Introduction au PaaS Cloud Foundry

AUSY

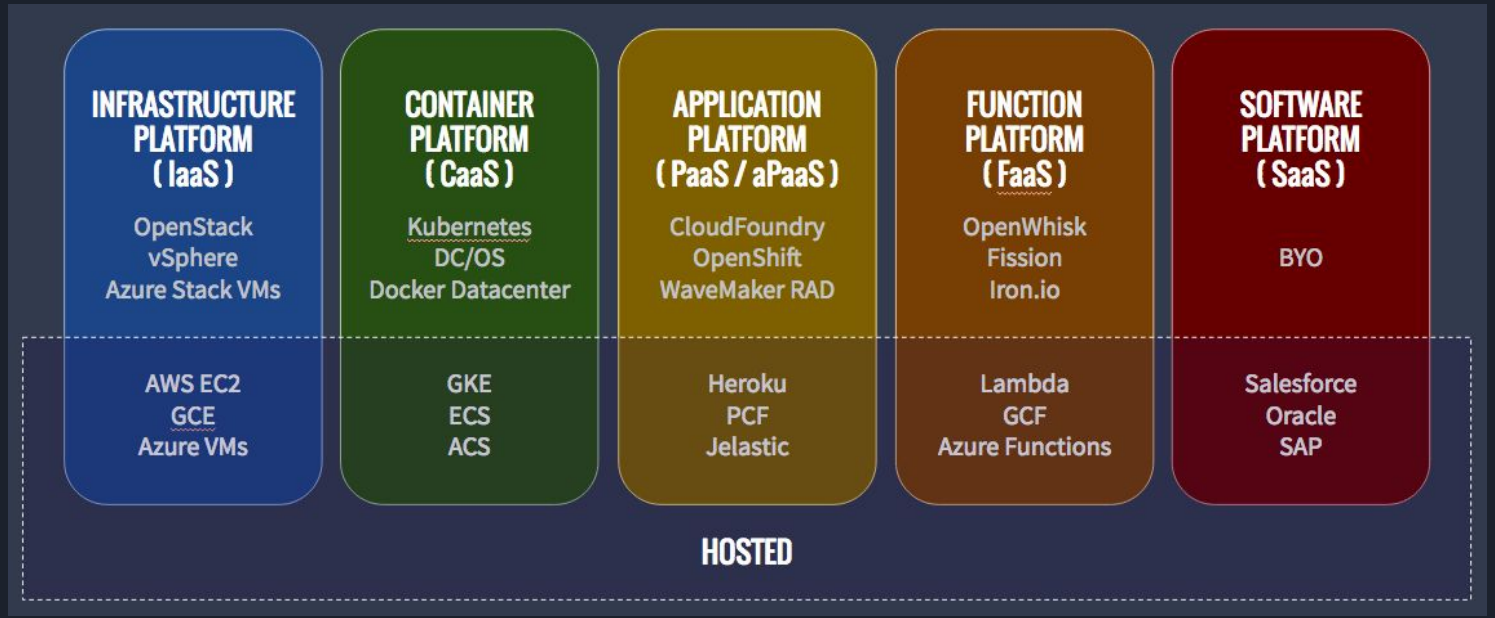
Shake Your Brain / 2018-02-08

Intro

Acronymes : ... as a Service

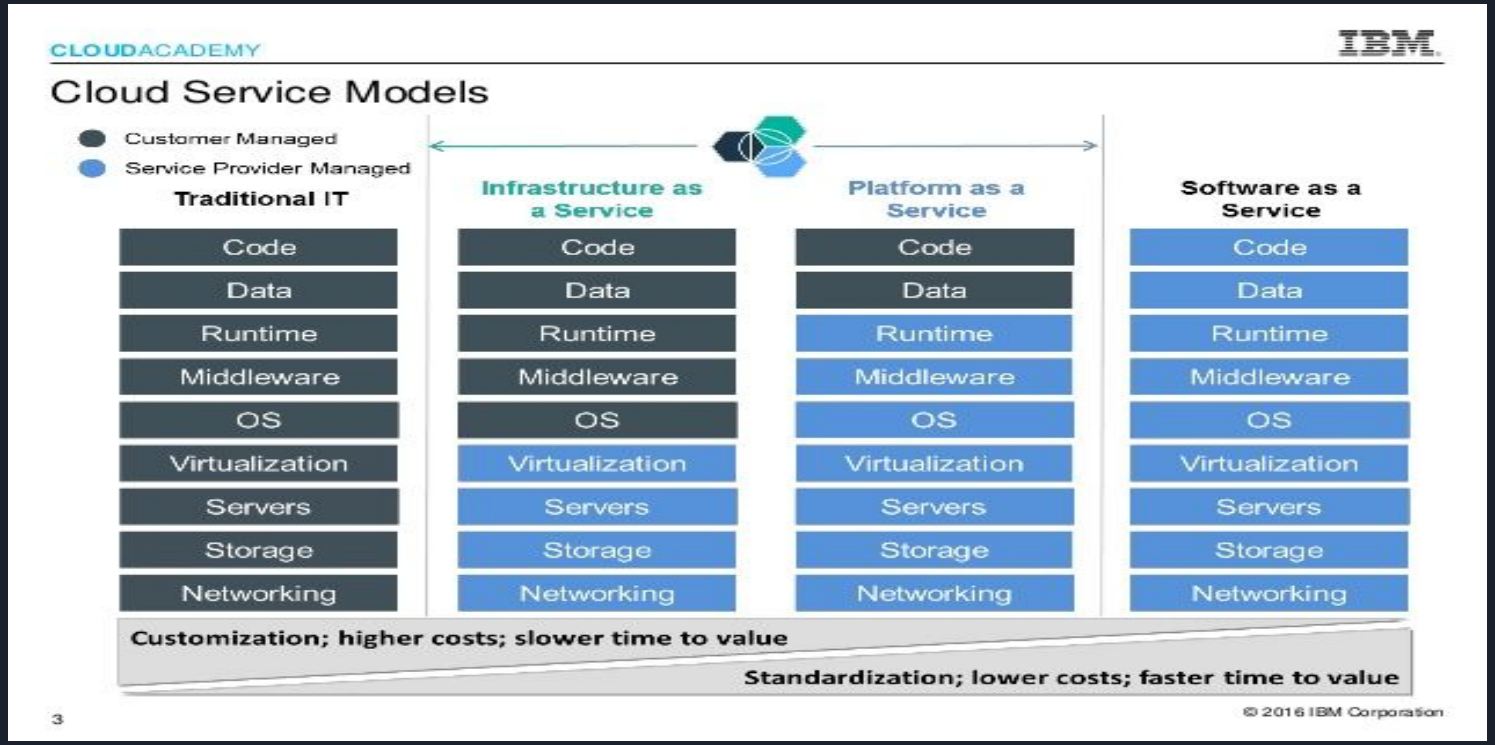
IaaS	Infrastructure as a Service
CaaS	Container as a Service
PaaS	Platform as a Service
FaaS	Function as a Service
SaaS	Software as a Service

