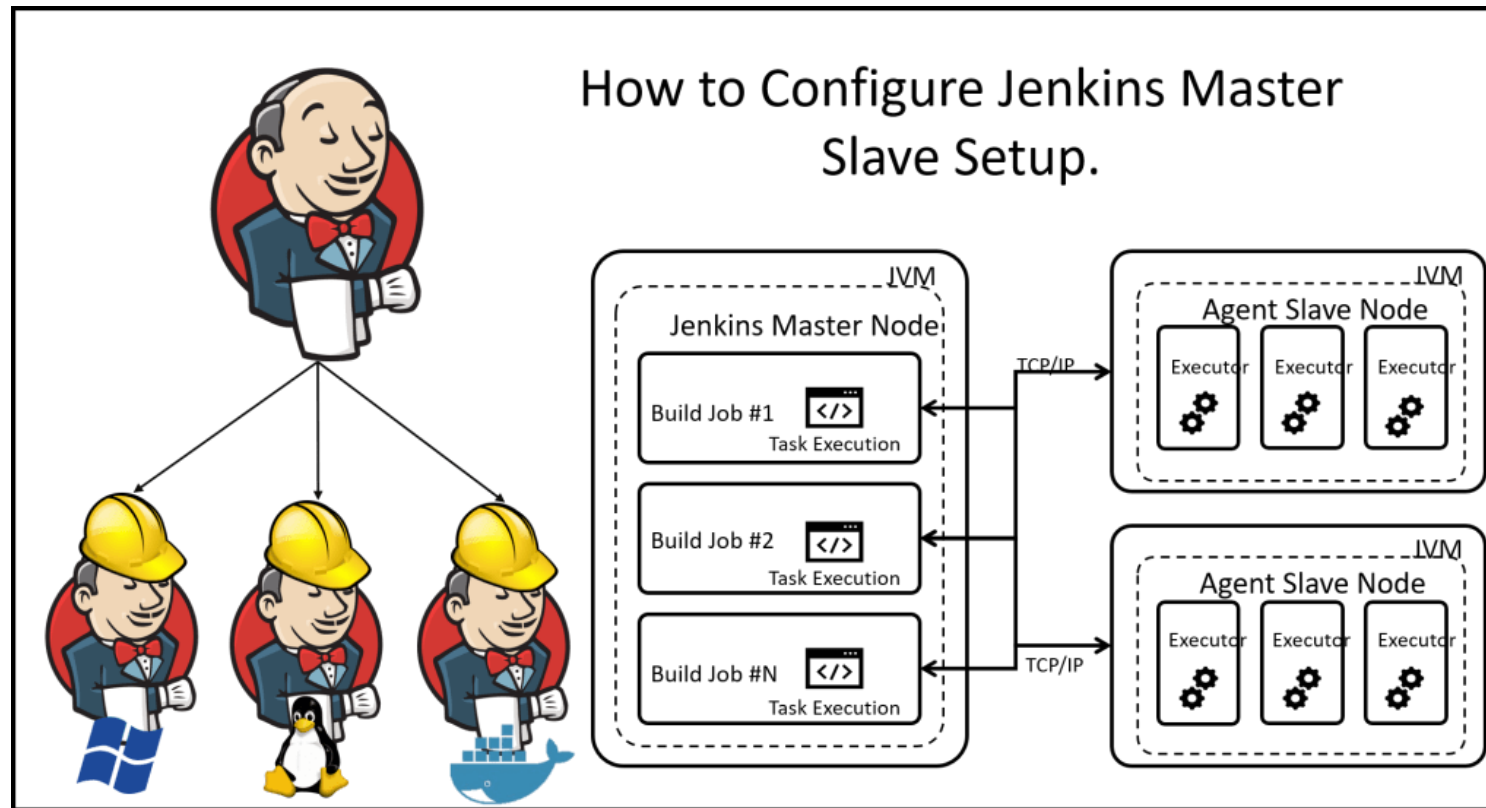
Sécurité

Mini-projet

Jenkins : Présentation



- Serveur d'intégration continue
- Des milliers de plugins
- Fork du projet Hudson
- Workflow Tasks
- OpenSource - JAVA
- Projet communautaire



Jenkins : Architecture



Jenkins : Alternatives

Plan

Présentation du formateur

Introduction au DevOps et au CI/CD

Jenkins

Installation et Découverte

Build

Test d'acceptance

Artefact

Déploiement

Test de fonctionnement

Shared Library

Sécurité

Mini-projet



Installation et Découverte (1/2): Installation

- From Scratch (à la main)
- From Scratch (avec un outil d'automatisation comme ansible)
- À l'aide de [docker](#)
- Autres ...

Installation et Découverte (2/2): Découverte

- Job
- View
- Master
- Slave
- Workspace
- Plugin
- Status

Views

Start job

List of available jobs

List of jobs being executed

All	glance	Dashboard	Glance	Keystone	Milestone-proposed	Nova	OpenStack-CI	Openstack-manuals	Overview	Quantum	Swift	Websites	+
S	W	Name	Last success	Last failure	Last duration								
		glance	15 hr (#95258)	5 days 15 hr (#89227)	2 min 3 sec								
		glance-migrate	15 hr (#118)	1 mo 2 days (#29)	3 sec								
		glance-pep8	15 hr (#286)	16 days (#200)	6.5 sec								
		keystone	12 hr (#403)	13 hr (#309)	1 min 45 sec								
		keystone-migrate	12 hr (#252)	6 days 15 hr (#267)	4.6 sec								
		keystone-pep8	12 hr (#339)	6 days 15 hr (#315)	9 sec								
		keystone-pep8	12 hr (#408)	6 days 15 hr (#383)	22 sec								
		nova	3 hr 44 min (#121728)	19 hr (#121717)	8 min 19 sec								
		nova-migrate	3 hr 44 min (#215)	21 hr (#194)	1 min 0 sec								
		nova-pep8	3 hr 44 min (#1388)	21 hr (#1567)	1 min 9 sec								
		quantum	9 hr 37 min (#52)	9 days 9 hr (#56)	9.1 sec								
		quantum-pep8	9 hr 37 min (#39)	N/A	4.4 sec								
		quantum-pep8	9 hr 35 min (#21)	N/A	20 sec								
		swift	21 hr (#119342)	1 mo 17 days (#119322)	14 sec								
		swift-migrate	21 hr (#115)	N/A	3.3 sec								

