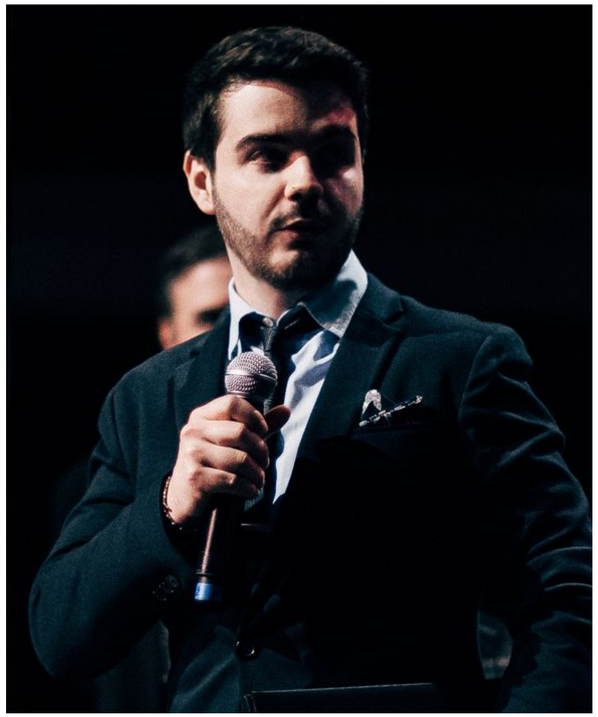


Samuel ANTUNES
Consultant Ingénieur DevSecOps
OCTO Technology
Email : contact@samuelantunes.fr

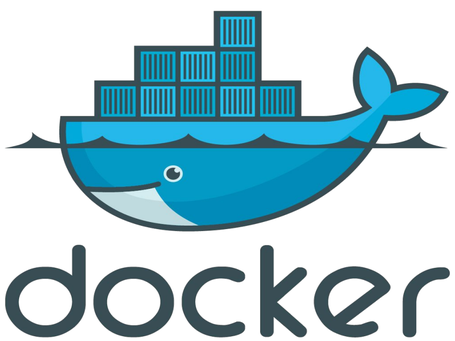
ICEBREAKER

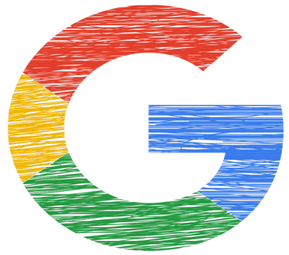
shorturl.at/HMOP2

AGENDA

1. Les bases de Kubernetes
2. Manipulation simple de Kubernetes
3. Mettre son application dans K8s
4. Le Continuous Delivery avec K8s
5. Conclusion & Take Away

“ Les bases de Kubernetes ”


docker


borg/omega


kubernetes

LA GENESE DE KUBERNETES

- D’anciens développeurs de Borg écrivent K8s en Go
- Directement pensé pour utiliser Docker (engine)
- Directement dans l’optique d’en faire un projet OpenSource
- Version 1.0 en Juin 2015

L'OBJECTIF DE KUBERNETES

- Définir et déployer des applications multi-conteneurs
- Répartir les conteneurs sur une flotte d'hôtes (nœuds)
- Optimiser et adapter le placement des conteneurs
- Surveiller la santé des conteneurs
- Définir et appliquer des contraintes de niveaux de services
- Gérer la disponibilité et la scalabilité des conteneurs
- Gérer le provisionnement et l'accès au stockage
- Isoler les conteneurs
  - Limitation de ressources
  - Sécurité (vision multi-tenant)

