REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohammed Khider-BISKRA.

Faculté des sciences exactes et des sciences de la nature et de la vie

Département d'informatique

Mémoire
en vue de l'obtention du diplôme de Magister en informatique

Option: Data mining and multimédia

Une approche basée agents pour l'allocation des ressources dans le Cloud Computing

Présenté par : FAREH M^{ed} el-kabir.

Membres de jury:

Dr. BENNOUI Hammadi	**Maître de conférences A à l'université de Biskra**	Président
Pr. KAZAR Okba	**Professeur à l'université de Biskra**	Rapporteur
Dr. TERISSA Labib Sadek	**Maître de conférences A à l'université de Biskra**	Examinateur
Dr. DERDOUR Makhlouf	**Maître de conférences A à l'université de Tébessa**	Examinateur

Année Universitaire: 2014-2015

Dédicace

À mon père et ma mère.

À ma petite famille.

À mes frères et mes sœurs.

Remerciements

Je remercie, tout d'abord, dieu qui m'a éclairé le chemin du savoir.

Ensuite, je tiens à remercier mon encadreur le ***Professeur KAZAR Okba*** *pour ses conseils, ses orientations et son aide.*

Je remercie également les membres de jury, ***Dr. BENNOUI Hammadi, Dr. TERISSA Labib Sadek*** *et* ***Dr. Derdour Makhlouf*** *d'avoir accepté de juger ce travail.*

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

Résumé

Le cloud computing a émergé au cours de la dernière décennie pour être largement adopté aujourd'hui dans plusieurs domaines de l'informatique. Il consiste à proposer les ressources sur le marché sous forme de services qui peuvent être consommés de manière souple et transparente. Dans ce travail, nous traitons le problème d'allocation de ces ressources, un des enjeux majeurs du cloud. A tout instant, l'utilisateur peut s'adresser au fournisseur de ressources pour qu'une nouvelle ressource lui soit allouée ou pour relâcher une ressource dont il n'a plus l'utilité. L'allocation et la libération sont des opérations qui, à défaut d'être instantanées, sont réalisées dans un laps de temps relativement court.

Le travail de recherche mené dans ce mémoire consiste à proposé une approche d'allocation des ressources dans le cloud computing basée sur le paradigme d'agent, une architecture en couche pour la modélisation d'allocation des ressources dans le cloud a été proposée, Nous avons opté pour l'approche UML pour décrire les principales fonctionnalités de notre système. Finalement, nous avons implémenté notre approche pour la valider expérimentalement.

Mots clés: cloud computing, allocation des ressources, agent.

Abstract

Cloud computing has emerged over the last decade to be widely adopted today in several areas of computing. It consists to provide resources on the market as a service that can be consumed in a flexible and transparent manner. In this work, we address the problem of allocation of these resources, one of the major challenges of the cloud. At any time, the user can go to the resource provider for a new resource to be allocated to him or to release a resource that is no longer used. The allocation and free up are operations that, by default be instantaneous, are carried out in a relatively short period of time.

The research conducted in this thesis consists of proposing a resources allocation approach in cloud computing based on the paradigm of agent, an architecture of layer for modeling resources allocation in the cloud has been proposed. We have opted for the UML approach to describe the main features of our system. Finally, we have implemented our approach to validate it experimentally.

Keywords: cloud computing, resource allocation, agent.

ملخص

برزت الحوسبة السحابية خلال العقد الماضي، ليتم اعتمادها على نطاق واسع اليوم في عدة مجالات حوسبة، وتتمثل في توفير الموارد و تسويقها علي شكل خدمات، ليتم استهلاكها بطريقة مرنة وشفافة. في هذا العمل، نعالج مشكل تخصيص هذه الموارد، واحد من التحديات الرئيسية للحوسبة السحابية، حيث يمكن للمستخدم في أي وقت، اللجوء إلى المزود بالموارد لكي يخصص له مورد جديد أو لكي يحرر مورد لم يعد في حاجة إليه. التخصيص و إلغائه عبارة عن عمليات لحظية، يجب أن تتم في فترة زمنية قصيرة نسبيا.

العمل البحثي الذي أجري في هذه المذكرة يتمثل في اقتراح نموذج لتخصيص الموارد في الحوسبة السحابية يرتكز على منهج الوكلاء،حيث تم اقتراح بنية طبقية لنمذجة تخصيص الموارد في السحابة، و قد اخترنا نهج UML لوصف الوظائف الرئيسية لنظامنا. وأخيرا، قمنا بتنفيذ منهجنا للتحقق من صحته تجريبيا.

الكلمات المفتاحية: الحوسبة السحابية، تخصيص الموارد، وكيل.

Table des matières

LISTE DES FIGURES

LISTE DES TABLES

Introduction Générale

L'informatique dans les nuages (appelé cloud computing en anglais et dans la suite du manuscrit) s'est imposé ces dernières années comme un paradigme majeur d'utilisation des ressources informatiques. Ces ressources mises à disposition par le fournisseur cloud sont accessibles au travers d'interfaces programmatiques via un réseau informatique (généralement via Internet). A tout instant, l'utilisateur peut s'adresser au fournisseur de ressources pour qu'une nouvelle ressource lui soit allouée ou pour relâcher une ressource dont il n'a plus l'utilité. L'allocation et la libération sont des opérations qui, à défaut d'être instantanées, sont réalisées dans un laps de temps relativement court. De nombreuses stratégies sont utilisées pour fournir des ressources aux utilisateurs et elles sont regroupées sous le terme "allocation des ressources".

Du point de vue de l'utilisateur, l'allocation dynamique des ressources, associée au modèle de paiement à l'usage, permet de diminuer le gaspillage de ressources inutilisées. Et du point de vue du fournisseur cloud, le principal intérêt réside dans le facteur d'échelle qu'il introduit en créant de grands ensembles de ressources accessibles au travers d'un réseau informatique (généralement via Internet). Il peut ainsi regrouper géographiquement ces ressources en fonction des coûts liés à leur hébergement (i.e. loyers), leur propre consommation énergétique ou celles des systèmes de refroidissement qu'elles induisent. En outre, le recours à la virtualisation permet de consolider le nombre de serveurs physiques nécessaires au fonctionnement d'un ensemble de services ou d'applications.

Dans le même temps, les systèmes multi-agents (SMA) représentent un autre paradigme de l'informatique distribuée basée sur multiples agents en interaction entre eux, ils sont très adaptés pour modéliser les phénomènes dans lesquels les interactions entre diverses entités sont assez complexes pour être appréhendées par les outils de modélisation classique. Les agents autonomes et flexibles ainsi que les SMA sont des outils appropriés pour négocier l'accès d'utilisateur, la découverte des services, et l'exploitation des ressources de cloud computing.

L'initiative de formuler une nouvelle discipline appelée cloud computing basée agents est présenté dans **[19]**, l'objectif de cette discipline est de fournir des solutions de cloud computing fondées sur la conception et le développement d'agents logiciels qui peuvent améliorer l'utilisation des ressources cloud, la gestion et la découverte des services, la négociation d'SLA et la composition de services. Les agents autonomes pourraient rendre les clouds plus intelligents dans leurs interactions avec les utilisateurs et plus efficace dans l'allocation de puissance de traitement et le stockage pour différentes applications.

'D. Talia' dans son article «Clouds Meet Agents: Toward Intelligent Cloud Services» a proposé deux approches pour l'intégration d'agents et les systèmes de cloud computing **[19]**:

- ✓ La première est basée sur le principe que la flexibilité, l'intelligence, la pro-activité et l'autonomie d'agent peuvent aider les plates-formes de cloud computing pour offrent

des solutions, des fonctionnalités, et des services intelligents qui ne sont pas disponibles dans les infrastructures de cloud actuel;

- ✓ La deuxième est basée sur l'idée que les infrastructures cloud peuvent offrir une plate-forme idéale sur laquelle fonctionnent les simulations, les applications et les systèmes à base d'agents, en raison de ses vastes ressources de traitement et de stockage.

L'objectif de ce travail est de concevoir un système multi-agents pour l'allocation des ressources dans le cloud computing. En effet, les systèmes multi-agents s'adaptent bien à la conception d'un système d'allocation des ressources où chaque membre doit gérer et échanger ses connaissances et collaborer avec les autres afin de réaliser ses buts. En plus, dans des tels environnements ouverts, dynamiques et complexes, il y a un besoin de distribuer les données, le contrôle ainsi que l'expertise ce qui rend l'utilisation des systèmes multi-agents bénéfique.

Pour l'essentiel, ce mémoire est structuré autour quatre chapitre :

- Le premier chapitre présente en détail au lecteur les deux domaines dans lesquels s'intègrent les travaux de ce mémoire. Il se compose en trois sections: une première section présente ce qu'est le cloud computing et ces différents concepts, une deuxième section qui introduite les notions d'agents et de systèmes multi-agents et il se termine en troisième section par un aperçu sur l'intégration des agents et le cloud computing.
- Le deuxième chapitre dresse un état de l'art sur l'allocation des ressources dans le cloud computing, ainsi qu'un tour d'horizon sur les travaux existants.
- Le troisième chapitre est destiné à l'étape de conception de notre approche proposée qui est devisée en deux parties, la conception globale et la conception détaillée.
- Le quatrième chapitre sera consacré à l'implémentation de notre approche, pour cela, l'environnement de développement, les choix techniques, la présentation de l'application ainsi que les résultats obtenus seront tous présentés dans ce chapitre.
- Nous terminons le mémoire par une conclusion générale qui synthétise notre projet et présente les perspectives de notre travail.

CHAPITRE I

LE CLOUD COMPUTING ET LES SYSTEMES MULTI-AGENTS

1 Introduction

Depuis quelques années, un nouveau paradigme nommé cloud computing révolutionne la façon dont les entreprises et les particuliers accèdent à des ressources informatiques telles que la puissance de calcul, la capacité de stockage...etc. Adoptant une approche élastique d'utilisation à la demande, le cloud computing est une solution très intéressante pour l'externalisation et la simplification des services informatiques.

D'autre part, de nos jours, les systèmes multi-agents ont pris une place de plus en plus importante en informatique, que ce soit dans le domaine de l'intelligence artificielle, dans celui des systèmes distribués, de la robotique et du génie logiciel. C'est une discipline qui s'intéresse aux comportements collectifs produits par les interactions de plusieurs entités autonomes et flexibles appelés agents.

Ce chapitre présente en détail les deux domaines dans lesquels s'intègrent les travaux de ce mémoire. Tout d'abord, en section 1, nous définissons ce qu'est le cloud computing et ces différents concepts. Ensuite, la section 2 introduit les notions d'agents et de systèmes multi-agents et il se termine en section 3 par un aperçu sur l'intégration des agents et le cloud computing.

2 Définitions et Concepts du cloud computing

2.1 Définition du contexte

Afin de comprendre les fondements de cloud computing, que l'on pourrait traduire en français par « informatique dans le nuage », Commençons par définir cette infrastructure «le nuage», De façon métaphorique le nuage en question est assimilé à Internet, ou plutôt à l'infrastructure matérielle et logicielle qui permet son fonctionnement **[1]**. Il est traditionnellement utilisé pour schématiser des réseaux de télécommunications, où il y représente la partie du réseau que l'on ne connaît pas, que l'on ne sait pas définir, mais qui permet d'interconnecter tous les réseaux déployés à travers le monde. L'exemple le plus courant est illustré par le schéma donné dans la **Figure 1.** Ce schéma représente un réseau informatique local (LAN), par exemple celui d'une petite entreprise. L'infrastructure gérée par l'entreprise en question (c-à-d. ordinateurs, commutateurs et routeurs) y est détaillée, alors que celle associée à Internet est représentée par un nuage.

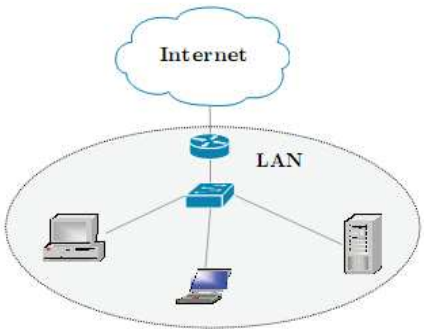


Figure 1. Métaphore du nuage **[1].**

L’idée de cette représentation est d’identifier la partie du schéma ou de la solution à représenter qui est à la charge d’un tiers. Une question se pose alors: si cette partie du réseau est gérée par un tiers, alors pourquoi la détailler? C’est l’idée soulevée par Velte et coll. dans le livre «Cloud Computing, a practical approach» (une approche pratique de l’informatique dans le nuage), et c’est cette idée qui, probablement, correspond le mieux à la place de la métaphore du nuage au sein de cloud computing **[1]**. Aussi cloud computing généralise ce concept de « partie du réseau que l’on n’administre pas, que l’on ne gère pas », à l’ensemble des ressources informatiques qui peuvent être accessibles via Internet en tant que service. Les détails techniques d’implantation de ces services, de leur hébergement ou de leur maintenance concernent alors uniquement le fournisseur, qui en contrepartie facture leur utilisation. Les services informatiques sont alors distribués publiquement.

2.2 Chronologie

D'un aspect purement technique, le Cloud Computing est loin d'être en soi une technologie nouvelle, le Cloud Computing provient de l'aboutissement de plusieurs technologies existantes antérieurement.

Ces premières traces remontent aux années soixante avec les mainframes où les utilisateurs accédaient depuis leurs terminaux a des applications fonctionnant sur des systèmes centraux, a cette époque l'expression Cloud n'était pas encore née, c'est avec l'apparition d'Internet que les architectes de réseaux la schématisaient par un nuage dans leurs croquis, on parlait alors de « the cloud ».

Par la suite, au début des années 2000, sont apparus des hébergeurs et des fournisseurs d'applications en ligne ASP (Application Service Providers) qui s'appuyaient sur une forme d'externalisation «Software as a Service». Et ce n'est que peu après, qu'Amazon Web Services «AWS», oriente vers les entreprises, et Google, oriente vers le grand public, font émerger le marché du cloud computing, suivis après par les autres éditeurs comme Microsoft et Oracle **[2]**.

2.3 Vers une définition de cloud computing

Le cloud computing est un domaine vaste et dynamique, si bien qu’à l’heure d’aujourd’hui personne ne prétend en posséder une définition claire et absolue, En effet, la plupart des définitions attribuées à cette discipline semblent se concentrer seulement sur certains aspects de la technologie, Nous citons deux définitions:

- *« Le cloud computing est un modèle qui permet un accès réseau à la demande et pratique à un pool partagé de ressources informatiques configurables (telles que réseaux, serveurs, stockage, applications et services) qui peuvent être provisionnées rapidement et distribuées avec un effort minimum de gestion ou d’interaction avec le fournisseur de services, ce modèle de cloud est composé de cinq caractéristiques essentielles, trois modèles de services et quatre modèles de déploiement »* **[3].** Cette définition est détaillée tout au long de cette section.

- *« le cloud computing peut également être vu comme un réseau informatique distribué composé de serveurs virtualisés, dont les ressources peuvent être approvisionnées de manière dynamique et reposant sur un contrat de service entre un fournisseur et un client, le SLA (Service Level Agreement)»* **[4]**.

En guise de synthèse des différentes propositions données dans la littérature, Willy M-M dans ca thèse de doctorat en 2011 a proposé la définition suivante de cloud computing, qui l'on trouve la plus adéquate :

- *« Le cloud computing s'appuie sur une infrastructure (le cloud) composée d'un grand nombre de ressources virtualisées (par exemple : réseaux, serveurs, stockage, applications ou services), distribuées dans le monde entier. Ces ressources peuvent être allouées, puis relâchées rapidement, avec des efforts de gestion minimaux et avec peu d'interactions entre le client et le fournisseur. Aussi, cette infrastructure peut être dynamiquement reconfigurée pour s'ajuster à une charge de travail variable (passage à l'échelle). Finalement, les garanties de prestation offertes par l'informatique dans le cloud prennent typiquement la forme de contrats de niveau de service »* **[1]**.

Note : Il se peut que l'on aille du mal à différencier entre Cluster, Grid et leur successeur le Cloud (**Table 1**):

- Le Cluster est un groupe d'ordinateurs généralement relies par un réseau local (LAN), tandis que les deux autres sont plus échelonnables et peuvent être distribues géographiquement.
- Grid Computing, il est plutôt utilise en recherche, une sorte de superordinateur distribue sur plusieurs machines il est plus oriente «resource on demand». La différence avec le cloud computing vient en réalité de son utilisation.

	Cluster	Grid	Cloud
On-demand self-service	No	No	Yes
Broad network access	Yes	Yes	Yes
Resource pooling	Yes	Yes	Yes
Rapid elasticity	No	No	Yes
Measured service	No	Yes	Yes

Table 1. Comparaison entre Cluster, Grid et Cloud Computing[1]

2.4 Caractéristiques principales

Le cloud computing est une nouvelle façon de délivrer les ressources informatiques, et non une nouvelle technologie. Il se caractérise par **[3]**:

Un accès en libre service à la demande: un client pourra commander des ressources informatiques en fonction de ses besoins. Les ressources informatiques sont fournies d'une manière entièrement automatisée.

[1] http://www.cloud-competence-center.com/understanding/difference-cloud-cluster-grid/

Une mise en commun des ressources: les ressources informatiques sont mises à la disposition des clients sur un modèle multi-locataires, avec une attribution dynamique des ressources physiques et virtuelles en fonction de la demande. Le client n'a généralement ni contrôle, ni connaissance sur l'emplacement des ressources, mais est en mesure de le spécifier à un plus haut niveau d'abstraction (pays, état ou centre de données).

Une élasticité rapide: les capacités informatiques mises à disposition du client peuvent être ajustées (augmenter ou diminuer) rapidement (quelques minutes ou quelques secondes) en fonction des besoins et/ou de la charge et de façon automatique dans certains cas.

Un service mesuré en permanence: Les ressources consommées sont contrôlées et communiquées au client et au fournisseur de service de façon transparente. Cela garantit un niveau de disponibilité adapté aux besoins spécifiques des clients.

2.5 Modèles de services de cloud

Il existe aujourd'hui trois modèles de base qui peuvent être utilisés sur un cloud selon les ressources allouées aux clients. La **Figure 2** montre ces trois modèles ainsi que le modèle traditionnel et les ressources fournies aux clients dans chaque modèle. On peut ainsi distinguer :

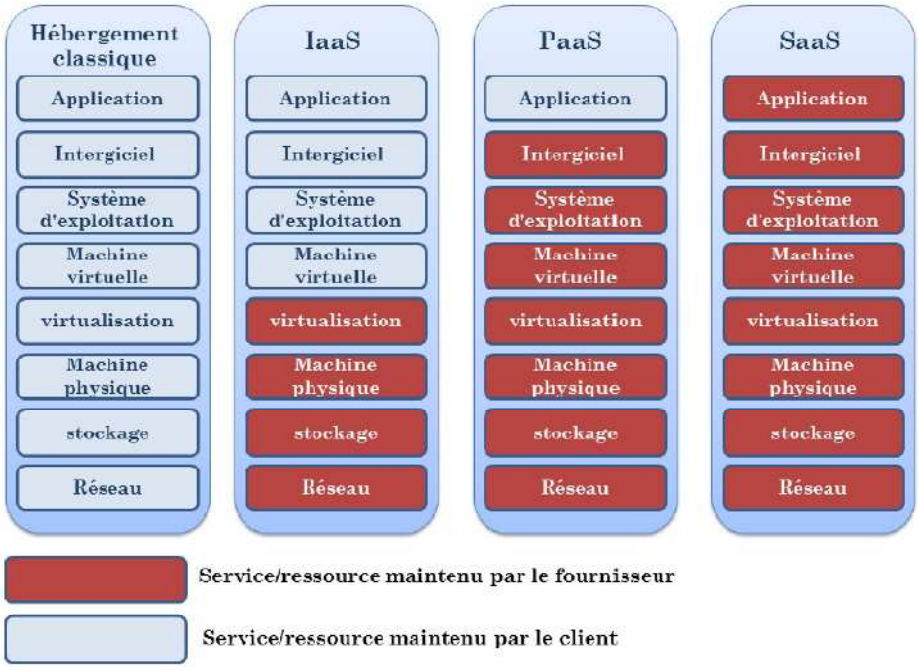


Figure 2. Modèles de cloud **[5]**

2.5.1 IaaS (Infrastructure as a service)

Le fournisseur met à disposition uniquement des ressources matérielles (machine, réseaux, disque) et le client gère le système et les applications **[5]**. On retrouve dans cette catégorie Amazon Elastic Compute Cloud[2] (EC2) et IELO Cloud[3] comme exemples d'IaaS.

[2] aws.amazon.com/ec2
[3] http://www.ielo.net/solutions/externalisation/cloud/

- Avantage: grande flexibilité, contrôle total des systèmes (administration à distance par SSH ou Remote Desktop…), qui permet d'installer tout type de logiciel métier.
- Inconvénient: besoin d'administrateurs système comme pour les solutions de serveurs classiques sur site.

2.5.2 PaaS (Platform as a service)

Le client ne gère et ne contrôle ici que ses applications métier, le reste est délégué au fournisseur de services **[5]**. Google App Engine[4] et Windows Azure[5] sont des exemples de PaaS.

- Avantage: le déploiement est automatisé, pas de logiciel supplémentaire à acheter ou à installer.
- Inconvénient: limitation à une ou deux technologies (ex.: Python ou Java pour Google AppEngine, .NET pour Microsoft Azure…). Pas de contrôle des machines virtuelles sous-jacentes. Convient uniquement aux applications Web.

2.5.3 SaaS (Software as a service)

C'est l'ultime niveau de transfert vers le fournisseur, le client ne gère plus que ses données métier, il utilise les applications fournies par le fournisseur. Il s'agit d'utiliser le logiciel à la demande **[5]**. Nous pouvons citer Google Apps[6] ou encore CORDYS[7] comme exemples de SaaS.

- Avantage: plus d'installation, plus de mise à jour (elles sont continues chez le fournisseur), plus de migration de données etc. Paiement à l'usage. Test de nouveaux logiciels avec facilité.
- Inconvénient: Pas de contrôle sur le stockage et la sécurisation des données associées au logiciel. Réactivité des applications Web pas toujours idéale.

La **table 2** présente les avantages et les inconvénients de chaque service cloud :

	Avantage	**inconvénient**
SaaS	-pas d'installation -plus de licence -migration	-logiciel limité -sécurité -dépendance des prestataires
PaaS	-pas d'infrastructure nécessaire -pas d'installation -environnement hétérogène	-limitation des langages -pas de personnalisation dans la configuration des machines virtuelles
IaaS	-administration -personnalisation -flexibilité d'utilisation	-sécurité -besoin d'un administrateur système

Table 2. Avantages et Inconvénients des services cloud **[6].**

Les deux modèles (IaaS et PaaS) sont généralement utilisés dans le cadre d'une relation entre entreprises (Business To Business (B2B)). Le modèle SaaS est au contraire utilisé

[4] developers.google.com/appengine
[5] http ://www.microsoft.com/windowsazure/
[6] www.google.com/intx/fr/enterprise/apps/business/
[7] http://www.rightscale.com/products-and-services/services/architect-cloud-applications

généralement dans des relations entre une entreprise et le grand public. (Business To Consumer (B2C)) **[5]**.

En fait, l'expansion du cloud fait naître le modèle de XaaS "signifiant Tout comme un service" (everything-as-a-service en anglais). On voit par exemple apparaitre la notion de Human as a Service (HuaaS) qui caractérise une couche supérieure a SaaS correspondant a une ressource humaine élastique. Cette intelligence « artificielle » peut, par exemple, servir pour des décisions arbitraires comme choisir les vidéos les plus intéressantes à afficher (pour un système comme Youtube) **[7]**.

2.6 Concepts liés au service cloud

Il nous semble important de définir les trois concepts liés au service de cloud :

- ✓ **Qualité de service (Quality of Services QoS) :** La qualité de service désigne la capacité d'un service à répondre par ses caractéristiques aux différents besoins de ses utilisateurs ou consommateurs **[5]**.
- ✓ **Accord de niveau de service (Service Level Agreement SLA) :** Un SLA est la formalisation d'un accord négocié entre deux parties. C'est un contrat qui définit la QoS requise entre un prestataire et un client. Il précise de manière claire les objectifs de performance et de qualité. Le fournisseur du service s'engage ainsi par contrat sur une disponibilité de l'outil vis-à-vis des utilisateurs **[5]**.
- ✓ **Objectifs de niveau de services (Service Level Objectives SLOs) :** Les SLOs définissent des caractéristiques des services qui seront fournis à un client en termes qualitatifs. Ils sont considérés comme des moyens permettant de mesurer la QoS dans l'objectif d'éviter le malentendu entre les deux parties. Ils peuvent être composés d'une ou plusieurs mesures de QoS. Par exemple, un SLO de disponibilité peut dépendre de plusieurs composants, chacun d'entre eux pourra avoir une mesure de QoS de disponibilité propre **[5]**.

2.7 Modèles de déploiement

On peut distinguer trois types principaux de modèles de déploiement pour le cloud computing: le cloud privé, le cloud public et le cloud hybride **[1]** (**Figure 3**).

2.7.1 Le cloud privé

L'infrastructure d'un cloud privé n'est utilisée que par un client unique. Elle peut être gérée par ce client ou par un prestataire de service et peut être située dans les locaux de l'entreprise cliente ou bien chez le prestataire, le cas échéant. L'utilisation d'un cloud privé permet de garantir, par exemple, que les ressources matérielles allouées ne seront jamais partagées par deux clients différents.

