



REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE  
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique  
Université Mohamed Khider – BISKRA  
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie  
**Département d'informatique**

N° d'ordre : RTIC20/M2/2019

## Mémoire

Présenté pour obtenir le diplôme de master académique en

# Informatique

Parcours : RTIC

---

## Équilibrage de charge dans le cloud computing

---

Par :

**BERRAMDAN Marouan**

Soutenu le 6 juillet 2019, devant le jury composé de :

BOUREKKACHE Samir	MCB	Président
HOUHOU Okba	MAA	Rapporteur
BELOUNNAR Saliha	MAA	Examineur

## Remerciements

*En premier lieu, je tiens à remercier dieu de m'avoir donnée le courage, la volonté, la santé et la force pour réaliser ce travail.*

*Ensuite, je tiens à remercier mon encadreur de mémoire Mr HOUHOU Okba pour ses conseils, ses orientations et son aide.*

*Je remercie également les membres de jury Mr BOUREKKACHE Samir et Madame BELOUNNAR Saliha pour avoir accepté de présider le jury de ma soutenance.*

*Je tiens à remercier aussi tous les enseignants, qui m'ont suivi le long de mon cycle d'études.*

*Enfin, je présente mes remerciements à toutes les personnes qui m'ont encouragé à construire et élaborer ce mémoire.*

# Table des matières

<b>Introduction général</b>	<b>1</b>
<b>1 CLOUD COMPUTING</b>	<b>3</b>
1.1 Introduction	3
1.2 Qu'est-ce que le « cloud »	4
1.3 Définition	5
1.4 Notion de base	6
1.5 La Virtualisation	7
1.6 Les caractéristiques du cloud computing	8
1.6.1 Libre-service à la demande	8
1.6.2 Large accès au réseau	8
1.6.3 Mise en commun des ressources	8
1.6.4 Une souplesse rapide	8
1.6.5 Service mesuré	9
1.7 Les modèles de service du cloud computing	9
1.7.1 IaaS : Infrastructure as a Service	10
1.7.2 PaaS : Platform as a Service	10
1.7.3 SaaS : Software as a Service	10
1.8 Les modèles de déploiement du cloud computing	11
1.8.1 Cloud privé	11
1.8.2 Cloud communautaire	11
1.8.3 Cloud public	11
1.8.4 Cloud hybride	11
1.9 Exemples connus de Cloud Computing	12
1.10 Avantages	13
1.10.1 Réduction des coûts	13
1.10.2 Souplesse	14
1.10.3 Recentrage sur le cœur de métier	14
1.11 Inconvénients	15
1.11.1 Protection des données	15
1.11.2 Récupération de données et la disponibilité	15
1.11.3 Capacités de gestion	16

1.11.4 Restrictions réglementaires et de conformité . . . . .	16
1.12 Conclusion . . . . .	16
<b>2 ÉQUILIBRAGE DE CHARGE</b>	<b>17</b>
2.1 Introduction . . . . .	17
2.2 L' allocation de ressources . . . . .	18
2.2.1 Définition . . . . .	18
2.3 Les ressources . . . . .	19
2.4 Algorithmes d'équilibrage de charge . . . . .	20
2.4.1 Algorithmes statiques . . . . .	21
2.4.2 Algorithmes dynamiques . . . . .	23
2.5 Comparaison des algorithmes existants . . . . .	26
2.6 Métriques d'équilibrage de charge . . . . .	27
2.7 Objectifs de l'équilibrage de charge . . . . .	28
2.8 Conclusion . . . . .	29
<b>3 CONCEPTION</b>	<b>30</b>
3.1 Introduction . . . . .	30
3.2 Description du problème . . . . .	31
3.3 Objectifs . . . . .	31
3.4 Description du projet . . . . .	32
3.5 Architecture global . . . . .	32
3.6 L'algorithme proposé . . . . .	33
3.7 Pseudo Algorithme . . . . .	36
3.8 Conclusion . . . . .	37
<b>4 IMPLÉMENTATION</b>	<b>38</b>
4.1 Introduction . . . . .	38
4.2 Environnement de l'application . . . . .	38
4.2.1 Environnement de développement matériel . . . . .	38
4.2.2 Environnement de développement logiciel . . . . .	39
4.3 Description du fonctionnement de notre application . . . . .	43
4.4 Interface principale . . . . .	46
4.4.1 Configuration des paramètres de simulation . . . . .	47
4.4.2 Lancement de la simulation . . . . .	50
4.5 Expérience et résultats . . . . .	50
4.6 Conclusion . . . . .	53
<b>Conclusion général</b>	<b>54</b>
<b>Bibliographie</b>	<b>55</b>
<b>Résumé</b>	<b>57</b>

# Liste des figures

1.1	L'environnement de Cloud computing [1]	5
1.2	Architecture de Cloud computing [6]	6
1.3	La Virtualisation dans Cloud computing[15]	7
1.4	les coches de Cloud computing [6]	9
1.5	Les modèles de déploiement du cloud computing [5]	12
2.1	Scénario d'utilisation des ressources cloud [20]	19
2.2	Algorithmes d'équilibrage dans le charge de Cloud computing [12]	20
2.3	Round robin load balancer algorithme	21
2.4	Comparaison des algorithmes existants d'équilibrage de charge	26
2.5	Comparaison des algorithmes existants d'équilibrage de charge basés sur diverses métriques de cloud	28
3.1	Fonctionnement des algorithmes d'équilibrage de charge	32
3.2	Fonctionnement de l'algorithme propose	33
4.1	Architecture de CloudSim	41
4.2	Diagramme de classes du simulateur CloudSim	42
4.3	Diagramme d'utilisation	43
4.4	Diagramme de classe	44
4.5	Diagramme de séquence	45
4.6	Interface principale	46
4.7	Configuration du Datacenter	47
4.8	Configuration du VM	48
4.9	Configuration du Cloudlet	49
4.10	le chois de l'algorithme	49
4.11	Button pour lancer la simulation	50
4.12	Résulta de Round Robin Load Balancer	51
4.13	Résulta de Throttled Load Balancer	51
4.14	Résulta de L'algorithme proposé	52
4.15	Résulta de comparaison d'algorithmes par rapport le temps d'exécution	52

# Introduction général

L'informatique est en train de devenir un modèle consistant en des services standardisés et fournis d'une manière similaire aux services publics traditionnels tels que l'eau, l'électricité, le gaz. Dans un tel modèle, les utilisateurs accèdent aux services en fonction de leurs besoins, sans tenir compte du lieu où ils sont hébergés ou de la manière dont ils sont livrés. Plusieurs paradigmes informatiques ont promis d'offrir cette vision de l'informatique utilitaire, notamment l'informatique en cluster "**cluster computing**", l'informatique en grille "**Grid computing**" et, plus récemment, l'informatique en nuage "**Cloud computing**". Le Cloud Computing fournit des applications, des plate formes et des infrastructures comme des services mises à la disposition du client à base d'inscription selon le modèle « **pay-as-you-go** ». Le Cloud permet d'exposer les puissances des centres de données comme un réseau virtuel de services ce qui permet aux fournisseurs de services la mise à disposition des utilisateurs des plate-formes dynamiques et des applications flexibles.

L'allocation de ressources dans les systèmes de **Cloud computing** suscite une attention croissante avec l'augmentation de la popularité de Cloud. En général, l'allocation de tâches est le processus d'affectation des tâches aux ressources disponibles sur la base des caractéristiques et des conditions des tâches. C'est un aspect important dans le fonctionnement efficace du Cloud, car de divers paramètres de tâches doivent être pris en considération pour un ordonnancement approprié. Les ressources disponibles devraient être utilisées efficacement sans affecter les paramètres de service du Cloud. L'équilibrage de charge est un problème de gestion de ressources qu'un fournisseur de services doit assurer afin de garantir une qualité de services satisfaisante. En effet une utilisation efficace des ressources des centres de données permet de minimiser le temps de réponse et le temps d'exécution des tâches.

Pour réaliser cette mémoire, nous avons organisé ce travail en quatre chapitres suivons le plant suivant :

- \* Le premier chapitre **CLOUD COMPUTING** : dans ce chapitre nous allons détaille les notions de base du cloud computing, les caractéristiques, les modèles de service (IaaS, PaaS, SaaS),et les modèles de déploiement ainsi que ses avantages et inconvénients.
- \* Le deuxième chapitre **ÉQUILIBRAGE DE CHARGES** : dans ce chapitre nous allons voir le mécanisme d'allocation des ressources et le problème d'équilibrage de charge .Après nous allons présenté les différents algorithmes pour résoudre cette problème ,Enfin nous allons voir les métriques et les objectifs d'équilibrage de charge.
- \* Le troisième chapitre **CONCEPTION** : dans ce chapitre nous allons voir la description du problème , l'objectif ,la description et architecture global de notre projet . Ensuite nous allons voir notre algorithme proposé et ces différentes étapes et le pseudo algorithme .
- \* Le quatrième chapitre **IMPLÉMENTATION** : ce chapitre représente l'environnement de développement,les interfaces et la configuration des paramètres de simulation. Ensuite nous allons voir la configuration des paramètres de simulation.Enfin nous avons voir un expérience et résultats obtenir.

## Chapitre 1

# CLOUD COMPUTING

### 1.1 Introduction

Le cloud computing (L'informatique en nuage) est un paradigme de l'informatique, où un grand groupement de systèmes sont connectés à des réseaux privés ou publics, de fournir une infrastructure dynamique évolutive pour l'application, les données et le stockage de fichiers. Avec l'avènement de cette technologie, le coût de calcul, l'hébergement d'applications, le stockage de contenu et la livraison est réduit de manière significative. Le cloud computing est une approche pratique de l'expérience des avantages de coûts directs et il a le potentiel de transformer un centre de données à partir d'un ensemble à forte intensité capitalistique à un environnement prix variable. L'idée du cloud computing repose sur un principe fondamental de « réutilisabilité des capacités informatiques. La différence que le cloud computing apporte par rapport aux concepts traditionnels de « grid computing », « calcul distribué », « informatique utilitaire », ou « autonomic computing » est d'élargir les horizons à travers les frontières organisationnelles.[3]



## 1.2 Qu'est-ce que le « cloud »

Le cloud vous donne l'opportunité de repenser à votre mode de fonctionnement, en réorganisant les choses d'une nouvelle façon. En réalité, le cloud est le point culminant de nombreuses parties : évolution des modèles commerciaux, évolution fonctionnelle de l'Internet (augmentation de la vitesse de transmission, fiabilité accrue), normes de base, coûts de matériel et de réseau considérablement réduits. Le cloud est considéré comme une approche de plus haut niveau afin de fournir des services sécurisés, globaux, hautement évolutifs et flexibles pouvant être livrés et consommés à l'aide des standards Internet via un modèle commercial en cas de besoin, mais qui sont payants.

Les services de cloud computing existent comme une incarnation actuelle des technologies et des modèles commerciaux qui sont en constante évolution. Ils impliquent implicitement un certain nombre de capacités évoluées en termes d'infonuage, d'opérations dans les nuages et de normes dans le nuage : L'infonuage est l'infrastructure qui comprend les centres de données, les réseaux et les normes de communication.

Les opérations dans les nuages sont les outils de gestion, les interfaces de programmation applicative et les nombreuses disciplines associées à la gestion des environnements cloud. Les normes de l'informatique dans les nuages évoluent rapidement à partir des normes de l'Internet et du Web 2.0 existantes qui ont conduit à cette étape. L'objectif des efforts de normalisation dans le cloud est d'aligner les capacités de cloud computing et d'exploitation du cloud. Dans les premiers temps, les normes relatives aux cloud ont été essentiellement de facto, mais un nombre important d'entreprises s'efforcent de les faire évoluer (ou de créer des normes de jure au bénéfice de tous). Les concepts d'utilisation à grande échelle des ressources informatiques telles que le modèle d'approvisionnement de service ou Utility computing et la grille informatique ou grid computing sont dérivés des fournisseurs de services publics traditionnels tout comme le pouvoir ou l'eau. L'adoption d'utility computing est considérée comme un bon parallèle pour les services de cloud computing, car ils ont été initialement utilisés pour le traitement non critique.[6]

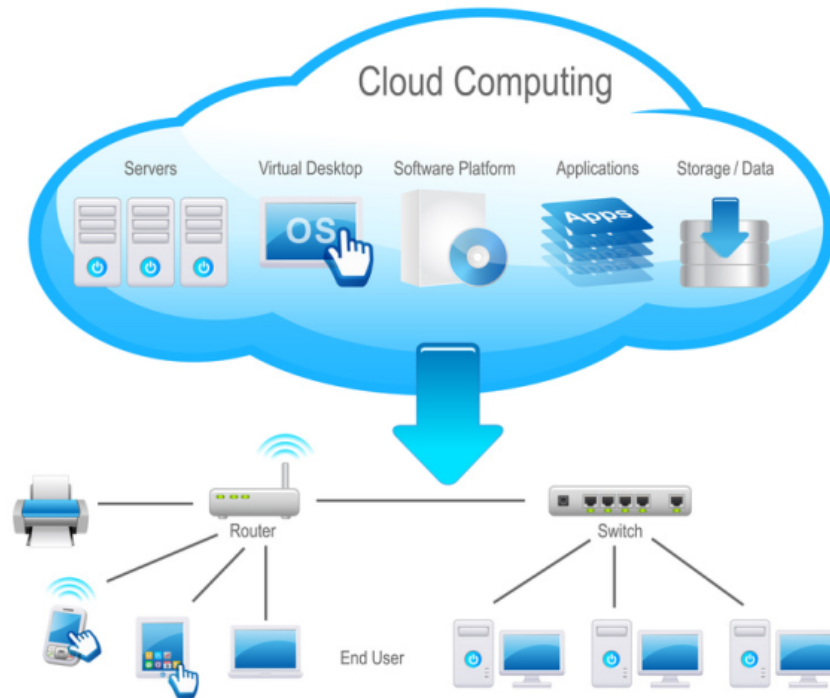


FIGURE 1.1 – L’environnement de Cloud computing [1]

### 1.3 Définition

Cloud computing selon l’Institut National des Normes et de la Technologie (NIST) : Le Cloud Computing est un modèle qui permet un accès réseau pratique et sur demande à un pool partagé de ressources informatiques configurables (par exemple, des réseaux, des serveurs, du stockage, des applications et des services) qui peut être rapidement approvisionné et disponible sans trop d’efforts de gestion ou d’interaction d’opérateurs. Ce modèle de cloud favorise la disponibilité et est composé de cinq caractéristiques essentielles, de trois modèles de service et de quatre modèles de déploiement. [1]

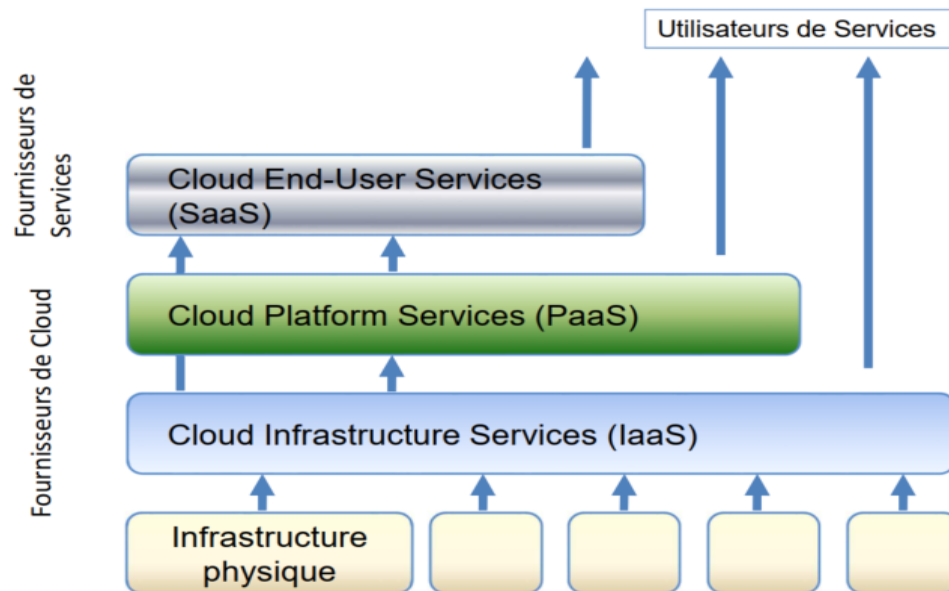


FIGURE 1.2 – Architecture de Cloud computing [6]

## 1.4 Notion de base

Voici quelques notions de base qui nous allons utiliser dans notre travail.

