Architecture à base de Micro-Services

Abdelhak-Djamel SERIAI

Cloud : Définition

- Le cloud computing ou informatique en nuage :

  - Est un modèle permettant d'établir un accès par le réseau à un réservoir partagé de ressources informatiques standard configurables (réseau, serveurs, stockage, applications et services) qui peuvent être rapidement mobilisées et mises à disposition en minimisant les efforts de gestion ou les contacts avec le fournisseur de service.

  - Est un ensemble de matériels, de raccordements réseau et de logiciels fournissant des services qu'individus et collectivités peuvent exploiter depuis n'importe où dans le monde.

Cloud : Principe

- La puissance de calcul ou de stockage de serveurs informatiques peuvent être exploités à distance.
    - Au lieu d'obtenir de la puissance de calcul par acquisition de matériel et de logiciel, le consommateur se sert de puissance mise à sa disposition par un fournisseur via l'Internet.
    - Location à la demande ou au forfait selon des critères techniques tels que la puissance ou la bande passante.

Cloud : Avantages

- Les caractéristiques du cloud computing intéressantes pour les entreprises sont :
  - La réduction du coût total de possession des systèmes informatiques.
  - La facilité d'augmenter ou de diminuer les ressources.
  - Décharger les équipes informatiques des entreprises, qui ont alors plus de disponibilité pour des activités à haute valeur ajoutée.
  - Permet également aux petites entreprises d'avoir accès à des services jusque-là réservés aux grandes entreprises en raison de leur coût.

Cloud : Caractéristiques

- Elasticité et disponibilité des ressources en libre-service
  - Adaptation automatique à la demande de la capacité de stockage et puissance de calcul selon le besoin du consommateur.
  - La demande est automatique et la réponse est immédiate.

- Ouverture
  - Les services de cloud computing sont accessibles via l'Internet, via des techniques standardisées, tant pour un ordinateur qu'un téléphone ou une tablette.

- Mutualisation
  - Elle permet de combiner des ressources hétérogènes (matériel, logiciel, trafic réseau) pour servir plusieurs consommateurs à qui les ressources sont automatiquement attribuées.
  - La mutualisation améliore l'évolutivité et l'élasticité ; elle facilite l'adaptation automatique des ressources aux variations de la demande.

- Paiement à l'usage
  - La quantité de service consommée dans le cloud est mesurée, à des fins de contrôle, d'adaptation des moyens techniques et de facturation.

Cloud : Types

- Un nuage peut être public, privé ou communautaire :
  - Un nuage est public :
    - S'il est mis à disposition du grand public.
    - Les services sont généralement mis à disposition par une entreprise utilisant une infrastructure lui appartenant (par exemples les GAFAM).
  - Un nuage est privé :
    - S'il est destiné exclusivement à une organisation qui peut le manipuler elle-même ou faire appel à des services fournis par des tiers.
  - Un nuage est communautaire :
    - S'il utilise une infrastructure provenant d'un ensemble de membres partageant un intérêt commun, comme dans le cas des milieux universitaires.

Cloud : Les services

- Les principaux services proposés en cloud sont :
  - SaaS - Software as a Service
  - PaaS - Platform as a Service
  - IaaS - Infrastructure as a Service

- Les autres services :
  - Data as a service
  - Business process as a service (BPaaS)
  - Desktop as a service (DaaS)
  - Network as a service (NaaS)
  - Etc.

Cloud : Les services

- IaaS - infrastructure as a service ou infrastructure en tant que service.
  - C'est le service de plus bas niveau. Il consiste à offrir un accès à un parc informatique virtualisé.
  - Des machines virtuelles sur lesquelles le consommateur peut installer un système d'exploitation et des applications.
  - Le consommateur est dispensé de l'achat de matériel informatique.
  - Ce service s'apparente aux services d'hébergement classiques des centres de traitement de données (datacenters).

