

docker

Formation Docker/Kubernetes



Mr :Walid SAAD - © 2019

Présentation

- Qui suis-je?



Présentation

- Qui êtes vous?



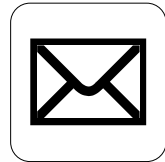
Logistique



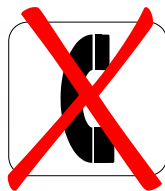
☐ Pause de mi-session



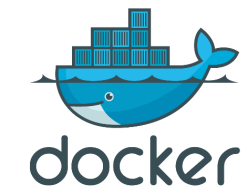
☐ Vos questions sont les bienvenues. N'hésitez pas !



☐ QCM à la fin de la session



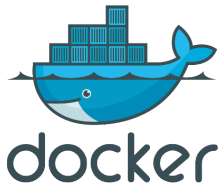
☐ Merci d'éteindre vos téléphones



Partie 1 : Technologie Conteneurs

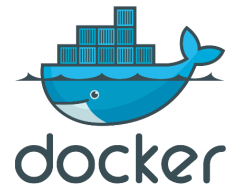
Introduction aux nouvelles architectures




Questions...



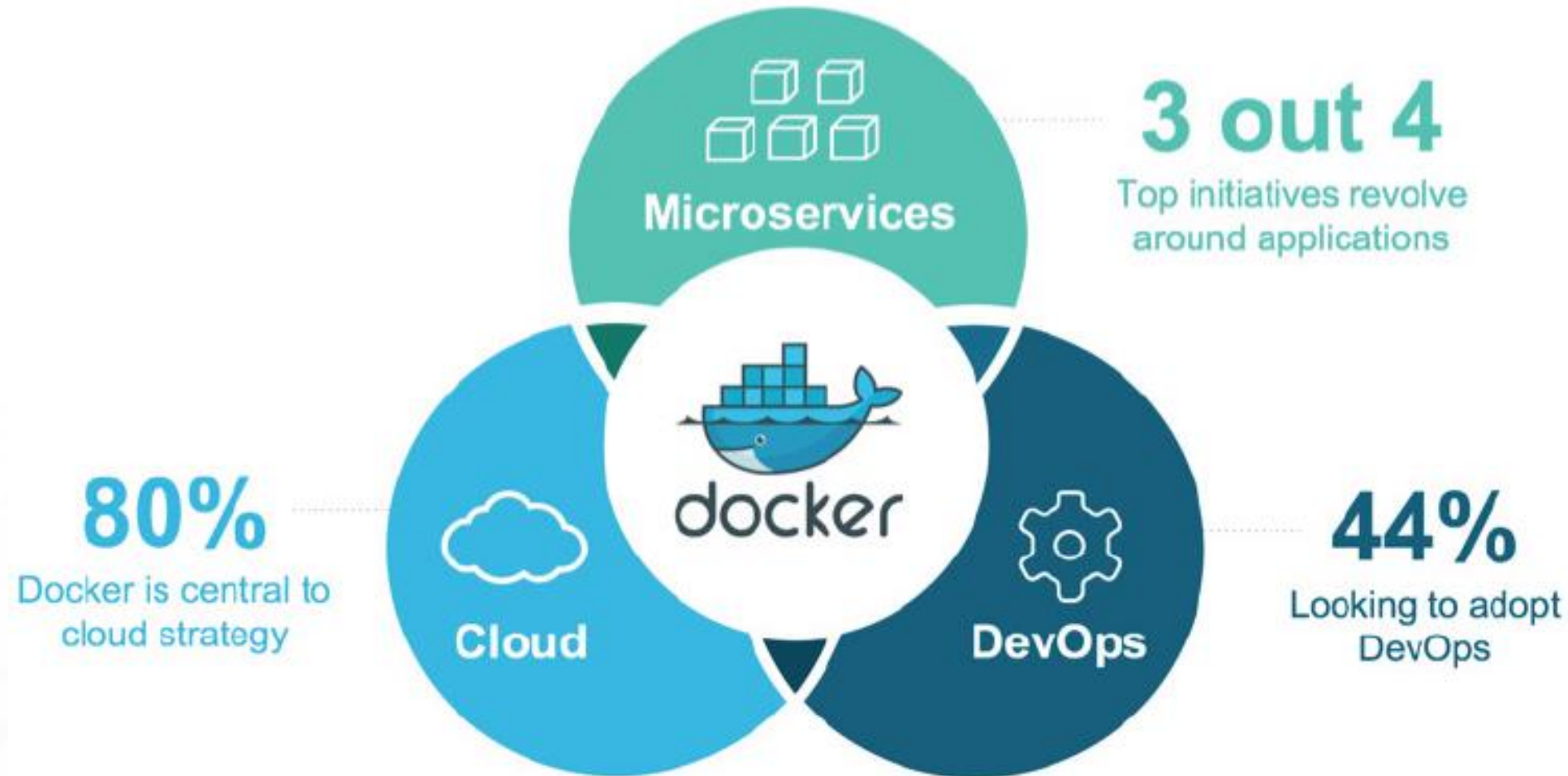
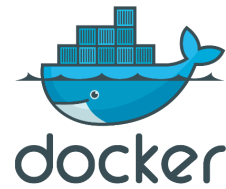
- Conteneurs Linux **LXC** et Conteneurs **Docker** : quelle(s) différence(s) ?
- Pourquoi des **conteneurs** plutôt que des **Machines Virtuelles** ?
- Pourquoi utilise-t-on Docker et quelles sont ses atouts dans le développement et le déploiement des applications cloud-native?
- Conteneurs, sont-ils vraiment populaires dans les Data Center et Cloud Computing?
- Docker, Cloud, DevOps et microservices, quel lien?

Infrastructure IT: Changement de vitesse



Datacenter	Virtualisation	Docker
		
Déploiement dans le mois	⇒ Déploiement dans la minute	⇒ Déploiement dans la seconde
Pendant des années	Pendant des mois	Pendant quelques heures/minutes
Développement en cascade	Agile	DevOps

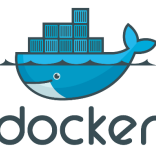
Adoption de la Technologie Conteneur Docker



State of App Development Survey: Q1 2016

https://www.innovate-systems.de/downloads/docker/The_definitive_Guide_to_Docker.pdf

Transition des applications : Monolithique vers Microservices



1

- Conteneuriser les applications monolithiques
- Garantir la portabilité, l'optimisation des ressources et la sécurité

2

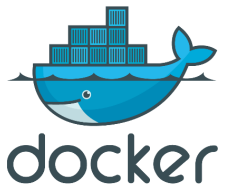
- Conteneuriser les applications monolithiques puis les transformer en microservices (Refactoring)
- Transformation selon l'importance de l'application

3

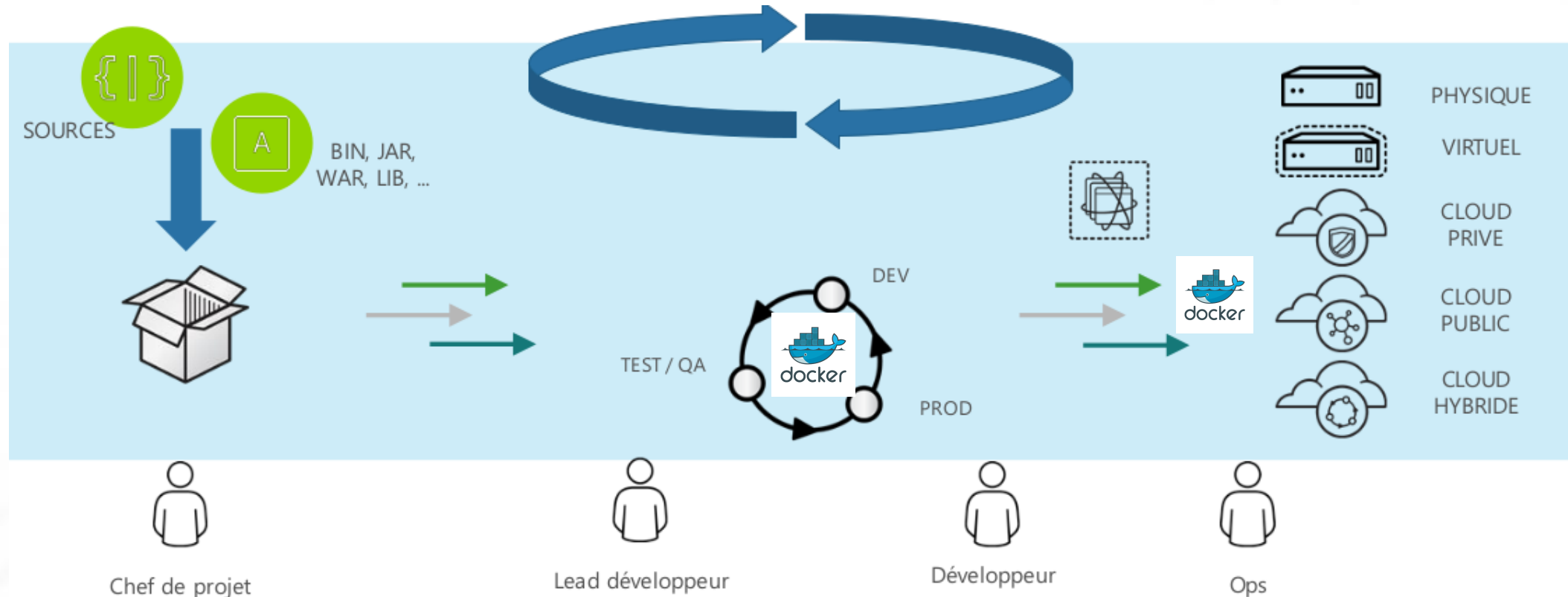
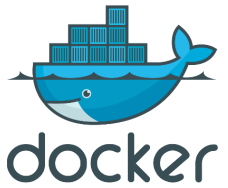
- Créer des nouvelles applications microservices (from scratch)
- Conteneuriser tous les microservices distribués et les déployer en une seule application en production.



La Technologie Conteneur révolutionne le Cloud



Docker = La voie vers DevOps

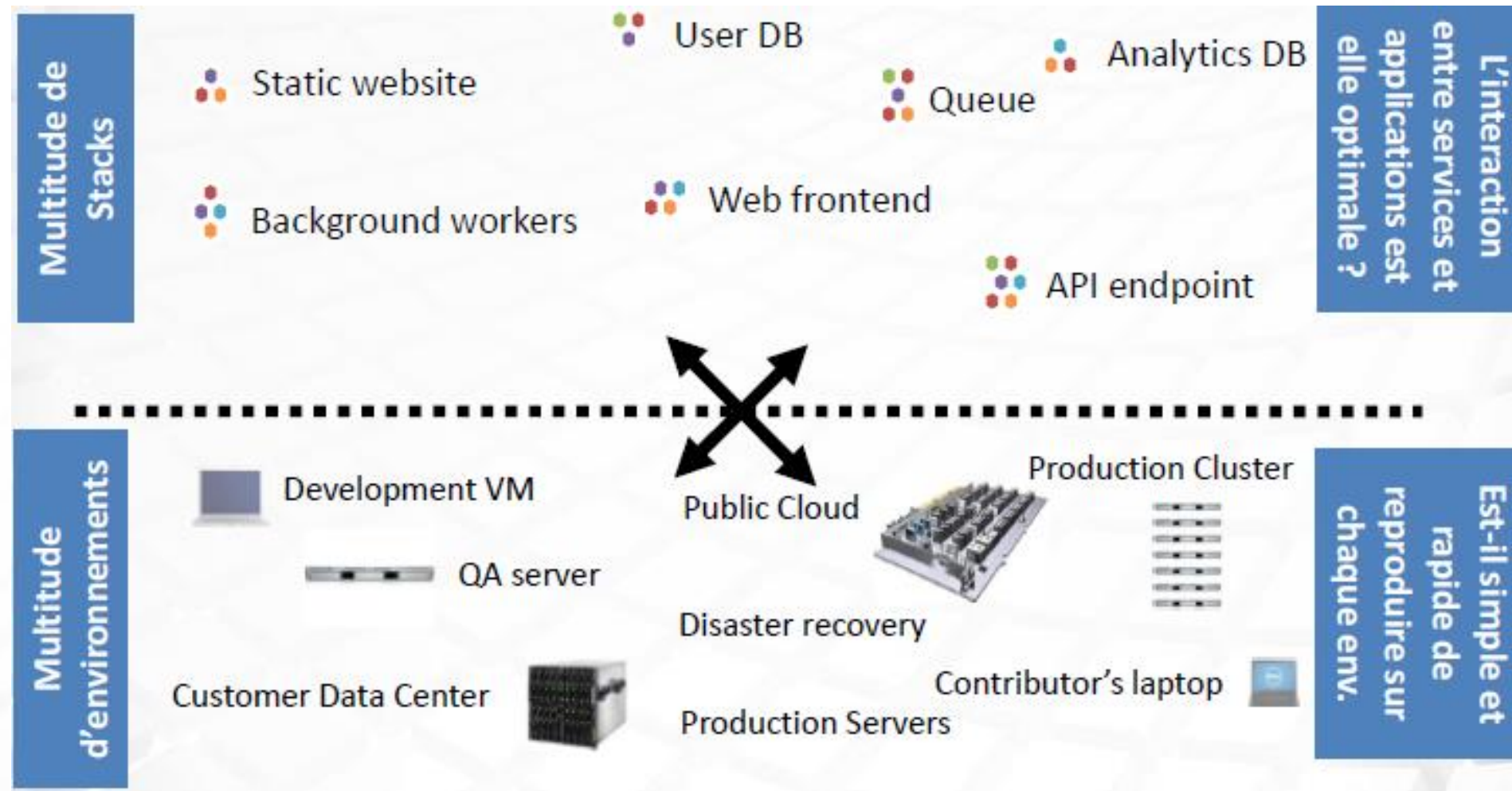
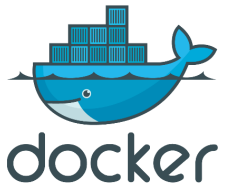


DevOps = Intégrations Continue et Livraison Continue

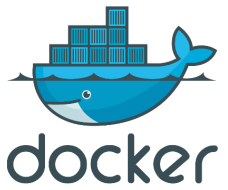
Partie 1 : Technologie Conteneurs







Le challenge...

Services multiples ET Environnement multiples = Comment déployer?



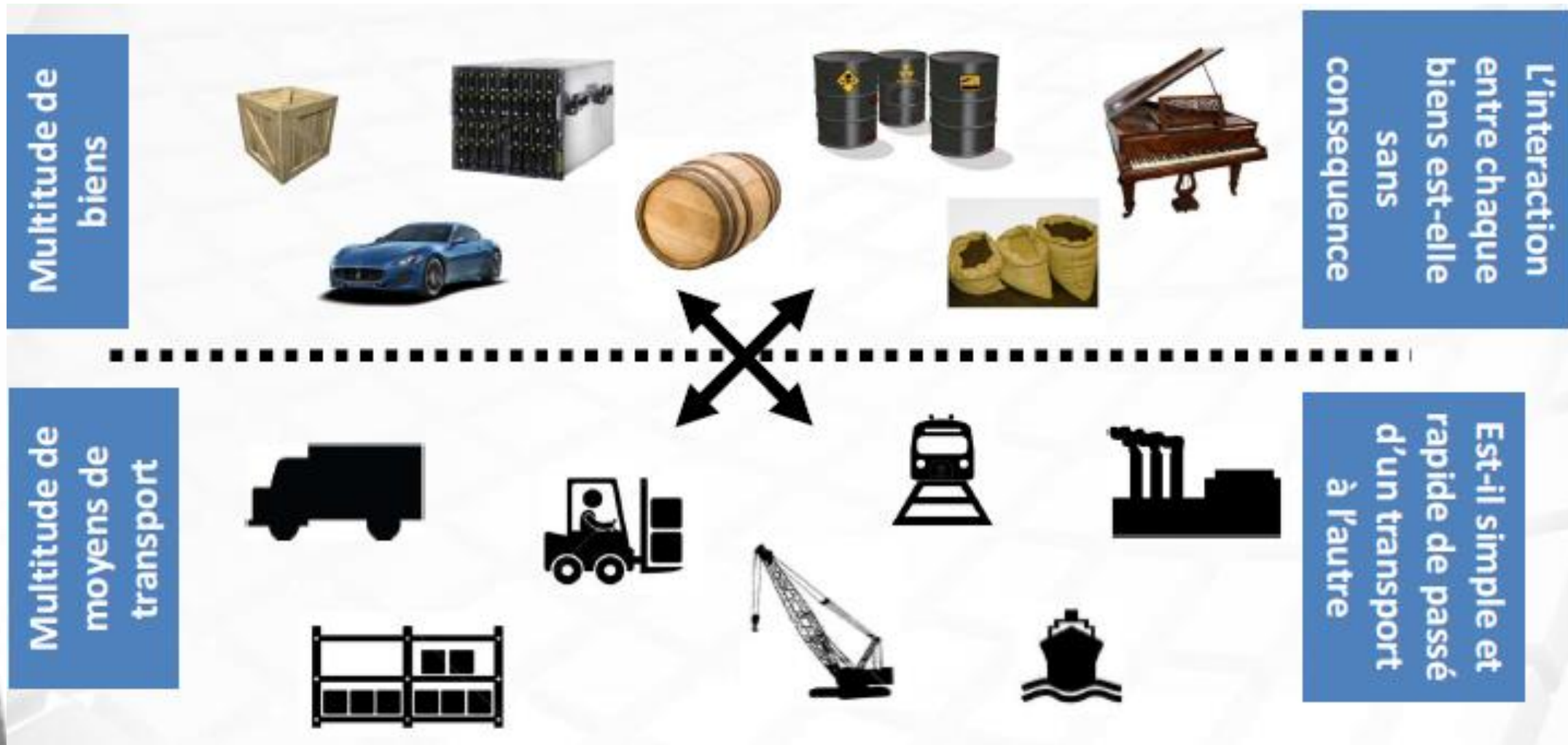
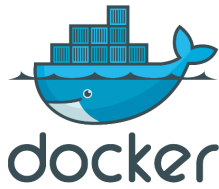
Matrice de déploiement (Comment résoudre les problèmes de compatibilité?)



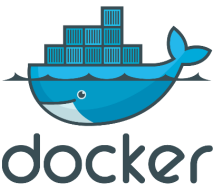
	Static website	?	?	?	?	?	?	?
	Web frontend	?	?	?	?	?	?	?
	Background workers	?	?	?	?	?	?	?
	User DB	?	?	?	?	?	?	?
	Analytics DB	?	?	?	?	?	?	?
	Queue	?	?	?	?	?	?	?
		Development VM	QA Server	Single Prod Server	Onsite Cluster	Public Cloud	Contributor's laptop	Customer Servers



Produits multiples ET Moyens multiples = Comment transporter?



Matrice de transport (Quelle compatibilité?)

