Sommaire

[Introduction 3](#_Toc360607770)

[1. Terminologies cloud 3](#_Toc360607771)

[a. Modèles de cloud 3](#_Toc360607772)

[Le cloud privé 3](#_Toc360607773)

[Le cloud publique 3](#_Toc360607774)

[Le cloud hybride 4](#_Toc360607775)

[Le cloud communautaire 4](#_Toc360607776)

[b. Caractéristiques d’une offre cloud 4](#_Toc360607777)

[Libre-service à la demande 4](#_Toc360607778)

[L’offre cloud doit être accessible sur l’ensemble d’un réseau 4](#_Toc360607779)

[Les ressources sont mutualisées 4](#_Toc360607780)

[Le cloud doit s’adapter rapidement aux différentes variations des besoins lors de l’utilisation des ressources 4](#_Toc360607781)

[Le service doit être mesurable 4](#_Toc360607782)

[2. Couches principales du cloud 5](#_Toc360607783)

[a. Infrastructure-as-a-Service (IaaS) 5](#_Toc360607784)

[b. Plateforme-as-a-Service (PaaS) 5](#_Toc360607785)

[i. Application Plateforme-as-a-Service (APaaS) 6](#_Toc360607786)

[c. Software-as-a-Service (SaaS) 6](#_Toc360607787)

[3. Présentation de certains leader PaaS du marché 6](#_Toc360607788)

[a. Heroku de SalesForces 6](#_Toc360607789)

[b. CloudFoundry 7](#_Toc360607790)

[c. CloudBees 8](#_Toc360607791)

[d. OpenShift Online de Red Hat 9](#_Toc360607792)

[e. Windows Azure de Microsoft 9](#_Toc360607793)

[f. Amazon Elastic Compute Cloud (Amazon EC2) 10](#_Toc360607794)

[g. Engine Yard Cloud 10](#_Toc360607795)

[h. CloudControl 11](#_Toc360607796)

[i. AppHarbor 12](#_Toc360607797)

[j. Cloudify de Gigaspace 12](#_Toc360607798)

[k. Static 13](#_Toc360607799)

[4. Comparaison de PaaS 14](#_Toc360607800)

[a. Caractéristiques générales 14](#_Toc360607801)

[b. Comparaison par les Framework supportés 16](#_Toc360607802)

[c. Comparaison par type de modules complémentaire disponible en extension 18](#_Toc360607803)

[d. Comparaison par les cas d’usage réalisés. 18](#_Toc360607804)

[e. Intérêts 21](#_Toc360607805)

[Conclusion 22](#_Toc360607806)

# Introduction

Le cloud computing décrit un environnement informatique composé de ressources de calcul mutualisées et distribuées basée sur la demande client. Il peut être utilisé tout type d’utilisateur.

Les services cloud sont utilisés pour traiter une variété d’applications et de besoins en infrastructures tels que les bases de données, les espaces de stockage, capacité de calcul… Cet environnement a la propriété d’être mise à disposition du consommateur en quelques minutes.

Notre étude porte sur la comparaison de certaines offres services de cloud computing. Nous avons ainsi ciblé parmi les leaders du marché, des fournisseurs permettant d’avoir les ressources nécessaires pour un environnement supportant tout le cycle de vie d’une application allant de la spécification des besoins à la livraison, et la maintenance.

# Terminologies cloud

Le cloud computing est un modèle pour permettre l’accès réseau à la demande de ressources configurables partagées omniprésentes (réseau, espace mémoire, applications, et services), qui peuvent être rapidement fournies et libérées avec un effort minimal de gestion ou d’interaction avec le fournisseur. Ce modèle est composé de modèles de déploiement, de caractéristiques essentielles et de modèles de services [(les différentes couches](#_Couches_principales_du)).

## Modèles de cloud

Il existe quatre modèles de déploiement cloud.

### Le cloud privé

L’infrastructure cloud est fournie en tant que service pour un usage exclusif par une seule organisation comprenant plusieurs clients. Ce cloud pourrait être possédé, géré, et exploité par cette même organisation, une tierce partie, ou une de leurs combinaisons, et pourrait exister à l’intérieur ou en dehors des locaux du fournisseur.

**Avantages**

* + Utilisation totalement personnalisée et accrue des ressources
  + Scalabilité et réallocation rapide à la demande de changement
  + Plus de sécurité avec le respect des standards de l’organisation
  + Bonne productivité des équipes de développement

**Inconvénients**

* Requiert la présence d’administrateurs

### Le cloud publique

L’infrastructure cloud est fournie en tant que service pour un usage libre par le publique. Ce cloud est possédé, géré, et exploité par une entreprise, une organisation académique ou gouvernementale, ou une de leurs combinaisons. Pourrait être en local ou à l’extérieur des locaux du fournisseur cloud.

**Avantages**

* + Accès rapide aux ressources informatiques
  + Pas d’investissement en infrastructure
  + Ne payer que ce l’on consomme
  + Baisse le coût des infrastructures, de la consommation en énergies et de l’installation
  + Meilleur productivité des équipes de développement

**Inconvénients**

* Le client s’occupe lui-même du respect de ses normes de sécurité
* Utilisation des ressources moins personnalisable qu’un cloud privé
* L’organisation n’obtient pas forcément l’environnement de travail voulu chez un seul fournisseur cloud, ce qui le pousse à combiner les offres de différents fournisseurs cloud

### Le cloud hybride

C’est une composition d’au moins deux modèles de déploiement cloud distinctes (privée, communautaire ou publique) qui demeurent des entités uniques, mais sont liées ensembles par des ressources technologies standardisées ou propriétaires qui permettent la portabilité des données et des applications.

### Le cloud communautaire

L’infrastructure cloud est fournie pour un usage exclusif par une communauté spécifique de consommateurs partageant les mêmes préoccupations (exigence en sécurité, mission…). Ce cloud devrait être possédé, géré, et exploité par une ou plusieurs organisations dans la communauté, une tierce partie, ou une de leurs combinaisons, et pourrait être dans les locaux ou à l’extérieur.

## Caractéristiques d’une offre cloud

Une offre cloud présente certaines caractéristiques essentielles.

### Libre-service à la demande

Un client peut unilatéralement augmenter les capacités de calculs de l’offre clou, tel que le temps d’exécution du serveur, l’utilisation du stockage réseau etc., de façon automatique selon les besoins et un catalogue prédéfinis, sans qu’il y ait forcément une intervention humaine avec le fournisseur.