Schéma

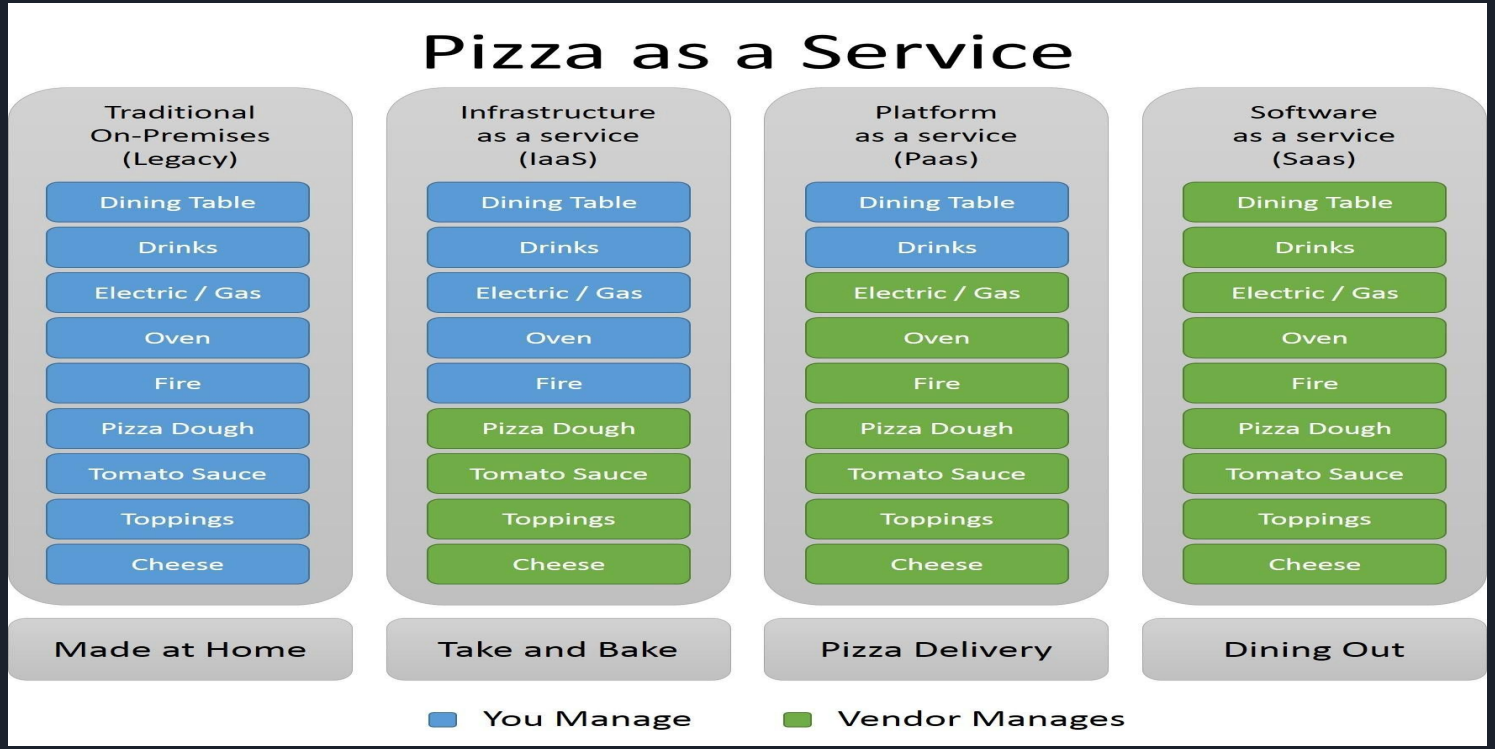

INFRASTRUCTURE PLATFORM (IaaS)
OpenStack
vSphere
Azure Stack VMs
CONTAINER PLATFORM (CaaS)
Kubernetes
DC/OS
Docker Datacenter
APPLICATION PLATFORM (PaaS / aPaaS)
CloudFoundry
OpenShift
WaveMaker RAD
FUNCTION PLATFORM (FaaS)
OpenWhisk
Fission
Iron.io
SOFTWARE PLATFORM (SaaS)
BYO
AWS EC2
GCE
Azure VMs
GKE
ECS
ACS
Heroku
PCF
Jelastic
Lambda
GCF
Azure Functions
Salesforce
Oracle
SAP
HOSTED

IaaS / PaaS / SaaS Providers

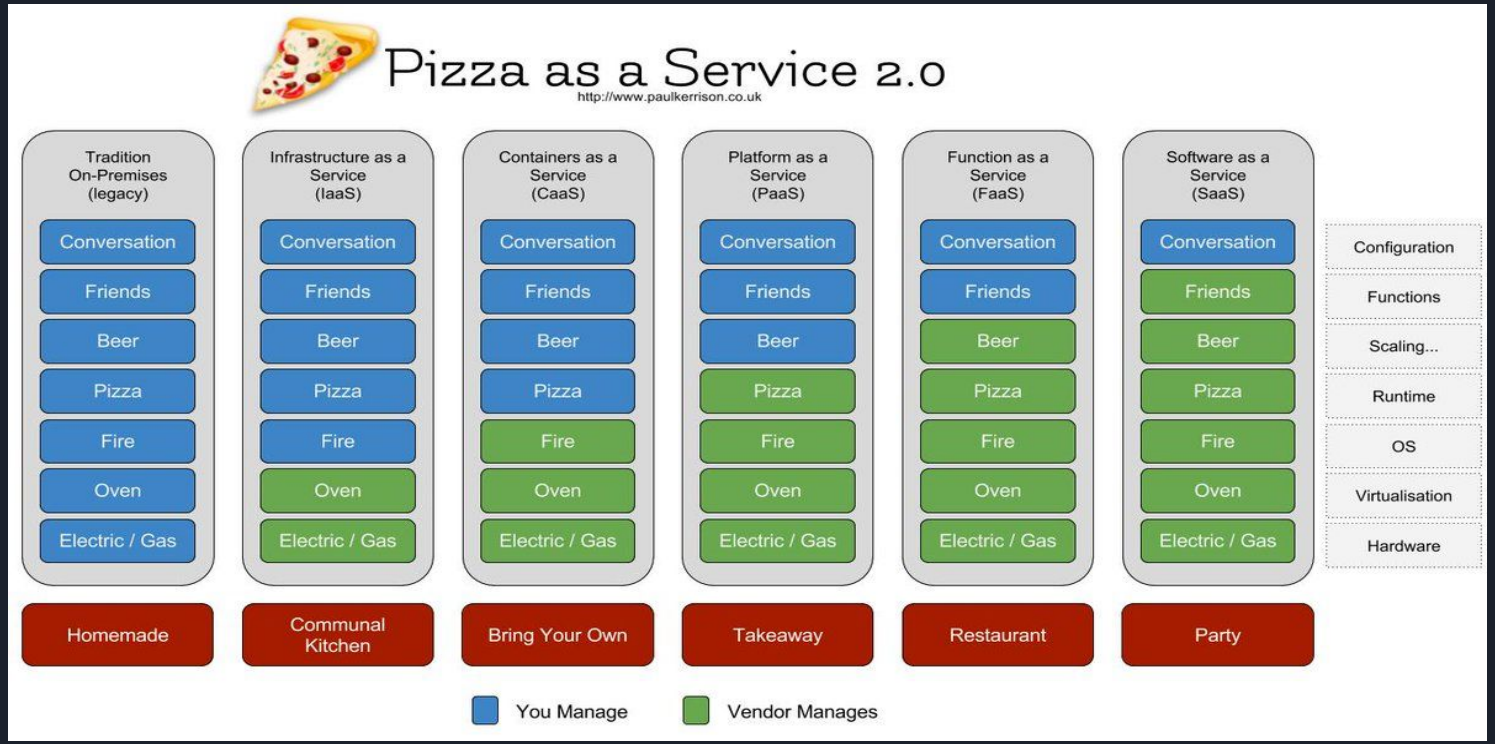
Modèles de service cloud



Pizza as a Service (by Albert Barron)

