



SaaS . IaaS . PaaS ...

As A Service ???

- l'expression « **As A Service** » marque la **différence** entre **informatique traditionnelle** et **Cloud Computing**
- historiquement, une entreprise gère ses matériels, OS, applications et données **en interne** : achat de licences, développements internes, installation, gestion du quotidien, maintenance ... **tout est fait au sein de l'entreprise**
- dans le **Cloud Computing**, tout est géré par un **fournisseur externe** qui met à disposition de l'entreprise les mêmes services (stockage, messagerie, applications ...)
- l'entreprise est **cliente** de ses propres ressources via Internet
- elle paie pour tous ces services (différentes formules)
- elle bénéficie de l'adaptation de ces services : **scalabilité**

As A Service ???

- ces services externes ont plusieurs **déclinaisons**
- SaaS, IaaS, PaaS, BPaaS, informatique sans serveur, FaaS, sont autant de **possibilités** offertes aux entreprises
- chaque offre a ses **avantages**
- ces offres **ne s'excluent pas**, elles se **complètent**
- la grande majorité des entreprises font appel à **plusieurs de ces offres simultanément**
- elles peuvent **coexister avec l'informatique traditionnelle**
- il n'y a pas de vérité absolue
- tout est question de **contexte**

SaaS

Software As A Service

Présentation

- le SaaS est un **service complet** : serveurs, stockage, réseau, applications métiers (standards ou spécifiques) **proposé par un prestataire Cloud**
- il met à disposition un **package d'applications Cloud prêtes à l'emploi**
- les utilisateurs (collaborateurs / clients) y accèdent via Internet et :
 - un PC, smartphone, tablette, IoT
 - un **simple navigateur Web**
 - une **application cliente** sur leur PC ou leur mobile

SaaS

Software
As A Service

Présentation

Le fournisseur garantit (**SLA** = **S**ervice **L**evel **A**greement) :

- la **maintenance** du matériel
- les **mise à jour** et **correctifs** des applications
- la **disponibilité** des services
- un niveau de **performances**
- la **sécurité** des connexions, la sécurité des données
- l'**évolution** des services selon les besoins de l'entreprise

Tout est **transparent** pour les utilisateurs

SaaS

Software
As A Service

Avantages

- **simplicité d'accès** : l'entreprise choisit un prestataire, crée un compte, choisit une offre et paie pour la prestation demandée
- **offres sans (trop de) risques** : période d'essai gratuite, formules de tests d'adéquation entre besoins et offres
- **offres facilement évolutives** : l'entreprise gère ses nouveaux besoins (à la hausse ou à la baisse) par un simple **Dashboard**
- **disponibilité des applications** : les utilisateurs travaillent en mode SaaS avec **n'importe quel périphérique connecté à Internet**

SaaS

Software
As A Service

Utilisation

Quelques exemples :

- réseaux sociaux
- **stockages Cloud** : tous les « Drive » et toutes les « Box »
- **messagerie** électronique
- logiciels **collaboratifs** : Jira, Google Workspace ...
- les grands éditeurs de logiciels **déclinent** progressivement en mode SaaS leurs applications initialement dédiées aux PC (éditeurs d'ERP et de CRM comme SAP, Salesforce, Oracle, Hubspot ...)



IaaS

Infrastructure As A Service

Présentation

- le fournisseur d'**IaaS** propose un **accès direct aux ressources physiques** : réseau, serveurs ...
- on accède soit :
 - au **matériel hôte** avec une solution de type **bare-metal** sur un hôte physique **dédié**
 - à des **machines virtuelles** hébergées sur un hôte physique **partagé**

IaaS

Infrastructure As A Service

Présentation

L'**administration** de ces ressources s'effectue :

- par l'intermédiaire de **tableaux de bord graphiques** (Dashboard sur navigateur Web ou application dédiée)
- par **programmation** via des **API**

Comme en SaaS, le fournisseur garantit (**SLA** = **S**ervice **L**evel **A**greement) :

- la **disponibilité** des infrastructures
- un niveau de **performance**
- la **sécurité** des connexions, la sécurité des données
- l'**évolution** des services selon les besoins de l'entreprise