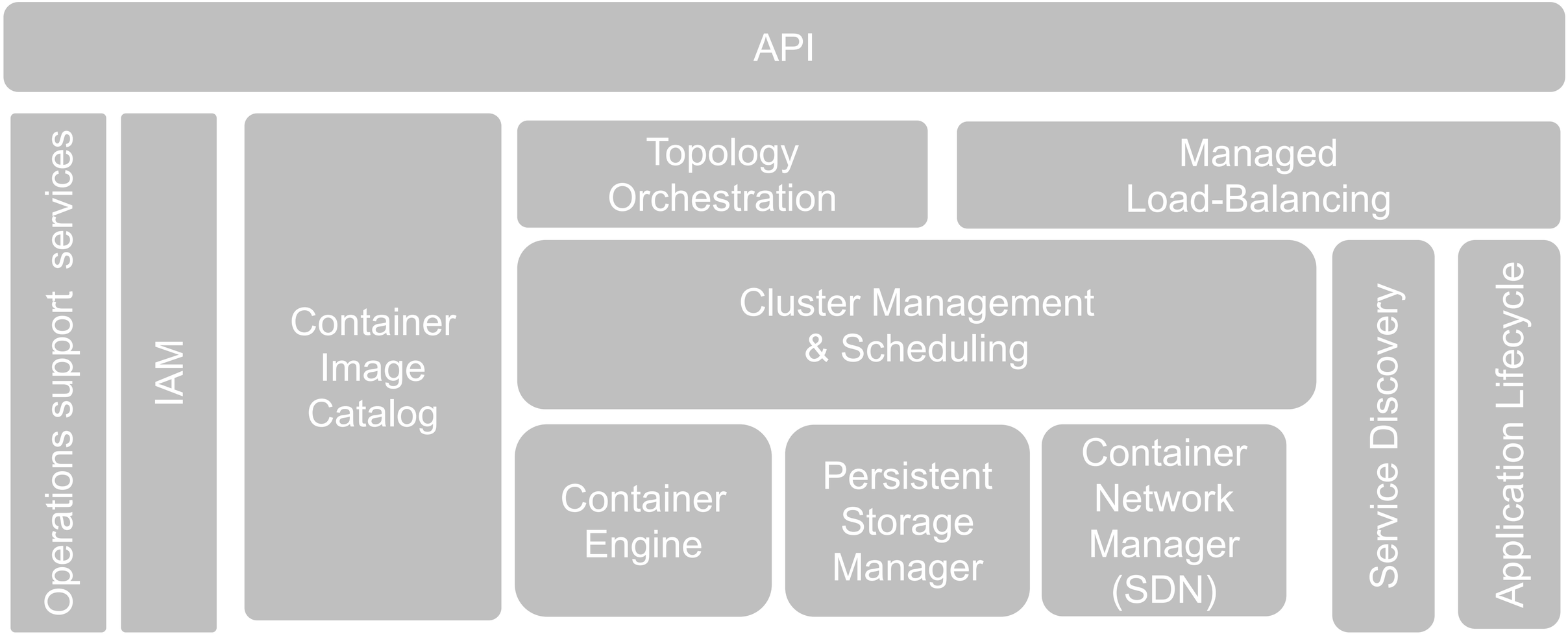
Tout ça de manière dynamique et pour des milliers de conteneurs !