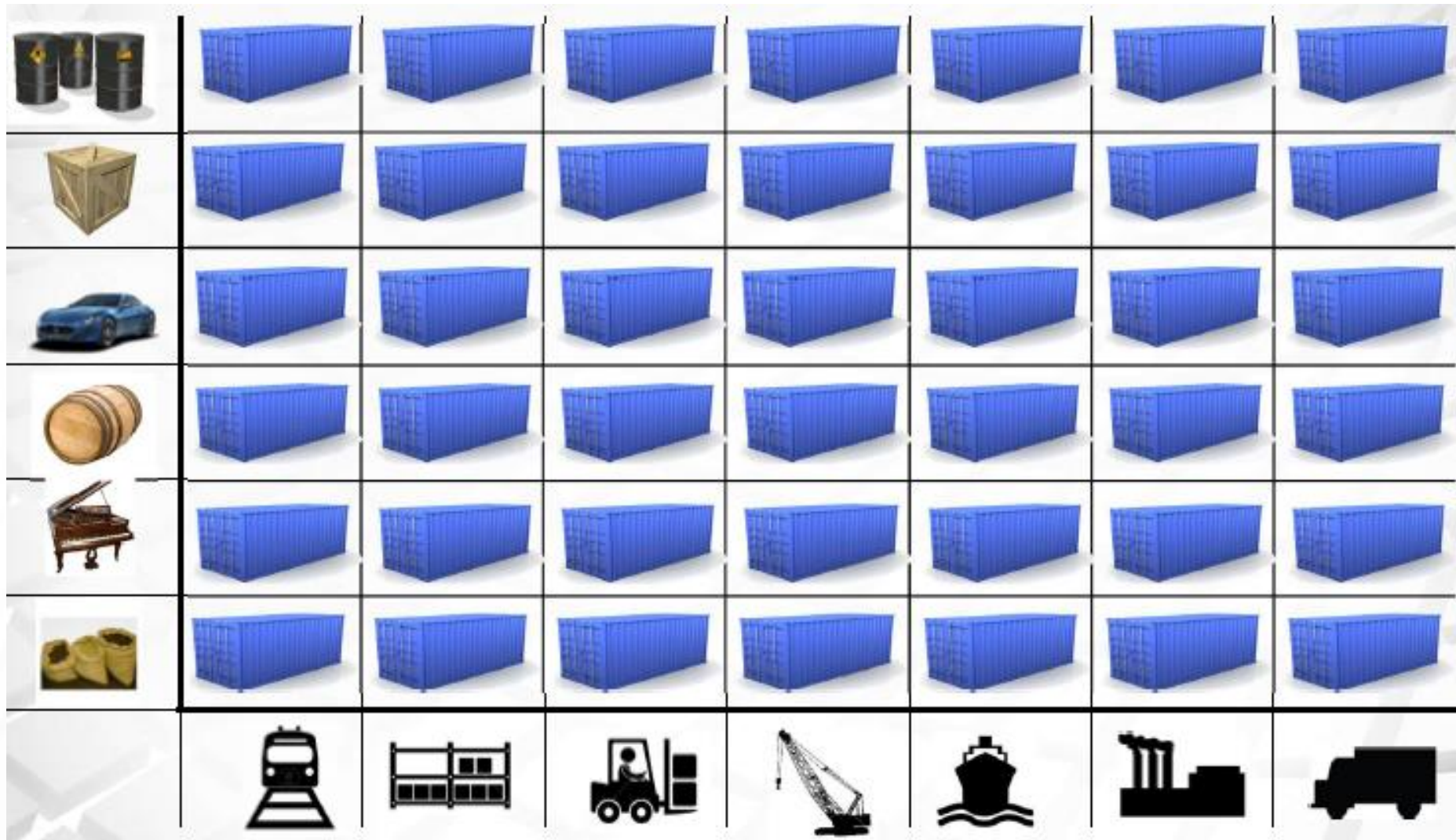
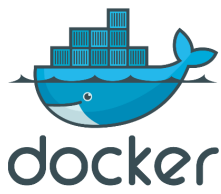


	?	?	?	?	?	?	?
	?	?	?	?	?	?	?
	?	?	?	?	?	?	?
	?	?	?	?	?	?	?
	?	?	?	?	?	?	?
	?	?	?	?	?	?	?
							

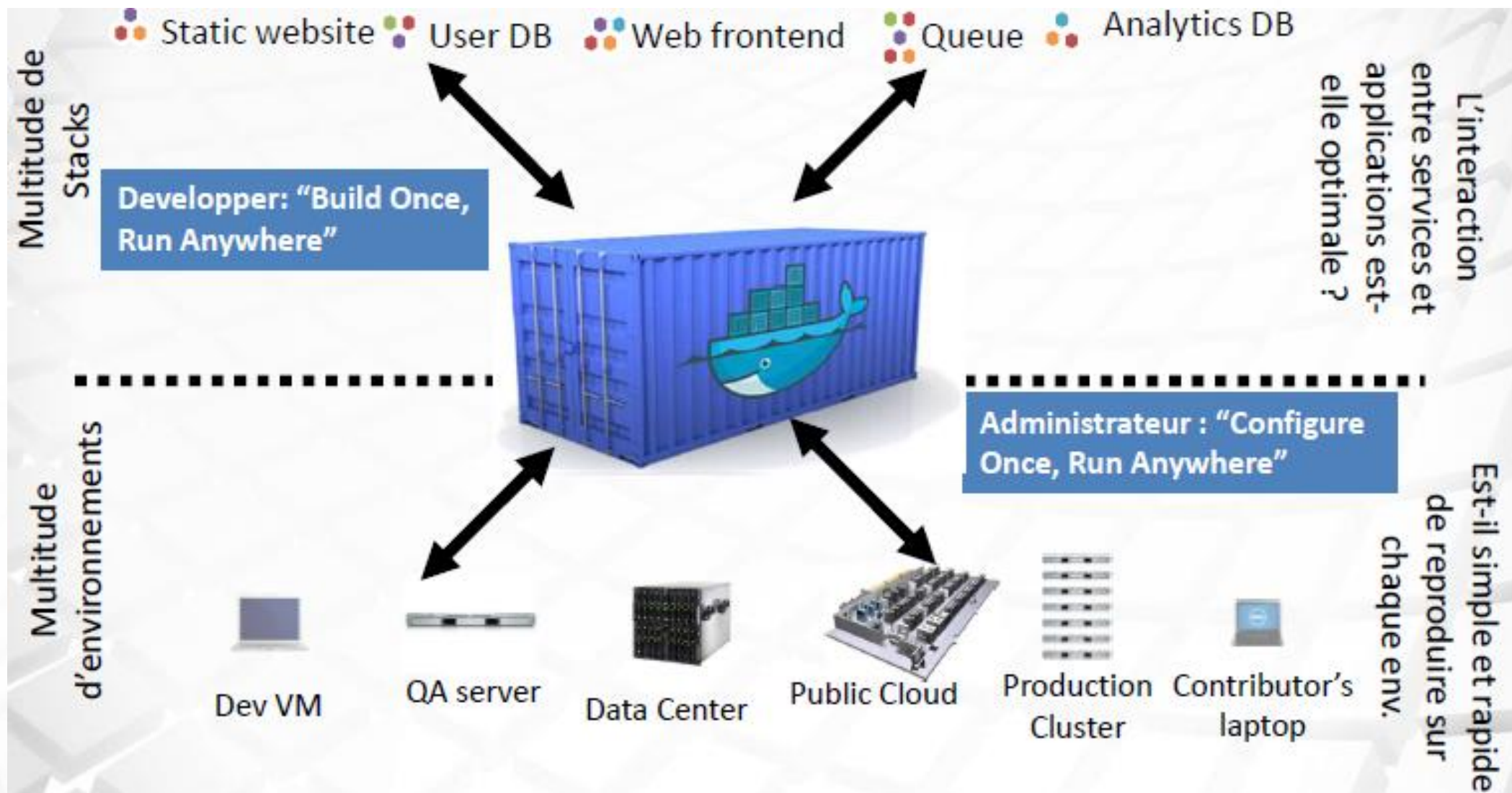
Usage des conteneurs = Portabilité



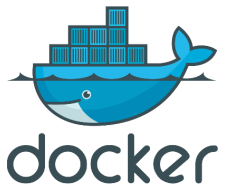
Standardisation














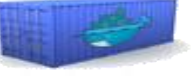











































Usage des conteneurs = Portabilité

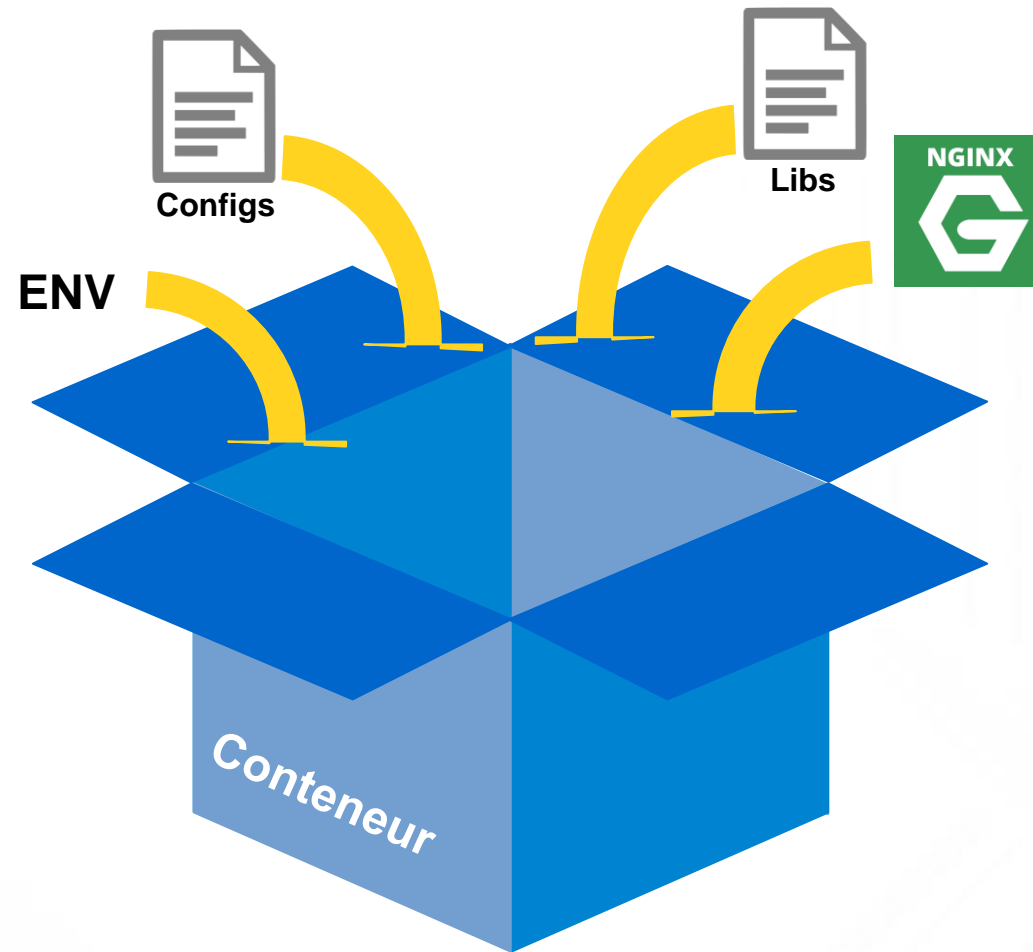
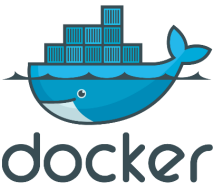


Matrice de déploiement (Résolution)



	Static website							
	Web frontend							
	Background workers							
	User DB							
	Analytics DB							
	Queue							
		Development VM	QA Server	Single Prod Server	Onsite Cluster	Public Cloud	Contributor's laptop	Customer Servers
								

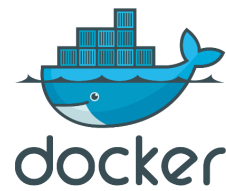
Conteneur Docker



Partie 1 : Technologie Conteneurs

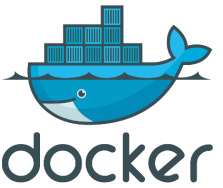
Conteneur linux lxc...

Petit rappel sur les conteneurs LXC



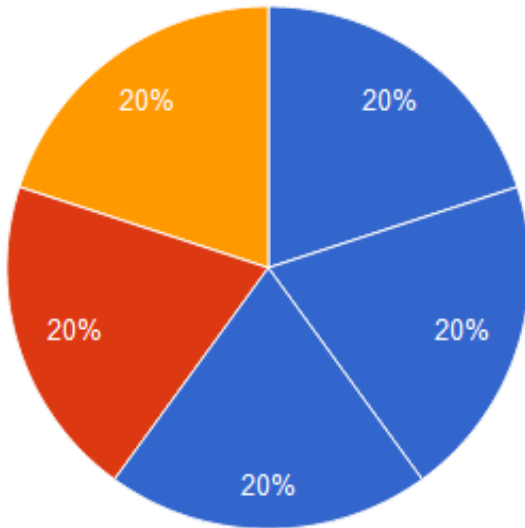
- LXC est utilisé pour faire fonctionner **plusieurs instances Linux isolés** les uns des autres dans des conteneurs **partageant le même noyau** et une petite partie du système hôte
- LXC repose sur la notion de groupes de contrôle Linux (**cgroups**) disponibles depuis sa version 2.6.24
- Chaque groupe de contrôle offre aux applications une **isolation totale des ressources** et d'environnement d'exécution (processeur, mémoire, réseau, système de fichier et accès E/S), et ce sans recourir à des machines virtuelles à part entière
- LXC repose aussi sur une isolation des espaces de nommage du noyau (**namespace**), permettant d'éviter à un système de **connaître les ressources** utilisées par le système hôte ou un autre conteneur (systèmes de fichiers, les ID réseau et les ID utilisateur)

Linux Cgroups

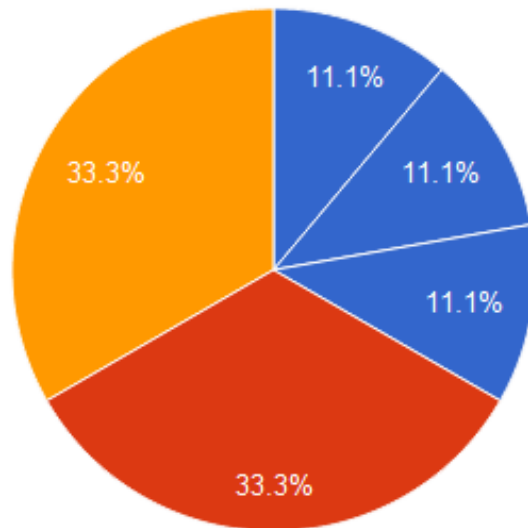


❑ Une isolation totale des ressources et d'environnement d'exécution

CPU usage per process without cgroups



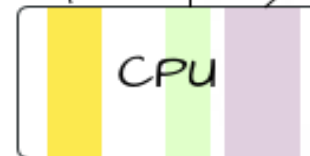
CPU usage per process with cgroups



- User A - process 1
- User A - process 2
- User A - process 3
- User B - process 4
- User C - process 5

SHARES:

1024 640 2048



CGROUP #1

Gets half as much CPU time as cgroup #3.

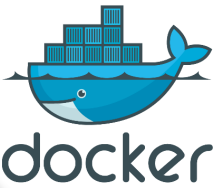
CGROUP #2

Gets the least CPU time.

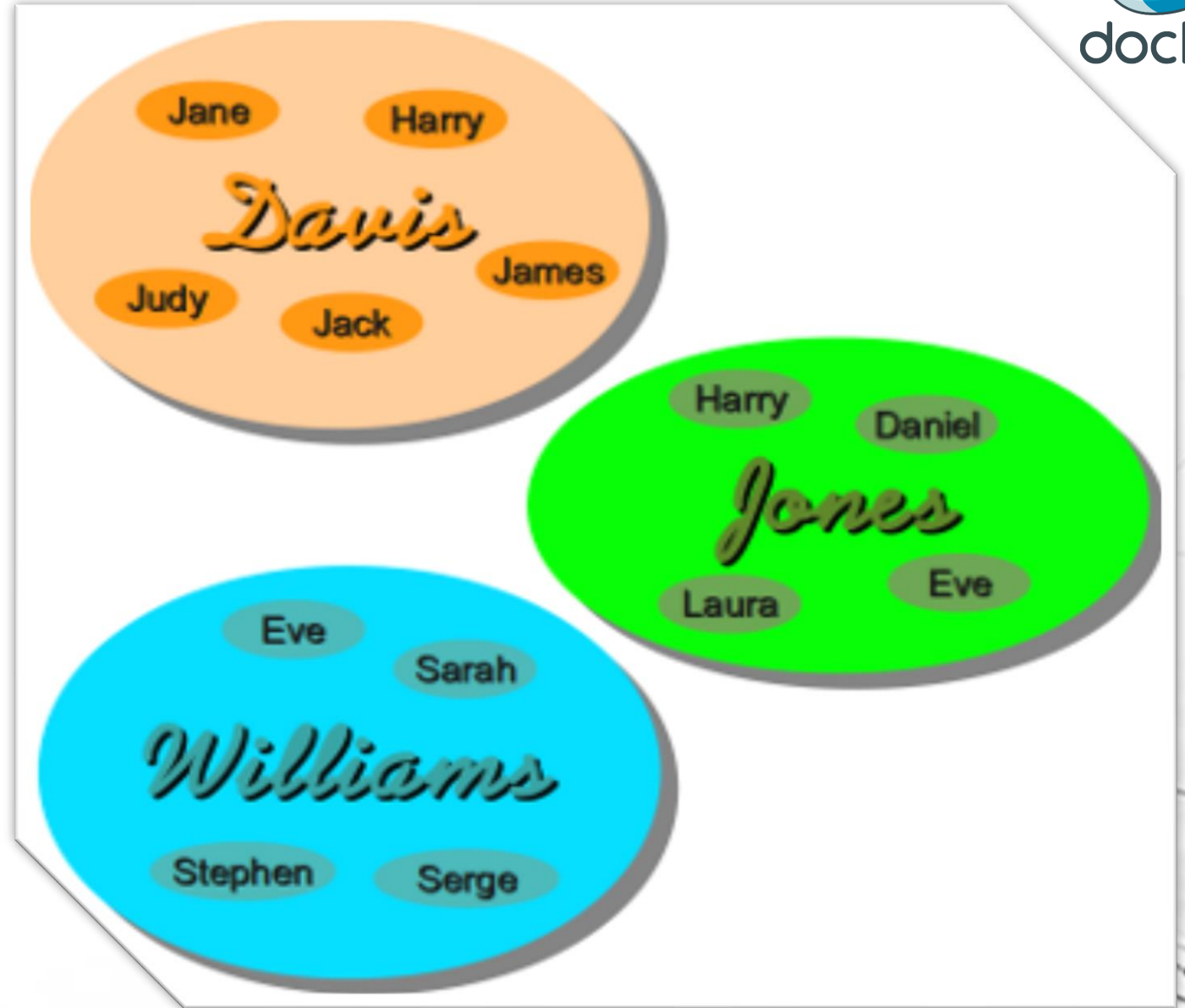
CGROUP #3

Gets the most CPU time.

