CONFÉRENCE DEP OVERVIEW

DIGITAL ENABLEMENT PLATEFORM (PLATEFORME DE MISE EN ŒUVRE DU NUMÉRIQUE)

The world is how we shape it *

MARDI 12 SEPTEMBRE 2022

Urssaf
Caisse nationale

sopra steria

* Le monde est tel que nous le façonnons.

SOMMAIRE

01 Introduction aux principes de la DEP

02 Présentation des outils

03 Pour aller plus loin

04 Conclusion

L'INDUSTRIALISATION

L'industrialisation informatique désigne les étapes successives qui mènent à une gestion optimisée des ressources dans un contexte spécifique. Les améliorations de la performance s'effectuent suivant 4 axes distincts que sont:

- La performance financière
- Le gain en efficience pour les processus internes
- La performance des métiers au sein de l'entreprise
- La capacité à mieux maitriser les risques et à assurer une amélioration continue

01

Introduction aux principes de la DEP

DEVELOPMENT ENVIRONMENT OVERVIEW

WHAT DOES DEP PROVIDE FOR YOUR DAY TO DAY DEVELOPMENTS?

Development Hybrid Cloud + DevOps Tools + Workstation as Code + Container Orchestration

A development infrastructure based on hybrid cloud

Azure

Azure Stack

The infrastructure required to build and test applications

GitLab + Templates capitalized from our experience as software vendor & system integrator

Added value tools (Engineering tools, Visual management solutions…)

Self-service installation (OpenShift CI/CD) OPENSHIFT

The CI/CD and tools required to build and deploy with the max level of automation

Workstation installation automation

GitLab

Homogenization of project team workstations

Ready to use orchestration solution

OPENSHIFT

Ability to run containerized applications

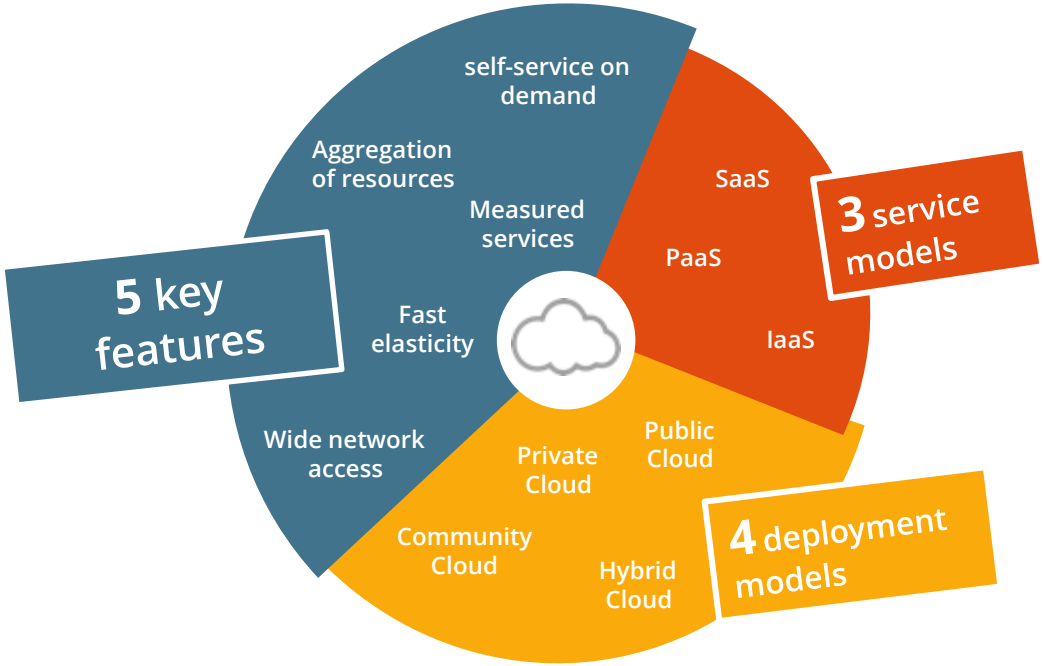
WHAT IS THE CLOUD ?