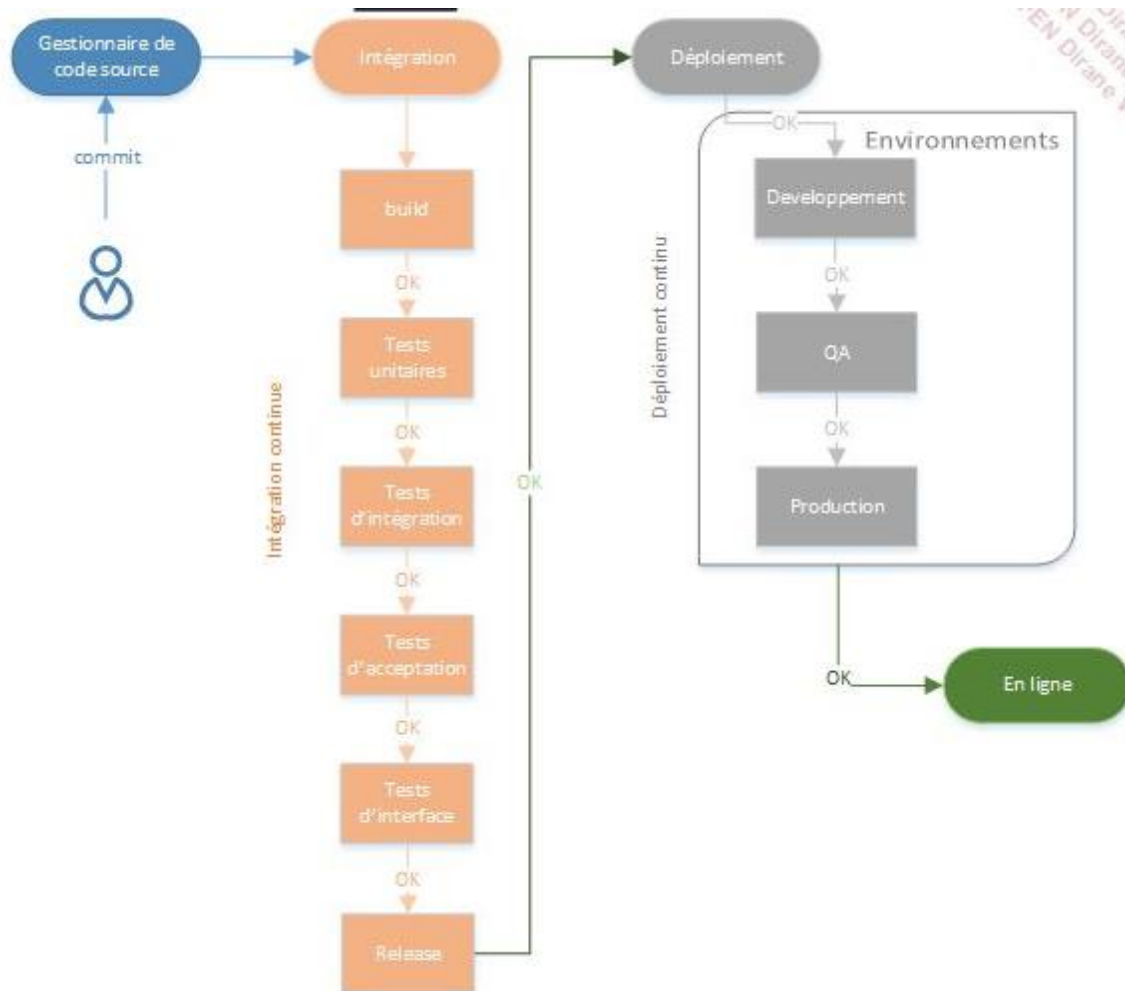
Build Queue

No builds in the queue.

Build Executor Status

#	Master
1	Idle
2	Idle
	build (offline)
	build1
1	Idle
	build2 (offline)
	build3 (offline)
	build4 (offline)
	ci
1	Idle
1	Idle

PROJET FIL ROUGE



- Mise en place d'une chaîne CI/CD sur une application
- On découvrira ainsi Jenkins par la pratique
- Projet : <https://github.com/heroku/alpinehellworld>

Lab-1: Installation de Jenkins



Sur une VM (de préférence dans le Cloud), déployez Jenkins à l'aide de ce fichier docker-compose <https://github.com/eazytrainingfr/jenkins-training/blob/master/docker-compose.yml>



Connectez-vous à l'interface et familiarisez-vous avec l'interface



Plateforme Jenkins By EAZYTraining

Plan

Présentation du formateur

Introduction au DevOps et au CI/CD

Jenkins

Installation et Découverte

Build

Test d'acceptance

Artefact

Déploiement

Test de fonctionnement

Shared Library

Sécurité

Mini-projet

Build (1/8): Types de job



Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.



Multi-configuration project

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.



Folder

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.



GitHub Organization

Scans a GitHub organization (or user account) for all repositories matching some defined markers.



GitHub Organization

Scans a GitHub organization (or user account) for all repositories matching some defined markers.



Multibranch Pipeline

Creates a set of Pipeline projects according to detected branches in one SCM repository.

Description

[Plain text] [Preview](#)

- ☐ Discard old builds ?
- ☐ GitHub project
- ☐ This build requires lockable resources
- ☐ This project is parameterized ?
- ☐ Throttle builds ?
- ☐ Disable this project ?
- ☐ Execute concurrent builds if necessary ?