L'APPROCHE DE KUBERNETES

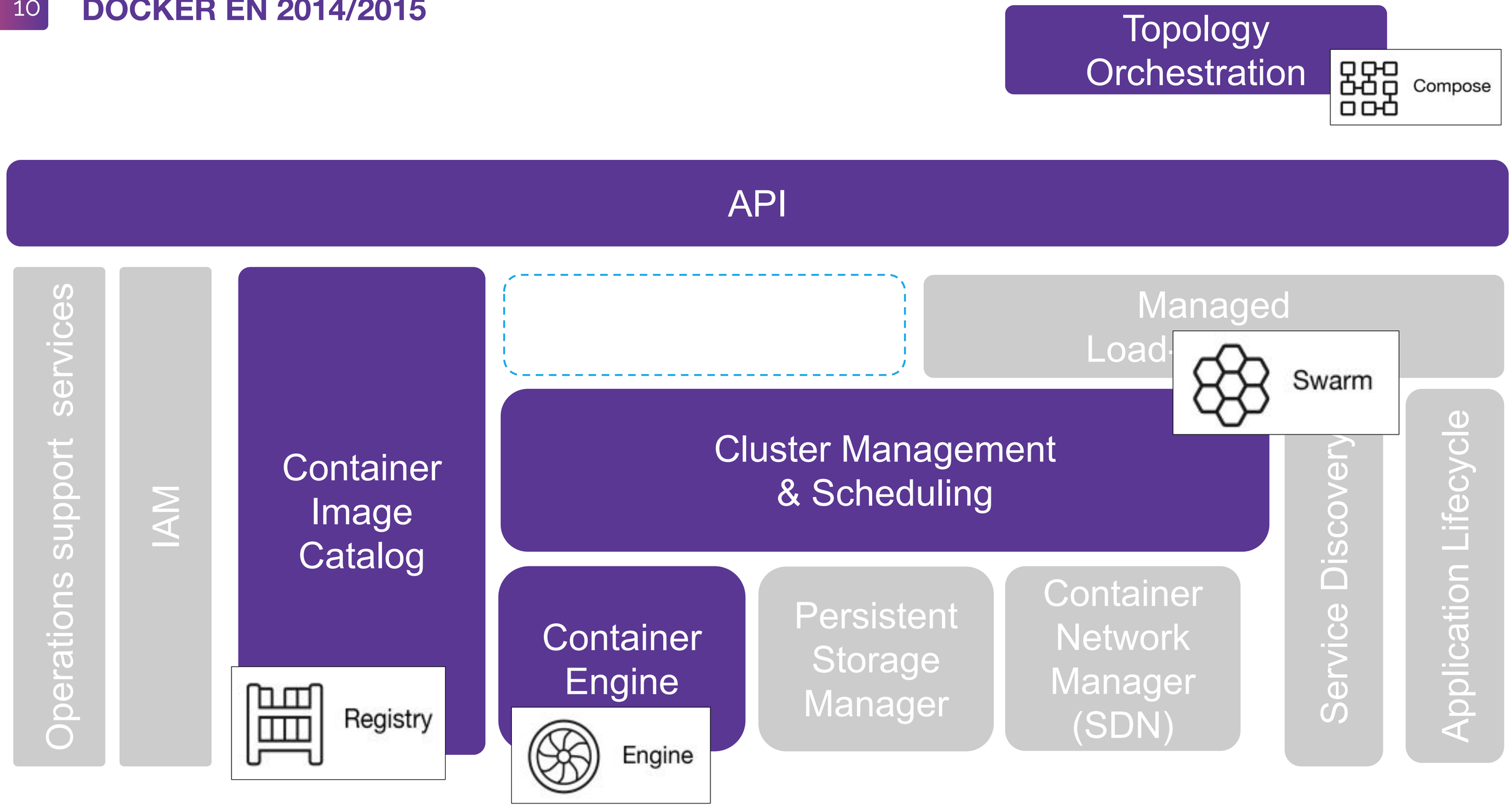
- Abstraction des concepts
  - *Apparition de différents types de ressources de haut niveau*
  - *On ne manipule que très rarement la notion de conteneur directement*

- Approche déclarative plutôt que procédurale
  - *On décrit ce que l'on souhaite, pas comment l'obtenir*
  - *Notion de Desired State Configuration*