DEFINITION OF THE NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY

Cloud computing is a way to deliver IT services

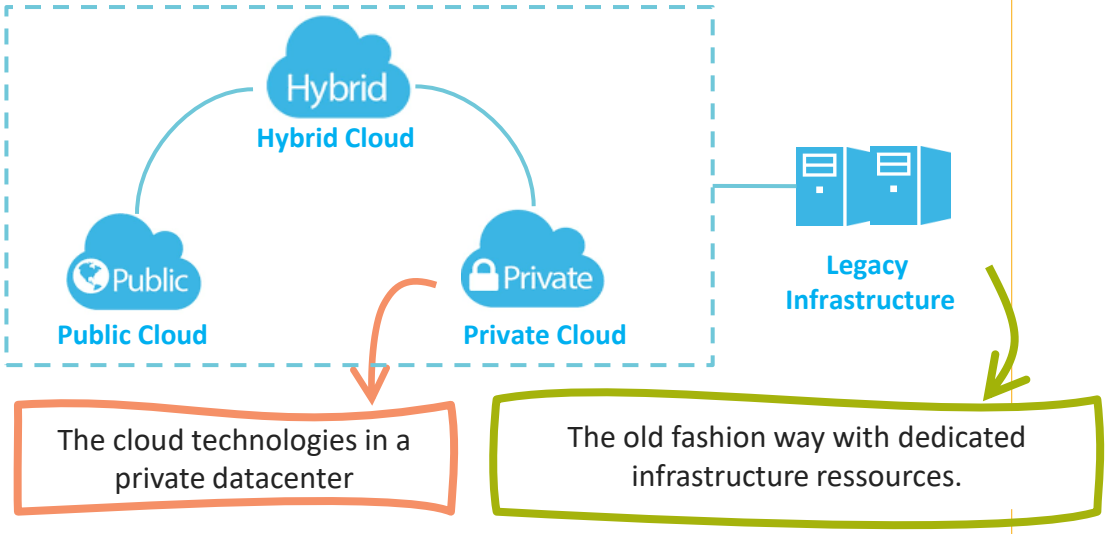
Resources are shared and accessible as services that can be directly used by customers :

payment methods, CRM, operating systemsbases "as-a-service" databases



Hybrid cloud: a combination between Public cloud and Private Cloud

- The Public Cloud offers:
  - The full power and elasticity of the Cloud
  - An access to a wide range of services
- The Private Cloud offers:
  - The control of the location, and thus data sovereignty
- Hybridation allows to mix both, with a possible access to legacy infrastructure, for example a database