[Advanced...](#)

Build (2/8): Description du job

Source Code Management

- ☒ None
- ☐ Git
- ☐ Subversion

Build (3/8): Source du code

Build (4/8): Déclencheur du job

Build Triggers

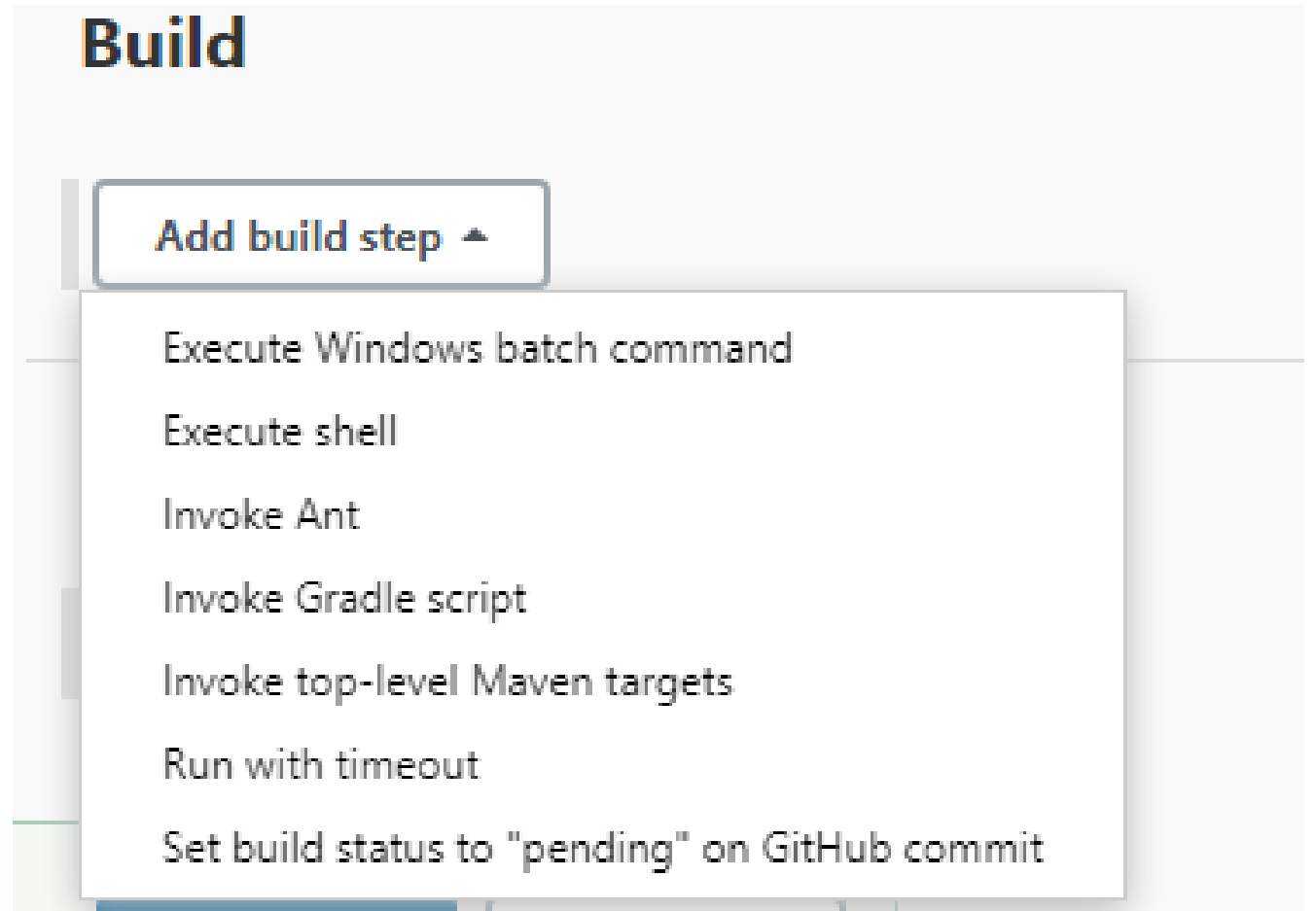
- ☐ Trigger builds remotely (e.g., from scripts)
- ☐ Build after other projects are built
- ☐ Build periodically
- ☐ GitHub hook trigger for GITScm polling
- ☐ Poll SCM

Build Environment

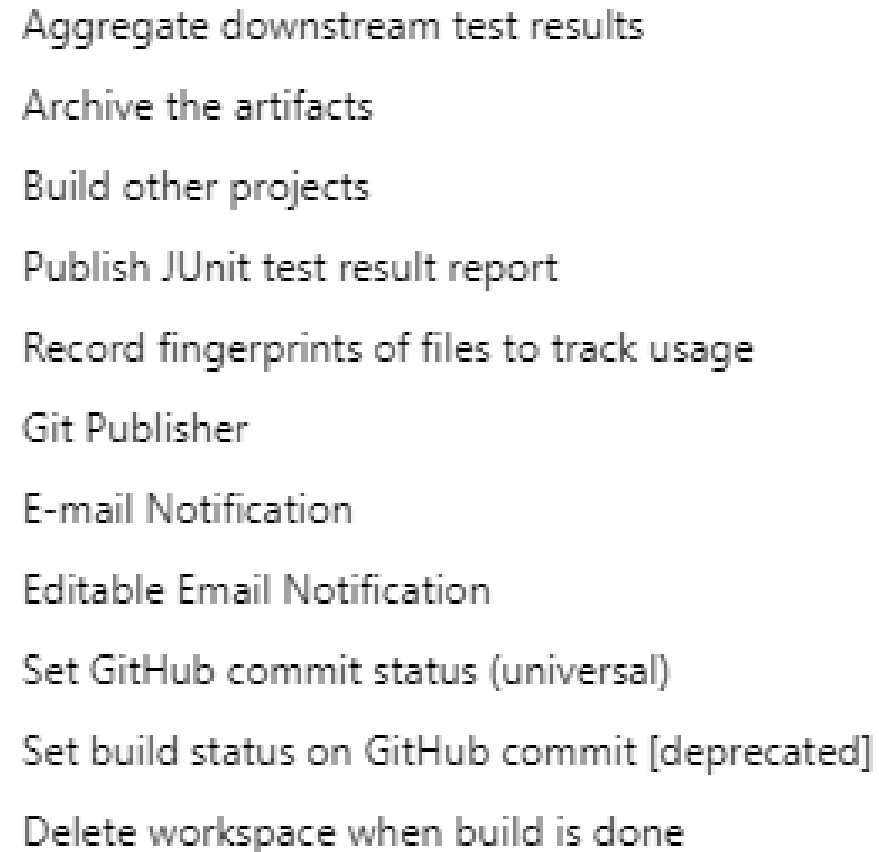
- ☐ Delete workspace before build starts
- ☐ Use secret text(s) or file(s)
- ☐ Abort the build if it's stuck
- ☐ Add timestamps to the Console Output
- ☐ Inspect build log for published Gradle build scans
- ☐ With Ant