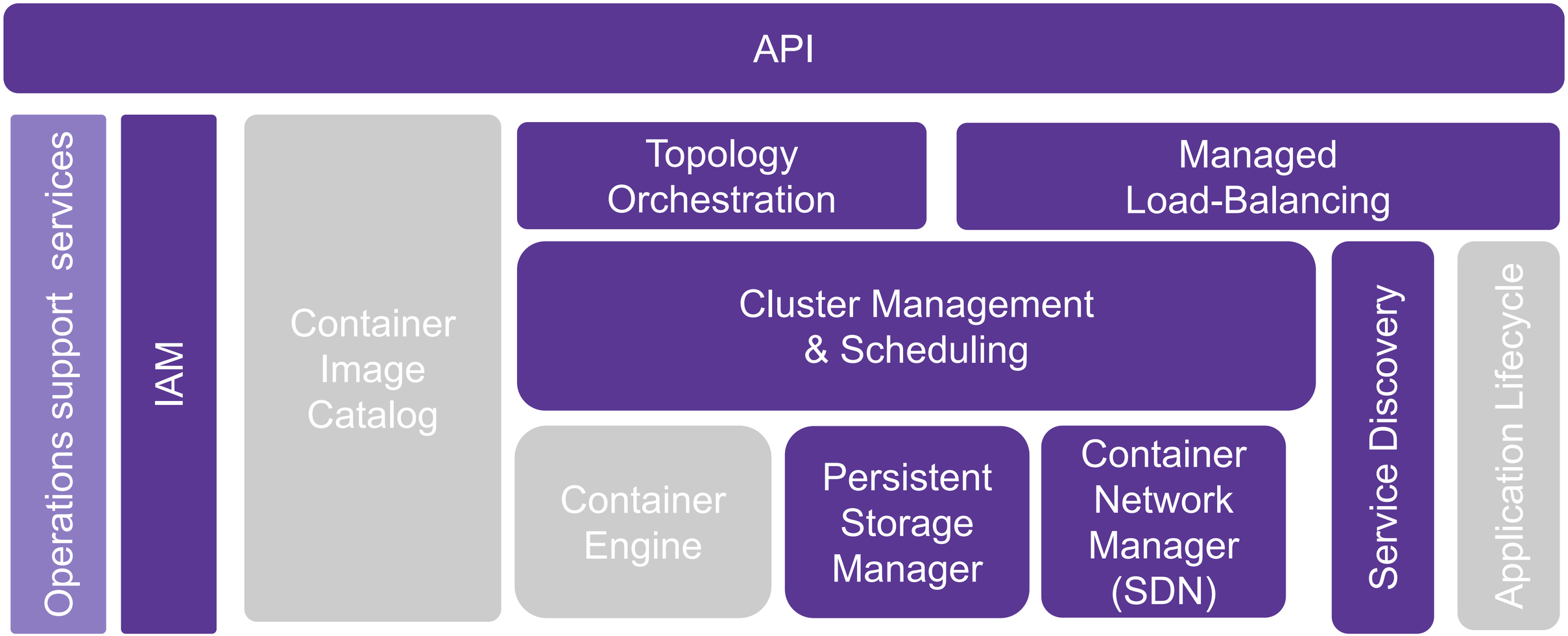
BRIQUES D'UN ORCHESTRATEUR



DOCKER EN 2014/2015



LE CAS DE KUBERNETES



KUBERNETES ET SON ARCHITECTURE

Comme Docker, K8s respecte ces différents principes d’architecture:

- Scalables horizontalement
- Immuables
- Sans état
- Share nothing
- Création et destruction facile, rapide et à faible coût

PETS VS CATTLE

- On leur donne des noms (ex: monchaton)
- Ils sont uniques, on en prend soin
- Quand ils vont mal, on les chouchoute pour qu’ils aillent mieux

- On leur donne des numéros (ex: 3402)
- Elles se ressemblent toutes
- Quand elles sont malades, on les remplace

EXEMPLES DE CAAS

Amazon EC2
Container Service

COMMENT UTILISER KUBERNETES ?

Chez soi

- Directement sur des serveurs physiques (bare metal)
- Sur des VMs traditionnelles (VMWare)
- Sur un cloud privé (OpenStack)
- Sur son laptop (minikube / Docker Desktop)

Sur le cloud (avec des capacités d'intégration avancées : LB, Volumes…)

- Amazon Web Services (VM ou EKS)
- Google Cloud Platform (VM ou GKE)
- Azure (VM ou AKS)
- Chez beaucoup d'autres fournisseurs de clusters Kubernetes managés

Des outils connexes au projet Kubernetes sont là pour aider les déploiements (kops, kubeadm…)

INSTALLATION MINIKUBE AKA KUBERNETES JUNIOR

https://kubernetes.io/fr/docs/tasks/tools/install-minikube/#installer-minikube

Pour vérifier la bonne installation des différents outils, il faut que vous lanciez les commandes suivantes et ayez un résultat similaire à l'image ci-dessous :