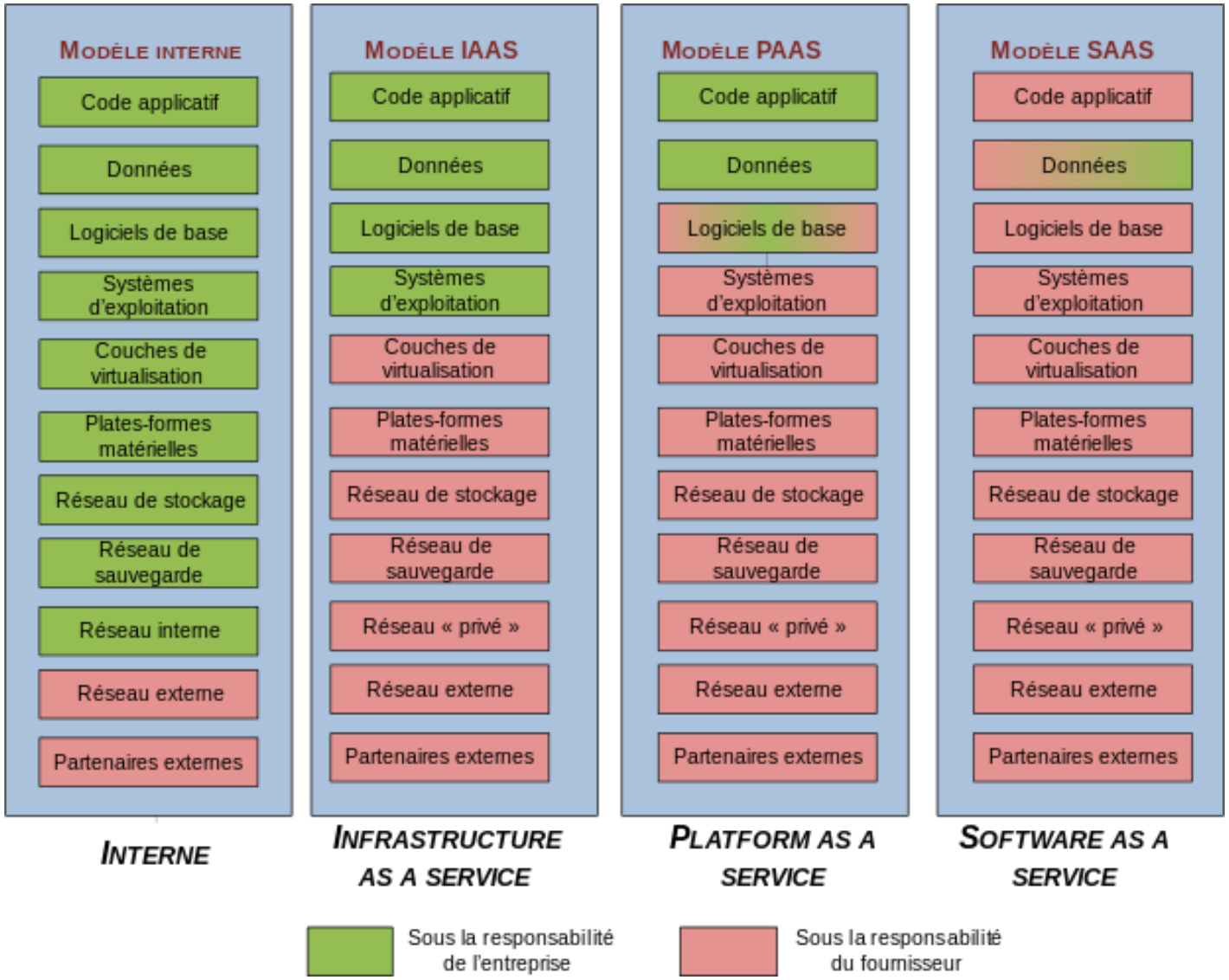
Cloud : Les services

- PaaS - platform as a service ou Plate-forme en tant que service.
    - Le système d'exploitation et les outils d'infrastructure sont sous la responsabilité du fournisseur.
    - Le consommateur a le contrôle des applications et peut ajouter ses propres outils.
    - La situation est analogue à celle de l'hébergement Web, où le consommateur loue l'exploitation de serveurs sur lesquels les outils nécessaires sont préalablement placés et contrôlés par le fournisseur.
        - La différence étant que les systèmes sont mutualisés et offrent une grande élasticité, alors que, dans une offre classique d'hébergement Web, l'adaptation fait suite à une demande formelle du consommateur.

Cloud : Les services

- SaaS - software as a service ou logiciel en tant que service.
  - Des applications sont mises à la disposition des consommateurs.
  - Les applications peuvent être manipulées à l'aide d'un navigateur Web ou installées de façon locative sur un PC, et le consommateur n'a pas à se soucier d'effectuer des mises à jour, d'ajouter des patches de sécurité et d'assurer la disponibilité du service.
  - Un fournisseur de software as a service peut exploiter des services de type platform as a service, qui peut lui-même se servir de infrastructure as a service.

Cloud : Les services



Cloud : Les autres services

- Data as a service
  - Correspond à la mise à disposition de données délocalisées quelque part sur le réseau.
  - Ces données sont principalement consommées par ce que l'on appelle des applications composites (en anglais mashups).
- Business process as a service (BPaaS)
  - Consiste à externaliser une procédure d'entreprise suffisamment industrialisée pour s'adresser directement aux maîtres d'ouvrage, sans nécessiter l'aide de professionnels de l'informatique.
- Desktop as a service (DaaS) - « bureau en tant que service »
  - Est l’externalisation d'une infrastructure de bureau virtuel (en anglais virtual desktop infrastructure) auprès d’un fournisseur de services.

Cloud : Les autres services

- Network as a service (NaaS)
  - Correspond à la fourniture de services réseaux, suivant le concept de software defined networking (SDN).

- Storage as a service (STaaS) :
  - Correspond au stockage de fichiers chez des prestataires externes qui les hébergent pour le compte de leurs clients.

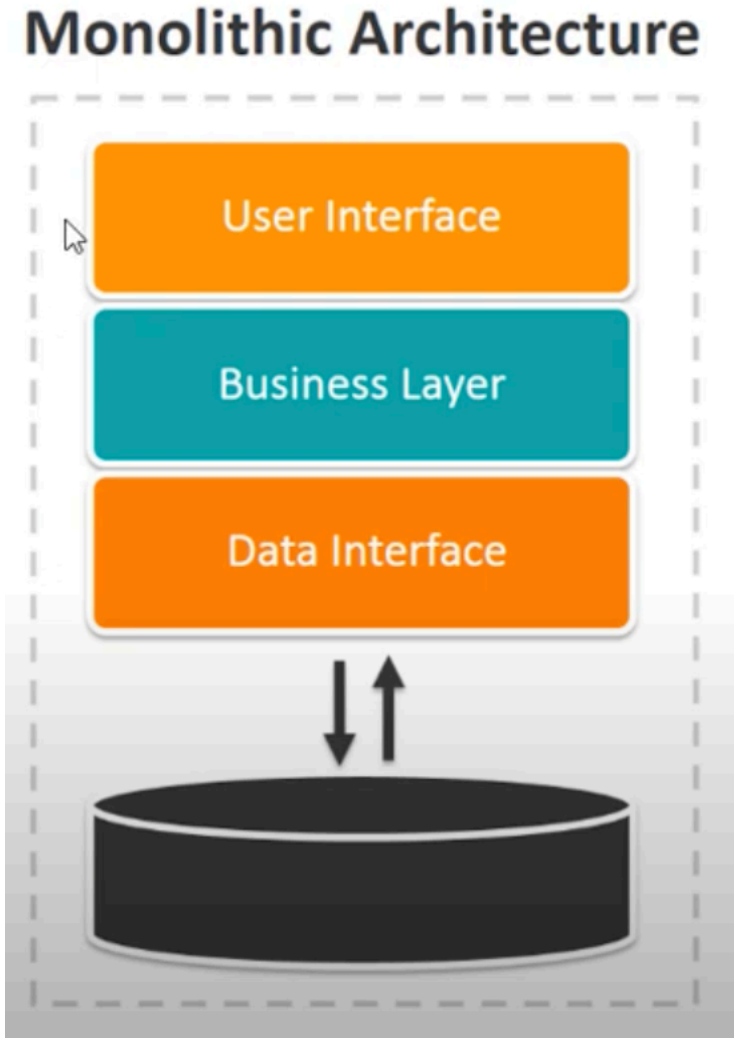
- Communication as a service (CaaS)
  - Correspond à la fourniture de solutions de communication substituant aux matériels et serveurs locaux (PABX, ACD, SVI…) des ressources partagées sur Internet.
- Etc.