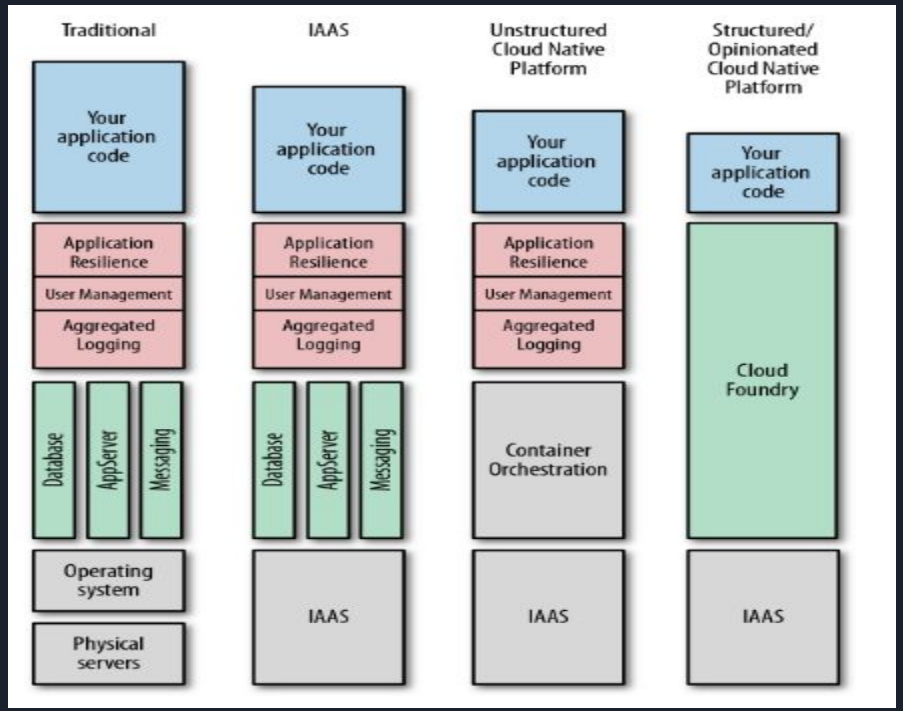


Pizza as a Service 2.0

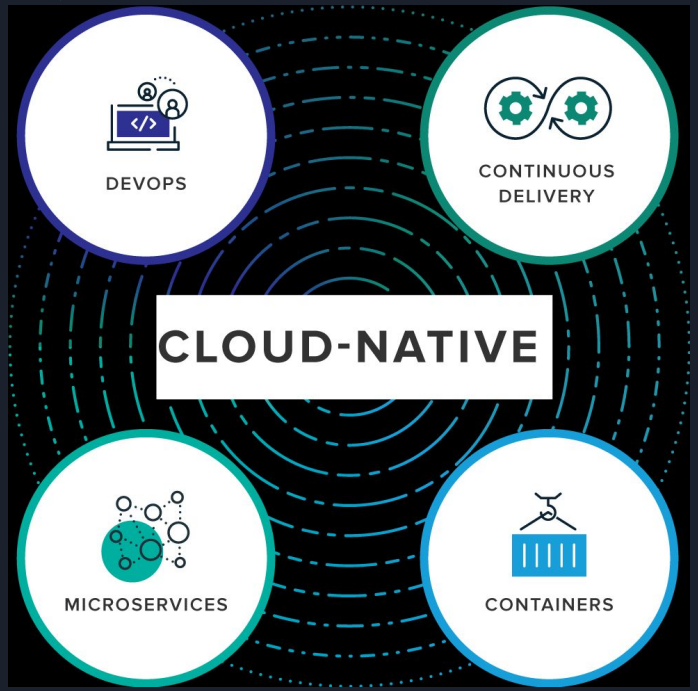


Cloud-Native

Cloud-Native : Platform Concepts



Cloud-Native : Applications



- Évolution rapide (Time To Market)
- Déploiement facile & automatique
- Montent en charge
  - scaling horizontal
  - scaling vertical
- Disponibles 24/7
- Sécurisées
- Accessibles par tous types de clients
- Stateless

Cloud-Native : Architecture

- The Twelve-Factor
- Microservices
- Infrastructure à la demande & en self-service
- Utilisation au travers d’API de services

Cloud Native : Twelve-factor (best-practices) https://12factor.net/fr/

I. Base de code
Une base de code suivie avec un système de contrôle de version, plusieurs déploiements

II. Dépendances
Déclarez explicitement et isolez les dépendances

III. Configuration
Stockez la configuration dans l'environnement

IV. Services externes
Traitez les services externes comme des ressources attachées

V. Build, release, run
Séparez strictement les étapes d'assemblage et d'exécution

VI. Processus
Exécutez l'application comme un ou plusieurs processus sans état

VII. Associations de ports
Exportez les services via des associations de ports

VIII. Concurrence
Grossissez à l'aide du modèle de processus

IX. Jetable
Maximisez la robustesse avec des démarrages rapides et des arrêts gracieux

X. Parité dev/prod
Gardez le développement, la validation et la production aussi proches que possible

XI. Logs
Traitez les logs comme des flux d'événements

XII. Processus d'administration
Lancez les processus d'administration et de maintenance comme des one-off-processes

Cloud Native Landscape

v1.0

App Definition & Development

Database & Data Analytics

Streaming

SCM

Application Definition

CI/CD

Orchestration & Management

Scheduling & Orchestration

kubernetes — CNCF Project

Coordination & Service Discovery

CoreDNS — CNCF Project

Service Management

linkerd — CNCF Project

GRPC — CNCF Project

envoy — CNCF Project

Runtime

Cloud-Native Storage

Container Runtime

containerd — CNCF Project

rkt — CNCF Project

Cloud-Native Network

CNI — CNCF Project

Provisioning

Host Management / Tooling

Infrastructure Automation

Container Registries

Secure Images

Notary — CNCF Project

TUF — CNCF Project

Key Management

Cloud

Public

Private

Platforms

PaaS / Container Service

Serverless/Event-based

Observability & Analysis

Monitoring

Prometheus — CNCF Project

Logging

fluentd — CNCF Project

Tracing

OpenTracing — CNCF Project

Jaeger — CNCF Project

github.com/cncf/landscape

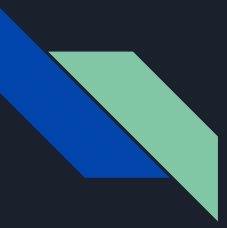
This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path.

CLOUD NATIVE COMPUTING FOUNDATION

Redpoint

Amplify Partners

Greyed logos are not open source



PaaS

Cloud Foundry

PaaS Cloud Foundry

Cloud Foundry est un PaaS open source qui permet de créer, de déployer, d'exécuter et de faire évoluer des applications sur des modèles de Cloud public et de Cloud privé. Cloud Foundry a été créé à l'origine par VMware et appartient désormais à Pivotal Software.

En février 2014, Pivotal, EMC, IBM, Rackspace et VMware ont formé la Cloud Foundry Foundation, qui compte à ce jour plus de 30 membres. Cette fondation indépendante à but non lucratif applique une politique de gouvernance qui permet à toute entreprise d'y contribuer.

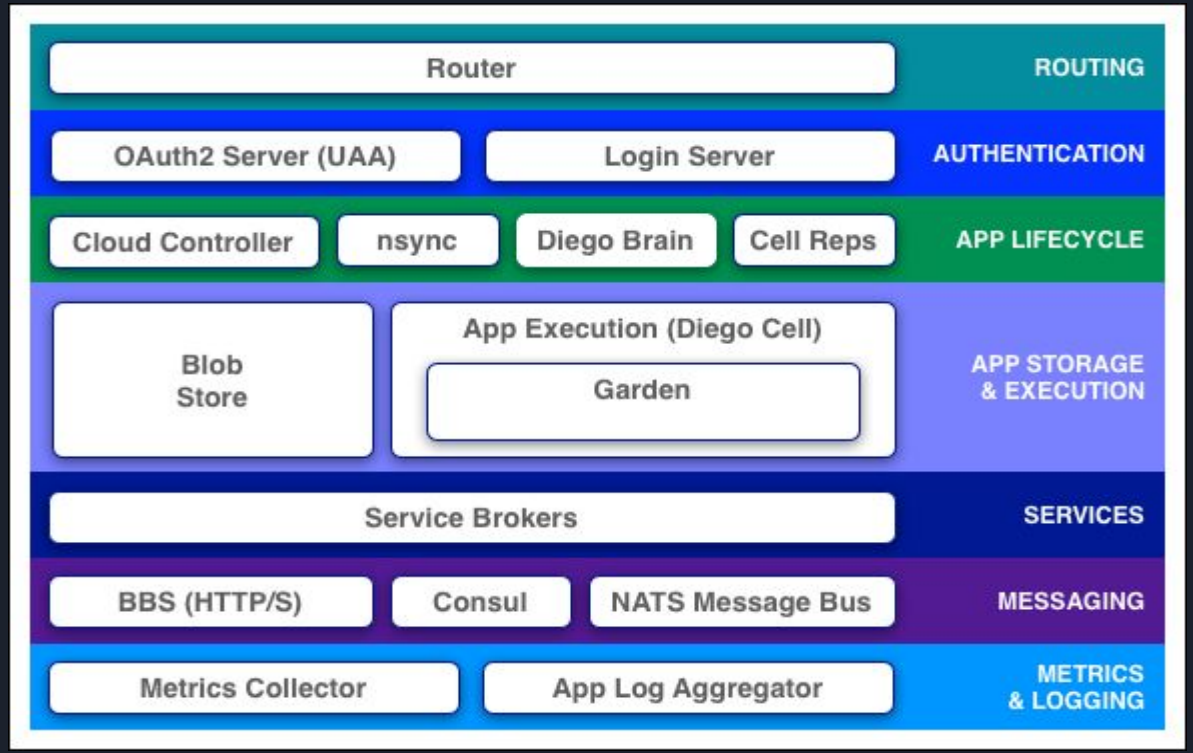
Source : http://www.lemagit.fr/definition/Cloud-Foundry

PaaS Cloud Foundry

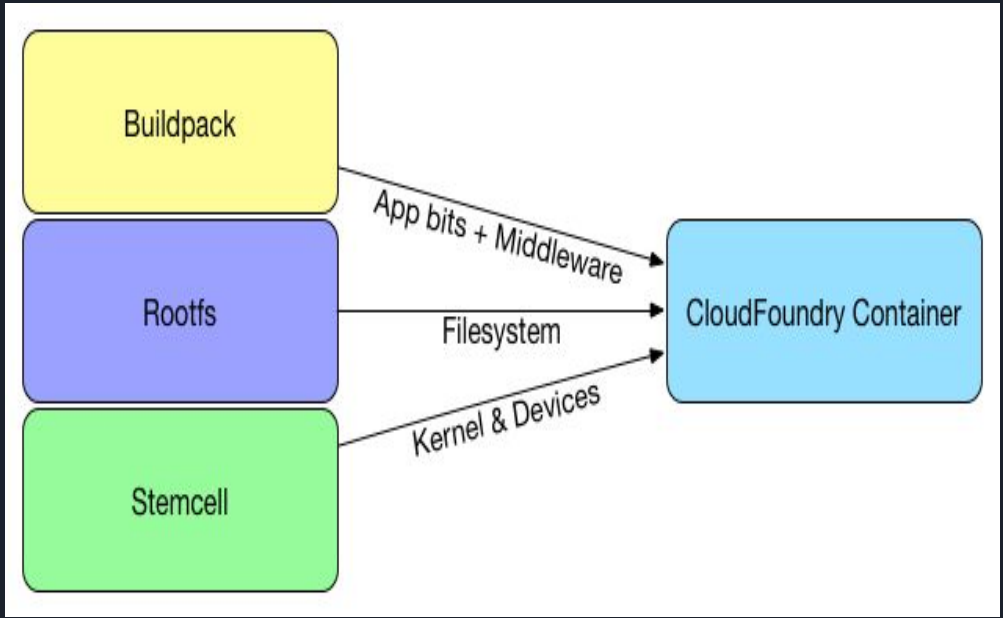
- Qu'est ce qu'un PaaS ?
  - Une VM
  - Un OS
  - Un environnement d'exécution pour une application
    (multi-langages : Java, PHP, Nodejs, Go, Ruby, Python, .NET, ...)