Build (5/8): Environment du job

Build (6/8): Build du job



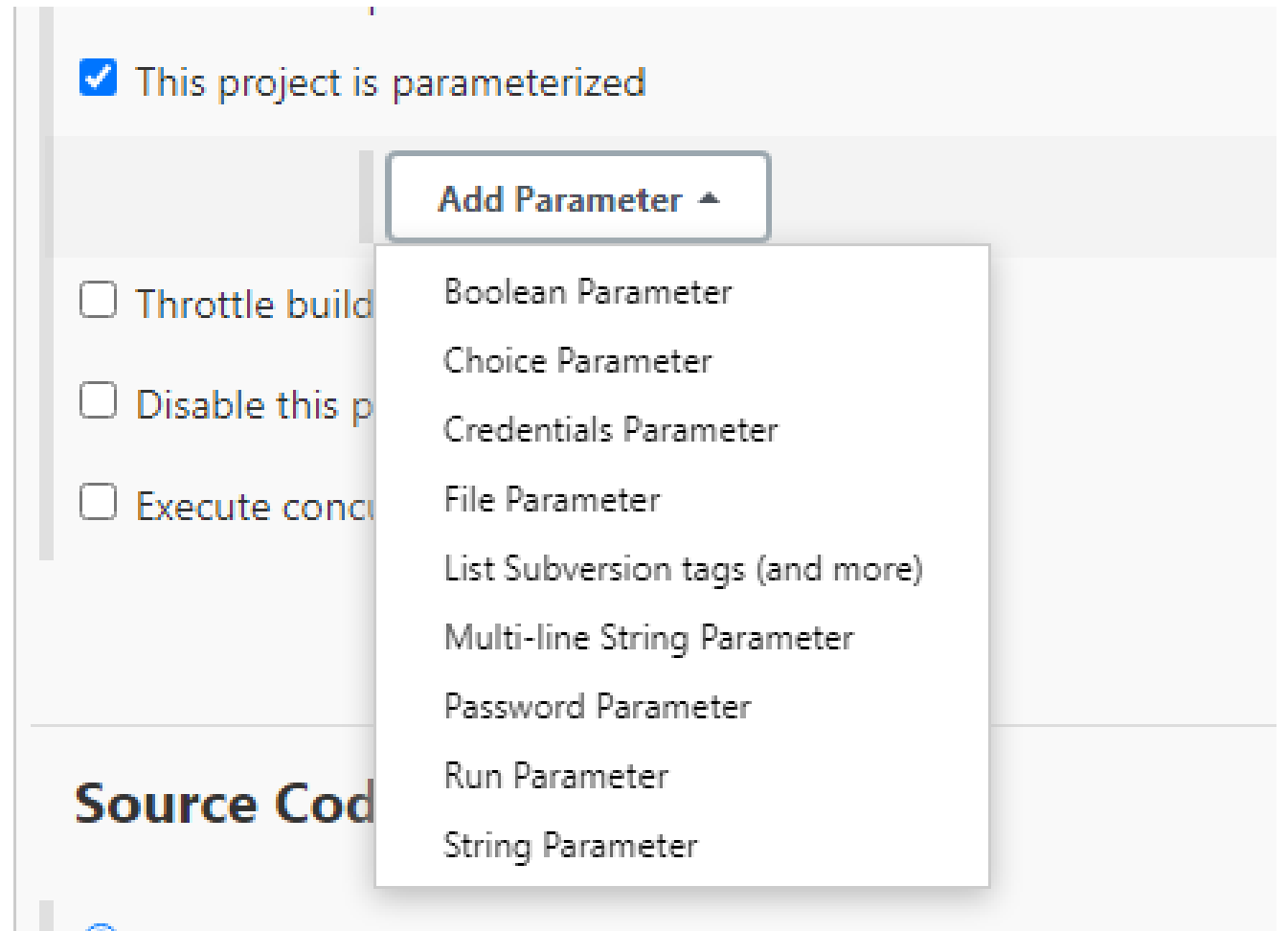
Build (7/8): Action Post-build

A screenshot of the Jenkins 'Post-build Actions' dropdown menu. The menu is white with a light gray border and is positioned over a blurred background of the Jenkins interface. It contains a list of ten post-build actions. At the bottom of the menu is a button labeled 'Add post-build action' with a small upward-pointing triangle icon.

- Aggregate downstream test results
- Archive the artifacts
- Build other projects
- Publish JUnit test result report
- Record fingerprints of files to track usage
- Git Publisher
- E-mail Notification
- Editable Email Notification
- Set GitHub commit status (universal)
- Set build status on GitHub commit [deprecated]
- Delete workspace when build is done

Add post-build action ▲

Build (8/8): Les paramètres



Lab-2: Build

- Créez un job qui s'appellera build
- Ce job permettra de builder l'image du repo suivant : <https://github.com/heroku/alpinehelloworld>
- Avant tout, vous allez forker ce repo pour qu'il soit dans notre environnement github afin de pouvoir le modifier
- Votre job pour l'instant sera du simple bash où vous allez suivre les étapes suivantes:
 - Git clone du repo ci-dessus
 - Build de l'image à partir du dockerfile contenu dans le repo cloné
- Vous allez utiliser la notion de parameter afin de variabiliser le nom de l'image ainsi que son tag
- Vérifiez que vous pouvez bien surcharger le nom de l'image et le tag
- NB: Vous trouverez peut-être cela très banal mais faites-le, on l'améliorera par la suite

Plan

Présentation du formateur

Introduction au DevOps et au CI/CD

Jenkins

Installation et Découverte

Build

Test d'acceptance

Artefact

Déploiement

Test de fonctionnement

Shared Library

Sécurité

Mini-projet



Plugins Index

Discover the 1500+ community contributed Jenkins plugins to support building, deploying and automating any project.