### L’offre cloud doit être accessible sur l’ensemble d’un réseau

Les services sont disponibles sur le réseau et accédées à travers des mécanismes standards qui favorisent l’utilisation par des plateformes clientes hétérogènes (poste de travail, tablettes…).

### Les ressources sont mutualisées

Les ressources fournies sont regroupées pour desservir plusieurs consommateurs utilisant un modèle multi-locataire, avec différentes ressources physiques, et virtuelles assignées et libérées dynamiquement selon la demande du consommateur. Le client n’a aucun contrôle ou connaissance sur le lieu exact des ressources fournies mais peut généralement spécifier les emplacements à un autre niveau d’abstraction (pays, lieu du Datacenter…). Ces ressources incluent : la mémoire, la taille de stockage, la capacité de calcul et la bande passante. Le but de cette caractéristique est de gagner en scalabilité.

### Le cloud doit s’adapter rapidement aux différentes variations des besoins lors de l’utilisation des ressources

Les ressources peuvent être fournies de façon élastique, dans certains cas automatiquement, pour s’adapter à la demande. Pour le consommateur, les capacités disponibles pour l’approvisionnement semblent souvent être illimitées et peuvent être appropriées dans n’importe quelle quantité à tout moment.

### Le service doit être mesurable

Le système cloud contrôle et optimise automatiquement la consommation des ressources en s'appuyant sur une capacité de mesure à un certain niveau d'abstraction approprié au type de service (par exemple, le stockage, le traitement, la bande passante et comptes utilisateurs actifs). L'utilisation des ressources peut être surveillée, contrôlée, et rapportée.

# Couches principales du cloud

Le cloud est structuré en trois couches principales.

**SaaS**

**PaaS**

**IaaS**

## Infrastructure-as-a-Service (IaaS)

Infrastructure en tant que services est une infrastructure informatique consommée en tant que service. Chaque utilisateur ou locataire accède à une portion de ressources fédérées pour créer et utiliser sa propre infrastructure au besoin, quand et comment il le veut.

Ce qui est offert au consommateur, c’est la capacité à approvisionner le traitement, le stockage, les réseaux, et autre ressources fondamentales où le consommateur est capable de déployer et exécuter un logiciel qui peut inclure des systèmes d’exploitations, ou applications. Le consommateur ne gère ni ne contrôle l’infrastructure cloud sous-jacente, mais a le contrôle sur les systèmes d’exploitations, le stockage, et les applications déployées ; et éventuellement un contrôle limité sur les composants réseaux.

L'IaaS est le premier modèle de cloud, où :

* **Le fournisseur Cloud** maintient : la virtualisation, le matériel serveur, capacité de stockage et capacité de calcul à la demande sur un réseau;
* **L'entreprise cliente** maintient : les applications, les environnements de développement et d'exécution d'applications (incluant systèmes d’exploitation), l'intégration SOA, les bases de données, le logiciel serveur.

L’IaaS est le modèle adapté aux entreprises souhaitant utiliser leur propre cloud sans avoir à supporter des investissements en infrastructure. Le client IaaS peut être un fournisseur PaaS. Cette couche donne la possibilité au consommateur d’avoir un contrôle total sur l’environnement qu’il veut fournir, mais nécessite plus de travail (administration, gestion…)

## Plateforme-as-a-Service (PaaS)

**Plateforme en tant que service (PaaS)** est un environnement de computing accédé au besoin sur un réseau à partir d’un fournisseur de service. PaaS est utilisé pour développer et exécuter des logiciels comme alternative à l’architecture, la construction et l’installation d’un environnement local de développement et de production.

Un client utilise la plateforme PaaS pour porter ses applications dans le cloud ou en développer d'autres à partir des outils de développement fournis par la plate-forme de Cloud Computing, sans se soucier du coût et de la complexité de l’achat et de la gestion du matériel sous-jacent, des logiciels et utilitaires.

PaaS fournit toute la facilité requise pour supporter le cycle de vie complet d’une application, de sa construction à la livraison.

La plateforme PaaS est le second modèle de cloud, où :

* **Le fournisseur Cloud** maintient : les applications, les environnements de développement et d'exécution d'applications (incluant systèmes d’exploitation), l'intégration SOA, les bases de données, le logiciel serveur, la virtualisation, le matériel serveur, le stockage, les réseaux. Exemple de fournisseur : OpenShift qui propose une suite de logicielle Linux+Apache+MySQL+Java;
* **L'entreprise cliente** maintient uniquement les applications qu’il déploie au travers des services, librairies, langages fournis. Exemple de services fournis: base de données MySQL, serveur JBoss.

Le PaaS est le modèle adapté aux entreprises souhaitant contrôler le déploiement de ses applications ou qui veulent développer leurs propres applications. Le client ici fournit l’application finale. Cette couche permet d’une part de gagner en temps de développement, en termes de coût des infrastructures et de consommation en ressource (pour une meilleure scalabilité). Dans une autre part, les équipes de développement ont moins de contrôle sur leurs environnements de travail.

### Application Plateforme-as-a-Service (APaaS)

**APaaS** pour application Platform-as-a-Service : environnement de développement plus facile à apprendre et utiliser que le PaaS, délivrant des applications paramétrables. L'APaaS fournit des services intégrés de développement, déploiement et gestion, optimisés pour un environnement de cloud computing et prenant en charge la fourniture de l'application finale qui est distribué comme un service (SaaS).

## Software-as-a-Service (SaaS)

**Software-as-a-service (SaaS**) est un logiciel utilisé sur un réseau sans qu’il soit téléchargé et installé dans un environnement local. Le logiciel est accédé par d’internet (avec un navigateur web ou autre programme interfacé) à partir d’un fournisseur SaaS et est exécutée dans un environnement prédéfini par ce même fournisseur. Le consommateur ne gère ni ne contrôle l’infrastructure cloud sous-jacente.

SaaS pour « Services-as-a-Software » est l'ultime modèle de cloud, où :

* **Le fournisseur Cloud** maintient : les applications qu’il déploie, les environnements de développement et d'exécution d'applications (incluant systèmes d’exploitation), l'intégration SOA, les bases de données, le logiciel serveur, la virtualisation, le matériel serveur, le stockage, les réseaux ;
* **Le client** accède aux applications et services via un navigateur internet. Par exemple, ce peut être une banque qui loue un logiciel de comptabilité en ligne, à la demande.