Style Monolithique : Définition

- Une application monolithique décrit une application logicielle à un seul niveau et dans laquelle l'interface utilisateur et le code d'accès aux données sont combinés en un programme unique à partir d'une plate-forme unique.

- Une application monolithique est autonome et indépendante des autres applications informatiques.

- Une application monolithique décrit une application logicielle conçue sans modularité.



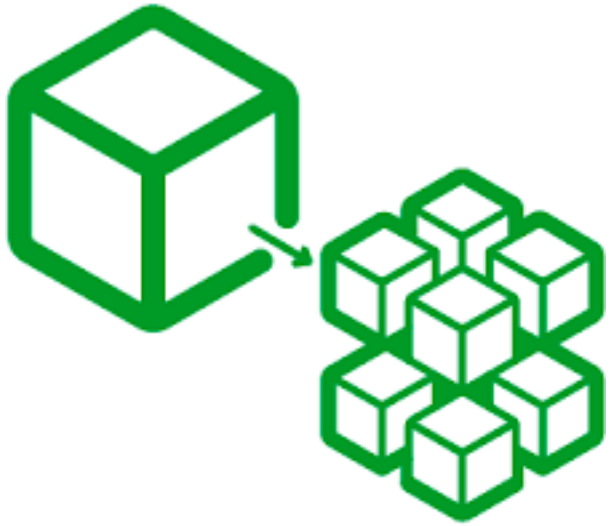
Style Monolithique : Avantages et inconvénients

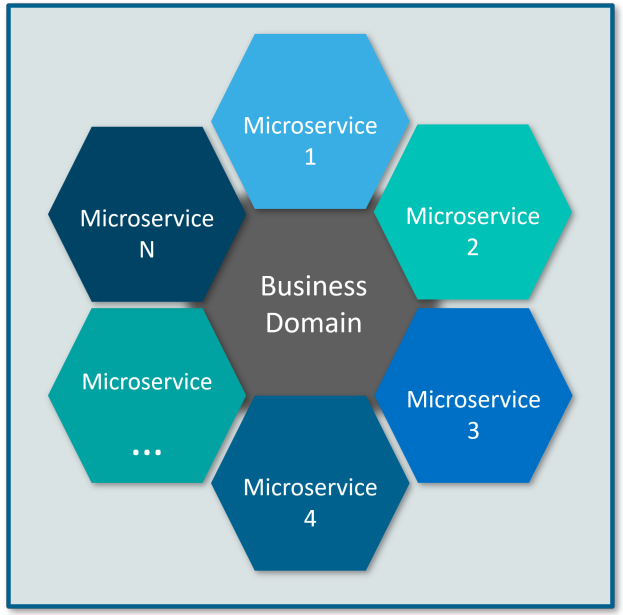
- Avantages
  - L'utilisation de la même technologie facilite le développement et l'intégration.
  - Déploiement facile.
  - Cohésion de l'équipe de développement.

- Inconvénients
  - L'utilisation d'une même technologie peut ne pas être adaptée à tous les besoins.
  - Nécessite de tout arrêter et de redéployer pour l'évolution et la maintenance.
  - Temps considérable de re(déploiement).
  - Problème de « scalabilité ».
  - Couplage fort entre les équipes de développement.

Style Micro-service : Définition

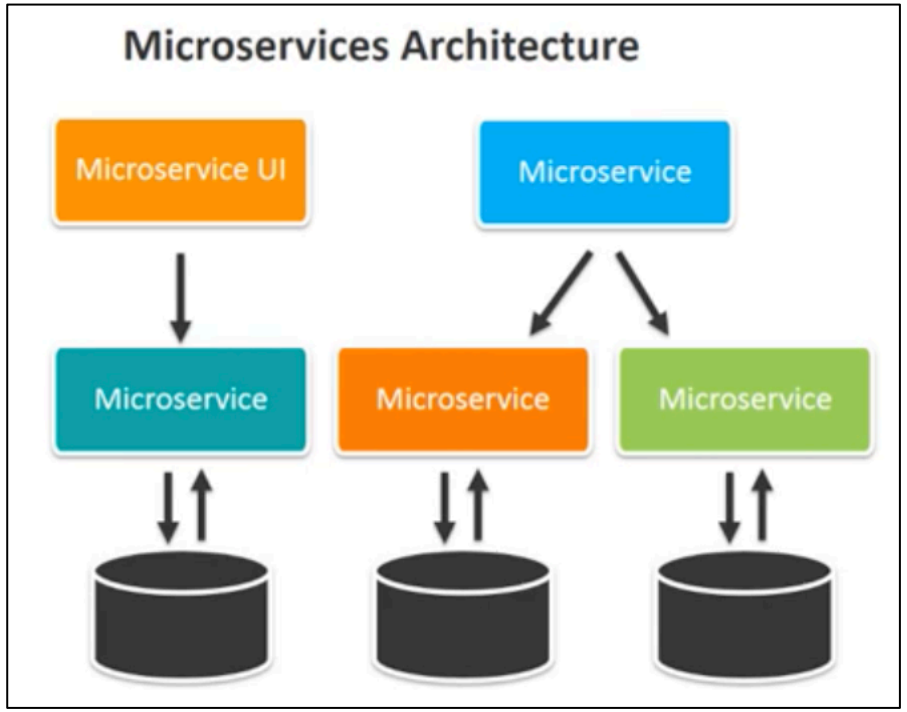
- « While there is no precise definition of this architectural style, there are certain common characteristics around organization around business capability, automated deployment, intelligence in the endpoints, and decentralized control of languages and data. » (Martin Fowler).