- kubectl version --client
- minikube version

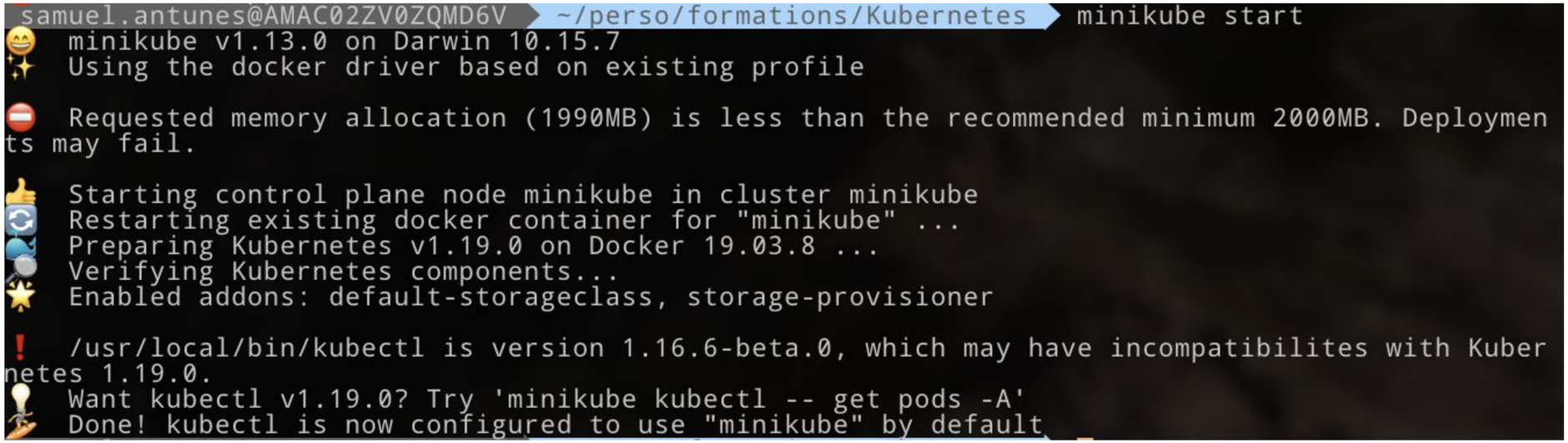
LANCEMENT DE MINIKUBE

On peut à présent démarrer notre minikube avec la commande :

- minikube start

A savoir que la première fois il est demandé de choisir un driver, référez-vous à ce lien pour choisir un driver en fonction de votre environnement. On utilise généralement “docker” par défaut : minikube start --driver=”docker”

https://minikube.sigs.k8s.io/docs/drivers/



COMMENT INTERAGIR AVEC KUBERNETES

Avec l'API
- Interaction programmatique pour manipuler les ressources
- Approche déclarative de l'État Attendu (Desired State)

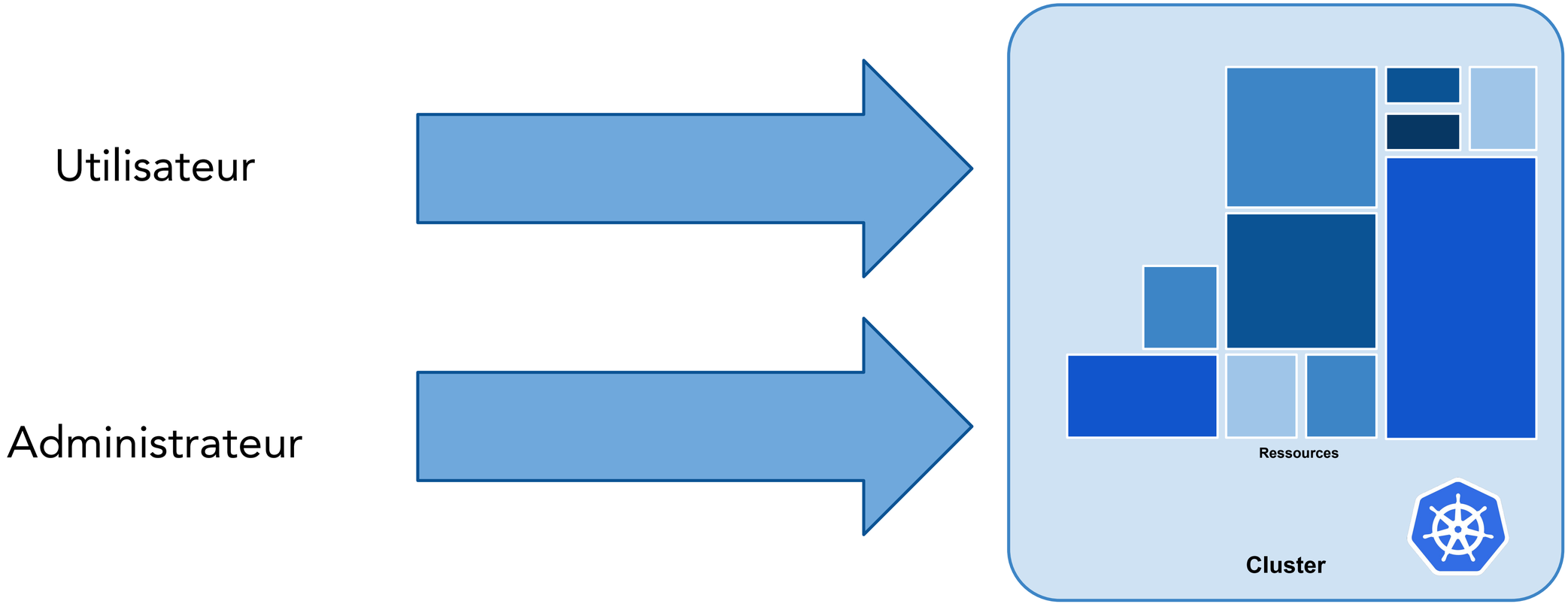
Avec un SDK (Golang, Python…) qui s'appuie sur l'API
- Terraform, Ansible