HOSTING PROJECT RESOURCES

NETWORK ACCESS, FLOW AND FIREWALL POSITIONNING

Development environment: hosting project resources

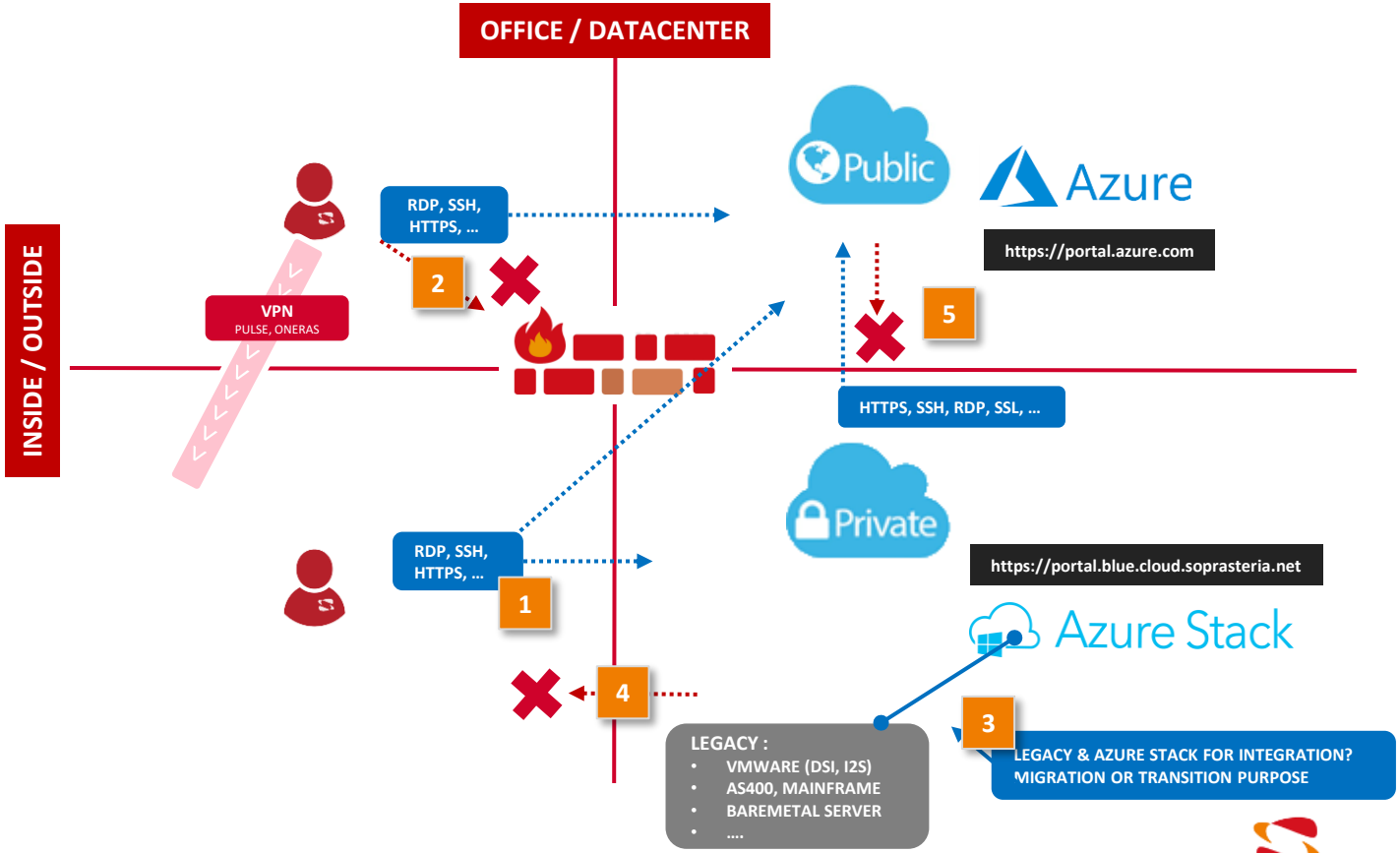
1/ Admin access to Azure/Azure Stack workload

2/ Can't access Azure Stack from Internet, need to make VPN with Office Network

3/ Azure Stack ⇔ Legacy (Servers) possible

4/ Azure Stack => Workstation, Non Server workload, … not authorized

5/ Hybrid scenario where Azure workload contact Azure Stack in 2020 roadmap



Service models

On-Premises	IAAS	PAAS	SAAS
Applications	Applications	Applications	Applications
Data	Data	Data	Data
Runtime	Runtime	Runtime	Runtime
Middleware	Middleware	Middleware	Middleware
Virtualization	Virtualization	Virtualization	Virtualization
O/S	O/S	O/S	O/S
Servers	Servers	Servers	Servers
Storage	Storage	Storage	Storage
Internal network	Internal network	Internal network	Internal network
External network	External network	External network	External network
External Partners	External Partners	External Partners	External Partners

Managed by IT teams

« I do it myself »