Style Micro-service : Principe

- Une application = un assemblage de « petits » services indépendants
  - Chaque micro-service réalise un processus métier ou une préoccupation transverse
    - « capability », « unité fonctionnelle ».
    - Ex : vente, CRM, comptabilité, front-end, GUI…
- Domain-Driven Design » :
  - Découpage d'un projet en groupements fonctionnels.
  - Isolation de ces groupements entre eux.
  - Chaque groupement est un mini-projet indépendant.



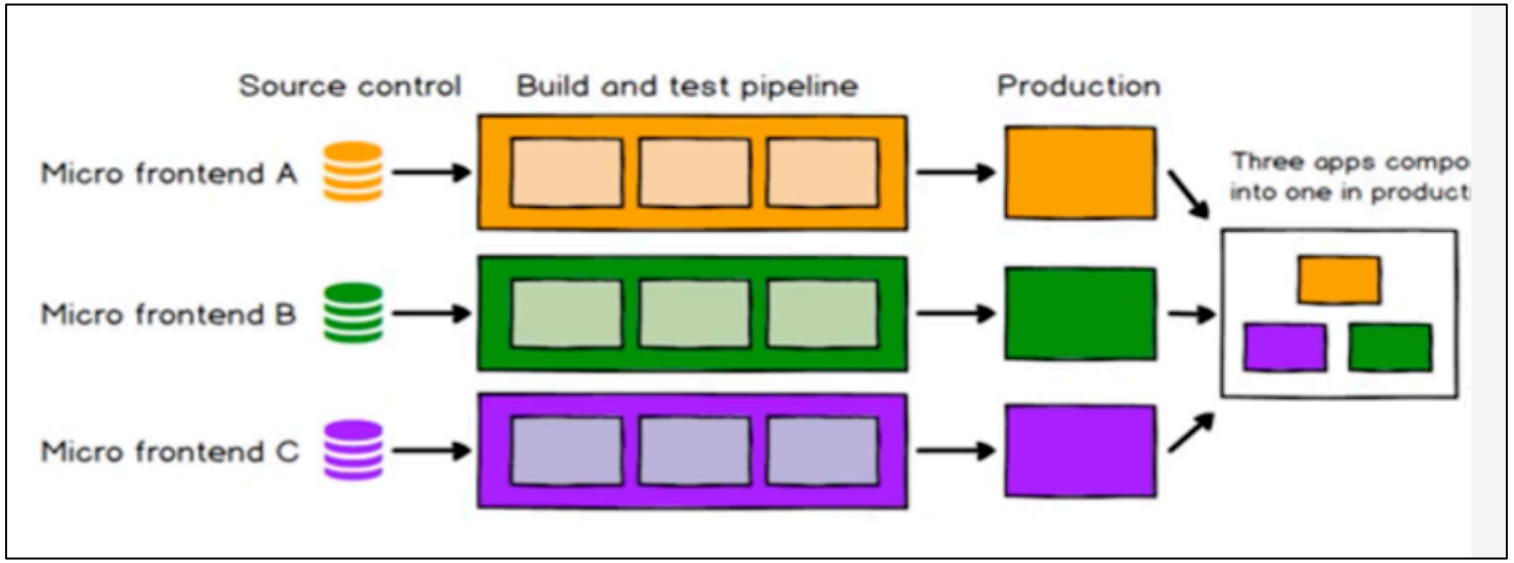
Style Micro-service : Technique

- Techniquement, les micro-services sont :
  - Programmés dans des langages hétérogènes.
  - Exécutés dans des processus séparés.
  - Liés à leurs propres supports de persistance.
  - Développés et déployés dans des projets distincts.

- La gestion centralisée de l'application est réduite au minimum :
  - Communication par mécanismes « légers ».
    - Exemple : REST
  - Les services peuvent utiliser des mécanismes de stockage différents.
  - L'« intelligence » de l'application est répartie dans les services.

Style Micro-service : Avantages

- Modularité, couplage faible :
  - Séparation des préoccupations (et des développements).
  - Liberté des choix technologiques.
  - Possibilité de redéployer composant par composant.
  - Indépendance des équipes de dev (Développement)
- Interfaces de communication réseau :
  - Pérennité des standards.
  - Distribution possible.
  - « Scalability by design »
    - Verticale : permet de ne répliquer que les services chargés.
    - Horizontale : déport des services les plus chargés vers des nœuds différents.