Avec le CLI
- Binaire kubectl

Avec le(s) Dashboard(s)
- Visualisation simplifiée des ressources présentes dans le cluster
- Pas complet, les ressources récentes de Kubernetes n'apparaissent pas en visualisation

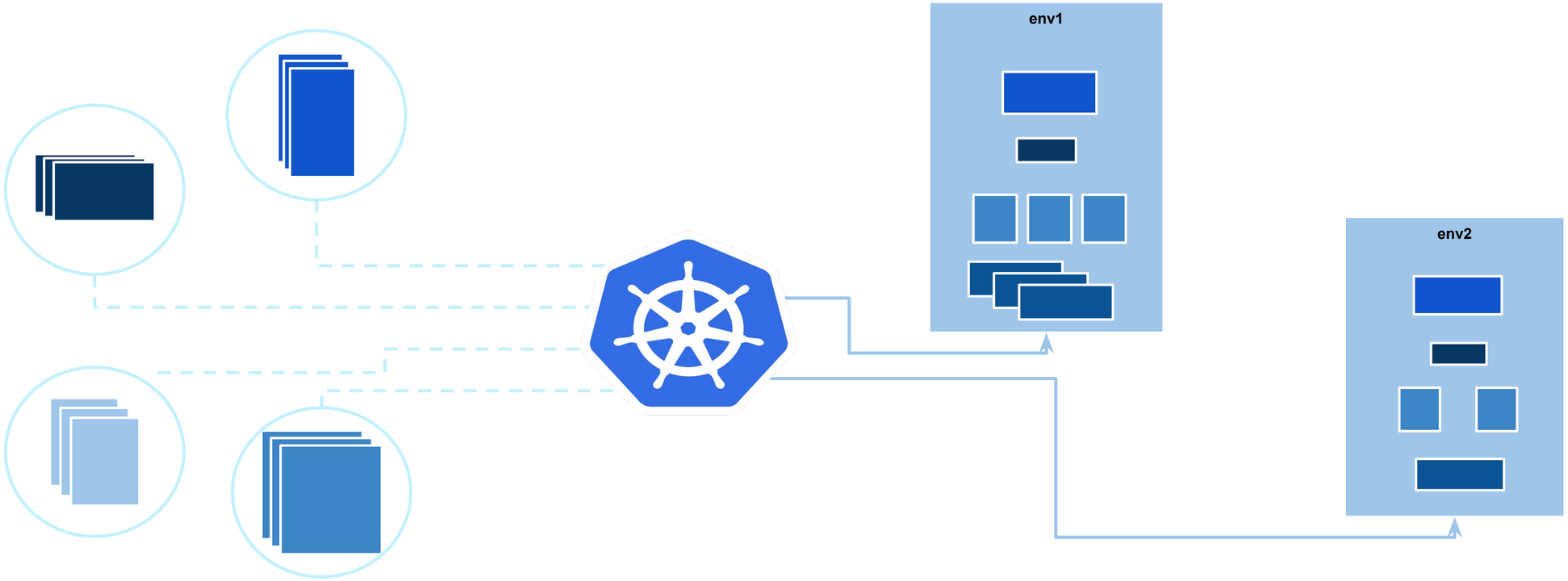
LES CONCEPTS KUBERNETES

Une ressource au sens Kubernetes représente un concept logique manipulé dans Kubernetes. Il en existe plus d'une 20aine et certaines sont réservées aux administrateurs.