Investments

I do a lot by myself, in my own way, with my own infrastructure

I can concentrate on my core business, but I must beware clouder vendor lock

Managed by Clouder

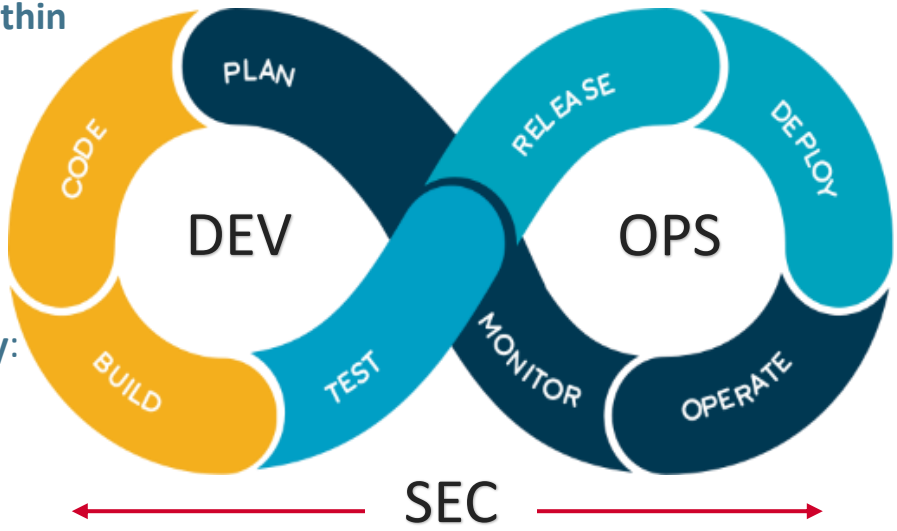
Adherence to Clouder choices

Costs per use

DEVOPS
KEY CONCEPTS

Continuous Integration/Continous Delivery/Continuous Deployment (CI/CD²)

- Software developed in **successive increments within short cycles** and **deployed as soon as ready**
- **Agile and DevOps cultures and practices**, strengthening **collaboration** between business, developers and operations
- **Automation mandatory** to ensure quality
- Enriched with a **permanent attention to Security**: DevOps ➔ **DevSecOps**