**A) DataCenter** : Il reçoit les requêtes des clients du cloud service pour les exécuter, en utilisant ses propres machines virtuelles dans un séquentielle ou parallèle. Le partage de ressources est l'un des principales caractéristiques du cloud computing. [9]

**B) Machine virtuelle** : Est un périphérique virtuel qui a une performance déterminée par le fournisseur de Cloud. Tel que capacité de stockage de la mémoire et les performances de son processeur pour le traitement des tâches souhaitées par le dataCenter. [9]

**C) Tâche** : a un ensemble des caractéristiques telles que la taille ,temps début d'exécution et taille de sortie ...etc .[14]

**D) Broker** : Il modifie plusieurs mécanismes : le mécanisme de soumission des demandes de provisioning de machine virtuelle aux centres de données et le mécanisme de soumission de machines virtuelles de tâches.[14]

**E) Host (hôte)** : L'hôte exécute des actions concernant la gestion des ordinateurs virtuels et mettre à jour le traitement des tâches pour pouvoir utiliser les VM.Un

bon hôte possède la politique définie pour provisionner la mémoire, les éléments de traitement et également la bande passante vers les machines virtuelles. un bon hôte est associé pour vous au centre de données. L'idée peut héberger des machines virtuelles.[14]

F) **Les algorithmes d'équilibrage de charge** : ils sélectionnent le virtuel machines dans un centre de données pour une nouvelle répartition des tâches, selon les mesures prédéterminées dans chaque algorithme. En général, on trouve deux types d'algorithmes, statique et dynamique.[9]

G) **DataCenterController** : Ce composant contrôle les activités du centre de données. [14]

## 1.5 La Virtualisation

La virtualisation est une technique qui permet de partager une seule instance physique d'une application ou d'une ressource entre plusieurs organisations ou locataires (clients). Il le fait en attribuant un nom logique à une ressource physique et fournir un pointeur sur cette ressource physique lorsqu'elle est demandée.

L'architecture multi comptes offre un isolement presque total entre les locataires multiples et par conséquent, les organisations peuvent utiliser et personnaliser l'application, comme si, ils possèdent chacun sa propre instance d'exécution.[21]

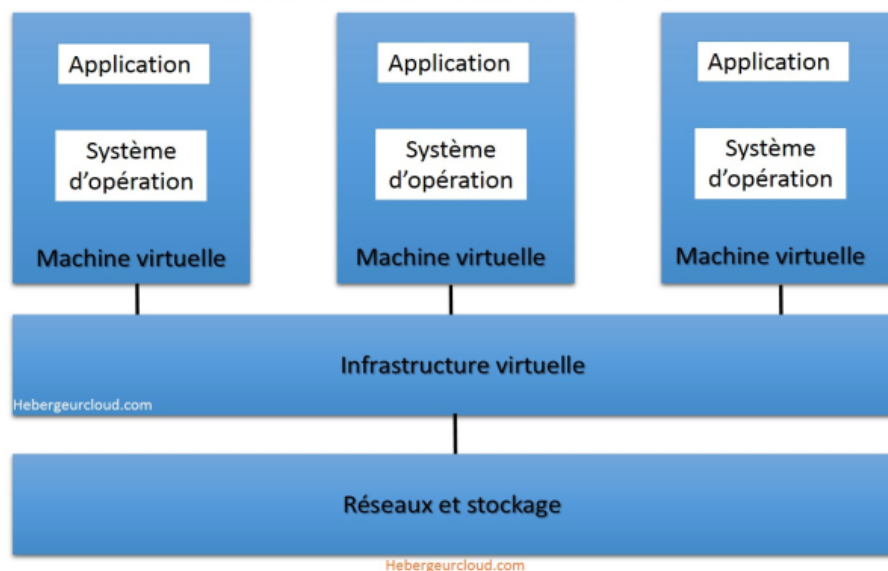


FIGURE 1.3 – La Virtualisation dans Cloud computing[15]

## 1.6 Les caractéristiques du cloud computing

### 1.6.1 Libre-service à la demande

un consommateur peut fournir unilatéralement des capacités informatiques telles que l'heure du serveur et le stockage en réseau, au besoin, sans nécessiter d'interaction humaine avec le fournisseur de chaque service.

### 1.6.2 Large accès au réseau

les capacités sont disponibles sur le réseau et sont accessibles via des mécanismes standards qui favorisent l'utilisation par des plates-formes clientes minces ou épaisses et hétérogènes (par exemple, les téléphones mobiles, les ordinateurs portables et les assistants personnels numériques ou PDA).[\[4\]](#)

### 1.6.3 Mise en commun des ressources

: les ressources informatiques du fournisseur sont regroupées pour servir plusieurs consommateurs à l'aide d'un modèle multi-locataire, avec différentes ressources physiques et virtuelles attribuées dynamiquement et réattribuées en fonction de la demande du consommateur. Il y a un sentiment d'indépendance de localisation puisque le client n'a généralement aucun contrôle ou connaissance sur l'emplacement exact des ressources fournies, mais peut être capable de spécifier l'emplacement à un niveau d'abstraction plus élevé (par exemple, pays, état ou centre de données). Des exemples de ressources comprennent le stockage, le traitement, la mémoire, la bande passante du réseau et les machines virtuelles.[\[4\]](#)

### 1.6.4 Une souplesse rapide

les capacités peuvent être rapidement et élastiquement provisionnées, parfois automatiquement, pour évoluer rapidement et être disponible dans un temps ré-

duit. Pour le consommateur, les capacités disponibles pour l'approvisionnement semblent souvent illimitées et peuvent être achetées en n'importe quelle quantité et à tout moment.[4]

### 1.6.5 Service mesuré

les systèmes de Cloud contrôlent et optimisent automatiquement l'utilisation des ressources en exploitant une capacité de comptage à un niveau d'abstraction ap-proprié au type de service (par exemple stockage, traitement, bande passante et comptes d'utilisateurs actifs).[4]

## 1.7 Les modèles de service du cloud computing

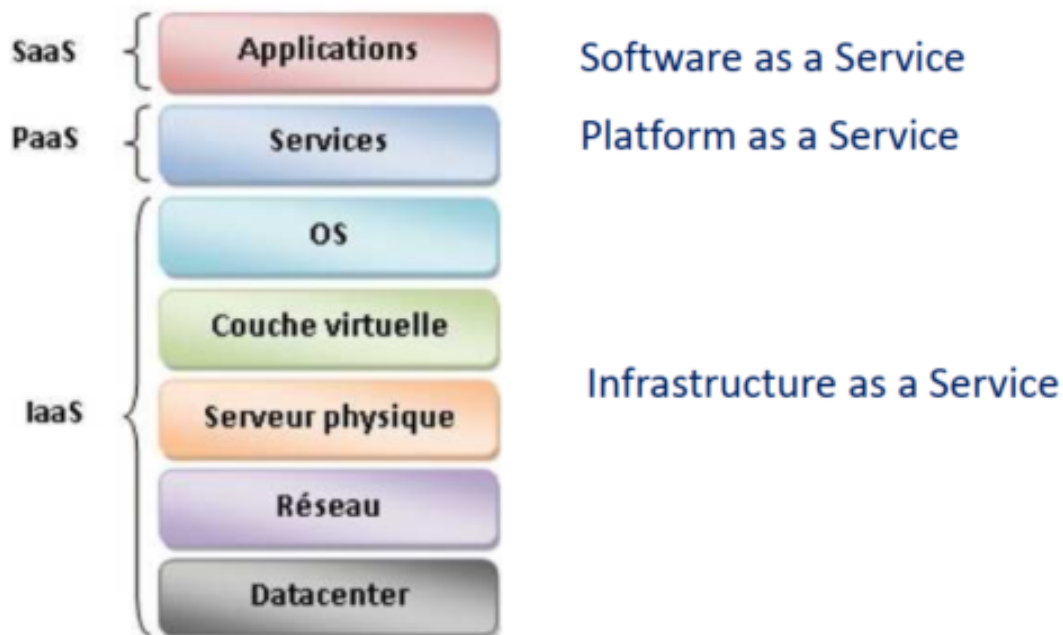


FIGURE 1.4 – les coches de Cloud computing [6]

### 1.7.1 IaaS : Infrastructure as a Service

Infrastructure as a Service (IaaS) fournit des ressources virtualisées, accessible à travers l'internet. Les ressources fournies sont virtualisées dans le sens où ces ressources sont une agrégation de plusieurs ressources fournit par plusieurs centres de données qui peuvent être géographiquement distribués. Grâce donc à la technologie de virtualisation, l'utilisateur aura l'impression qu'il a en possession un bloc de ressources, hors en réalité ces ressources peuvent être une agrégation de ressources distribuées géographiquement.[10]

### 1.7.2 PaaS : Platform as a Service

Platform as a Service (PaaS) est un modèle du cloud computing. Dans le modèle de PaaS, le fournisseur délivre tous ce qui est nécessaire afin que l'utilisateur puisse développer l'application désirée. Dans ce modèle, l'utilisateur a le contrôle sur l'application ainsi que l'environnement de développement de l'application, mais tout ce qui est attrait à l'infrastructure sous-jacente est géré par le fournisseur. [10]

### 1.7.3 SaaS : Software as a Service

Software as a Service (SaaS) est un modèle d'approvisionnement d'application où les applications sont hébergées au sein du vendeur ou d'un fournisseur de service. L'accès à de telles applications se fait à travers un réseau généralement l'internet. Grâce à ce modèle, l'utilisateur n'a pas à se préoccuper de l'ensemble des éléments essentiels pour l'exécution de l'application demandée : Infrastructure sous-jacente, système d'exploitation, etc.[10]

## 1.8 Les modèles de déploiement du cloud computing

### 1.8.1 Cloud privé

: l'infrastructure cloud est exploitée uniquement pour une organisation. Elle peut être gérée par l'entreprise elle-même ou par un tiers et peut exister sur site ou hors site.[16]

### 1.8.2 Cloud communautaire

: l'infrastructure cloud est partagée par plusieurs organisations, réunies au sein d'une communauté et partageant des préoccupations spécifiques communes (par exemple, la mission, les exigences de sécurité, des politiques et des considérations de conformité). Elle peut être gérée par les entreprises elles-mêmes ou par un tiers et peut exister sur site ou hors site.[16]

### 1.8.3 Cloud public

: l'infrastructure cloud est mise à la disposition du grand public ou d'un grand groupe industriel et appartient à une organisation vendant des services cloud. [16]

### 1.8.4 Cloud hybride

: l'infrastructure cloud est une composition de deux ou plusieurs clouds (privés, communautaires ou publics). Ceux-ci demeurent des entités uniques mais sont connectés par une technologie normalisée ou exclusive qui permet le partage des données et des applications des nuages).[16]



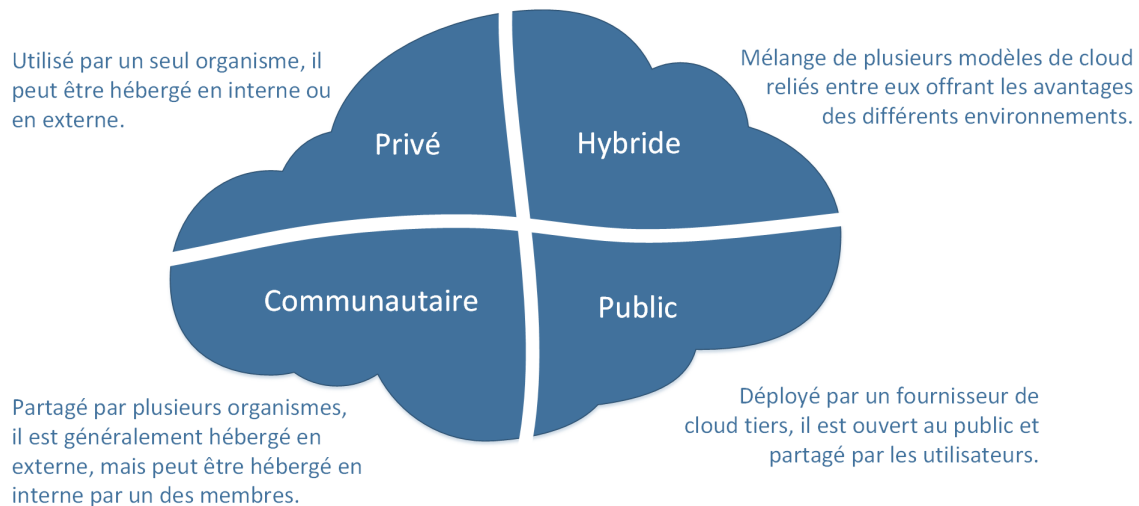


FIGURE 1.5 – Les modèles de déploiement du cloud computing [5]

## 1.9 Exemples connus de Cloud Computing

Voici quelques exemples de cloud computing : [6]

**Google Drive** : Google Drive est un pur service Cloud Computing. Il propose un stockage en ligne, et fonctionne avec les applications Cloud Google Docs, Google Sheets et Google Slides. Ce service est accessible depuis un ordinateur, depuis une tablette, ou même depuis un smartphone, au même titre que les applications mobiles Docs et Sheets. La plupart des services Google peuvent d'ailleurs être classés dans la catégorie du Cloud Computing. C'est le cas de Gmail, Google Calendar, et Google Maps par exemple.

**Apple iCloud** : Principalement utilisé pour le stockage en ligne, le back-up, Apple iCloud est également utile pour la synchronisation des mails, des contacts, ou encore du calendrier. Toutes les données sont disponibles sur iOS, Mac OS, ou sur les appareils Windows depuis le panneau de contrôle iCloud. Apple propose également des versions Cloud de son traitement de texte Pages, de sa feuille de calcul Spreadsheet, et de son logiciel de présentation Keynote pour tous les utilisateurs d'iCloud. La plateforme permet en outre aux possesseurs d'iPhone d'utiliser la fonctionnalité Localiser mon iPhone.