LES CONCEPTS KUBERNETES - TOPOLOGIE LOGIQUE

C’est donc avec des concepts logiques que nous allons pouvoir configurer et interagir avec notre cluster Kubernetes (ici minikube).
Concrètement qu’est-ce que ça veut dire ?

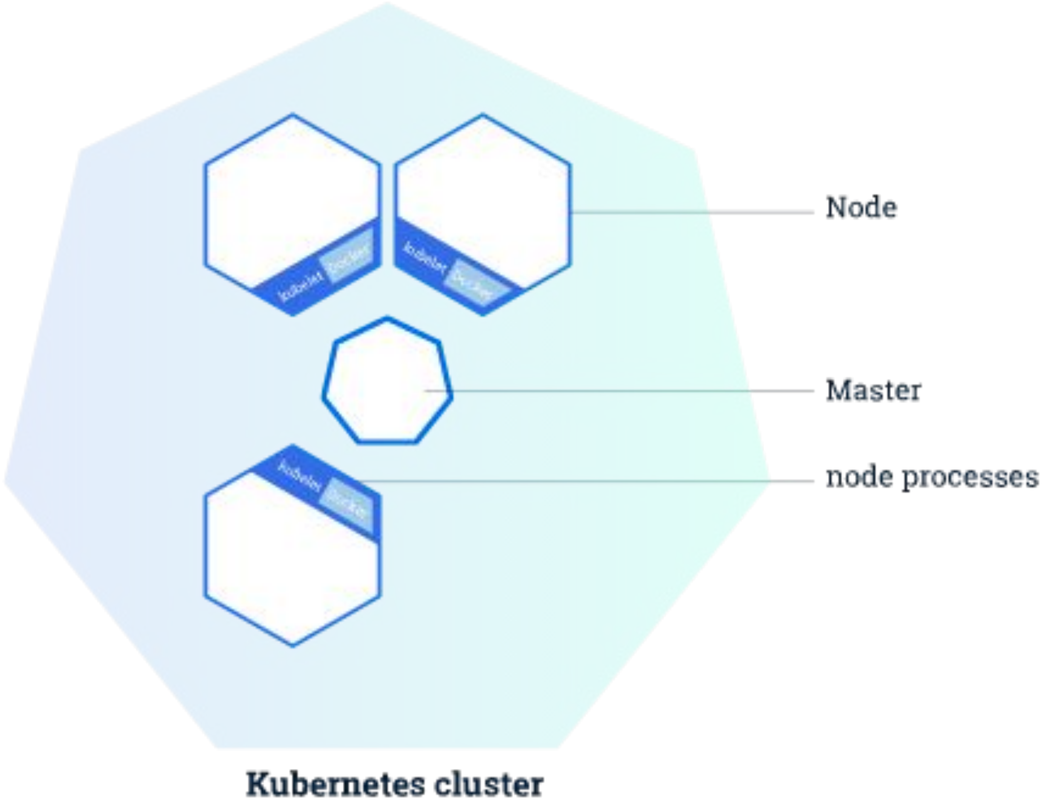


LES CONCEPTS KUBERNETES - LE NODE

La première ressource que nous allons découvrir est le “node”. Il y en a de deux types, un node dit “master” et des nodes “slaves”.

Le master contrôle le cluster à travers des agents posés sur les noeuds slaves : le Kubelet.