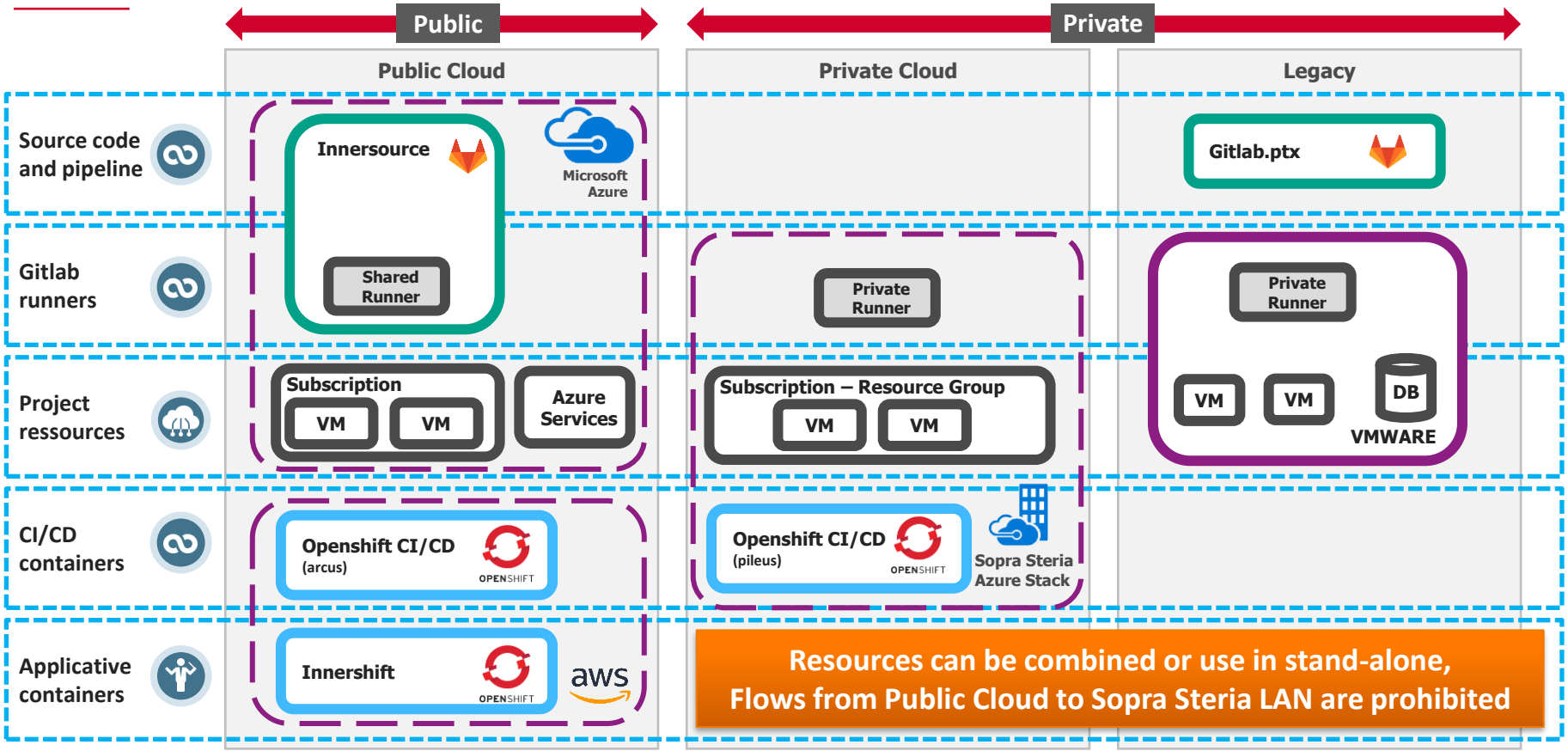


02

Présentation des outils

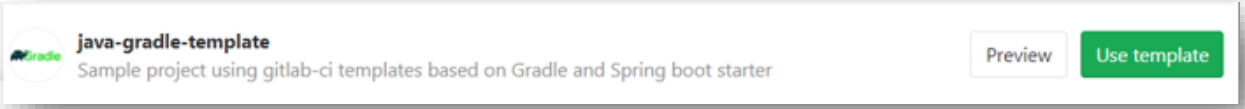
DEVELOPMENT ENVIRONMENT OVERVIEW

WHERE ARE DEP RESOURCES FOR DEV ENVIRONMENT LOCATED?



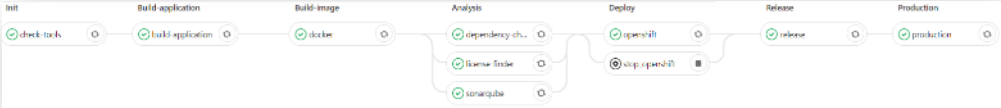
DEVOPS TOOLS

CREATE A NEW PROJECT FROM TEMPLATE



Template content

- ❑ A Gitlab CI/CD pipeline



- ❑ A simple Springboot app, built with Gradle, package with Docker

- ❑ A deployment template to deploy the application in Innershift
- ❑ Pre-configured files for Sonarqube analysis, license finder, Dep. Check, Intools as code

DEVOPS TOOLS

PIPELINE & TOOLS: BUILD & PACKAGING

Overview of DevOps Tools

Build & Packaging | Analysis | Acceptance | Deployment

Automatically build the application and run the unit tests

Compilation	Automated Unit Testing	Code Review	Packaging
maven	JUnit	GitLab	GitLab
Gradle	nunit		docker
nuget	PHPUnit		Nexus
npm			
php			

DevOps mindset:
All repetitive activities must be automated.

It's mandatory when you want to be able to deliver often and rapidly

DEVOPS TOOLS

PIPELINE & TOOLS: ANALYSIS

Overview of DevOps Tools

Build & Packaging | **Analysis** | Acceptance | Deployment

Ensure the code quality and security

Static Code Analysis	Static Application Security Testing	Static Containers Security Testing	Technical Debt Management
sonarqube	sonarqube	clair	themis
	DEPENDENCY-CHECK		
	LicenceFinder		
	CHECKMARX *		

Remember our target:

Being highly professional and combine TTM, Security and Quality

Automating the analysis is fine, but checking the results is essential!