**Amazon Cloud Drive** : Amazon propose essentiellement le stockage de musique au

format MP3 et de photos. Les abonnés Amazon Prime bénéficient d'une capacité de stockage illimitée. Amazon Cloud Drive sert également à stocker le contenu acheté pour Kindle. En résumé, cette plateforme sert à stocker tout le contenu numérique acheté auprès d'Amazon.

Des services hybrides comme Box, Dropbox ou SugarSynch se positionnent dans la catégorie Cloud en proposant une version synchronisée sur internet des fichiers stockés. La synchronisation est une pierre angulaire du Cloud Computing, même si l'accès aux fichiers se fait de façon locale. De même, si plusieurs personnes avec des appareils séparés travaillent sur les mêmes données synchronisées, on peut considérer qu'il s'agit de Cloud Computing.

## 1.10 Avantages

Voici quelques-uns des avantages possibles pour ceux qui offrent des services de cloud computing et les applications [6] :

### 1.10.1 Réduction des coûts

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

Le cloud computing peut permettre d'effectuer des économies, notamment grâce à la mutualisation des services sur un grand nombre de clients. Certains analystes indiquent que 20 à 25

L'abonnement à des services de cloud computing peut permettre à l'entreprise de ne plus avoir à acquérir des actifs informatiques comptabilisés dans le bilan sous forme de CAPEX et nécessitant une durée d'amortissement. Les dépenses informatiques peuvent être comptabilisées en tant que dépenses de fonctionnement.

### 1.10.2 Souplesse

Les caractéristiques du cloud sont qualifiées par les Anglo-Saxons sous le vocable *elastic computing capacity*. Le National Institute of Standards and Technology en a donné une définition succincte qui reprend ces principes de base : « L'informatique dans les nuages 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. »

L'« élasticité » du nuage permet de fournir des services évolutifs et peut permettre de supporter des montées en charge. Inversement, le fournisseur a la maîtrise sur les investissements, est maître des tarifs et du catalogue des offres, et peut se rémunérer d'autant plus facilement que les clients sont captifs.

### 1.10.3 Recentrage sur le cœur de métier

Le recours au cloud computing permet de décharger les équipes informatique des entreprises, qui ont alors plus de disponibilité pour des activités à haute valeur ajoutée.

La maintenance, la sécurisation et les évolutions des services étant à la charge exclusive du prestataire, dont c'est généralement le cœur de métier, celles-ci ont tendance à être mieux réalisées et plus rapidement que lorsque sous la responsabilité du client (principalement lorsque celui-ci n'est pas une organisation à vocation informatique)

## 1.11 Inconvénients

Voici quelques-uns des défis notables liés à l'informatique en nuage, et bien que certains d'entre eux peuvent causer un ralentissement lors de la livraison plus de services dans le nuage, la plupart peuvent également offrir des possibilités, si résolus avec soin et attention aux étapes de planification.[3]

### 1.11.1 Protection des données

La sécurité des données est un élément crucial qui justifie un examen minutieux. Les entreprises sont réticentes à acheter une assurance de la sécurité des données d'entreprise auprès des vendeurs. Ils craignent de perdre des données pour la concurrence et la confidentialité des données des consommateurs. Dans de nombreux cas, le l'emplacement de stockage n'est pas divulgué, ce qui ajoute aux préoccupations de sécurité des entreprises. Dans les modèles existants, pare-feu à travers les centres de données (appartenant à des entreprises) protègent cette information sensible. Dans le modèle Cloud, les prestataires de services sont responsables maintenir la sécurité des données et les entreprises devraient compter sur eux.

### 1.11.2 Récupération de données et la disponibilité

Toutes les applications d'entreprise ont des accords de niveau de service qui sont rigoureusement suivis. Les équipes opérationnelles jouent un rôle clé dans la gestion des accords de niveau de service et gouvernance d'exécution des applications. Dans les environnements de production, les équipes opérationnelles prise en charge des clusters appropriés et échec de la surveillance du système de réplique des données (surveillance des transactions, suivi des journaux et autres) maintenance (gouvernance d'exécution) capacité de reprise après sinistre et gestion du rendement si, l'une des les services sont sous-desservis par un fournisseur de Cloud, les dommages et l'impact pourrait être grave.

### 1.11.3 Capacités de gestion

Malgré qu'il y ait plusieurs fournisseurs de Cloud, la gestion de la plate-forme et l'infrastructure en est encore à ses balbutiements. Des fonctions telles que «Auto-Scaling», par exemple, sont indispensables pour de nombreuses entreprises. Il existe un énorme potentiel d'amélioration fonctionnelles d'évolutivité et d'équilibrage de charge fournies aujourd'hui.

### 1.11.4 Restrictions réglementaires et de conformité

Dans certains pays européens, la réglementation gouvernementale n'autorise pas les clients les informations personnelles et autres informations sensibles doivent être physiquement situées à l'extérieur l'état ou le pays. Afin de répondre à ces exigences, les fournisseurs de cloud doivent configurer un centre de données ou un site de stockage exclusivement dans le pays pour se conformer aux règlements. Avoir une telle infrastructure n'est pas toujours faisable et c'est un gros problème. Défi pour les fournisseurs de cloud.

## 1.12 Conclusion

Au cours de cette première chapitre, nous avons fourni une base théorique sur le Cloud Computing, en présentant ses caractéristiques, ses modèles de service (IaaS, PaaS, SaaS), et ses modèles de déploiement, ainsi que ses avantages et inconvénients. Dans le chapitre suivant, nous allons voir l'allocation de ressources et le problème d'équilibrage de charge dans les environnement cloud computing, ensuite nous allons voir les différents algorithmes.

## Chapitre 2

# ÉQUILIBRAGE DE CHARGE

## 2.1 Introduction

Le Cloud Computing fournit des applications, des plateformes et des infrastructures comme des services mises à la disposition du client à base d'inscription selon le modèle « pay-as-you-go ». Ces services sont référés par « Infrastructure as service » (IaaS), « plateforme as service » (PaaS) et « software as service » (SaaS). Le Cloud permet d'exposer les puissances des centres de données comme un réseau virtuel de services, ce qui permet aux fournisseurs de services la mise à disposition des utilisateurs des plateformes dynamiques et des applications flexibles. L'utilisateur sera donc capable d'accéder et déployer des applications de n'importe où sur internet. Cet accès est conditionné par une certaine qualité de services (QoS) offerte par le fournisseur. L'équilibrage de charge est un problème de gestion de ressources qu'un fournisseur de services doit assurer afin de garantir une qualité de services satisfaisante. En effet une utilisation efficace des ressources des centres de données permet de minimiser le temps de réponse et le temps d'exécution des tâches. [8]

## 2.2 L'allocation de ressources

### 2.2.1 Définition

De nos jours, il y a eu une augmentation dramatique dans la popularité des systèmes du cloud computing qui offrent des ressources informatiques à la demande, basent sur la facturation à l'usage, afin que les utilisateurs puissent augmenter ou diminuer leur taux de consommation de ressources en fonction de leurs besoins. Ces environnements peuvent multiplexer de nombreux utilisateurs sur la même infrastructure physique [20]

Dans les environnements de cloud computing, il ya deux acteurs : les fournisseurs et les utilisateurs de cloud computing. D'une part, les fournisseurs détiennent des ressources informatiques massives dans leurs grands centres de données et louent ces ressources à des utilisateurs. D'autre part, il ya les utilisateurs qui ont des applications avec des charges variées et louent des ressources de la part des fournisseurs pour exécutant leurs applications.

L'un des aspects intéressants de cloud computing est que ces acteurs ont leurs propres intérêts. En règle générale, l'objectif des fournisseurs est de maximiser que possible leurs revenus avec un investissement minimum. À cet effet, les fournisseurs veulent maximiser l'utilisation de leurs ressources informatiques, à titre d'exemple, par l'hébergement du plus grand nombre que possible de machines virtuelles sur chaque machine. D'autre part, les utilisateurs veulent accomplir leur travail à un coût minimal ou, en d'autres termes, ils veulent maximiser leur performance économique

## 2.3 Les ressources

Nous avons identifié plusieurs définitions des ressources de cloud :

" Les Ressources de cloud peuvent être vues comme n'importe quelle ressource (physique ou virtuel) que les utilisateurs peuvent demander du cloud, Par exemple, les utilisateurs peuvent demandées des exigences de réseau, telles que la bande passante et les délais, et des exigences computationnelle, telles que le processeur, la mémoire et le stockage. En général, les ressources sont situées dans un centre de données qui est partagé par plusieurs clients, et doivent être attribués et ajustés dynamiquement en fonction de la demande" . [18]

"Les ressources de cloud consistent en ressources physiques et virtuelles. Les ressources physiques sont partagées entre plusieurs demandes grâce à la virtualisation et le provisioning<sup>13</sup>, La demande des ressources virtualisées est décrite par un ensemble de paramètres détaillant les besoins de traitement, de mémoire et de disque. Le provisioning satisfait la demande de ressources par le mapping (la correspondance) les ressources virtualisées sur les ressources physiques. Les ressources matérielles et logicielles sont allouées aux applications cloud sur la base de la demande" .[19]

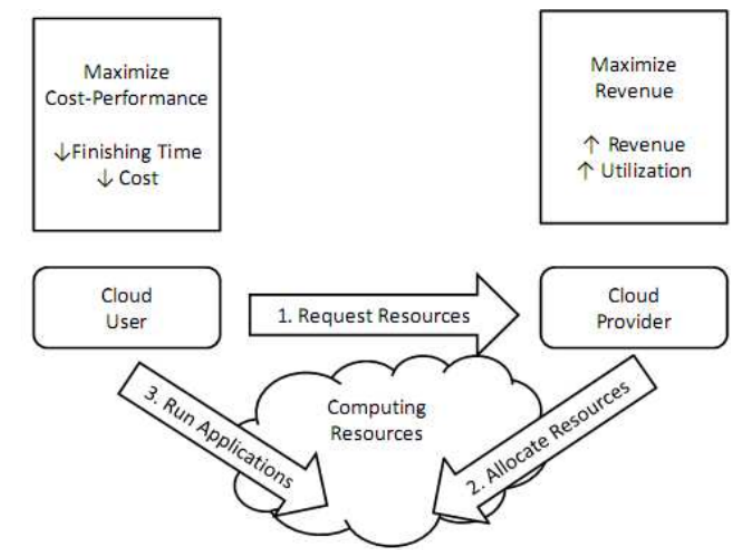


FIGURE 2.1 – Scénario d'utilisation des ressources cloud [20]



## 2.4 Algorithmes d'équilibrage de charge

Les algorithmes d'équilibrage de charge des machines virtuelles sont utilisés pour affecter les tâches des différents utilisateurs à un ensemble de machines virtuelles hétérogènes. Celles-ci sont créées dans les hôtes du centre de données par le « datacenter controller ». Ce contrôleur utilise un module « ActiveVmLoadBalancer », à base d'un algorithme d'équilibrage de charge, pour déterminer la machine virtuelle à assigner à la prochaine requête (tâche). Les algorithmes d'équilibrage de charge les plus utilisés sont « Round Robin Load Balancer », « Throttled Load Balancer », « Active Monitoring Load Balancer », et « Efficient Load Balancer ». [8]

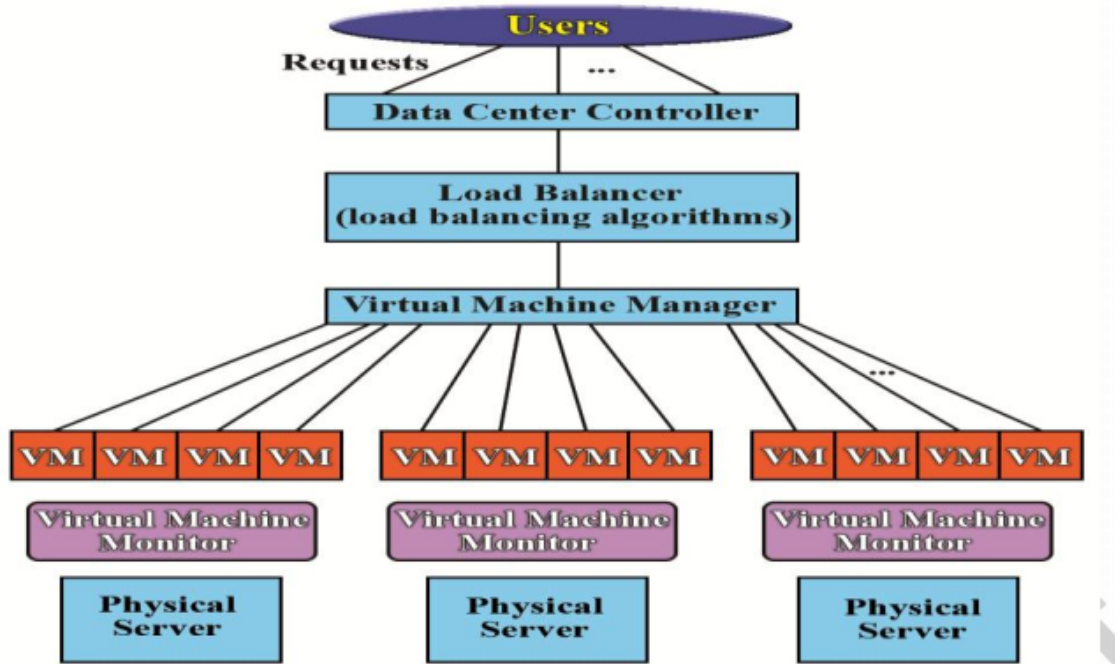


FIGURE 2.2 – Algorithmes d'équilibrage dans le charge de Cloud computing [12]

### 2.4.1 Algorithmes statiques

Avant chaque attribution de tâches à la machine virtuelle, les algorithmes d'équilibrage de charge statique déterminent les performances et la distribution des charges, et le contrôleur du centre de données reçoit les résultats à la fin de chaque traitement.[9]

#### Round Robin Load Balancer

Round robin load balancer est le plus simple des algorithmes d'équilibrage de charge de machines virtuelles. Les tâches sont affectées aux machines virtuelles en faisant la rotation sur l'ensemble des machines virtuelles instanciées par le contrôleur du centre de données. Les requêtes de cet algorithme sont réparties entre des machines virtuelles, l'autre s'appuyant sur l'aide du contrôleur du data centre. L'ordre d'allocation des tâches a lieu dans chaque machine virtuelle localement et indépendamment de l'autre machine distante, sur la base du nombre de tâches disponibles et du nombre de machines virtuelles.[8]

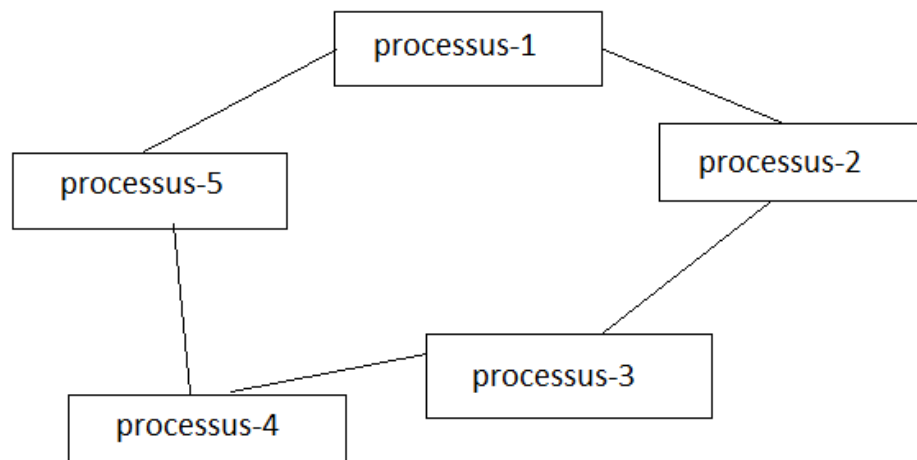


FIGURE 2.3 – Round robin load balancer algorithme

**Étape 1 :** Les équilibreurs de charge Round Robin VM ont un index de VMs. Initialement, toutes les machines virtuelles n'ont aucune allocation.