# Présentation de certains leader PaaS du marché

## Heroku de SalesForces

Heroku a été fondée en  2007 rachetée en 2010 par Salesforces. Heroku étend alors son offre  au développement Java. C’est une plateforme de d’application cloud initialement prévue pour les langages Ruby, Python, Scala et Clojure. La zone Europe est disponible depuis avril 2013 sur le Datacenter d’Amazon situé en Irlande; ce qui permet un gain de 100ms par requête par rapport à la zone US.

La philosophie de Heroku, c’est d’augmenter la productivité du développeur en minimisant le nombre d’étapes, en le permettant de passer moins de temps dans la gestion des serveurs d’applications, le déploiement ou la configuration de la scalabilité. Heroku est une plate-forme PaaS pour construire, déployer et exécuter des applications sur cloud. L’architecture de la plate-forme inclut des outils pour le déploiement et la gestion (plugin et SDK pour l'exécution en ligne de commande), un environnement d'exécution ainsi que des modules complémentaires.

|  |  |
| --- | --- |
| Type de cloud | Fournisseurs d’infrastructure |
| Publique | AWS |
| Fonctionnalités | **Interfaces** |
| * Déploiement * Connexion aux services * Contrôle de l’application et des ressources * Collaboration * Journalisation des évènements de l’application, de la plateforme, des services… * Configuration de la scalabilité | * CLI * Plugin IDE * Console web * API REST full |
| Frameworks | **Services complémentaires** |
| Rails, Express, Play, Sinatra, Spring, Jetty, Django, Flask | Base de données, outils de recherche, mail et SMS, worker et queuing, analyse de données, analyse de performance, caching, outils de facturation, journalisation et autres utilitaires |
| Avantages | **Inconvénients** |
| * Un large choix de modules complémentaires ; * Possibilité d’intégration à un IDE. | * Il n’y a pas de conteneurs ou serveurs d’applications, ces dernières sont exécutées par des processus configurés dans des fichiers Procfile. * Voir les limites [\*](https://devcenter.heroku.com/articles/limits) |

## CloudFoundry

Cloud Foundry est une plate-forme PaaS libre, au départ développée par VMware sous les termes de la licence Apache License 2.0. Elle est développée en [Ruby](http://www.ohloh.net/p/cloudfoundry) et actuellement maintenue par l’entreprise [Pivotal](http://www.gopivotal.com). Cloud Foundry fournis un choix de clouds, de Frameworks de développement et de services pour les applications. Cloud Foundry rend simple la construction, le test, le déploiement et la mise à l’échelle d’une application. C’est un projet open source et est accessible à travers [une variété de distributions clouds](https://core.cloudfoundry.org/listings) privées incluant [CloudFoundry.com](http://cloudfoundry.com).

Cloud Foundry permet au développeur de plus se concentrer sur l’implémentation des applications que sur l’infrastructure. Cette PaaS permet d’écrire des applications portables, tester, adapter, déployer sans change le code source. CloudFoundry est un socle à partir duquel d’autres fournisseurs se basent pour offrir des services PaaS plus Elargis. Il s’agit de :

* Static
* TIER 3
* Uhuru
* NTT Communications
* Appfog
* MoPaaS

|  |  |
| --- | --- |
| Type de cloud | Fournisseurs d’infrastructure |
| Publique, privé, hybride | * Par défaut : * Autres partenaire : vSphere, AWS, OpenStack, Rackspace |
| Fonctionnalités | **Interfaces** |
| * Déploiement * Gestion du cycle de vie d’une application * Surveillance des applications * Gestion d’identité * Collaboration pour le développement * Accès au système par avec des API REST | * CLI * Plugin IDE * Console web * API et librairies |
| Frameworks | **Services complémentaires** |
| Grails, Java Web, Lift, Node.js, Play, Rack, Sinatra, Spring, Standalone | Base de données, outils de recherche, queuing, analyse de données, analyse de performance |
| Avantages | **Inconvénients** |
| * Possibilité d’intégration à un IDE * Possibilité d’ajouter plusieurs environnements d’exécution pour une même application * Il existe plusieurs fournisseurs CloudFoundry pour étendre les services et fonctionnalités offertes | * Très peu de modules complémentaires * La console web n’offre pas large contrôle sur les applications, services et sur les utilisateurs * Pas de gestionnaire de code source * Pas de sauvegarde d’application |

## CloudBees

La plateforme PaaS  a été créée en 2010; l’équipe est actuellement composée d’un ancien directeur technique de JBoss de Kohsuke Kawaguchi, le fondateur et développeur principal de Jenkins CI et de travailleurs aux USA, en Europe, et en Australie. La vision de CloudBees est d’éviter au  développeur les tâches de maintenances des infrastructures ou mises à jour logicielles, lui permettant ainsi de mieux se concentrer sur l’implémentation de son application. CloudBees offre une plate-forme qui englobe la totalité du cycle de vie d’une application allant du développement à la mise en production.

|  |  |
| --- | --- |
| Type de cloud | Fournisseurs d’infrastructure |
| Publique, privé |  |
| Fonctionnalités | **Interfaces** |
| * Création d’application * Intégration continue avec Jenkins * Notification d’évènements de diverses sources * Contrôle de la scalabilité * Gestion de code source * Choix du fournisseur cloud * Surveillance d’applications * Intégrations de services externes | * CLI * Console Web * Plugin Eclipse * API basée sur HTTP |
| Frameworks | **Services complémentaires** |
| Tomcat 7, Hibernate, Java EE 6 Web Profile, Glassfish, Jetty, JBoss 7, Akka, Liferay, Grails, iU Web Apps, SOASTA CloudTest, MacOS executors, Google Go Webapp, Facebook App, Drupal, Play, Lift, Webmachine, Dart App, Clojure/Compojure, Node.js, Metor.js, Angular.js, Backbone.js, JAX RS, JRuby on Rails, Railo | Analyse de performance, message queuing, ALM, gestionnaire de dépôts, base de données SQL et NoSQL, journalisation, analyse continue de code source, moteur de recherche, mail, wiki, IDE partenaires, mémoire cache, analyse de données Big Data, déploiement continu (CF, GAE, Codenvy). |
| Avantages | **Inconvénients** |
| * Possibilité d’intégration à un IDE. * Une variété de modules complémentaire * Large choix de Framework, * Choix du fournisseur d’infrastructure * Console web offrant un contrôle étendu des applications, des services et des utilisateurs * No vendor lock-in |  |

## OpenShift Online de Red Hat

OpenShift Online est l’offre cloud de Red Hat. C’est une plateforme Open source où les développeurs peuvent construire, tester, déployer et exécuter leurs applications. Elle fournit également des outils intégrés pour le développement pour supporter le cycle de vie d’une application; il s’agit de l’intégration à Eclipse, Jenkins, Maven et Git. La plateforme fournit une scalabilité manuelle ou automatique des ressources d’une application.