- Projet Open Source
  - www : https://www.cloudfoundry.org
  - Github : https://github.com/cloudfoundry

Cloud Foundry : Architecture

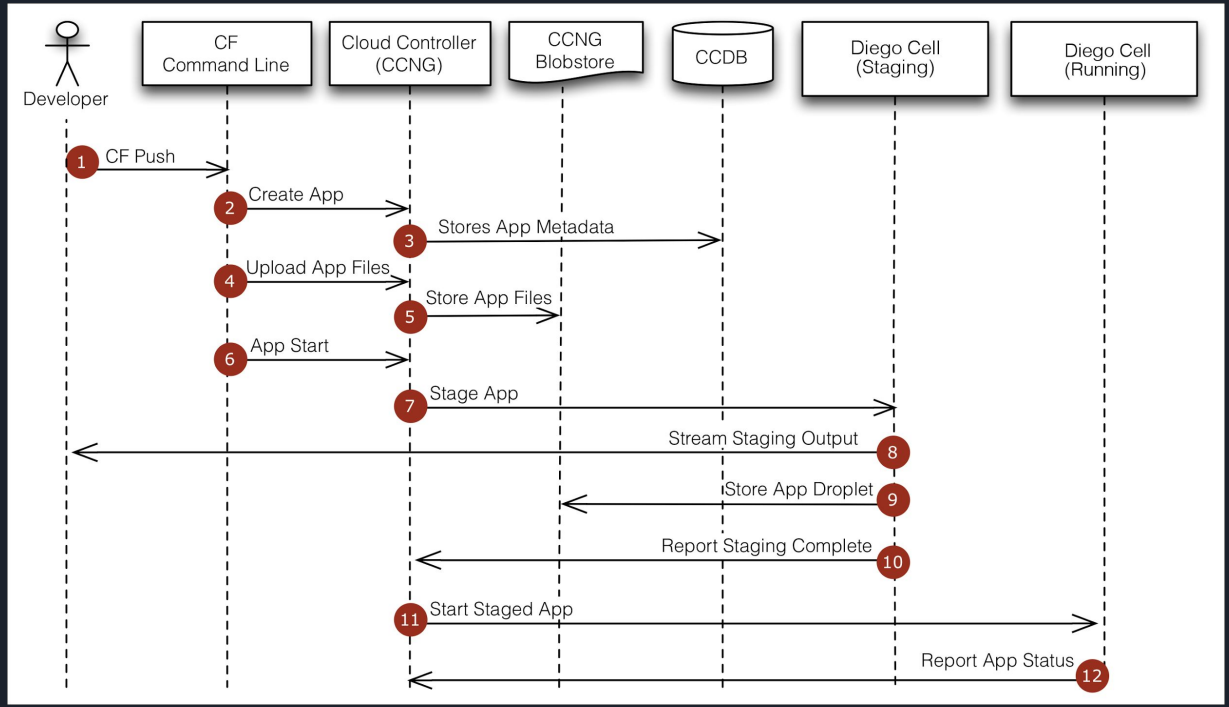


Cloud Foundry : Container

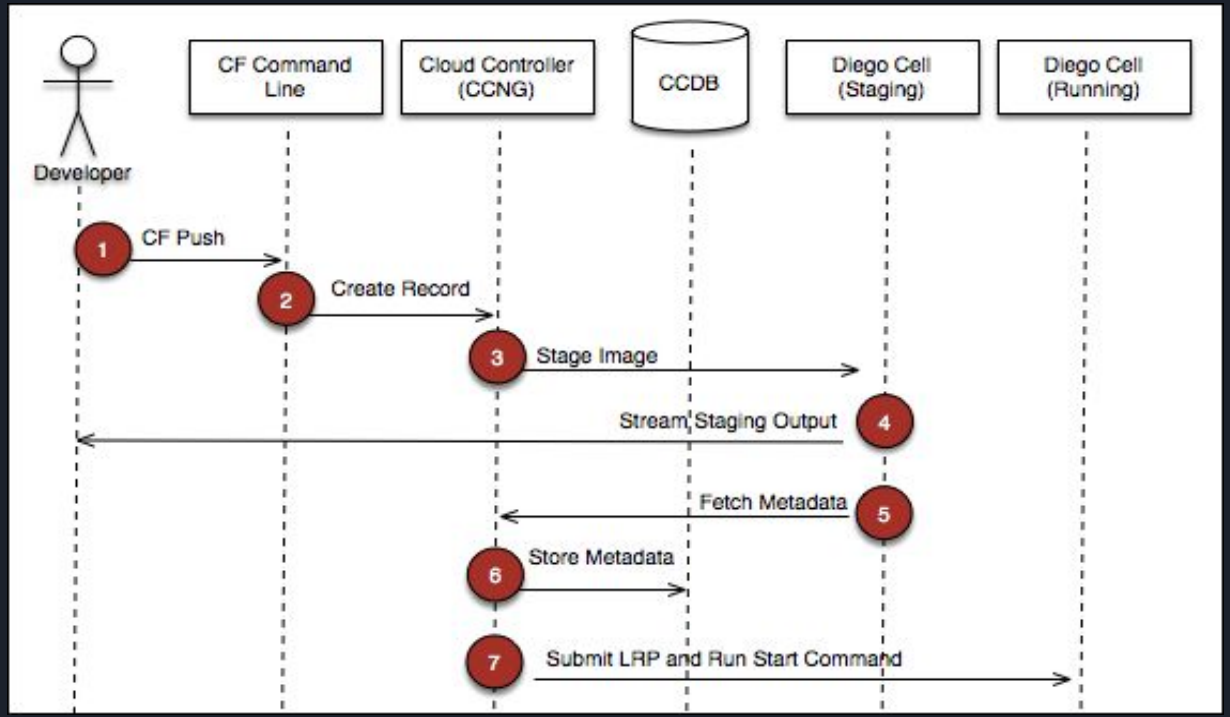


- Buildpack
- Rootfs
- Stemcell

APP (1/2) :
How Diego Stages Buildpack Applications



APP (2/2)
How Diego Stages Docker Images



CLI : Installation (cf command)

Use a Compressed Binary

Download the compressed binary for Mac OS X, Windows, or Linux from the cf CLI GitHub repository and install it on your system.

The specific procedures vary by operating system, but the following example illustrates downloading and installing the binary on Mac OS X:

1. Download and extract the Mac OS X binary:

```
$ curl -L "https://cli.run.pivotal.io/stable?release=macosx64-binary&source=github" | t
```

2. Move it to `/usr/local/bin`, or another location in your `$PATH`:

```
$ mv cf /usr/local/bin
```

3. Confirm your cf CLI version:

```
$ cf --version
```

CLI : cf help [-a]

- https://docs.cloudfoundry.org/cf-cli/cf-help.html

Commandes Cloud Foundry (cf)

Dernière mise à jour : 2017-05-03

L'interface de ligne de commande Cloud Foundry (cf) fournit un ensemble de commandes permettant de gérer vos applications. Les informations ci-après répertorient les commandes cf le plus souvent utilisées pour gérer les applications avec leurs noms, leurs options, leur syntaxe, les éléments prérequis, leur description et des exemples. Pour afficher toutes les commandes cf et les informations d'aide associées, entrez `cf help`. Entrez `cf nom_commande -h` afin d'afficher des informations d'aide détaillées pour une commande particulière.