**Étape 2;**

- a) Le contrôleur du centre de données reçoit l'utilisateur requêtes / cloudlets.
- b) Les demandes sont affectées aux machines virtuelles de manière circulaire.
- c) L'équilibreur de charge Round Robin VM allouera le quantum de temps pour l'exécution de la demande de l'utilisateur.

**Étape 3 :** Après l'exécution des cloudlets / requêtes, Les machines virtuelles sont désallouées par la charge de machine virtuelle Round Robin balancier.

**Étape 4 :** Le contrôleur du centre de données vérifie les nouveaux / demandes en attente / en attente dans la file d'attente.

**Étape 5 :** Continuer à partir de l'étape 2.[7]

**MIN-MIN Load Balancing Algorithm** Une liste de tâches est maintenue et le temps d'exécution minimum est calculé pour tous les nœuds disponibles. Une tâche avec un temps d'exécution minimum est assignée à la machine. D'où le nom de l'algorithme est min-min . Mettez à jour la liste et temps d'exécution de la machine. Il donne de bons résultats lorsque de petites tâches sont plus complexes.[12]

**MIN-MAX Load Balancing Algorithm** Une liste de tâches est gérée et le temps d'achèvement minimal est calculé pour tous les nœuds disponibles. Une tâche avec un temps d'exécution maximum est assignée à la machine. D'où le nom de l'algorithme est min-max . Mettez à jour la liste et temps d'exécution de la machine .[12]

**Active Minoriting Load Balancer** Active minoriting VM load balancer maintien des informations concernant chaque machine virtuelle, telle que le nombre de tâches affectées à chacune d'elles. Lors de la réception de nouvelle requête, la machine virtuelle dont le nombre de tâches affectées est minimal est identifiée pour l'allocation. Le « ActiveVmLoadBalancer » retourne l'ID de la machine virtuelle au « DatacenterControler ». Ce dernier envoie la requête à la machine identifiée et notifie le « ActiveVmLoadBalancer » pour l'allocation. « Active VmLoadBalancer » incrémente alors le nombre de tâches affectées à la machine virtuelles. [8]

## 2.4.2 Algorithmes dynamiques

Il diffère des algorithmes statiques en ce que la charge de travail est distribuée entre les processeurs lors de l'exécution. Le maître assigne de nouveaux processus aux esclaves en fonction du nouveau informations recueillies. Contrairement aux algorithmes statiques, algorithmes dynamiques allouent des processus dynamiquement quand on des processeurs devient sous chargé. Au lieu de cela, ils sont mis en mémoire tampon dans la file d'attente de l'hôte principal et alloué dynamiquement à la demande des hôtes distants. [12]

**Throttled Load Balancer** Cet algorithme maintient une collection d'états des machines virtuelles (valable/NonValable). Pour chaque nouvelle tâche, il envoie l'ID de la première machine virtuelle dont l'état « Valable » au « DatacenterController » pour l'affectation. Ensuite le « DatacenterController » envoie la tâche à la machine virtuelle identifiée et notifie l'algorithme pour l'allocation. Lorsque la machine virtuelle termine l'exécution de la tâche, elle notifie de son tour le contrôleur du centre de données. Enfin ce dernier notifie l'algorithme pour la désallocation.[10]

**Étape 1 :** L'algorithme conserve une table d'index des machines virtuelles et l'état de la machine virtuelle (**Occupé / disponible**). Au début, tous les VMs sont disponibles.

**Étape 2 :** **DataCenterController** reçoit une demande de la base d'utilisateurs.

**Étape 3 :** **DataCenterController** interroge l'algorithme pour la prochaine allocation.

**Étape 4 ;** L'algorithme analyse d'allocation table jusqu'à la première machine virtuelle disponible .

**Si trouvé :**

a) : L'algorithme renvoie l'ID de la machine virtuelle à le contrôleur de centre de données.

b) : **DataCenterController** envoie la demande au VM identifiée par cet identifiant.

c) : Le **DataCenterController** notifie l'algorithme pour la nouvelle allocation.

d) : L'algorithme met à jour l'allocation table en conséquence.

**Si non trouvé :**

a) : L'algorithme renvoie -1.

b) : Le contrôleur de centre de données met la demande en file d'attente.

**Étape 5 :** Lorsque la VM termine le traitement de la demande, et le contrôleur de

centre de données obtient la réponse Cloudlet, il informe L'algorithme de la désallocation de la VM.

**Étape 6 : DataCenterController** vérifie s'il existe des toutes les demandes en attente dans la file d'attente. S'il y en a, ça porte à partir de l'étape 3.

**Étape 6** Continuer à partir de l'étape 2. [10]

**Central Queue Algorithme** : Fonctionne sur le principe de distribution dynamique. Il stocke de nouvelles activités et non remplies demandes en tant que file d'attente FIFO cyclique sur l'hôte principal. Chaque nouvelle l'activité qui arrive au gestionnaire de files d'attente est insérée dans le queue.

Ensuite, chaque fois qu'une demande d'activité est reçue par le gestionnaire de files d'attente, il supprime la première activité de la file d'attente et l'envoie au demandeur. S'il n'y a pas d'activités prêtes dans la file d'attente, la demande est mise en mémoire tampon, jusqu'à ce qu'une nouvelle activité soit lancée. disponible. Si une nouvelle activité arrive dans le gestionnaire de files d'attente pendant qu'il y a des demandes sans réponse dans la file d'attente, la première de ce type demande est retirée de la file d'attente et la nouvelle activité est attribué à elle.

Lorsque la charge du processeur tombe en dessous du seuil, la charge locale gestionnaire de charge envoie une demande pour une nouvelle activité à la centrale gestionnaire de charge. Le gestionnaire de charge central répond à la demande immédiatement si une activité prête est trouvée dans la demande de processus file d'attente ou met la demande en file d'attente jusqu'à l'arrivée d'une nouvelle activité. [12]

**Efficient Response Time Load Balancer** Cet algorithme est obtenu en modifiant l'algorithme « Throttled », il est divisé en trois phases. La première phase est la phase d'initialisation, dans laquelle le temps de réponse pour chaque machine virtuelle est calculé. Dans la deuxième phase, l'algorithme trouve la machine virtuelle la plus efficace ayant un temps de réponse minimal. La dernière phase, l'algorithme retourne l'ID de la machine virtuelle au contrôleur du centre de données. L'algorithme est détaillé comme suit : [7]

**Étape 1** L'algorithme trouve le temps de réponse pour chaque machine virtuelle.

**Étape 2** Lorsqu'une nouvelle requête arrive au contrôleur du centre de données, l'algorithme sélectionne la machine virtuelle qui a un temps de réponse minimal.

**Étape 3** L'algorithme retourne l'ID de la machine sélectionnée au contrôleur du centre de données.

**Étape 4** Le contrôleur notifie l'algorithme de cette nouvelle allocation.

**Étape 5** L'algorithme met à jour la table d'allocation en incrémentant le nombre de tâches affectées à la machine virtuelle.

**Étape 6** Lorsque la machine virtuelle termine le traitement de la requête et le contrôleur du centre de données reçoit la réponse, ce dernier notifie l'algorithme pour la désallocation et retourne à la 2ème étape.

L'algorithme calcule le temps de réponse pour chaque machine virtuelle en utilisant la formule suivante :

$$ResponseTime = FinT - ArrT + TDelay$$

Avec **ArrT** est le temps d'arrivé des requêtes, **FinT** le temps de fin de requête utilisateur et **TDelay** le délai de transmission de la requête. Ce dernier est calculé comme suit :

$$TDelay = TLatency + TTransfer$$

Avec **TLatency** est la latence du réseau, et **TTransfer** le temps de transfert de la requête de la source vers la destination :

$$TTransfer = D / BwperUser$$

Avec  $D$  la taille de données de la requête, et

$$BwperUser = BwTotal / Nr$$

telle que **BwTotal** est la bande passante valable, et **Nr** le nombre de requête utilisateur en transmission. **Active Clustering** Le clustering actif est une méthode améliorée d'échantillonnage aléatoire, Le concept de clustering est utilisé dans cet algorithme. Le principe de base de cet algorithme est de regrouper des nœuds similaires

Le regroupement de noeuds aide les ressources à augmenter efficacement le débit. Dans cet algorithme, une méthode appelée matchmaker est introduite, Lors du démarrage d'une exécution, le premier noeud sélectionne le noeud voisin. Le noeud voisin est considéré comme un nœud correspondant, qui connecte le nœud voisin

identique au nœud initial.

Enfin, le nœud de correspondance est déconnecté. Et ce processus est fait de manière itérative pour équilibrer la charge de manière égale. Les performances du système sont fortement améliorées en augmentant le débit. Il y a une utilisation efficace des ressources lorsque le débit augmente

## 2.5 Comparaison des algorithmes existants

Algorithme	Dynamique /statique	Avantages	Inconvénient
Round Robin Load Balancer	Statique	Moins d'utilisation des ressources Traite le serveur entier de manière égale.	Aucun processus n'est connu à l'avance. La migration de processus n'est pas possible
Active <u>Minority</u> Load Balancer	Statique	Maintient des informations concernant chaque machine virtuelle	la puissance de traitement n'est pas prise en compte
MIN-MIN Load Balancing	Statique	Effectue mieux petit temps d'exécution.	Conduit à la famine.
MIN-MAX Load Balancing	Statique	Améliore l'efficacité en augmentant l'exécution simultanée.	L'exécution qui prend le temps maximal doit attendre longtemps
Throttled Load Balancer	Dynamique	La liste des VM est maintenue avec le statut de chaque VM Bonne performance	Analyse la liste complète des ordinateurs virtuels depuis le début
Central Queue	Dynamique	Moins d'utilisation des ressources	La migration de processus n'est pas possible
Active Clustering load balancing	Dynamique	Fonctionne mieux avec des ressources élevées. Des nœuds similaires sont regroupés.	Les performances sont médiocres lorsque la variété de nœuds augmente.

FIGURE 2.4 – Comparaison des algorithmes existants d'équilibrage de charge

## 2.6 Métriques d'équilibrage de charge

Différentes métriques prises en compte dans l'équilibrage de charge dans le cloud computing .[12]

- \* **Débit** : Cette métrique permet de calculer le nombre de processus terminés par unité de temps.
- \* **Temps de réponse** : Il mesure le temps total nécessaire au système pour exécuter une tâche soumise.
- \* **Tolérance aux pannes** : détermine la capacité de l'algorithme à équilibrer la charge dans l'événement. de certains échecs dans certains nœuds ou des liens.
- \* **Durée de la migration** : durée nécessaire au transfert d'une tâche d'un nœud surchargé vers un un sous-chargé.
- \* **Scalability** : Il détermine la capacité du système à réaliser un algorithme d'équilibrage de charge avec un nombre restreint de processeurs ou de machines.
- \* **Performance** : Il mesure l'efficacité du système après l'exécution d'un algorithme d'équilibrage de charge.
- \* **Consommation d'énergie** : Il calcule la quantité d'énergie consommée par tous les nœuds. L'équilibrage de charge permet d'éviter la surchauffe et donc la réduction de la consommation d'énergie en équilibrant la charge sur tous les nœuds.



Algorithme	Métrique	Débit	Temps de réponse	Tolérance aux pannes	Scalability	Performance	Consommation d'énergie	Durée de la migration
Round Robin Load Balancer		Oui	Oui	Non	Oui	Oui	Non	Non
Active <b>Minority</b> Load Balancer		Oui	Oui	Non	Oui	Non	Oui	Oui
MIN-MIN Load Balancing		Oui	Oui	Non	Non	Oui	Oui	Non
MIN-MAX Load Balancing		Oui	Oui	Non	Non	Oui	Oui	Oui
Throttled Load Balancer		Non	Oui	Oui	Oui	Oui	Non	Oui
Central Queue		Oui	Oui	Oui	Non	Oui	Oui	Oui
Active Clustering load balancing		Oui	Oui	Non	Oui	Oui	Oui	Non

FIGURE 2.5 – Comparaison des algorithmes existants d'équilibrage de charge basés sur diverses métriques de cloud

## 2.7 Objectifs de l'équilibrage de charge

- \* **Utilisation optimale des ressources** : c'est l'un des objectifs primordiaux de l'équilibrage de la charge en tant que ressource appropriée l'utilisation est essentielle pour l'efficacité du modèle de cloud . [13]
- \* **Haut débit** : un débit élevé est souhaité attribut requis pour un système hautes performances, ce qui n'est possible que si la charge de travail et les ressources sont distribuées de manière uniforme aux différents nœuds . [13]
- \* **Temps de réponse court** : Pour de meilleures performances , la quantité de temps prise pour réagir par un algorithme d'équilibrage de charge devrait être réduit. [13]

## 2.8 Conclusion

Dans ce chapitre nous avons vu le mécanisme d'allocation des ressources et le problème d'équilibrage de charge dans l'environnement cloud computing. Ensuite nous avons présenté les différents algorithmes pour résoudre ce problème. Enfin nous avons vu les métriques et objectifs d'équilibrage de charge. Dans le chapitre suivant, nous allons montrer la conception de notre projet et l'algorithme proposé.

## Chapitre 3

# CONCEPTION

### 3.1 Introduction

Les systèmes de cloud computing sont caractérisés par l'utilisation d'un grand pool de ressources informatiques. Pour une utilisation optimale de ces ressources, il est important l'amélioration des algorithmes de répartition de charge pour minimiser le temps de réponse et le temps de traitement des applications des utilisateurs. Dans ce chapitre, nous allons voir la conception de notre système, qui a consisté à l'utilisation d'un algorithme hybride proposé afin de résoudre le problème de répartition de charge dans le cloud computing. Nous allons voir au début les deux algorithmes utilisés pour faire une combinaison des avantages de chaque algorithme. Ensuite, nous allons voir l'idée de base et tous les paramètres utilisés, ainsi que les différentes étapes de l'algorithme proposé.

## 3.2 Description du problème

Le cloud computing est un modèle offre un accès réseau commode et à la demande à un pool partagé de ressources informatiques configurables (par exemple, réseaux, serveurs, stockage, applications et services) qui peuvent être provisionnées rapidement et libérés avec un effort minimum de gestion ou d'interaction avec le fournisseur de service.