** External component*

DEVOPS TOOLS

PIPELINE & TOOLS: ACCEPTANCE

Overview of DevOps Tools

Build & Packaging | Analysis | **Acceptance** | Deployment

Perform automated tests

Non Regression Testing	Performance Testing	Dynamic Application Security Testing	Traceability
cucumber, NoraUi, Selenium, SoapUI, Soapower	APACHE JMeter™, Gatling	OWASP ZAP	D² COCKPIT BY CDK

Automate all the type of tests but do it with a strategy!

Automating is key, automating without focus can be a loss of time

Go progressively:

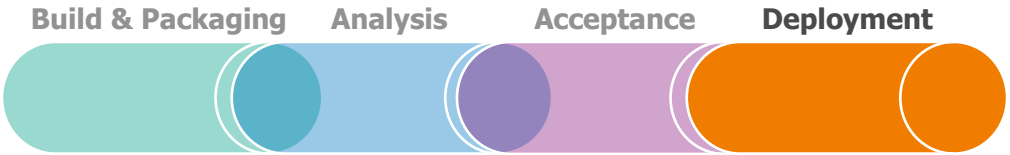
Automate first the non regression tests you realize each time, and the most critical. Focus on the tests that are not modified everytime

It requires functional and technical skills

DEVOPS TOOLS

PIPELINE & TOOLS: DEPLOYMENT

Overview of DevOps Tools



Automate the build of the infrastructure, set up configurations and deploy applications

Infrastructure Management	Configuration Management	Automated Deployment
Terraform	ANSIBLE	docker

These steps will be required several times:

- **Deployment in our testing environment to be able to run Acceptance activities**
- **Deployment in integration environment, performance tests environment, pre-production…**
- **Finally deployment in production environment, if you go full DevOps**

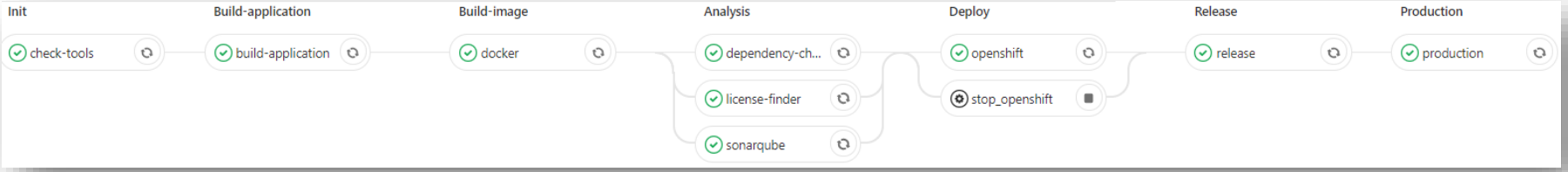
Automate the deployment to be sure you can deploy with serenity whenever you need it, including hot fixes

Automation is key! Infrastructure as code is part of the answer

GITLAB CONCEPTS

GITLAB-CI – PIPELINE TEMPLATE

Using the templates



Share common stages and jobs

Simple way to override or enhance existing jobs

gitlab-ci.yml available as file template too

Open for contributions (ci library is a gitlab-project)

```
include:
  - "/configuration/common.yml"

  - "/stages/init.yml"
  - "/stages/build.yml"
  - "/stages/analysis.yml"
  - "/stages/deploy.yml"
  - "/stages/acceptance.yml"
  - "/stages/release.yml"
  - "/stages/production.yml"
```