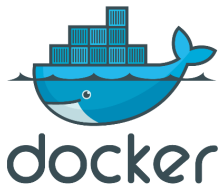
Linux Kernel Namespaces



- ❑ Une isolation des espaces de nommage du noyau
 - Mounts (CLONE_NEWNS)
 - UTS (CLONE_NEWUTS) host et domain
 - IPC (CLONE_NEWIPC)
 - PID (CLONE_NEWPID)
 - Networks (CLONE_NEWNET)
 - User

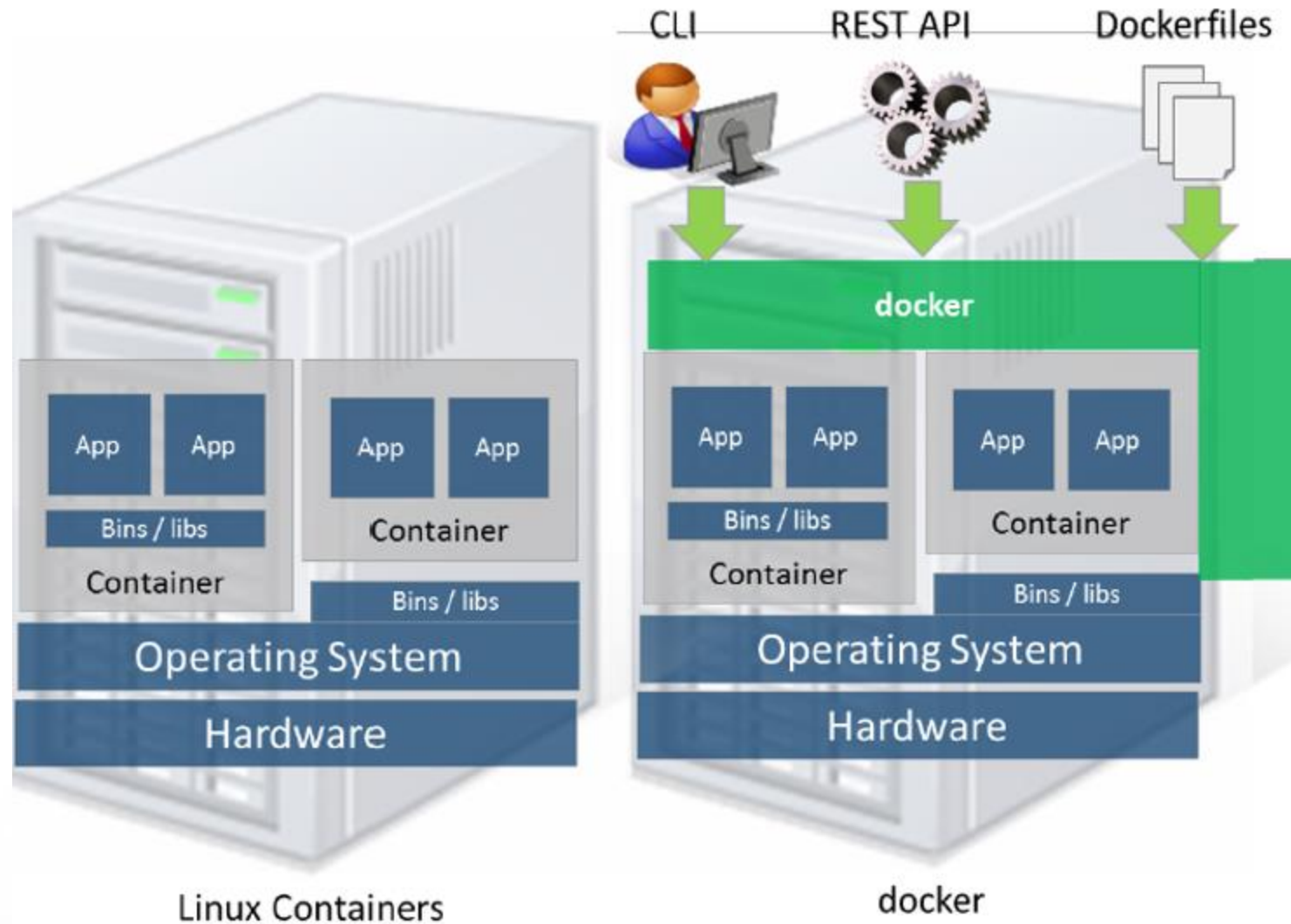
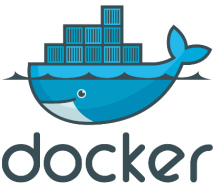


Docker : un LXC augmenté

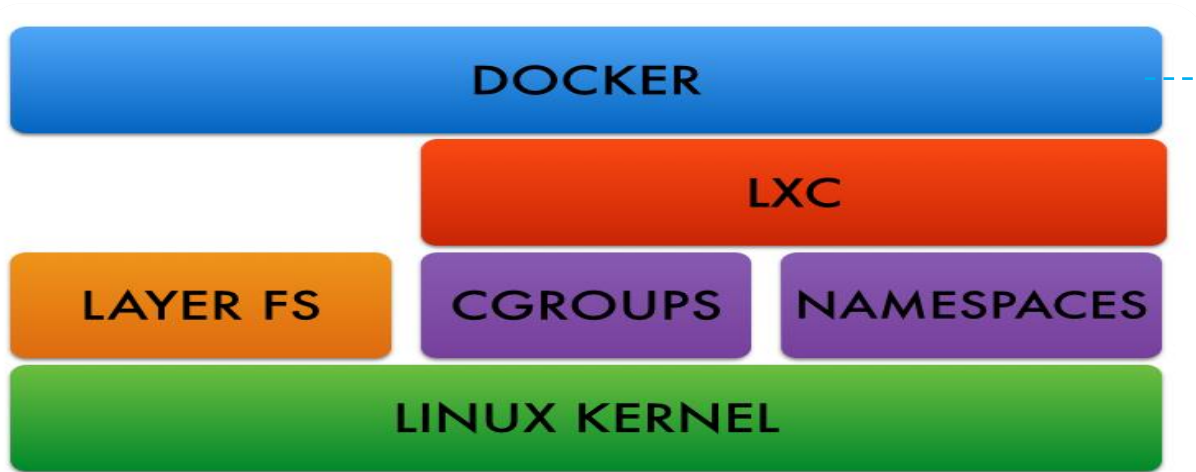
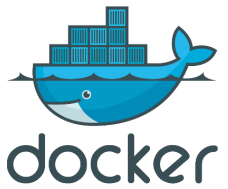


- Docker ne remplace pas les conteneurs Linux LXC (lancé en 2013)
- L'idée consiste à utiliser LXC comme base, puis à ajouter des capacités de niveau supérieur
- Docker autorise la **portabilité** entre machines (qui exécutent aussi Docker) et permet ainsi à une application et à ses composants d'exister en tant qu'objet mobile unique.
- Avec LXC, déplacer une application sur une autre machine peut introduire des différences susceptibles d'empêcher le conteneur de l'application de s'exécuter.

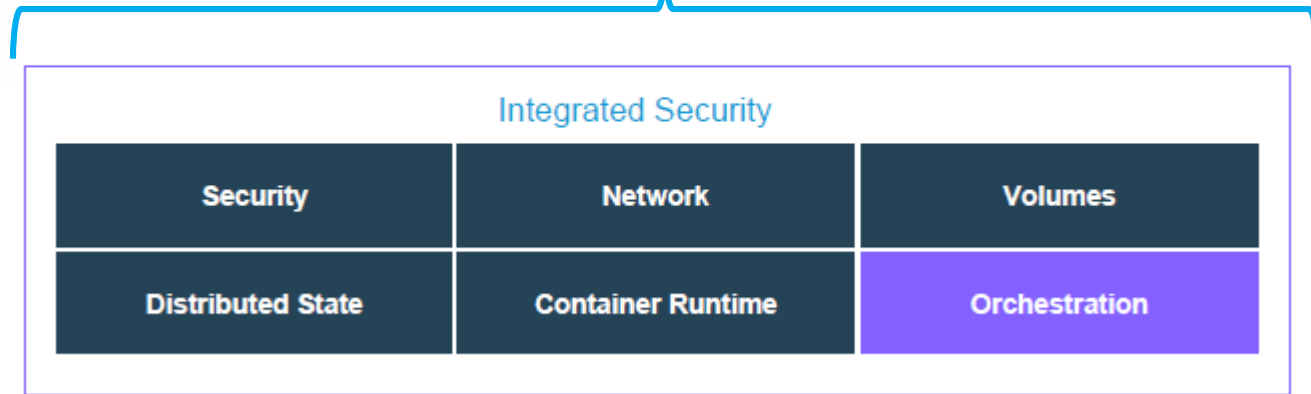
Docker : un LXC augmenté



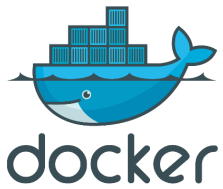
Architecture Docker (Docker Engine)



Docker Engine



Editions Docker : CE et EE



Open source **framework** for assembling core components that make a container platform

Intended for:
Open source contributors + ecosystem developers



Subscription-based, commercially supported **products** for delivering a secure software supply chain

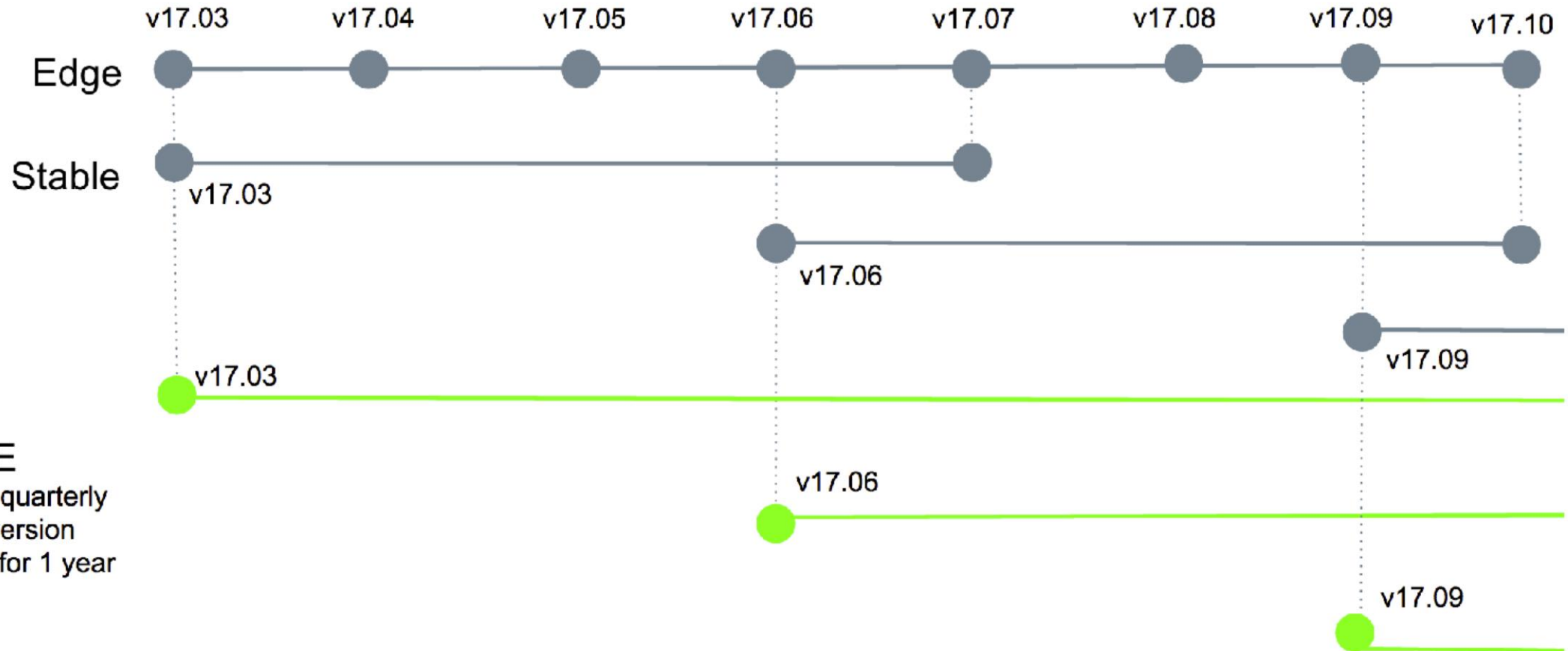
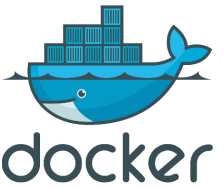
Intended for:
Production deployments + Enterprise customers



Free, community-supported **product** for delivering a container solution

Intended for:
Software dev & test

Versions Docker



Docker CE

Docker EE

EE

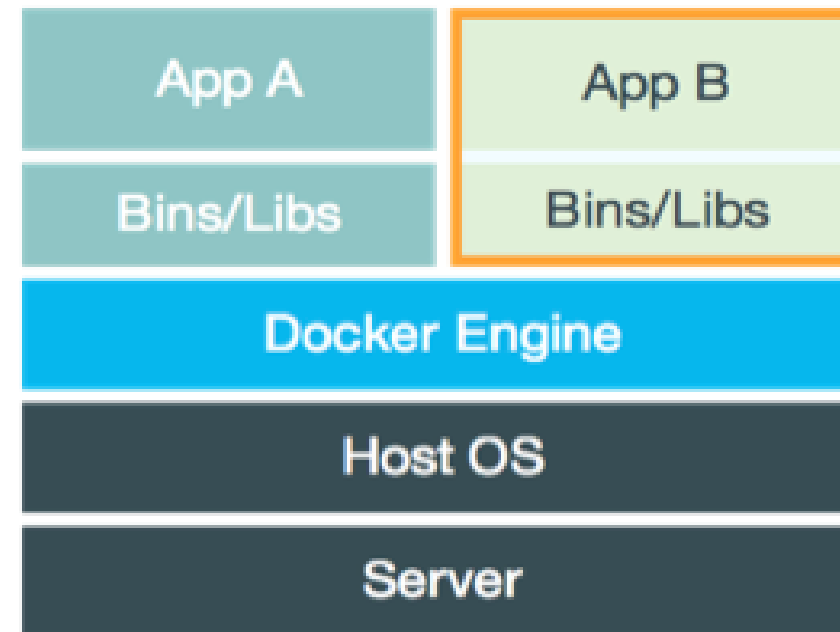
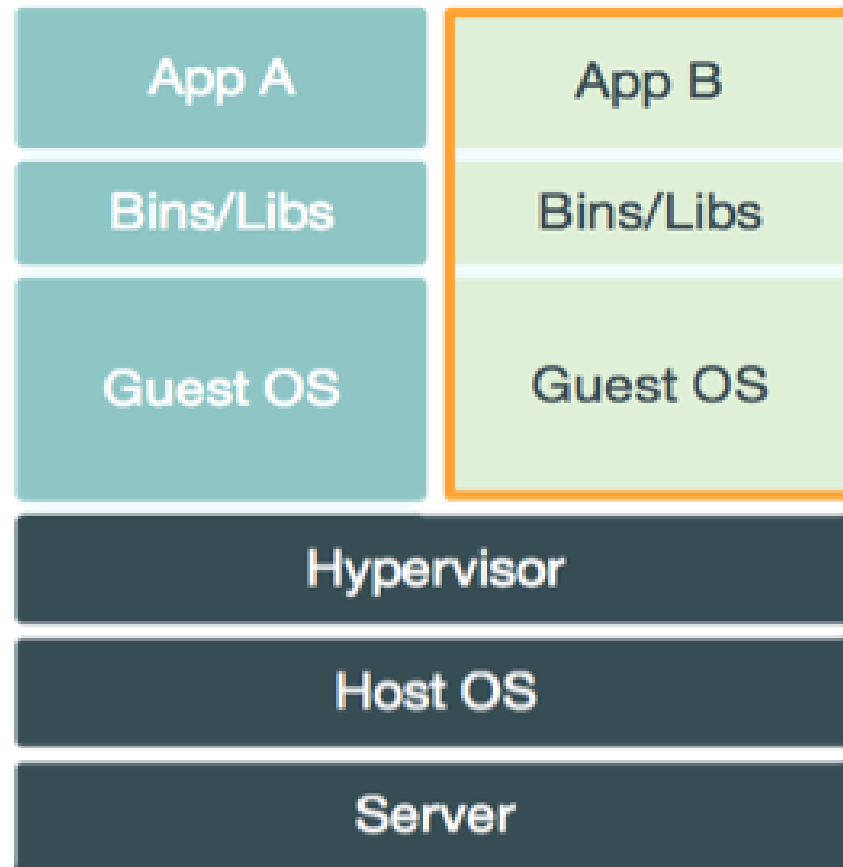
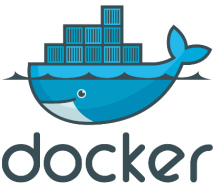
Released quarterly
Each version
supported for 1 year

Version courante v18,03

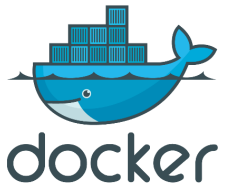
Partie 1 : Technologie Conteneurs

Conteneurs vs machine virtuelle...