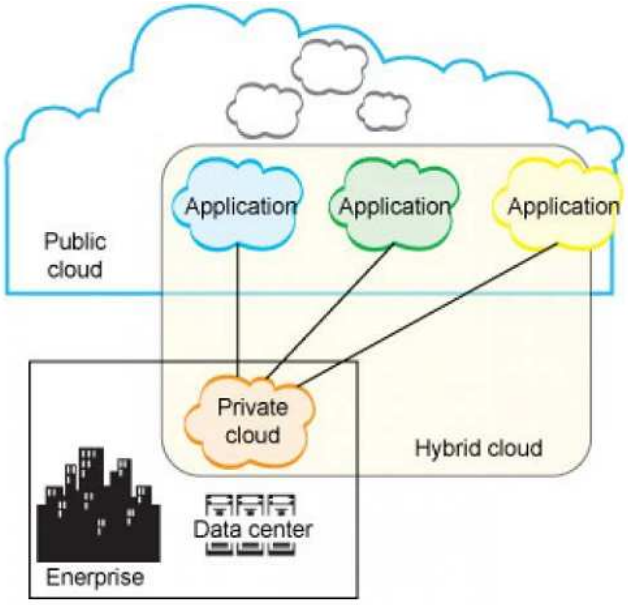


Figure 3. Modèles de déploiement de cloud computing **[8].**

2.7.2 Le cloud public

L'infrastructure d'un cloud public est accessible publiquement ou pour un large groupe industriel. Son propriétaire est une entreprise qui vend de l'informatique en tant que service.

2.7.3 Le cloud hybride

L'infrastructure d'un cloud hybride est une composition de deux ou trois des types de clouds précédemment cités. Les différents clouds qui la composent restent des entités indépendantes à part entière, mais sont reliés par des standards ou par des technologies propriétaires qui permettent la portabilité des applications déployées sur les différents clouds. Une utilisation type de cloud hybride est la répartition de charge entre plusieurs clouds pendant les pics du taux d'utilisation.

En plus de ces trois modèles de déploiement de cloud, on trouve dans les littératures la notion de cloud communautaire.

2.7.4 Le cloud communautaire

L'infrastructure d'un cloud communautaire est partagée par plusieurs organisations indépendantes et est utilisée par une communauté qui est organisée autour des mêmes besoins, vis-à-vis de son utilisation. Par exemple, dans le projet Open Cirrus, le cloud communautaire est partagé par plusieurs universités dans le cadre d'un projet scientifique commun. Son infrastructure peut être gérée par les organisations de la communauté qui l'utilise ou par un tiers et peut être située, soit au sein des dites organisations, soit chez un prestataire de service.

2.8 Concepts cloud liés à l'allocation des ressources

L'allocation des ressources est la première opération réalisée dans le cloud. Elle attribue au consommateur ou client sa portion de ressources exploitables. Par ressources, nous regroupons à la fois la mémoire, le processeur, l'espace de stockage, la bande passante et les équipements informatiques. Mutualisées entre tous les utilisateurs, les ressources représentent le point de rentabilité pour le fournisseur. Ainsi, la conception et l'implémentation des

politiques d'allocation dans le cloud dépendent de la stratégie commerciale du fournisseur. Notons que l'allocation est provoquée entre autres par une réservation du consommateur. Le fournisseur peut dont proposer plusieurs manières de réservation **[9]**:

1. Réservation pour une durée indéterminée : dans ce mode, un contrat est établi avec le consommateur pour une durée infinie et continue (on peut également parler de forfait).
2. Réservation pour une utilisation à venir et pour une durée limitée : dans ce mode, la difficulté se situe dans la gestion des plages de réservation. On retrouve le problème très connu et complexe qui est celui de la planification.
3. Réservation pour une utilisation immédiate et limitée dans le temps : dans ce cas, les ressources requises doivent être disponibles dans l'immédiat.

La prise en compte de ces modes de réservation peut complexifier l'allocation dans le cloud. Notamment, il peut être amené à mettre en place des files les d'attente, avec des notions de priorité. Ainsi, en guise d'exemple, les ressources obtenues par le dernier mode de réservation peuvent être considérées comme moins prioritaires que celles obtenues via les deux premiers. Dans ce cas, le cloud doit être capable d'identifier les ressources à libérer en cas de besoin d'une application plus prioritaire.

Nous avons énoncé dans la section précédente quelques principes fondamentaux du cloud. Les plus importants d'entre eux liées directement au problème d'allocation des ressources sont la mutualisation, la facturation des ressources à l'usage et la virtualisation :

2.8.1 La mutualisation

C'est la pratique qui consiste à partager l'utilisation d'un ensemble de ressources par des consommateurs (ou entités quelconques) n'ayant aucun lien entre elles. Les ressources peuvent être de diverses natures : logicielles ou matérielles (machines, équipements réseau, énergie électrique). Cette pratique dépend du désir des entreprises de délocaliser leurs services informatiques vers des infrastructures de cloud.

Le cloud doit cependant faire face aux problèmes liés à son exploitation et utilisation. Il s'agit des problèmes classiques tels que la sécurité, la disponibilité, l'intégrité, la fiabilité et l'uniformité d'accès aux données **[9]**.

2.8.2 L'allocation et facturation à la demande

Le concept de «pay as you go» permet à l'utilisateur de ne payer que ce qu'il consomme réellement. Le prix est donc calculé à l'aide de ratios tels que processeurs par heure, giga-octets de disques par mois, etc.

Cette notion permet de réduire radicalement les coûts liés aux consommation des ressources informatiques.

Selon **[10]** y'a trois cas d'utilisation des ressource informatique: sous-utilisation, sur-utilisation, et le cas où la capacité correspond à la demande, les graphiques ci-dessous illustrent ces différents cas.

- ✓ Le premier graphique (**Figure 4** ci-dessous) présente le cas d'un utilisateur (entreprise ou particulier) qui a les ressources informatiques pour absorber tous les pics. Les zones grisées montrent la part de budget perdue dû à une sous utilisation des ressources.

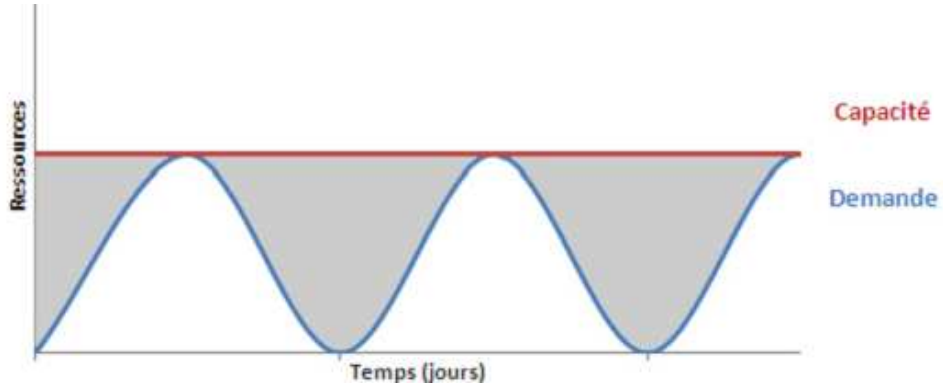


Figure 4. Besoins en ressources informatiques, surestimation

- ✓ Ce deuxième graphique (**Figure 5** ci-dessous), présente le cas d'un utilisateur (entreprise ou particulier) qui ne peut pas absorber tous les pics. Si ce dernier utilise ces ressources pour des prestations, il risque de perdre des clients, car ces derniers ne seront pas satisfaits de la prestation offerte. Les zones grisées montrent la part de temps où les capacités informatiques ont été sous-estimées. Cette zone est généralement difficile à estimer d'un point de vue technique, mais encore plus d'un point de vue budgétaire.

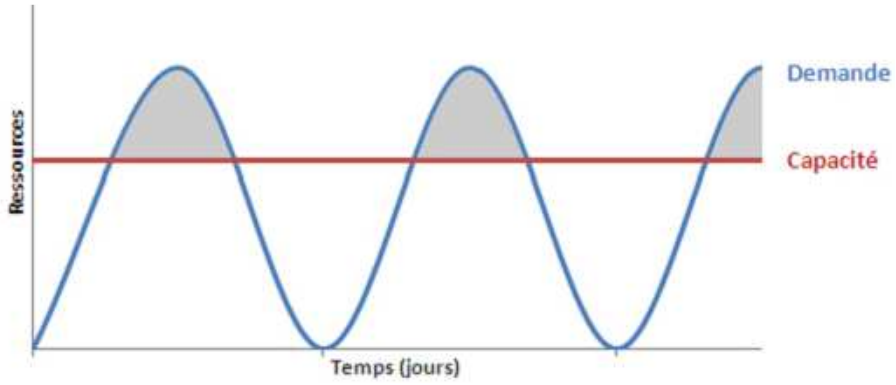


Figure 5. Besoins en ressources informatiques, sous-estimation

- ✓ Finalement, ce dernier graphique (**Figure 6** ci-dessous) montre le cas idéal. Le consommateur ne paie que pour ce qu'elle consomme grâce aux mécanismes de scalabilté et d'élasticité (sous-section 1.8.3) offerts par le « Cloud Computing ».

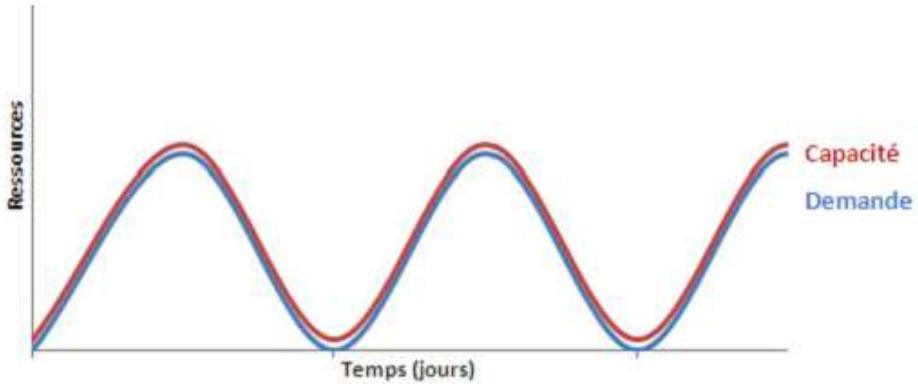


Figure 6. Besoins en ressources informatiques, la capacité correspond à la demande

En conclusion, nous pouvons dire que le « Cloud Computing » avec son modèle économique « pay as you go » permet en général d'éviter le gaspillage des ressources informatiques et donc de diminuer les coûts d'exploitation liés à l'informatique.

2.8.3 Scalabilité et élasticité

Les concepts de scalabilité et d'élasticité sont les pendants techniques du concept décrit ci-dessus «pay as you go». Ils offrent à l'utilisateur l'impression d'avoir en permanence des ressources illimitées. Ces ressources peuvent être facilement et rapidement ajoutées ou retirées, de manière automatique parfois, afin de répondre aux besoins de l'utilisateur **[10]**.

- **La scalabilité** est la capacité d'un système de s'adapter aux dimensions du problème qu'il a à traiter. C'est-à-dire que la puissance de calcul, la mémoire ou le stockage utilisé par le système peuvent facilement être augmentés ou diminués en fonction des besoins **[10]**.
- **L'élasticité** est l'aptitude d'un corps à reprendre, après sollicitations, la forme et les dimensions qu'il avait avant d'être soumis à ces sollicitations. Dans notre cas le « corps » est le système informatique et les « sollicitations » sont les besoins en matière de puissance, de mémoire ou de stockage **[10]**. L'élasticité permet donc d'automatiser le mécanisme de scalabilité des ressources informatiques mises à disposition sur le « Cloud Computing ».

2.8.4 La virtualisation

Le concept de virtualisation **[10]** offre une vue logique plutôt que physique, de la puissance de calcul, de la capacité de stockage, et des autres ressources informatiques. Elle permet de faire tourner sur une même machine physique une ou plusieurs machines logiques. La virtualisation permet aussi de diminuer le gaspillage des ressources tel que nous l'avons décrit dans la sous-section 2.8.2 « pay as you go ».

2.8.4.1 Définition

La virtualisation est un processus qui va permettre de masquer les caractéristiques physiques d'une ressource informatique de manière à simplifier les interactions entre cette ressource et d'autres systèmes, d'autres applications et les utilisateurs. Elle va permettre de percevoir une

ressource physique comme plusieurs ressources logiques et, inversement, de percevoir plusieurs ressources physiques comme une seule ressource logique **[11]**.

La virtualisation repose sur trois éléments importants :

1. L'abstraction des ressources informatiques ;
2. La répartition des ressources par l'intermédiaire de différents outils, de manière à ce que celles-ci puissent être utilisées par plusieurs environnements virtuels ;
3. La création d'environnements virtuels.

2.8.4.2 Domaines de la virtualisation :

2.8.4.2.1 La virtualisation d'applications

La virtualisation d'application est une technologie logicielle qui va permettre d'améliorer la portabilité et la compatibilité des applications en les isolants du système d'exploitation sur lequel elles sont exécutées. Elle consiste à encapsuler l'application et son contexte d'exécution système dans un environnement cloisonné. La virtualisation d'application va nécessiter l'ajout d'une couche logicielle supplémentaire entre un programme donné et le système d'exploitation (**Figure 7**); son but est d'intercepter toutes les opérations d'accès ou de modification de fichiers ou de la base de registre afin de les rediriger de manière totalement transparente vers une localisation virtuelle (généralement un fichier). Puisque cette opération est transparente, l'application n'a pas notion de son état virtuel. Le terme virtualisation d'application est trompeur puisqu'il ne s'agit pas de virtualiser l'application mais plutôt le contexte au sein duquel elle s'exécute (registres du processeur, système de fichiers,...) **[11]**.

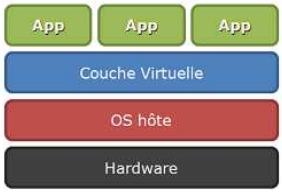


Figure 7. Virtualisation d'applications **[11].**

2.8.4.2.2 La virtualisation de réseaux

De manière générale, la virtualisation des réseaux consiste à partager une même infrastructure physique (débit des liens, ressources CPU des routeurs,...) au profit de plusieurs réseaux virtuels isolés (**Figure 8**). Un VLAN (Virtual Local Area Network) est un réseau local regroupant un ensemble de machines de façon logique et non physique. Puisqu'un VLAN est une entité logique, sa création et sa configuration sont réalisées de manière logicielle et non matérielle **[11]**.

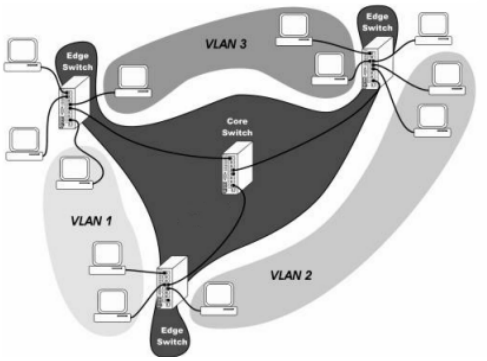


Figure 8. Réseaux virtuels **[11]**.

2.8.4.2.3 La virtualisation de stockage

La virtualisation de stockage est un procédé qui va séparer la représentation logique et la réalité physique de l'espace de stockage. Son but est de faire abstraction des périphériques de stockage utilisés et des interfaces qui leur sont associés (SATA, SCSI,...) afin de limiter l'impact des modifications structurelles de l'architecture de stockage. Il s'agit d'une couche logicielle qui va permettre de regrouper plusieurs espaces de stockage, appelés volumes physiques, pour ensuite découper cet espace global suivant la demande en partitions virtuelles appelées volumes logiques **[11]**.

Figure 9. Virtualisation de stockage **[11]**.

2.8.4.2.4 La virtualisation de serveurs

La virtualisation des serveurs consiste à masquer les ressources du serveur, c.-à-d. le nombre et les caractéristiques de chaque machine physique, de chaque processeur et de chaque système d'exploitation pour les utilisateurs de ce serveur. L'administrateur du serveur va utiliser un logiciel grâce auquel il va diviser un serveur physique (constitué ou non de plusieurs machines distinctes) en plusieurs environnements virtuels isolés les uns des autres. Ces environnements isolés sont parfois appelés serveurs privés virtuels, hôtes, instances, containeurs ou émulations **[11]**.

2.8.4.3 Migration de machines virtuelles

La migration consiste à déplacer une machine virtuelle (VM) d'un hyperviseur[8] à un autre afin de réorganiser la consommation de ressources. Une communication réseau entre les serveurs est nécessaire pour cette opération. Il existe deux types de migration : la migration à froid et la migration à chaud **[12]**.

- **La migration à froid** consiste à éteindre la VM, à transférer son image disque sur un autre hyperviseur en utilisant une communication réseau puis à allumer la VM une fois le transfert terminé. Les calculs en cours d'exécution ainsi que les données non sauvegardées (stockées en mémoire) sont définitivement perdus. De plus, la taille de l'image disque d'une VM peut être conséquente et donc son transfert peut être long. Afin de réduire le temps de la migration, il est très courant, pour ne pas dire systématique, que les hyperviseurs utilisent un disque dur externe accessible via le réseau. Le disque dur devient alors un stockage partagé par les hyperviseurs et donc, toutes les images disque sont elles aussi partagées. Le transfert de l'image disque n'est donc plus utile et la migration est effectuée très rapidement.
- **La migration à chaud** pallie ce problème en permettant de déplacer une VM sans l'arrêter. Le transfert de l'image disque n'étant plus nécessaire, le réseau va être utilisé pour transférer la mémoire de la VM. Afin de réduire le temps d'arrêt de la VM, l'hyperviseur commence par transférer la mémoire non utilisée. Afin de finaliser la migration, les calculs de la VM sont suspendus pendant un court instant (quelques millisecondes) et le reste de la mémoire est copié. L'exécution de la VM reprend alors sans perte de données.

2.8.4.4 Avantages et Inconvénients de la virtualisation

Dans les centres de données, de nombreux serveurs sont sous-utilisés. La virtualisation permet le rassemblement de plusieurs serveurs consommant peu de ressources sur un seul. Il sera alors possible de diminuer le nombre de serveurs allumés et donc de réduire l'électricité consommée. Outre le fait de réduire le nombre de serveurs utilisés et les coûts associés, la virtualisation permet **[12]**:

- d'optimiser la charge de travail des serveurs physiques (Une optimisation de l'infrastructure).
- de faciliter le déploiement d'applications ou de systèmes d'exploitation. Toute VM peut être clonée fournissant alors rapidement une nouvelle VM prêt à l'emploi ;
- d'isoler les applications sur un même serveur. Si une application est corrompue ou pose problème, on peut redémarrer la VM sans affecter les autres VM et leurs applications ;
- déplacer une application d'un serveur à un autre sans interrompre son fonctionnement grâce à la migration à chaud. Par exemple, déplacer une application sur un serveur plus puissant ;

[8]Un hyperviseur est une entité logicielle chargée de l'attribution des ressources d'une ou plusieurs machines physiques aux machines virtuelles, ainsi que de la gestion de ces dernières **[13]**.

- de sauvegarder facilement des VM.

Cependant, quelques inconvénients existent autour de la virtualisation:

- une dégradation des performances des applications s'exécutant dans les VM est observée. Les applications consomment de ressources supplémentaires;
- la hausse du coût d'administration des serveurs. Il faut ajouter l'administration des VM;
- en cas de panne matérielle sur un serveur, toutes les VM de l'hyperviseur sont impactées ;
- la corruption d'une VM peut impacter la sécurité des autres VM hébergées ;
- le partage des ressources et des périphériques peut engendrer des pertes de performances;

2.9 Challenges

Le Cloud computing n'est pas une révolution technologique en soit mais constitue une orientation vers un mode de gestion des ressources informatiques. Cependant, l'idée d'héberger plusieurs applications d'utilisateurs différents pose quelques défis que doit surmonter le cloud. Il s'agit de l'isolation, et l'interopérabilité et la portabilité des applications entre plusieurs plateformes.

2.9.1 Isolation

La mutualisation de ressources dans le cloud (comme dans toutes les infrastructures) implique la mise en place de divers mécanismes (sécurité, comptage de ressources, conflit d'accès, etc.). Le plus important de ces derniers est la gestion des conflits/interférences d'accès entres les utilisateurs. Une réponse idéale pour la mise en place de la mutualisation est l'isolation. Nous regroupons sous ce terme plusieurs types d'isolation à savoir : l'isolation des ressources, l'isolation d'espaces utilisateurs, l'isolation des performances et l'isolation des défaillances **[9]**.

- ✓ **L'isolation des ressources (ou encore partitionnement)** garantit au client l'exclusivité d'accès à un ensemble de fractions ("morceaux") de ressources durant toute sa présence dans le cloud (malgré la mutualisation). Le client a l'illusion d'être le propriétaire et considère l'ensemble comme des machines entières. Cette isolation permet d'éviter les situations de famine aux applications du client (situation dans laquelle une application attend indéfiniment une ressource détenue par une autre). De plus, elle permet au fournisseur du cloud d'identifier et de compter les utilisations de ressources pour chaque utilisateur. Ce décompte servira par la suite à la facturation.
- ✓ **L'isolation d'espaces utilisateurs** donne à chaque client du cloud l'illusion d'être le seul utilisateur. Rien ne doit le laisser présager la présence d'autres utilisateurs ou applications.
- ✓ **L'isolation des performances** permet au cloud d'assurer le non monopole des ressources globales du cloud par un seul client. Prenons l'exemple des ressources réseaux pour illustrer cela. Une utilisation intensive de la bande passante sur le cloud par une application cliente peut affecter l'ensemble du réseau et ainsi avoir un impact sur les autres utilisateurs.

- ✓ **L'isolation des défaillances** permet d'assurer le non violation des espaces utilisateurs dans le cloud. Il comprend également le défi de sécurité. En tant que centre d'hébergement d'applications multiutilisateurs, le cloud doit garantir l'intégrité de chaque espace utilisateur vis-à-vis des autres. Ainsi, aucune action malveillante réalisée par un client ne doit altérer ni le fonctionnement du cloud ni celui des applications appartenant à d'autres clients.

2.9.2 Interopérabilité et Portabilité

Face à la multiplication des plateformes de cloud, les clients pourront être confrontés plus tard à deux choix : (1) la migration d'une application d'un cloud vers un autre et (2) l'utilisation de plusieurs clouds pour l'hébergement de la même application **[9]**.

- ✓ Le choix (1) se pose par exemple lorsque la concurrence entraine un client à partir du cloud qui héberge son application vers un autre plus attrayant. Elle peut également survenir lorsque la plateforme initiale décide de rompre ses services, ce qui oblige le client à trouver une autre plateforme pouvant accueillir ses applications. Dans ces deux situations, il se pose le problème de portabilité de l'application du cloud initial vers le cloud de destination.
- ✓ Le choix (2), il survient lorsque le cloud hébergeant l'application se retrouve à cour de ressources. Dans ce cas, le client ou le fournisseur peut décider d'associer au cloud initial des ressources venant d'une autre plateforme. Ainsi, la même application s'exécute dans deux environnements de cloud différents appartenant à des fournisseurs distincts. L'ensemble formé par les deux plateformes constitue un "cloud hybride". Cette situation soulève le problème d'interopérabilité entre les plateformes de cloud computing.

2.10 Critiques

Avec tout ce que le Cloud offre comme avantages, il présente des inconvénients lies à sa nature **[2]**:

- ✓ L'utilisation des réseaux publics, entraine des risques supplémentaires de cyber attaques lies à la sécurité du cloud.
- ✓ La sécurité du cloud devient cruciale, vu la sensibilité de certains services. Le service en lui-même peut devenir une vulnérabilité et une source d'attaque. En 2009, par exemple, un cheval de Troie a utilise illégalement un service du cloud public d'Amazon pour infecter des ordinateurs[9].
- ✓ Le client d'un service devient très dépendant de la qualité du réseau et a la disponibilité du fournisseur.
- ✓ Juridiquement parlant, le cloud n'a pas de statut clair notamment du a l'absence de localisation précise des données. La question de la confidentialité est toujours posée.

[9] The Register, “PlayStation Network hack launched from Amazon EC2”. (14/05/2011). http://www.theregister.co.uk/2011/05/14/playstation_network_attack_from_amazon/

3 Agents et Systèmes multi-agents

3.1 Les théories orientées agent

Pour présenter de manière structurée les théories orientées agent, nous adoptons dans cette partie, une décomposition basée sur l'approche voyelles **[13]**: Agent, Environnement, Interaction, Organisation.

3.1.1 La vue agent

3.1.1.1 Les agents

Il n'existe pas, actuellement, une seule définition pour le concept d'agent. La plupart des travaux francophones font référence à la définition fournie par Ferber **[14]** qui stipule qu'un agent est une entité physique ou virtuelle:

- ✓ qui est capable d'agir dans un environnement,
- ✓ qui peut communiquer directement avec d'autres agents,
- ✓ qui est mu par un ensemble de tendance (sous la forme d'objectifs individuels ou d'une fonction de satisfaction, voire de survie, qu'elle cherche à optimiser),
- ✓ qui possède des ressources propres,
- ✓ qui est capable de percevoir (mais de manière limitée) son environnement,
- ✓ qui ne dispose que d'une représentation partielle de cet environnement (et éventuellement aucune),
- ✓ qui possède des compétences et offre des services,
- ✓ qui peut éventuellement se "reproduire",
- ✓ qui a un comportement,
- ✓ qui tend à satisfaire ses objectifs, en tenant compte des ressources et des compétences dont elle dispose, et en fonction de sa perception, de ses représentations et des communications qu'elle reçoit.

3.1.1.2 Les modèles d'agents

Nous présentons dans ce qui suit, les principaux modèles d'agents :

3.1.1.2.1 Modèle d'agents réactifs

La structure des agents purement réactifs tend à la simplicité. Ce sont des agents qui fonctionnent selon un mode stimuli/réponse. Dès qu'ils perçoivent une modification de leur environnement, ils répondent par une action programmée. L'agent réactif ne possède pas une représentation complète de son environnement et n'est pas capable de tenir compte de ses actions passées.

De ce fait, il ne peut avoir des capacités d'apprentissage. Ainsi, les agents réactifs sont souvent considérés comme n'étant pas "intelligents" par eux mêmes. Chaque individu pris séparément possède une représentation faible de l'environnement et n'a pas de buts. Mais ces derniers peuvent être capables d'actions évoluées et coordonnées **[15]**.

3.1.1.2.2 Modèle d'agents cognitifs

Les agents cognitifs quant à eux possèdent une mémoire et sont capables de représentations symboliques. Ils possèdent une connaissance propre qui comprend une représentation

(partielle) de leur environnement, des autres agents, ainsi que de leur savoir-faire, ce qui leur permet de gérer leurs interactions avec l'environnement et les autres agents. On parle aussi agents d'intentionnels, délibératifs ou rationnels car ils possèdent des représentations explicites de leurs buts sur lesquelles ils sont capables de raisonner afin de produire des plans d'actions **[13]**.