GITLAB CONCEPTS

GITLAB-CI - EXAMPLES

Gitlab concepts

```yaml
# https://hub.docker.com/r/ciricihq/gitlab-sonar-scanner/
sonarqube:
  stage: analysis
  image: ciricihq/gitlab-sonar-scanner
  variables:
    SONAR_ANALYSIS_MODE: publish
  script:
    - gitlab-sonar-scanner -Dsonar.login=$SONAR_TOKEN
    # - gitlab-sonar-scanner -Dsonar.login=$SONAR_TOKEN -Dsonar.analysis.mode=preview -Dsonar.gitlab.project_id=$CI_PROJECT_PATH -Dsonar.git
  artifacts:
    name: "${CI_JOB_ID}_${CI_JOB_NAME}"
    expire_in: 1 week
    paths:
      - codeclimate.json
  only:
    variables:
      - $SONAR_URL
      - $SONAR_TOKEN
    refs:
      - master

license-finder:
  stage: analysis
  image:
    name: licensefinder/license_finder
  before_script:
    - source /etc/profile.d/rvm.sh
  script:
    - license_finder report --format html --decisions_file license_finder_decisions.yml --save license_finder_report.html
  artifacts:
    paths:
      - "license_finder_report.html"
```

```yaml
snapshot:
  except:
    - master
    - tags
  stage: Build
  image: docker
  services:
    - name: docker:dind
      command: ["--insecure-registry=gitlab.ptx.fr.s
  variables:
    DOCKER_HOST: tcp://docker:2375
  before_script:
    - docker login -u gitlab-ci-token -p "$CI_JOB_TO
  script:
    - version=$(docker run --rm -t -v "$(pwd):/proje
    - tagname=$CI_COMMIT_REF_SLUG
    - imagename="$GITLAB_REGISTRY/$CI_PROJECT_PATH:$
    - docker build -t "$imagename" .
    - docker push "$imagename"
```

