

# Infrastructure as Code (IaC)

quand & comment le  
mettre en place?

Bertrand Thomas, Architecte Solution



**Promyze Webinars**

Connect Developers' Knowledge

# Introduction

## Infrastructure as Code



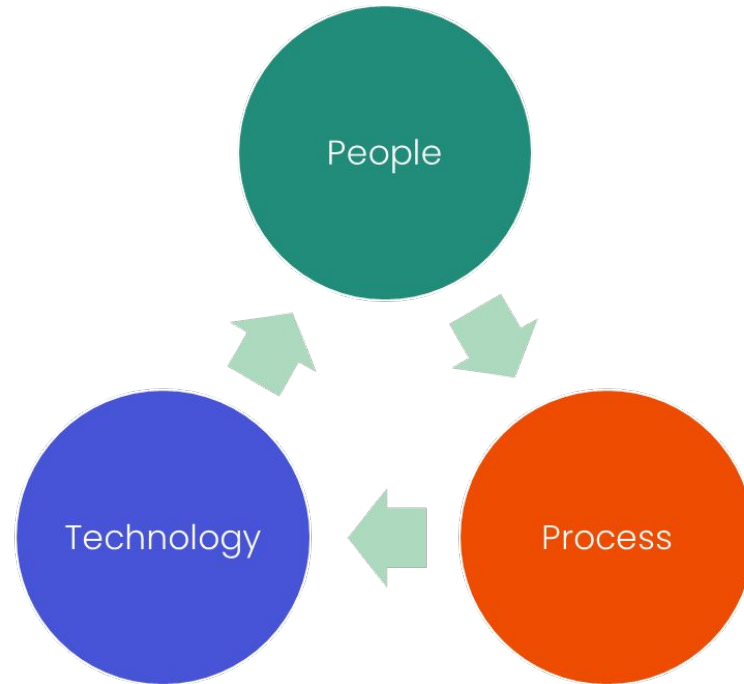
Source: [urolime.com/blogs](https://urolime.com/blogs)



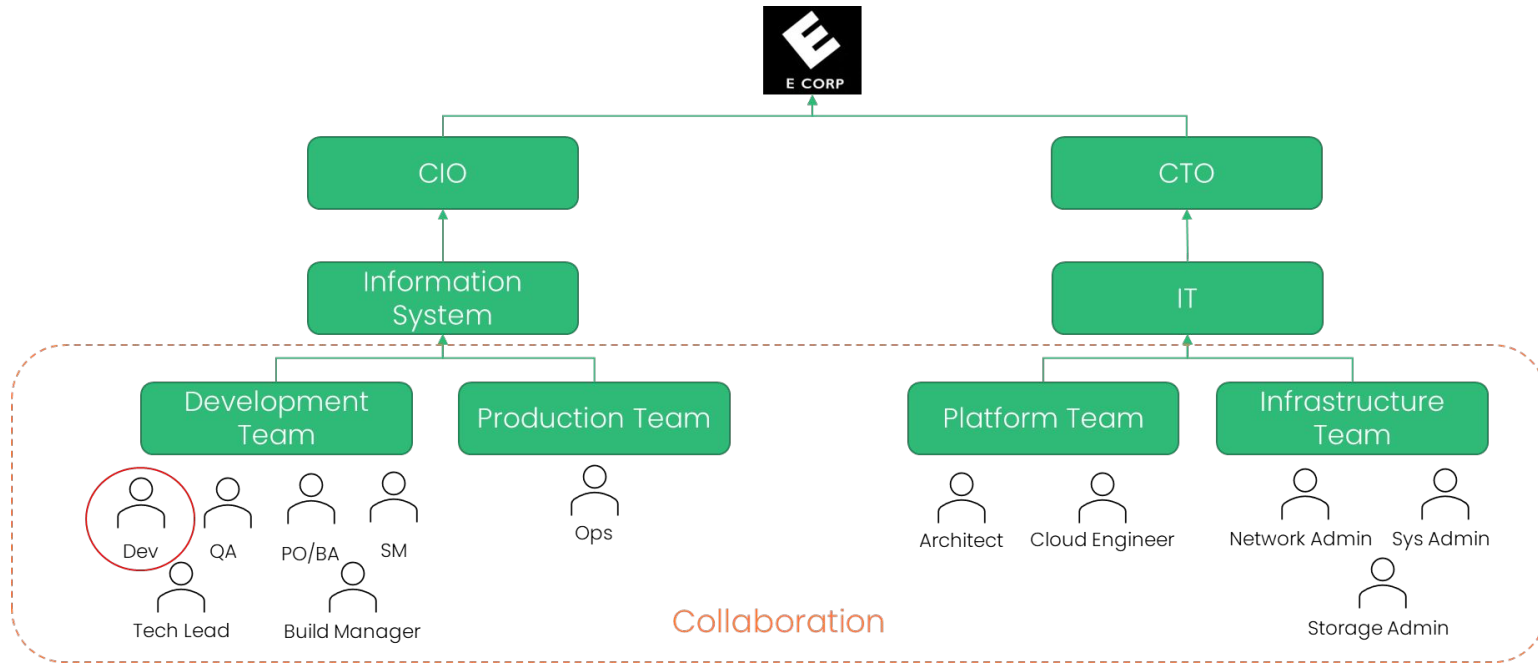
# Le DevOps



# Les 3 piliers d'une transformation



# Exemple d'organisation





QA = Quality Assurance  
PO/BA = Product Owner/Business Analyst  
SM = Scrum Master