3.1.1.2.3 Modèle d'agents hybrides

Ce sont des agents ayant à la fois des capacités cognitives et réactives. Ils conjuguent la rapidité de réponse des agents réactifs ainsi que les capacités de raisonnement des agents cognitifs. Ce modèle a été proposé par plusieurs auteurs afin de palier aux problèmes liés au temps de décision et au temps d'action relativement élevé pour les agents réactifs **[13]**.

3.1.2 La vue environnement

L'environnement désigne le monde réel ou bien l'univers virtuel dans lequel les agents évoluent. L'environnement d'un agent est donc constitué de tout ce qui l'entoure : les objets et/ou les autres agents. La description d'un environnement dépend fortement de la nature d'un problème **[16]**. On peu distinguant pour un environnement les propriétés suivantes **[13]**:

- ✓ *accessible versus inaccessible*: un environnement est dit accessible si les agents ont accès à l'intégralité de son état. Inaccessible dans le cas contraire.
- ✓ *déterministe versus indéterministe*: un environnement est déterministe si un changement de son état est déterminé uniquement par son état courant et les actions des agents. Un environnement indéterministe pourra produire un résultat différent pour une même action.
- ✓ *statique versus dynamique*: un environnement est dit statique si il ne change d'état que sous l'effet des actions des agents. Au contraire, un environnement dynamique possède ses propres processus d'évolution qui peuvent modifier son état sans intervention des agents.
- ✓ *discret versus continu*: on parle d'environnement discret lorsque le nombre de perceptions et d'actions possibles est limité. Si ce n'est pas le cas, l'environnement est alors continu.

3.1.3 La vue interaction

Le simple fait de placer un ensemble d'agents dans un même environnement n'est pas suffisant pour définir un système multi-agent (SMA). Les différents agents doivent être en mesure d'interagir et de se comprendre mutuellement afin de pouvoir se coordonner et éventuellement coopérer. L'étude des mécanismes d'interaction est donc primordiale dans la conception d'un SMA.

Ferber **[14]** donne la définition suivante de l'interaction: «Pour un agent, interagir avec un autre constitue à la fois la source de sa puissance et l'origine de ses problèmes. C'est en effet parce qu'ils coopèrent que des agents peuvent accomplir plus que la somme de leurs actions, mais c'est aussi à cause de leur multitude qu'ils doivent coordonner leurs actions et résoudre des conflits».

On distingue différents types d'interaction que les agents peuvent adopter comme : la communication, la coopération, la négociation et la coordination **[15]**.

3.1.3.1 La communication

La communication est un des éléments importants du système multi-agents. Elle permet l'échange des informations, la coopération et la coordination. Dans la communication agent, l'intention de communiquer est de produire un effet sur les destinataires: ils exécuteront probablement une action demandée par l'émetteur ou répondront à une question.

3.1.3.2 La coopération

La coopération se traduit par le fait qu'un ensemble d'agents travaillent ensemble pour satisfaire un but commun ou individuel. L'ajout ou la suppression d'un agent influe considérablement sur la performance du groupe. Le besoin de faire coopérer des agents, vient essentiellement du fait qu'un agent ne peut atteindre son objectif individuellement et a, par conséquence besoin de l'aide des autres agents du système.

3.1.3.3 La coordination

La coordination est présente lorsqu'il existe une interdépendance dans les actions des agents, leurs buts ou même les ressources qu'ils utilisent. A ce stade, coordonner les activités des agents devient un aspect essentiel afin d'éviter des problèmes dans le comportement global du système. En effet, la coordination met de l'ordre dans le processus global effectué par des agents. Elle consiste à synchroniser leurs activités ou à régler les conflits qui existent entre eux.

3.1.3.4 La négociation

Un système multi-agents est composé de plusieurs agents. Ces derniers peuvent entrer en conflit pour plusieurs raisons: conflits d'intérêts ou de buts, accès à des ressources ou proposition de plusieurs solutions différentes à un seul problème. Afin de résoudre ces conflits et de trouver une situation qui satisfasse tout le monde, les agents négocient entre eux en faisant des concessions ou en cherchant des alternatives.

3.1.4 La vue organisation

L'organisation permet de structurer les différentes composantes du système ainsi que leur mode de fonctionnement global. L'organisation s'intéresse aussi aux dimensions sociales des agents. Elle peut être considérée comme une structure décrivant les interactions et autres relations qui existent entre les membres de la dite organisation (dans le but d'assouvir un objectif commun). L'organisation peut donc apparaître comme une structure de coordination et de communication **[15]**.

Quand on parle d'organisation au niveau des systèmes multi-agents, différents aspects sont étudié :

3.1.4.1 Les topologies de l'organisation

Les topologies décrivent les structures organisationnelles en fonction des besoins du système. Nous retrouvons dans la littérature **[16]** différents types d'organisation. L'organisation des agents peut être :

- ✓ les topologies à structure hiérarchique: ces organisations ont un mode d'adaptation fixe, le contrôle est centralisé sur un agent qui communique les ordres aux autres agents (principe maître/esclave). Les autres agents se contentent d'exécuter l'ordre reçu,
- ✓ les topologies de type marché: ces organisations sont composées d'agents coordinateurs et d'agents d'exécutants. L'allocation de tâches est régie sous la forme d'un appel d'offres.
- ✓ les topologies de type communauté: les agents dans ce type d'organisation possèdent les mêmes capacités et le contrôle est fortement distribué (ils communiquent les résultats des problèmes aux autres membres de la communauté),
- ✓ les topologies de type société: le contrôle est décentralisé dans ce type d'organisation, les agents poursuivent chacun un objectif opérationnel et le comportement global s'ajuste par des principes de négociations.

3.1.4.2 L'émergence

Un système multi-agents est un ensemble d'agents situés dans un même environnement et qui interagissent. De ces interactions peuvent émerger différents phénomènes. L'émergence traite donc de l'apparition soudaine non programmée et irréversible de phénomènes dans un système **[15]**.

3.1.4.3 L'auto-organisation

L'émergence et l'auto-organisation sont décrites conjointement car elles décrivent les changements caractéristiques du système. En réalité, l'auto-organisation est une technique d'émergence qui consiste à changer les interactions et les places dans l'organisation des entités, afin de s'adapter globalement à l'environnement (et aux éventuelles perturbations de l'environnement). Il en résulte une évolution dynamique en fonction du contexte permettant de satisfaire au mieux les objectifs du système. Pour aboutir à ce phénomène, il faut agir sur les agents afin d'adapter le comportement global. L'auto-organisation est donc une technique complexe pour la résolution de problèmes **[16]**.

3.2 Systèmes multi-agents

3.2.1 Système Multi-Agents

L'agent est le composant principal des systèmes multi-agents. Selon Ferber **[14]** un système multi-agents est un système composé des éléments suivants :

- ✓ Un environnement E, c'est-à-dire un espace disposant généralement d'une métrique.
- ✓ Un ensemble d'objets O. Ces objets sont situés, c'est-à-dire que, pour tout objet, il est possible, à un moment donné, d'associer une position dans E. Ces objets sont passifs, c'est-à-dire qu'ils peuvent être perçus, créés, détruits et modifiés par les agents.
- ✓ Un ensemble A d'agents, qui sont des objets particuliers, lesquels représentent les entités actives du système.
- ✓ Un ensemble de relations R qui unissent des objets (et donc des agents) entre eux.
- ✓ Un ensemble d'opérations Op permettant aux agents de A de percevoir, produire, consommer, transformer et manipuler des objets de O.

- ✓ Des opérateurs chargés de représenter l'application de ces opérations et la réaction du monde à cette tentative de modification, que l'on appellera les lois de l'univers.

3.2.2 Méthodologies de conception des systèmes multi-agents

De nombreuses méthodologies ont été proposées pour le développement des systèmes multi-agents. Selon **[17]** on peut les classer en trois catégories.

3.2.2.1 Les méthodologies utilisant UML et constituant une extension des méthodes orientées objet

3.2.2.1.1 MaSE

MaSE **[17]** (Multiagent Software Engineering) est une approche complète pour le développement des systèmes multi-agents de l'analyse au déploiement. L'objectif de MaSE est d'aider le concepteur à analyser, concevoir et implémenter un SMA a partir d'un cahier de charges initial.

Elle utilise de nombreux modèles graphiques pour décrire le type d'agents et les interfaces inter-agents. En effet, de nombreux diagrammes sont utilises dans cette méthodologie. Ils sont très proches de ceux utilises dans la conception orientée objet, avec quelques caractéristiques de plus et quelques modifications de la sémantique du paradigme objet pour pouvoir capter les concepts d'agent et les comportements coopératifs des agents.

3.2.2.1.2 AAII

La méthodologie AAII **[17]** (Australian Artificial Intelligence Institute) a été développée en se basant sur l'expérience accumulée dans les travaux effectues pour la gestion du trafic aérien. Elle constitue un mélange de concepts des méthodologies orientées objet avec certains concepts orientes agents. Cette méthodologie vise à construire un ensemble de modèles qui définissent des spécifications du système d'agent a base d'architecture BDI (Belief, Desires, Intentions - Croyance, Désirs, Intentions-). AAII fournit deux modèles:

- ✓ Le modèle externe (le point de vue externe): il a pour but de définir les relations entre les classes d'agents, l'organisation, les rôles, les services et les responsabilités.
- ✓ Le modèle interne (le point de vue interne): il représente les comportements des agents.

3.2.2.2 Les méthodologies organisationnelles

3.2.2.2.1 Aalaadin

Aalaadin **[17]** est une approche organisationnelle pour le développement des SMA fondée sur le concept AGR (Agent, Groupes et Rôles). Il s'agit, d'une part, d'un cadre de développement des systèmes multi-agents, fournissant des indications méthodologiques et d'autre part, d'un environnement de prototypage et d'exécution pour des agents reposant sur les notions du modèle AGR.

Un agent est une simple entité autonome qui joue des rôles dans des groupes. Il peut avoir plusieurs rôles et être membre de plusieurs groupes. L'agent dans Aalaadin n'a aucune

contrainte ou pré-requis ni sur l'architecture interne de l'agent ni sur son modèle de comportement.

Un agent est identifie dans son groupe par un rôle qui est une représentation abstraite d'une fonction ou d'un service qui définit un ensemble de contraintes que l'agent jouant ce rôle doit respecter. Un rôle peut être joue par plusieurs agents. Un groupe est un regroupement d'ensembles d'agents partageant une caractéristique. La communication entre deux agents appartenant aux groupes différents n'est pas possible.

3.2.2.2.2 Gaia

La méthodologie Gaia **[15]** utilise une approche centrée sur l'organisation pour analyser et concevoir un système multi-agents. Elle est considérée comme une méthodologie générique permettant le développement de tous types de systèmes. Les concepts principaux de GAIA sont **[17]**: Rôle, Permission, Responsabilité, Protocole, Propriété d'animation et Propriété de sécurité.

- ✓ Un rôle est défini par quatre attributs : responsabilités, permissions, activités et protocoles.
- ✓ Une responsabilité détermine la fonctionnalité principale liée à un rôle. Elle est décrite par des propriétés d'animation, décrivant le cycle de vie du rôle et des propriétés de sécurité, qui sont des conditions que doivent vérifier certaines variables afin de garantir la survie du rôle.
- ✓ Les permissions sont les droits de l'agent sur les ressources utilisées par un rôle. Ces droits peuvent être de type lecture, modification ou génération. Les permissions énoncent les limites de ressources dans lesquelles l'exécuteur de rôle doit opérer.
- ✓ Une activité représente une opération interne de l'agent, sans qu'il n'ait nécessairement recours à une interaction avec d'autres agents.
- ✓ Les protocoles définissent la manière dont les rôles peuvent agir avec d'autres rôles.

3.2.2.3 Les méthodologies formelles

3.2.2.3.1 DESIRE

DESIRE (Design and Specification of Interacting Reasoning components) est directement issue de l'ingénierie des connaissances. Elle se base sur un modèle qui traite la connaissance, l'interaction, la coordination des tâches et les possibilités de raisonnement dans les SMA.

DESIRE est basée sur trois concepts clés :

- ✓ la représentation de connaissances sous forme de simples graphes nœuds/lien,
- ✓ le modèle générique d'agents: un agent est considère comme un composant décompose en tâches,
- ✓ le modèle de composition de tâches : permet de définir les tâches et leurs relations. Les tâches sont décomposées en sous tâches et sont présentées sous forme de composants avec des informations d'entrée et de sortie.

Le processus de développement de DESIRE se décompose comme suit :

- ✓ La description du problème qui inclut les besoins imposes a la conception ;
- ✓ Les principes de conception ou «design rational » pour spécifier les choix faits lors de la conception ;
- ✓ La conception conceptuelle ou «conceptual design» qui définit un modèle conceptuel pour chaque agent, le monde extérieur et les interactions entre agents ;
- ✓ La conception détaillée qui va spécifier les aspects de connaissance et de comportement ;
- ✓ La conception opérationnelle qui va spécifier les paramètres nécessaires a l'implémentation.

3.2.2.3.2 PASSI

PASSI (Process for Agent Societies Specification and Implementation) **[15]** est une méthodologie se basant sur des concepts provenant de l'ingénierie orientée objet et de l'intelligence artificielle. PASSI est une méthodologie générique pouvant s'appliquer à n'importe quel domaine. PASSI a la particularité de prendre en compte la modélisation des agents mobiles. Cette modélisation respecte les spécifications FIPA.

Dans PASSI, chaque agent est censé satisfaire ses besoins fonctionnels exprimes sous la forme de cas d'utilisation. Ces besoins sont attribues a l'agent dans la phase d'identification d'agent. Il peut jouer plusieurs rôles impliques dans des scenarios et pouvant fournir des services a la société d'agents.

Le rôle que joue l'agent remplit au moins une tâche. Ces rôles sont aussi des mediums d'information par message. Chaque message contient la connaissance échangée et le langage utilise pour décrire son contenu.

Le processus de PASSI est compose de cinq phases :

- ✓ La phase de définition des besoins du système qui permet d'exprimer les besoins et de décrire le contexte du système.
- ✓ La phase de définition de la société d'agents qui permet d'identifier les interactions sociales et les dépendances entre les agents.
- ✓ La phase d'implémentation qui permet de définir la structure et le comportement du SMA et ceux de chaque agent.
- ✓ La phase de codage qui permet d'identifier les modules réutilisables en cas de développement préexistants. Elle permet de rendre exécutable le code des agents.
- ✓ La phase de déploiement qui permet de spécifier la configuration du système.

3.2.3 Domaines d'application des systèmes multi-agents

Les domaines d'application des SMA sont particulièrement riches **[17]**: Support et aide à la décision, commerce électronique, systèmes manufacturiers, télécommunications, supervision de réseaux, robotique, simulation de systèmes sociaux et naturels, applications embarquées et sans fil, applications Internet, filtrage et recherche d'information, gestion des transports et du trafic, ingénierie médicale, jeux, e-Learning. En effet, les SMA ont attiré l'attention à cause de leur grande souplesse d'utilisation puisqu'il n'y a aucune limite fixée sur les comportements des agents ni sur leur manière de s'exprimer mais aussi à cause

de leur pouvoir de représentation d'un univers intuitif où chaque composante d'un système peut être modélisée en un agent. Il existe cependant quelques grandes catégories d'applications des SMA:

- La résolution de problèmes qui concerne toutes les situations dans lesquelles des agents logiciels, purement informatiques, accomplissent des tâches utiles aux êtres humains.

- La robotique distribuée porte sur la réalisation d'un ensemble de robots qui coopèrent pour accomplir une mission et utilise des agents concrets qui se déplacent dans un environnement réel.

- La simulation multi-agents : la simulation est la démarche scientifique qui consiste à réaliser une reproduction artificielle, appelée modèle, d'un phénomène réel que l'on désire étudier, à observer le comportement de cette reproduction lorsqu'on en fait varier certains paramètres, et à en induire ce qui se passerait dans la réalité sous l'influence de variations analogues. La simulation multi-agents offre un outil puissant pour expliquer les changements au sein d'un système naturel ou artificiel en fonction des changements des individus qui le composent.

4 Vers des services cloud intelligent à base d'agents

4.1 Avant-propos

Le cloud computing peut être vu comme une convergence de plusieurs technologies (**Figure 10**), à savoir **[18]**:

- ✓ La virtualisation, décrite dans la partie 1.8.4;
- ✓ L'informatique distribuée, qui consiste à répartir le traitement d'une tâche sur un réseau de plusieurs machines physiques;
- ✓ La gestion automatisée des systèmes, qui définit des principes permettant à une architecture informatique de devenir autonome: celle-ci optimise automatiquement l'utilisation de ses propres ressources, détecte, évite et répare les pannes;
- ✓ Les architectures orientées services, dont l'émergence a permis de faire communiquer plusieurs applications différentes par le biais d'un protocole commun.

L'utilisateur final voit la solution cloud comme étant dynamique, capable de s'adapter à la charge à laquelle il est soumis, comme une solution qu'il peut payer à la demande **[8]**. Le cloud computing offre des services hautement disponibles, élastiques, et ont une grande capacité de passage a l'échelle.

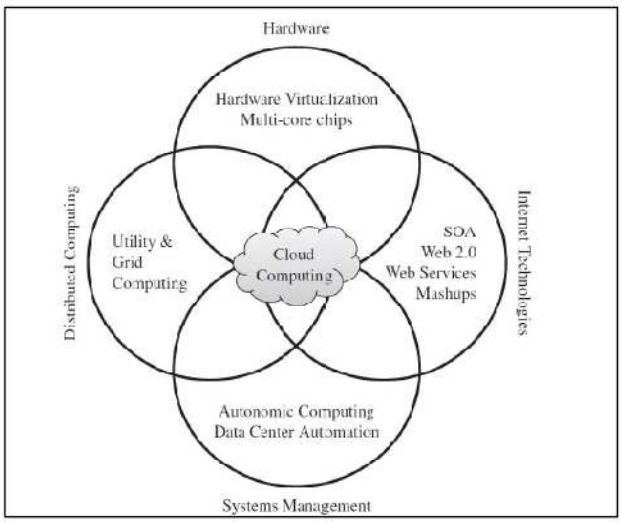


Figure 10. Technologies composant le cloud computing **[18]**

Les agents sont situes dans un environnement où ils interagissent en vue de réaliser conjointement une tâche ou d'atteindre conjointement un but particulier. Ainsi, c'est la notion d'interaction qui distingue un SMA d'une collection d'agents indépendants. Par ailleurs, cette notion est une caractéristique principale d'un SMA en plus de l'organisation qui est produite par l'interaction entre les agents. Dans un SMA, les traitements et les données sont distribues ainsi que les compétences, les rôles et les buts des agents. Ces agents ont chacun un point de vue partiel et il n'y a aucun contrôle global du système. En plus, ils sont en activité permanente et prennent leurs propres décisions en fonction de leurs objectifs et leurs connaissances et sont capables de communiquer entre eux selon des langages plus ou moins élaborés **[17]**. Les principales caractéristiques des agents sont les suivantes: autonomie, pro-activité, réactivité, communication et coopération, négociation et l'apprentissage.

Les SMA représentent un paradigme informatique distribué basé sur plusieurs agents interagissant capables d'un comportement intelligent. SMA peut souvent résoudre des problèmes en utilisant une approche décentralisée, dans laquelle plusieurs agents coopèrent pour générer des solutions efficaces. Sur la base collective, toutes les approches, les développeurs peuvent incorporer l'intelligence au sein des agents logiciels et les déployer ou distribués sur des machines parallèles pour atteindre la haute performance requise afin de résoudre les grands problèmes complexes tout en gardant un temps d'exécution faible.

Le cloud computing et les SMA sont deux modèles d'informatique distribuée. L'intégration de ces deux modèles permettre d'offrir la haute performance, des systèmes complexes et des applications intelligentes qui utilisent les plates-formes de cloud computing et des agents logiciels [19].

4.2 Le cas d'intégration

Les activités de recherche actuelles dans le domaine de cloud computing se concentrent principalement sur la prestation de services, le stockage évolutif de données, les techniques de virtualisation, la consommation énergétique et l'utilisation efficace de l'infrastructure informatique. L'objectif principal de cloud computing est d'utiliser efficacement

l'infrastructure, tout en minimisant les coûts. D'autre part, Les activités de recherche actuelles sur les agents logiciels, se concentrent beaucoup plus sur leurs aspects intelligents et leur utilisation pour le développement d'applications complexes. Ici, les principaux problèmes sont liés à la simulation des systèmes complexes, les systèmes adaptatifs, les applications intensivement orientés logicielles, l'intelligence computationnelle [10] distribuée, et l'apprentissage collectif.

Le cloud computing peut offrir une infrastructure informatique très performante et évolutive dans laquelle les utilisateurs peuvent exécuter les SMA qui mettent en œuvre des applications complexes à base d'agents. Les chercheurs et les développeurs peuvent aussi exploiter cette infrastructure afin de modéliser et simuler les différents systèmes. D'un autre part, les agents logiciels peuvent être utilisé comme composants de base pour la mise en œuvre d'intelligence dans les systèmes de cloud computing, ils peuvent rendent ces derniers plus adaptative, flexible, autonome pour la gestion des ressources, la prestation du service, et l'exécution d'applications à grande échelle **[19]**.

4.2.1 Les clouds comme des plates-formes évolutives

Le cloud computing est le résultat le plus récent de l'évolution des différentes technologies informatiques[11]: du côté matériel (la virtualisation et les architectures multi-cœurs…etc.) et du côté logiciel (informatique en grappe, le grid computing, les services Web, les architectures orientée services, l'informatique autonome, stockage de données à grande échelle, et ainsi de suite). En particulier, la virtualisation dans le cloud computing est l'élément clé qui sépare la fonctionnalité et l'implémentation de système dès ressources physiques. Par l'exploitation des techniques de virtualisation, les fournisseurs peuvent partitionner une infrastructure de cloud en plusieurs machines virtuelles parallèles, configurable dynamiquement selon les besoins des utilisateurs. Les caractéristiques principales de cloud sont: le service libre à la demande, l'accès au réseau ubiquitaire, mutualisation de ressources indépendamment de l'emplacement, élasticité rapide, et paiement selon l'utilisation **[19]**.

Il ya environ cinq ans, quand Amazon a déployé la première infrastructure cloud et la offerte à des clients dans le monde entier, le seul modèle de déploiement était le cloud public. Ensuite il a utilisé son Elastic Compute Cloud (EC2) et Simple Storage Service (S3) pour mettre en œuvre Elastic Beanstalk, après, plusieurs autres entreprises mis en place de grandes solutions de clouds et construisent des environnements de programmation dans laquelle les développeurs peuvent programmer des applications comme des services logiciels en cloud. Par exemple, Microsoft mis en œuvre la technologie .NET sur Azure, Google fournit la AppEngine. Au même temps, la communauté de recherche de cloud computing a développé des applications open source qui doivent être déployées et configurées par les utilisateurs sur des serveurs, des fermes informatiques, ou des centres de données à fin de mettre en œuvre des infrastructures clouds privées, publiques, communautaires, ou hybrides. Des exemples de tels systèmes sont : OpenNebula, Eucaliptus, OpenQRM, Contrail, Puppet, et OpenStack.

[10] L'**intelligence computationnelle** (IC) est un successeur de l'intelligence artificielle. Elle s'appuie sur l'heuristique, comme les systèmes flous, les réseaux de neurones et les algorithmes évolutionnistes.
[11] http://www.nist.gov/itl/cloud/

4.2.2 Les Agents comme des entités distribuées

Les agents offrent généralement des caractéristiques telles que l'autonomie, pro-activité, la communication et la coopération, la négociation et l'apprentissage. Le paradigme d'agent a été conçu comme un modèle d'informatique distribué, dans lequel un ensemble d'agents interagir avec d'autre ensemble. Un agent peut effectuer une tâche donnée, mais l'exécution des tâches complexes nécessitant l'interaction, l'intelligence, l'adaptation et la dynamicité des agents. En fait, il est difficile d'imaginer l'existence et le fonctionnement d'un seul agent comme une entité autonome sans interagir avec d'autres agents (réel ou artificielle) dans un environnement **[19]**.

Les SMA partagent plusieurs caractéristiques avec d'autres paradigmes distribués tels que les acteurs (qui sont essentiellement des pro-géniteurs des agents), les objets concurrents, les réseaux P2P, le grid computing, les réseaux de capteurs, l'informatique autonome, et le cloud computing. Au même temps, les agents possèdent certaines propriétés qui les différencient des autres modèles d'informatique distribué.

'D. Talia' dans son article « Clouds Meet Agents: Toward Intelligent Cloud Services » a proposé deux approches pour l'intégration des agents et les systèmes de cloud computing **[19]**:

- ✓ La première est basée sur le principe que la flexibilité, l'intelligence, la pro-activité et l'autonomie d'agent peuvent aider les plates-formes de cloud computing pour offrent des solutions, des fonctionnalités, et des services intelligents qui ne sont pas disponibles dans les infrastructures de cloud actuel;
- ✓ La deuxième est basée sur l'idée que les infrastructures cloud peuvent offrir une plate-forme idéale sur laquelle fonctionnent les simulations, les applications et les systèmes à base d'agents, en raison de ses vastes ressources de traitement et de stockage.

4.3 Cloud computing basée sur des agents

Actuellement, le grand effort des recherches dans le cloud computing est consacré à la production des infrastructures, des technologies de virtualisation et de gestion des data centers; un peu d'attention est consacré sur des méthodes novatrices qui permettront aux utilisateurs et aux développeurs de découvrir, demander, assembler et utiliser les ressources de cloud computing. Les agents autonomes et flexibles ainsi que les SMA sont des outils appropriés pour négocier l'accès d'utilisateur, la découverte des services, et l'exploitation des ressources de cloud computing.

L'initiative de formuler une nouvelle discipline appelée cloud computing basée agents est présenté dans **[19]**, l'objectif de cette discipline est de fournir des solutions de cloud computing fondées sur la conception et le développement d'agents logiciels qui peuvent améliorer l'utilisation des ressources cloud, la gestion et la découverte des services, la négociation SLA et la composition de services. Les agents autonomes pourraient rendre les clouds plus intelligents dans leurs interactions avec les utilisateurs et plus efficace dans l'allocation de puissance de traitement et le stockage pour différentes applications.