|  |  |
| --- | --- |
| Type de cloud | Fournisseurs d’infrastructure |
|  |  |
| Fonctionnalités | **Interfaces** |
| * Variables d’environnement * Exécution de tâches * Surveillance d’applications * Gestion des services * Sauvegarde des applications * Environnements multiples * Intégration continue avec Jenkins * Gestion de code source * Scalabilité automatique ou manuelle | * CLI * Console Web * Plugin IDE |
| Frameworks | **Services complémentaires** |
| JBoss, Spring, Tomcat, Zend, Codelgniter, Rails, Node.js, Django, Flask, Perl, DIY (Do-It-Yourself) | Base de données SQL et NoSQL, analyse de données Big Data, gestionnaire de base de données, intégration continue, mail, exécution de tâches, analyse de la charge, collaboration, gestionnaires de code sources |
| Avantages | **Inconvénients** |
| * Support technique de la communauté actif |  |

## Windows Azure de Microsoft

Windows Azure est une plateforme Cloud ouverte et flexible qui permet de générer, déployer et gérer rapidement des applications à travers un réseau global de centres de données gérés par Microsoft. On peut développer des applications à l'aide de n'importe quel langage, n'importe quel outil ou n'importe quelle infrastructure, ou intégrer des applications de Cloud publique à son propre environnement informatique existant.

|  |  |
| --- | --- |
| Type de cloud | Fournisseurs d’infrastructure |
| Publique | Microsoft |
| Fonctionnalités | **Interfaces** |
| * Gestion de données * Networking * Analyse de données * Message queuing * Caching * Media Services * Marketplace * Big Compute | * CLI * Console Web * Plugin IDE * API |
| Langages et Frameworks | **Services complémentaires** |
| .NET, Java, PHP, Python, Node.js, Ruby | Gestion de certificat, Caching, email et voix, SMS, traitement de fichier media, message queuing, surveillance et gestion d’applications |
| Avantages | **Inconvénients** |
| * De nombreuses API | * Applications non portables sur d’autres plateformes |

## Amazon Elastic Compute Cloud (Amazon EC2)

Amazon EC2 est un service web qui fournit une capacité de calcul redimensionnable dans le cloud. Il est conçu pour faciliter l’accès aux ressources informatiques pour les développeurs. L’interface web permet fournit un contrôle complet de ces ressources permet d’exécuter une application. Amazon EC2 présente un environnement informatique virtuel, permettant d'utiliser des interfaces de service Web pour lancer des instances avec une variété de systèmes d'exploitation, de les charger avec un environnement d'applications personnalisées, de gérer les autorisations d'accès au réseau, et d'exécuter une image en utilisant autant ou aussi peu de systèmes.

|  |  |
| --- | --- |
| Type de cloud | Fournisseurs d’infrastructure |
| Publique | Amazon |
| Fonctionnalités | **Interfaces** |
| * Contrôle total des instances des machines virtuelles * Flexibilité dans le choix du système d’exploitation * Contrôle de la capacité de calcul * Interopérabilité avec d’autres services Amazon Web Services * Choix de l’emplacement des données | * Console Web * Plugin IDE |
| Frameworks | **Services complémentaires** |
|  |  |
| Avantages | **Inconvénients** |
| * On peut tout faire avec puisqu’on manipule une instance d’un système d’exploitation | * Manipulation d’instances EC2 * Le développeur doit configurer tout son environnement |

## Engine Yard Cloud

Engine Yard Cloud est un PaaS conçu pour les applications Ruby on Rail, PHP et Node.js. Cette plateforme exécute aussi bien des applications web à petite échelle qui requièrent une seule instance que des applications à grande échelle qui requièrent de la fiabilité, de l’élasticité et la performance de cluster hautement disponible.

|  |  |
| --- | --- |
| Type de cloud | Fournisseurs d’infrastructure |
| Publique | Amazon |
| Fonctionnalités | **Interfaces** |
| * Intégration continue * Ressources dédiées * Déploiement d’applications * Gestion et configuration d’applications * Réplication de base de données * Surveillance et alerte * Clonage d’applications * Sauvegarde et restauration d’applications | * Console Web * CLI * API |
| Frameworks | **Services complémentaires** |
| Grape, Rails, Sinatra | Mail, surveillance et analyse de performance, intégration continue, téléphonie, modules de paiement, test de la montée en charge, journalisation, bases de données SQL et NoSQL, gestionnaires de base de données, message queuing, analyse de données Big Data, mémoire cache |
| Avantages | **Inconvénients** |
| * Possibilité de créer plusieurs environnements de développement pour une même application * Grand nombre de modules complémentaires * Choix de la région de développement | * Très peu de langages * Pas d’intégration aux IDE |

## CloudControl

CloudControl est une plateforme en tant que service supportant divers langage ainsi que leurs écosystèmes tout en restant portables. C’est une solution PaaS pour le cycle de vie complet d’une application et permet de se décharger de la configuration de l’infrastructure.

|  |  |
| --- | --- |
| Type de cloud | Fournisseurs d’infrastructure |
| Publique, privé, hybride | AWS Irlande, ou alternativement dans le Datacenter souhaité |
| Fonctionnalités | **Interfaces** |
| * Gestion d’utilisateur * Déploiement d’applications * Gestion de code source * Sauvegarde et restauration d’applications * Environnements multiples * Journalisation * Scalabilité * Analyse de performance * Exécution de tâche | * CLI * Console Web * API |
| Frameworks | **Services complémentaires** |
| Symphony, Code Igniter, Flask, CakePHP, Django, Zend Framework, Joomla, Rails, Sinatra, Java, Python | Analyse de données Big Data, message queuing, moteur de recherche, exécution de tâche, bases de données SQL et NoSQL, mémoire cache, intégration continue, journalisation, encryptage SSL, SMS, Mail, analyse de performance, analyse de la montée en charge |
| Avantages | **Inconvénients** |
|  | * Pas d’intégration IDE |

## AppHarbor

AppHarbor est une solution PaaS entièrement .NET. AppHarbor peut déployer et mettre à l’échelle, toute application .NET sur le cloud. Cette PaaS est utilisée par les développeurs et entreprises pour héberger n’importe qu’elle type d’application, allant du blog aux applications web dont le trafic est élevé.