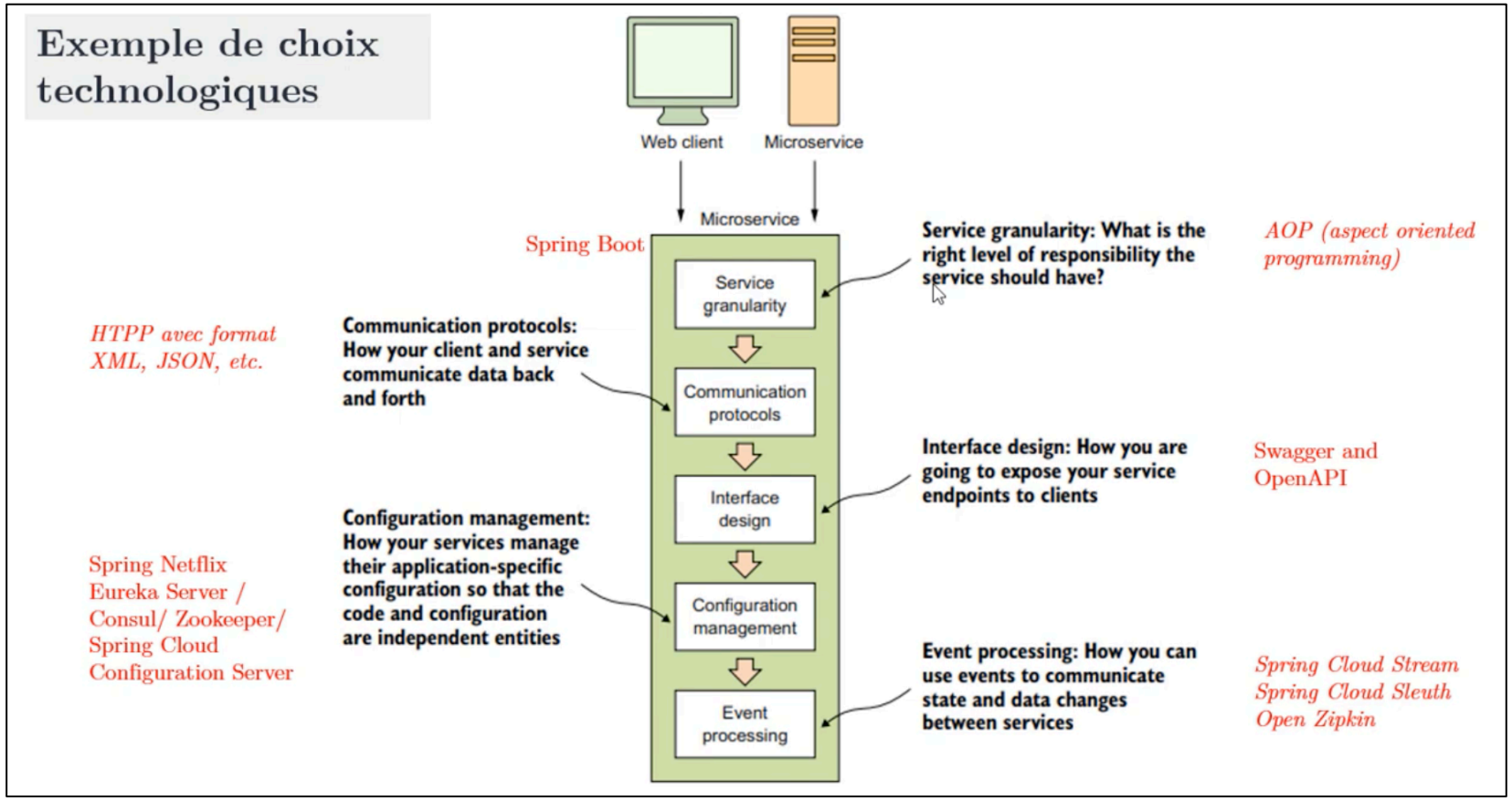
Style Micro-service : Inconvénients

- Surcoût
  - Nécessite une communication orientée-message entre les services (vs. appel de méthodes).
  - Nécessite une « surveillance » du fonctionnement des services (monitoring, tolérance aux pannes, pilotage).
  - Accès aux ressources partagées.

- Humain
  - Nécessite que les équipes soient effectivement structurées selon l'architecture du produit.
  - Vision / mise au point globale de l'application.
  - Pas triviale, peut nécessiter plusieurs itérations.

Outils et plateformes

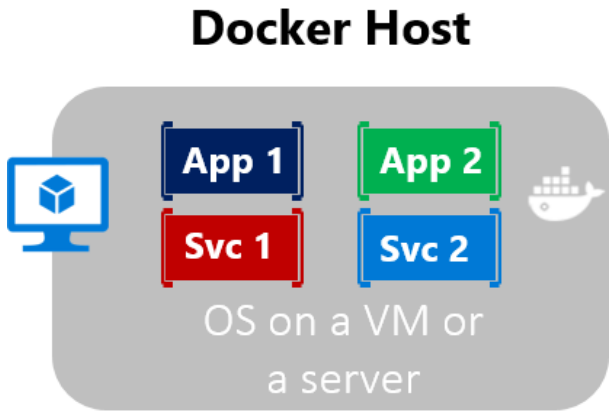
- Langage dédié
  - Jolie
- Plateformes d'exécution
  - **Docker**
  - Microsoft Service Fabric
  - MicroService4Net
  - NetKernel
- Plateformes de déploiement
  - VM dédiées
    - CoreOS, RancherOS , Ubuntu Snappy, Boot2Docker, docker-machine…
  - Cloud
    - NetFlix, Cloud Foundry, Amazon, MS Azure…



Conteneurisation : Définition

- La conteneurisation est une approche du développement logiciel dans laquelle une application ou un service, ses dépendances et sa configuration (matérialisé en tant que fichiers manifest de déploiement) sont regroupées sous la forme d'une image de conteneur.

- L'application conteneurisée peut être testée en tant qu'unité et déployée en tant qu'instance d'image de conteneur sur le système d'exploitation hôte.



Conteneurisation : Principe

- Analogie avec les conteneurs utilisés en commerce : permettent le transport de marchandises par bateau, train ou camion, quelle que soit la cargaison à l'intérieur.

- Les conteneurs de logiciels constituent une unité standard de déploiement de logiciels pouvant contenir différents codes et dépendances.

- Permet aux développeurs et aux professionnels de l'informatique de les déployer dans des environnements avec peu ou pas de modifications.