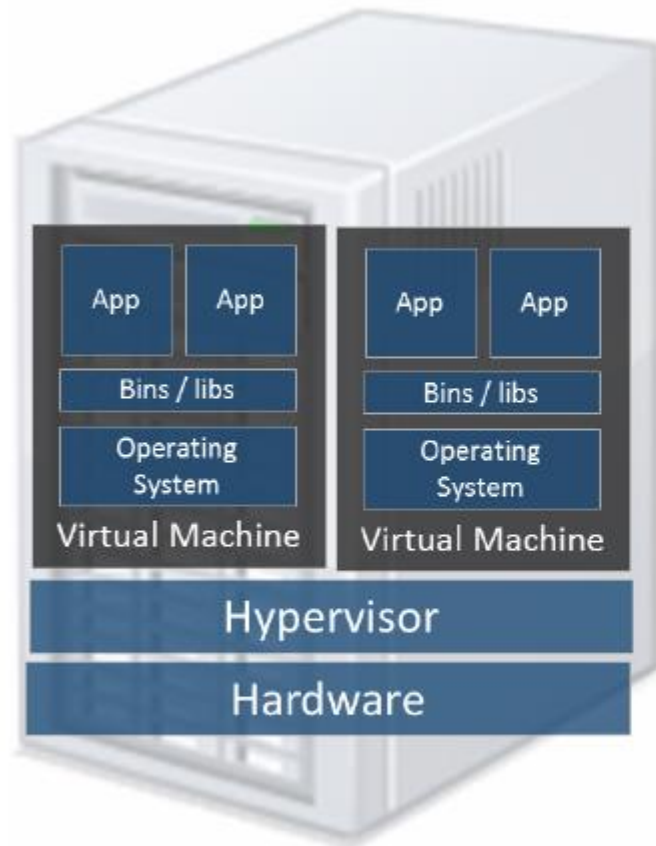
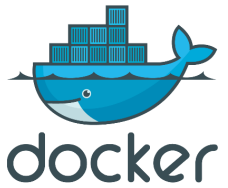
Hyperviseur vs Docker



Hyperviseur vs Docker (Analogie)



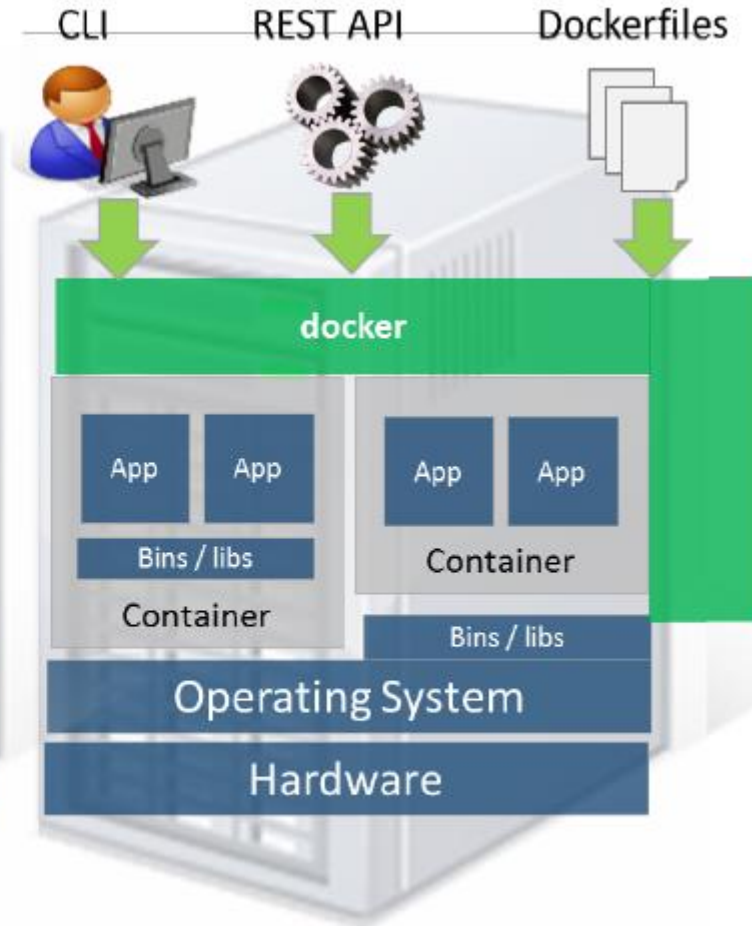
Hyperviseur vs LXC vs Docker



Type 1 Hypervisor

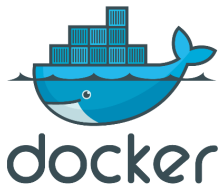


Linux Containers



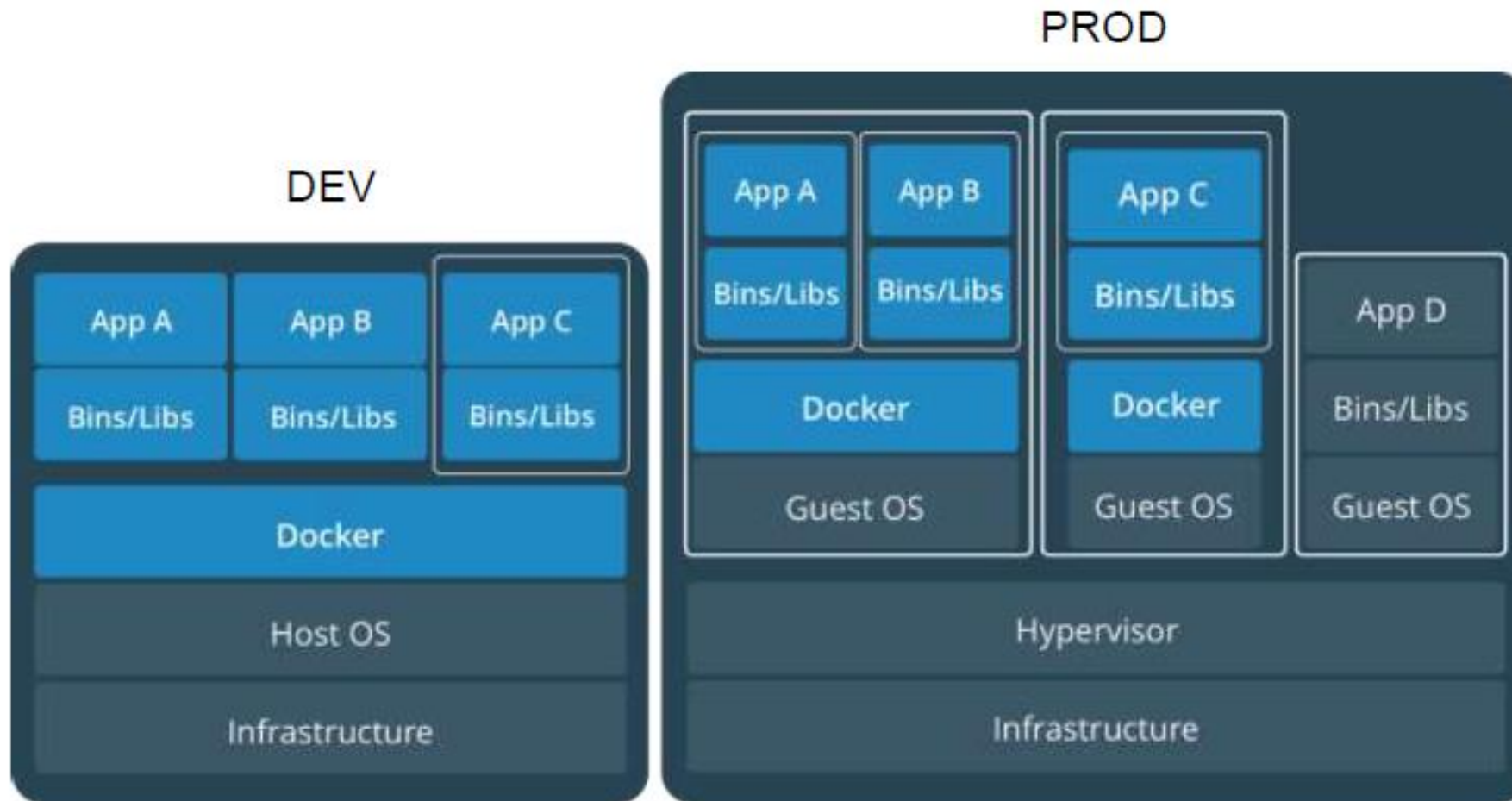
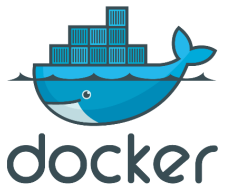
docker

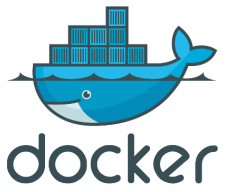
Pourquoi des conteneurs plutôt que des machines virtuelles ?



- Les conteneurs Docker **révolutionnent** la virtualisation car l'idée n'est pas nouvelle
- La virtualisation a un coût car les OS invités requièrent chacun des ressources (mémoire, CPU, réseaux)
- Augmentation de la taille de chaque machine virtuelle, ce qui limite le nombre de machines virtuelles qu'un serveur peut héberger
- La conteneurisation vise à virtualiser les applications **sans trop alourdir** le système
- Favoriser l'intégration rapide d'applications dans des conteneurs.

Des conteneurs dans des VMs ?

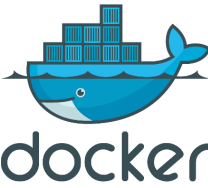




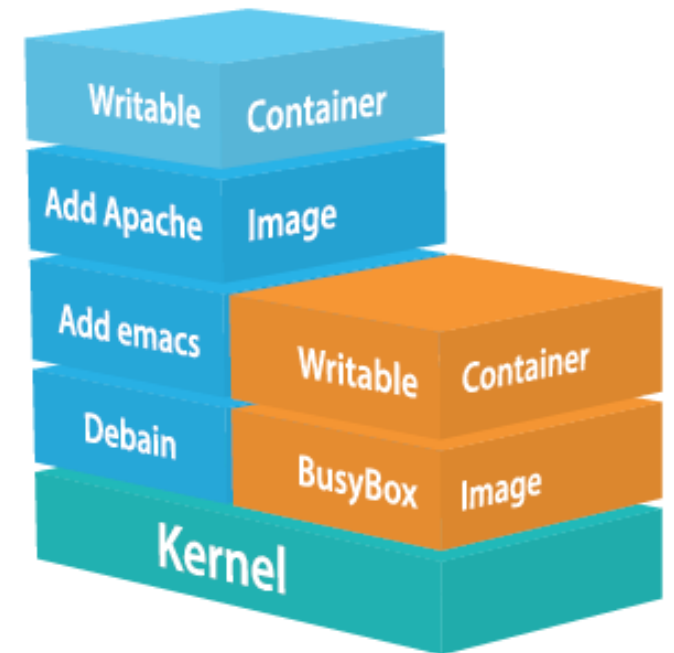
Partie 1 : Technologie Conteneurs

Outils Docker...

Terminologie Docker

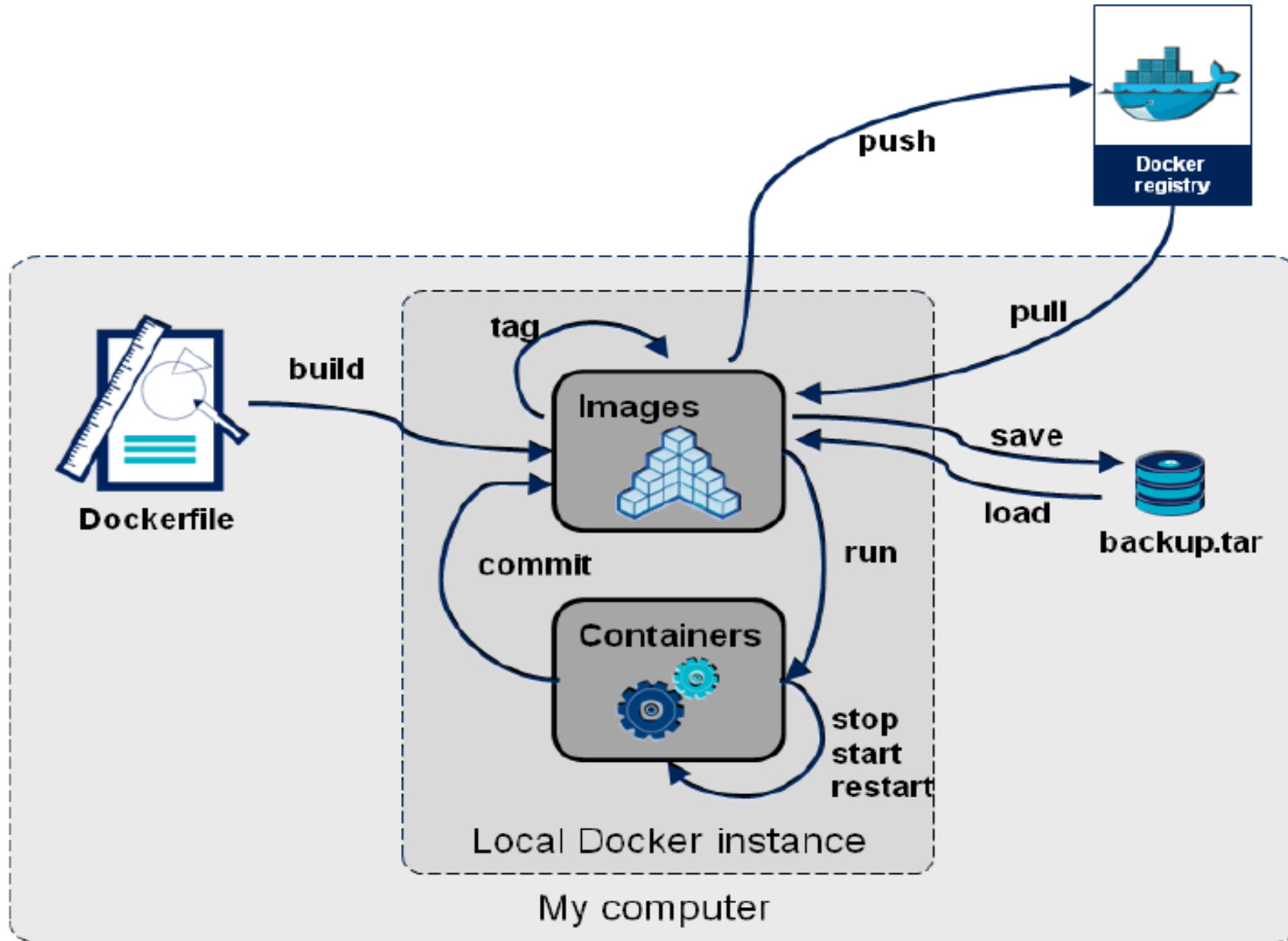
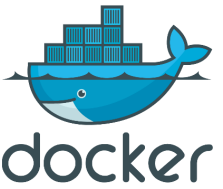


- ❑ Architecture Client / Serveur: Serveur = moteur ou daemon, Client = docker CLI
 - Le client Docker communique avec le Docker daemon pour construire et gérer les conteneurs
 - Le client Docker et le daemon Docker peuvent tourner sur la même machine, comme sur des machines différentes
- ❑ Image
 - Conteneur en lecture seule (couches ou instantané)
- ❑ Conteneur
 - Élément manipulable et accessible en écriture
- ❑ Analogie avec le développement objet :
 - Image = classe
 - Couches = Héritage
 - Conteneurs = Instance

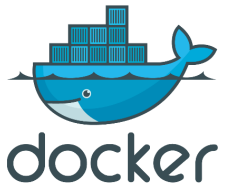


Outils	Description
DOCKER ENGINE	C'est le moteur (daemon Docker) qui gère les conteneurs (cycle de vie, réseaux, volumes, sécurité, etc.)
DOCKER IMAGE	Une image est un template prêt à l'emploi avec des instructions pour la création de conteneur (ou bien un instantané d'un conteneur). Une image Peut être construit à partir d'un Dockerfile ou bien d'une image existante ou être récupéré depuis un Registre Docker.
DOCKER REGISTRY	Service de stockage et de gestion des images locales ou bien hébergées (Docker Hub et Docker Store)
DOCKER MACHINE	Permet de créer automatiquement un environnement pour lancer Docker (une machine Docker)
DOCKER COMPOSE	Permet de lancer des applications multi-containers (sur la même machine)
DOCKER SWARM	Permet de gérer les containers Docker dans un cluster

Cycle de vie Docker

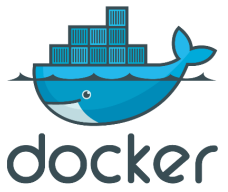


Dockerfile (Exemple)



- FROM ubuntu:16.04
- RUN apt-get update && apt-get install -y openssh-server git apache2 python vim
- RUN mkdir /var/run/sshd
- RUN echo 'root:root' | chpasswd
- RUN sed -i 's/PermitRootLogin prohibit-password/PermitRootLogin yes/' /etc/ssh/sshd_config
- RUN git clone https://github.com/walidsaad/training-app-front /var/www/html/MyApp
- ADD MyApp /var/www/html/MyApp
- WORKDIR /var/www/html/MyApp
- EXPOSE 22 80

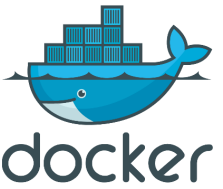
attach	Attach to a running container	pause	Pause all processes within a container
build	Build an image from a Dockerfile	ps	List containers
commit	Create new image from container's changes	pull	Pull image or repo from docker registry
cp	Copy files from containers fs to host	push	Push image or repo to docker registry
diff	Inspect changes on a container's fs	restart	Restart a running container
events	Get real time events from the server	rm	Remove one or more containers
export	Stream contents of container as tar	rmi	Remove one or more images
history	Show the history of an image	run	Run a command in a new container
images	List images	save	Save an image to a tar archive
import	Create new fs image from a tarball	search	Search for an image in the docker index
info	Display system-wide information	start	Start a stopped container
inspect	Return low-level info on a container	stop	Stop a running container
kill	Kill a running container	tag	Tag an image into a repository
load	Load an image from a tar archive	top	Lookup running processes of a container
login	Login to the docker registry server	unpause	Unpause a paused container
logs	Fetch the logs of a container	version	Show the docker version information
port	Lookup public-facing port	wait	Block and print exit code upon cont exit



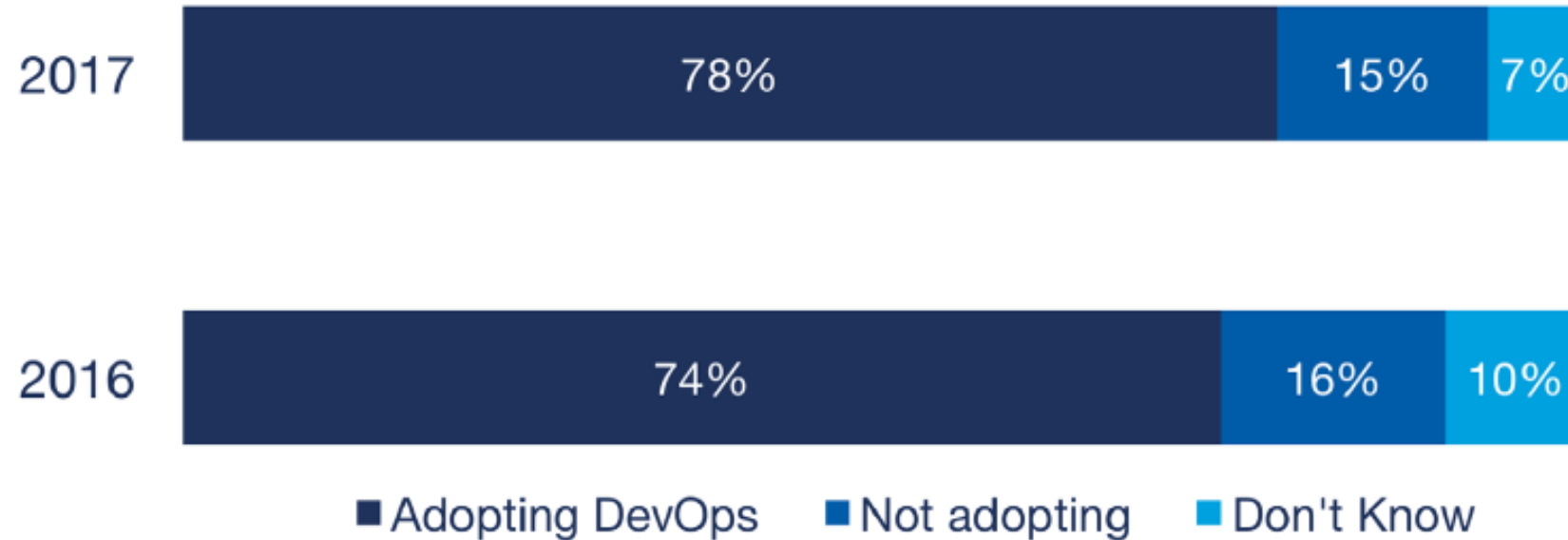
Partie 1 : Technologie Conteneurs

Docker Entreprise et DevOps...