CLI : cf login

```
$ cf login -a https://api.example.com -u username@example.com
API endpoint: https://api.example.com

Password>
Authenticating...
OK

Select an org (or press enter to skip):
1. example-org
2. example-other-org

Org> 1
Targeted org example-org

Select a space (or press enter to skip):
1. development
2. staging
3. production

Space> 1
Targeted space development
```

CLI : cf org-users

```
$ cf org-users example-org
Getting users in org example-org as username@example.com...

ORG MANAGER
  username@example.com

BILLING MANAGER
  huey@example.com
  dewey@example.com

ORG AUDITOR
  louie@example.com
```

CLI : cf push

```
$ cf push my-awesome-app -b ruby_buildpack
Creating app my-awesome-app in org example-org / space development as username@example.com
OK

Creating route my-awesome-app.example.com...
OK
...

1 of 1 instances running

App started
...

requested state: started
instances: 1/1
usage: 1G x 1 instances
urls: my-awesome-app.example.com
last uploaded: Wed Jun 8 23:43:15 UTC 2016
stack: cflinuxfs2
buildpack: ruby_buildpack

     state     since                  cpu    memory    disk      details
#0   running   2016-06-08 04:44:07 PM   0.0%   0 of 1G   0 of 1G
```

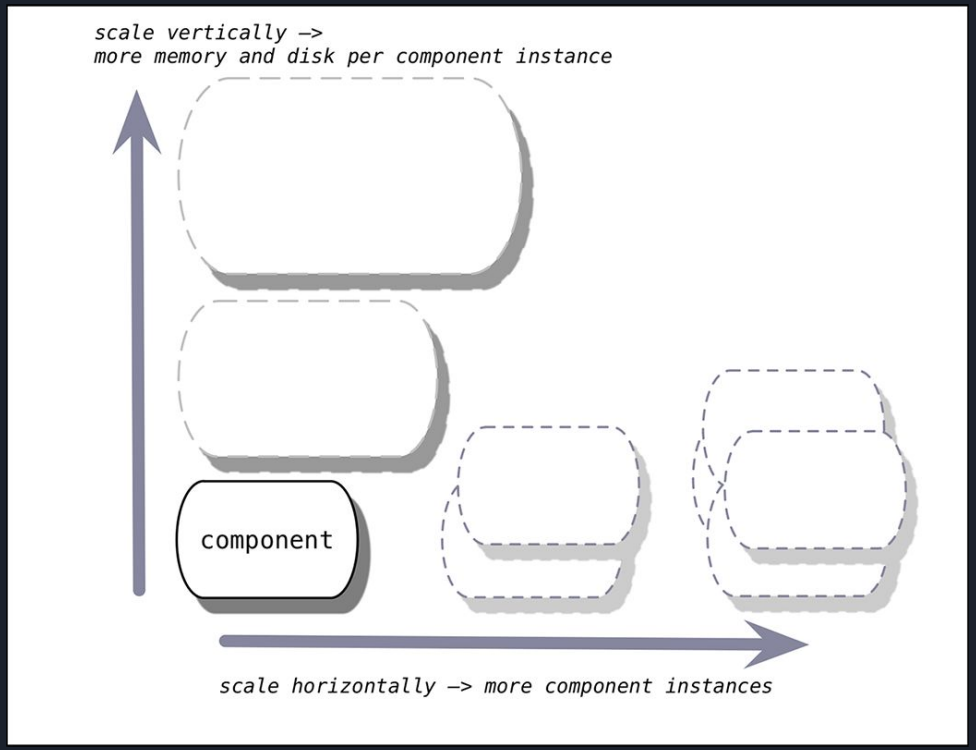
CLI : cf apps

```
ubuntu@km-dev:~$ cf apps
Getting apps in org default_organization / space space01 as admin...
OK

name         requested state   instances   memory   disk   urls
httpserver   started           1/1         1G       1G     httpserver.140.211.168.100.xip.io
ubuntu@km-dev:~$ cf app httpserver
Showing health and status for app httpserver in org default_organization / space space01 as admin...
OK

requested state: started
instances: 1/1
usage: 1G x 1 instances
urls: httpserver.140.211.168.100.xip.io
last uploaded: Mon Oct 31 13:30:17 UTC 2016
stack: cflinuxfs2
buildpack: unknown

     state     since                    cpu    memory        disk         details
#0   running   2016-10-31 01:32:36 PM   0.1%   17.2M of 1G   1.3M of 1G
ubuntu@km-dev:~$
```

CLI : cf scale

```
praneethramesh@praneeth-ramesh:~/Software/Development/PCFDev$ cf scale pcf-demo-app -i 2
Scaling app pcf-demo-app in org pcfdev-org / space pcfdev-space as admin...
OK
praneethramesh@praneeth-ramesh:~/Software/Development/PCFDev$ cf app pcf-demo-app
Showing health and status for app pcf-demo-app in org pcfdev-org / space pcfdev-space as admin...

name:              pcf-demo-app
requested state:   started
instances:         2/2
usage:             1G x 2 instances
routes:            pcf-demo-app.local.pcfdev.io
last uploaded:     Wed 26 Apr 21:51:13 EDT 2017
stack:             cflinuxfs2
buildpack:         https://github.com/cloudfoundry/java-buildpack

     state      since                  cpu    memory          disk            details
#0   running    2017-04-27T01:53:10Z   0.2%   336.6M of 1G    153.1M of 512M
#1   starting   2017-04-27T01:55:26Z   0.0%   57.7M of 1G     153.1M of 512M
praneethramesh@praneeth-ramesh:~/Software/Development/PCFDev$
```

Scaling concepts



CLI : cf delete

```
$ cf delete -h
NAME:
   delete - Delete an app

USAGE:
   cf delete APP_NAME [-f -r]

ALIAS:
   d

OPTIONS:
   -f        Force deletion without confirmation
   -r        Also delete any mapped routes
```

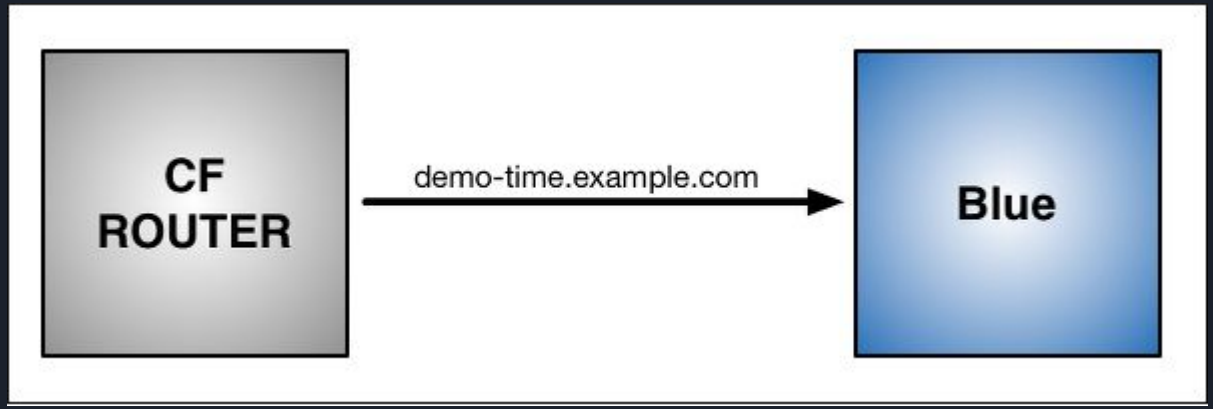
CLI : cf en vrac

- cf target
- cf buildpacks
- cf stacks
- cf marketplace
- cf services
- cf service-brokers
- cf domains
- cf routes
- cf plugins (https://plugins.cloudfoundry.org)
- cf security-groups
- cf restage APP_NAME
- cf env APP_NAME
- cf logs APP_NAME
- cf ssh APP_NAME
- ...