L'allocation efficace des ressources dans le cloud est une tâche très difficile, car elle doit prendre en compte plusieurs contraintes à savoir : satisfaire les exigences de l'utilisateur, assurer la performance des serveurs...etc.

Équilibrage de charge dans le cloud computing est défini comme l'affectation des ressources disponibles telles que le processeur, la mémoire, le stockage, la bande passante ... etc, d'une manière efficace. Il s'agit de la partie principale de la gestion des ressources.

## 3.3 Objectifs

Notre travail a deux objectifs principaux sont :

- \* La proposition d'une algorithmme d'équilibrage de charge dans le cloud computing.
- \* La simulation de cet algorithmme et comparaison avec les autres algorithmme .

Nous avons choisi comme métrique le temps de exécution.

### 3.4 Description du projet

Après une étude détaillée de différents algorithmes d'équilibrage de charge présentés dans la deuxième chapitre nous avons constaté qu'un meilleur algorithme d'équilibrage de charge de machines virtuelles dans un environnement Cloud Computing doit prendre en considération toutes les métriques d'équilibrage de charge, ainsi que la charge actuelle des machines virtuelles et les caractéristiques de différentes tâches.

Dans notre projet de fin d'étude, nous sommes intéressés à la proposition d'un algorithme d'équilibrage de charge dans le cloud computing qui prend en considération toutes les métriques d'équilibrage de charge possible

### 3.5 Architecture globale

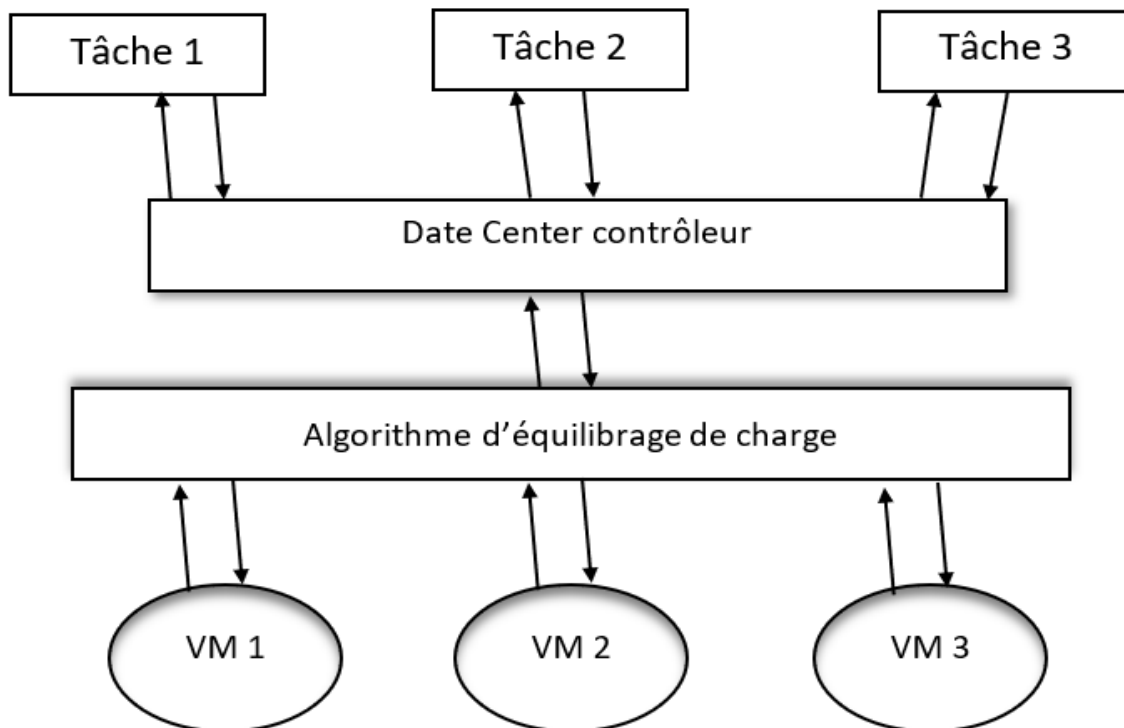


FIGURE 3.1 – Fonctionnement des algorithmes d'équilibrage de charge

Lors l'arrivée des tâches des utilisateurs, le contrôleur de **datacenter** notifier l'algorithme d'équilibrage de charge pour la nouvelle allocation en suit l'algorithme sélectionnée la machine virtuelle qui donne le meilleur performance salon métriques d'équilibrage de charge. lorsque la machine virtuelle termine le traitement des tâches et le contrôleur reçoit la réponse , ce dernier notifier les machine virtuelles pour la desallocation des tâches.

### 3.6 L'algorithme proposé

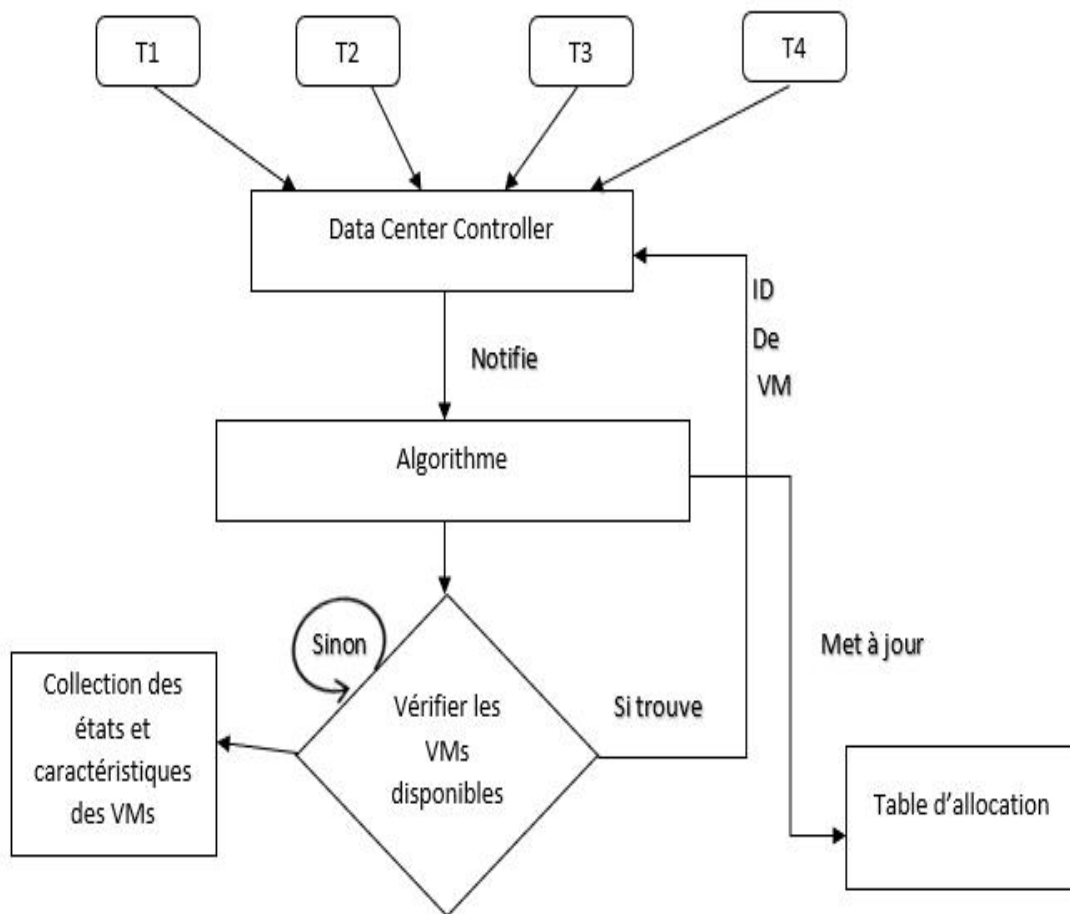


FIGURE 3.2 – Fonctionnement de l'algorithme proposé

L'algorithme d'équilibrage de charge « **Throttled Load Balancer** » ne prennent pas en considération les caractéristiques de différentes tâches et les machine virtuelles . Nous proposons une amélioration de l'algorithme « **Throttled Load Balancer** » Notre algorithme maintient une collection d'états des machines virtuelles (Occupé / disponible), et une collection des caractéristiques pour chaque machine virtuelle .

Pour chaque nouvelle tâche, il envoie l'ID de la meilleur machine virtuelle dont l'état « **Valable** » au « **DatacenterControler** » pour l'affectation Ensuite le « **DatacenterControler** » envoie la tâche à la machine virtuelle identifiée et notifie l'algorithme pour l'allocation. Lorsque la machine virtuelle terminal 'exécution de la tâche, elle notifie de son tour le contrôleur du centre de données. Enfin ce dernier notifie l'algorithme pour la désallocation.

**Étape 1** ; L'algorithme conserve une table d'index de l'état de la machine virtuelle (**Occupé / disponible**), une collection des caractéristiques pour chaque machine virtuelle. Au début, tous les VMs sont disponibles.

**Étape 2** : **DataCenterController** reçoit une demande d'utilisateurs, après il notifie l'algorithme pour la prochaine allocation.

**Étape 3** : L'algorithme analysée tous les VMs disponible ,pour sélectionne la meilleur VM ,après la comparaison entre les caractéristiques de l'tâche et les caractéristiques des VMs . Si trouvé :

- a) L'algorithme renvoie l'ID de la VM à la **DataCenterController** .
- b) **DataCenterController** envoie la demande au VM identifiée par cet ID, et notifie l'algorithme pour la nouvelle allocation.
- c) L'algorithme met à jour la table d'allocation en conséquence.

**Si non trouvé**

- a) L'algorithme renvoie -1.
- b) **DataCenterController** met la demande en file d'attente

**Étape 5** ; Lorsque la VM termine le traitement de la demande, et le contrôleur de centre de données obtient la réponse , il informe L'algorithme de la désallocation de la VM.

**Étape 6** : Le **DataCenterController** vérifie s'il existe des demandes en attente . S'il y en a, ça porte à partir de l'étape 3.

Ensuit en calculer la temps de traînement et la temps d'exécution comme suit :

*Le temps d'exécution = FinishTime - StartTime*

- FinishTime :représente le temps de la démarrage de l'exécution de cloudlet.
- StartTime :représente le temps de la terminaison de l'exécution de cloudlet.

*Le temps de traitement = temps d'exécution + temps d'attente*

- temps d'attente : est le temps qui prendre un cloudlet pour démarrer sont exécution.



### 3.7 Pseudo Algorithmme

---

**Algorithm 1** AlgorithmmePropose
 

---

```

1: vmStatesList : list de état de vm
2: VmAccpted : list de vms accepte
3: cloudletList : list de cloudlets
4: state : état de Vm
5: temp : int
6: vmId : int
7: MinVmId : int
8: for i 0 jusqu'à cloudletList.size() do
9:   if vmStatesList.size() > 0 then
10:     temp=0
11:     vmId=-1
12:     for chaque Vm en vmStatesList do
13:       temp= actuelle Vm en vmStatesList
14:       if temp < vmStatesList.size() then
15:         temps = vmStatesList.get(temp)
16:         if (state = AVAILABLE) et (Ram de actuelle Vm >= taille de ac-
tuelle cloudlet) then
17:           vmId = temp
18:           ajoute actuelle vm à VmAccpted
19:         end if
20:       end if
21:     end for
22:     MinVmId = id de la vm que possède le min ram dans VmAccpted
23:     bindCloudletToVm(MinVmId,i)
24:     cloudletList.remove(i);
25:   end if
26: end for
27:

```

---

## 3.8 Conclusion

Au cours de cette chapitre , nous avons voir la description du problème , l'objectif ,la description et architecture global de notre projet . Ensuite nous avons voir notre algorithme proposé et ces différentes étapes et le pseudo algorithme .

Dans le chapitre suivant nous allons voir l'environnement d'application (matériel et logiciel),description du fonctionnement de notre application,Ensuite nous allons voir la configuration des paramètres de simulation ,Enfin nous allons voir la résultats de la simulation.

## Chapitre 4

# IMPLÉMENTATION

### 4.1 Introduction

Ce chapitre a pour objectif de présenter l'environnement logiciel et matériel et aussi les fonctions principales utilisés pour développer et implémenter notre application ensuite nous présentons l'interface et les résultats de l'application.

### 4.2 Environnement de l'application

#### 4.2.1 Environnement de développement matériel

L'algorithme a été implémenté sur une machine PC avec :

- \* **Processeur** : Intel (R) Core (TM) i5-2450M CPU @ 2.50 GHz 2.50 GHz.
- \* **RAM** : 4GB
- \* **Système d'exploitation** : Microsoft Windows 10 professionnel 64 bits.

## 4.2.2 Environnement de développement logiciel

Notre algorithme a été implémenté en utilisant le langage de programmation **JAVA** en utilisant la framework **CloudSim** sur le **Eclipse IDE**

**JAVA** : Java est un langage de programmation moderne développé par Sun Microsystems (aujourd'hui racheté par Oracle). Il ne faut surtout pas le confondre avec JavaScript (langage de scripts utilisé principalement sur les sites web), car Java n'a rien à voir.

Une de ses plus grandes forces est son excellente portabilité : une fois votre programme créé, il fonctionnera automatiquement sous Windows, Mac, Linux, etc.

**JavaFX** : JavaFX est une bibliothèque Java utilisée pour créer des applications riches. Les applications écrites à l'aide de cette bibliothèque peuvent s'exécuter de manière cohérente sur plusieurs plates-formes. Les applications développées à l'aide de JavaFX peuvent être exécutées sur divers appareils tels que des ordinateurs de bureau, des téléphones portables, des téléviseurs, des tablettes, etc.

*Fonctionnalités de JavaFX*

- \* **Écrit en Java** : La bibliothèque JavaFX est écrite en Java et est disponible pour les langages pouvant être exécutés sur une machine virtuelle Java, notamment **Java**, **Groovy** et **JRuby**. Ces applications JavaFX sont également indépendantes de la plate-forme.
- \* **FXML** : JavaFX propose un langage connu sous le nom de FXML, qui est un langage HTML similaire à un langage de balisage déclaratif. Le seul but de ce langage est de définir une interface utilisateur.
- \* **Scene Builder** : JavaFX fournit une application appelée Scene Builder. Lors de l'intégration de cette application dans des environnements IDE tels que Eclipse et NetBeans, les utilisateurs peuvent accéder à une interface de conception glisser-déposer, utilisée pour développer des applications FXML (tout comme les applications Swing Drag Drop et DreamWeaver).

**Eclipse IDE** : Eclipse est un environnement de développement intégré (IDE) utilisé dans la programmation informatique. En 2014, il était l'IDE Java le plus utilisé. Il contient un espace de travail de base et un système de plug-in extensible permettant de personnaliser l'environnement.

Eclipse est écrit principalement en Java et est principalement utilisé pour le développement d'applications Java. Il peut également être utilisé pour développer des applications dans d'autres langages de programmation via des plug-ins.

**CloudSim** : est une structure de simulation généralisée et extensible qui permet la modélisation des environnements hétérogènes, la simulation et l'expérimentation de Cloud émergent des infrastructures de calcul et des services d'application. *Il offre les fonctionnalités suivantes :*

- \* Une plate-forme indépendante pour la modélisation des DataCenters, des Brokers, de l'ordonnancement et des politiques d'allocation des ressources .
- \* Un Support pour la modélisation et la simulation 'a grande échelle d'infrastructure de Cloud Computing.