|  |  |
| --- | --- |
| Type de cloud | Fournisseurs d’infrastructure |
| Publique | AWS à l’est des US |
| Fonctionnalités | **Interfaces** |
| * Déploiement * Surveillance * Gestion de la montée en charge * Gestion de la scalabilité * Gestion de domaines | * CLI * Console Web * API |
| Frameworks | **Services complémentaires** |
| .NET | Surveillance d’exceptions, analyse d’application ASP .NET et JavaScript, traitement d’image, analyse de performance, message queuing, mail, bases de données SQL et NoSQL, moteur de recherche, test de la montée en charge, journalisation, mémoire cache, gestionnaire de base de données, gestionnaire de code source |
| Avantages | **Inconvénients** |
|  | * Que le Framework .NET |

## Cloudify de Gigaspace

Cloudify est une solution PaaS Open Source téléchargeable gratuitement conçue pour toute application sur tout type de cloud publique, privé ou hybride. Cette plateforme permet de déployer rapidement et de gérer des applications sur n’importe quel cloud sans changement de code source ou d’architecture.

|  |  |
| --- | --- |
| Type de cloud | Fournisseurs d’infrastructure |
| Publique, privé, hybride | AWS, OpenStack, Rackspace, Windows Azure, HP Cloud VMware, Citrix Cloud Stack1, Terramark, XenServe |
| Fonctionnalités | **Interfaces** |
| * Surveillance d’application * Envoi d’alertes et évènement * Création de machines virtuelles * Remplacement automatique des instances arrêtées * Scalabilité automatique * Déploiement sur plusieurs plateformes * Automatisation du cycle de vie | * REST API * CLI * Console web |
| Frameworks | **Services complémentaires** |
| Spring, Java EE, Ruby on Rails, Drupal, Play | Bases de données SQL et NoSQL, mémoire cache, moteur de recherche de données |
| Avantages | **Inconvénients** |
| * Application portables vers d’autres plateformes PaaS * Cloud local | * Pour utiliser la plateforme, il faut la télécharger et l’installer |

## Static

Static Corporation a été fondé en 2005 et s’appelait alors ONESite. Static est une plateforme qui se base sur le cœur de CloudFoundry. La plateforme Open Source est conçue pour les entreprises et développeurs voulant offrir une solution cloud d’hébergement à leurs clients. Static est également installable.

|  |  |
| --- | --- |
| Type de cloud | Fournisseurs d’infrastructure |
| Publique et privé | Static, AWS, Rackspace, DigitalOcean |
| Fonctionnalités | **Interfaces** |
|  | * Console web |
| Frameworks | **Services complémentaires** |
| Bootstrap, Django, Express, Flask, FuelPHP, Grails, Kohana, Lavarel, Lift, Play, Rack, Rails, Sinatra, Spring, Symphony, Tomcat, Zend, WSGI | Bases de données SQL et NoSQL, message queuing, moteur de recherche de données, outils de partage de connaissance (wiki), gestionnaire d’artefacts, gestionnaire de bases de données |
| Avantages | **Inconvénients** |
|  |  |