Déploiement Blue / Green

Step 1: Push an App

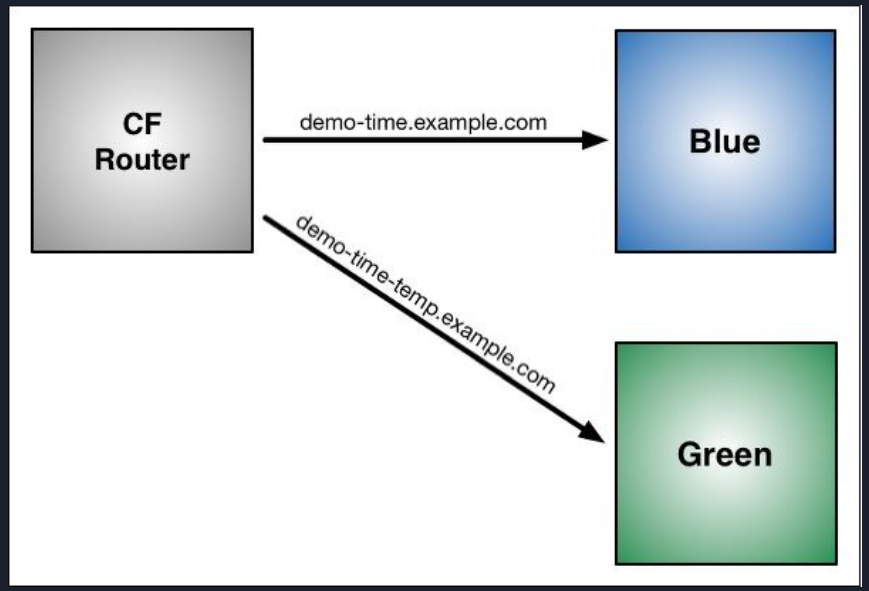
```
$ cf push Blue -n demo-time
```



Déploiement Blue / Green

Step 2: Update App and Push

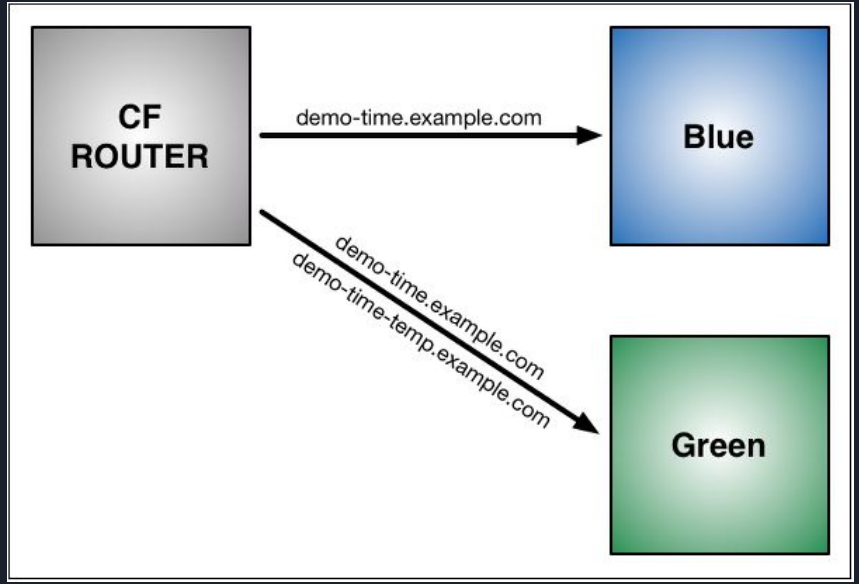
```
$ cf push Green -n demo-time-temp
```



Déploiement Blue / Green

Step 3: Map Original Route to Green

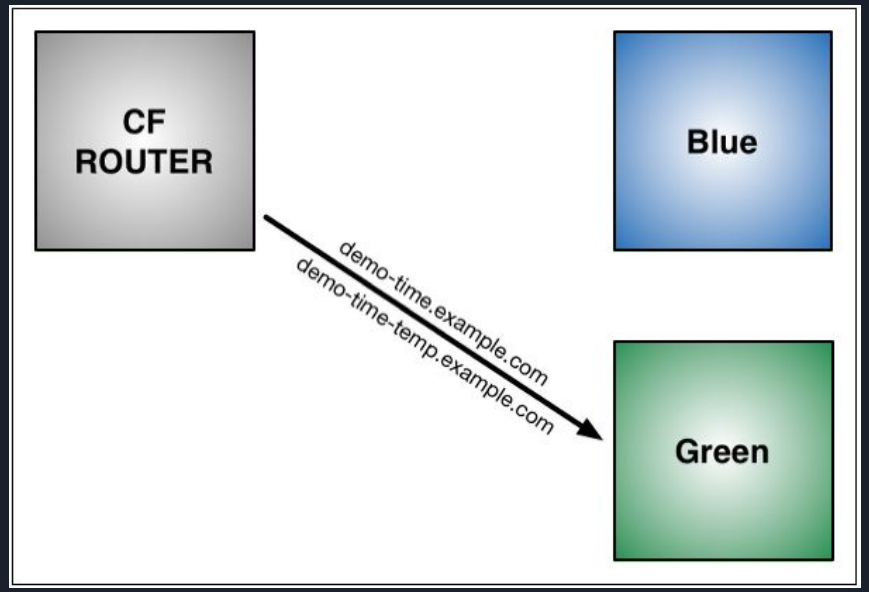
```
$ cf map-route Green example.com -n demo-time
Binding demo-time.example.com to Green... OK
```



Déploiement Blue / Green

Step 4: Unmap Route to Blue

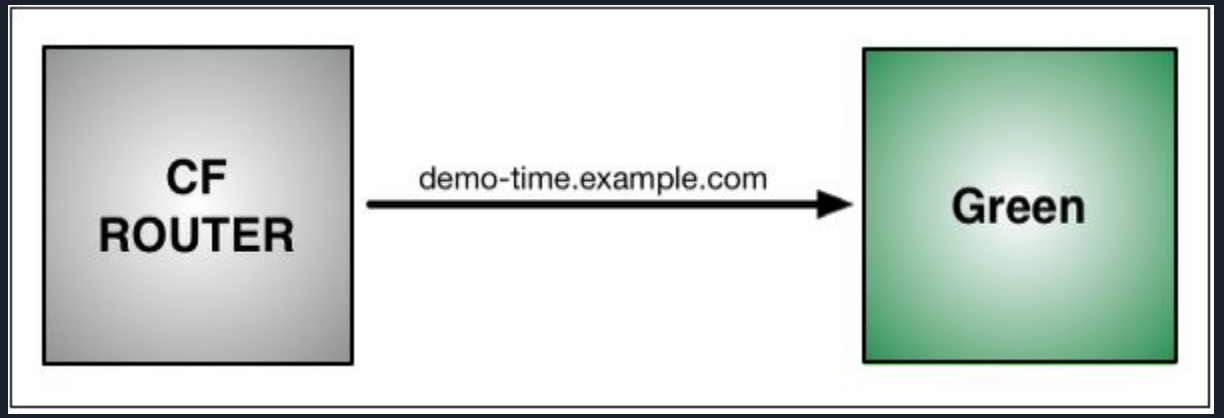
```
$ cf unmap-route Blue example.com -n demo-time
Unbinding demo-time.example.com from blue... OK
```



Déploiement Blue / Green

Step 5: Remove Temporary Route to Green

```
$ cf unmap-route Green
example.com -n demo-time-temp
```



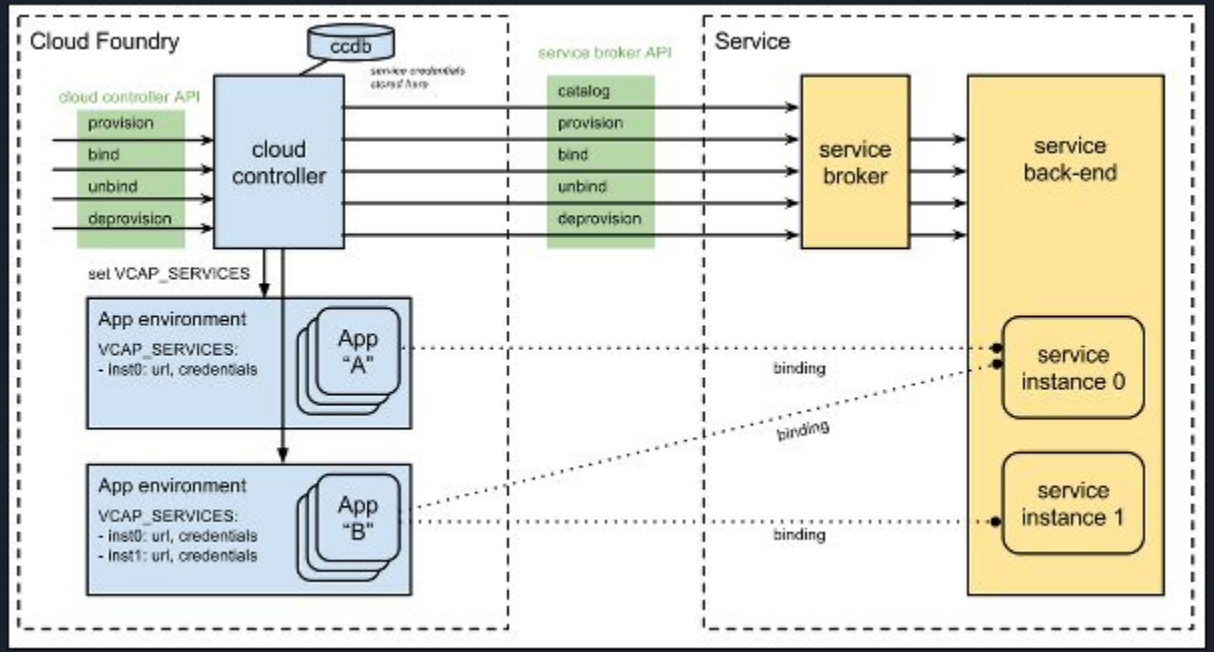
Service : Définition

Cloud Foundry permet aux applications hébergées d'accéder à des services. Ces services offrent à vos applications un moyen d'accès à divers types de ressources. Les exemples sont multiples :