*les principales caractéristiques de CloudSim :*

- \* Une simulation des spécifications, de la création et de la destruction de machines virtuelles.
- \* Une simulation de la définition de matériel de centre de traitement des données (Data Center) en termes de machines physiques composées de processeurs, de dispositifs de stockage, de mémoire et de largeur de bande interne
- \* Une simulation de l'exécution des programmes utilisateurs ou des tâches (Cloudlet) sur les machines virtuelles.

Dans notre travail nous avons utilisé la version CloudSim 3.0.3.

**Architecture de CloudSim** La structure logicielle de CloudSim et ses composants sont représentés par une architecture en couches comme indique dans la Figure 4.1 , Parmi les fonctionnalités, nous avons la formation d'une file d'attente et le traitement d'évènements, la création de composants système (les services, les machines (Host), le centre de données (Data Center), le courtier (Broker), les machines virtuelles), la communication entre les composants et la gestion de l'horloge de simulation.

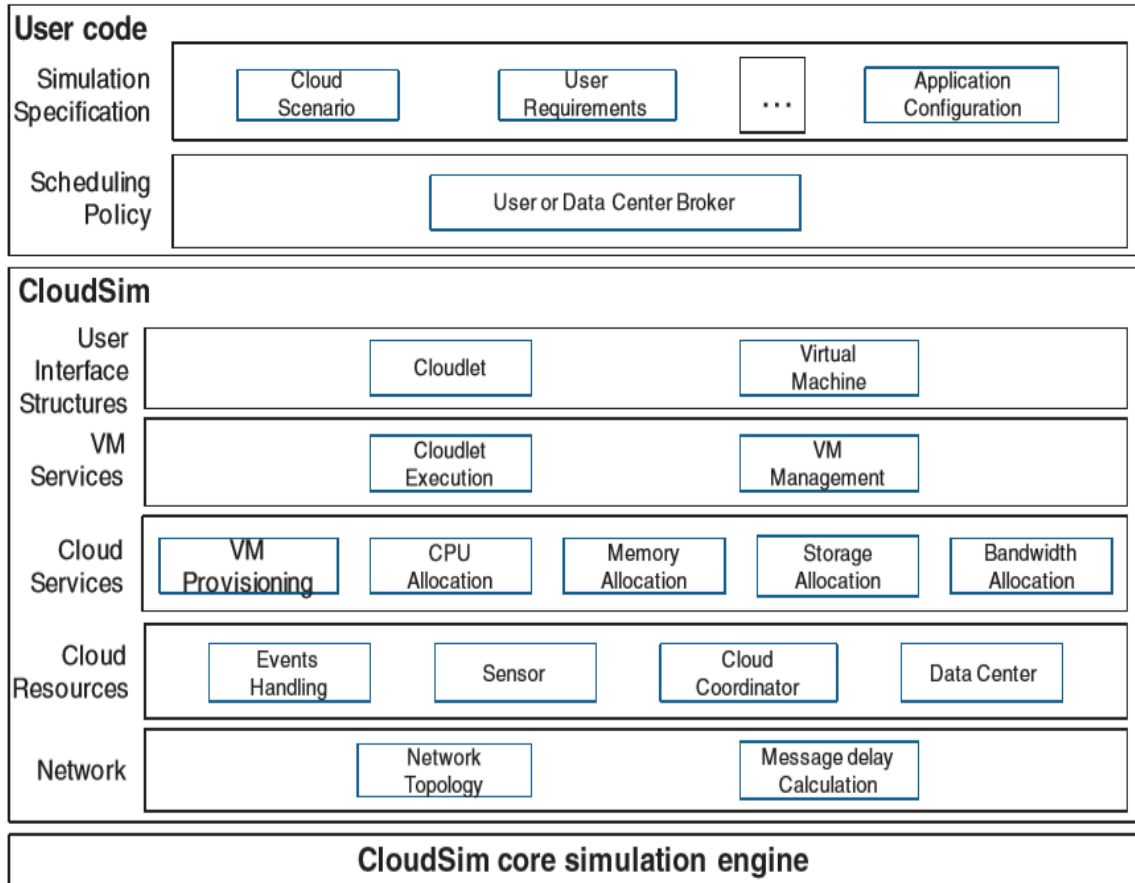


FIGURE 4.1 – Architecture de CloudSim

### Classes de CloudSim :

Le simulateur CloudSim est composé de plusieurs classes que nous pouvons classer en deux catégories :

- \* Classes qui modélisent les entités comme le Data Center, le Broker, le NetUser, Cloudlet, etc...
- \* Classes modélisant les politiques d'allocation. Parmi les classes fondamentales qui forment les blocs constitutifs du simulateur CloudSim.



### 4.3 Description du fonctionnement de notre application

Pour décrire les différentes Étapes de notre travail, nous avons opté pour le langage UML. Parmi les diagrammes d'UML que nous avons choisi pour notre conception, nous citons le diagramme d'utilisation, de classes et de séquence.

**Diagramme d'utilisation** Nous présentons maintenant une vue globale de notre simulateur qui permet de détailler les différentes étapes effectuées pour réaliser une simulation. La Figure 4.3 représente les fonctionnalités d'utilisation nécessaires à l'application.

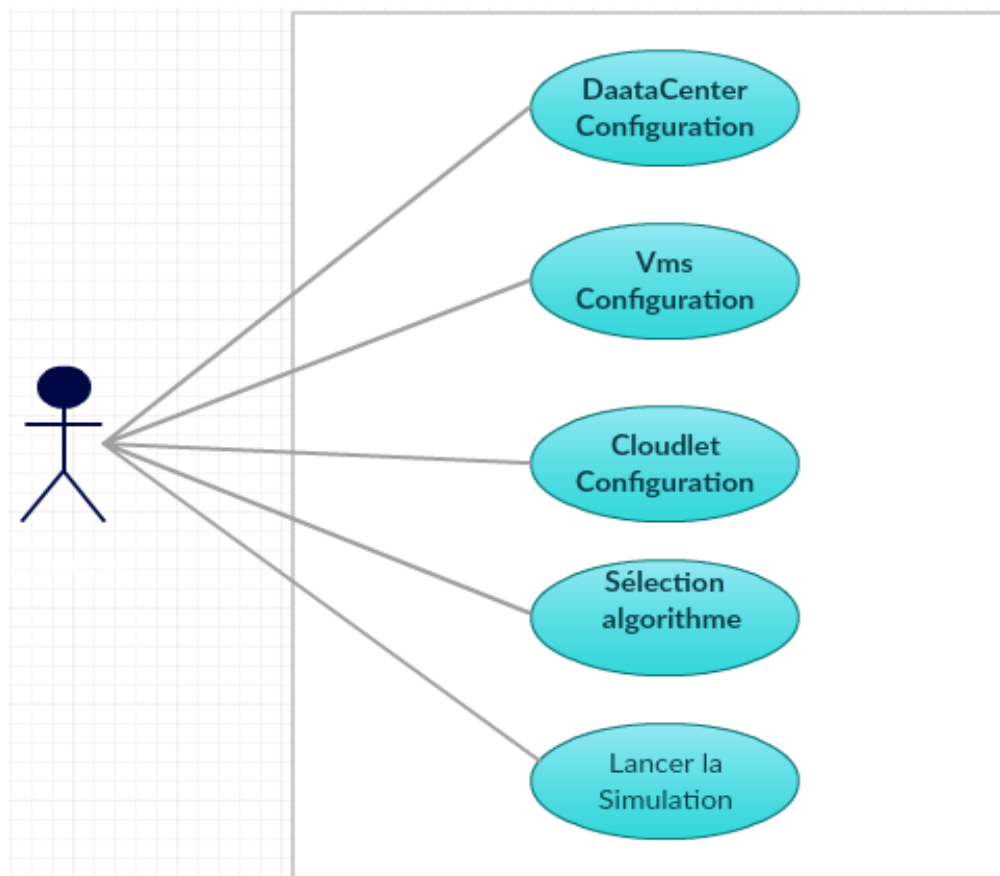


FIGURE 4.3 – Diagramme d'utilisation



**Diagramme de classes** Le diagramme de classes permet de représenter la structure du système comme un ensemble de relations entre classes .

Figure 4.4 représente notre 3 principales classes que nous ajoute pour les classes de CloudSim .

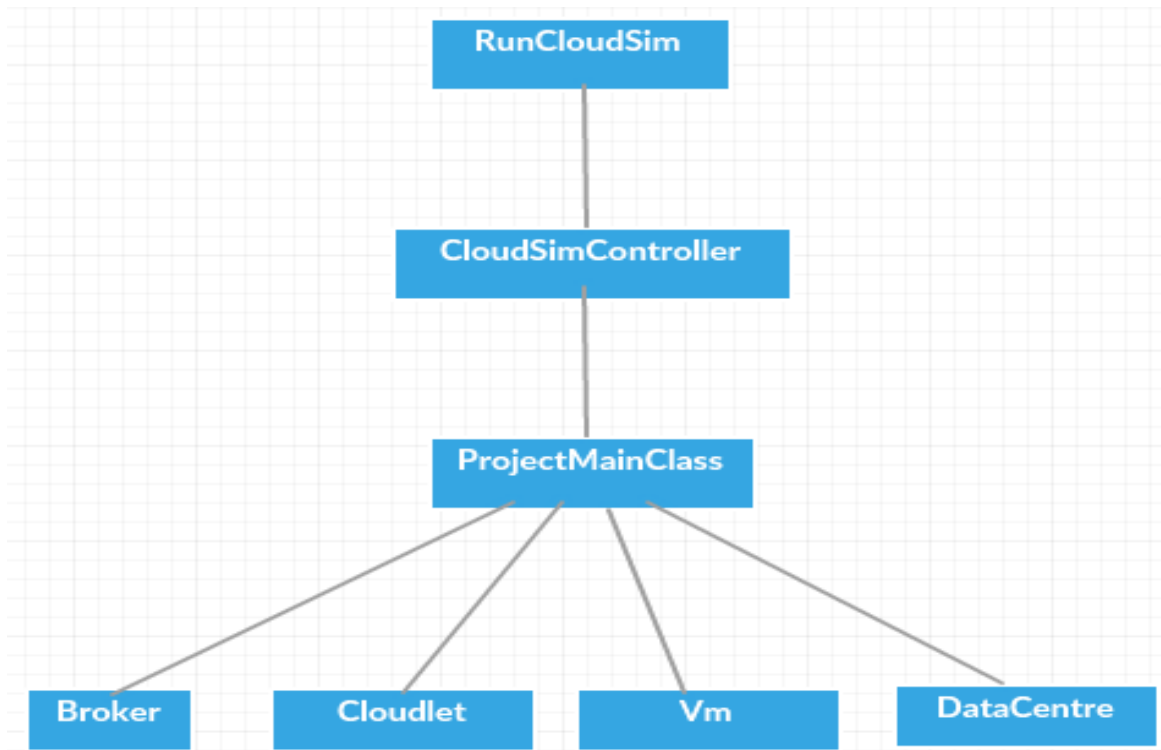


FIGURE 4.4 – Diagramme de classe

**RunCloudSim** : Cette classe représente la classe main de notre application, elle hérite de la classe **Application** (classe de JavaFx). Elle est responsable d'afficher l'interface principale.

**CloudSimController** : Cette classe représente le contrôleur de l'application, il est responsable de l'interaction entre l'utilisateur et l'interface.

**ProjectMainClass** : la classe principale de notre application, elle est responsable de la création des **Vms, DataCentres, Hosts et Cloudlets**, et de l'application de l'algorithme de répartition de charge choisi par l'utilisateur.

**Diagramme de séquence** Le diagramme de séquence est l'un des vues dynamiques les plus importantes d'UML. La Figure 4.5 illustre le flux de communication entre les principaux acteurs de CloudSim.

Au début de la simulation l'utilisateur entre tout les paramètres de la simulation (l'exécution de la classe **RunCloudSim**). après l'initialisation de la simulation la classe **RunCloudSim** fait un appel a la classe **CloudSimController** qui fait la relation entre l'interface utilisateur est le code de notre application .

Ensuit la classe **CloudSimController** à son tour appeler la classe principales **ProjectMainClass** , qui fait la création de la Broker,Vms,dataCeners,CloudLets et démarrer la simulation , démarrer affiche la resulta.

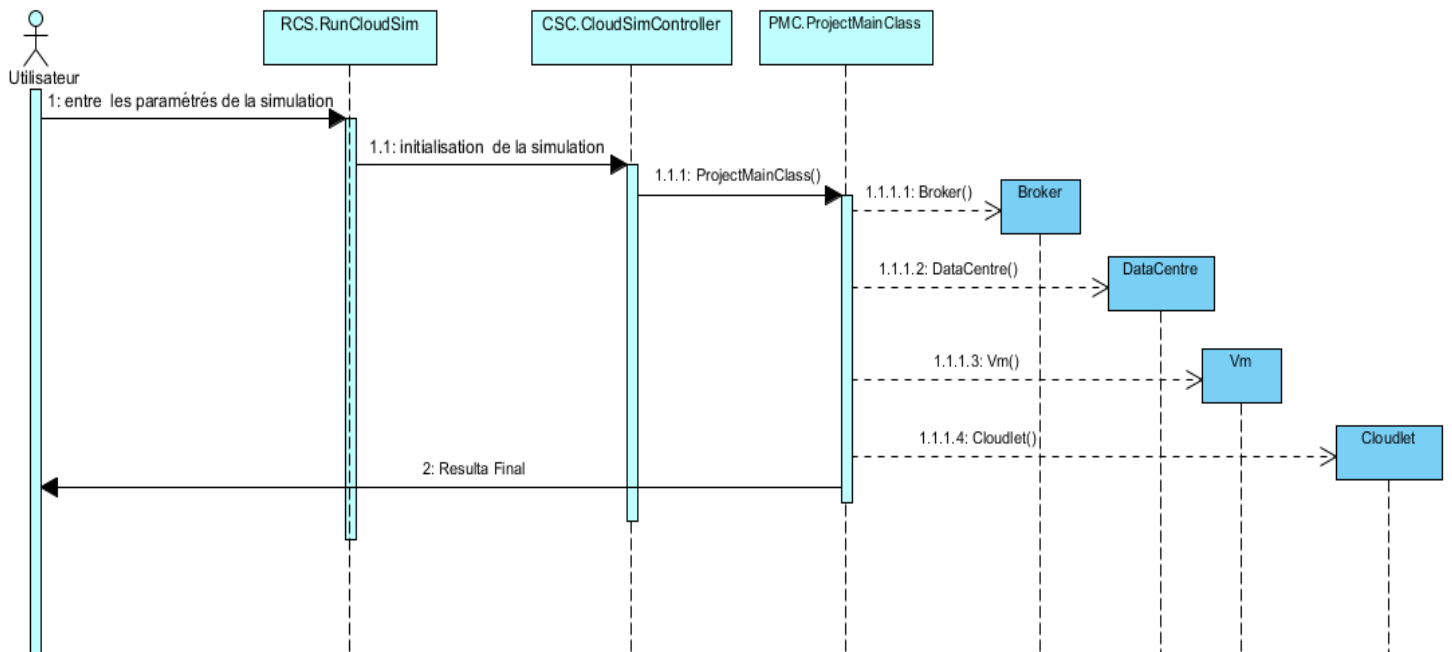


FIGURE 4.5 – Diagramme de séquence

## 4.4 Interface principale

La version de Cloudsim n'a pas d'interface graphique, son exécution se fait sur console, donc nous avons créé une interface qui facilite l'accès au simulateur. L'interface doit faire appel à Cloudsim ainsi qu'aux différentes classes qui se trouvent dans différents packages.