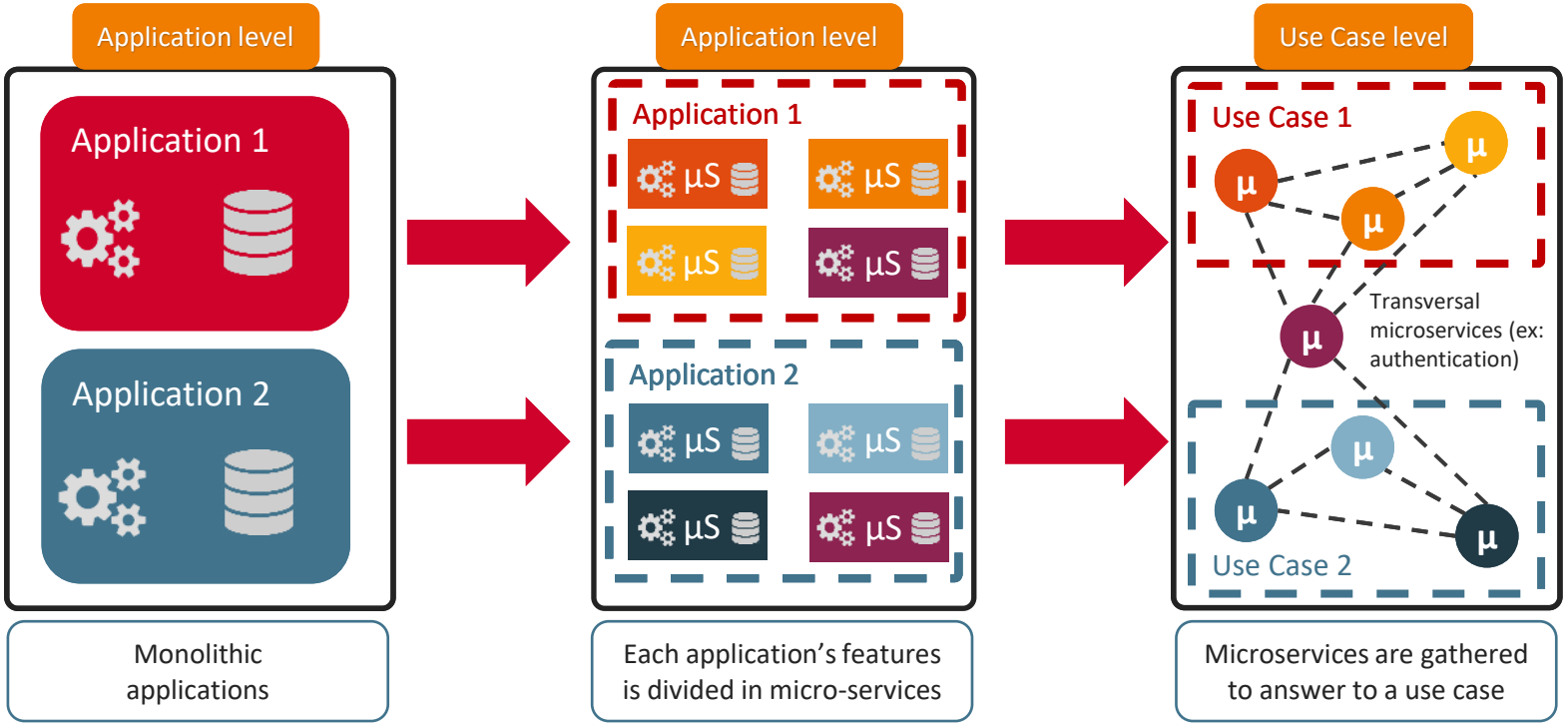
03

Pour aller plus loin

CLOUD NATIVE APPROACH

MONOLITHIC VS MICROSERVICES ?

Cloud Native Approach



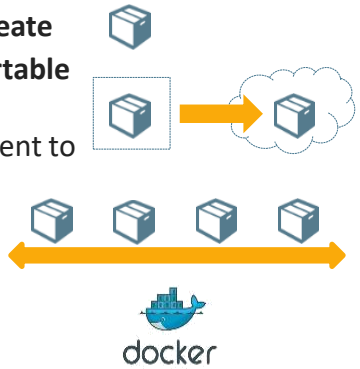
CLOUD NATIVE APPROACH

THE LEADING CLOUD NATIVE ARCHITECTURE

Cloud Native Approach

Packaged in containers

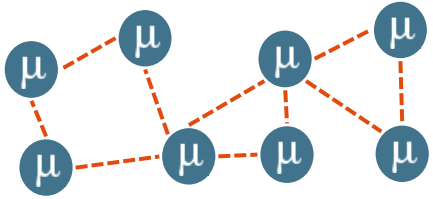
- Containers provide **isolation context**
- Containers are :
  - **Fast to create**
  - **Easily portable** from one environment to the other
  - **Scalable**



Microservices architecture

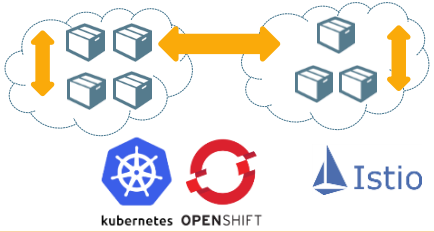
- Each application is a **collection of small services** that can be :
  - **Built**
  - **Deployed**
  - **Operated**

Independently from each other



Dynamically Managed in Cloud

- Containers are orchestrated to use the power of Cloud computing:
  - **Scalability**
  - **Resilience**
  - **Resources optimization**
- Service Mesh allow to handle microservices at scale



These principles are not the only one allowing to build a Cloud Native architecture, but they are leading the way

04

Conclusion

WHAT ABOUT YOUR PROJECT CONTEXT ?

PROJECT ASSESSMENT

What about your project context ?

What makes sense and provides benefits in my context?

How I am going to use DEP?

Assess added value

Automation

Quality

Efficiency

Security

DevOps

Time To Market

Progressive App Modernization

App Rearchitecture

Information System agility

Go To Cloud

End 2 End

Keep Data control

WHAT ABOUT YOUR PROJECT CONTEXT ?

What about your project context ?

PROJECT ASSESSMENT

Improvement areas	Development Hybrid Cloud	Production Hybrid Cloud	DevOps Tools	Code Marketplace	Containers orchestrator	Micro-services stack
Automation	●		●		●	
Quality			●			
Productivity	●		●	●		
Re-use				●		●
Security	●	●	●		●	
DevOps			●			
Time To Market	●		●	●	●	●
Progressive App Modernization					●	●
App Refactoring					●	●
Information System agility					●	●
Go To Cloud	●	●	●		●	●
End 2 End	●	●	●			
Keep Data control	●	●				

Des questions ?

N’hésitez pas à me contacter

MERCI.