IaaS

Infrastructure As A Service

Avantages

- **pas de fort investissement initial** d'infrastructure interne
- **pas d'investissements de complément** en cas de fortes variations d'utilisation (activités saisonnières)
- l'entreprise a le **contrôle du plus bas niveau des ressources**
- l'entreprise bénéficie d'une **plus grande souplesse** pour **créer des ressources** en fonction de ses besoins **par rapport à une infrastructure interne**
- elle peut **créer facilement des serveurs de redondance** pour renforcer son infrastructure et la **disponibilité** de ses applications
- la mise à disposition de ces nouveaux serveurs est **plus rapide que dans une infrastructure interne**

IaaS

Infrastructure As A Service

Avantages

- elle peut **choisir les emplacement** de ses serveurs pour :
 - répondre à des **contraintes légales**
 - **minimiser les risques** liés à des **zones géographiques plus sensibles** (géopolitique, catastrophes naturelles ...)
- l'**IaaS** diminue (élimine ?) les **risques de pannes** et **d'indisponibilités** plus fréquents avec une infrastructure interne grâce à la mise en place d'un **SLA**
- l'**IaaS** offre une **sécurité généralement supérieure** à celle d'une infrastructure interne

IaaS

Infrastructure As A Service

Utilisation

Start-up

- pour démarrer son activité, une start-up préférera une solution IaaS **souple et à moindre risque**, à un **investissement lourd en interne**

E-commerce

- c'est un secteur d'activité soumis à de **fortes fluctuations de charges et de trafic**
- l'IaaS :
 - répond parfaitement à cette contrainte de **fluctuations**
 - permet une **montée en charge rapide** (ou baisse)
 - propose une **sécurité adaptée** à la gestion de **milliers de comptes utilisateurs**
 - répond parfaitement à une problématique de **disponibilité de services 24/24, 7/7**

PaaS

Platform As A Service

Présentation

- le fournisseur de PaaS propose aux **développeurs d'applications** une **plateforme complète** : matériel, environnement de développement, logiciels de tests ...
- les développeurs ne sont **pas obligés de gérer les services**
- ils accèdent à la plateforme via une **interface graphique**
- ils créent, exécutent, testent leurs codes sur la plateforme
- un service de type PaaS utilise des **conteneurs** (Docker, Kubernetes) qui virtualisent l'environnement complet et permettent aux développeurs **d'exécuter ensuite leurs applications sur n'importe quelle plateforme**
- on parle alors de **CaaS = Container As A Service**

PaaS

Platform As A Service

Avantages

- **simplicité** de configuration, d'usage et de maintenance par rapport à une plateforme de développement en interne
- **création rapide d'environnements** de développement : quelques heures seulement au lieu de quelques jours en interne
- **développement Agile et DevOps** : un service de type **PaaS** correspond exactement à ces deux concepts → souplesse, réactivité, intégration continue et automatisée (**CI / CD**)
- **accès aux technologies les plus récentes** : une **PaaS** prend en charge les développements en mode **Cloud Natif**, la **conteneurisation**, les **dernières versions des langages**, notamment ceux utilisés pour les applications **IoT** ... **sans avoir à investir massivement**

Autres modèles

BPaaS : Business Process As A Service

- c'est un service Cloud qui concerne uniquement les processus « métier » : gestion commerciale, gestion financière, ressources humaines ...
- il s'agit généralement d'un SaaS ou d'une combinaison SaaS et IaaS

Informatique sans serveur

- c'est un modèle de cloud computing où le fournisseur gère intégralement le **backend** (serveur, bases de données, répliquions, sauvegardes, optimisation des bases ...)
- les développeurs se consacrent **uniquement au code**

Autres modèles

FaaS : Function As A Service (Fonction sous forme de service)

- c'est une déclinaison d'un **SaaS** ou d'une **PaaS**
- le développeur exécute uniquement des parties (des **fonctions**) du code global, mais pas l'application complète
- le fournisseur de **FaaS** mesure précisément ce que la fonction exécutée consomme comme ressources
- il ne facture donc que ce qui se situe entre **le début de l'exécution de la fonction** et **l'arrêt de l'exécution**
- à l'extrême, si vous codez sans jamais tester, les coûts sont quasiment nuls