Plusieurs activités de recherche pour la mise en œuvre des solutions cloud computing efficaces a base d'agents dans les différentes classes de livraison de services «*-as-a-service » **[19]**:

- ✓ IaaS : les agents peuvent être utilisés pour aider la provision intelligente des ressources pour les applications d'utilisateurs.
- ✓ Paas : les agents peuvent jouer un rôle efficace dans le déploiement et l'exécution des environnements de programmation (y compris les langues, les bibliothèques et SDK[12]) que les développeurs utilisent pour mettre en œuvre des applications de cloud computing.
- ✓ SaaS : les agents peuvent optimiser l'utilisation des applications en tant que services et faciliter la gestion des infrastructures matérielles et logicielles sous-jacentes, en s'assurant que ces derniers sont utilisés efficacement tout en conservant la qualité de service (QoS) déclarée.

Le cloud computing à besoin des techniques et des méthodes qui s'adaptent aux comportements dynamiques inhérentes aux environnements de cloud computing. Des techniques autonomes pourraient aider les fournisseurs et les utilisateurs d'atteindre cet objectif. Les SMA peuvent gérer le changement des configurations, l'hétérogénéité et la volatilité pour répondre à cette exigence.

L'interopérabilité de cloud est une question clé, elle offre l'utilisation combinée de différentes plates-formes de cloud computing, les agents logiciels peuvent être utiles pour la mise en œuvre de l'interopérabilité de cloud. En outre, La sécurité et la sureté sont aussi des questions critiques de cloud computing parce que les données et les logiciels sont stockés, accessibles, et fonctionnent sur des machines qui ne sont pas détenus ou gérés directement par leurs propriétaires. Les modèles et les algorithmes à base d'agents pour la sureté et la sécurité dans les infrastructures de cloud computing peuvent fournir des solutions décentralisées et efficaces.

L'intégration des solutions à base d'agent logiciel à l'infrastructure de cloud offre ce qui suit **[19]**:

- ✓ *Des services cloud intelligents et flexibles:* Il s'agit notamment des SLA basés sur des agents de négociation, des services de consolidation mises en œuvre par des agents de contrôle, des services d'équilibrage de charge basé sur la coopération d'SMA, et des services d'analyse de performance réalisés par l'intermédiaire des agents d'apprentissage.
- ✓ *Des services autonomes et proactifs:* Les agents peuvent commencer a exécuté sans événement déclencheur, cette possibilité permettre aux services cloud d'incarner des agents qui fonctionnent de manière autonome dans le cloud pour recueillir des données ou des informations.

[12] SDK (*Software Development Kit*) : Un **kit de développement** ou **trousse de développement logiciel**.

- ✓ *Des clouds autonomes:* Une question clé pour les environnements de cloud computing est la conception et le développement de techniques et de méthodes qui s'adaptent aux changements des états et des comportements conformément aux directives de haut niveau des utilisateurs. Des techniques d'autogestion, auto-détection, et d'autonomie qui répondent à ces exigences peuvent être basées sur des SMA afin de repondre à cette question.

4.4 Des agents utilisant le cloud computing

Les applications complexes à base d'agents ou les simulations à grande échelle basées sur les SMA ont souvent besoin à des systèmes informatiques de haute performance et des grands espaces de stockage de données. Par conséquent, les infrastructures cloud offrant une plate-forme idéale pour les simulations et l'exécution des applications basées sur des SMA en raison de ces grandes quantité des ressources de traitement et de mémoire qui les utilisateurs peuvent les utilisées et les configurées dynamiquement.

Une application SMA est exécutée sur une infrastructure de cloud d'une façon où la partie de calcul intensif doit être hébergée dans le cloud, et la partie la plus légère doit être fonctionnée sur un serveur local ou le PC client. De cette façon, les agents peuvent devenir plus efficaces et plus léger. En utilisant des installations puissantes de cloud, Les agents peuvent améliorer leur intelligence et exactitude en exécutant des algorithmes plus sophistiqués. En fait, la puissance de traitement et la quantité de stockage disponible pour un SMA qui s'exécute dans le cloud sont plus grandes que dans d'autres environnements informatiques, et le rendre plus puissant **[19]**.

Les agents activé dans le cloud peuvent couplés des solutions à base d'agents avec les plates-formes informatiques à grande échelle, dynamiques, et distribués pour apporter de nouvelles opportunités au domaine d'agent logiciel et d'élargir les connaissances des agents au-delà de ce que les plates-formes informatiques traditionnels fournissent. Les mécanismes de virtualisation offertes par le cloud computing peut être exploitée pour la composition des machines parallèles, distribuées et à grande échelle sur lesquels exécuter des agents concurrent avec des contraintes de temps réel ou ceux qui nécessitent une haute performance pour obtenir des résultats dans un délai raisonnable. Les systèmes d'agents mis dans le cloud peuvent s'adapter aux machines virtuelles disponibles en utilisant les propriétés de base d'agent tel que l'autonomie, la pro-activité, la négociation et l'apprentissage. L'élasticité de cloud est utile pour exécuter d'une manière évolutive les applications et les simulations SMA qui peuvent s'adapter aux ressources disponibles.

Dans les infrastructures de cloud computing, les agents peuvent trouver une plate-forme appropriée sur laquelle exécutent et accédant à des grandes quantités de données. La communauté des agents devrait exploiter pleinement cette opportunité pour mettre en œuvre des SMA efficaces, d'un point de vue plus général, promouvoir une nouvelle façon de concevoir et mettre en œuvre des agents logiciels à grande échelle **[19]**.

5 Conclusion

Dans ce chapitre les aspects fondamentaux et les concepts qui sont présents, tout en faisant face aux domaines de cloud computing et les systèmes multi-agents sont présentés. En conclusion, le couplage entre les clouds et les agents peuvent être commode pour les deux parties. La convergence d'intérêts entre les systèmes multi-agents qui sont des systèmes fiable, distribués et les infrastructures de cloud computing qui ont besoin à des logiciels intelligents, dynamiques, flexibles et autonomes se résulte par de nouveaux systèmes et applications intelligentes. Tous les deux communautés de recherche doivent être conscients de cette possibilité et devrait mettre en place des activités de recherche conjointes pour atteindre cet objectif.

Le cloud computing basé agents est une discipline novatrice, son objectif est de fournir des solutions de cloud computing fondées sur la conception et le développement d'agents logiciels qui peuvent améliorer l'utilisation des ressources cloud, la gestion et la découverte des services, la négociation SLA et la composition de services. Les agents autonomes pourraient rendre les clouds plus intelligents dans leurs interactions avec les utilisateurs et plus efficace dans l'allocation des ressources.

CHAPITRE II

L'ALLOCATION DES RESSOURCES DANS LE CLOUD COMPUTING

1 Introduction

L’allocation des ressources est un sujet qui a été abordé dans de nombreux domaines informatiques, tels que les systèmes d'exploitation, le grid computing, la gestion des data centers (centre de données) et le cloud computing.

Les objectifs les plus importants de cloud est d’intégré les ressources inutilisées pour créer des pools de ressources partagées et virtualisées, rendre l'accès à la demande des utilisateurs aux ressources commodément, améliorer l'utilisation des ressources, etc. Cependant, les ressources disponibles des fournisseurs et les exigences des consommateurs en ressources sont à la fois variées dynamiquement. Par conséquent, définir un mécanisme d’allocation des ressources aux utilisateurs d’une manière souple, dynamique et fiable est l'un des principaux enjeux dans le Cloud Computing.

2 L'allocation des ressources dans le Cloud

2.1 Présentation du contexte

De nos jours, il y a eu une augmentation dramatique dans la popularité des systèmes du cloud computing qui offrent des ressources informatique à la demande, basent sur la facturation a 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 loyers 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.

La **Figure 11** montre l'interaction entre les fournisseurs et les utilisateurs. Tout d'abord, l’utilisateur envoie une demande contient ces besoins en ressources au fournisseur. Lorsque ce dernier reçoit la demande, il cherche des ressources pour satisfaire la demande et alloue ces ressources à l'utilisateur demandeur, généralement sous forme des machines virtuelles (VM). Ensuite, l'utilisateur utilise les ressources assigné a lui pour exécuter leur applications et paie les ressources qui a utilisé. Lorsque l'utilisateur se termine avec ces ressources, ils les retournés au fournisseur.

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.

Chacun de ces deux partis ne veut pas partager ces informations avec l'autre, ce qui rend l'allocation optimale des ressources plus difficile. Par exemple, les fournisseurs ne veulent pas exposer combien et quel genre de machines qu'ils ont et comment ils sont connectés. De même, les utilisateurs ne veulent pas exposer les détails de leur charge de travail, y compris les codes sources et les ensembles de leurs propres données à d'autres personnes, y compris les fournisseurs. Par conséquent, les utilisateurs ne peuvent pas expriment leurs demandes de ressources de sorte que les ressources allouées sont optimales, car ils ne savent pas exactement ce qui est disponible. De même, les fournisseurs ne peuvent pas allouer des ressources de manière plus appropriée aux demandes des utilisateurs, car il na y'ont pas d'informations sur les charge de travail de ces derniers.

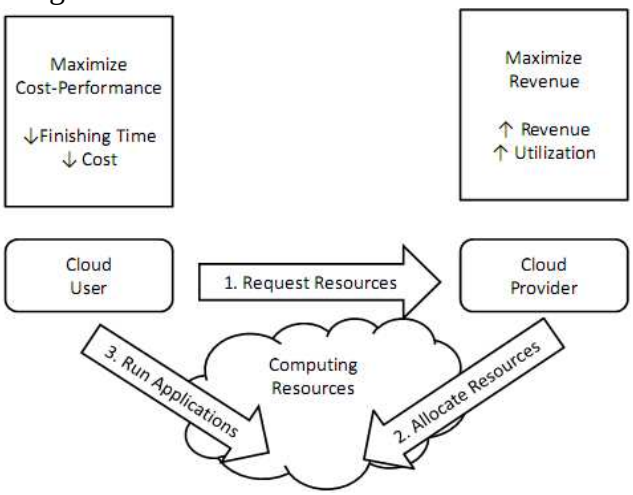


Figure 11. Scénario d'utilisation des ressources cloud **[20]**

Cette section présente les principaux concepts liés a l'allocation des ressources dans le cloud computing. Tout d'abord, les définitions attribuées à l'allocation des ressources dans le cloud computing dans les plus récentes travaux seront exposés:

"Dans le cloud computing, l'allocation des ressources est le processus d'attribution des ressources disponibles pour les requises des applications de cloud. Les ressources cloud peuvent être provisionnées à la demande à grains fins, d'une manière multiplexée. Dans le cloud l'allocation des ressources est basée sur l'infrastructure en tant que service (IaaS)" **[21]**.

*"L'allocation des ressources est un élément important du cloud computing. Son efficacité va influencer directement sur la performance de l'ensemble de l'environnement de cloud. Il requiert le type et la quantité de ressources nécessaires pour chaque application afin de terminer le travail de l'utilisateur"***[22].**

"Dans le cloud computing, l'allocation des ressources (RA) est le processus d'attribution des ressources disponibles pour les requises des applications de cloud via Internet. L'allocation des ressources affame des services si l'allocation n'est été pas gérée avec précision" **[23].**

2.2 Les ressources de cloud

Nous avons identifié plusieurs définitions des ressources de cloud :

" Les Ressources de cloud peuvent être vus 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" **[22].**

"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[13]*, 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"***[24].**

2.3 L'allocation efficace des ressources

L'allocation des ressources disponibles aux consommateurs de cloud est une tâche énorme. Le provisioning des ressources cloud se fait par tenu en compte des accords de niveau de service (SLA). Une allocation efficace des ressources, devrait éviter les situations suivantes **[25]**:

a. *Sur-provisioning* : se pose lorsque le fournisseur alloue pour le consommateur des ressources plus que la demande.
b. *Sous-provisioning* : se produit lorsque l'allocation des ressources est moins que la demande.
c. *Situation conflictuelle de ressource* : se pose lorsque deux applications tentent d'accéder à la même ressource en même temps.
d. *Le manque en ressources :* se pose lorsque les ressources sont limitées.
e. *la fragmentation des ressources* : cette situation se pose lorsque les ressources sont isolées. [Il y a suffisamment de ressources, mais l'allocation est impossible.]

[13] Le **provisioning**, mot anglais désignant l'approvisionnement, est un terme utilisé dans le monde de l'informatique, désignant l'**allocation automatique de ressources**, Au sens large, le provisioning est l'affectation plus ou moins automatisée de ressources à un utilisateur.

2.4 Le système d'allocation des ressources au sein du cloud

Dans les environnements de cloud computing y a deux acteurs ; les fournisseurs et les utilisateurs qui ont différents objectifs, les fournisseurs veulent maximiser les revenus en réalisant une forte utilisation des ressources, tandis que les utilisateurs veulent réduire les dépenses tout en satisfaisant leurs exigences de performance. Cependant, il est difficile d'allouer les ressources de façon optimale mutuellement en raison de l'absence de partage d'informations entre eux **[22]**.

La **Figure 12** présente les entités qui composent l'écosystème[14] du cloud computing :

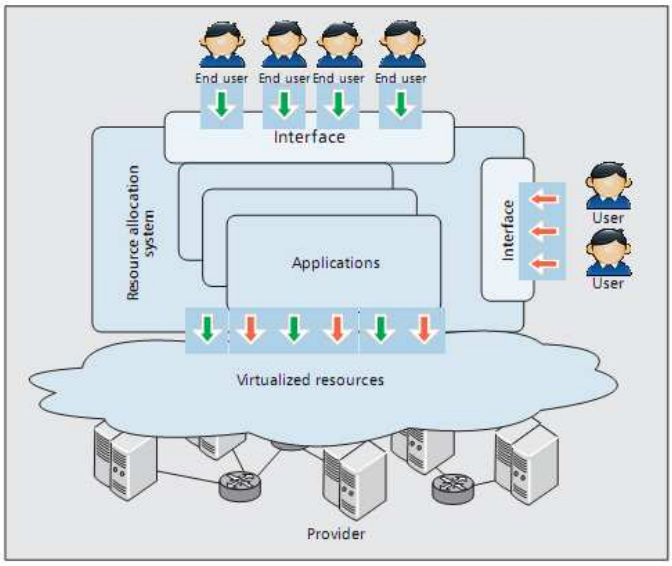


Figure 12. Les entités qui composant l'écosystème du cloud computing **[26].**

Le premier point notable est que le fournisseur face à deux types d'utilisateurs **[26]**: l'utilisateur final, l'utilisateur de cloud.

- *L'utilisateur de cloud :* il est situé au milieu, entre les utilisateurs finaux et le fournisseur de cloud. Un utilisateur de cloud peut être considéré comme un fournisseur de services, qui loue ressources/services offerts par le fournisseur afin d'héberger des applications qui seront consommés par les utilisateurs finaux.
- *L'utilisateur final :* c'est le client d'une application qui utilise simplement les services de cloud. Il est important de souligner que, dans certains scénarios (par exemple, calcul scientifique ou traitement par lots) les utilisateurs de cloud computing peuvent se comporter comme des utilisateurs finaux dans le cloud.
- *Le fournisseur de cloud :* c'est le propriétaire de l'infrastructure. c'est le responsable de la gestion des ressources physiques et virtuelles **[26]**.
- *Les applications de cloud :* pratiquement sont illimitées, un système de cloud computing pourrait exécuter tous les programmes qui peuvent fonctionner sur un ordinateur normal. Potentiellement, n'importe quelle application (logiciel de

[14] **Écosystème du Cloud** est un terme utilisé pour décrire le système complexe de composants interdépendants qui travaillent ensemble pour établir les services de cloud computing.

traitement de texte, programmes informatiques personnalisés conçus pour une société spécifique, etc.) pourrait fonctionner sur le cloud computing **[27].** Les applications de cloud computing peuvent être de différents types qui ont tous des besoins différents **[26].**

- *Les ressources virtualisées*: le Cloud computing offre des ressources virtualisées pour les utilisateurs de cloud computing **[21].** Il est basé sur la technologie de virtualisation qui permettre d'allouer les ressources des centres de données d'une manière dynamique pour les besoins des applications.
- *L'interface d'accès de l'utilisateur*: elle définit le protocole de communication du cloud avec l'utilisateur. L'Interface d'accès doit être équipée par les outils pertinents nécessaires pour une meilleure performance du système. En outre, l'interface d'accès peut être conçue comme des lignes de commande, basée sur des requêtes, basée sur console ou bien une interface de forme graphique. l'interface d'accès est très importante car si l'interface fournie à l'utilisateur n'est pas convivial, l'utilisateur ne peut pas utiliser facilement les services cloud. Dans un autre scénario, on suppose qu'il y a deux fournisseurs de services cloud computing qui fournissent les mêmes services, mais si on a une interface conviviale et la deuxième non, alors l'utilisateur serait préfère certainement celle avec une interface conviviale. Dans de tels scénarios, l'interface d'accès joue un rôle important et c'est pourquoi il est utilisé comme une fonction de comparaison dans le cloud **[25]**.
- *Le système d'allocation des ressources (**RAS**)*: peut être considéré comme tous mécanismes qui visent à garantir que les exigences des applications sont prises en charge correctement par l'infrastructure du fournisseur. ces mécanismes d'allocation des ressources devraient également examiner l'état actuel de chaque ressource dans l'environnement Cloud, afin d'appliquer des algorithmes pour mieux allouer les ressources physiques et/ou virtuels pour les applications des utilisateurs. Il est important de noter que les clients et les utilisateurs peuvent voir les ressources limitées comme illimité grâce au RAS **[28]**.

2.4.1 Les entrées du système

En raison des nécessités de l'environnement de cloud, de l'hétérogénéité des ressources, les restrictions de localité, la nature dynamique de la demande de ressources, etc. il est nécessaire d'avoir un système d'allocation des ressources efficace qui convient à ce type d'environnement. Ce système nécessite des entrées fournis par les fournisseurs et les utilisateurs de cloud à la fois (**Table 3**), Du côté des utilisateurs de cloud computing, les exigences de l'application et l'accord de niveau de service (SLA) sont les entrées principales au RAS. Les offres, l'état des ressources et les ressources disponibles doivent être fournis par l'autre côté (côté des fournisseurs) pour gérer et allouer les ressources de cloud **[24]**.

Paramètre	Fournisseur	Utilisateurs
Les offres de fournisseur	√	-
L'état des ressources	√	-
les ressources disponibles	√	-
les exigences d'application	-	√
Contrat d'accord entre le client et le fournisseur	√	√

Table 3. Les entrées du système d'allocation des ressources **[24].**

2.4.2 Les sorties du système

Les utilisateurs de cloud computing veulent que leurs travail doit être achevé dans le délai avec un coût minime. Mais le fait de prédire la nature dynamique des utilisateurs, leurs demandes, et les besoins de ces applications est impraticable. Pour cela, généralement, de nombreux des fournisseurs de cloud provisioning toujours plus de ressources afin de satisfaire leurs consommateurs, le provisioning en plus rendre des ressources inutiles, qui sera conduire à l'indisponibilité des ressources pour de nouveaux consommateurs. Le rôle de système d'allocation des ressources c'est de trouver la meilleure correspondance (qu'est théoriquement la sortie du RAS) entre les ressources disponibles avec les demandes des consommateurs (Mapping), qui ne devrait pas conduire ni à un sur-provisioning des ressources du point de vue des fournisseurs ni à un sous-provisioning des ressources du point de vue des utilisateurs.

3 Les défis inhérents à l'allocation des ressources

Un RAS doit faire face à un ensemble des défis qui sont divisés en quatre parties: la modélisation et la description des ressources, L'offre et le traitement des ressources, la découverte et gestion des ressources, et la sélection des ressources.

Lors de développement d'un système d'allocation des ressources, il faut réfléchir à la façon de décrire les ressources présentes dans le cloud. **La description** et le développement **d'un modèle de ressources** approprié est le premier défi auquel doit adresser un RAS. Un RAS est également confronté au défi de représenter les exigences des applications, appelé : **offre et traitement des ressources**.

Un RAS automatique et dynamique doit être conscient de l'état actuel des ressources cloud en temps réel, grâce aux mécanismes de **découverte et gestion des ressources** qui font une partie essentielle de système. Ces deux mécanismes sont également les entrées pour les algorithmes **d'optimisation**, car il est nécessaire de connaître les ressources et leurs statuts afin de **sélectionner** ceux qui remplissent toutes les exigences.

La **Figure 13** présente de quelle façon ces défis sont liés. Premièrement, le fournisseur est confronté aux problèmes regroupés dans la **phase de conception**, où

les ressources doivent être modélisées selon une variété des services que le cloud fournira et le type de ressources qu'il offrira.

Les deux autres défis se trouvent dans le cadre de la **phase opérationnelle**. Lorsque les demandes de ressources arrivent, la RAS doit entreprendre la découverte des ressources pour déterminer s'il y a effectivement des ressources disponibles dans le cloud, afin de répondre à une demande. Enfin, si c'est le cas, la RAS peut sélectionnées et les allouées les ressources demandées pour répondre à la demande. Ces défis sont décrits avec plus de détails ci-dessous.

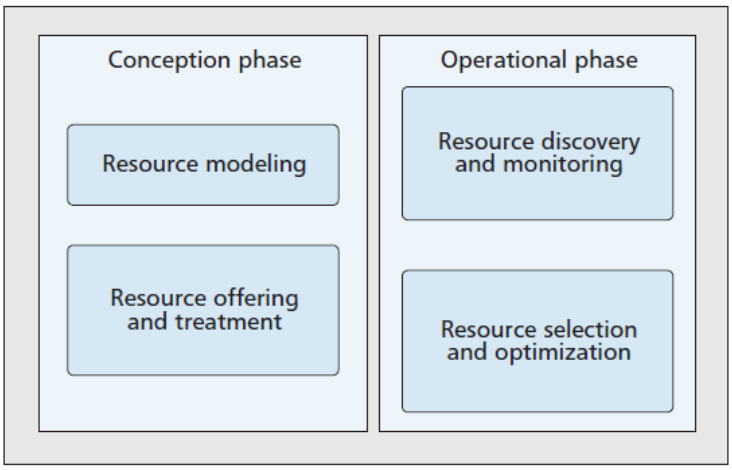


Figure 13. La Relation entre les défis d'allocation des ressources **[26]**.

3.1 La phase de conception

3.1.1 La modélisation des ressources

La modélisation des ressources cloud définit la façon de décrire les ressources d'infrastructure dans le cloud. Cette modélisation est essentielle pour toutes opérations dans le cloud, y compris la gestion, le contrôle et l'allocation des ressources. Les algorithmes de gestion, contrôle et d'optimisation sont fortement dépendants de la modélisation des ressources choisies par le fournisseur. Les services que le cloud fournira aux utilisateurs dépendent aussi de ce concept.

Les ressources réseau et informatiques peuvent être décrits par plusieurs spécifications existantes, tel que : Resource Description Framework (RDF) et Réseau Description Language (NDL). Cependant, dans un environnement cloud computing, il est très important que la modélisation des ressources tient en compte des schémas pour représenter les ressources virtuelles, les réseaux virtuels et les applications virtuelles. Les ressources virtuelles doivent être décrites en termes de propriétés et fonctionnalités.

La difficulté de parvenir à la correspondance de la demande avec les ressources disponibles dans le cloud est liée au degré de détail qui doit être pris en considération lors de la description des ressources, si les ressources sont décrites à l'aide de beaucoup de détails, en risque que la sélection des ressources et la phase d'optimisation pourrait devenir difficile et complexe à gérer. D'autre part, plus de détails offre plus de souplesse et flexibilité dans

l'utilisation des ressources. Ce concept est appelé **la granularité de la description des ressources**.

3.1.2 L'offre et le traitement des ressources

Une fois les ressources modélisées, le fournisseur cloud peut offrir et gérer ces ressources à travers une **interface**, cette dernière peut être considérer comme un moyen pour que les utilisateurs décrivent clairement les exigences de leur application, ces exigences peuvent être représentées par certaines formes de SLA qui doivent être assurée par le fournisseur par une surveillance continue de la qualité de service, en raison de la nature dynamique du cloud. A un niveau inférieur, le RAS doit **gérer** les ressources cloud et à un niveau supérieur, il adresse les exigences des utilisateurs.

Il est important de souligner que la manière dont les ressources sont offertes aux utilisateurs n'est pas nécessairement dépend de la modélisation des ressources. Par exemple, le fournisseur cloud pourrait modéliser chaque ressource individuellement, comme des éléments indépendants dans une échelle à grain fin (tel que GHz de CPU ou GO de mémoire), mais peuvent les offrir couplée, ou comme une collection d'éléments, comme des classes de machines virtuelles.

3.2 La phase opérationnelle

3.2.1 La découverte et la gestion des ressources

La découverte de ressources peut être décrite comme étant la tâche dans laquelle le fournisseur doit trouver les ressources appropriées afin de se conformer aux requêtes entrantes des utilisateurs. En outre, l'une des caractéristiques principales du cloud computing est la capacité d'acquérir et de libérer des ressources sur demande, cette nature dynamique de l'environnement nécessite un contrôle continu des ressources. Ce contrôle peut être passif ou actif.

- **Contrôle passif** : lorsque y est une entité (ou un ensemble d'entités) qui collecte des informations en continu à partir des nœuds, c'est à dire, les nœuds sont passifs par rapport à cette entité. L'entité peut envoyer des messages à des nœuds en demandant des informations ou tout simplement pour récupérer des informations si nécessaire.
- **Contrôle actif** : les nœuds sont autonomes et peuvent décider du moment de l'envoi d'informations d'état à une entité centrale.

Les cloud peuvent également faire usage des deux alternatives afin d'améliorer simultanément la solution de suivi.

3.2.2 La sélection et l'optimisation des ressources

Après l'acquisition d'information sur les ressources disponibles dans le cloud (lors de la phase de découverte), un ensemble des choix appropriés est produit. Le

mécanisme de sélection de ressource choisit la solution candidate qui remplit toutes les exigences et optimise l'utilisation de l'infrastructure. Il est clair que le choix des solutions à partir d'un ensemble de celles disponibles n'est pas une tâche triviale, en raison de la nature dynamique du scénario.