# Comparaison de PaaS

## Caractéristiques générales

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Critères de comparaison | Fournisseur Cloud PaaS | | | | | | | | | | |
|  | Heroku | CloudFoundry | CloudBees | OpenShift | Azure | AWS EC2 | Engine Yard Cloud | CloudControl | AppHarbor | Cloudify | Static |
| Fournisseur | SalesForces | Pivotal | CloudBees | Red Hat | Microsoft | Amazon | Engine Yard | CloudControl |  | Gigaspace | Static |
| Type de logiciel | Open Source | Open Source |  | Open Source | Propriétaire |  | Open Source | Open et Propriétaire | Propriétaire | Open Source dans la majorité des composants | Open Source |
| Système d’exploitation | Ubuntu 10.04 | Ubuntu |  | Red Hat Entreprise Linux | Windows server | RHEL, Windows Server, Oracle EL, Ubuntu, SUSE LE, AMI Amazon Linux, Debian, Fedora, Gentoo Linux | Gento Linux | Ubuntu [\*](https://www.cloudcontrol.com/dev-center/Platform%20Documentation#available-stacks) | Windows Server 2008 R2 (x64) |  | Ubuntu, Debian, FreeBSD, CentOS |
| Datacenter | AWS US, AWS EU Irlande |  | Par défaut : AWS US.  Et localisation selon le souhait avec AnyCloud, AWS US & UE, OpenStack, HP Cloud Services, Entreprise Datacenter |  | Azure US centre-nord, centre-sud, ouest, est. Asie de l’est, sud-est. Europe de l’ouest, nord. [\*](http://en.wikipedia.org/wiki/Windows_Azure#Datacenters) | AWS US est, ouest, AWS UE, AWS Asie-Pacifique, AWS Amérique du sud, AWS US GovCloud  9 régions au total | Tous les Datacenter AWS EC2 US et UE. | AWS Irlande, ou alternativement dans le Datacenter souhaité | AWS à l’est des US | AWS, OpenStack, Rackspace, Windows Azure, HP Cloud VMware, Citrix Cloud Stack1, Terramark, XenServe. Peut être installé sur tous cloud supportant JClouds API | AWS, Rackspace : US, Hong Kong, Royaume Unis et Australie, DigitalOcean : NY et Amsterdam, Static Datacenter |
| Types de cloud | Publique | Publique, privé, hybride | Publique, privé | Publique, privé | Publique | Privé, publique | Publique, travaille pour un future PaaS privé et hybride | Publique, privé, hybride | Publique | Publique, privé, hybride | Publique, privé |
| Fourniture Self-service | Oui, on peut choisir 512Mo ou 1024Mo, avec stockage personnalisable | Oui | Oui | Oui | Oui | Oui | Oui | Oui, la mémoire et tout autre service complémentaire | Oui | Oui | Oui |
| Disponibilité du système | 99,22%  à 99,99%, 99,97% en moyenne | N/A | 99,9% |  | 99,95% [\*](http://www.windowsazure.com/fr-fr/support/legal/sla/) | 99,95% [\*](http://aws.amazon.com/fr/ec2-sla/) | 99,9% (à travers le support Premium) | 100% [\*](http://checks.paaswatch.com/600030/history) | 99,95% | Haute disponibilité et auto-recouvrement des applications | 24/7/365 |
| Sécurité des données | Les applications s’exécutent dans des environnements séparés. Les données sont stockées dans des BDs d’accès contrôlées. [\*](https://policy.heroku.com/security) |  |  |  |  |  | https://www.engineyard.com/products/cloud/cloud-security | <https://www.cloudcontrol.com/privacy-policy> |  | Sécurité par authentification, autorisation et dans le transport des données | Conformité PCI de niveau 3 et réalisation d’audits de sécurité |
| Application privées | Oui | Non, applications accessible par tous | Oui |  | Oui | Oui | Non, mais application derrière un pare-feu, ou doit être implémenté. | Non, mais doit être implémenté au niveau de l’application dans le cas du PaaS publique | Les applications peuvent être portées sur d’autres applications avec un minimum de changement de la configuration |  | Oui |
| Environnements | Un seul confondu | Développement, test, et autres environnements que l’on peut ajouter | Développement, local et production [\*](http://wiki.cloudbees.com/bin/view/RUN/CloudBeesWebXml#HApplicationEnvironments) | Un seul |  | Développement, production… | Développement, production, staging, et d’autre à ajouter | Développement, staging et production |  |  | Production, test et développement, possibilité d’ajouter d’autres environnements |
| Interopérabilité | Oui | Oui | Oui |  |  | Oui, avec plusieurs autres services cloud Amazon | Oui | Oui |  | Oui | Oui, avec d’autre fournisseur Cloud comme Amazon, Rackspace et DigitalOcean |
| Applications portables / No Lock-in | Oui, no-lock in pour la plateforme | Oui | Oui | Oui | Non, l’architecture des applications est propriétaire | Oui, grâce au large choix de OS,  Framework, serveur d’applications | Oui | Oui, les applications peuvent être exécutées sur des PaaS compatible buildpack-API (Heroku, Stackato…) ou autre serveur régulier |  | Oui, no-lock in concernant les PaaS, OS et code. | Oui |
| Polyglotte ou non | Oui.  Ruby est le plus utilisé, puis Python (l’architecte en chef Ruby est le créateur du langage) | Oui | Oui | Oui | Oui | Oui | Oui. Ruby est le plus utilisé. PHP est encore nouveau pour cette plateforme. | Oui, tout langage utilisé sur un environnement Linux. Les plus utilisés dans la plateforme sont Java, Ruby, Python et PHP | Que le Framework .NET | Oui | Oui |
| Mode de scalabilité | Automatique | Automatique |  | Automatique, manuelle, ou pas de scalabilité | Automatique ou manuelle | Automatique, avec frais supplémentaires (Amazon CloudWatch) | Automatique avec paramétrage | Automatique avec paramétrage | Automatique | Automatique avec règles à définir. | Automatique |
| Load Balancing | Oui [\*](https://devcenter.heroku.com/articles/http-routing#routing) | Oui | Oui, automatique | Oui | Oui | Oui avec frais supplémentaires (Elastic Load Balancing) | Oui. A activer manuellement |  | Oui |  |  |
| Politique en cas de panne |  |  |  |  |  |  | Des volumes EBS (Elastic Block Storage) sont montés pour l’application et les serveurs de base de données. Possibilité d’avoir une base de données en master et slave | Le système est déployé à travers plusieurs zones disponibles, donc une panne peu probable. La plateforme peut résoudre les problèmes avant que les utilisateurs finaux en soient affectés |  | Auto recouvrement |  |
| Plateforme mobile avec support d’application | Oui | Non | Oui, Android, iOs |  | Oui, Windows 8, iOs, Android | Oui | Oui avec quelques modules complémentaires [\*](https://www.engineyard.com/docs/Mobile_Solutions_Brief.pdf) | Oui | Non |  |  |
| Outils d’interaction | Console web, CLI Heroku Toolbelt, plugin Heroku | Console web, CLI CF, plugin CloudFoundry, API | Console web Grand Central, CLI bees, plugin CloudBees | Console web, CLI RHC, plugin OpenShift via JBoss | Console Web, plugin Azure pour IDE, API, Windows Azure PowerShell | Console Web, AWS Toolkit pour Eclipse, AWS Toolkit pour Visual Studio | Console Web et CLI EY | CLI CCTRL, Console Web, API | Console web, API | CLI Cloudify Shell, Console web | Console Web, CLI STATIC |
| Services complémentaires | Oui, en grand nombre | Oui, mais encore très peu |  | Oui | Oui | Oui | Oui | Oui | Oui | Oui |  |
| Méthode de reporting |  | Pas de reporting |  |  |  |  | Alertes mails à activer. Alertes sur les ressources utilisées |  |  | Envoi d’alertes et évènements |  |
| Méthodes de Gestion des ressources | Via la console web seule | Via le plugin |  | Console web, CLI |  | Via la console web et de façon automatique | Via la console web et CLI |  | Via la console web | Via la console web et CLI |  |
| Prix |  |  |  | <https://www.openshift.com/products/pricing> |  |  | Le prix dépend de plusieurs facteurs : nombre et taille des serveurs, support technique, bande passante, nombre de backups. [\*](https://www.engineyard.com/products/cloud/pricing) |  | https://appharbor.com/pricing | 0.03$ /Go/hr | http://www.static.com/pricing/cloud-hosting/ |
| Compte gratuit |  | Version d’essai de 60 jours. |  | OpenShift Online : 1,5Go de mémoire et 3Go de stockage | 150€ gratuits, mais carte de crédit demandée à l’inscription. | Niveau gratuit de 750h/mois d’utilisation d’instances | Version d’essai de 500h |  | Oui, 1 unité de travail |  | Oui, $10 de crédit gratuit |
| Support technique |  |  |  | Support de la communauté et professionnel |  |  | Support standard en live, support premium. |  | Support sur le site AppHarbor et Stackoverflow.com | Support par téléphone 24/7, support de la communauté |  |