DevOps (Adoption)

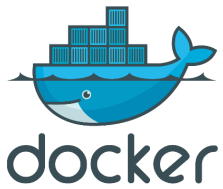


DevOps Adoption Up in 2017

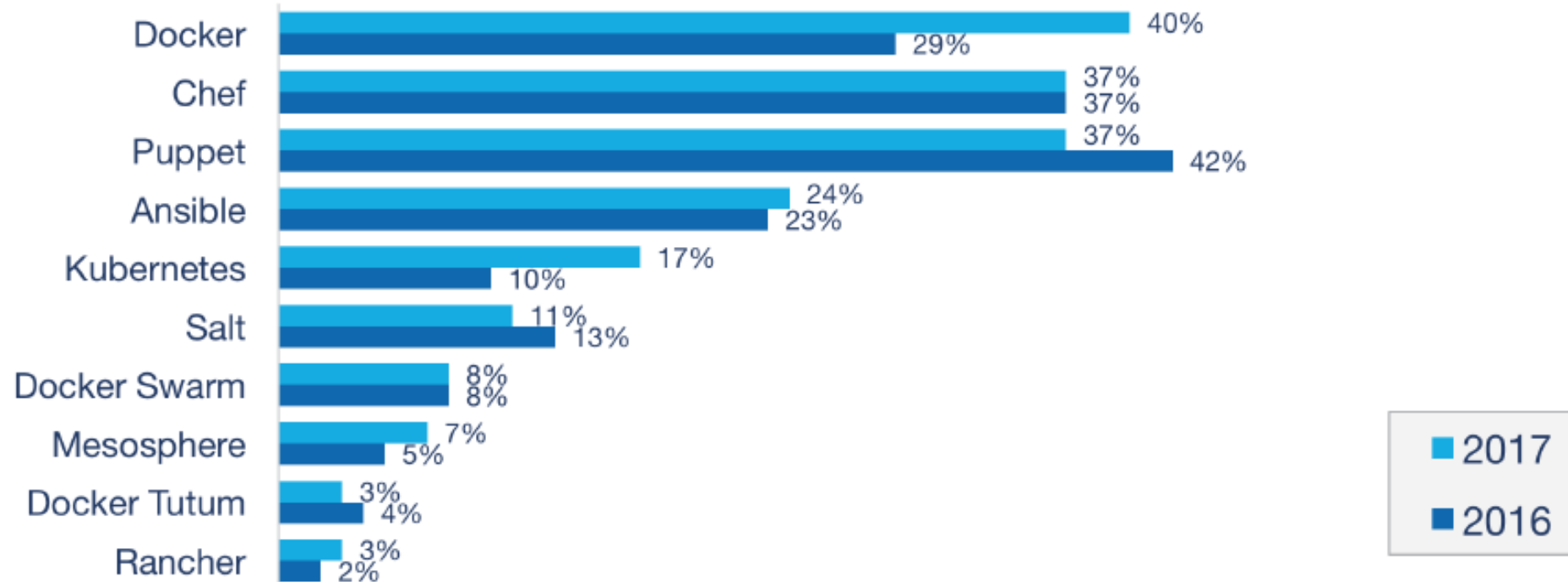


Source: RightScale 2017 State of the Cloud Report

Docker (utilisation 2016 vs 2017)

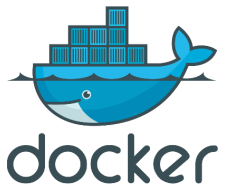


Enterprise Respondents Using DevOps Tools

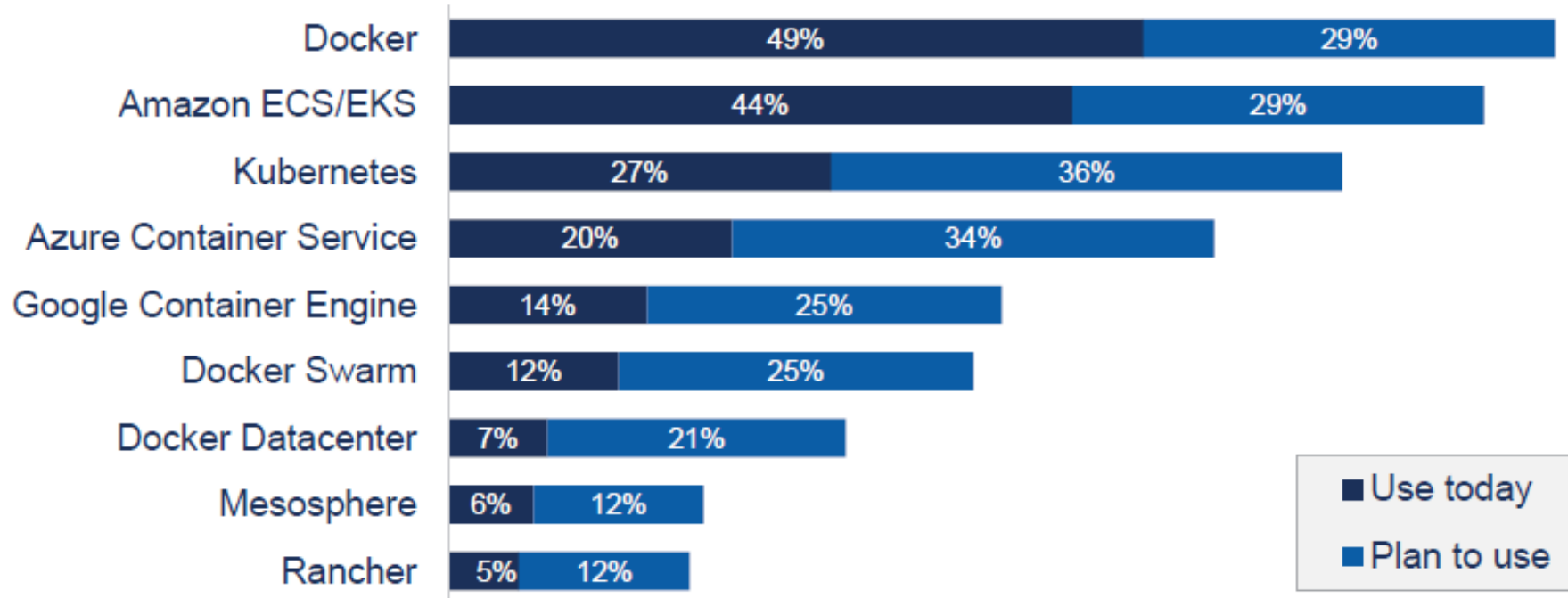


Source: RightScale 2017 State of the Cloud Report

Docker (utilisation 2018)

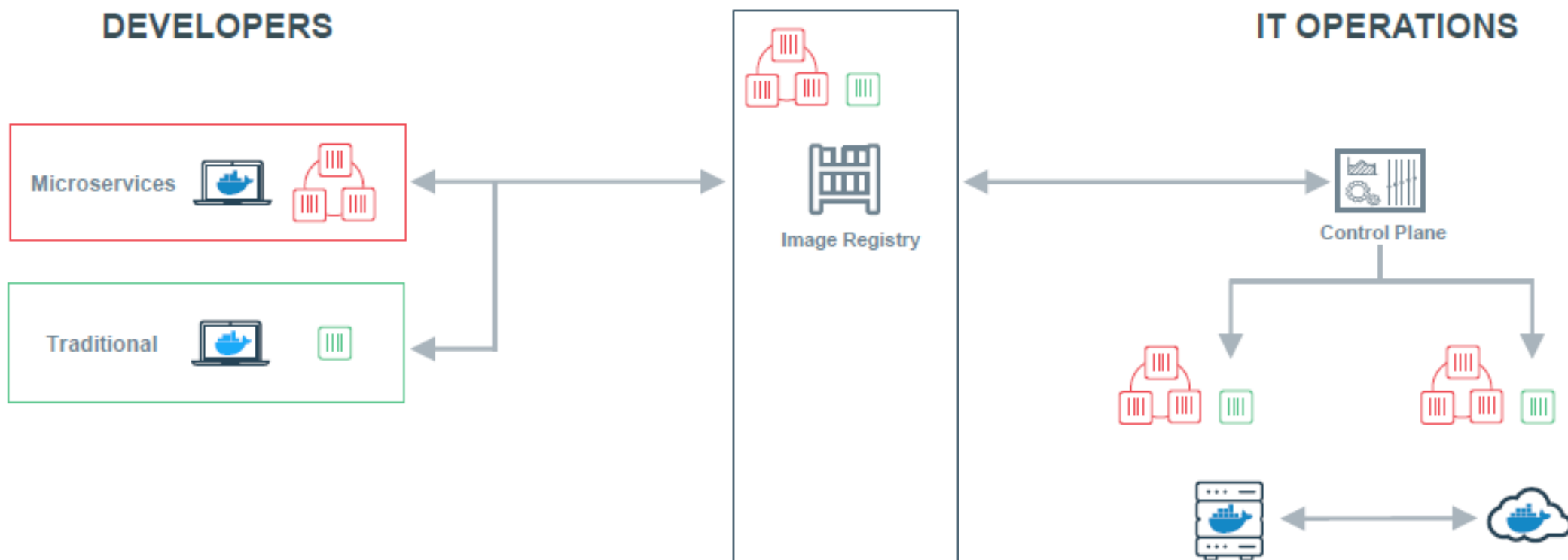


Respondents Using Container Tools

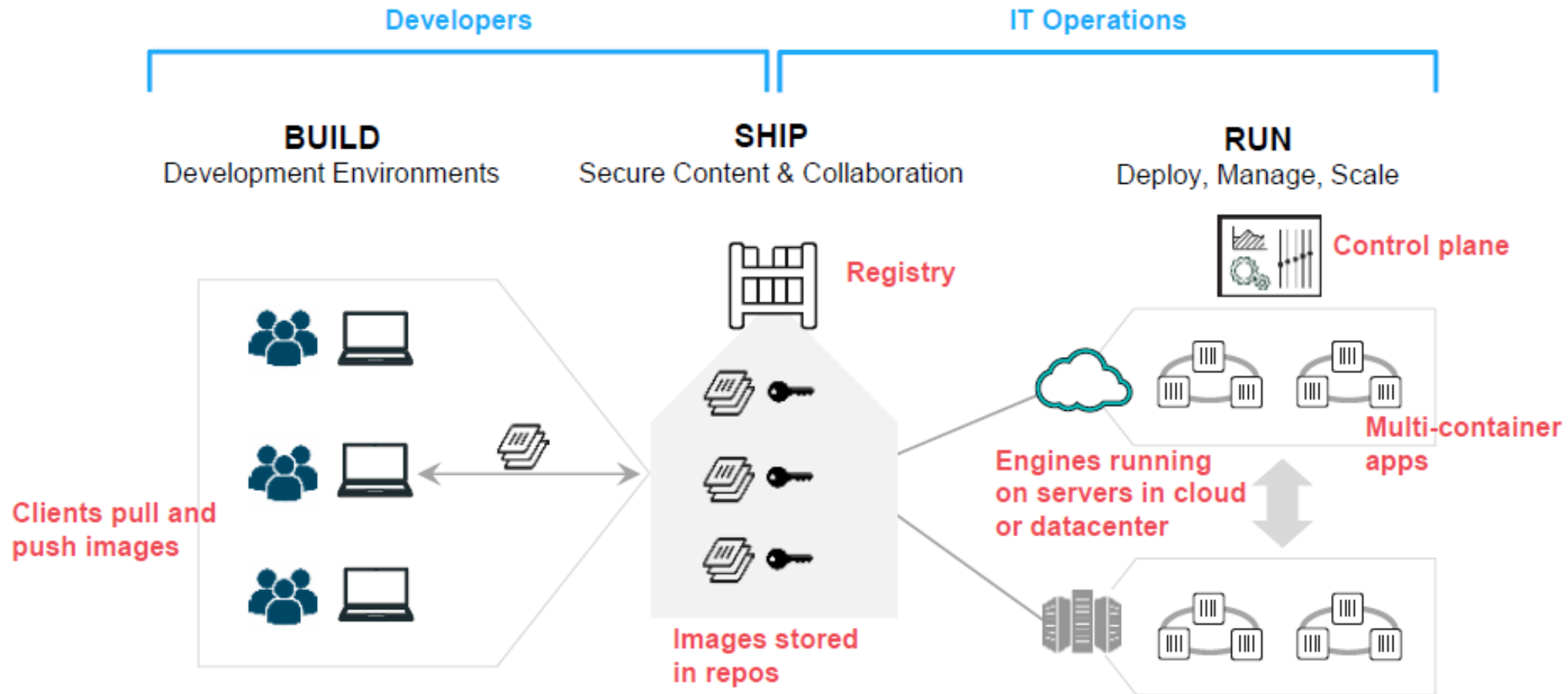
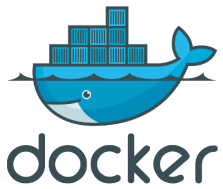


Source: RightScale 2018 State of the Cloud Report

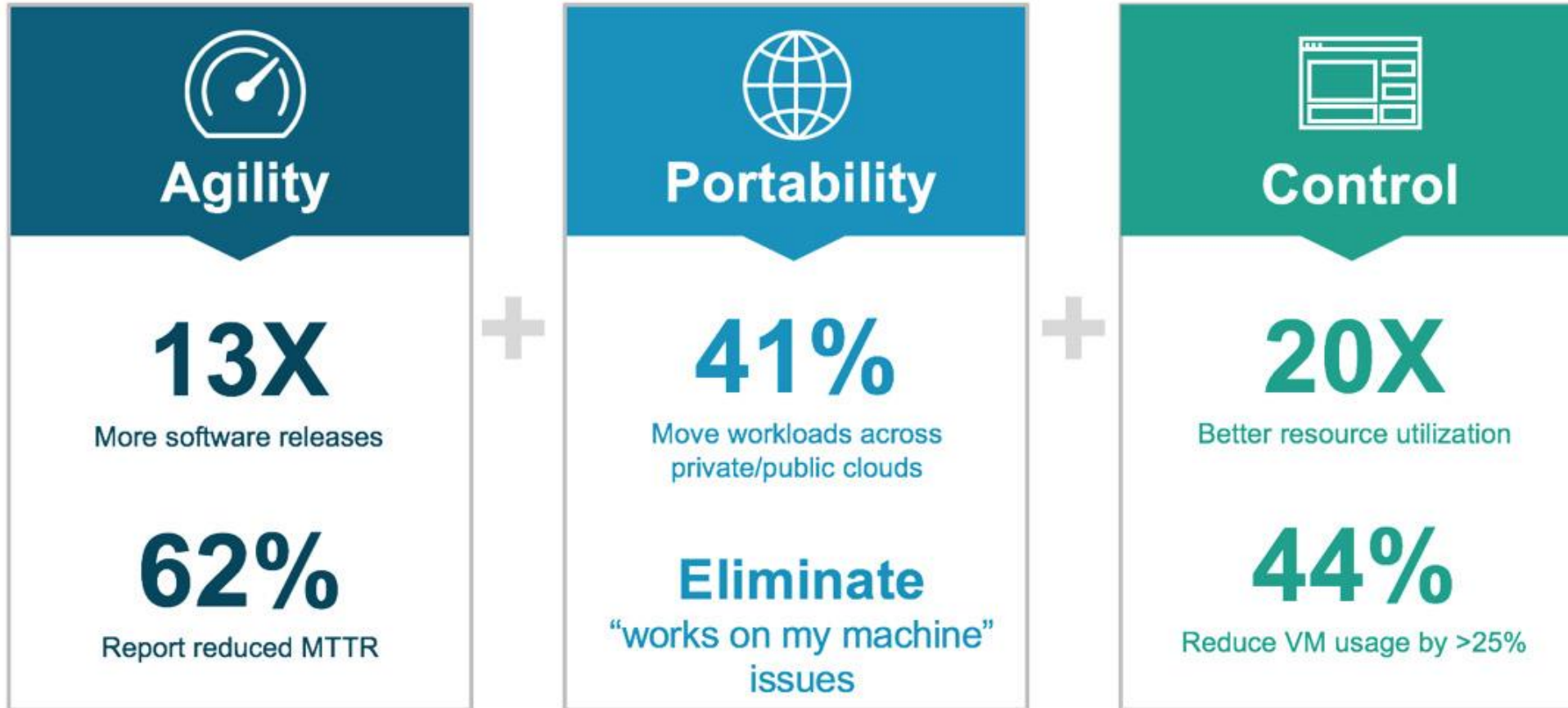
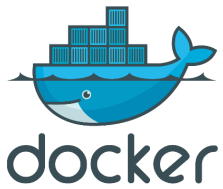
Docker EE + Microservices = Bonnes pratiques DevOps



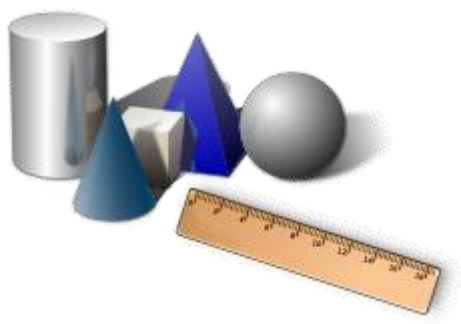
Docker EE + Microservices = Bonnes pratiques DevOps



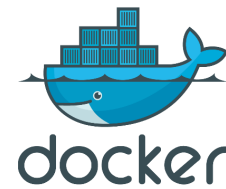
Retenons...



https://www.innovate-systems.de/downloads/docker/The_definitive_Guide_to_Docker.pdf



Partie 1 Technologie Conteneur



- Labs Docker...



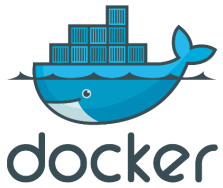
Partie 2

Orchestration des

Conteneurs

Docker Datacenter...