La sélection des ressources appropriées faite en utilisant généralement un algorithme d'optimisation. Beaucoup de stratégies d'optimisation peuvent être utilisées, des techniques simples et bien connus tels que : les heuristiques simples avec des seuils ou bien la programmation linéaire. En outre, les algorithmes d'intelligence artificielle traditionnels - colonie de fourmis et la théorie des jeux par exemple - sont également viables pour le cloud. Ces technique peuvent-être classer en deux : techniques a priori et a posteriori.

- **A priori:** dans ces techniques, la première solution d'allocation est déjà la solution optimale, pour atteindre cet objectif, la stratégie d'optimisation doit déterminer le problème, prendre en considération toutes les variables qui influent sur le processus l'allocation, présenter une solution (ou un ensemble de possibilités) qui satisferont à toutes les contraintes, et rejoindre les objectifs d'une manière optimale.
- **A posteriori:** Une fois que l'allocation initiale des ressources est faite, qui peut être une solution optimale, le fournisseur doit gérer ses ressources de manière continue pour améliorer cette solution. Si nécessaire, des décisions comme celle relative à ajouter ou réallouer des ressources devraient être prises afin d'optimiser l'utilisation du système et de se conformer aux exigences d'utilisateur.

4 Les politiques d'allocation des ressources dans le C.C

La plupart des méthodes d'allocation des ressources actuelles **[30]** (soit fondées sur des algorithmes d'allocation de ressources statique ou dynamique) supposant que l'ensemble des ressources dans les systèmes sont fixes, tous les besoins en ressources sont déterminés à l'avance, les ressources sont isomorphes, et serait traitées de la même manière, etc. Ces pré-requis ne répond pas aux caractéristiques de cloud computing. En plus de ca ces méthodologies peuvent achever l'allocation des ressources avec exactitude dans les petits serveurs, mais en faisant face des milliers de serveurs dans les systèmes de cloud computing, elles devenir très compliquées.

Il ya trois politiques d'allocation de ressources dans les systèmes de cloud computing actuellement **[30]**:

- **Politique 1:** Lorsque les besoins en ressource arrivent, le système de cloud computing analyse l'état d'utilisation des ressources en fonction des SLA dans tous les serveurs, pour trouver les ressources restantes satisfaisant SLA et les affecter aux utilisateurs.

Les **inconvénients** de cette politique sont:

(1) S'il ya beaucoup de types de serveurs différents dans le centre de données, le balayage de chaque serveur serait demander beaucoup de temps et conduirait à la plus mauvais réponse et la performance de l'allocation des ressources serait plus faible.

(2) Lorsque le nombre d'utilisateurs des ressources augmente dans une période de temps, l'allocation des ressources pour chaque exigence en analysant chaque exigence séparément augmenterait considérablement le temps d'attente et conduire à une efficacité inférieure du système.

- **Politique 2:** L'information dynamique sur les ressources physiques est enregistrée dans un centre de gestion des ressources. Les systèmes de cloud computing collectent les changements des ressources dans chaque serveur en temps régulier par un programme spécial et mettre à jour leurs données d'utilisation. Lorsque les besoins en ressources arrivent, les données d'utilisation des ressources sont recherchées et les ressources sont allouées si les ressources répondant au SLA ont été découvert.

  Cette politique peut améliorer grandement le temps de réponse et réduire le cycle d'attente moyen des utilisateurs.

  L'**inconvénient** de cette politique est la suivante:

  Le manque de planification coordonnée pour l'allocation globale des ressources conduirait à un taux de rejet plus élevées des besoins en ressources.

- **Politique 3:** elle est basée sur la politique 2, quand un grand nombre d'utilisateurs des ressources arrivent dans un certain temps, les exigences sont triées par SLA. Les systèmes de cloud computing satisfont d'abord les besoins qui méritent une grande granularité des ressources et en suite satisfont les exigences qui méritent des petites granularités des ressources.

  Les **inconvénients** de cette politique sont les suivantes:

(1) Le changement des exigences de ressources dans les SLA est presque inconsidéré et la QoS est encore très faible. Surtout, le cycle moyen d'attente des ressources de l'utilisateur est significativement augmenté en raison du processus de file d'attente.

(2) Le manque de précision dans l'allocation des ressources conduirait à une fragmentation excessive des ressources.

Globalement, le cycle d'attente moyen des ressources dans la politique 1 est plus long que d'autres politiques.

5 Les avantages et les limites

L'allocation des ressources dans le cloud computing offre de nombreux avantages indépendamment de la taille de l'organisation, des marchés et des affaires. Mais il ya des limites aussi, puisqu'il s'agit d'une évolution de la technologie. Ayons un regard comparatif sur les avantages et les limites de l'allocation des ressources dans le cloud **[24]**.

A. Avantages

1. Le plus grand avantage de l'allocation des ressources est que l'utilisateur n'a besoin pas ni de matériel ni d'installer des logiciels pour accéder, développer et d'héberger des applications sur Internet.
2. Le second avantage majeur est qu'il n'y a pas de limitation de lieu et support. Les utilisateurs peuvent atteindre leurs applications et leurs données partout dans le monde, sur n'importe quel système.
3. L'utilisateur n'a pas besoin de dépenser pour acquérir des systèmes, matériels et logiciels.
4. Les fournisseurs de cloud peuvent partager leurs ressources sur Internet pendant la rareté des ressources.

B. Limitations

1. Les utilisateurs n'ont pas le contrôle sur leurs ressources allouées puisque elles sont situées sur des serveurs distants.
2. Un problème de migration se produit, lorsque les utilisateurs veulent passer d'un fournisseur à un autre pour une meilleure conservation de leurs données. En plus il n'est pas facile de transférer des données énormes d'un fournisseur à l'autre.
3. Dans le cloud public, les données des clients peuvent être sensibles au piratage ou des attaques. En plus les serveurs cloud sont interconnectés, il est facile aux logiciels malveillants de se propager.
4. Les périphériques comme les imprimantes ou les scanners peuvent ne pas fonctionner dans le cloud, beaucoup d'entre eux nécessitant des logiciels à installer localement, les périphériques réseau ont moins de problèmes.

6 La revue de la littérature

6.1 Les Modèles et frameworks proposés

Allouer les ressources d'une manière efficace dans un environnement de cloud computing n'est pas seulement une question importante mais aussi un axe de recherche, elle été étudié intensivement dans les références **[31]-[41]**.

6.1.1 Basé sur la microéconomie

Un système d'allocation des ressources cloud basé sur la microéconomie est proposé dans **[31]** *-Cloud Resource Allocation Scheme Based on Microeconomics and Wind Driven*

Optimization – où les auteurs combinent le mécanisme de l'enchère double continu avec le mécanisme de l'enchère multi-attributs pour proposer un mécanisme d'allocation des ressources basé sur l'enchère multi-attributs double contenu. Plusieurs attributs autres que le prix sont décrits pour les fournisseurs de ressources et les consommateurs afin de mieux répondre à leurs besoins. Ils sont utilisées la rétro-propagation du gradient (Back Propagation-BP-) de réseau de neurones pour le traitement de ces attributs. L'algorithme SVM (Support Vector Machine) est appliqué pour prédire le prix à l'avance grâce a l'utilisation de l'historique d'information d'enchères. L'algorithme WDO (Wind Driven Optimization) est utilisé pour obtenir le schéma d'allocation finale, dont l'objectif de l'optimisation est la satisfaction des utilisateurs.

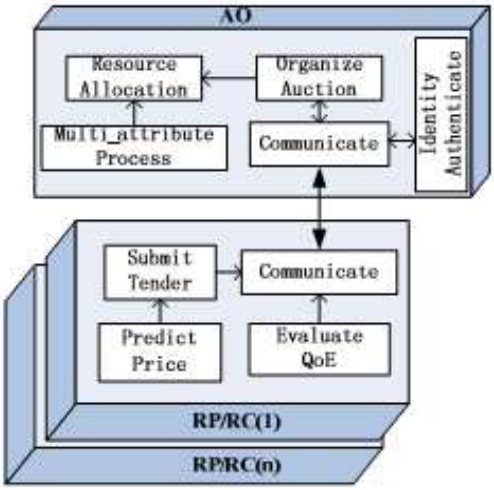


Figure 14. Un framework d'allocation des ressources basé sur la Microéconomie **[31]**

Le framework (**Figure 14**) du mécanisme d'allocation des ressources proposée sur la base d'enchère comprend trois rôles : RC (les consommateurs des ressources), RP (les fournisseurs de ressources) et AO (l'organisateur d'enchère). RC et RP agissent respectivement comme les acheteurs et les vendeurs dans les transactions, ils ont des modèles de fonctions similaires. L'AO est principalement le responsable sur les interactions entre RC et RP, y compris l'authentification des identités de ces derniers, la collecte des offres, l'organisation de l'enchère, la coordination dans le processus d'enchère et l'utilisation de l'algorithme d'optimisation pour trouver l'allocation des ressources optimale. RC et RP doivent intégrer le prix prévu et d'autres informations afin de déterminent les offres et évaluent la QoS.

6.1.2 Basé sur le mécanisme de marché

❖ Une stratégie d'allocation des ressources automatique basée sur le mécanisme de marché dans le cloud computing est proposée dans **[32] -Automatic Resource Allocation Strategy Based on Market Mechanism in Cloud Computing (ARAS-M) –** l'objectif de la stratégie est d'atteindre l'état d'équilibre grâce à l'utilisation d'un algorithme automatique d'ajustement de prix fondé sur l'AG (Genetic Algorithm); les auteurs introduisent la théorie de l'**équilibre**

où l'état d'équilibre est défini et son optimalité est prouvé. Un Monitor de Machine Virtual (VMM) avec une haute performance est construit sur la machine physique directement, et avoir le contrôle de la ressource physique sous-jacent. L'architecture de la stratégie d'allocation des ressources basée sur le marché ARAS-M est représentée dans la **Figure 15.**

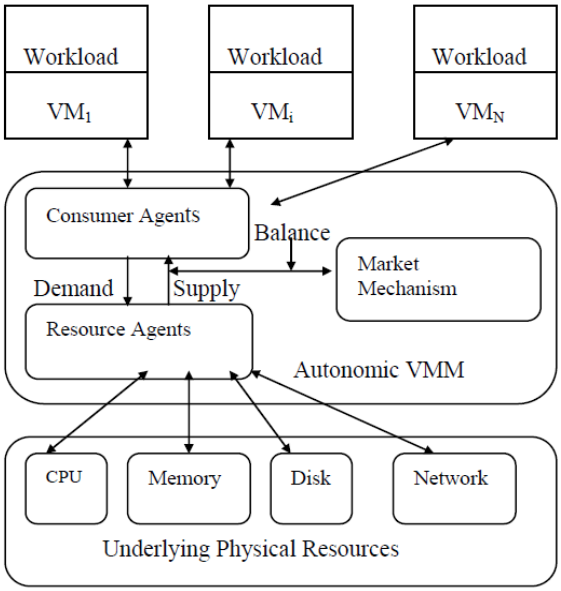


Figure 15. Une stratégie allocation des ressources basée sur le mécanisme de marché **[32]**

Le travail soumis par un consommateur cloud est exécuter sur une machine virtuelle, Le nombre de fractions qui peuvent être utilisé par une machine virtuelle est déterminé par ARAS-M, et peut être ajusté dynamiquement. L'architecture de l'ARAS-M se compose principalement de trois parties:

- Agent Consommateurs (CA) : il participe au système de marché et vise à obtenir un bénéfice maximal pour le consommateur.
- Agent de ressource (RA) : c'est le délégué d'un type de ressource, il publier le prix des ressources et ajuster ce dernier en fonction de l'offre par rapport au demande dans le système du marché.
- Mécanisme d'économie de marché : c'est le responsable de l'équilibre des offres et les demandes des ressources.

6.1.3 Basé agent

❖ Un modèle d'allocation des ressource adaptatif basé agent dans le Cloud Computing est proposé dans **[33]-Agent-based Adaptive Resource Allocation on the Cloud Computing Environment**- Pour trouver un centre de données approprié a la demande de consommateur, le modèle proposé considère à la fois : la distance géographique entre le lieu de consommateur et les centres de données, et la charge de travail des centres de données.

Un testbed[15] à base d'agents a été conçu et mis en œuvre pour démontrer le modèle d'allocation des ressources adaptatif proposé. Dans le testbed, les auteurs sont utilisés deux types d'agents : les consommateurs et les fournisseurs de service. Ils se comportent au nom de chaque consommateur et fournisseur. La vue d'ensemble du testbed est représentée dans la **Figure 16**.

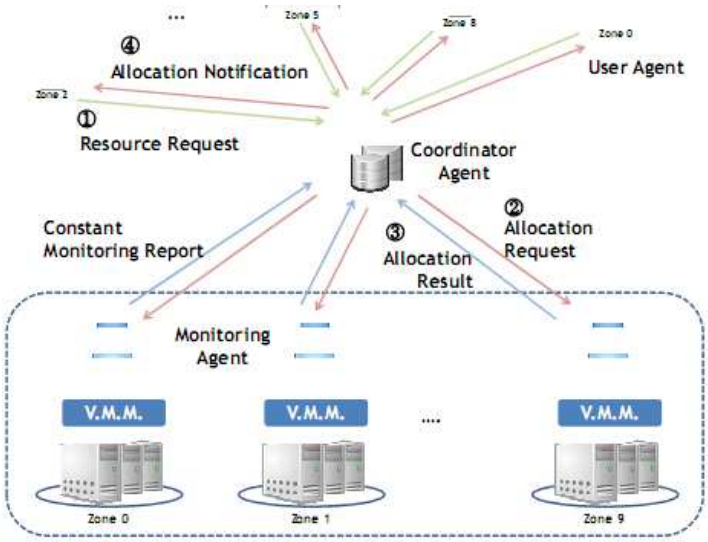


Figure 16. Vue d'ensemble sur le testbed d'allocation des ressources basé agent **[33]**

Le testbed se compose de plusieurs centres de données qui sont répartis géographiquement, De même, il existe des consommateurs de différents lieux. Il est constitué de trois types d'agents:

1) Agent Utilisateur: Il envoie un message de demande d'allocation pour le compte d'un consommateur à l'agent coordinateur. Puis, il attend de recevoir un message de résultat d'allocation de l'agent coordonnateur.

2) Agent Coordinateur: Un agent coordinateur est chargé de coordonner l'allocation des ressources pour les participants (les consommateurs et les fournisseurs). Son rôle principal est de trouver un centre de données approprié pour une demande de consommateur.

3) Agent de contrôle: Un agent de contrôle est chargé de contrôler chaque centre de données distribuées. Cela signifie que l'agent est au même endroit avec le centre de données.

❖ Une allocation automatisé des ressources cloud basé-agent avec l'utilisation de micro-accords (agreements) est proposé dans **[34]-Agent based automated Cloud resource allocation using micro-agreements**- les auteurs ont proposés un service intelligent d'allocation des ressources cloud (ICRAS) qui nécessite une architecture sous-jacente composé de trois éléments principaux: 1) un consommateur, 2) un fournisseur de service cloud (CSP) et 3) un agent ICRAS. Ces éléments représentent les trois rôles sur le marché, qui

[15] Un **testbed** (généralement écrit : test bed dans les publications de recherche) est une plate-forme d'expérimentation de grands projets de développement permet d'effectuer des tests rigoureux, transparent et reproductible les théories scientifiques, les outils informatiques et les nouvelles technologies.

peut contenir plusieurs instances de chaque un de ces derniers. En outre, cette architecture fournit les mécanismes et les protocoles qui permettent à ces éléments de communiquer entre eux et négocier les micro-SLA d'une manière autonome. Cette architecture est illustrée sur la **Figure 17**.

1) Le consommateur : Chaque consommateur interagit directement avec un agent ICRAS. Il précise ses exigences dans une offre SLA. Ce document permet à un consommateur d'indiquer ces besoins en hardware, software, les priorités, une plage d'options, et les dépendances entre ces exigences. Si les besoins en ressources sont changés, le consommateur doit informer un l'agent d'ICRAS par ces changements.

2) CSP (fournisseur de service cloud) : Pour permettre au CSP de participer dans l'architecture ICRAS, il doit offrir une interface compatible qui est accessible par l'agent ICRAS. Cette interface doit prendre en charge deux fonctions principales: la négociation et la migration.

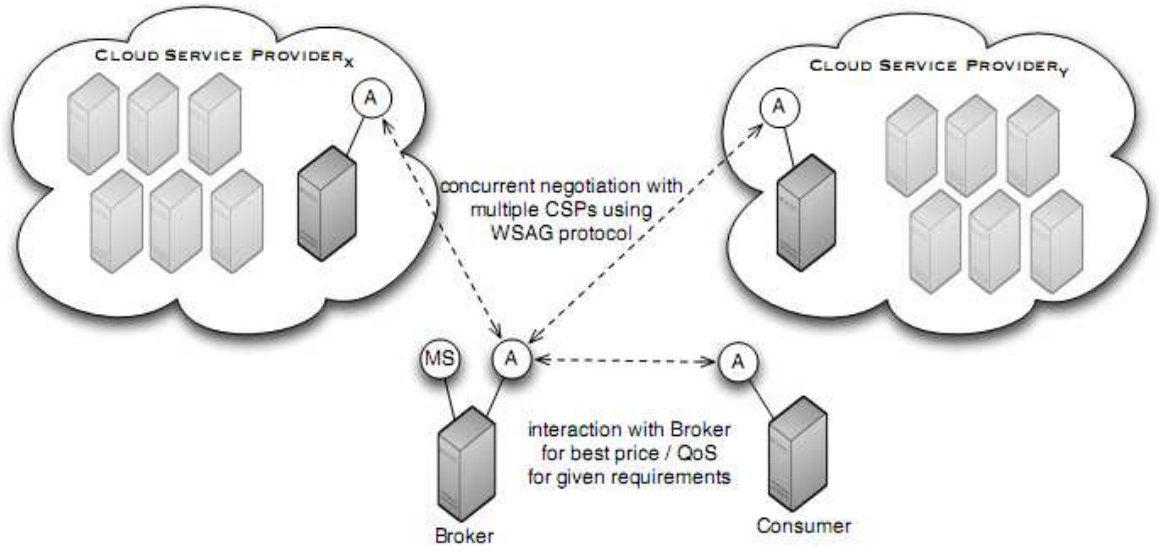


Figure 17: Une allocation des ressources cloud basé-agent avec l'utilisation des micro-accords **[34]**

3) L'agent ICRAS : il a cinq responsabilités: découvrir les offres de CSP, évaluer ces offres, négocier un contrat de service avec le CSP au nom du client, gérer l'approvisionnement des nouvelles ressources cloud pour détecter les violations de SLA et gère la migration d'un CSP vers un autre CSP.

6.1.4 Basé SLA "service level agreement"

Un framework de cloud computing basée sur SLA qui facilite l'allocation des ressources dans les centres de données est proposé dans **[35] -SLA-based cloud computing that facilitates resource allocation in the distributed data centers of a cloud provider.** Où le consommateur cloud peut établir un SLA concernant le prix du service et le temps de réponse par le biais d'un système de négociation d'SLA automatisé. La **Figure 18** montre le framework de cloud Computing proposé, où les consommateurs de service cloud exécutant

leurs applications sur des machines virtuelles offertes par un fournisseur de cloud. Cet environnement de cloud computing se compose de: les fournisseurs et les consommateurs de service cloud et un courtier de service qui aide les consommateurs de service pour trouver le fournisseur qui détient le service demandé.

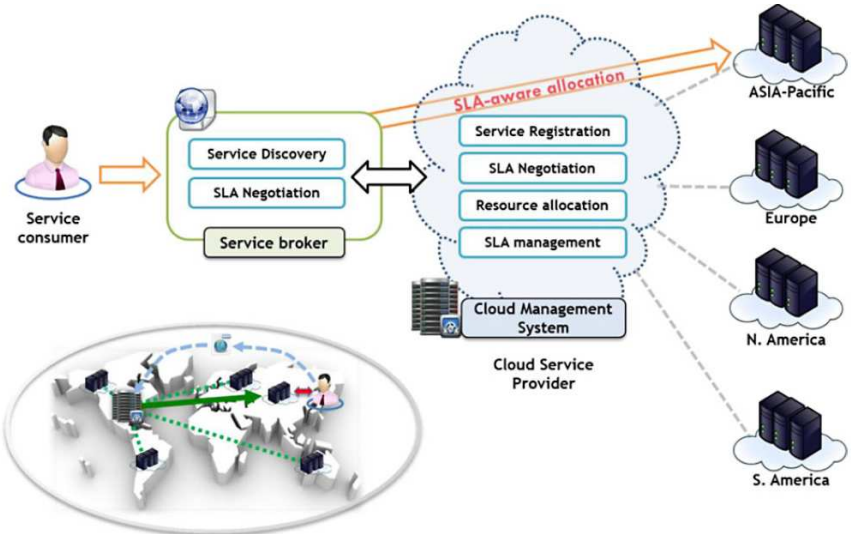


Figure 18. Un framework facilite l'allocation des ressources dans cloud computing basée SLA **[35]**

Le fournisseur et le courtier établirent les termes d'SLA. Si la négociation automatique est effectuée avec succès (ils parviennent à un accord accepté mutuellement), le SLA est établi et le fournisseur reçoit le paiement du consommateur a travers le courtier. Le composant de gestion des SLA sélectionne parmi les centres de données distribués un centre de données à allouer au service demandé. Chaque centre de données comprend un gestionnaire qui gère les machines physique.

6.1.5 Basé sur des métriques de qualité

Une architecture de qualité pour l'allocation des ressources dans le Cloud Computing est proposée dans **[36] -Quality Architecture for Resource Allocation in Cloud Computing-** Pour fournir une allocation des ressources efficace dans le cloud, les auteurs ont proposés une architecture organisée en unités. Cette architecture est orientée disponibilité de ressources en fournissant de multiples scénarios d'allocation des ressources afin de satisfait le SLA convenu entre les utilisateurs et les clients.

L'architecture proposée présenté sur la **Figure 19**, est composé de cinq unités principales: Unité de service (SU), Unité de gestion (MU), Unité d'allocation (UA), Unité de décision (DU), Unité des utilisateurs (UU).

1) L'unité de service (SU) : présente les services de cloud computing proposés par les fournisseurs. Ces services sont organisés en trois couches : SaaS, PaaS et IaaS. Cette unité gère les ressources virtuelles disponibles pour servir les services de cloud computing. Elle communique avec l'unité de contrôle et l'unité des utilisateurs.

2) L'unité de gestion (MU) : elle collecte les informations sur les services et les ressources de cloud et gère ces derniers. Elle est liée à l'unité d'allocation afin de gérer l'allocation des ressources.

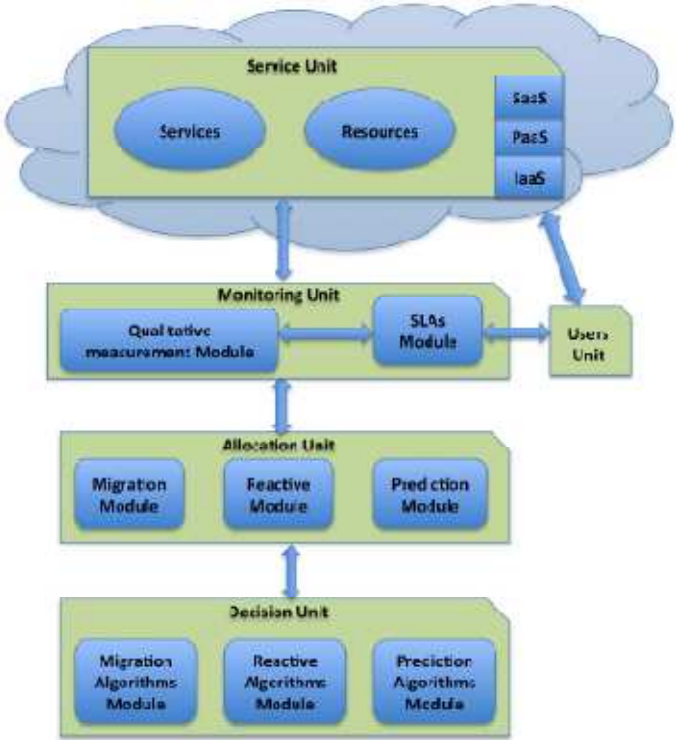


Figure 19. Une architecture de qualité pour l'allocation des ressources dans le Cloud Computing **[36]**

3) L'unité d'allocation (UA) : c'est le cœur de l'architecture, puisque tous les services cloud computing tournent autour de la disponibilité des ressources. Cette unité gère l'allocation des ressources en fonction des informations transmises par MU. L'allocation de ressources est basée sur trois stratégies organisées en trois modules: Module de migrations (MM), Module réactif (RM) et Module de prédiction (PM). Chaque module peut interagir avec les autres. La migration peut être faite suivant l'information fournie par le module réactif ou le module de prédiction.

- Le module de migration (MM) fournit les algorithmes bien connus pour offrir comme service la migration des ressources d'un serveur physique à un autre.
- Le module réactif (RM) fournit une interface pour assurer l'allocation des ressources dynamique pour les services, il exploite les informations envoyées par l'unité de migration.
- Le module de prédiction (PM) : ce module utilise l'information de charge actuelle fourni par MU pour prédira la future charge.

4) L'unité de décision (DU): c'est l'unité d'exécution de la stratégie choisie par UA. Elle possède trois algorithmes pour chaque catégorie d'allocation: Module des algorithmes de migration (MAM), Module des algorithmes réactives, (RAM) Module des algorithmes de prédiction (PAM).

5) L'unité des utilisateurs (UU): gère les utilisateurs et les clients de services de cloud computing. Chaque utilisateur est en relation avec son SLA et ces services Cloud. Cette unité est alors liée à l'unité de service et l'unité de gestion.

6.1.6 Autres solutions

- Une Approche de théorie des jeux pour une allocation équitable et efficace des ressources dans le Cloud Computing est proposée dans **[37] -Game Theory Approach to Fair and Efficient Resource Allocation in Cloud Computing-** ou le problème d'allocation des ressources est modélisée comme un jeu extensif fini. Les serveurs physiques avec des ressources inutilisées sont modélisés comme les joueurs égoïstes et chaque joueur à un nombre limité de matrices d'allocation possible. Chaque serveur physique fournir des ressources est considéré comme un joueur et sait les informations d'utilité des autres joueurs. Pour parvenir à une allocation équitable entre les utilisateurs avec un niveau élevé d'utilisation des ressources, les auteurs ont concevez une fonction d'utilité de compromis équité-utilisation.

