DOCKER

1. PRESENTATION

Docker est un [logiciel libre](https://fr.wikipedia.org/wiki/Logiciel_libre) permettant facilement de lancer des applications dans des conteneurs logiciels. « 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 ». Il ne s'agit pas de [virtualisation](https://fr.wikipedia.org/wiki/Virtualisation), mais de [conteneurisation](https://fr.wikipedia.org/wiki/Conteneur_(informatique)), une forme plus légère qui s'appuie sur certaines parties de la machine hôte pour son fonctionnement. Cette approche permet d'accroître la flexibilité et la portabilité d’exécution d'une application, laquelle va pouvoir tourner de façon fiable et prédictible sur une grande variété de machines hôtes, que ce soit sur la machine locale, un cloud privé ou public, une machine nue, etc.

Techniquement, Docker étend le format de conteneur Linux standard, [LXC](https://fr.wikipedia.org/wiki/LXC), avec une [API](https://fr.wikipedia.org/wiki/Interface_de_programmation) de haut niveau fournissant une solution pratique de virtualisation qui exécute les processus de façon isolée. Pour arriver à ses fins, Docker utilise entre autre LXC, [groups](https://fr.wikipedia.org/wiki/Cgroups) et le [noyau Linux](https://fr.wikipedia.org/wiki/Noyau_Linux) lui-même. Contrairement aux machines virtuelles traditionnelles, un conteneur Docker n'inclut pas de système d'exploitation, mais s'appuie au contraire sur les fonctionnalités du système d’exploitation fournies par la machine hôte.

1. AVANTAGES

Nous avons plusieurs avantages à l’utilisation de Docker dans les domaines du Web mais aussi de l’IT. Les services IT proposent de plus en plus de services orientés web, le Docker pourrait être un élément clef pour accélérer leurs transitions vers le cloud.

* LÉGÈRETÉ DES CONTENEURS

Contrairement à un serveur virtuel sous Linux, le conteneur docker n’a besoin que de quelques centaines Mo de disques. L’empreinte mémoire est aussi réduite, car nous n’avons que de la mémoire utilisée pour l’application (pas de couche OS). Il est ainsi plus rapide à démarrer mais aussi à déplacer d’une machine à une autre.

* RAPIDITÉ ET FACILITÉ DE DÉPLOIEMENT DES APPLICATIONS

Après la récupération d’un template, qui se fait en une commande, il suffit d’une autre commande pour exécuter le conteneur docker, celui-ci démarre en quelques secondes. On peut ajouter des paramètres lors du démarrage qui seront transmis au docker. On va par exemple spécifier l’accès à la base de données ou d’autres paramètres d’automatisation.

L’idéal reste la mise en place d’une gestion de configuration, grâce à elle, vos conteneurs seront autonomes. On gère ses conteneurs comme on gère un package logiciel, cette facilité permet au développeur de pouvoir provisionner les environnements qui sont déployés en production par exemple sans connaissance particulière d’infrastructures ou d’administration système.

* PORTABILITÉ ET MULTICLOUD

Il y a peu de formats d’enveloppes virtuelles qui soient nativement multi-cloud. Il est toujours compliqué de récupérer sur sa ferme VMware son POC fait sur AWS, ou encore migrer sa machine virtuelle Openstack sur une Ressource Group Azure…

Avec Docker, vous pouvez, en quelques secondes, déployer votre test fait en local sur votre poste, en production sur la très grande majorité des Clouds (pour ne pas dire tous).

* ORIENTATION MICRO-SERVICES

Les nouvelles applications métiers, sites Web et nouvelles applications IT, se composent maintenant de micro-services, à l’inverse d’une application contenue dans un seul serveur et une seule application.

On va trouver des serveurs HTTP, des serveurs de cache, des répartiteurs de charge, des bases de données clés/valeurs, moteur d’API en nodejs….

Cette tendance répond à plusieurs objectifs :

* Améliorer l’élasticité d’une application,
* Augmenter la performance, chaque brique à un usage précis, dans laquelle elle est optimisée,
* L’architecture logicielle est aussi orientée microservices, le déploiement en continu est facilité car on n’impacte pas tout le code de l’application lors d’une modification, mais un seul module par exemple.

Bref on fonctionne de manière modulaire, l’application n’est plus monolithique et cela apporte de la flexibilité et réduit les risques.

Docker facilite grandement ce mode de fonctionnement, que cela soit dans le déploiement et le maintien en condition opérationnelle des briques de l’application, que pour le déploiement et le développement. Chaque microservice de l’application pourra être packagé dans un conteneur.

1. INCONVENIENTS

Même si Docker apporte beaucoup d’autonomie, les situations sont nombreuses dans lesquelles le besoin d’accompagnement par un prestataire d’infrastructures spécialisé se fera sentir :

* Dans le cas où le développeur n’a pas d’équipe infrastructures, il doit se débrouiller seul pour déployer sur une plateforme publique, et cela peut être moins simple que cela ne semble.
* Passé le cap du développement, on doit souvent sortir de l’environnement de conteneur pour revenir dans un environnement plus classique de production (IaaS, machine virtuelle, etc…).
* Enfin les équipes infrastructures internes d’une entreprise peuvent proposer ce type de service à leurs développeurs, mais rechercheront un accompagnement externe pour la phase de production.