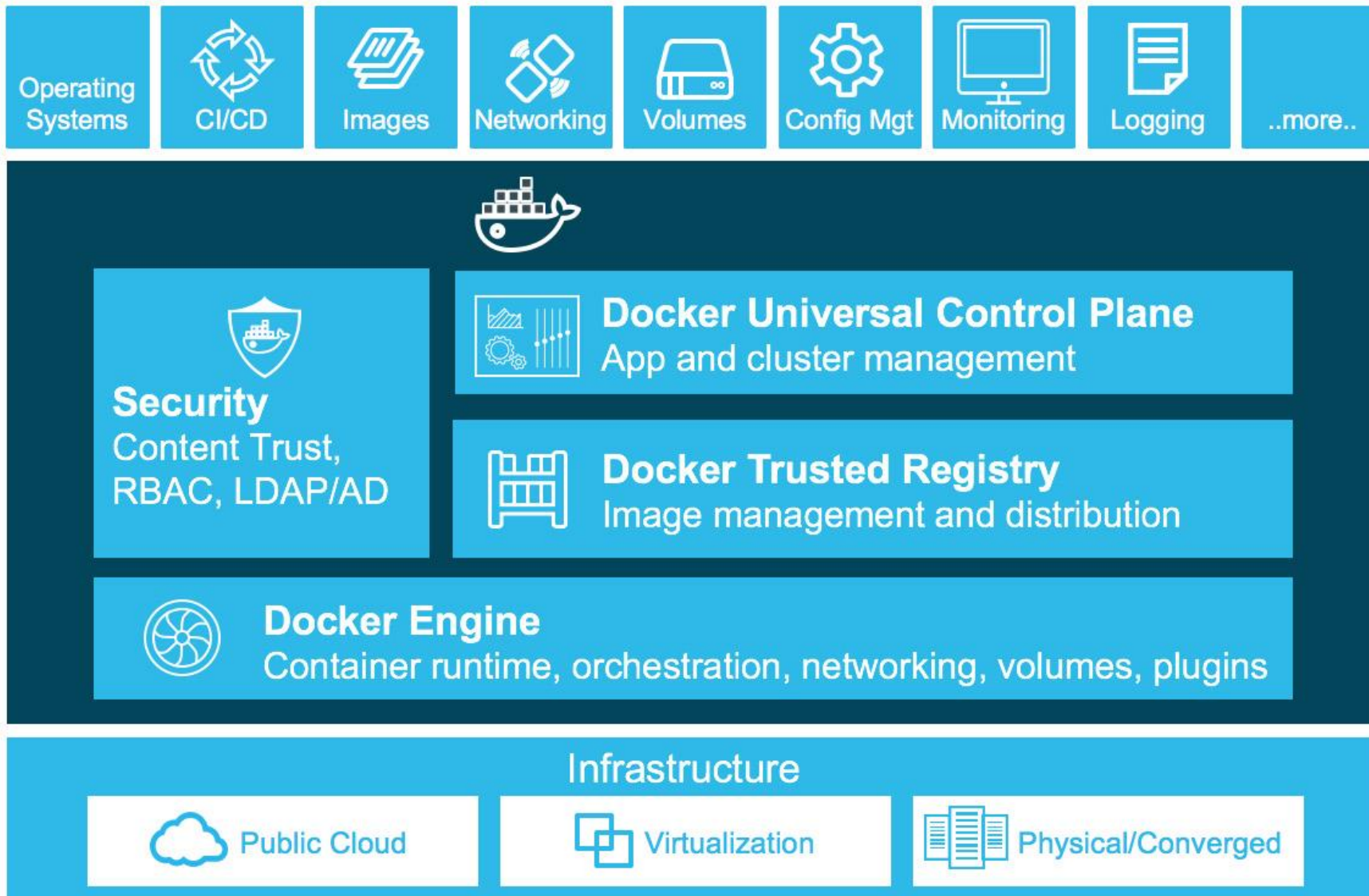
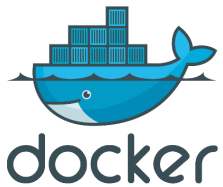
Questions...



- Orchestration des conteneurs à large échelle, pourquoi et comment?
- Quelles sont les solutions d'orchestration open source les plus utilisées sur le marché?
- Comment déployer une solution d'orchestration pour un environnement DEV et PROD?
- Quelles sont les technologies qu'on doit utiliser pour mettre en place un Cluster Docker?
- Que signifie les termes Docker Data Center, Container as Service, Infrastructure as Code?
- Peut-on combiner une solution PaaS à une infrastructure CaaS afin de fournir du DevOps as a

Service?

Docker Datacenter CaaS = Container as a Service



**Application Lifecycle
Workflow (PaaS / DevOps)**

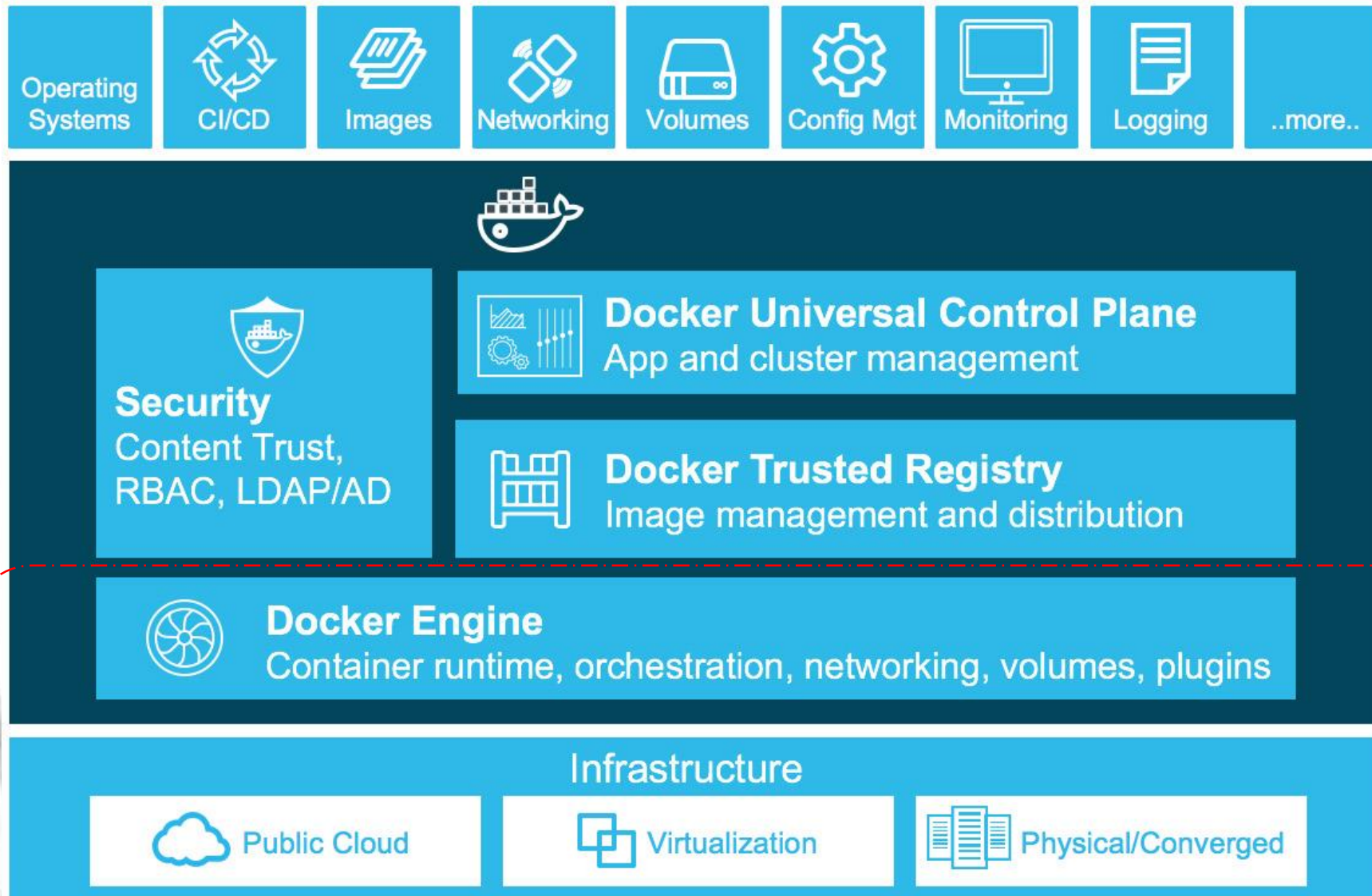
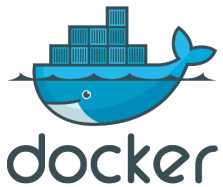
**Cluster Management
(Kubernetes / Swarm)**

**Docker Engine
(Daemon / CLI)**

**Infrastructure (On-premises
/ Cloud)**

53

Docker Datacenter CaaS = Container as a Service



**Application Lifecycle
Workflow (PaaS / DevOps)**

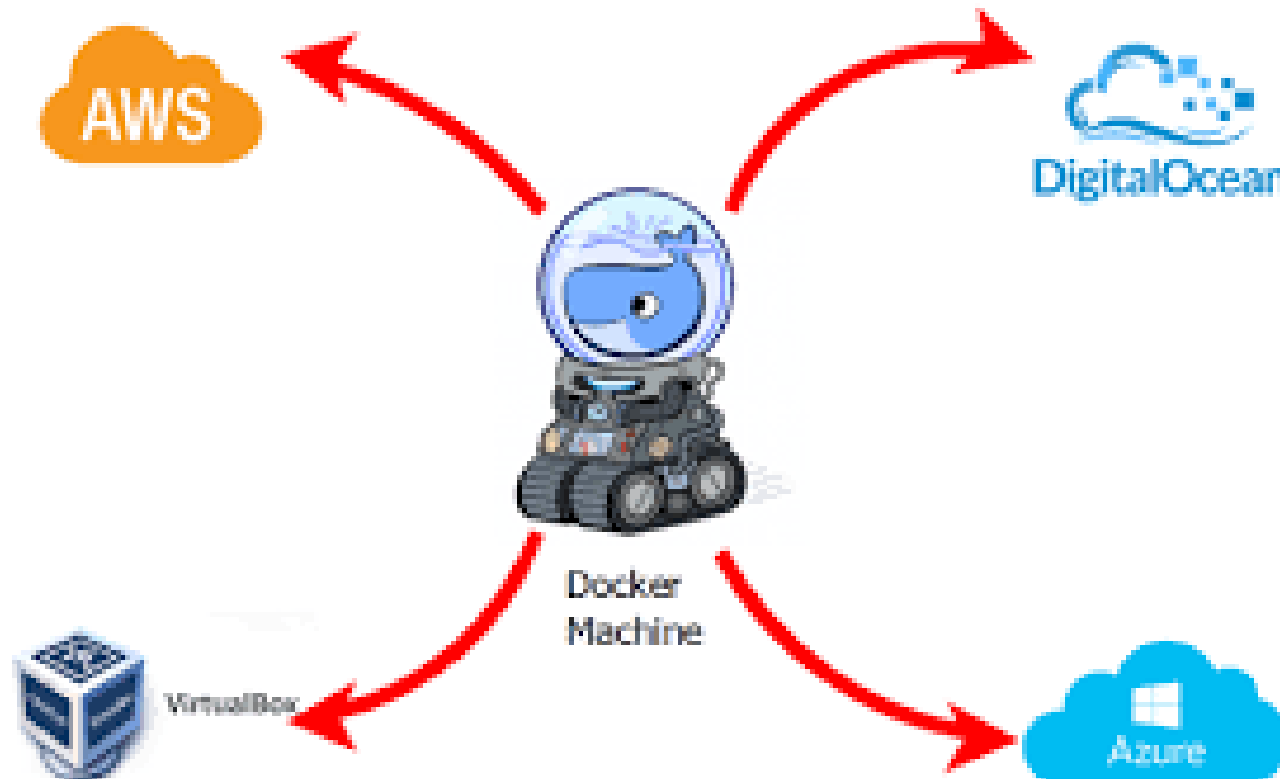
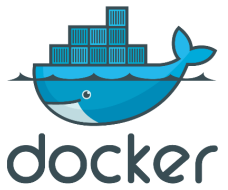
**Cluster Management
(Kubernetes / Swarm)**

**Docker Engine
(Daemon / CLI)**

**Infrastructure (On-premises
/ Cloud)**

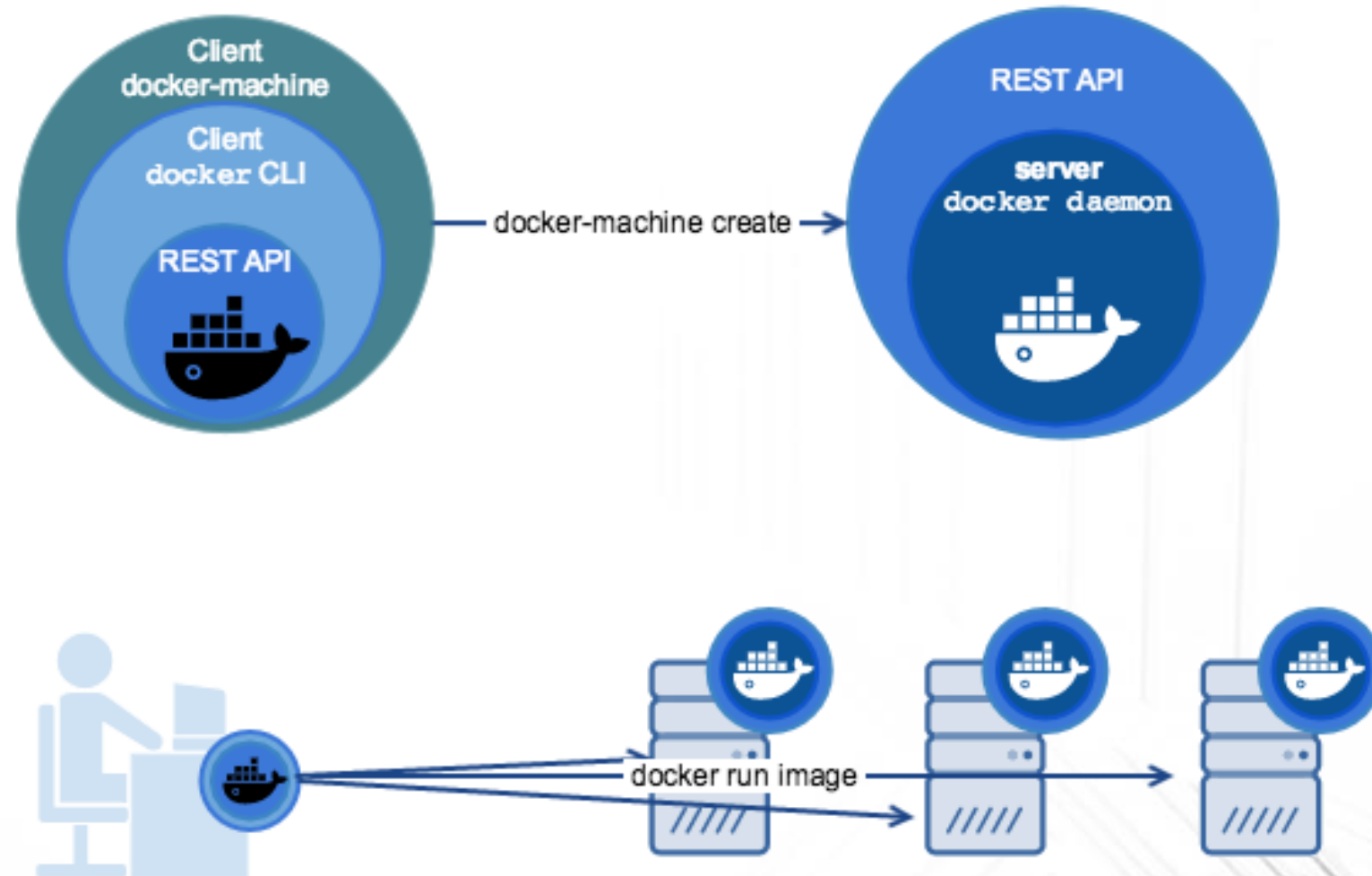
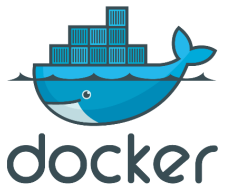
54

Docker-Machine (Container-based Host / Cluster)

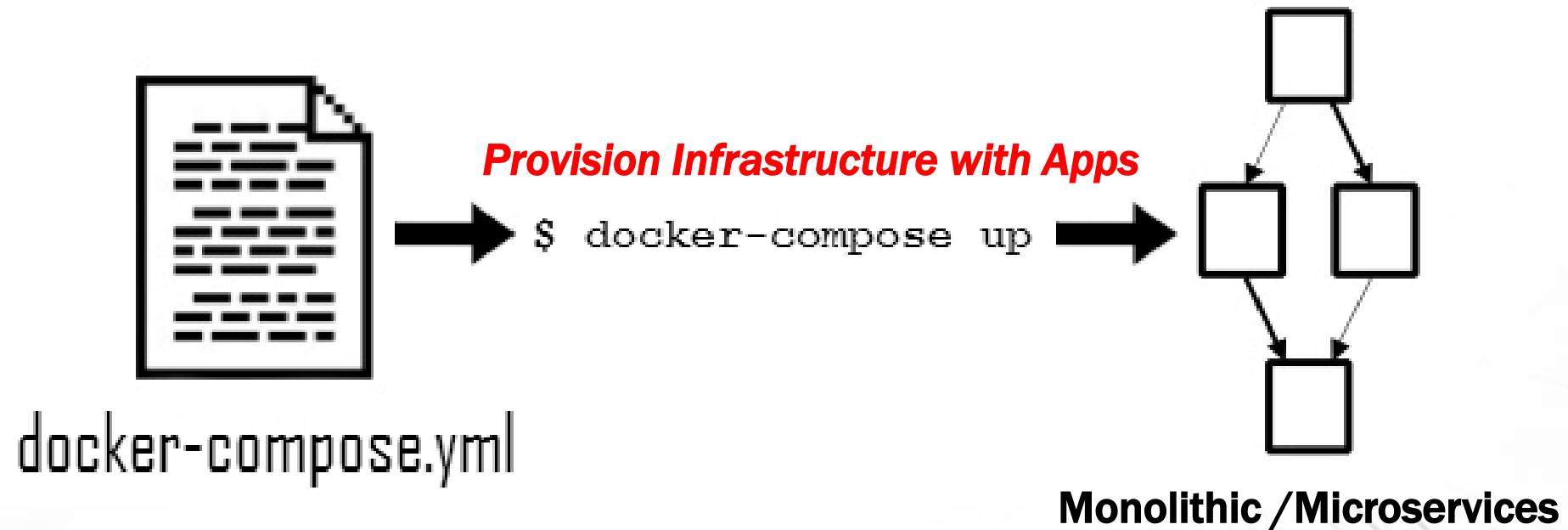
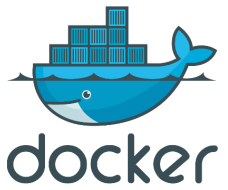


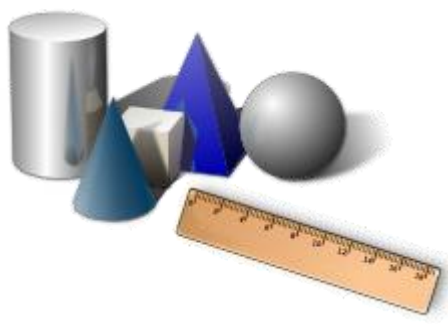
Multiple Cloud Drivers

Docker-Machine (Gestion à distance d'un Cluster)

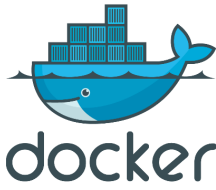


Docker-Compose (Multi-conteneurs sur la même machine)





Partie 2 Orchestration des Conteneurs

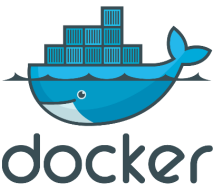


- Labs Docker Compose & Docker Machine...