- Une recherche pour l'optimisation d'allocation des ressources de Cloud Computing basée sur la structure de soutien est abordée dans **[38] -Research on Optimization of Resources Allocation in Cloud Computing Based on Structure Supportiveness-** Les auteurs construisent une relation de soutien de structure entre les composants des applications et les ressources de cloud basés sur la compatibilité en fonction des paramètres de performance. Le poids des ressources limitées et les composants populaires sont calculés de manière itérative, et les prix des deux (les fournisseurs et les consommateurs de cloud) sont ajustés en fonction de leurs poids pour atteindre la meilleure adéquation entre les fournisseurs de cloud.

- Une stratégie d'allocation des ressources basée sur l'algorithme d'essaim de particules dans l'environnement de Cloud Computing est proposée dans **[39]-A Resource Allocation Strategy Based on Particle Swarm Algorithm in Cloud Computing Environment-** Une fonction d'utilité est conçu pour évaluer la qualité de service selon les caractéristiques de charge de travail, et selon la demande de ressources de toutes les charges, les prix des ressources sont ajustées dynamiquement par des agents de ressources correspondants afin d'obtenir le maximum de profits pour chaque charge de travail.

6.2 Les algorithmes heuristiques utilisées pour l'allocation des ressources

Le problème d'allocation des ressources est un problème d'optimisation combinatoire, où une quantité limitée de ressources doivent être alloué à un certain nombre d'événements de telle sorte que l'allocation la plus efficace des ressources est atteinte grâce à l'optimisation des

différents objectifs. Les objectifs de problème d'allocation de ressources sont généralement de réduire le coût global, le temps de traitement, gérer les conflits, maximiser le bénéfice, l'efficacité ou la compatibilité. Le problème d'allocation des ressources implique généralement un nombre énorme de variables entières et multiples objectifs dans un espace de recherche discret, ce qui provoque leur complexité à être NP-complet dans le pire des cas. Cette nature rendre des méthodes exactes, comme la programmation en nombres entiers ou bien linéaire, inadéquates pour traiter le problème d'allocation de ressources, et favorise des techniques heuristiques comme : les algorithmes génétiques (GA), recuit simulé (SA), optimisation par essaim de particules (PSO), et l'optimisation par colonie de fourmis (ACO) pour obtenir des solutions approximatives.

Les principaux algorithmes et formulations utilisées pour le problème d'allocation des ressources dans le cloud sont brièvement décrits dans cette partie.

6.2.1 Algorithme de recuit simulé

La technique de recuit simulé (SA) a été initialement proposé pour résoudre les problèmes difficiles d'optimisation combinatoire **[40]**. C'est une technique pour trouver une meilleure solution pour un problème d'optimisation en essayant variations aléatoires de la solution actuelle. La caractéristique principale est que la variation la plus mauvaise peut être acceptée comme une nouvelle solution avec une probabilité, c'est le majeur avantage de SA par rapport aux autres méthodes de recherche.

Dans l'algorithme de recuit simulé de base, la solution initiale est toujours sélectionnée à partir de la plage de variation de chacun des paramètres au hasard. Mais dans **[41]**, les auteurs obtiennent la solution initiale comme suit: d'abord, ils maintiennent une liste de toutes les machines virtuelles dans l'ordre décroissant en fonction de leurs charges. Ensuite, ils ont mapper la première machine virtuelle dans la liste qui a le plus charge de travail sur une machine physique qui a la plus grande capacité résiduelle, et répéter ce processus pour la prochaine machine virtuelle jusqu'à ce que toutes les machines virtuelles ont été alloués aux serveurs physiques. Enfin, ils obtiennent une solution initiale comme entrée pour l'algorithme d'équilibrage de charge de recuit simulé. Grâce à cette dotation initiale, ils peuvent obtenir une assez bonne solution initiale possible.

6.2.2 Algorithme génétique

Les algorithmes génétiques sont des algorithmes d'optimisation s'appuyant sur des techniques dérivées de la génétique et des mécanismes d'évolution de la nature : croisements, mutations, sélections…etc. Les auteurs dans **[32]** ont proposés un algorithme génétique (GA) pour atteindre un état d'équilibre entre les demandes et les offres grâce à l'ajustement automatique de prix. Les procédures de GA, méthode de codage de l'AG, la fonction de fitness de l'AG et les trois opérateurs de GA (Opérateur de sélection, Opérateur de croisement, Opérateur de mutation) sont décrites en détail dans **[32]**.

6.2.3 Optimisation par essaim de particules

L'algorithme d'optimisation par essaim de particules (PSO) est une méthode de calcul d'optimisation bionique. L'algorithme est simple et Il nécessite moins d'ajustement des paramètres. Les auteurs dans **[39]** utilisent l'algorithme pour ajuster le prix des différentes ressources respectivement. L'algorithme comporte plusieurs étapes : dans la première étapes le vecteur initial de prix pour chaque ressource sera définit, en deuxième étape quelques paramètres tel que la vitesse initiale de vol, l'échelle de l'essaim de particules, les facteurs d'apprentissage, la vitesse de vol maximale et la condition itérative de terminaison doivent être réglés, ensuite l'algorithme de PSO est utilisé pour obtenir la solution optimale, sur la base de cette dernière l'algorithme proposé calcule la demande totale des charges de travail pour chaque ressource. Plus de détails dans **[39].**

6.2.4 Optimisation par colonie de fourmis

Les ressources de cloud computing sont distribuées largement avec une grande diversité. Les changements dynamiques en temps réel des exigences des utilisateurs sont très difficiles à prévoir avec précision. L'heuristique de l'algorithme de colonie de fourmis peut être utilisée pour résoudre ce genre de problèmes. Les auteurs dans **[41]** ont proposés une méthode pour l'initialisation des phéromones de chaque nœud basé sur les ressources matérielles (puissance de traitement, la capacité de mémoire, la bande passante ...) de la machine virtuel. La mise à jour des phéromones sont effectuées au même temps lorsqu'une nouvelle tâche est attribuée à un nœud. Basant sur la formule de calcul du prochain saut dans l'algorithme classique de colonie de fourmis, les auteurs ont proposés une formule pour calculer la probabilité que les fourmis choisissant le point suivant, parmi les nouds possibles. Plus de détails dans **[41].**

7 Conclusion

Le cloud computing est un paradigme émergent dans le monde de l'informatique, dans lequel l'allocation des ressources est l'un des plus des éléments importants. Une stratégie d'allocation des ressources efficace est nécessaire pour atteindre la satisfaction de l'utilisateur et de maximiser les bénéfices des fournisseurs de services de cloud computing.
Dans ce chapitre les aspects fondamentaux ainsi que les principaux défis qui sont présents, tout en faisant face au sujet d'allocation des ressources sont présentées. La revue de littérature exposée nous permet de survoler sur les frameworks, les modèles et les algorithmes utilisés dans cet axe.

CHAPITRE III

PROPOSITION ET CONCEPTION DE l'APPROCHE

1 Introduction

Les systèmes de cloud computing sont caractérisés par l'utilisation d'un grand pool de ressources informatiques accessibles à travers Internet à la demande **[42]**. Il offre une variété de ressources, tels que le réseau, le stockage, et d'autres ressources informatiques aux utilisateurs adoptées par IaaS, PaaS, SaaS et d'autres formes de service. Ces ressources sont vastes, hétérogène et distribuée **[43]**; De nombreuses stratégies sont utilisées pour fournir des ressources aux utilisateurs et elles sont regroupées sous le terme " stratégies d'allocation des ressources". Cependant, L'allocation des ressources dans un environnement de cloud computing est très complexe.

D'autre part, l'agent software est une entité logicielle qui fonctionne en continu et indépendant de l'environnement donné, habituellement associé à d'autres agents pour la résolution de problème. Les systèmes multi-agents ont été de plus en plus attirés par les chercheurs dans divers domaines.

Nous proposons dans ce chapitre de concevoir un système multi-agents pour l'allocation des ressources dans le cloud computing. En effet, les systèmes multi-agents s'adaptent bien à la conception d'un système d'allocation des ressources où chaque membre doit gérer et échanger ses connaissances et collaborer avec les autres afin de réaliser ses buts. En plus, dans des tels environnements ouverts, dynamiques et complexes, il y a un besoin de distribuer les données, le contrôle ainsi que l'expertise ce qui rend l'utilisation des systèmes multi-agents bénéfique.

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. Les ressources des fournisseurs sont regroupées pour desservir plusieurs consommateurs en utilisant un modèle multi-locataire, avec des ressources physiques et virtuelles assigné et réassigné dynamiquement selon la demande des consommateurs. Il ya une indépendance de lieu, c-à-d que le client n'a pas généralement ni de contrôle ni de connaissance de l'emplacement exact des ressources fournies. Les ressources sont généralement hébergées dans des DataCenters. Le DataCenter est un ensemble de machines physiques qui sont interconnectés, virtualisées, et géographiquement distribuées.

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. L'allocation des ressources dans un environnement de 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 Modélisation conceptuelle d'allocation des ressources dans le C.C

Les fournisseurs de cloud computing (Cloud Providers) installent plusieurs data denter à différents endroits géographiques accessibles par Internet afin de répondre d'une façon optimale aux besoins de leurs clients à travers le monde **[42]**. Les deux principaux acteurs de l'environnement cloud sont les utilisateurs cloud et les fournisseurs cloud:

• Du point de vue des fournisseurs cloud (FCs), un grand nombre de ressources doit être affectées à des milliers utilisateurs répartis, d'une façon rentable, équitable, dynamique **[44]**. Les fournisseurs cloud sont modélisés comme un ensemble: FC = {FC_1, FC_2, ..., FC_n}, où n est le nombre des fournisseurs dans le cloud à un moment donné.

• Du point de vue des utilisateurs cloud (UCs), les utilisateurs sont des entités orientées économie quand ils prennent la décision d'utiliser les services de cloud **[44]**. Les utilisateurs cloud sont modélisés comme un ensemble UC = {UC_1, UC_2, ..., UC_m}, où m est le nombre d'utilisateurs qui demandent des ressources à un moment donné.

La **Figure 20** présente la modélisation conceptuelle de l'environnement Cloud Computing pour l'allocation des ressources.

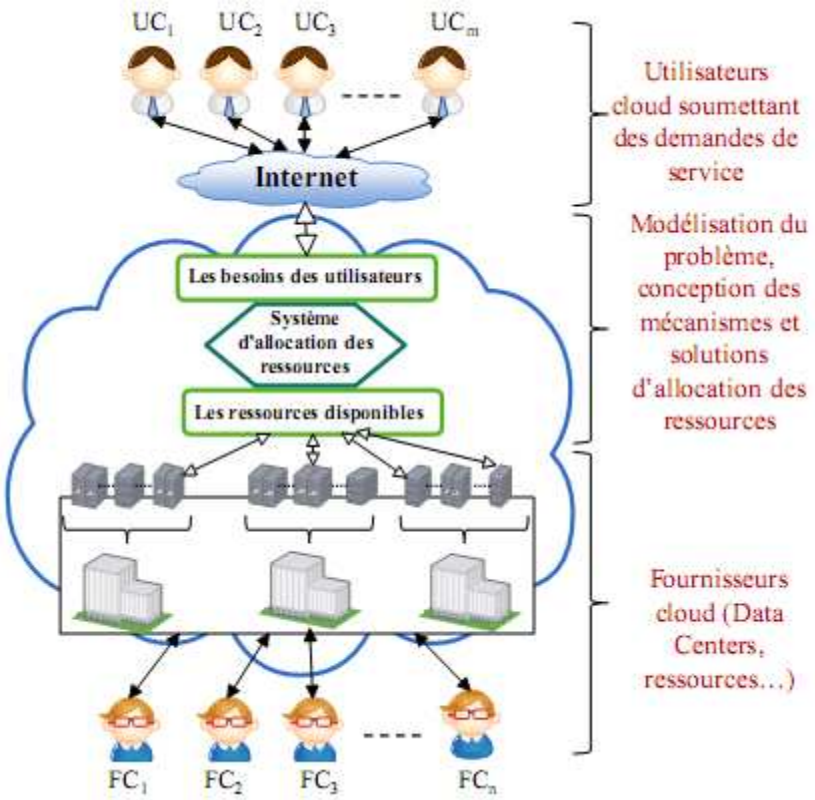


Figure 20. Modélisation conceptuelle de l'environnement cloud pour l'allocation des ressources. **[44-26]**

En supposant que les fournisseurs cloud (FC) possédant plusieurs Datacenters (DCs), chaque Datacenter comprenant un ensemble de nœuds physiques (PNs), sur chaque nœud physique résident de nombreuses machines virtuelles (VM). Chaque VM est appelé une ressource et elle peut être assigné à une partie de la demande d'utilisateur.

- ✓ Chaque fournisseur cloud $FC_{i=1..N}$ a plusieurs datacenters modélisées comme un ensemble : DC= $\{DC_1^{CP_i},\ldots,DC_j^{CP_i},\ldots,DC_d^{CP_i}\}$; où d signifié le nombre des datacenters fournies par le fournisseur CP_i a un moment donné.
- ✓ Chaque datacenter $DC_{j=1..d}$ est généralement composée d'un grand nombre des nœuds physique modélisés comme un ensemble PN= $\{PN_1^{DC_j},\ldots, PN_k^{DC_j},\ldots,PN_p^{DC_j}\}$; où p signifié le nombre des nœuds physique dans DC_j a un moment donné.
- ✓ Chaque nœud physique $PN_{k=1..p}$ dans le cloud peut exécuter plusieurs machines virtuels (VMs) modélisées comme un ensemble VM= $\{VM_1^{PN_k},\ldots,VM_l^{PN_k},\ldots,VM_w^{PN_k}\}$; où w signifié le nombre des machines virtuelle exécutées sur PN_k a un moment donné.

La **Figure 21** présente la modélisation conceptuelle de Cloud Provider pour l'allocation des ressources.

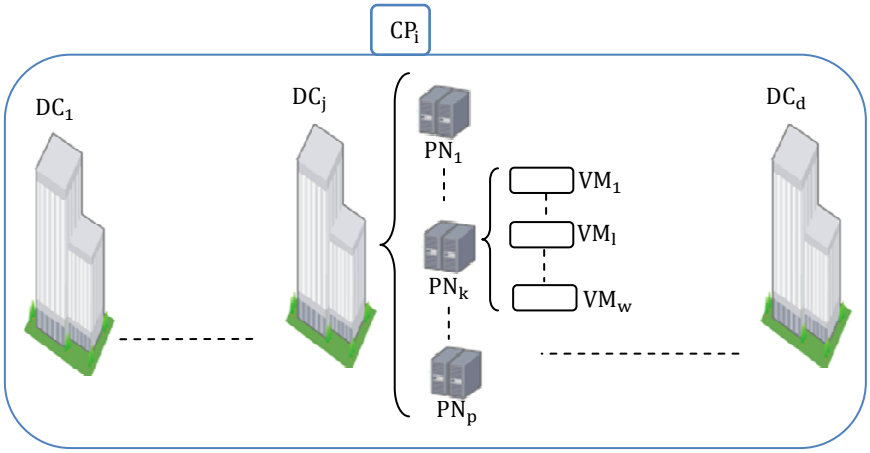


Figure 21. Modélisation conceptuelle d'un fournisseur cloud pour l'allocation des ressources

4 L'architecteur proposée

4.1 Architecture générale

Comme nous l'avons vu dans la section précédente, Le processus d'allocation des ressources dans le cloud computing exige des utilisateurs de cloud, des fournisseurs de cloud, ainsi qu'un système intermédiaire entre ces deux partenaires (système d'allocation des ressources (RAS)).

Sur cette étude, l'architecture proposée fournit une architecture afin de connecté les fournisseurs cloud de différents cloud hétérogènes avec les utilisateurs cloud, dont l'objectif est de permettre aux consommateurs de choisir les ressources appropriés qui peuvent satisfaire leurs besoins; au même temps elle permettre aux fournisseurs cloud d'allouent efficacement ces ressources.

Nous voulons fondre un SMA pour l'allocation des ressources dans le cloud computing. Pour cela, une organisation d'agents doit être conçue (un ensemble d'agents et des relations d'interaction inter-agents). Elle se compose des agents fournisseurs cloud (AFCs), des agents utilisateurs (AUCs), ces deux agents agissant au compte des fournisseurs de ressources et des

utilisateurs, respectivement. Un ensemble d'agents de courtage (ACs) doivent mener des activités de négociation à deux types (d'un coté avec l'agent utilisateur cloud et de l'autre coté avec l'agent fournisseur cloud).

Afin de bien gérer les ressources et découvrir les ressources disponibles dans les Datacenters, un autre ensemble d'agents est nécessaires (agents de gestion de Datacenter AGDCs). Finalement pour allouer les ressources d'un fournisseur cloud d'une manière efficace, un agent coordinateur (ACo) joue le rôle de coordination entre les différents agents de gestion de Datacenter.

L'organisation multi-agents, que nous proposons, définit les cinq agents suivants :

- Agents Utilisateur Cloud **(AUC)**: Il représente l'utilisateur dans l'environnement cloud, il participe dans le système et vise à obtenir un bénéfice maximal pour l'utilisateur.
- Agent Fournisseur Cloud **(AFC)**: Il représente le fournisseur dans l'environnement cloud, il participe dans le système et vise à obtenir un bénéfice maximal pour le fournisseur.
- Agent Courtier **(AC)**: Le rôle principal de l'agent courtier est de trouver la meilleure offre qui satisfait la requête de l'utilisateur dans le cloud.
- Agent Gestion de DataCenter **(AGDC)**: Sa tâche principale est de découverte les capacités des ressources sur les nœuds physiques –Physical Nodes- (PNs) et de gérer les machines virtuelles –Virtuels Machines (VMs) dans le Datacenter.
- Agent Coordinateur **(ACo)**: IL coordonne entre les AGDCs pour bien gérer et contrôler les ressources (physiques/virtuelles) d'un fournisseur cloud.

La **Figure 22** présente l'architecture multi-agents proposée pour l'allocation des ressources dans la cloud computing :

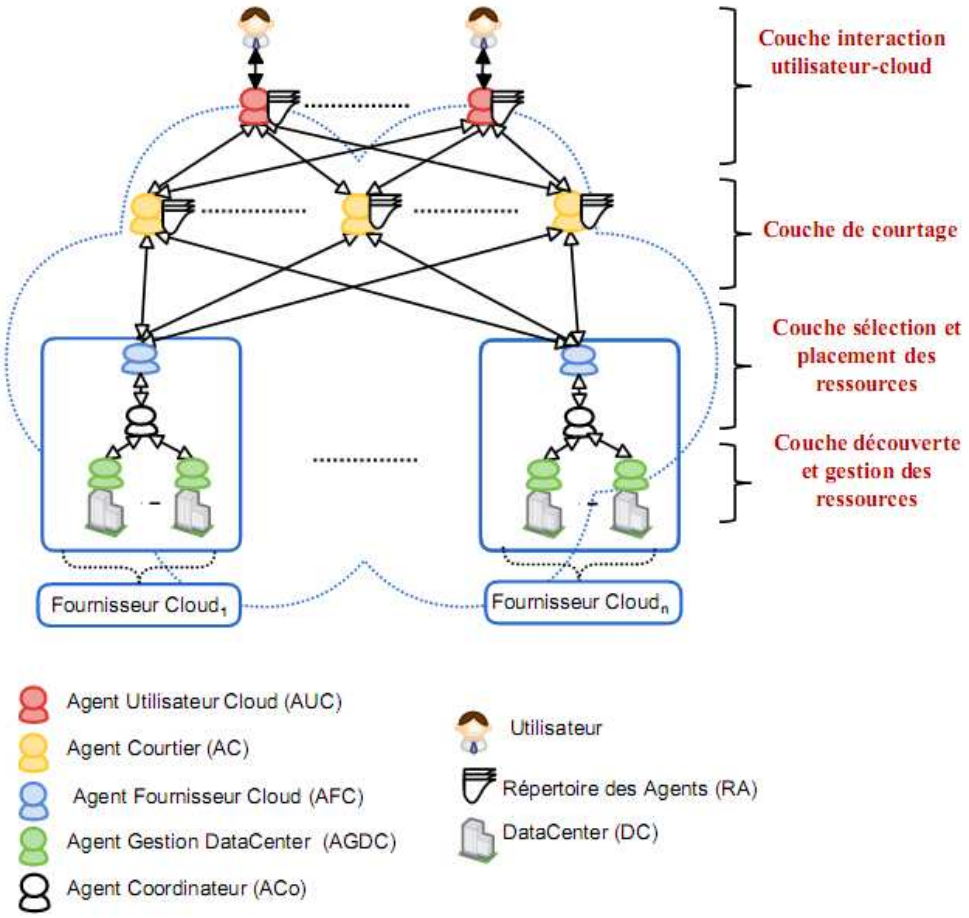


Figure 22. Architecture multi-agents pour l'allocation des ressources dans la cloud computing

L'architecture générale du notre système est de nature hiérarchique qui s'articule autour de quatre principales couches en interaction :

4.1.1 Couche d'interaction utilisateur-cloud

C'est la couche intermédiaire entre le système cloud computing et l'utilisateur cloud, elle permet aux utilisateurs cloud d'interroger le système. Cette couche offre une interface graphique d'utilisateur (IGU) (pour récupérer les besoins d'utilisateur, afficher les résultats d'allocation des ressources…etc.). Afin de préciser ces exigences, L'utilisateur doit interagit avec cette IGU, cette interaction traduit par l'agent utilisateur cloud (AUC) a une description de la demande d'utilisateur, qui doit être transmise ensuite a des agents courtiers. Les agents utilisateurs cloud aident les utilisateurs à trouver les ressources appropriées à leurs besoins et ils se comportent au nom de chaque utilisateur dans la phase de définition de la QoS requise et la formalisation de l'accord négocié entre les deux parties (utilisateur cloud et fournisseur cloud) **(Figure 23).**

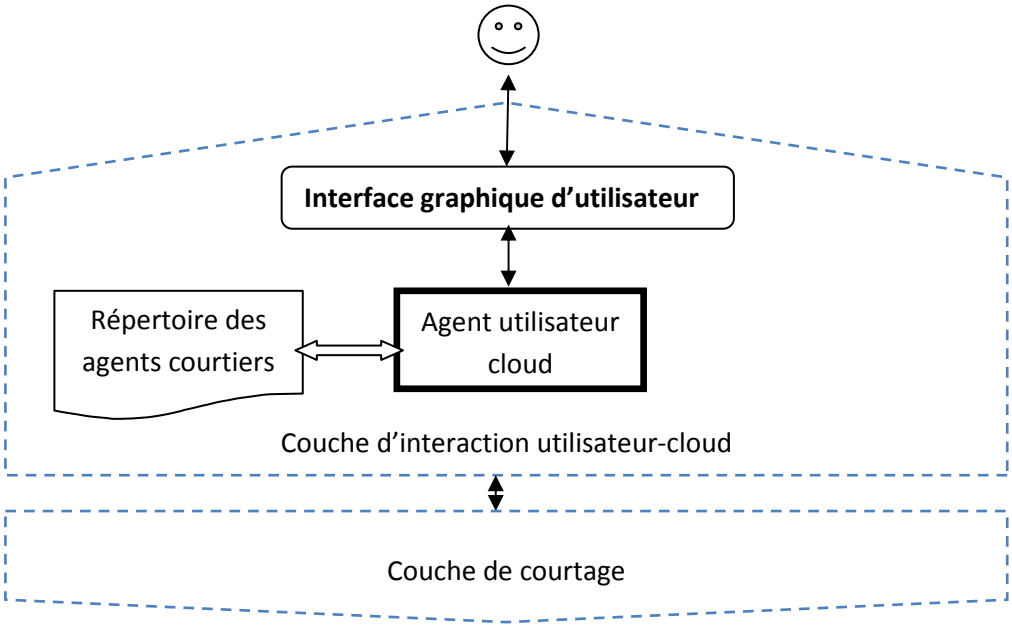


Figure 23. Schéma descriptif de la couche d'interaction utilisateur-cloud

4.1.2 Couche de courtage

C'est la couche intermédiaire entre la première couche (Couche d'interaction utilisateur-cloud) et les fournisseurs cloud, elle comprend des agents courtiers qui cherchant l'offre le plus approprié a la demande d'utilisateur parmi les offres des fournisseurs.

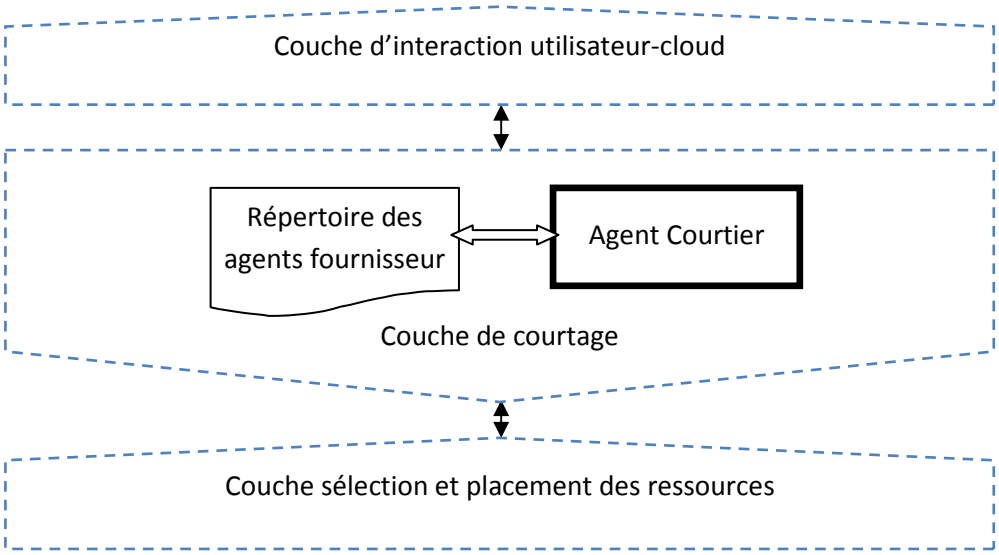


Figure 24. Schéma descriptif de la couche de courtage

Les agents courtiers sont menés par des activités de négociation des conditions mutuellement acceptables pour établir la SLA entre les fournisseurs et utilisateurs cloud, afin de satisfaire les exigences des utilisateurs cloud **(Figure 24)**.