# Gestion d'un data center

On-site	IaaS	PaaS	SaaS
Applications	Applications	Applications	Applications
Data	Data	Data	Data
Runtime	Runtime	Runtime	Runtime
Middleware	Middleware	Middleware	Middleware
O/S	O/S	O/S	O/S
Virtualization	Virtualization	Virtualization	Virtualization
Servers	Servers	Servers	Servers
Storage	Storage	Storage	Storage
Networking	Networking	Networking	Networking

 You manage

 Service provider manages

# Challenges

19 av.

■ New Visitor ■ Returning Visitor





# Compétences transverses

- Concevoir une **solution simple** basée sur l'**idempotence**
- Réaliser un **code maintenable**
- Mutualiser le code et gérer le **cycle de vie applicatif**
- Exécuter dans un **environnement neutre**
- Connaître le **fonctionnement** des ressources infra (et leurs API)
- Gérer différentes versions d'une même ressource
- Éviter les interruptions de service
- Sauvegarder les sorties des actions





# Manipulation des données

- Configuration management database (CMDB)
- Source de vérité
- Inventaire dynamique
- Valeurs secrètes
- Rapports d'exécution
- Observabilité des ressources



# Anti-patterns

- Fonctionnement en Silo
- Mode “Héro”
- “Fast & Furious”
- Liberté totale
- Un seul outil pour tout faire

# Solutions techniques

19 av.

■ New Visitor ■ Returning Visitor





# Outils IaC

Outil	Société	Open source	Method	Artifacts
<a href="#">Ansible</a>	Red Hat	★ <a href="#">57.2k</a>	Push	Collections (playbooks, roles)
<a href="#">Chef</a>	Progress	★ <a href="#">7.2k</a>	Push	Cookbooks
<a href="#">Pulumi</a>	Pulumi	★ <a href="#">15.8k</a>	Push	Libraries
<a href="#">Puppet</a>	Perforce	★ <a href="#">6.9k</a>	Pull (& Push)	Modules
<a href="#">Terraform</a>	HashiCorp	★ <a href="#">37.1k</a>	Push	Modules



# Ansible

```

- name: Install required components for Debian systems
  when: ansible_os_family == "Debian"
  block:
    - name: Install aptitude
      ansible.builtin.apt:
        name: aptitude
        state: present
        update_cache: true
    - name: Install Debian packages required for installation
      ansible.builtin.apt:
        name: "{{ packages }}"
        state: present
        update_cache: true
  vars:
    packages:
      - apt-transport-https
      - ca-certificates
      - curl
      - gnupg-agent
      - software-properties-common
      - gpg-agent
```

```

# checks the hosts can be reached ok
ansible all -m ping -i inventories/lab01

# run a playbook
ansible-playbook -i inventories/lab01 playbooks/demo.yml
```



# Terraform

```
terraform {
  required_providers {
    azurerm = {
      source = "hashicorp/azurerm"
      version = "~> 3.0.2"
    }
  }
  required_version = ">= 1.1.0"
}

provider "azurerm" {
  features {}
}

resource "azurerm_resource_group" "rg" {
  name     = "myTFResourceGroup"
  location = "westus2"
}
```

```
# initializes the configuration
terraform init

# creates an execution plan
terraform plan

# applies the configuration
terraform apply

# inspects the current state
terraform show

# destroys all resources
terraform destroy
```



# Leçons apprises



# Pattern d'utilisation des outils

- Configuration des **équipements réseau** (Switch) -> **Ansible**
- Création et gestion des **ressources** (Cloud, Kubernetes) -> **Terraform**
- Mise à jour des **systèmes** (VM, IoT) -> **Ansible**
- Contrôle continu de la configuration des systèmes -> **Puppet**
- Déploiement **applicatif** + dépendances -> **Cloud Native w/ GitOps**





# Comment démarrer

- Identifier un **périmètre** fonctionnel / cas d'utilisation
- Monter une équipe / squad **pluridisciplinaire**
- Réaliser un **ADR** (Architecture Decision Record)
- Développer **en itération** et viser la solution **minimaliste**
- Faire évoluer suivant les **retours** (démonstration et tests utilisateurs)
- Embarquer le plus de personnes possible
- Suivre une **DOD** (Definition of Done) respectant l'État de l'Art

# Merci !

Chaîne Youtube : [@devprofr](#)