[Browse](#)

Browse categories

- Platforms
- User interface
- Administration
- Source code management
- Build management

New Plugins

- Pipeline: Keep Environment Step
- Perfecto
- Uleska
- valid-network-plugin
- Venafi Machine Identity Protection

Recently updated

- Pipeline: Nodes and Processes
- PostgreSQL Database
- Artifactory
- database
- Google OAuth Credentials

Trending

- Snakeyaml API
- OkHttp
- Extended Read Permission
- Trilead API
- Localization Support

Test d'acceptance (1/1): Plugins

Lab-3: Test d'acceptance

- Vous allez installer le plugin https://plugins.jenkins.io/http_request/
- Utilisez ce plugin pour mettre en place le job suivant (Nom du job: test acceptance)
 - Utilisez la section source du job pour spécifier le lien github du projet (<https://github.com/<votre user>/alpinehelloworld.git>)
 - Build de l'image (sans push)
 - démarrage d'un conteneur utilisant l'image buildée à l'aide du plugin installé précédemment
 - Utilisez la plugin pour verifier que le conteneur retourne bien le code 200 en http
- NB: Bien évidemment il vous faudra variabiliser le nom et le tag de l'image comme au lab-2

Plan

Présentation du formateur

Introduction au DevOps et au CI/CD

Jenkins

Installation et Découverte

Build

Test d'acceptance

Artefact

Déploiement


Test de fonctionnement

Shared Library


Sécurité


Mini-projet


Artefact (1/2): Les identifiants


 **Jenkins**


Jenkins > Credentials


 New Item


 People


 Build History

 Manage Jenkins


 My Views

 Lockable Resources

 New View


Build Queue 

No builds in the queue.

Build Executor Status 

1 Idle



2 Idle

 **Credentials**

T	P	Store ↓	Domain
---	---	---------	--------

Icon: [S](#) [M](#) [L](#)

Stores scoped to Jenkins

P	Store ↓	Domains
	Jenkins	 (global)

Artefact (2/2): Les types de secret

▸ Global credentials (unrestricted) ▸

Kind

- Username with password
- Username with password**
- GitHub App
- SSH Username with private key
- Secret file
- Secret text
- Certificate

ID

Description

Lab-4: Artefact

- Refaites exactement le même job qu'au lab-3 mais sous le nom artefact
- Installez le plugin docker-build-step et paramétrez le pour utiliser la socket unix de docker (**unix:///var/run/docker.sock**) dans le system config de Jenkins
- Vous allez pousser votre image buildée toujours à l'aide du plugin docker-build-step sur votre dockerhub
- Pour passer vos login et mot de passe dockerhub, utilisez la notion de secret
- Si vous n'avez pas de compte dockerhub, merci d'en créer un via ce lien : <https://hub.docker.com/>
- NB: N'oubliez pas que pour pousser une image sur notre dockerhub il faut le préfixer par notre nom d'utilisateur dockerhub

Plan

Présentation du formateur

Introduction au DevOps et au CI/CD

Jenkins

Installation et Découverte

Build

Test d'acceptance

Artefact

Déploiement

Test de fonctionnement

Shared Library

Sécurité

Mini-projet

Déploiement (1/6): Le Jenkinsfile

```
Jenkinsfile x
1 pipeline {
2   agent {
3     'the agent'
4   }
5
6   environment {
7     MY_VAR = 'this value'
8     OTHER_VAR = "${INITIAL_VAR ?: 'default value'}"
9   }
10
11   stages {
12     stage('does the thing') {
13
14
```

- Pipeline as code
- Groovy ou DSL
- Linter disponible
- Embarquer le pipeline avec le code

any

Execute the Pipeline, or stage, on any available agent. For example: `agent any`

none

When applied at the top-level of the `pipeline` block no global agent will be allocated for the entire Pipeline run and each `stage` section will need to contain its own `agent` section. For example: `agent none`

label

Execute the Pipeline, or stage, on an agent available in the Jenkins environment with the provided label. For example: `agent { label 'my-defined-label' }`

Label conditions can also be used. For example: `agent { label 'my-label1 && my-label2' }` or `agent { label 'my-label1 || my-label2' }`

node

`agent { node { label 'labelName' } }` behaves the same as `agent { label 'labelName' }`, but `node` allows for additional options (such as `customWorkspace`).

docker

Execute the Pipeline, or stage, with the given container which will be dynamically provisioned on a `node` pre-configured to accept Docker-based Pipelines, or on a node matching the optionally defined `label` parameter. `docker` also optionally accepts an `args` parameter which may contain arguments to pass directly to a `docker run` invocation, and an `alwaysPull` option, which will force a `docker pull` even if the image name is already present. For example: `agent { docker 'maven:3-alpine' }` or

Déploiement (2/6): Les agents

```
agent {  
  docker {  
    image 'maven:3-alpine'  
    label 'my-defined-label'  
    args '-v /tmp:/tmp'  
  }  
}
```

```
agent {  
  docker {  
    image 'myregistry.com/node'  
    label 'my-defined-label'  
    registryUrl 'https://myregistry.com/'  
    registryCredentialsId 'myPredefinedCredentialsInJenkins'  
  }  
}
```

Déploiement (3/6): l'agent docker

The following variables are available to shell scripts

BRANCH_NAME

For a multibranch project, this will be set to the name of the branch being built, for example in case you wish to deploy to production from master but not from feature branches (refer to `CHANGE_ID` and `CHANGE_TARGET`).

CHANGE_ID

For a multibranch project corresponding to some kind of change request, this will be set to the change ID, such as a pull request number, if supported; else unset.

CHANGE_URL

For a multibranch project corresponding to some kind of change request, this will be set to the change URL, if supported; else unset.

CHANGE_TITLE

For a multibranch project corresponding to some kind of change request, this will be set to the title of the change, if supported; else unset.

CHANGE_AUTHOR

For a multibranch project corresponding to some kind of change request, this will be set to the username of the author of the proposed change, if supported; else unset.

CHANGE_AUTHOR_DISPLAY_NAME

For a multibranch project corresponding to some kind of change request, this will be set to the human name of the author, if supported; else unset.

CHANGE_AUTHOR_EMAIL

For a multibranch project corresponding to some kind of change request, this will be set to the email address of the author, if supported; else unset.

CHANGE_TARGET

For a multibranch project corresponding to some kind of change request, this will be set to the target or base branch to which the change could be merged, if supported; else unset.

BUILD_NUMBER

The current build number, such as "153"

BUILD_ID

The current build ID, identical to `BUILD_NUMBER` for builds created in 1.597+, but a `YYYY-MM-DD_hh-mm-ss` timestamp for older builds.

BUILD_DISPLAY_NAME

The display name of the current build, which is something like "#153" by default.

JOB_NAME

Name of the project of this build, such as "foo" or "foo/bar".

JOB_BASE_NAME

Short Name of the project of this build stripping off folder paths, such as "foo" for "bar/foo".