- des mécanismes de persistance : *Amazon S3*, *Hubic*, mais aussi *MySQL*, *PostgreSQL*, *Cassandra*…
- des moteurs d'indexation : *ElasticSearch*, *Solr*…
- des systèmes de messagerie : *RabbitMQ*, *Postfix*…
- …

Source : http://www.cloudmagazine.fr/avis-expert/les-services-sous-cloud-foundry-partie-1-kesako

Services : Architecture

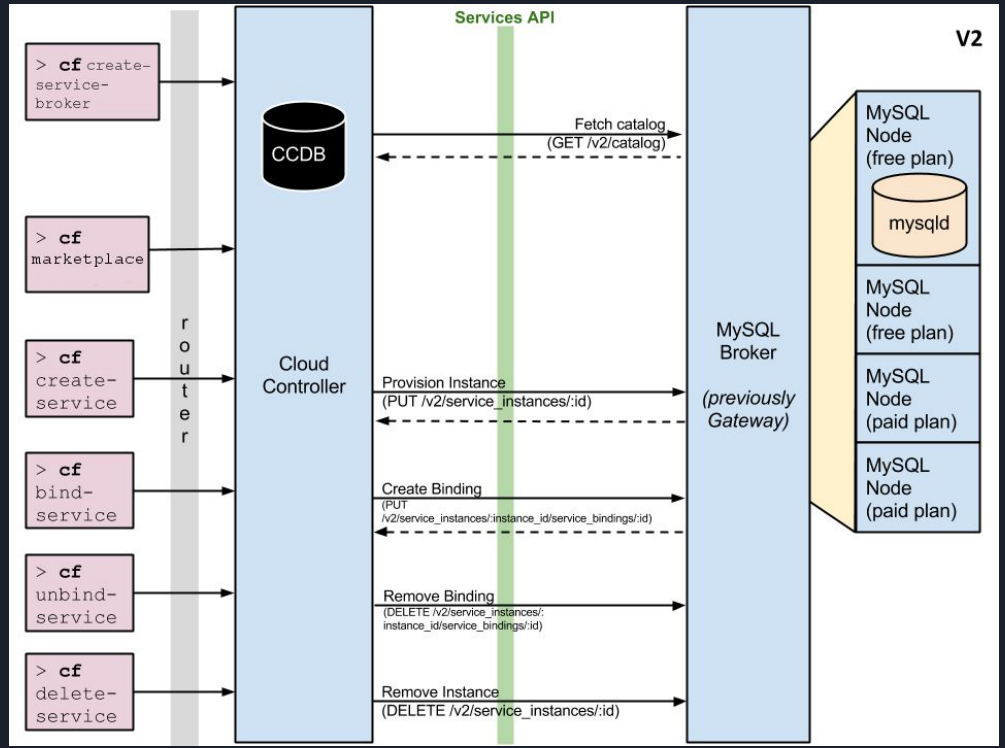


Service broker : Définition

Dans le jargon Cloud Foundry, un *service broker* est un service web permettant aux applications hébergées d'interagir avec un service dans le cadre d'une offre. Il n'est pas en charge de l'accès à ce service mais a pour responsabilité ***d'associer sur demande*** une application à une offre. On parle bien ici de *service web* au sens le plus général du terme. Cloud Foundry n'impose aucune contrainte particulière si ce n'est que ce service web doit être accessible depuis Cloud Foundry et implémenter une API spécifique. Libre à vous de l'implémenter via la stack de ***votre choix*** – Java, Ruby, Python, PHP…

Source : http://www.cloudmagazine.fr/avis-expert/les-services-sous-cloud-foundry-partie-1-kesako

Service broker : Fonctionnement



Service brokers : Rôles

On peut donc imaginer qu'il est en charge des éléments suivants :

- la création du compte utilisateur
- la création de l'instance ou de la base de données MySQL
- l'attribution des droits
- la mise à disposition des informations de connexions à travers l'API Cloud Foundry

Source : http://www.cloudmagazine.fr/avis-expert/les-services-sous-cloud-foundry-partie-1-kesako

Routes & domains

```
$ cf routes
Getting routes as user@private-domain.example.com ...

space      host     domain                        port   path       type  apps
my-space   myapp    shared-domain.example.com                       myapp
my-space   myapp    private-domain.example.com                      myapp
my-space   store    shared-domain.example.com          /products    products
my-space   store    shared-domain.example.com          /orders      orders
my-space   store    shared-domain.example.com                       storefront
my-space            shared-domain.example.com   60000              tcp   tcp-app
```

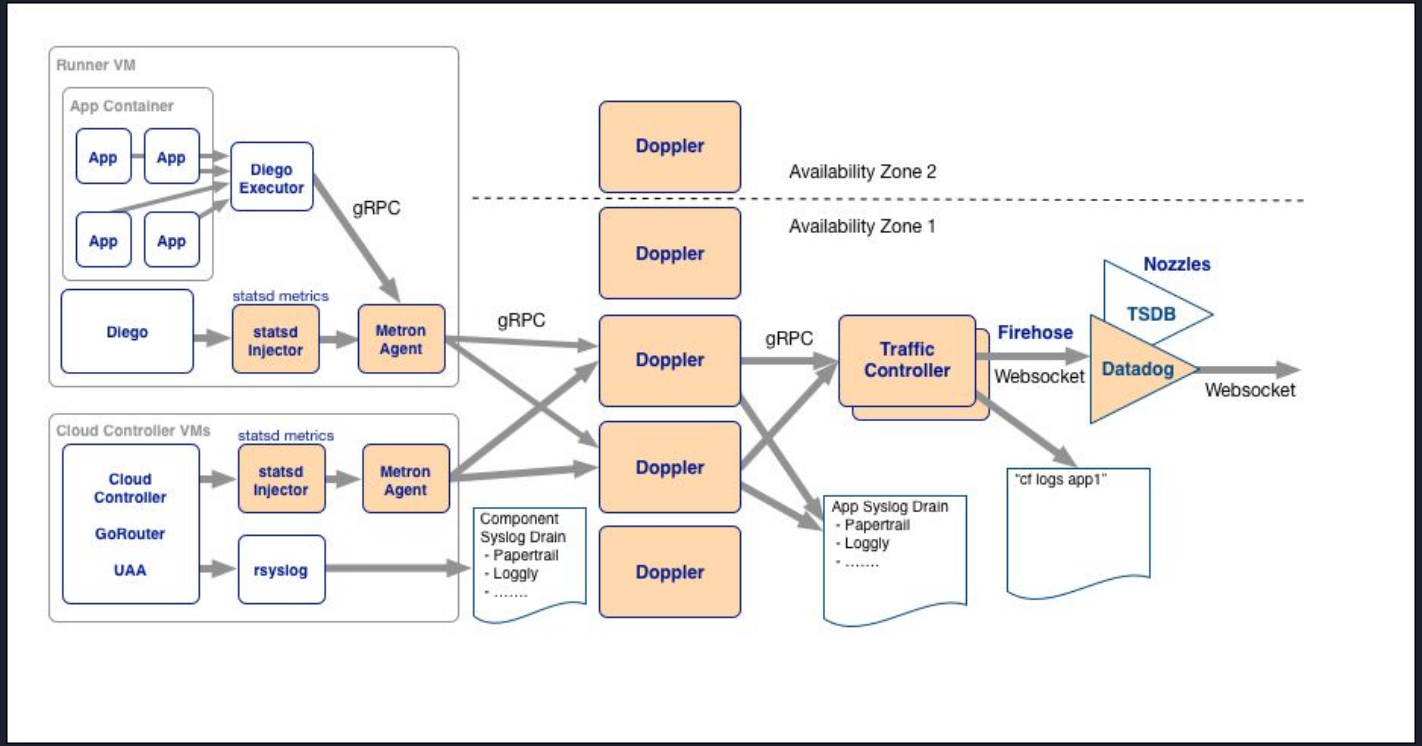
```
$ cf domains
Getting domains in org my-org as user@example.com... OK
name                      status   type
shared-domain.example.com         shared
    shared   tcp
private-domain.example.com        owned
```

Source : https://docs.cloudfoundry.org/devguide/deploy-apps/routes-domains.html

Security groups

```
[
  {
    "protocol": "tcp",
    "destination": "10.0.0.0-10.255.255.255",
    "ports": "443"
  },
  {
    "protocol": "tcp",
    "destination": "172.16.0.0-172.31.255.255",
    "ports": "443"
  },
  {
    "protocol": "tcp",
    "destination": "192.168.0.0-192.168.255.255",
    "ports": "443"
  }
]
```

Logging & Metrics
Loggregator Architecture



BOSH : Définition

BOSH communique avec un seul IaaS qui fournit le réseau sous-jacent et les VM (ou conteneurs).