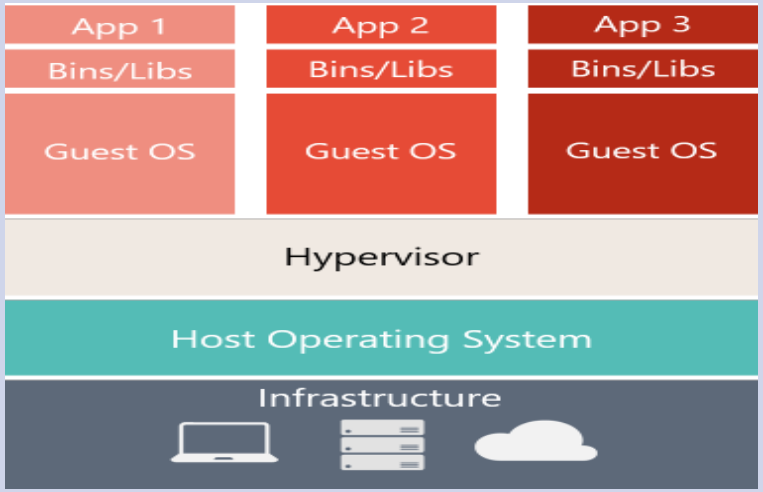
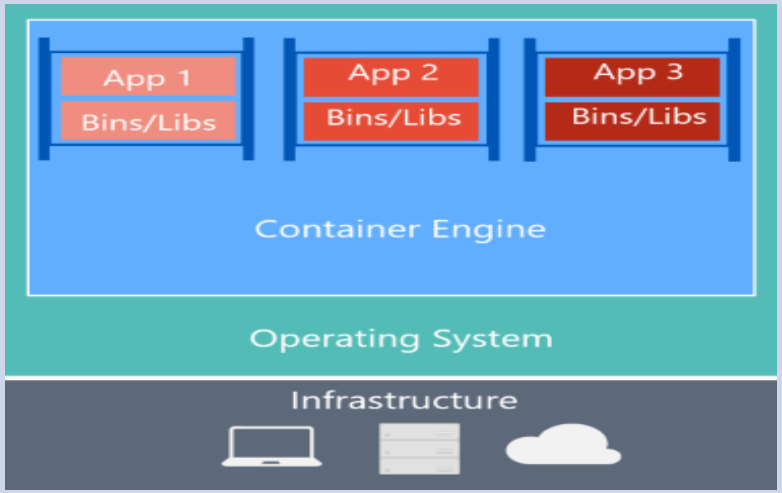
- Isolent les applications les unes des autres sur un système d'exploitation partagé.

- Les applications conteneurisées s'exécutent sur un conteneur hôte qui s'exécute à son tour sur le système d'exploitation. Les conteneurs ont donc une empreinte nettement plus petite que les images de machine virtuelle (VM).

Conteneurisation : Avantages

- Isolation de l'environnement fourni entre Dev et Ops.
- Portabilité.
- Agilité.
- Evolutivité.

Conteneurs Versus Machines Virtuelles

Virtual Machines	Docker Containers
	
Les machines virtuelles incluent l'application, les bibliothèques ou fichiers binaires requis et un système d'exploitation complet. La virtualisation complète nécessite plus de ressources que la conteneurisation.	Les conteneurs incluent l'application et toutes ses dépendances. Cependant, ils partagent le noyau du système d'exploitation avec d'autres conteneurs et s'exécutent en tant que processus isolés dans l'espace utilisateur du système d'exploitation hôte. (Sauf dans les conteneurs Hyper-V, où chaque conteneur s'exécute à l'intérieur d'une machine virtuelle spéciale par conteneur.)

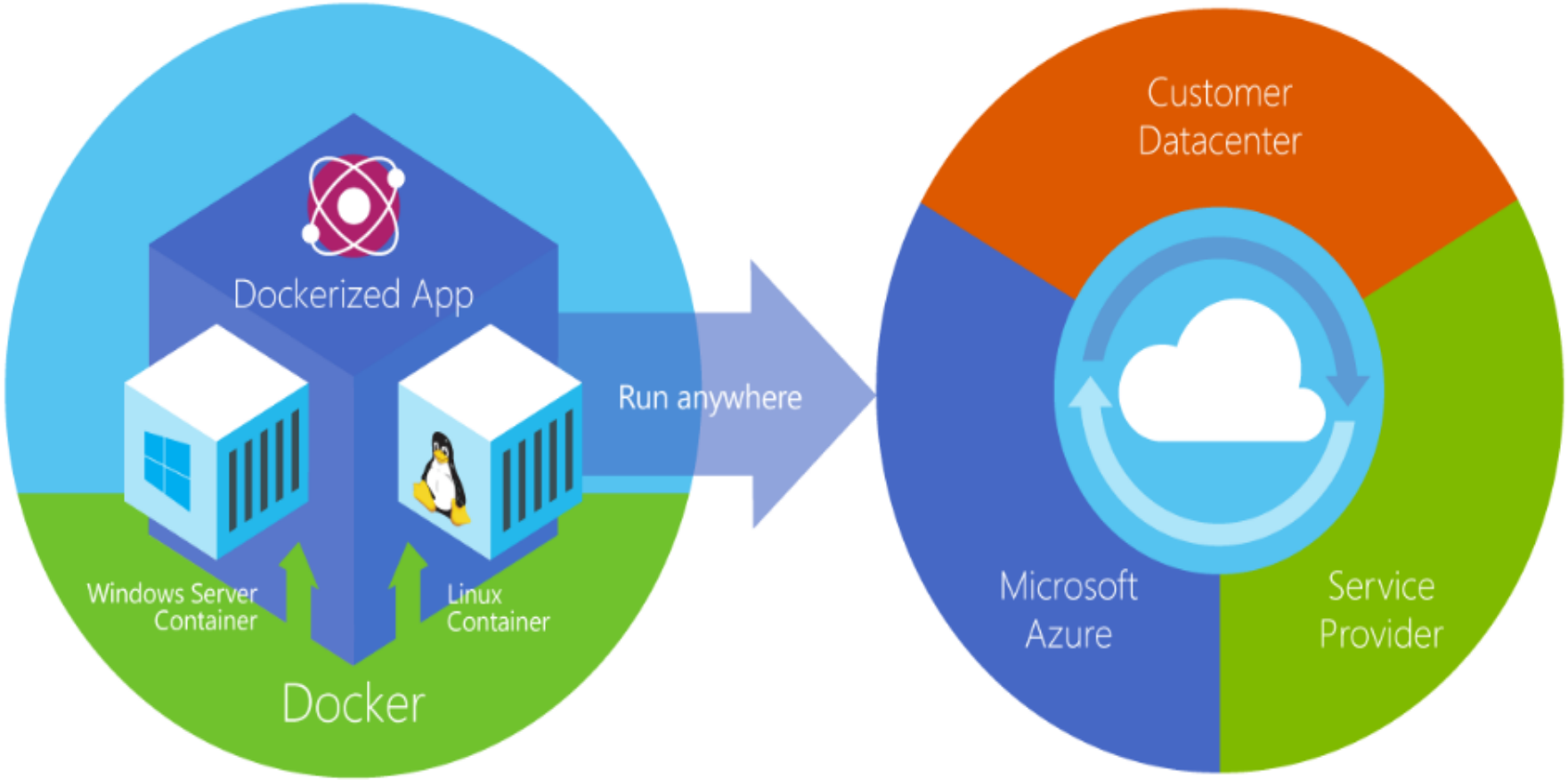
Conteneur Docker : Présentation

- Docker est un projet open-source permettant d'automatiser le déploiement d'applications sous la forme de conteneurs portables et autonomes pouvant s'exécuter sur le cloud ou sur serveur.