BUILD_TAG

String of "jenkins-`{JOB_NAME}`-`{BUILD_NUMBER}`". All forward slashes ("/") in the `JOB_NAME` are replaced with dashes ("-"). Convenient to put into a resource file.

EXECUTOR_NUMBER

The unique number that identifies the current executor (among executors of the same machine) that's carrying out this build. This is the number you see in the console output.

NODE_NAME

Name of the agent if the build is on an agent, or "master" if run on master.

NODE_LABELS

Whitespace-separated list of labels that the node is assigned.

WORKSPACE

The absolute path of the directory assigned to the build as a workspace.

JENKINS_HOME

The absolute path of the directory assigned on the master node for Jenkins to store data.

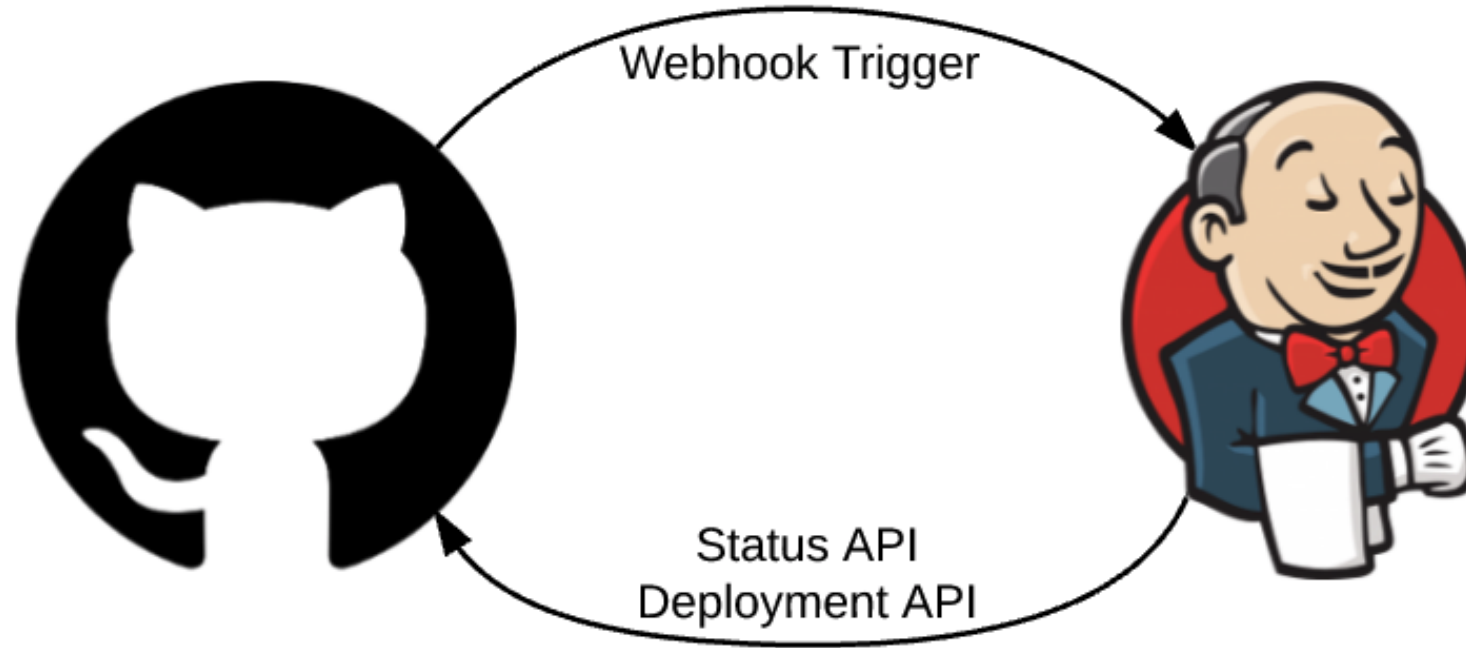
JENKINS_URL

Full URL of Jenkins, like `http://myserver:8080/jenkins/` (note: only available if `Jenkins URL` set in system configuration).

Déploiement (4/6): Les Variables d'environnement

```
node {  
    stage('Example') {  
        if (env.BRANCH_NAME == 'master') {  
            echo 'I only execute on the master branch'  
        } else {  
            echo 'I execute elsewhere'  
        }  
    }  
}
```

Déploiement (5/6): Les structures conditionnelle



Déploiement (6/6): Intégration avec Github

Lab-5: Création d'un compte héroku



CRÉEZ UN COMPTE SUR
[HTTPS://WWW.HEROKU.COM/](https://www.heroku.com/)



RECUPÉREZ LA CLÉ API



CRÉEZ UN SECRET SUR
JENKINS NOMMÉ
HEROKU_API_KEY



VOUS ÊTES PRÊT POUR LA
SUITE

Lab-6: Déploiement

- Vous allez créer un job de type Pipeline s'appelant **déploiement**
- Vérifiez que le plugin « **github intégration** » est bien installé et configurer le webhook sur votre projet github afin d'automatiser le build exécuté par jenkins
- Votre pipeline sera déclenché par toute modification de votre repo sur github (configurer la partie build trigger en conséquence)
- Vous allez également créer un fichier Jenkinsfile dans votre repo qui aura les étapes suivantes
 - Build de l'image (agent none)
 - Création d'un conteneur à partir de cette image (agent none)
 - Test de l'image à l'aide de curl (agent none)
 - Utilisation de la cli heroku déjà installée pour builder , pousser et déployer votre image sur heroku suivant cette documentation uniquement si nous sommes sur la branche master : <https://devcenter.heroku.com/articles/container-registry-and-runtime> (agent none)
- NB:
 - Le déploiement sur héroku devra être fait dans deux projets, un portant le nom <user>-production et l'autre <user>-staging
 - Variabiliser au maximum les nom des projets afin de vous faciliter la tâche par la suite