Dès le lancement de notre logiciel, la Figure 4.6 apparaît en premier aux utilisateurs, elle est constituée d'une barre de menu contenant les menus suivants : Le menu Fichier, Modifier, et le menu Aide. Chacun de ces menus contient un ensemble d'items qui vont être détaillés par la suite. Nous allons nous intéresser dans cette partie à la démonstration de notre application à travers un exemple en faisant référence à quelques interfaces graphiques obtenues.

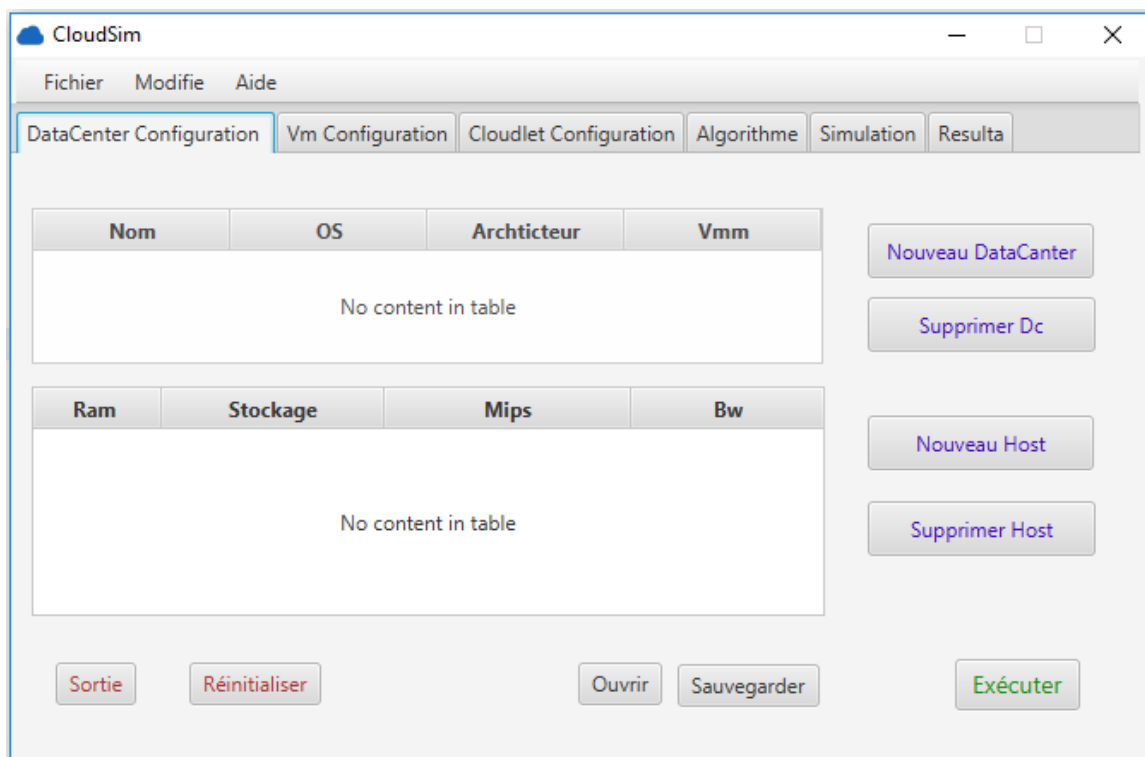


FIGURE 4.6 – Interface principale

### 4.4.1 Configuration des paramètres de simulation

Les Figures 4.7, 4.8, 4.9 et 4.10 montrent les fenêtres de configuration des paramètres de simulation, faite par l'utilisateur, elle contient 4 parties principales :

#### Datacenter configuration :

Cette étape consiste à faire entrer les paramètres nécessaires de **Datacenter Host** :

- **Nom** : le nom de la Datacenter.
- **OS** : le system d'exploitation .
- **Architcteur** : l'architcteur (X64,X86).
- **Vmm** : Virtual Machine Monitor ,qui contrôlent l'invité OS(guest OS) sur l'hôte.[17]
- **Ram** : taille de RAM de l'host (MB).
- **Stockage** : quantité de STOCKAGE de l'host (MB)(MB).
- **Mips** : Défini la classe Pe(Processing Element) qui représente l'unité CPU.
- **BW** : la bande passant de l'host.

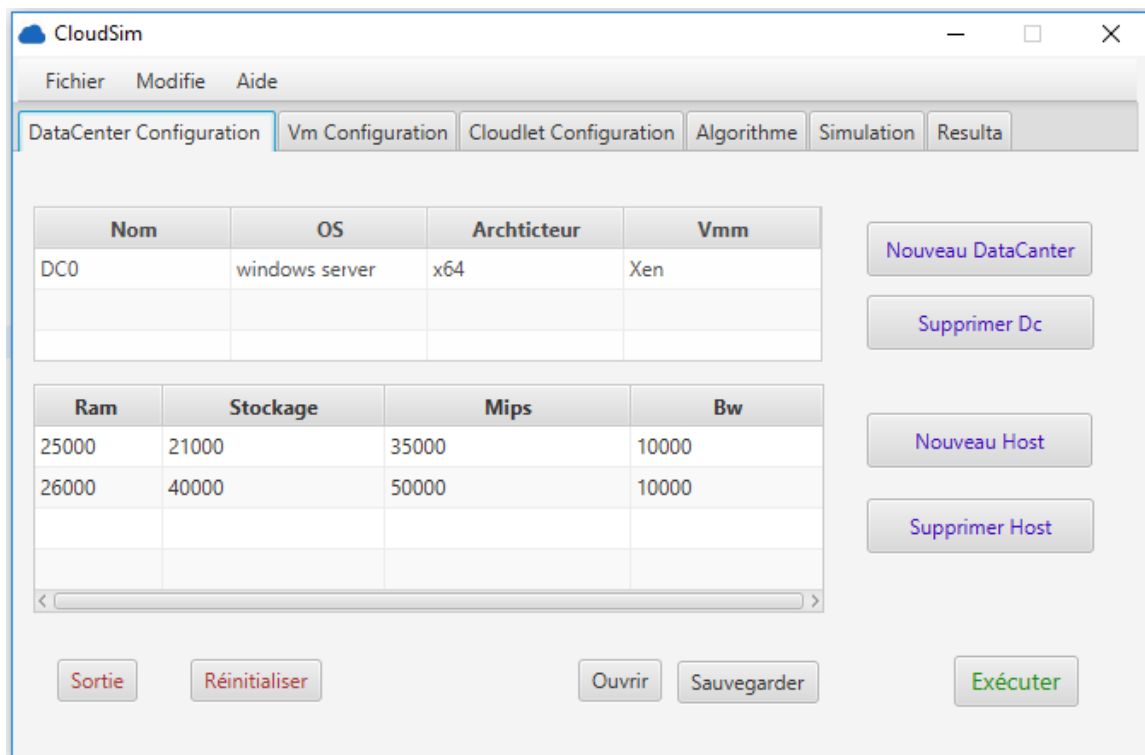


FIGURE 4.7 – Configuration du Datacenter

**VM configuration :**

Cette étape consiste à faire entrer les paramètres nécessaires de **Vm** :

- **Nom** : le nom de Vm.
- **Stockage** : quantité de STOCKAGE de Vm (MB).
- **Ram** : taille de RAM de Vm (MB).
- **Nombre de CPU** : le nombre de cpu de la VM.
- **Mips** : la vitesse de CPU de Vm.
- **BW** : la bande passant de VM.

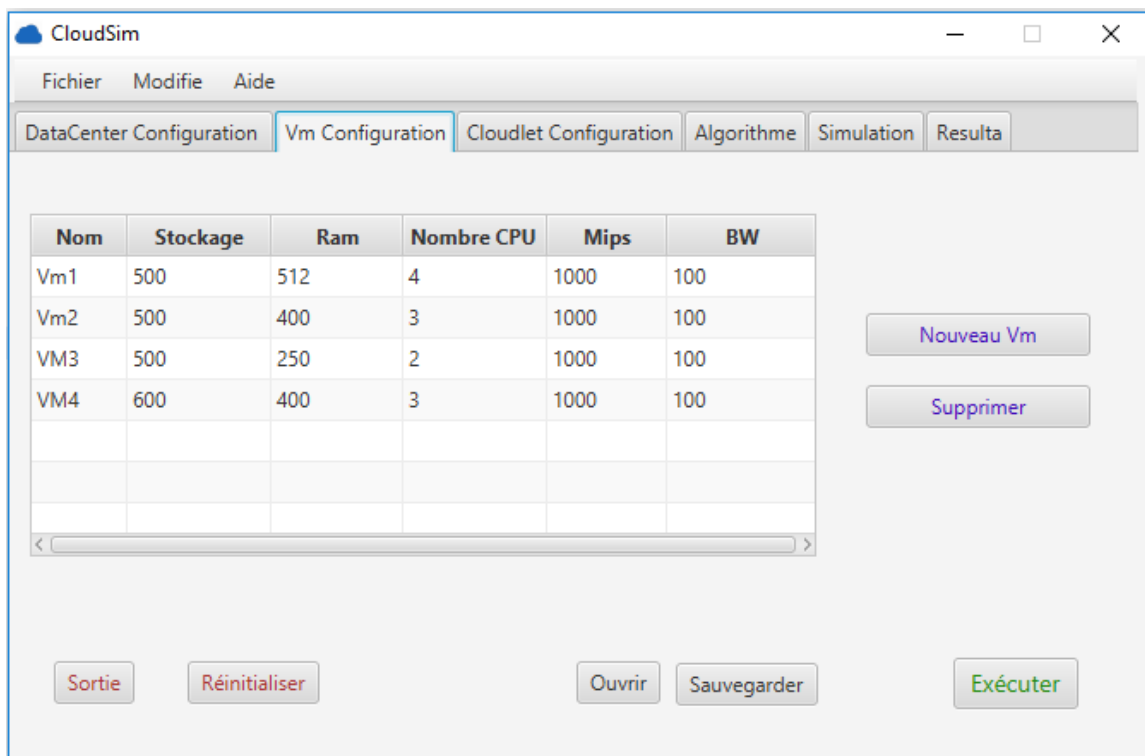


FIGURE 4.8 – Configuration du VM

**Cloudlet configuration :**

Cette étape consiste à faire entrer les paramètres nécessaires de **Cloudlet** :

- **Taille de Cloudlet** : la taille de la Cloudlet à exécuter dans un Vm.
- **taille du fichier** : la taille du fichier d'entrée de la Cloudlet avant l'exécution
- **taille du fichier de sortie** : la taille du fichier de sortie de la Cloudlet après l'exécution.

**Algorithme** : Cette étape consiste à choisir l'**algorithme** :

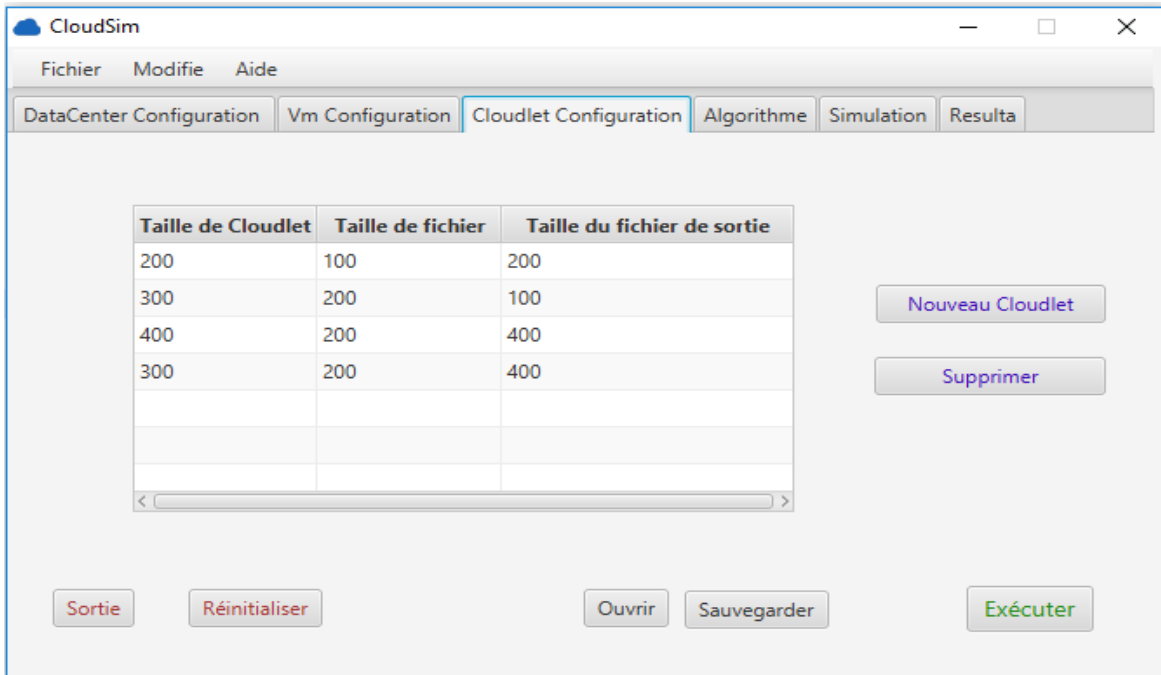


FIGURE 4.9 – Configuration du Cloudlet

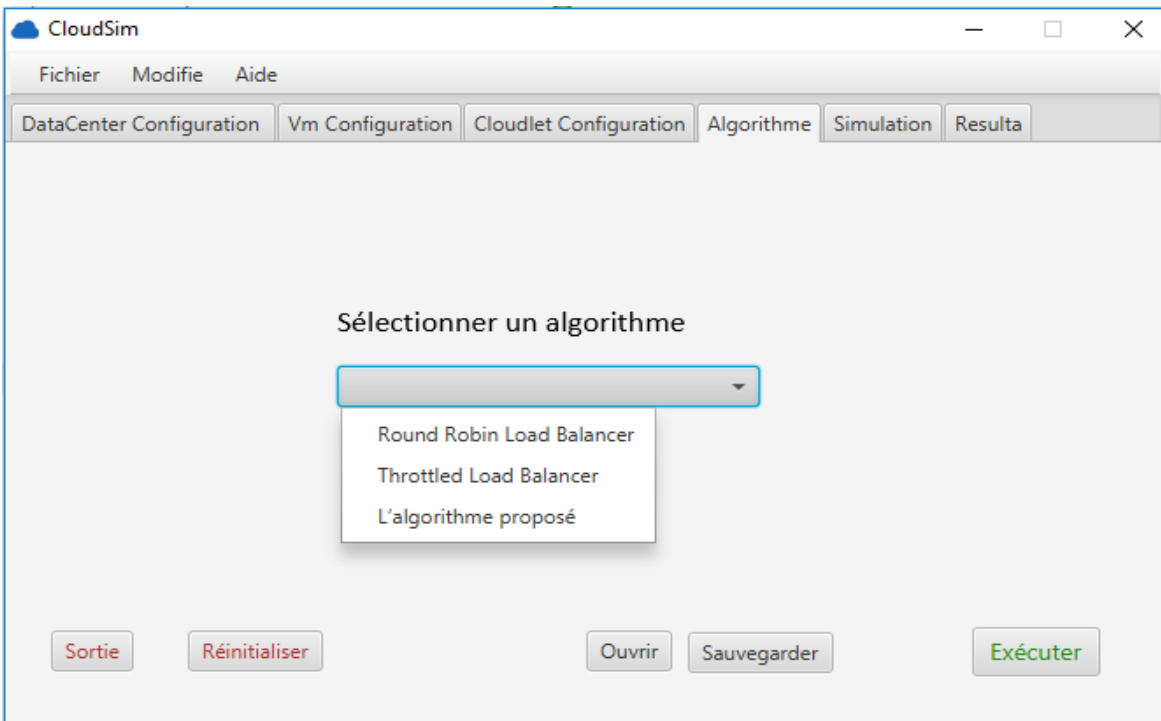


FIGURE 4.10 – le choix de l'algorithme

### 4.4.2 Lancement de la simulation

Après avoir personnalisé les paramètres de simulation, l'utilisateur peut lancer la simulation en cliquant sur le bouton *Exécuter* (voir Figure 4.11).