4.1.3 Couche sélection et placement des ressources

Cette couche contient l'ensemble des agents fournisseurs cloud (**AFC**s) et les agents coordinateurs (**ACo**s). A ce niveau, et pour chaque demande d'utilisateur, l'agent fournisseur communique avec l'agent coordinateur (**ACo**) pour sélectionner les ressources convenables à chaque demande de ressources reçue, ce dernier coordonne entre les agents de la couche basse (plus précisément avec les agents de gestion de DataCenter -**AGDC**s) pour acquérir les capacités des ressources de chaque nœud physique **(Figure 25)**.

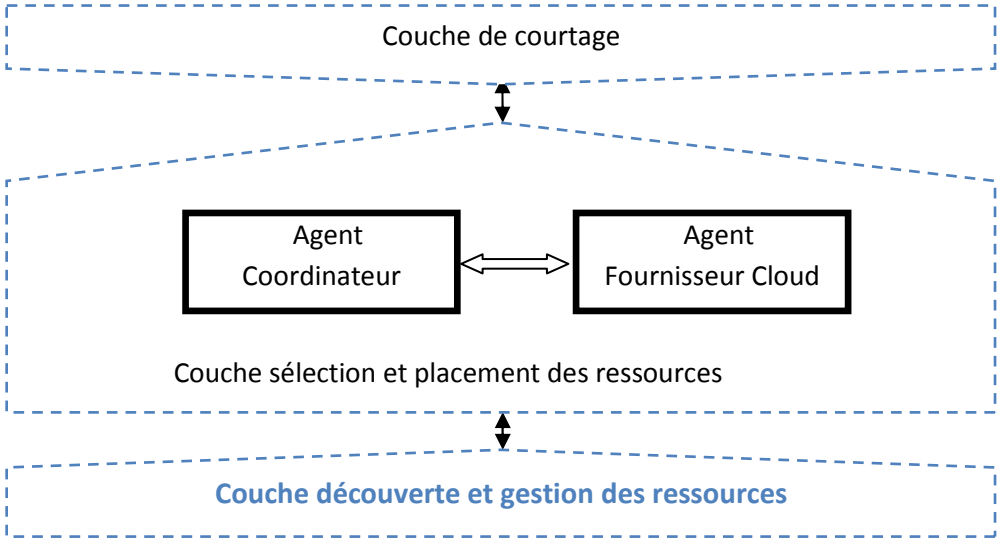


Figure 25. Schéma descriptif de la couche sélection et optimisation des ressources

4.1.4 Couche découverte et gestion des ressources

C'est la couche plus basse dans notre système, elle comprend un ensemble des agents de gestion de DataCenter (AGDCs), chaque AGDC est chargé de gérer les ressources d'un DataCenter, et de découvrir les capacités des ressources de chaque nœud physique a un moment donné (**Figure 26**).

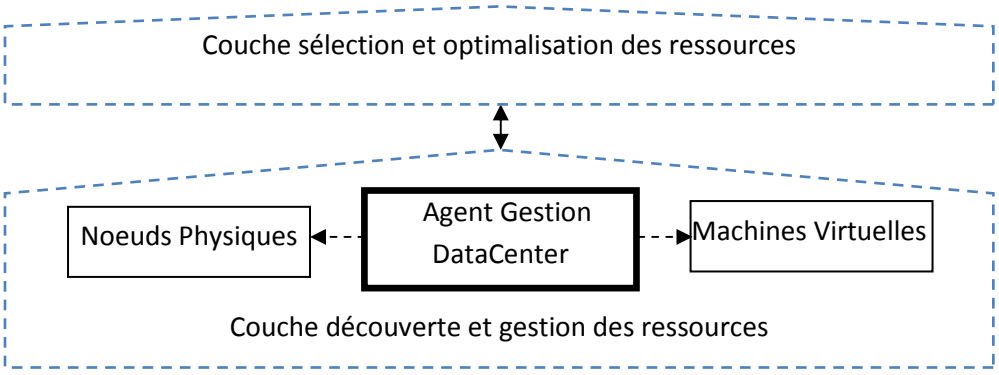


Figure 26. Schéma descriptif de la couche découverte et gestion des ressources

4.1.5 Le modèle économique utilisé

Un grand nombre de modèles économique existant. Certains utilisent des interactions simples. D'autres un mécanisme de négociation plus ou moins évolué. Le modèle que nous avons utilisé c'est le modèle d'appel d'offre.

4.1.5.1 Principe du modèle d'appel d'offre

Le modèle d'appel d'offres est inspiré des mécanismes mis en place dans le monde des affaires pour gouverner de biens et de services. Il permet au client de trouver le fournisseur de services approprié pour travailler sur une tâche donnée. Dans ce contexte, le client (courtier) est appelé "manager" et le fournisseur est un "contractant potentiel".

1. Chaque contractant potentiel s'enregistre auprès de l'annuaire.
2. Le manager contacte l'annuaire pour obtenir la liste des contractants potentiels.
3. Le manager lance un appel d'offres auprès des contractants potentiels, en leur fournissant ses exigences.
4. Chaque contractant potentiel l'évalue et lui soumet une offre s'il est intéressé.
5. Le manager choisit la meilleure offre et lui attribue le contrat.

4.1.5.2 Inspiration du modèle d'appel d'offre

En inspirant du modèle d'appel d'offre, nous allons utiliser un répertoire des agents contient une liste des contacts des agents qui va un agent les contactés pour accomplir ces tâches, chaque agent peut développer son propre répertoire (ajout, suppression…etc. des contacts) selon ces compétences. Dans notre système on a deux agents qui ont besoin de ce type de répertoire a savoir :

- ✓ L'agent utilisateur cloud: chaque agent utilisateur (AUC) détient un répertoire des agents courtiers (ACs) qui va les contactés pour rechercher les ressources appropriées a une demande d'utilisateur cloud (**Figure 27**).

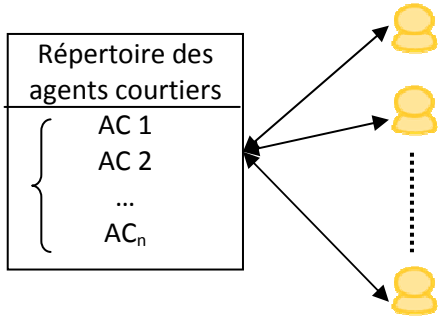


Figure 27. Répertoire des agents courtiers

- ✓ L'agent courtier : chaque agent courtier (AC) détient un répertoire des agents fournisseurs cloud (AFCs) qui va les contactés pour trouver les ressources appropriées a une demande d'utilisateur cloud (**Figure 28**).

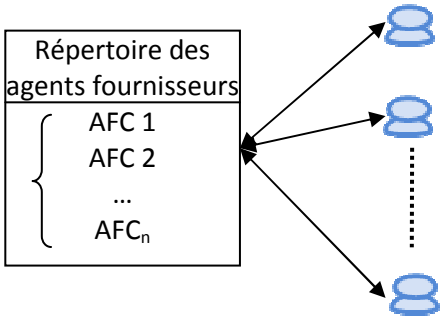


Figure 28. Répertoire des agents fournisseurs cloud

4.2 Architecture détaillée

4.2.1 Architecture interne des agents

Cette section focalise sur la conception de l'architecture interne des agents. Une telle architecture est définie comme étant un modèle structurel des composants qui constituent l'agent. Selon Wooldridge **[45]**, l'architecture interne d'un agent est une carte de ses modules internes, de ses structures de données ainsi que des opérations qui peuvent être appliquées à ces structures de données. Du point de vue de l'architecture interne, les différents agents de notre SMA se distinguent par la nature des modules internes qui les composent ainsi que par la manière dont ces modules sont imbriqués les uns aux autres. En effet, dans notre conception du SMA nous avons opté pour la décomposition de l'architecture interne des agents en plusieurs modules. Chaque module est dédié à une fonction spécifique requise pour l'accomplissement de la mission de l'agent **[46]**. Cette décomposition modulaire de l'architecture interne d'agents favorise l'amélioration, l'ajout, et la réutilisation des modules principaux.

4.2.1.1 Agent Utilisateur Cloud (AUC)

Il représente l'utilisateur dans l'environnement cloud. C'est un agent stationnaire qui s'exécute sur un périphérique de l'utilisateur et fournit une interface graphique pour permettre aux utilisateurs d'interagir avec le système, Il est responsable principalement d'acquérir les requêtes d'utilisateur, envoyer ces requêtes aux agents appropriés et présenter les résultats aux utilisateurs. Il participe dans le système et vise à obtenir un bénéfice maximal pour l'utilisateur (optimiser le rapport coût-performances, réduire le temps d'utilisation de ressource, minimiser le coût).

La **Figure 29** illustre l'architecture interne de l'Agent Utilisateur Cloud (AUC), ainsi que l'imbrication des modules utilisés. Ces principaux modules sont :

- *Un module d'interface*: Ce module offre des méthodes de base pour supporter les interactions du type humain/agent. Il offre en particulier des méthodes pour aider l'utilisateur à saisir ces besoins de la meilleure façon, des méthodes pour acquérir ses besoins, et d'autre pour l'affichage des résultats.
- *Un module de connaissances*: qui se décompose en deux parties:
  - ✓ les connaissances individuelles: reflètent la vue qu'a l'agent de lui-même, notamment :
    - son nom: Agent Utilisateur Cloud (AUC) ;
    - son adresse: s'exécute sur un périphérique de l'utilisateur ;
    - ses objectifs individuels: offre et gère l'interface graphique d'utilisateur (IGU), obtenir un bénéfice maximal pour l'utilisateur ;
  - ✓ ses différents états possibles: envoi de la requête d'utilisateur, attente des réponses des agents courtiers, affiche les résultats …etc.
  - ✓ les connaissances sociales: reflètent la représentation qu'a l'agent de l'environnement social dans lequel il évolue. Ces connaissances particulières portent sur:
    - les agents courtiers avec lesquels peut-il coopérer et coordonner.

- les protocoles de communication ou d'interaction à utiliser qui seront détaillés dans la section suivante.

- *Un module de traitement*: Au niveau de ce module, l'agent utilisateur cloud: analyse les données d'utilisateur, collecte les informations à partir de l'IGU, créer la requête qui représente les besoins d'utilisateur et la diffuser sur la liste des agents courtiers qui figurants dans son répertoire des agents courtiers. Il se base sur un ensemble de connaissances acquises et sur les repenses des agents courtiers afin de sélectionner les ressources a alloué pour l'utilisateur et afficher la meilleur allocation trouvée.
- *Un module de communication*: c'est un module pour la gestion des communications avec les autres agents. D'une part, ce module reçoit de la part du module de traitement des demandes de transmission de message à plusieurs destinations (diffusion sur l'ensemble des agents courtiers); de l'autre, il lui relaie les messages émanant des autres agents.

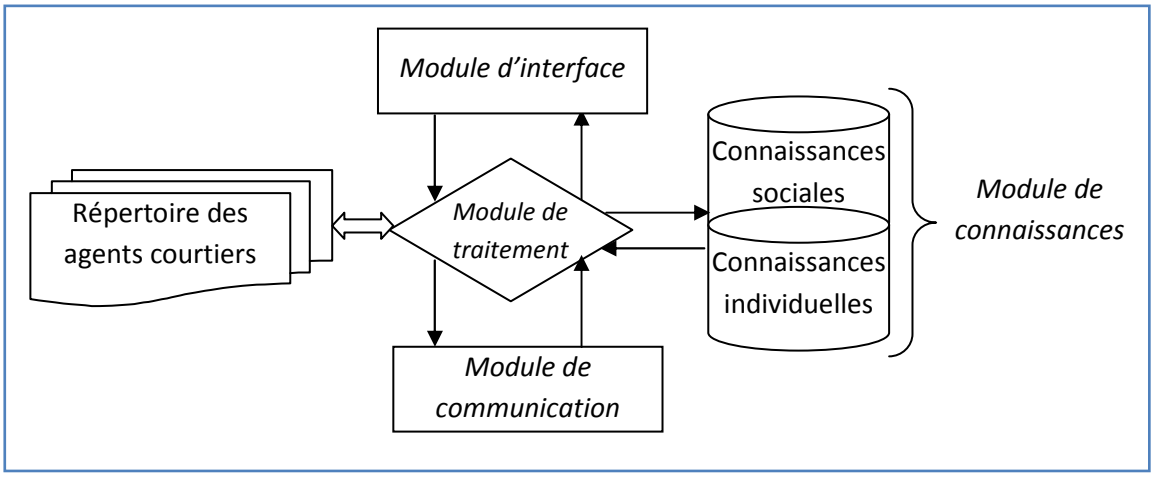


Figure 29. Architecture interne de l'Agent Utilisateur Cloud.

4.2.1.2 Agent Courtier (AC)

Le rôle principal de l'agent courtier est de trouver la meilleure offre convenable à la requête de l'utilisateur dans le cloud. Il reçoit la demande de l'agent utilisateur cloud et contacte les agents fournisseurs qui se trouvant dans son répertoire pour trouver les ressources appropriés a la demande reçue. Pour cela, l'agent courtier doit être mené par des méthodes de négociation des conditions mutuellement acceptables à deux types, d'un coté, il négocie avec l'agent utilisateur a fin de satisfaire leurs exigences, et de l'autre coté il négocie avec le fournisseur de ressources pour la réservation de ressources et l'établissement de SLA.

La **Figure 30** illustre l'architecture interne de l'Agent Courtier (AC), ainsi que l'imbrication des modules utilisés. Ces principaux modules sont :

- *Un module de connaissances*: qui se décompose en deux parties:
  - ✓ les connaissances individuelles: reflètent la vue qu'a l'agent de lui-même, notamment :
    - son nom: Agent Courtier (AC);
    - son adresse: s'exécute sur un site dans le cloud;

- ses objectifs individuels: trouver la meilleure offre convenable a la requête de l'utilisateur cloud, contacte les différents fournisseurs cloud pour un approvisionnement «sourcing» dynamique, collecte et combine les offres des fournisseurs cloud afin de proposer des offres globales à l'utilisateur cloud.
- ses différents états possibles: en attente des réponses des agents fournisseurs, en phase de négociation avec l'utilisateur cloud ou bien le fournisseur cloud.

✓ les connaissances sociales: reflètent la représentation qu'a l'agent de l'environnement social dans lequel il évolue. Ces connaissances particulières portent sur:

- les agents utilisateur, les agents fournisseurs avec lesquels peut-il coopérer et coordonner.
- les protocoles de communication ou d'interaction à utilisé qui seront détaillés dans la section suivante.

➢ *Un module de traitement*: Au niveau de ce module, l'agent courtier analyse la requête d'utilisateur cloud, et il exploite leur accointances pour répondre a cette requête de meilleure façon, pour ce faire il va dispatcher la demande sur les différents fournisseurs existants dans son répertoire des agents fournisseurs cloud, il collecte les réponses des agents fournisseurs cloud et trouve les offres appropriées a la requête d'utilisateur.

➢ *Un module de communication*: c'est un module pour la gestion des communications avec les autres agents (agents utilisateurs et agents fournisseurs). D'une part, ce module reçoit de la part du module de traitement des demandes de transmission de message à destination d'un ou plusieurs agents (liste des agents fournisseur, l'agent utilisateur); de l'autre, il lui relaie les messages émanant des autres agents.

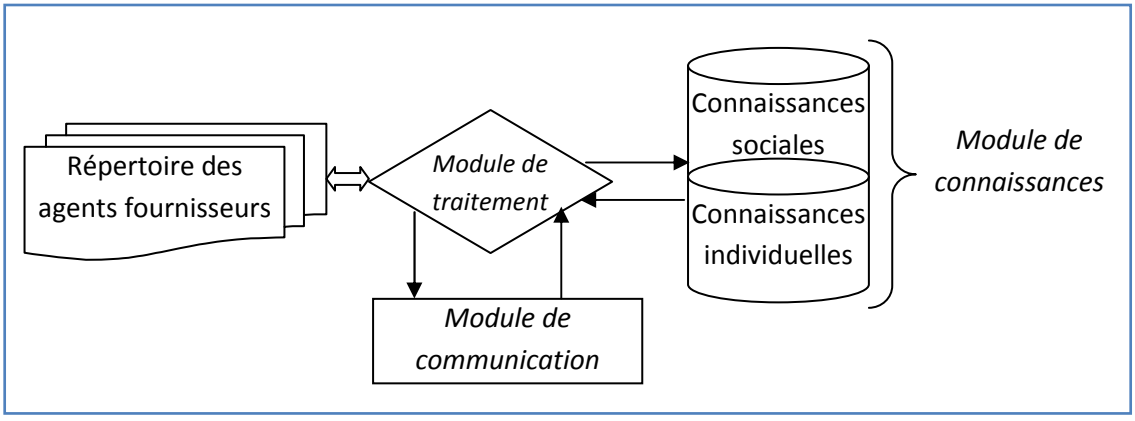


Figure 30. Architecture interne de l'Agent Courtier.

4.2.1.3 Agent Fournisseur Cloud (AFC)

Il représente le fournisseur dans l'environnement cloud, il participe dans le système et vise à obtenir un bénéfice maximal pour le fournisseur. L'objectif de l'agent fournisseur cloud est de maximiser que possible le revenu de fournisseur cloud avec un investissement minimum. À cet effet, il faut qu'il maximise l'utilisation de ces ressources, à titre d'exemple, par l'hébergement du plus grand nombre possible de machines virtuelles sur chaque machine. Pour cela il doit coopérer avec un agent coordinateur pour la bonne gestion des ressources dans les différentes Datacenters.

La **Figure 31** illustre l'architecture interne de l'Agent Fournisseur Cloud (AFC), ainsi que l'imbrication des modules utilisés. Ces principaux modules sont:

- *Un module de connaissances*: qui se décompose en deux parties:
  - ✓ les connaissances individuelles: reflètent la vue qu'a l'agent de lui-même, notamment :
    - son nom: Agent Fournisseur Cloud (AFC);
    - son adresse: s'exécute sur le site de fournisseur cloud;
    - ses objectifs individuels: construire un offre pour satisfaire la requête de l'utilisateur cloud, cet offre doit être basé sur les ressources disponibles dans touts les Datacenter qu'il détient, pour ce faire, il doit coordonner avec l'agent coordinateur. Son objectif principal est de maximiser que possible leur revenu avec un investissement minimum, et d'assurer les objectifs de niveau de services.
    - ses différents états possibles: en attente de réponse de l'agent coordinateur, en préparation d'une offre (prix, QoS…etc.), en phase de négociation d'une offre avec l'agent courtier.
  - ✓ les connaissances sociales: reflètent la représentation qu'a l'agent de l'environnement social dans lequel il évolue. Ces connaissances particulières portent sur:
    - les agents utilisateur et les agents courtiers avec lesquels peut-il coopérer et coordonner.
    - les protocoles de communication ou d'interaction à utiliser qui seront détaillés dans la section suivante.
- *Un module de traitement*: Au niveau de ce module, l'agent fournisseur cloud répond a une requête de courtier par son proposition (un offre contient la liste des ressources a allouées, prix, QoS…etc.). Pour ce faire une phase de coopération avec l'agent coordinateur doit être accomplit afin d'exploiter les ressources disponibles existants dans les différents Datacenter du fournisseur.
- *Un module de communication*: c'est un module pour la gestion des communications avec les autres agents (agent utilisateur et agents courtiers). D'une part, ce module reçoit de la part du module de traitement des demandes de transmission de message à destination d'un agent (agent courtier ou bien agent utilisateur); de l'autre, il lui relaie les messages émanant des autres agents.
- *Un module établissement d'offre* : ce module détaille l'établissement d'offre, basé sur la réponse de l'agent coordinateur concernant une demande d'utilisateur cloud, il ajoute l'attribut prix, et le niveau de SLA.

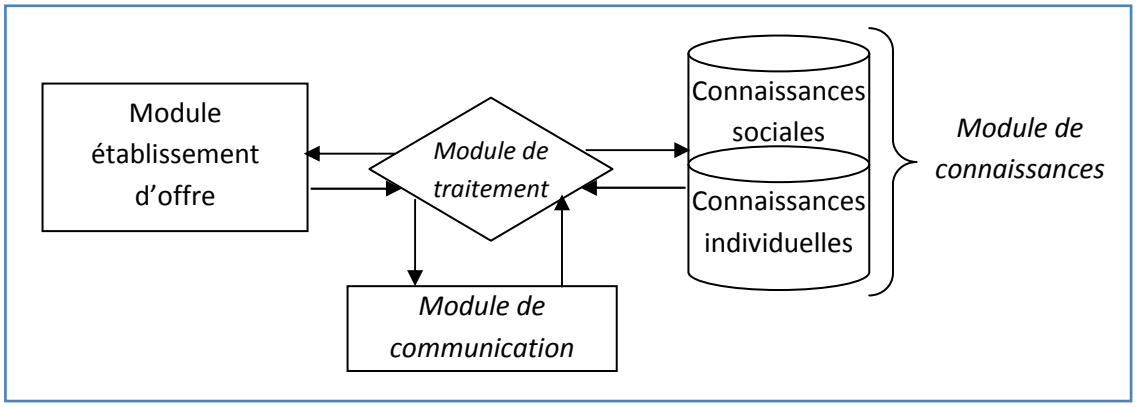


Figure 31. Architecture interne de l'Agent Fournisseur Cloud (AFC).

4.2.1.4 Agent Coordinateur (ACo)

IL est chargé par la coordination entre les différents agents de gestion de datacenters (AGDCs). A chaque fois qu'il reçoit une demande de la part de l'agent fournisseur cloud (AFC), il diffuse un message sur l'ensemble des agents de gestion des datacenters qui a comme objet demande des capacités de ressources des différents nœuds physiques en ce moment; une fois les réponses de ces derniers ont été bien reçues, il prépare l'état global des capacités de l'ensemble des nœuds physiques détenues par le fournisseur.

- *Un module de connaissances*: qui se décompose en deux parties:
  - ✓ les connaissances individuelles: reflètent la vue qu'a l'agent de lui-même, notamment :
    - son nom: Agent Coordinateur (ACo);
    - son adresse: s'exécute sur le site de fournisseur cloud;
    - ses objectifs individuels: coordonne avec les agents de gestion de datacenters afin d'avoir l'état actuel de toutes les ressources, répondre le plus vite possible à la requête de l'agent fournisseur.
  - ✓ ses différents états possibles: en attente de réponse des différents AGDCs, en cours de consolidation des différentes ressources disponibles, élaboration de la réponse pour l'agent fournisseur cloud (AFC).
  - ✓ les connaissances sociales: reflètent la représentation qu'a l'agent de l'environnement social dans lequel il évolue. Ces connaissances particulières portent sur:
    - les agents de gestion des DataCenters et l'agent fournisseur avec lesquels peut-il coopère et coordonne.
    - les protocoles de communication ou d'interaction à utiliser qui seront détaillés dans la section suivante.
- *Un module de traitement*: Au niveau de ce module, l'agent coordinateur traite la requête de l'agent fournisseur cloud, il base sur les réponses des AGDCs et le module de consolidation pour sélectionner les ressources afin de répondre a la demande de l'agent fournisseur cloud.
- *Un module de communication*: c'est un module pour la gestion des communications avec les autres agents (agent fournisseur cloud et les agents de gestion de datacenters). D'une part, ce module reçoit de la part du module de traitement des demandes de

transmission de message à destination d'un agent (agent fournisseur cloud ou bien les agents de gestion de datacenters); de l'autre, il lui relaie les messages émanant des autres agents.

- *Un module de consolidation* : ce module détaille la méthode de sélection et consolidation a utilisé par cet agent qui lui permettra d'améliorer sa fonction d'utilité ou d'atteindre son but.

La **Figure 32** illustre l'architecture interne de l'Agent Coordinateur (ACo), ainsi que l'imbrication des modules utilisés. Ces principaux modules sont:

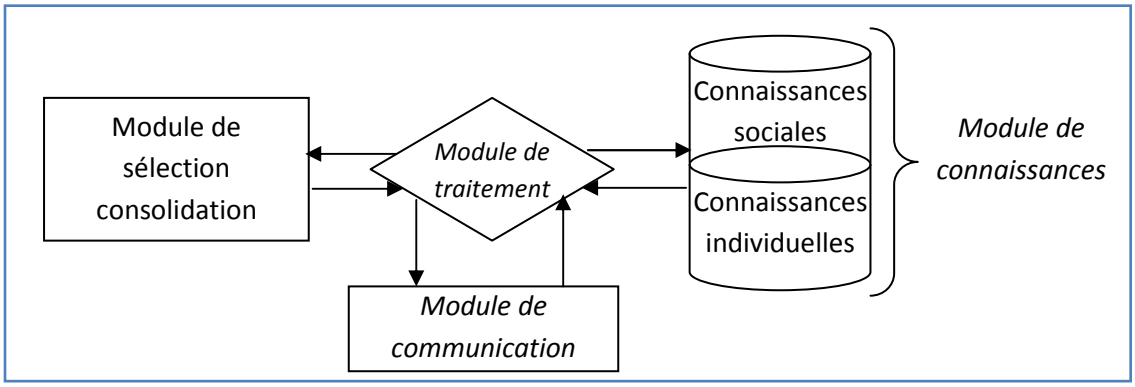


Figure 32. Architecture interne de l'Agent Coordinateur (ACo).