Plan

Présentation du formateur

Introduction au DevOps et au CI/CD

Jenkins

Installation et Découverte

Build

Test d'acceptance

Artefact

Déploiement

Test de fonctionnement

Shared Library

Sécurité

Mini-projet



jenkins APP 4:12 PM

styleguide master #814 [f0ed411d] failed

shape-react-core master #836 [f0ed411d] @2.6.1 failed

new messages



jenkins APP 5:20 PM

shape-react-core master #838 [7e29a9fe] @2.7.0 returned to passing

hello master #795 [7e29a9fe] failed

styleguide master #816 [7e29a9fe] returned to passing

Test de fonctionnement (1/3): Notification

Preview	Code
<div>build passing</div>	<p>Protected</p> <pre>http://127.0.0.1:8080/job/node-bittorrent-sync/statusbadges-build/icon</pre> <p>Unprotected</p> <pre>http://127.0.0.1:8080/statusbadges-build/icon?job=node-bittorrent-sync</pre>
<div>grade 2.0</div>	<p>Protected</p> <pre>http://127.0.0.1:8080/job/node-bittorrent-sync/statusbadges-grade/icon</pre> <p>Unprotected</p> <pre>http://127.0.0.1:8080/statusbadges-grade/icon?job=node-bittorrent-sync</pre>
<div>coverage 92%</div>	<p>Protected</p> <pre>http://127.0.0.1:8080/job/node-bittorrent-sync/statusbadges-coverage/icon</pre> <p>Unprotected</p> <pre>http://127.0.0.1:8080/statusbadges-coverage/icon?job=node-bittorrent-sync</pre>

Test de fonctionnement (2/3): Badge

blazemeter

log out

ENABLE AUTO REFRESH

Back to Dashboard

Status

Changes

Workspace

Build Now

Delete Project

Configure

Performance Trend

Build History

find

#9

22.12.2017 18:03

#8

22.12.2017 18:00

#7

22.12.2017 17:53

#6

22.12.2017 17:51

#5

22.12.2017 17:18

#4

22.12.2017 17:18

#3

22.12.2017 17:12

#2

22.12.2017 17:11

Project blazemeter

Workspace

Last Successful Artifacts

dashBoard_stats.xml

668 B view

Recent Changes

Permalinks

- Last build (#9), 1 min 39 sec ago
- Last stable build (#7), 11 min ago
- Last successful build (#8), 4 min 34 sec ago
- Last failed build (#9), 1 min 39 sec ago
- Last unstable build (#8), 4 min 34 sec ago
- Last unsuccessful build (#9), 1 min 39 sec ago
- Last completed build (#9), 1 min 39 sec ago

Performance Trend

add description

Disable Project

Response time

Build	90% line (ms)	average (ms)	median (ms)
#6	75	65	70
#7	75	60	70
#8	78	65	70
#9	75	60	70

Percentage of errors

Build	errors (%)
#6	0
#7	0
#8	50
#9	50

Test de fonctionnement (3/3): Jmeter

Ulrich MONJI | Jenkins | Eazytraining

60

Lab-7: Test de fonctionnement

- Installez et configurez le plugin slack afin d'intégrer les notifications (si vous n'avez pas de slack, créez en un ou utilisez celui de eazytraining)
- Il vous faudra créer un channel dédié à ses notifications
- Installez le plugin de badge : <https://plugins.jenkins.io/embeddable-build-status/> et rajouter la balise dans le fichier README.MD de votre repo après avoir créé votre job pipeline
- Modifiez votre Jenkinsfile afin d'y intégrer la tâche d'envoi de la notification de réussite ou d'échec du pipeline via slack