Après la création du Cloud et le lancement de la simulation, l'utilisateur peut visualiser les résultats et l'état de traitement de Cloudlets. Les résultats obtenus après la simulation, ils sont accessibles en cliquant sur les onglets **Résulta** et **Simulation**.

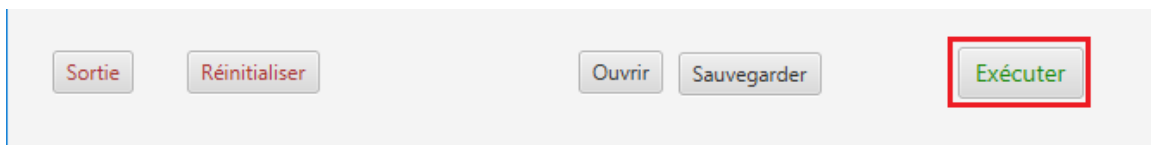


FIGURE 4.11 – Button pour lancer la simulation

## 4.5 Expérience et résultats

Afin d'entrer les paramètres nécessaires pour démarrer l'exécution de notre application, nous avons appliqué les trois algorithmes implémentés dans notre application à la configuration qui est représentée dans les figures 4.7, 4.8, 4.9 et 4.10 (décrit ci-dessus), après nous avons obtenu les résultats suivants qui sont présentés dans les figures 4.12, 4.13 et 4.14.

...	Statut	DC ID	VM ID	Temps	Start Time	Finish Time	Temp d'exuction	Temp de traitemenet
0	SUCCESS	2	0	22:08:32:413	22:08:32:398	22:08:32:411	13.00	15.00
1	SUCCESS	2	1	22:08:32:414	22:08:32:398	22:08:32:414	16.00	18.00
2	SUCCESS	2	2	22:08:32:414	22:08:32:398	22:08:32:414	16.00	18.00
3	SUCCESS	2	3	22:08:32:414	22:08:32:398	22:08:32:414	16.00	18.00

FIGURE 4.12 – Résulta de Round Robin Load Balancer

...	Statut	DC ID	VM ID	Temps	Start Time	Finish Time	Temp d'exuction	Temp de traitemenet
0	SUCCESS	2	0	22:09:36:801	22:09:36:788	22:09:36:799	11.00	12.00
1	SUCCESS	2	1	22:09:36:802	22:09:36:788	22:09:36:802	14.00	15.00
2	SUCCESS	2	2	22:09:36:802	22:09:36:788	22:09:36:802	14.00	15.00
3	SUCCESS	2	3	22:09:36:802	22:09:36:788	22:09:36:802	14.00	15.00

FIGURE 4.13 – Résulta de Throttled Load Balancer



...	Statut	DC ID	VM ID	Temps	Start Time	Finish Time	Temp d'exuction	Temp de traitement
0	SUCCESS	2	0	22:09:36:801	22:09:36:788	22:09:36:799	11.00	12.00
1	SUCCESS	2	1	22:09:36:802	22:09:36:788	22:09:36:802	14.00	15.00
2	SUCCESS	2	2	22:09:36:802	22:09:36:788	22:09:36:802	14.00	15.00
3	SUCCESS	2	3	22:09:36:802	22:09:36:788	22:09:36:802	14.00	15.00

FIGURE 4.14 – Résulta de L’algorithme proposé

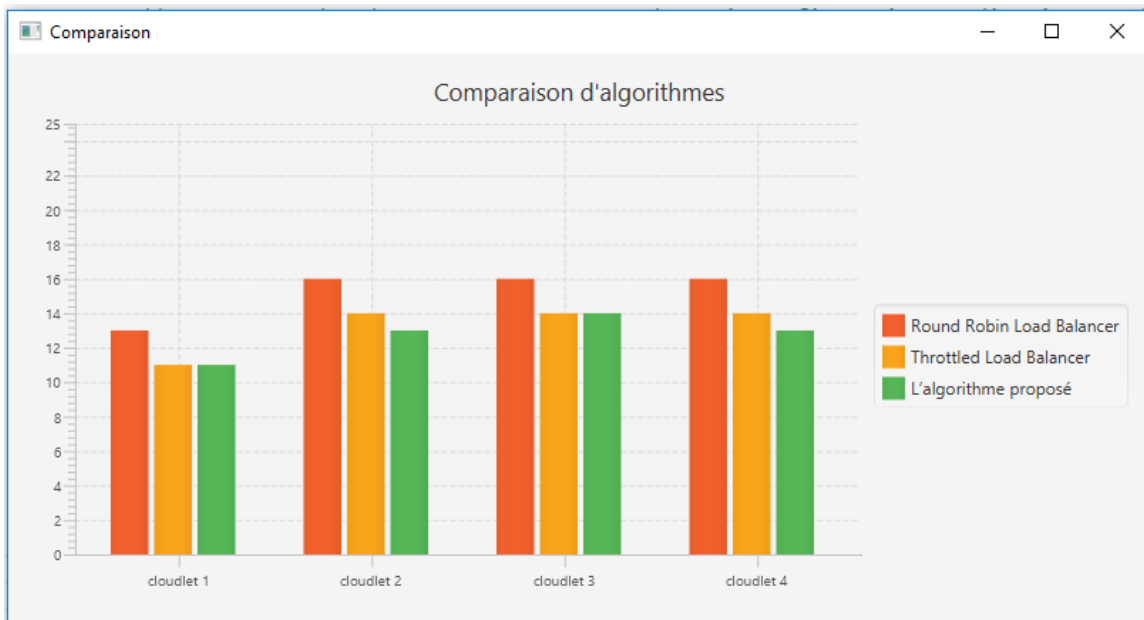


FIGURE 4.15 – Résulta de comparaison d’algorithmes par rapport le temps d’exécution

## 4.6 Conclusion

Dans ce chapitre nous avons montré en détail les interfaces principales de notre application, l'environnement d'application (matériel et logiciel). Ensuite nous allons voir la configuration des paramètres de simulation. Enfin nous allons voir une expérience et les résultats obtenus qui prouvent l'efficacité de notre algorithme proposé par rapport aux algorithmes Round Robin et Throttled Load Balancer. Les résultats montrent que notre algorithme minimise le temps d'exécution et le temps de traitement.

# Conclusion général

Les récents efforts visant à concevoir et développer des technologies Cloud tout en focalisant sur la définition de nouvelles méthodes, des politiques et des mécanismes efficaces de gestion des infrastructures et d'allocation des ressources. Ce mémoire avait pour objectif de proposer un Algorithme d'équilibrage de charge dans le cloud computing . Dont le but est de satisfaire les besoins des utilisateurs cloud, tout en proposant l'allocation des ressources la plus efficace qui permette de réduire le coût de service ou d'application.

En début nous avons détaillé les notions de base du cloud computing et présenté les caractéristiques, les modèles de service (IaaS, PaaS, SaaS), et les modèles de déploiement. Ensuite nous avons vu le mécanisme d'allocation des ressources , le problème d'équilibrage de charge dans les environnements cloud computing et les différents algorithmes pour résoudre ce problème. Puis nous avons proposé un algorithme d'équilibrage de charge qui est basé sur l'algorithme Throttled Load Balancer , notre algorithme basé sur la métrique de temps d'exécution. A la fin de notre mémoire nous avons exposé la phase de réalisation de notre application , les environnements d'application (matériel et logiciel) et nous avons présenté une expérience et les résultats obtenus qui montrent que notre algorithme minimise le temps d'exécution et le temps de traitement.

# Bibliographie

- [1] The NIST Definition of Cloud Computing . NIST. 2014. SP-800 145. [cf. p. [iv](#), [5](#)]
- [2] Introduction to Cloud Computing. Dialogic Corporation. 2017. [cf. p. ]
- [3] CLOUD COMPUTING – An Overview . Torry Harris. [cf. p. [3](#), [15](#)]
- [4] Cloud Services for Dummies. IBM Limited Edition. Judith Hurwitz, Marcia Kaufman, et Dr. Fern Halper. [cf. p. [8](#), [9](#)]
- [5] [www.hebergeurcloud.com/definition-cloud-computing-selon-nist/](http://www.hebergeurcloud.com/definition-cloud-computing-selon-nist/). 2/11/2018. 20 :24. [cf. p. [iv](#), [12](#)]
- [6] CLOUD COMPUTING Généralités et enjeux V0 . Benjamin LISAN. 20/03/2015. [cf. p. [iv](#), [iv](#), [4](#), [6](#), [9](#), [12](#), [13](#)]
- [7] Abdelkader El Mahdaouy, Mohammed Oumsis . « Evaluation et amélioration de performances des algorithmes d'équilibrage de charges dans un environnement Cloud Computing ». Laboratoire LIM, Département d' Informatique, Faculté des Sciences Dhar Mahraz, Université Sidi Mohamed Ben Abdellah, Fès, Maroc. [cf. p. [22](#), [24](#)]
- [8] Youssef FAHIM, Elhabib BEN LAHMAR, El houssine LABRJI, Ahmed ED-DAOUI . «The load balancing based on the estimated finish time of tasks in cloud computing ». Hassan II Universit - Mohammedia - Casablanca. [cf. p. [17](#), [20](#), [21](#), [22](#)]
- [9] Gaurang Sangani , Shreyas Patel. «Load Balancing Algorithm in Cloud Computing For Heterogeneous Configuration » . International Journal of Advanced Engineering Research and Technology. May 2015 [cf. p. [6](#), [7](#), [21](#)]
- [10] Mohamed Raouf HABES . « ALLOCATION DE RESSOURCES DANS LE CLOUD COMPUTING ». These Présentée En Vue De L'obtention Du Diplôme De : Doctorat 3ème Cycle Lmd 2016. UNIVERSITE BADJI MOKHTAR – ANNABA. [cf. p. [10](#), [23](#), [24](#)]
- [11] Sandeep Shama, Sarabjit Singh, Meenashi Sharma. «Performance Analysis of Load Balancing Algorithms». World Academy of Science, Engineering and Technology. [cf. p. ]

- [12] Einollah Jafarnejad Ghomi, Amir Masoud Rahmani, Nooruldeen Nasih Qader. «Load-balancing Algorithms in Cloud Computing : A Survey». Journal of Network and Computer Applications. 2017. [cf. p. [iv](#), [20](#), [22](#), [23](#), [24](#), [27](#)]
- [13] Amritpal Singh. «A Review of Existing Load Balancing Techniques in Cloud Computing». International Journal of Advanced Research in Computer Engineering and Technology. 2015 [cf. p. [28](#)]
- [14] Nitika Kadam. « VIRTUAL MACHINE LOAD BALANCING USING CLOUD SIM IN PUBLIC CLOUD». International Journal of Research In Science and Engineering. Volume : 3 Issue : 4 July-August 2017. [cf. p. [6](#), [7](#)]
- [15] <https://www.hebergeurcloud.com/les-technologies-du-cloud-computing/.17/06/2019.22:40>. [cf. p. [iv](#), [7](#)]
- [16] Documentation technique Définition Cloud Computing. Laurent MILTGEN-DELINCHAMP ,Elgon S.A . 2017 . [cf. p. [11](#)]
- [17] Bohar Singh, Pawan Luthra and Bindu Bala. «Performance Analysis of VMM Using LINPACK and CLOUDSIM». Computer Science and Engineering, Shaheed Bhagat Singh State Technical, Campus, Ferozepur, India. [cf. p. [47](#)]
- [18] V.Vinothina, Sr.Lecturer , Dr.R.Sridaran Dean ,Dr.Padmavathi Ganapathi. «A Survey on Resource Allocation Strategies in Cloud Computing». International Journal of Advanced Computer Science and Applications. 2012. [cf. p. [19](#)]
- [19] N. Asha<sup>1</sup> ,Dr.G.Raghavendra Rao. «A Review on Various Resource Allocation Strategies in Cloud Computing». International Journal of Emerging Technology and Advanced Engineering. [cf. p. [19](#)]
- [20] G. Lee. «Resource Allocation and Scheduling in Heterogeneous Cloud Environments». Electrical Engineering and Computer Sciences University of California at Berkeley, Technical. May 2012. [cf. p. [iv](#), [18](#), [19](#)]
- [21] S. François. “La Virtualisation”. Projet de recherche et communication scientifique, université libre de Bruxelles. 2010. [cf. p. [7](#)]

# Résumé

Le Cloud Computing fait référence à l'utilisation des capacités de calcul des ordinateurs distants, où l'utilisateur dispose d'une puissance informatique considérable sans avoir à posséder des unités puissantes. L'allocation de ressources est le processus d'affectation des tâches aux ressources disponibles sur la base des caractéristiques et des conditions des tâches. C'est un aspect important dans le fonctionnement efficace du Cloud. L'équilibrage de charge est un problème de gestion de ressources qu'un fournisseur de services doit assurer afin de garantir une qualité de services satisfaisante. Les algorithmes d'équilibrage de charge trouvés pour résoudre ce problème. Notre mémoire de fin d'étude présente une proposition d'un algorithme qui donne un résultat mieux que les autres algorithmes que nous avons présentés.

**Mots clés :** *Cloud Computing , L'allocation de ressources , L'équilibrage de charge.*

# Abstract

Cloud Computing refers to the use of computing capabilities of remote computers, where the user has considerable computing power without having to have powerful units. Resource allocation is the process of assigning tasks to available resources based on the characteristics and conditions of the tasks, this is an important aspect in the efficient operation of the Cloud. Load balancing is a resource management problem that a service provider must provide to ensure a satisfactory quality of service, the load balancing algorithms found to solve this problem. Our thesis presents a proposal for an algorithm that gives a better result than the other algorithms we have.

**Keywords :** *Cloud Computing , Resource allocation , Load balancing*