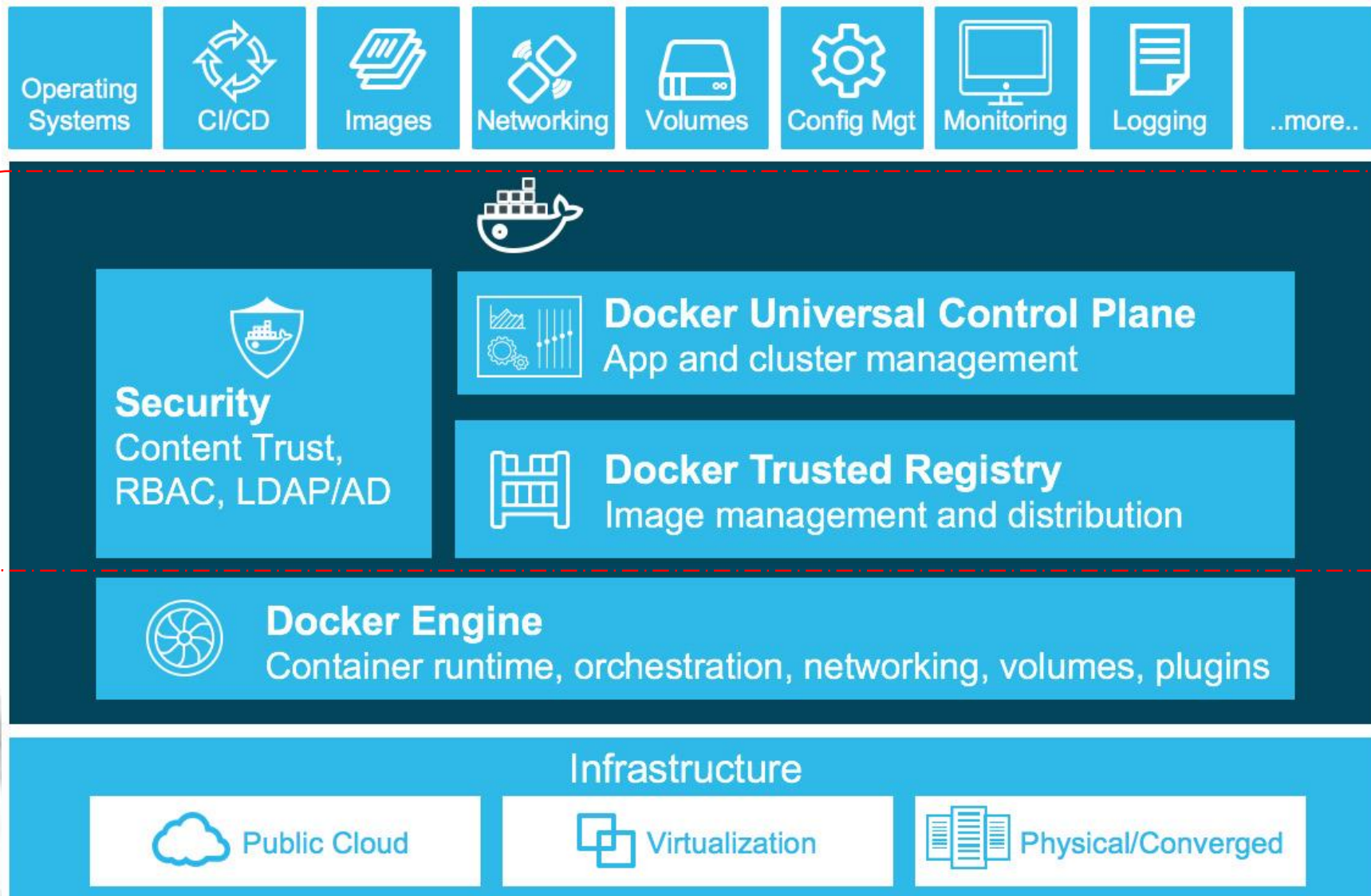
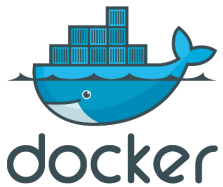
Partie 2

Orchestration des Conteneurs



Orchestration a large échelle...

Docker Datacenter CaaS = Container as a Service



**Application Lifecycle
Workflow (PaaS / DevOps)**

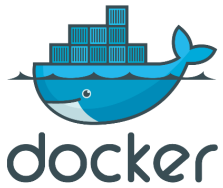
**Cluster Management
(Kubernetes / Swarm)**

**Docker Engine
(Daemon / CLI)**

**Infrastructure (On-premises
/ Cloud)**

60

Solutions d'orchestration open source



❑ Docker Machine/Compose/Swarm

- (+) Outils certifiés compatible Docker
- (-) Pas d'interface graphique libre
- (-) Difficiles à utiliser pour un déploiement à grand échelle sans développer ses propres outils d'administration.

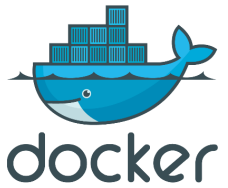


❑ Kubernetes (Google puis CNCF)

- (+) Utilisable en production (v1.1)
- (+) Support plusieurs type de Containers (Docker, Rkt, Hyper)
- (+) Permet de gérer des milliers de nœuds
- (+) HA et Autoscaling
- (-) Utilise ses propres outils réseau
- (-) API différente de Docker



Solutions d'orchestration open source



❑ Apache Mesos

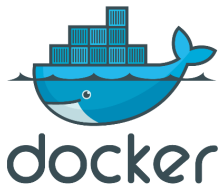
- (+) Permet de gérer plus de 10 000 noeuds.
- (+) Produits plus ancien que Docker.
- (+) Vient du monde cluster
- (-) Complexité de mise en oeuvre (nombreuses briques)
- (-) Marathon couche d'abstraction pour faire tourner Docker



❑ Openshift Origin v3

- (+) Très bon produit PaaS
- (+) Basé sur Kubernetes
- (+) DevOps as a Service (CI/CD)
- (-) Outil Redhat adapté à Docker

Docker-Swarm (Éléments clés)

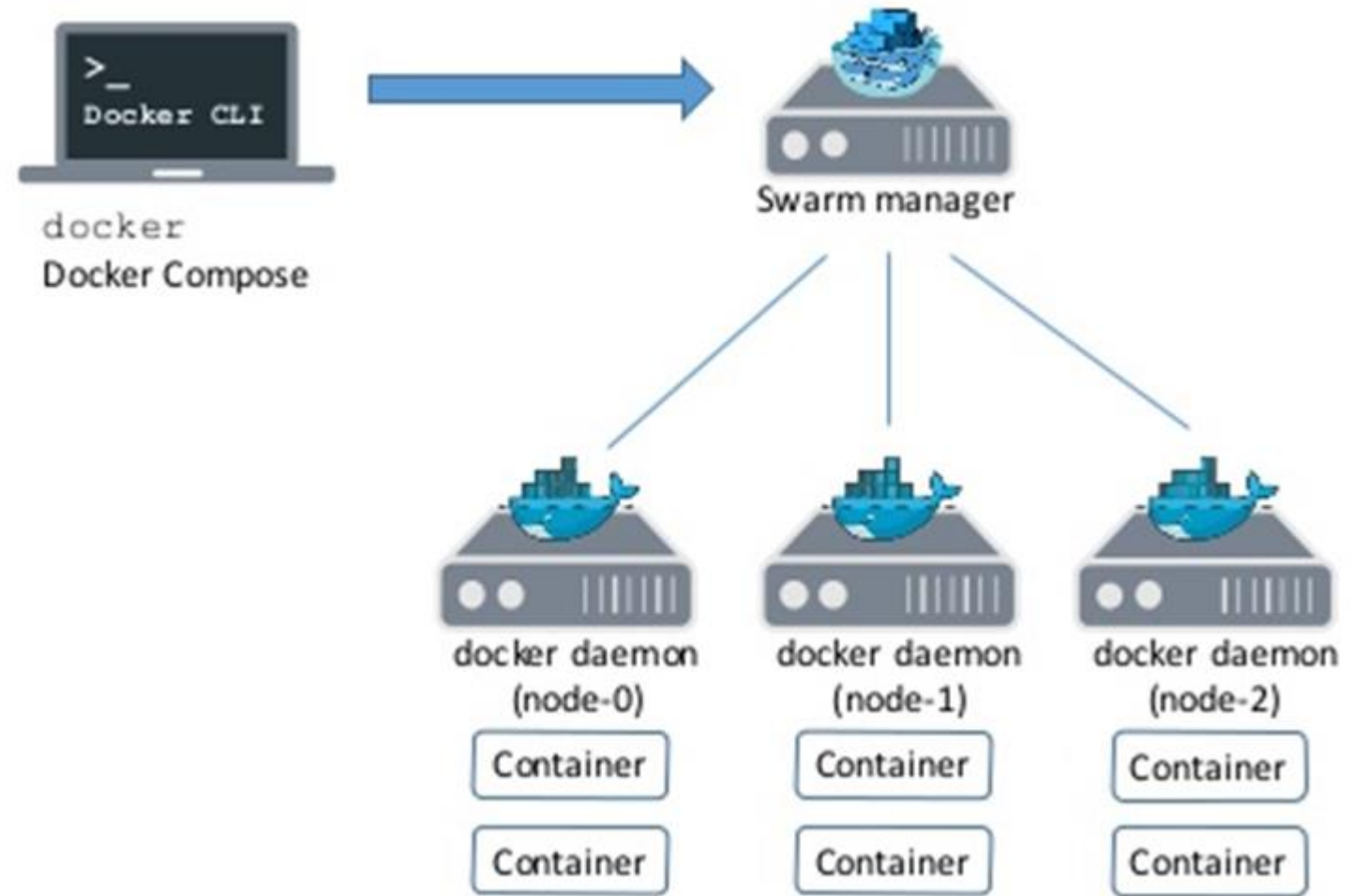


❑ Gestionnaire Swarm (manager)

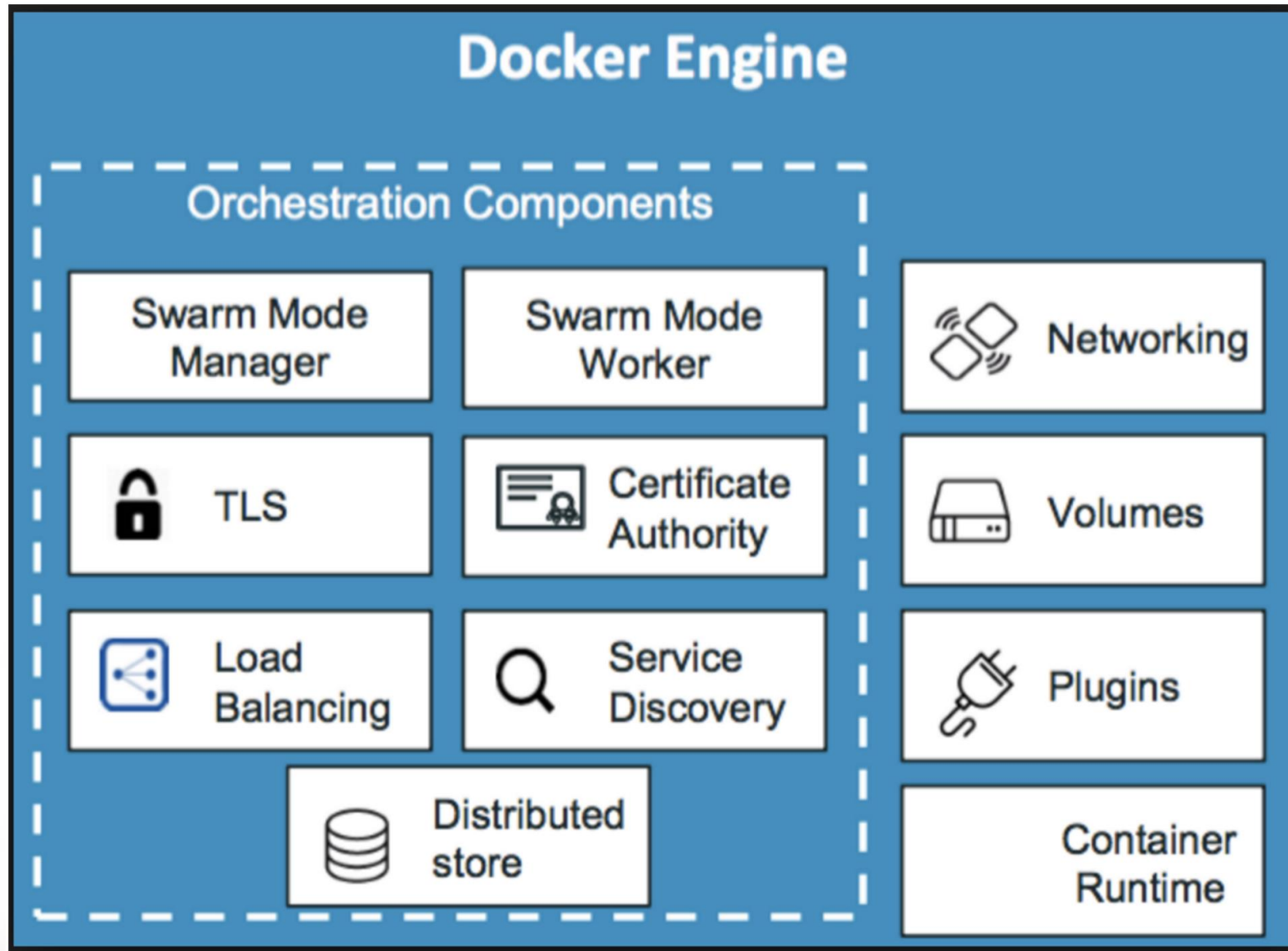
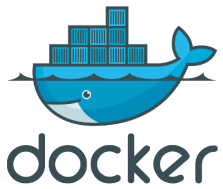
- Ordonnancement des conteneurs sur les agents
- Gestion du cluster
- Mise à l'échelle

❑ Agent (worker):

- Exécuteur des conteneurs
- Daemon Docker

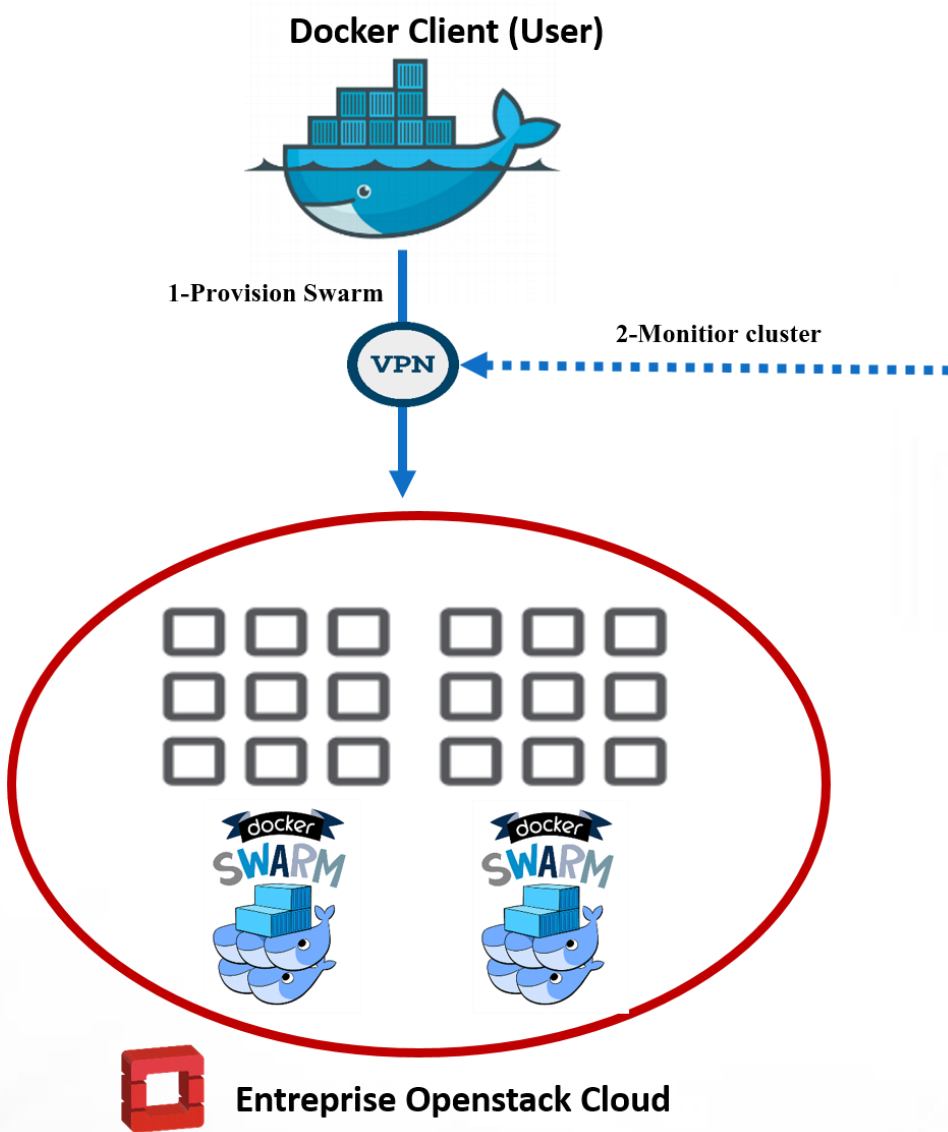


Docker-Swarm (disponible dans Docker Engine)





Docker-Swarm (en production)

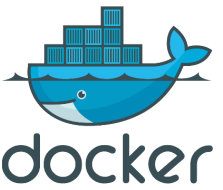


Openstack API (User)

Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
ibarta-25-04	server-cloudimg-amzn-ami-2018.03	10.0.0.101	m1.xlarge	ssh	Active	nova	Scale	Running	1 hour, 47 minutes	Create Stop
ibarta-25-04	server-cloudimg-amzn-ami-2018.03	10.0.0.102	m1.xlarge	ssh	Active	nova	Scale	Running	1 hour, 47 minutes	Create Stop
ibarta-25-04	server-cloudimg-amzn-ami-2018.03	10.0.0.103	m1.xlarge	ssh	Active	nova	Scale	Running	1 hour, 47 minutes	Create Stop
ibarta-25-04	server-cloudimg-amzn-ami-2018.03	10.0.0.104	m1.xlarge	ssh	Active	nova	Scale	Running	1 hour, 47 minutes	Create Stop

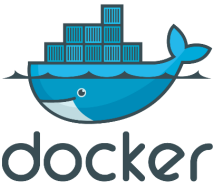
Partie 2

Orchestration des Conteneurs



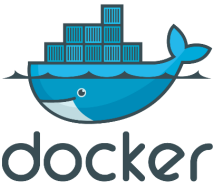
Kubernetes...