Plan

Présentation du formateur

Introduction au DevOps et au CI/CD

Jenkins

Installation et Découverte

Build

Test d'acceptance

Artefact

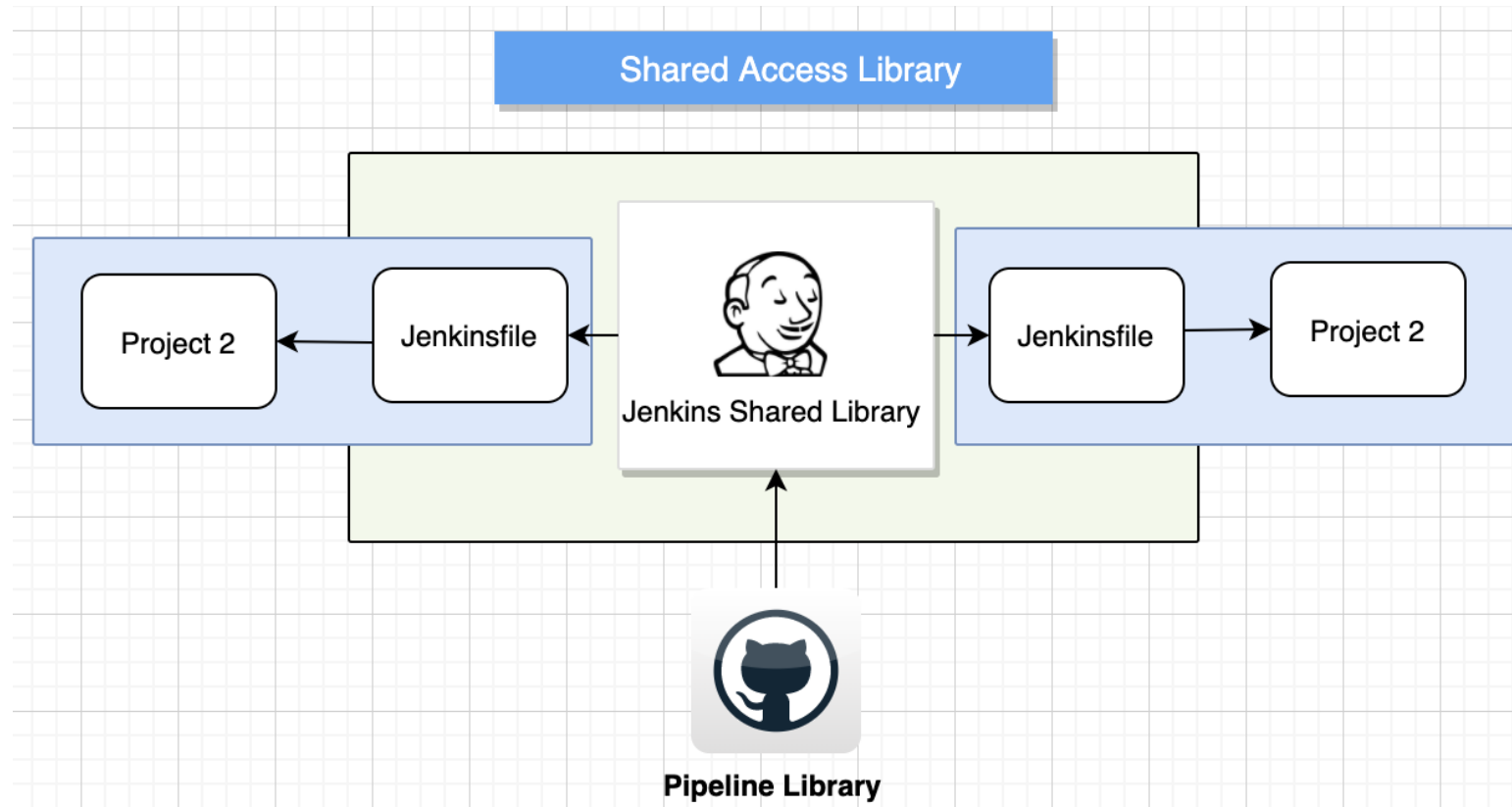
Déploiement

Test de fonctionnement

Shared Library

Sécurité

Mini-projet



Shared Library (1/1): Présentation



Lab-8: Shared Library

Créez une shared library pour factoriser l'étape de notification slack

Plan

Présentation du formateur

Introduction au DevOps et au CI/CD

Jenkins

Installation et Découverte

Build

Test d'acceptance

Artefact


Déploiement

Test de fonctionnement

Shared Library

Sécurité

Mini-projet

 [Back to Dashboard](#)

 [Manage Jenkins](#)

 [Create User](#)

Create User

Username:

Password:

Confirm password:

Full name:

E-mail address:

Create User

Sécurité (1/4): Utilisateur



Manage and Assign Roles

Global roles

Role	Overall	Credentials							Agent							Job				
	Administer	Read	Create	Delete	ManageDomains	Update	View	Build	Configure	Connect	Create	Delete	Disconnect	Provision	Build	Cancel	Configure	Create	Delete	Discover
<input checked="" type="checkbox"/> admin	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/> developer	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

3

Role to add

1
developer

2
Add

Project roles

Role	Pattern	Credentials							Job							Run			SCM
		Create	Delete	ManageDomains	Update	View	Build	Cancel	Configure	Create	Delete	Discover	Move	Read	Workspace	Delete	Replay	Update	Tag

Role to add

Pattern

Add

4

Save

Apply



Sécurité (2/4): Rôles

Sécurité (3/4): Assignment



Assign Roles

Global roles

User/group	admin	developer
 guru99	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
 Anonymous	<input type="checkbox"/>	<input type="checkbox"/>

1



Manage and Assign Roles

Project roles

Role	Pattern	Credentials					Job					Run			SCM			
		Create	Delete	ManageDomains	Update	View	Build	Cancel	Configure	Create	Delete	Discover	Move	Read	Workspace	Delete	Replay	Update
<input checked="" type="checkbox"/> tester	tester.*	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

4

Role to add

Pattern

1
tester

2
tester.*

3
Add

5
Save Apply

Sécurité (4/4): Project Role

Lab-9: Sécurité

- Vérifiez que le plugin **Role-based Authorization Strategy** est installé
- Créez un utilisateur eazytraining
- Créez un rôle nommé viewer avec uniquement le droit de consulter le projet « test acceptance »
- Assignez ensuite ce rôle à l'utilisateur eazytraining
- Enfin, connectez vous avec l'utilisateur eazytraining pour vérifier qu'il n'a que les droits qui lui ont été conféré et qu'ils ne voit que le projet test acceptance (vous pouvez créer un autre job avec un nom différent pour tester cela)

Plan

Présentation du formateur

Introduction au DevOps et au CI/CD

Jenkins

Installation et Découverte

Build

Test d'acceptance

Artefact

Déploiement

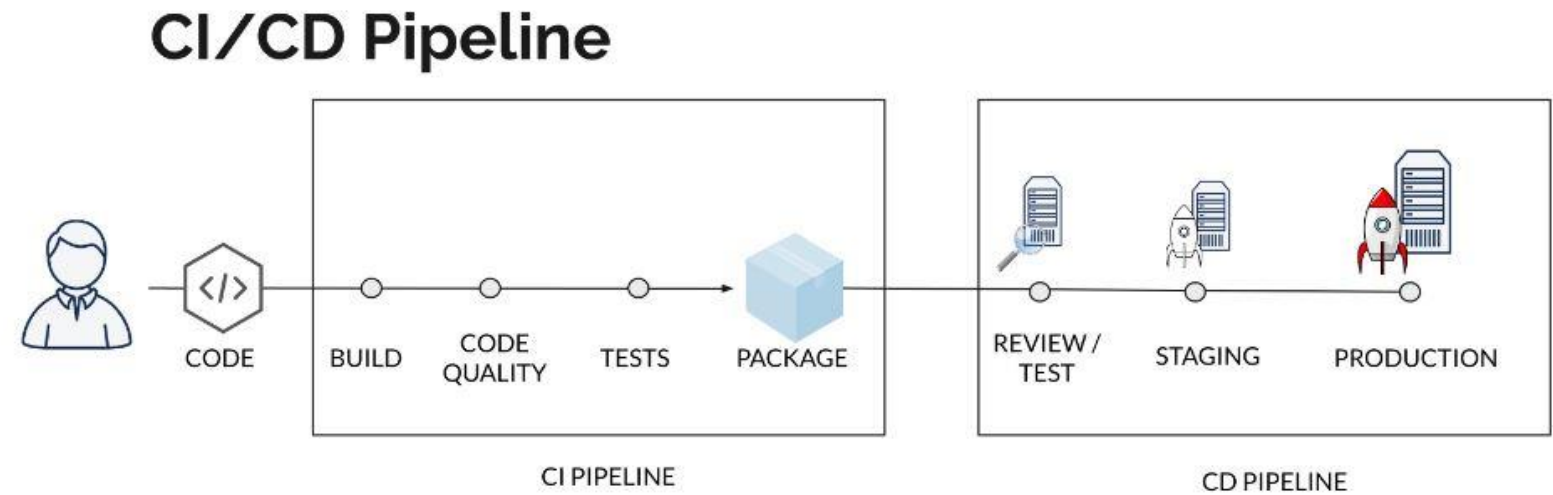
Test de fonctionnement

Shared Library

Sécurité

Mini-projet

Mini-projet: Static Website



- <https://github.com/diranetafen/static-website-example.git>



Merci pour votre attention