## Comparaison par les Framework supportés

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Frameworks | Fournisseur Cloud PaaS | | | | | | | | | | |
|  | Heroku | CloudFoundry | CloudBees | OpenShift | Azure | AWS EC2 | Engine Yard | CloudControl | AppHarbor | Cloudify | Static |
| Spring |  |  |  |  |  |  |  |  |  |  |  |
| Java EE 6 Web Profile |  |  |  |  |  |  |  |  |  |  |  |
| Glass Fish App Server |  |  |  |  |  |  |  |  |  |  |  |
| Jetty |  |  |  |  |  |  |  |  |  |  |  |
| Zend PHP |  |  |  |  |  |  |  |  |  |  |  |
| Symphony |  |  |  |  |  |  |  |  |  |  |  |
| .NET |  |  |  |  |  |  |  |  |  |  |  |
| ASP. |  |  |  |  |  |  |  |  |  |  |  |
| Java |  |  |  |  |  |  |  |  |  |  |  |
| Play 2.0 |  |  |  |  |  |  |  |  |  |  |  |
| Lift |  |  |  |  |  |  |  |  |  |  |  |
| Grails |  |  |  |  |  |  |  |  |  |  |  |
| Python |  |  |  |  |  |  |  |  |  |  |  |
| Rails |  |  |  |  |  |  |  |  |  |  |  |
| Grape |  |  |  |  |  |  |  |  |  |  |  |
| Rack |  |  |  |  |  |  |  |  |  |  |  |
| Sinatra |  |  |  |  |  |  |  |  |  |  |  |
| Node.js |  |  |  |  |  |  |  |  |  |  |  |
| Liferay |  |  |  |  |  |  |  |  |  |  |  |
| JBoss AS |  |  |  |  |  |  |  |  |  |  |  |
| JBoss EAP |  |  |  |  |  |  |  |  |  |  |  |
| Tomcat |  |  |  |  |  |  |  |  |  |  |  |
| Django |  |  |  |  |  |  |  |  |  |  |  |
| Flask |  |  |  |  |  |  |  |  |  |  |  |
| Express |  |  |  |  |  |  |  |  |  |  |  |
| Clojure |  |  |  |  |  |  |  |  |  |  |  |
| Erlang |  |  |  |  |  |  |  |  |  |  |  |
| Google Go Webapp |  |  |  |  |  |  |  |  |  |  |  |
| Facebook App |  |  |  |  |  |  |  |  |  |  |  |
| iUI web Apps |  |  |  |  |  |  |  |  |  |  |  |
| Dart App |  |  |  |  |  |  |  |  |  |  |  |
| Compojure |  |  |  |  |  |  |  |  |  |  |  |
| Metor.js |  |  |  |  |  |  |  |  |  |  |  |
| Backbone.js |  |  |  |  |  |  |  |  |  |  |  |
| Angular JS |  |  |  |  |  |  |  |  |  |  |  |
| Railo (CFML) |  |  |  |  |  |  |  |  |  |  |  |
| Procfile |  |  |  |  |  |  |  |  |  |  |  |
| Perl |  |  |  |  |  |  |  |  |  |  |  |
| CakePHP |  |  |  |  |  |  |  |  |  |  |  |
| Reveal.js |  |  |  |  |  |  |  |  |  |  |  |
| WordPress |  |  |  |  |  |  |  |  |  |  |  |
| Drupal |  |  |  |  |  |  |  |  |  |  |  |
| Joomla |  |  |  |  |  |  |  |  |  |  |  |
| Switchyard |  |  |  |  |  |  |  |  |  |  |  |
| Cron |  |  |  |  |  |  |  |  |  |  |  |
| Bootstrap |  |  |  |  |  |  |  |  |  |  |  |
| FuelPHP |  |  |  |  |  |  |  |  |  |  |  |
| Lavarel |  |  |  |  |  |  |  |  |  |  |  |
| Kohana |  |  |  |  |  |  |  |  |  |  |  |
| WSGI |  |  |  |  |  |  |  |  |  |  |  |
| Rack |  |  |  |  |  |  |  |  |  |  |  |

## Comparaison par type de modules complémentaire disponible en extension

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Services complémentaires | Fournisseurs | | | | | | | | | | |
|  | Heroku [\*](https://addons.heroku.com/) | CloudFoundry | CloudBees [\*](http://www.cloudbees.com/platform/ecosystem/tech-partners.cb) | OpenShift | Azure | AWS EC2 | Engine Yard | CloudControl | AppHarbor | Cloudify | Static |
| Analyse de performance |  |  |  |  | App Dynamics |  |  |  |  |  |  |
| Test de la montée en charge | Blitz |  |  |  |  |  |  |  |  |  |  |
| SCM | Seulement hébergé |  | Git, SVN | Git hébergé ou personnel |  | Subversion, GitLab … 18 au total | Propose github, mais peut être changé |  | Bitbucket, CodePlex, et GitHub |  |  |
| ALM |  |  |  |  |  |  |  |  |  |  |  |
| Intégration continue | Tddium, Codeship, Semaphore |  | Jenkins | Jenkins | Non mais un plugin Windows Azure Storage peut être installé sur Jenkins |  |  |  |  |  |  |
| Livraison continue | Wercker |  |  |  |  |  |  |  |  |  |  |
| Gestion de projet |  |  |  |  |  |  |  |  |  |  |  |
| Bases de données SQL | PostgreSQL |  |  |  | ClearDB |  |  |  |  |  |  |
| Bases de données NoSQL | MongoDB |  |  |  | Table Service |  |  |  |  |  |  |
| Logging | Logentries |  |  |  |  |  |  |  |  |  |  |
| Email | Mailgun |  |  |  | SendGrid |  |  |  |  |  |  |
| SMS | Blower.io |  |  |  | Twilio |  |  | + MMS |  |  |  |
| Queuing | IronMQ |  |  |  | Service Bus Queues, Windows Azure Queue Service |  |  |  |  |  |  |
| Cache | MemCachier |  |  |  | Windows Azure Caching |  |  |  |  |  |  |
| Monitoring | New Relic |  |  |  | New Relic |  |  |  |  |  |  |
| Paiement | Spreedly |  |  |  |  |  |  |  |  |  |  |
| Collaboration | Wercker |  | xWiki | Wercker | VS Anywhere |  |  |  |  |  |  |
| Analyse de données Big Data | Treasure Data |  |  |  |  |  |  |  |  |  |  |
| Recherche | ElasticSearch |  |  |  | Bing Search API |  |  |  |  |  |  |
| Analyse de codes sources |  |  |  |  |  |  |  |  |  |  |  |

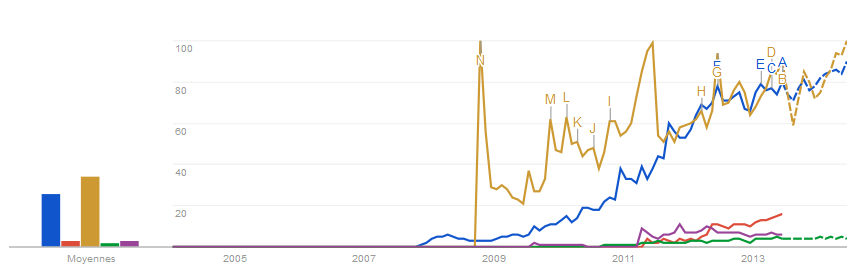
## Comparaison par les cas d’usage réalisés.