- « Docker est un outil qui peut empaqueter une application et ses dépendances dans un conteneur isolé, qui pourra être exécuté sur n'importe quel serveur ».

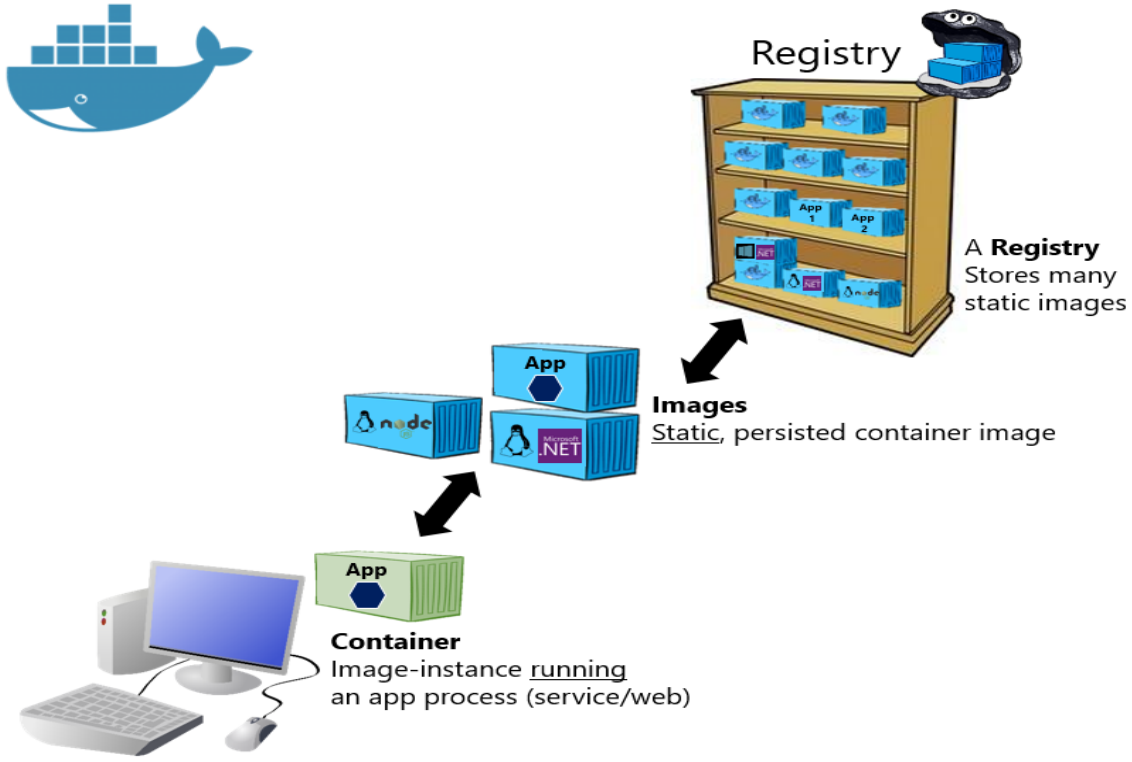
- Docker est également une entreprise qui promeut et fait évoluer cette technologie, en collaboration avec les fournisseurs de cloud, Linux et Windows, y compris Microsoft.

Conteneur Docker : Présentation



Conteneur Docker : Terminologie

Basic taxonomy in Docker



- **Hosted Docker Registry**
- **Docker Trusted Registry on-prem.**

→ **On-premises** (‘n’ private organizations)

- **Docker Hub Registry**
- Docker Trusted Registry on-cloud
- **Azure Container Registry**
- AWS Container Registry
- Google Container Registry
- Quay Registry
- Other Cloud

→ **Public Cloud** (specific vendors)

Conteneur Docker : Terminologie

- **Container image**/Image de conteneur:
  - un package avec toutes les dépendances et informations nécessaires pour créer un conteneur. Une image inclut toutes les dépendances (telles que les frameworks) ainsi que la configuration de déploiement et d'exécution à utiliser par un environnement d'exécution de conteneur. Habituellement, une image dérive de plusieurs images de base qui sont des couches empilées les unes sur les autres pour former le système de fichiers du conteneur. Une image est immuable une fois qu'elle a été créée.

- **Dockerfile**:
  - fichier texte contenant des instructions sur la création d'une image Docker. C'est comme un script batch, la première ligne indique l'image de base pour commencer, puis suivez les instructions pour installer les programmes requis, copier les fichiers, etc., jusqu'à ce que vous obteniez l'environnement de travail dont vous avez besoin.

- **Build**:
  - action de création d'une image de conteneur en fonction des informations et du contexte fournis par son Dockerfile, ainsi que des fichiers supplémentaires dans le dossier dans lequel l'image est créée. Vous pouvez créer des images avec la commande de construction Docker docker.

Conteneur Docker : Terminologie

- **Container**/Conteneur:
  - une instance d'une image Docker. Un conteneur représente l'exécution d'une seule application, processus ou service. Il se compose du contenu d'une image Docker, d'un environnement d'exécution et d'un ensemble d'instructions standard. Lors de la mise à l'échelle d'un service, vous créez plusieurs instances d'un conteneur à partir de la même image. Ou un batch job peut créer plusieurs conteneurs à partir de la même image, en transmettant différents paramètres à chaque instance.