Objectifs



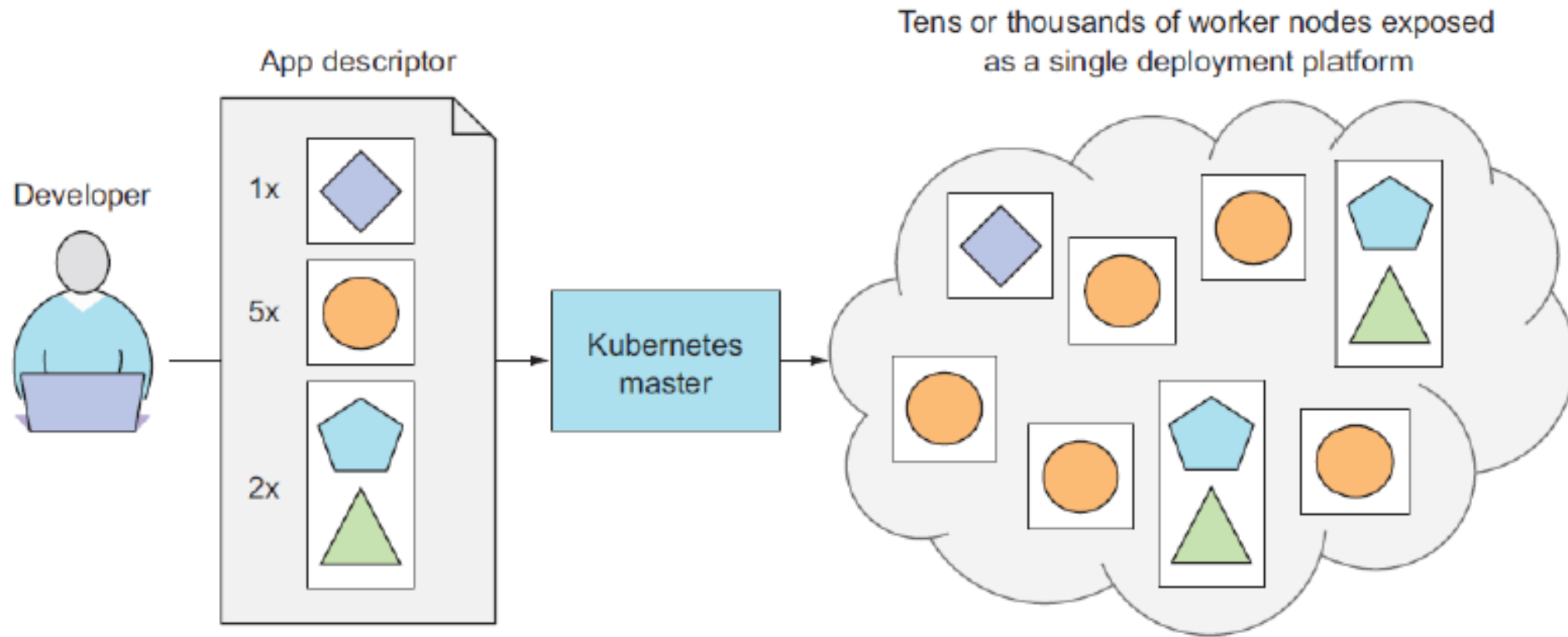
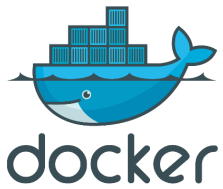
- ❑ Gérer un ensemble d'instances exécutant des conteneurs
- ❑ Gérer des ensembles de conteneurs s'exécutant sur ces instances
- ❑ Automatiser les services d'infrastructure de l'environnement de conteneurs (Réseau, volumes de stockage, DNS, DHCP, health check...
- ❑ Garantir la disponibilité des conteneurs sur l'infrastructure en fonction de règles et de filtres
- ❑ Intégrer des fonctionnalités identifiées PaaS – auto-scalabilité, centralisation des logs, reporting...

Fonctionnalités

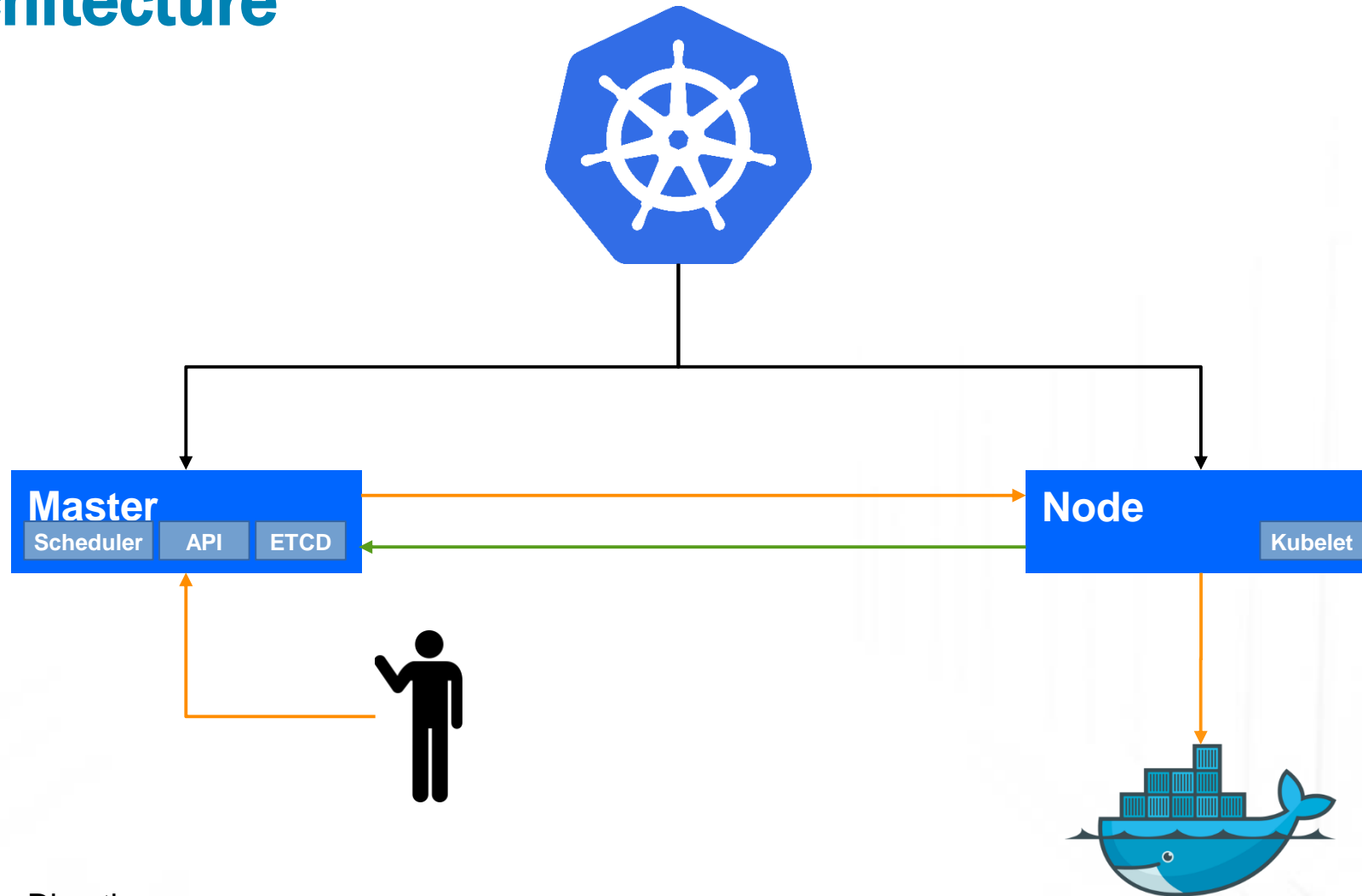
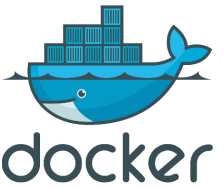


- ❑ Clustering d'instances exécutant des conteneurs
- ❑ Fonctionnalités d'orchestration :
 - Plan de contrôle (*Control Plane*)
 - Scheduler
 - Mise à l'échelle (Auto-scalabilité)
 - Supervision (*healthcheck* des instances, des conteneurs...)
 - Gestion des réplicas de conteneurs

Fonctionnement

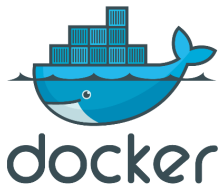


Architecture



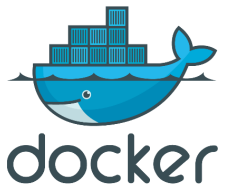
- Directives
- collecte d'informations

Eléments de base (Nœud Maître)



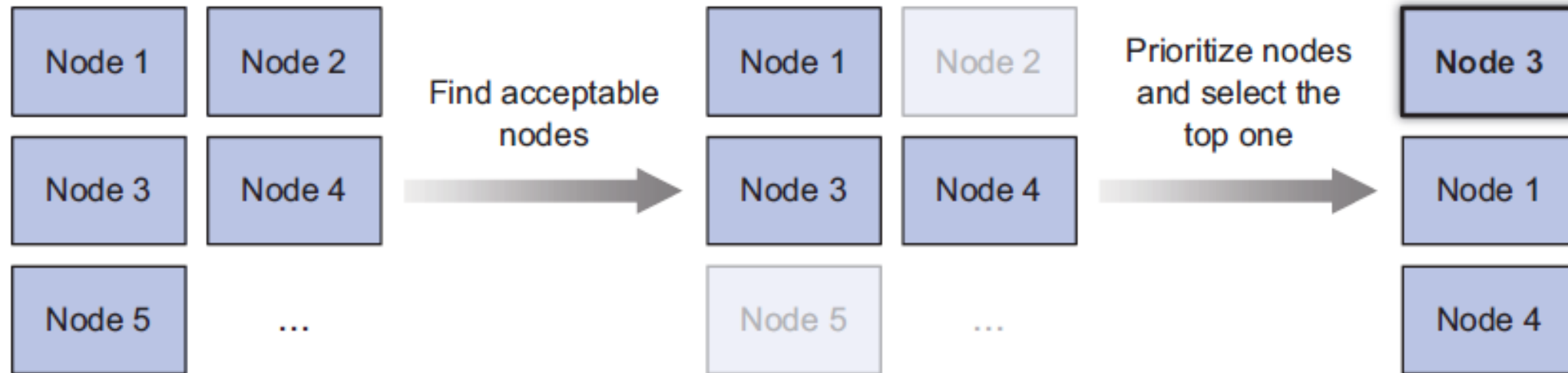
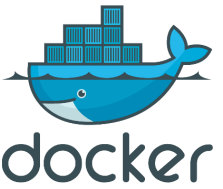
- ❑ **Maître** : exécute l'API Kubernetes et contrôle le cluster;
- ❑ **API Server** : Composant avec lequel les utilisateurs et les autres composants communiquent
- ❑ **Scheduler** : Composant qui ordonnance le placement des containers applicatifs (affecte un nœud worker sur lequel déployer une application)
- ❑ **Controller Manager** : Composant réalisant toutes les tâches de niveau cluster comme la réplication des containers (Replication Controller/Set), le suivi des nodes, la gestion des pannes, le respect des stratégies
- ❑ **ETCD** : Stockage clef/valeur fiable et distribué permettant de stocker de façon persistante l'état et la configuration du cluster

Eléments de base (Nœud Worker)

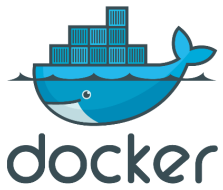


- ❑ **Container runtime** : Docker, rkt ou autre (ex : gVisor)
- ❑ **Kubelet** : Composant qui communique avec le serveur API et gère les conteneurs s'exécutant sur le nœud
- ❑ **Kubernetes Service Proxy (kube-proxy)** : Composant permettant de répartir le trafic réseau entre les containers de même nature

Scheduler (sélection d'un Nœud Worker)



Scheduler (Fonctions de prédicat et de priorisation)

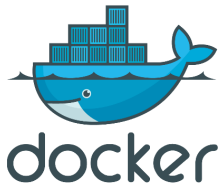


❑ Fonctions de prédicat :

- Est-ce que le nœud possède le matériel spécifié ?
- Est-ce que le nœud possède encore assez de ressources physiques (RAM, disque...) ?
- Si le POD nécessite d'être binder sur un port particulier, est-ce que ce port est disponible sur le nœud ?
- Existe-t-il des règles d'affinités/anti-affinités ?

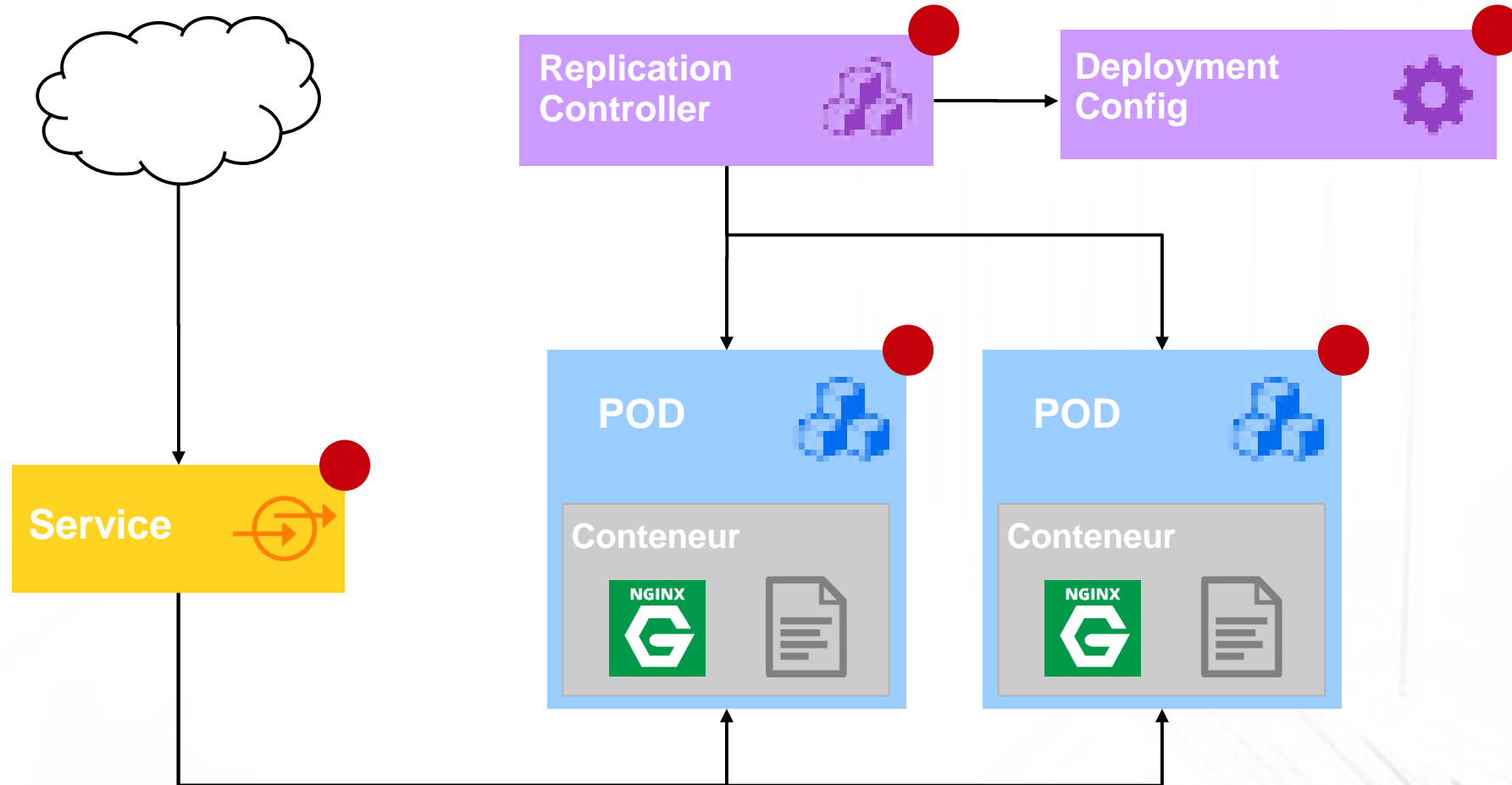
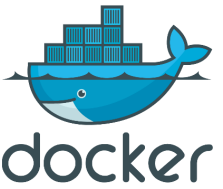
❑ Comment prioriser un nœud ?

- Nombre de containers sur un nœud par rapport à un autre
- Noeud cloud ou noeud local

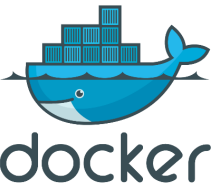


- ❑ **Label** : paire clé/valeur utilisée pour la découverte des services. Il marque les conteneurs et les fédère au sein du groupe;
- ❑ **Configuration du déploiement** : décrit configuration d'un pod (nom, network, volume, images, ports, etc.);
- ❑ **Contrôleur de réplication** : garantit que les nombres de pods demandés s'exécutent selon les exigences de l'utilisateur;
- ❑ **Service** : intégrateur et équilibreur de charge configuré automatiquement et qui s'exécute dans le cluster (Multi-pods).

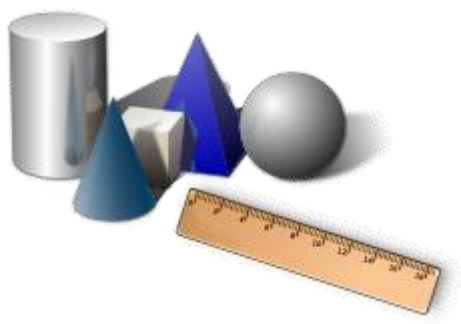
POD, RC, Service



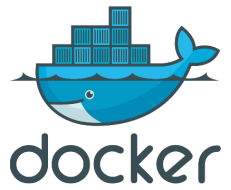
Controller Manager



- ❑ S'assurer que l'état désiré est valide et agir dans le cas contraire (utilise le mécanisme de **WATCH** pour attendre les changements transmis par le Server API).
- ❑ Une combinaison de multiples contrôleurs, chaque contrôleur est un processus unique
 - Replication Manager
 - ReplicaSet, DaemonSet, and Job controllers
 - Deployment controller
 - StatefulSet controller
 - Node controller
 - Service controller
 - Endpoints controller
 - Namespace controller
 - PersistentVolume controller



Partie 2 Orchestration des Conteneurs



- Lab kubernetes...



Questions & Réponses

Q&R