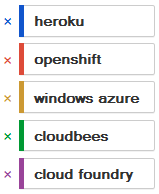
|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Cas d’usages | Fournisseurs | | | | | | | | | | |
|  | Heroku | CloudFoundry | CloudBees | OpenShift | Azure | AWS EC2 | Engine Yard | CloudControl | AppHarbor | Cloudify | Static |
| 1 |  |  |  |  |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |  |  |  |  |  |
| 4 |  |  | Avec Jenkins | Avec Jenkins |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |  |  |  |
| 6 | Seulement avec le gestionnaire Git hébergé |  | Possibilité d’utiliser son propre SCM | SCM au choix |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |  |  |  |  |  |
| 11 |  |  |  |  |  |  |  |  |  |  |  |
| 12 | Contrôle de la taille mémoire CPU seulement, les DNS | Contrôle du nombre d’instances et de la taille de la mémoire, les domaines | Contrôle du nombre d’instances, domaines | Contrôle du nombre de Gears, de la taille de stockage et de la scalabilité, les alias |  |  |  |  |  |  |  |
| 13 |  |  |  |  |  |  |  |  |  |  |  |
| 14 |  |  |  |  |  |  |  |  |  |  |  |
| 15 |  |  |  |  |  |  |  |  |  |  |  |
| 16 | Interaction par un SDK, et IDE |  | Interaction avec API, SDK et IDE |  |  |  |  |  |  |  |  |
| 17 |  |  |  |  |  |  |  |  |  |  |  |
| 18 | Système de backup intégré |  |  |  |  |  |  |  |  |  |  |
| 19 | CloudVertical Beta |  | CloudBees Metering and Billing |  |  |  |  |  |  |  |  |

|  |  |
| --- | --- |
|  | User Stories |
| 1 | En tant que développeur,  Je souhaite mettre en place un environnement de développement déterminé (serveur ou conteneur d’application, base de données, etc.) sur le cloud,  Afin de développer une application. |
| 2 | En tant que développeur,  Je souhaite utiliser environnement de développement existant sur le cloud,  Afin de participer au développement d’une application. |
| 3 | En tant que développeur,  Je souhaite mettre en place un gestionnaire de dépendances  Afin de disposer d’un dépôt de librairies partagées et de proximités |
| 4 | En tant que développeur,  Je souhaite mettre en place un environnement d’intégration continue sur le cloud,  Afin de vérifier qu’il n’y a pas de régression de la qualité d’une application à chaque modification du code source. |
| 5 | En tant que développeur,  Je souhaite analyser le code source de mon application et en obtenir les métriques de qualités,  Afin d’évaluer rapidement la dette technique de mon application et recenser les corrections à apporter |
| 6 | En tant que développeur,  Je souhaite stocker et gérer les versions de mes fichiers,  Afin de partager mes fichiers et conserver l’historique des modifications. |
| 7 | En tant que chef de projet,  Je souhaite planifier, piloter, suivre l’avancement et gérer les ressources d’un projet,  Afin de m’assurer de la conformité par rapport aux plans initiaux. |
| 8 | En tant que membre du projet,  Je souhaite mémoriser et organiser mes connaissances,  Dans le but de capitaliser et les partager avec tous mes collaborateurs. |
| 9 | En tant que testeur ou développeur,  Je souhaite mettre en place un système de suivis d’anomalies  Afin de corriger des erreurs fonctionnelles ou techniques d’une application. |
| 10 | En tant que testeur,  Je souhaite mettre en place un environnement de test (serveur ou conteneur d’application, base de données, etc.) sur le cloud,  Afin de tester une application dans un environnement déterminé (homologation, recette, pré-production). |
| 11 | En tant que développeur,  Je souhaite identifier, partager et centraliser les versions de mon logiciel binaire de ses dépendances  Afin d’éviter les conflits et les erreurs liées aux versions des logiciels et des dépendances |
| 12 | En tant CP,  Je souhaite configurer les ressources allouées sur le cloud  Afin de mieux en contrôler le coût |
| 13 | En tant qu’administrateur,  Je souhaite choisir des fournisseurs d’infrastructures différents pour mon environnement d'exécution  Afin de contrôler les coûts de consommation. |
| 14 | En tant qu’expert,  Je souhaite surveiller, diagnostiquer une application  Afin de contrôler ses performances |
| 15 | En tant qu’architecte,  Je souhaite modéliser mon système par des vues logiques, d'implémentations, de déploiements et de processus,  Afin de mieux faire comprendre et avoir une meilleure visibilité du système. |
| 16 | En tant que développeur,  Je souhaite interagir avec la plateforme cloud avec mon code source (via une API), avec un SDK (exemple de RHC) ou un IDE,  Afin d’ajouter à mon application un nouveau service. |
| 17 | En tant que décideur,  Je souhaite collecter et analyser mes données Big Data  Afin d’avoir une vue d’ensemble sur une activité et de pouvoir prendre des décisions. |
| 18 | En tant qu’administrateur,  Je souhaite sauvegarder les images de mes fichiers, base de données et applications  Afin de pouvoir les restaurer en cas de perte de données ou de défaillance du système. |
| 19 | En tant CP,  Je souhaite consulter un rapport de la consommation en ressources / être prévenu d’un seuil...  Afin de maîtriser leurs coûts |

## Intérêts

Le tableau ci-dessous représente les intérêts des recherches des mots clés liés aux différentes plateformes étudiées.





# Conclusion

Complexité de comparaison des offres cloud car les offres varient en fonctions des ressources allouées, le type d’application, la manière de gérer le cycle de vie d’une application, les services supplémentaires offerts, le mode de scalabilité, et tout un tas d’options. Pour être sûr d’utiliser la meilleure plateforme, à partir d’un type d’application, cibler les plateformes pouvant le contenir, comparer en fonction des ressources mémoire CPU consommées dans le temps et services complémentaires souhaités, et enfin, procéder à une élimination en fonction du coût.

Intérêts. Utilisation du cloud dans le monde : [file:///C:/Users/idiallo.stage/Documents/doc/sources/Gartner%20Says%20Worldwide%20Platform%20as%20a%20Service%20Revenue%20Is%20on%20Pace%20to%20Reach%20$1.2%20Billion.htm](file:///C:\Users\idiallo.stage\Documents\doc\sources\Gartner%20Says%20Worldwide%20Platform%20as%20a%20Service%20Revenue%20Is%20on%20Pace%20to%20Reach%20$1.2%20Billion.htm)

<http://france.emc.com/corporate/glossary/cloud-computing-services.htm> : bénéfices du cloud service