- **Volumes**:
  - offrent un système de fichiers accessible en écriture que le conteneur peut utiliser. Puisque les images sont en lecture seule mais que la plupart des programmes ont besoin d'écrire sur le système de fichiers, les volumes ajoutent une couche inscriptible, au-dessus de l'image du conteneur, afin que les programmes aient accès à un système de fichiers inscriptible. Le programme ne sait pas qu'il accède à un système de fichiers en couches, c'est juste le système de fichiers comme d'habitude. Les volumes résident dans le système hôte et sont gérés par Docker.

- **Tag**/Balise:
  - marque ou étiquette appliquée aux images afin que différentes images ou versions de la même image (en fonction du numéro de version ou de l'environnement cible) puissent être identifiées.

Conteneur Docker : Terminologie

- **Multi-stage Build**:
  - est une fonctionnalité, depuis Docker 17.05 ou supérieur, qui aide à réduire la taille des images finales. En quelques phrases, avec la construction en plusieurs étapes, vous pouvez utiliser, par exemple, une grande image de base, contenant le SDK, pour compiler et publier l'application, puis utiliser le dossier de publication avec une petite image de base uniquement à l'exécution, pour produire un image finale beaucoup plus petite.

- **Repository (repo)**/Référentiel (référentiel):
  - une collection d'images Docker associées, étiquetées avec une balise indiquant la version de l'image. Certains dépôts contiennent plusieurs variantes d'une image spécifique, comme une image contenant des SDK (plus lourds), une image contenant uniquement des runtimes (plus légers), etc. Ces variantes peuvent être marquées avec des balises. Un référentiel unique peut contenir des variantes de plate-forme, telles qu'une image Linux et une image Windows.

- **Registry**/Registre:
  - un service qui permet d'accéder aux référentiels. Le registre par défaut pour la plupart des images publiques est Docker Hub (appartenant à Docker en tant qu'organisation). Un registre contient généralement des référentiels de plusieurs équipes. Les entreprises ont souvent des registres privés pour stocker et gérer les images qu'elles ont créées. Azure Container Registry est un autre exemple.

Conteneur Docker : Terminologie

- **Multi-arch image**:
  - Image multi-arcade: pour multi-architecture, est une fonctionnalité qui simplifie la sélection de l'image appropriée, en fonction de la plate-forme sur laquelle Docker est exécuté, par ex. lorsqu'un Dockerfile demande une image de base FROM mcr.microsoft.com/dotnet/core/sdk:2.2 du registre, il obtient en fait 2.2-sdk-nanoserver-1709, 2.2-sdk-nanoserver-1803, 2.2-sdk-nanoserver-1809 ou 2.2-sdk-stretch, selon le système d'exploitation et la version sur laquelle Docker est exécuté.

- **Docker Hub**:
  - un registre public pour télécharger des images et travailler avec elles. Docker Hub fournit l'hébergement d'images Docker, des registres publics ou privés, des déclencheurs de construction et des hooks Web, et une intégration avec GitHub et Bitbucket.

- **Azure Container Registry**:
  - une ressource publique pour travailler avec des images Docker et ses composants dans Azure. Cela fournit un registre qui est proche de vos déploiements dans Azure et qui vous donne un contrôle sur l'accès, ce qui permet d'utiliser vos groupes et autorisations Azure Active Directory.

Conteneur Docker : Terminologie

- **Docker Trusted Registry (DTR)**:
  - un service de registre Docker (de Docker) qui peut être installé sur site pour vivre dans le centre de données et le réseau de l'organisation. C'est pratique pour les images privées qui doivent être gérées au sein de l'entreprise. Docker Trusted Registry fait partie du produit Docker Datacenter. Pour plus d'informations.

- **Docker Community Edition (CE)**:
  - outils de développement pour Windows et macOS pour créer, exécuter et tester des conteneurs localement. Docker CE pour Windows fournit des environnements de développement pour les conteneurs Linux et Windows. L'hôte Linux Docker sur Windows est basé sur une machine virtuelle Hyper-V. L'hôte des conteneurs Windows est directement basé sur Windows. Docker CE pour Mac est basé sur le framework Apple Hypervisor et l'hyperviseur xhyve, qui fournit une machine virtuelle hôte Linux Docker sur Mac OS X. Docker CE pour Windows et pour Mac remplace Docker Toolbox, qui était basé sur Oracle VirtualBox.

- **Docker Enterprise Edition (EE)**:
  - une version à l'échelle de l'entreprise des outils Docker pour le développement Linux et Windows.

Conteneur Docker : Terminologie

- **Compose**:
  - un outil de ligne de commande et un format de fichier YAML avec des métadonnées pour définir et exécuter des applications multi-conteneurs. Vous définissez une seule application basée sur plusieurs images avec un ou plusieurs fichiers .yml qui peuvent remplacer des valeurs en fonction de l'environnement. Après avoir créé les définitions, vous pouvez déployer l'ensemble de l'application multi-conteneurs avec une seule commande (docker-compose up) qui crée un conteneur par image sur l'hôte Docker.

- **Cluster**:
  - ensemble d'hôtes Docker exposés comme s'il s'agissait d'un seul hôte Docker virtuel, afin que l'application puisse évoluer vers plusieurs instances des services réparties sur plusieurs hôtes au sein du cluster. Les clusters Docker peuvent être créés avec Kubernetes, Azure Service Fabric, Docker Swarm et Mesosphere DC / OS.

- **Orchestrator**:
  - un outil qui simplifie la gestion des clusters et des hôtes Docker. Les orchestrateurs vous permettent de gérer leurs images, conteneurs et hôtes via une interface de ligne de commande (CLI) ou une interface utilisateur graphique. Vous pouvez gérer la mise en réseau des conteneurs, les configurations, l'équilibrage de charge, la découverte de services, la haute disponibilité, la configuration de l'hôte Docker, etc. Un orchestrateur est responsable de l'exécution, de la distribution, de la mise à l'échelle et de la réparation des charges de travail sur un ensemble de nœuds. En règle générale, les produits d'orchestrateur sont les mêmes produits qui fournissent une infrastructure de cluster, comme Kubernetes et Azure Service Fabric, entre autres offres du marché.