Plusieurs fournisseurs IaaS sont pris en charge:

- Amazon Web Services EC2
- Apache CloudStack
- Google Compute Engine
- Microsoft Azure
- OpenStack
- VMware vSphere.

Dans le but de prendre en charge plus d'infrastructures, BOSH utilise un concept d'interface fournisseur de cloud (CPI). Il y a une implémentation de CPI pour chacune des IaaS listées ci-dessus. Plus généralement, le CPI est utilisé pour déployer des machines virtuelles, mais il peut être utilisé pour déployer des conteneurs.

BOSH : Architecture

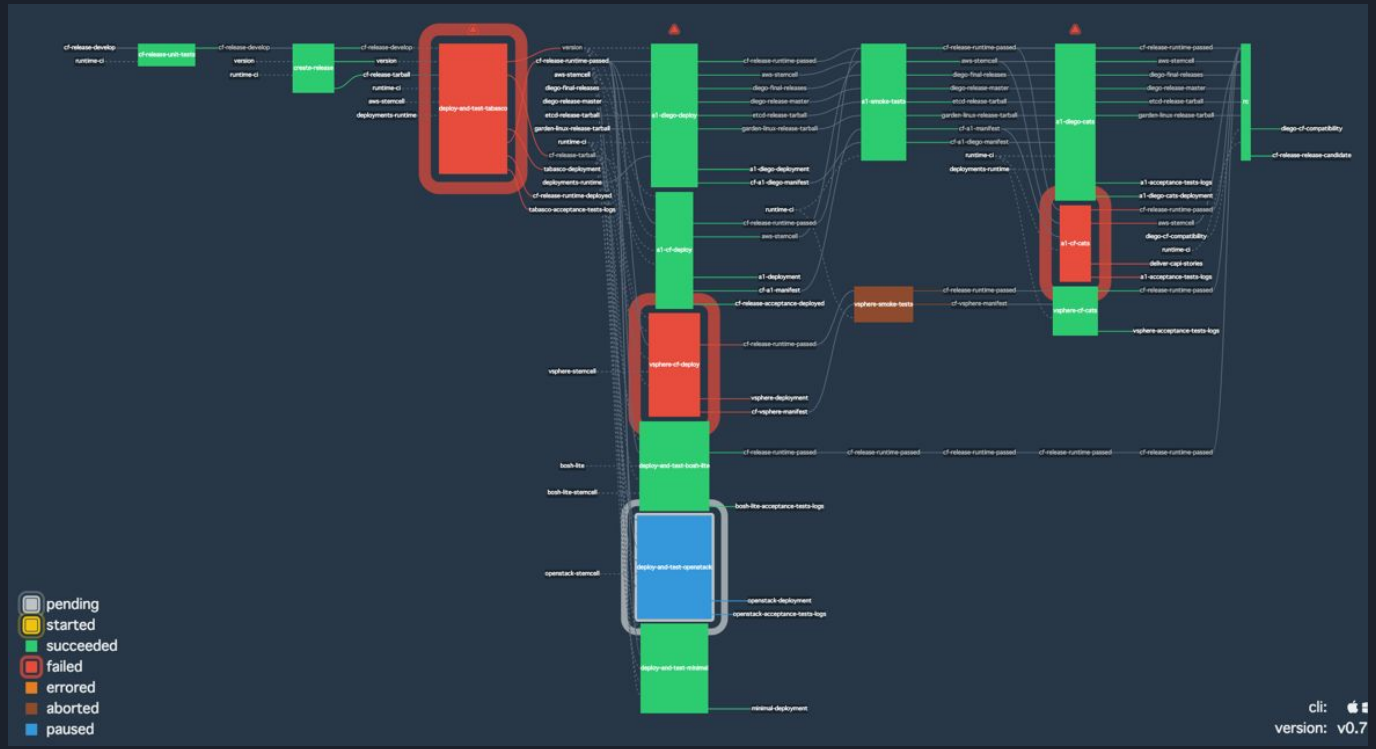
```
ubuntu@km-dev:~$ bosh vms
RSA 1024 bit CA certificates are loaded due to old openssl compatibility
Acting as user 'admin' on 'my-bosh'
Deployment 'cf-diego'

Director task 449

Task 449 done

+-------------------------------------------------------------------------+---------+-----+---------------+-----------------+
| VM                                                                      | State   | AZ  | VM Type       | IPs             |
+-------------------------------------------------------------------------+---------+-----+---------------+-----------------+
| api_z1/0 (79377423-eee8-4c55-b1ec-3b67d4b16e24)                         | running | n/a | small_z1      | 172.16.1.159    |
| blobstore_z1/0 (7ff71444-db44-46b4-ab1b-867560bf8177)                   | running | n/a | small_z1      | 172.16.1.158    |
| consul_z1/0 (2602ca79-65fa-47d0-90bb-93232fca13e5)                      | running | n/a | small_z1      | 172.16.1.155    |
| diego_brain_z1/0 (a4035921-3441-4590-8fb6-0003746fcc40)                 | running | n/a | diego_z1      | 172.16.1.157    |
| diego_cell_z1/0 (f51ad1a5-0ca8-40e3-90e8-7281858f9e60)                  | running | n/a | diego_cell_z1 | 172.16.1.156    |
| doppler_z1/0 (cd44db9c-0504-4efb-bf73-8d5ab1c5bc12)                     | running | n/a | small_z1      | 172.16.1.160    |
| etcd_z1/0 (ce4e3336-7d36-4395-892c-0a4c8c2f5da3)                        | running | n/a | small_z1      | 172.16.1.154    |
| ha_proxy_z1/0 (05a813dc-589c-4131-a50d-feda1f8701c7)                    | running | n/a | small_z1      | 172.16.1.150    |
|                                                                         |         |     |               | 140.211.168.100 |
| loggregator_trafficcontroller_z1/0 (6801f3b1-c95d-4eda-9678-39066351ac4d) | running | n/a | small_z1      | 172.16.1.161    |
| nats_z1/0 (95369766-11b0-4419-a14f-73db64b63e8b)                        | running | n/a | small_z1      | 172.16.1.153    |
| postgres_z1/0 (a3aadc80-1d23-4af3-bce2-5bdb53576994)                    | running | n/a | small_z1      | 172.16.1.151    |
| router_z1/0 (f43b2001-3ee5-4635-919c-20f2bc3749bd)                      | running | n/a | small_z1      | 172.16.1.152    |
| uaa_z1/0 (1cdb1fdc-11df-433b-ab17-16db9394d021)                         | running | n/a | small_z1      | 172.16.1.162    |
+-------------------------------------------------------------------------+---------+-----+---------------+-----------------+

VMs total: 13
ubuntu@km-dev:~$
```

Concourse : Définition

Concourse est un système CI / CD remasterisé pour les équipes qui pratiquent le développement agile et doivent gérer des déploiements complexes. Concourse a été conçu en raison des frustrations des ingénieurs de chez Pivotal avec les systèmes d'intégration continue (CI) existants. Découvrez une nouvelle approche avec Concourse lorsque vous avez besoin de :

- Automatisez le développement piloté par les tests
- Maintenir la compatibilité entre plusieurs versions de build
- Cibler plusieurs plates-formes et configurations sur différents cloud
- Livraison fréquente : chaque semaine, tous les jours ou même plusieurs fois par jour

Concourse : A quoi cela ressemble ?



Liens utiles

- Cloud Foundry : https://www.cloudfoundry.org
  - Docs : https://docs.cloudfoundry.org
  - Github : https://github.com/cloudfoundry
  - CLI : https://github.com/cloudfoundry/cli
- Bosh : https://bosh.io
  - Docs : https://bosh.io/docs
  - Github : https://github.com/cloudfoundry/bosh
- Concourse : https://concourse.ci
  - Docs : https://concourse.ci/introduction.html
  - Github : https://github.com/concourse
  - Pipelines : https://buildpacks.ci.cf-app.com
- Pivotal : https://pivotal.io
  - Pcf-dev : https://pivotal.io/pcf-dev
- Cloud-Native Computing Foundation
  - https://github.com/cncf/landscape
- Autres liens :
  - Guide reference : https://cf-docs.jp-east-1.paas.cloud.global.fujitsu.com/en/manual/ref/ref/topics/preface.html

Merci

Questions ?