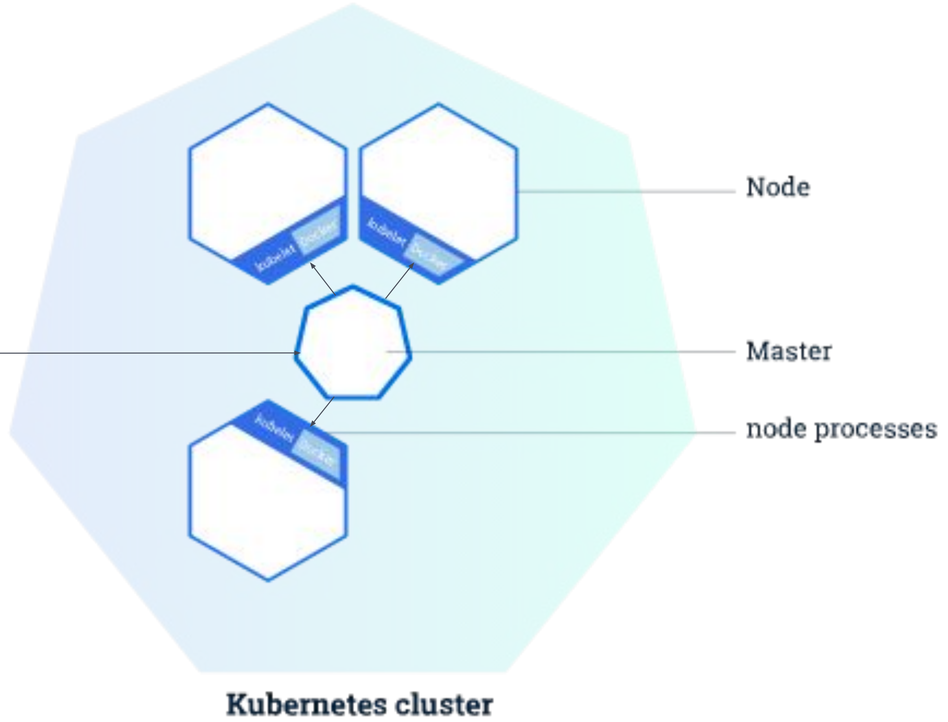
Un node au sens propre du terme peut être une machine virtuelle, physique ou bien on conteneur docker.



LES CONCEPTS KUBERNETES - LE CHEMINEMENT

Utilisateur

kubectl



KUBECTL

C’est un exécutable binaire écrit en Go qui existe pour la plupart des plateformes classiques (Windows, Linux, MacOSX)

Le client kubectl suit la même numérotation de version que la partie serveur

Pour connaître la dernière version stable :

https://storage.googleapis.com/kubernetes-release/release/stable.txt

UTILISATION DE KUBECTL - LES “GETTERS”

Lister toutes les ressources d’un type

```
$ kubectl get (type1|type2|type3|...)
```

Lister une ou des ressources spécifique(s)

```
$ kubectl get type1/nom-ressource1 type2/nom-ressource2
$ kubectl get type3 nom-ressource3
```

Pour avoir plus de détails

```
$ kubectl get type1/nom-ressource -o wide
$ kubectl get type1/nom-ressource -o (yaml|json)
$ kubectl describe type8/nom-ressource
```

UTILISATION DE KUBECTL - FORMAT DE SORTIE AVEC L'OPTION "-o"

```
json
yaml
wide
custom-columns=...
custom-columns-file=...
[go-]template=...
[go-]template-file=...
jsonpath=...
jsonpath-file=...
```

```
$ kubectl get nodes/minikube -o=go-template \
 --template="{{.metadata.name}} est sous {{.status.nodeInfo.operatingSystem}}"
minikube est sous Linux

$ kubectl get nodes \
 -o=custom-columns=NAME:.metadata.name,CPU:.status.allocatable.cpu
NAME       CPU
minikube   2
```

UTILISATION DE KUBECTL - LES “SETTERS”

Le mode automagique

```
$ kubectl run [plein d’options]
```

Pour faire le ménage

```
$ kubectl delete (type1|type2|type3|...) NAME
$ kubectl delete type6/NAME
$ kubectl delete type8 --all
```

KUBERNETES TP1

Vous aurez dans ce repo un fichier TP1.md à lire et réaliser :

git clone https://token_kubernetes:eWNyr6S5QGzz8TayhMB4@gitlab.com/santunes-formations/kubernetes.git