4.2.1.5 Agent Gestion de DataCenter (AGDC)

Comme son nom l'indique; Il est responsable de la gestion de Datacenter, il détient une base de donnée qui contient les adresses et les capacités totales de toutes les machines physiques, et il est chargé par les tâches suivantes:

- La découverte de l'état actuel de chaque machine (marche, arrêt, capacité utilisé, capacité libre…).
- Optimisation de l'utilisation des machines physiques, il peut réaliser les fonctions en dessous grâce à la migration (**Figure 33**) des machines virtuelles[16]:
  - ✓ *Équilibrage de charge* (**Figure 34**): les machines virtuelles peuvent être migrées dynamiquement en fonction de leur charge de travail, pour offrir une utilisation plus efficace des ressources informatiques.
  - ✓ *Consolidation des machines virtuelles* (**Figure 35**): les machines virtuelles qui ont des faibles charges de travail peuvent être consolidées sur un moindre nombre des machines physiques, ce qui rend possible de libérer des nœuds pour héberger des machines virtuelles qui ont des grandes charges de travail.
  - ✓ *Changement d'emplacement vers des nœuds plus puissants*: la migration est utilisée dans ce cas si une machine physique possède des bonnes caractéristique (plus puissante, de grande capacité…) est devenue disponible dans le Datacenter (nouvelle machine ajouté ou bien ancienne machine libérée…), et l'hébergement des machines virtuelles sur cette machine physique offre des atouts pour le système d'allocation des ressources.

[16] La migration est le fait de déplacer une machine virtuelle d'un nœud physique à un autre.

- La gestion des machines virtuelles (Création, Suspension, Reprise, Enregistrement, Restauration, Arrêt, Redémarrage …).

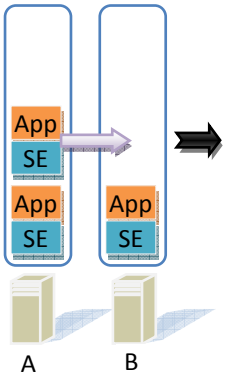


Figure 33. Migration de VM

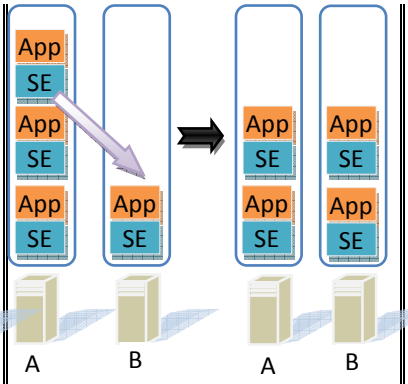


Figure 34. Equilibrage de charge

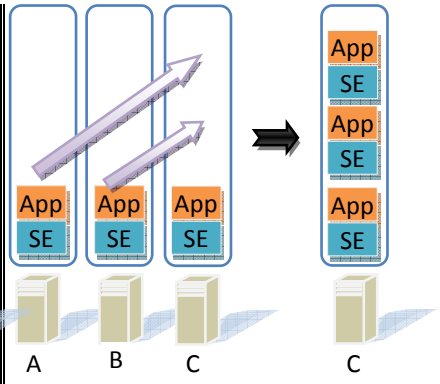


Figure 35. Consolidation des VMs

La **Figure 36** illustre l'architecture interne de l'agent de gestion de Datacenter (AGDC), ainsi que l'imbrication des modules utilisés. Ces principaux modules sont:

- *Un module de connaissances*: qui se décompose en deux parties:
  - ✓ les connaissances individuelles: reflètent la vue qu'a l'agent de lui-même, notamment :
    - son nom: l'Agent de Gestion de DataCenter (AGDC);
    - son adresse: s'exécute sur le DataCenter de fournisseur cloud;
    - ses objectifs individuels: gère l'ensemble des machines physiques de datacenter et optimise l'utilisation des ressources cloud, gère les machines virtuelles, avoir l'état actuel des ressources cloud a n'importe quel moment pour répondre le plus vite possible à la requête de l'agent coordinateur.
  - ✓ ses différents états possibles: en phase de mise à jour de la base de données des machines physiques, on phase d'optimisation d'utilisation des ressources cloud, en cours d'élaborer une réponse pour l'agent coordinateur (ACo).
  - ✓ les connaissances sociales: reflètent la représentation qu'a l'agent de l'environnement social dans lequel il évolue. Ces connaissances particulières portent sur:
    - l'agent coordinateur avec lequel peut-il coordonne et coopère.
    - les protocoles de communication ou d'interaction à utiliser qui seront détaillés dans la section suivante.
- *Un module de traitement*: Au niveau de ce module, l'agent de gestion de Datacenter exploite les données de la base de données des machines physiques et optimise l'utilisation des ressources cloud pour élaborer une réponse à l'agent coordinateur qui contient les ressources disponibles dans le Datacenter.
- *Un module de communication*: c'est un module pour la gestion des communications avec l'agent coordinateur. D'une part, ce module reçoit de la part du module de traitement des demandes de transmission de message à destination de l'agent coordinateur; de l'autre, il lui relaie les messages émanant de ce dernier.

- Un module de gestion des machines virtuelles: Ce module contient les mécanismes de gestion des machines virtuelles (Création, Suspension, Reprise, Enregistrement, Restauration, Arrêt, Redémarrage …).

En plus de ces modules l'agent de gestion de Datacenter détient une base de données contient l'ensemble des machines physiques existants dans le Datacenter.

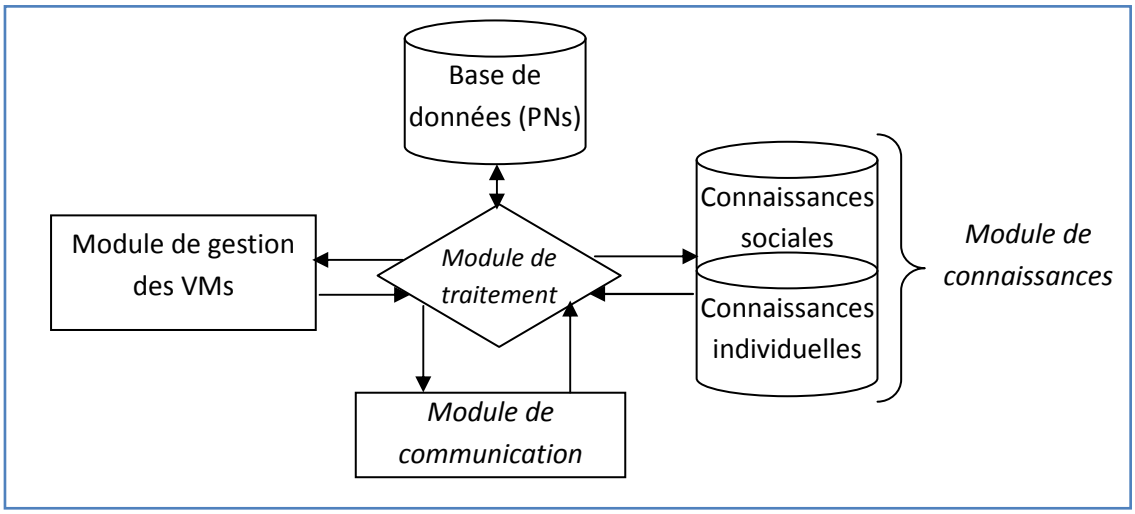


Figure 36. Architecture interne de l'Agent de gestion de Datacenter (AGDC).

4.2.2 Déroulement d'un processus d'allocation de ressources

Dans cette partie, nous présentons un exemple de scénario de déroulement d'un processus d'allocation des ressources dans le cloud computing (**Figure 37**).

1. L'utilisateur de cloud demande les services souhaités à travers une interface graphique d'utilisateur (IGU). Ces services sont prétraités a fin d'extraire les exigences de l'utilisateur, A la fin de cette étape, l'agent utilisateur cloud (AUC) va créer une demande des besoins nécessaires.
2. L'agent utilisateur cloud (AUC) diffuse la demande de l'utilisateur sur l'ensemble des agents courtiers (ACs) figurant dans son propre répertoire des agents courtiers.
3. Chaque agent courtier (AC) va dispatcher la même demande reçue sur l'ensemble des agents fournisseurs cloud (AFCs) figurant dans son propre répertoire des agents fournisseurs cloud.
4. A fin de préparer son offre pour répondre a une demande reçue de la part de l'agent courtier, l'agent fournisseur cloud va extraire les besoins en ressources de cette demande. A ce niveau on suppose que le service demandé par l'utilisateur est composé d'un ensemble d'applications (a_1, a_2, …, a_v), l'exécution de chaque application exige des ressources multidimensionnelles qui doivent être virtualisées sous la forme d'une machine virtuelle hébergée sur un nœud physique. Soit R la requête de l'utilisateur:
   - ✓ v le nombre de machines virtuelles nécessaire pour satisfaire la requête de l'utilisateur.
   - ✓ w la dimension de ressource de chaque machine virtuelle.
   - ✓ r_{ij} la capacité de la ressource i de la machine virtuelle j où $1 \leq i \leq q$ et $1 \leq j \leq w$.

$$R = \begin{array}{c} \begin{matrix} R_1 & R_2 & \cdots & R_w \end{matrix} \\ \begin{pmatrix} & & \cdots & \\ & & \cdots & \\ \vdots & \vdots & \ddots & \vdots \\ & & \cdots & \end{pmatrix} \end{array} \begin{matrix} VM_1 \\ VM_2 \\ \vdots \\ VM_v \end{matrix}$$

Ensuite, la demande sera transmette â l'agent coordinateur pour rechercher une sélection de ressources qui satisfaite la demande des ressources parmi l'ensemble des ressource délocalisées sur les différentes datacenter.

5. L'agent coordinateur (ACo) demande de différents agents de gestion des DataCenters d'établir la situation actuelle des différentes ressources.

6. L'agent de gestion de datacenter (AGDC) répondre par la situation actuelle de chaque machine physique figure dans le datacenter qu'il s'occupe. Soit C_{DC} l'état des ressources dans DataCenter (DC):

- ✓ *p* le nombre des nœuds physiques dans une DataCenter.
- ✓ *w* la dimension de ressource de chaque machine physique.
- ✓ C_{ij} la capacité de la ressource i de la machine physique j où 1≤i≤p et 1≤j≤w.

$$C_{DC} = \begin{array}{c} \begin{matrix} C_1 & C_2 & \cdots & C_w \end{matrix} \\ \begin{pmatrix} & & \cdots & \\ & & \cdots & \\ \vdots & \vdots & \ddots & \vdots \\ & & \cdots & \end{pmatrix} \end{array} \begin{matrix} PN_1 \\ PN_2 \\ \vdots \\ PN_p \end{matrix}$$

7. L'agent coordinateur (ACo) combine les réponses de l'ensemble des agents de gestion de datacenter pour établir la situation de tous les nœuds physiques de fournisseur cloud délocalisées sur les différentes datacenters ($E_{FC} = \cup E_{DC}$). Soit C_{FC} l'état de ressources du fournisseur cloud (FC).

$$C_{FC} = \begin{array}{c} \begin{matrix} C_1 & C_2 & \cdots & C_w \end{matrix} \\ \begin{pmatrix} & & \cdots & \\ & & \cdots & \\ \vdots & \vdots & \ddots & \vdots \\ & & \cdots & \end{pmatrix} \end{array} \begin{matrix} PN_1 \\ PN_2 \\ \vdots \\ PN_q \end{matrix}$$

- ✓ *q* le nombre des nœuds physiques du fournisseur cloud.
- ✓ *w* la dimension de ressource de chaque machine physique.
- ✓ C_{ij} la capacité de la ressource i de la machine physique j où 1≤i≤q et 1≤j≤w.

En suite L'agent coordinateur va rechercher la bonne sélection qui satisfaite la demande d'utilisateur, et il la envoyer a l'agent fournisseur cloud (AFC).

On dit que le nœud physique PN_k peut héberger la machine VM_l si et seulement si: $R_{li} \leq C_{ki} / \ \forall i \epsilon \{1..w\}$.

Pour facilite la modélisation du problème, nous adoptant un vecteur (V) de bit (0/1) pour chaque nœud physique, il prend la valeur 1, si le nœud physique PN_k peut héberger la machine VM_l. donc ce vecteur sera de dimension égale au nombre des machines virtuelles.

V_{ki} = 1 si le nœud physique K peut héberger la machine VM_i ($1 \leq i \leq v$), et la valeur 0 autrement.

Ce qui nous donne par la suite une matrice binaire P (0/1) de dimension (qxv) :

$$P_{FC} = \begin{array}{c} \begin{matrix} VM_1 & VM_2 & \cdots & VM_v \end{matrix} \\ \begin{pmatrix} & & \cdots & \\ & & \cdots & \\ \vdots & \vdots & \ddots & \vdots \\ & & \cdots & \end{pmatrix} \end{array} \begin{matrix} PN_1 \\ PN_2 \\ \vdots \\ PN_q \end{matrix}$$

$$\begin{cases} P_{kl} = 1, \text{si la machine } PN_k \text{ peut héberger la machine} VM_l. \\ P_{kl} = 0 \text{ autrement.} \end{cases}$$

Pour une allocation correcte des ressources, la formule suivante doit être vérifiée :

$$\sum_{l=1}^{v} R_{lz} \cdot P_{kl} \leq C_{kz} \qquad \text{Où } \begin{cases} 1 \leq k \leq q \\ 1 \leq z \leq w \end{cases}$$

8. Après la réception de la liste des ressources sélectionnées par l'agent coordinateur (ACo), l'agent fournisseur cloud (AFC) ajoute l'attribut prix pour préparer son offre et envoyer ce dernier à l'agent courtier demandeur.

9. L'agent courtier (AC) choisi parmi les offres qu'il a reçu, les offres les plus adéquats à la requête de l'utilisateur et les envoyées comme réponse vers l'agent utilisateur cloud (AUC).

10. S'il ya un offre qui correspond totalement a la demande d'utilisateur, L'agent utilisateur cloud (AUC) affiche cet offre a l'utilisateur cloud, sinon, dans le cas ou il ya une correspondance partiel, l'agent courtier compose un offre pour satisfaire complètement la demande d'utilisateur.

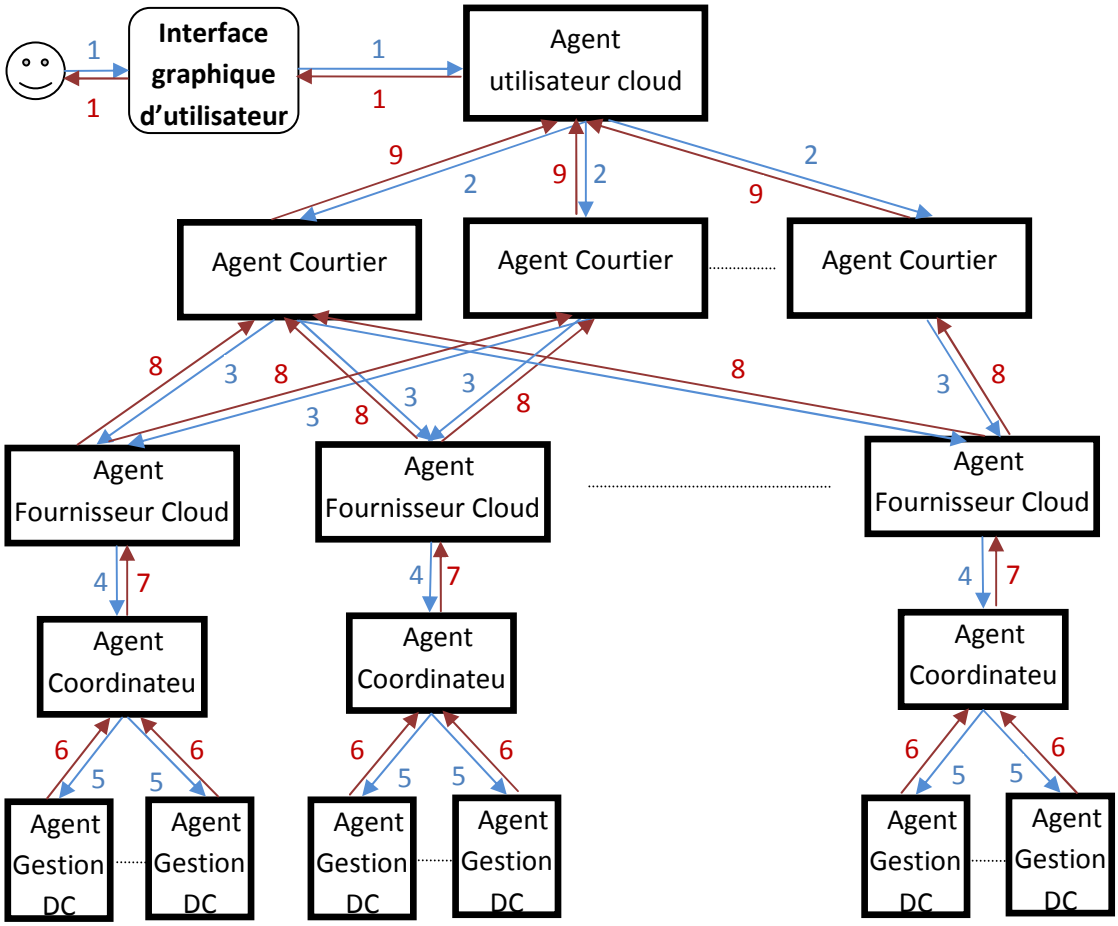


Figure 37. Déroulement d'un processus d'allocation de ressources

4.2.3 Diagramme de Cas d'utilisation UML

Nous avons opté pour l'approche UML pour présenter les fonctionnalités du système. Nous avons utilisé le diagramme de cas d'utilisation UML pour présenter les fonctionnalités du système et les acteurs qui participent à son fonctionnement.

L'objectif des cas d'utilisation UML est l'expression des besoins en termes de services que doit assurer le système. Les diagrammes de cas d'utilisation définissent deux concepts principaux : les acteurs et les cas d'utilisation.

- Un acteur est une entité du système qui peut initier l'un de ses cas d'utilisation.
- Un cas d'utilisation est une fonctionnalité offerte par le système.

Dans la **Figure 18**, nous présentons le diagramme de cas d'utilisation de notre système.

Le client final (utilisateur cloud) et l'Agent Fournisseur Cloud (AFC) représentent les acteurs principaux et les autres acteurs représentent les acteurs secondaires. Les acteurs secondaires sont: L'agent utilisateur cloud (AUC), L'agent courtier (AC), L'Agent Coordinateur (ACo), L'Agent de gestion de DataCenter (AGDC).

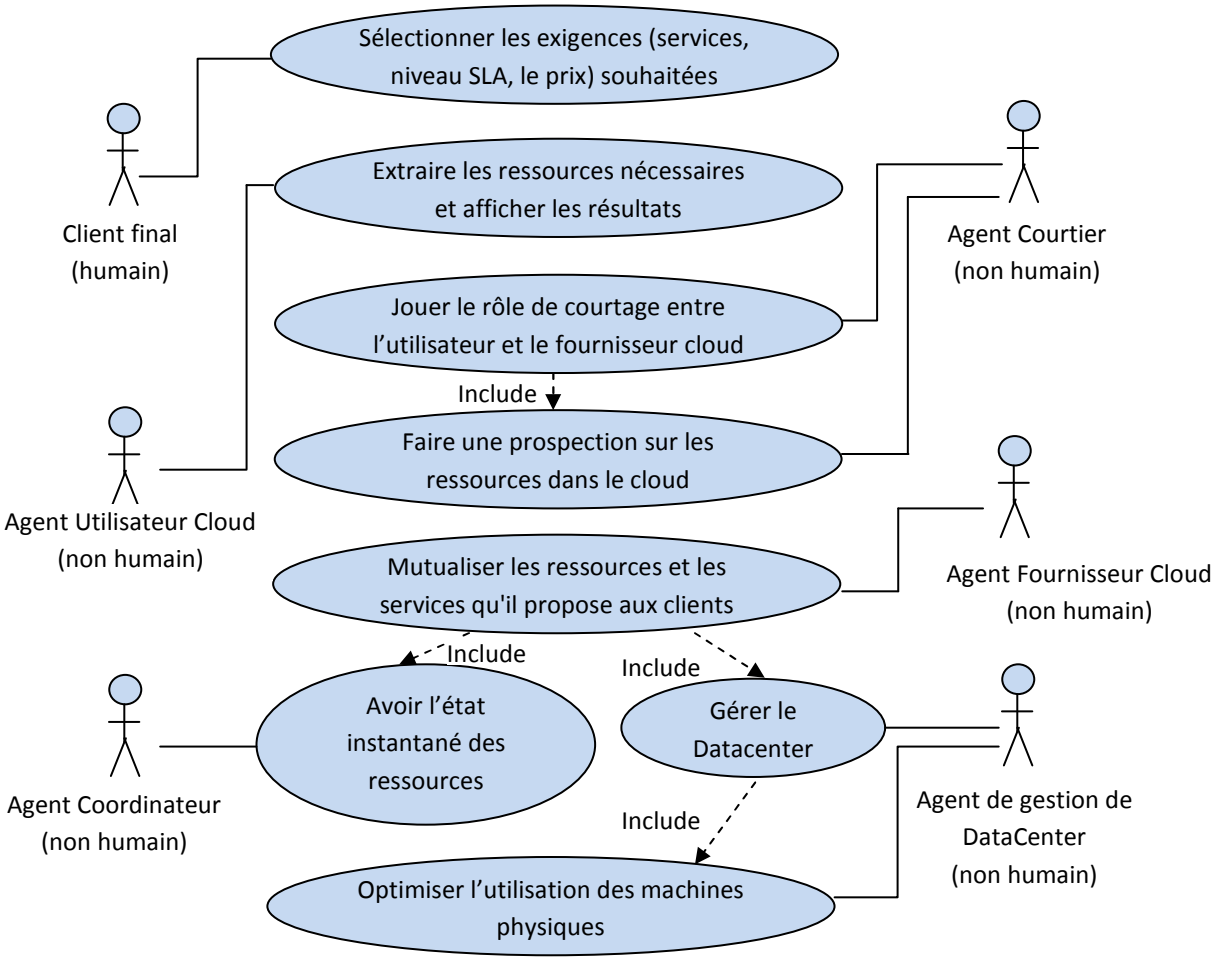


Figure 38. Le diagramme de cas d'utilisation du système.

De plus, à travers ce diagramme nous avons déterminé les fonctions de notre système que sont:

- Sélectionner les exigences (services, niveau SLA, le prix) souhaitées ;
- Extraire les ressources nécessaires et afficher les résultats ;
- Jouer le rôle de courtage entre l'utilisateur et le fournisseur cloud ;
- Faire une prospection sur les ressources dans le cloud ;
- Mutualiser les ressources et les services qu'il propose aux clients ;
- Avoir l'état instantané des ressources délocalisées sur les DC ;
- Gérer le Datacenter ;
- Optimiser l'utilisation des machines physiques.

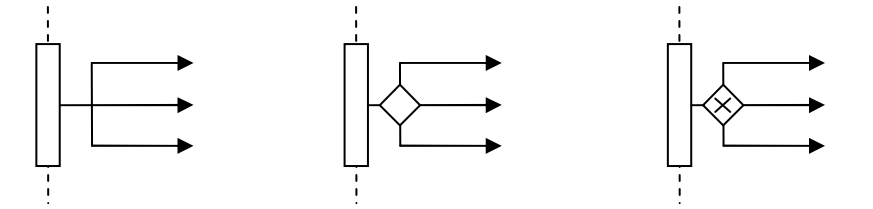
4.2.4 Interaction entre les agents en AUML

Dans cette section nous modélisons les interactions entre les agents dans notre SMA, pour ce faire nous avons choisi d'utiliser AUML (Agent UML), une extension du standard UML qui a été proposée dés le début des travaux sur les SMA. L'avantage de cette approche est qu'elle est basée sur UML destinée à la POO et très bien connue dans le domaine de génie logiciel, ce qui permet aux développeurs de passer facilement d'une méthodologie orientée-objet vers l'approche orientée-agent.

4.2.4.1 Présentation de AUML

AUML a été proposée pour la première fois par (Bauer, 1999), qui a suggéré deux nouvelles spécifications pour pouvoir présenter les interactions: le diagramme de séquence et le diagramme de classe d'agent. Nous détaillons ces deux spécifications **[47]**:

1) Les diagrammes de séquence
   Appelés aussi *diagrammes de protocole*, leur objectif est de montrer l'échange de messages entre les agents dans le temps en utilisant les protocoles de communication. Ces diagrammes sont composés de deux axes : un vertical représentant le temps, et un horizontal représentant différents agents ou différents rôles d'agents. Plus trois connecteurs AND, OR et XOR, (la **Figure 39** montre ces trois types de connecteurs). Dans les diagrammes de séquences AUML il est possible d'envoyer un message à plusieurs agents ou rôles au même temps.



a) L'opérateur AND b) L'opérateur OR c) L'opérateur XOR

Figure 39. Types de connecteurs dans AUML

2) Les diagrammes de classes d'agent
   Les diagrammes de classes UML montrent un ensemble de classes avec leurs attributs, leurs opérations ou les méthodes et les relations entre les différentes classes. Considérant la différence entre un objet et un agent, ces diagrammes doivent être modifiés pour qu'ils correspondent aux caractéristiques des agents. Les diagrammes résultants sont appelés des diagrammes de classes d'agent. Ils contiennent les éléments suivant : le nom de l'agent, une description d'état (vs attributs), les rôles (actions, méthodes, capacités et protocoles), la position organisationnelle de l'agent (sa représentation dans son groupe).

4.2.4.2 Le diagramme de séquence

Nous présentons ci-dessous les différentes interactions entre les agents, nous commençons par présenter les actes utilisés pour la communication entre agents, ensuite nous modélisons les différentes interactions par un diagramme de séquence AUML.

4.2.4.2.1 Actes échangés entre les agents

Le tableau suivant **(Tab 4)** présente les actes utilisés, et pour chacun des actes Nous définissons: L'émetteur, le récepteur, un code pour l'identifier sur le diagramme de séquence du protocole d'interaction, ainsi que la signification de chaque acte.

Acte	Code	Emetteur	Récepteur	signification
DEM_Sce	a	Client final	AUC	– Les services souhaités (puissance de calcul, de stockage, d'archivage, de bande passante …etc.) – Prix [Min, Max] – Le niveau de service (SLA)
REQ_Ut	b	AUC	AC	– Les besoins en ressources (machines, stockage, archivage, réseau) – Prix [Min, Max] – Le niveau de service (SLA) – Code de Requête.
DEM_R	c	AC	AFC	– Une demande des ressources (machines, stockage, archivage, réseau)
DEM_R_DISPO	d	AFC	ACo	– Disponibilité de ressources
DEM_ETAT_DC	e	ACo	AGDC	– L'état instantané du DataCenter
REP_ETAT_DC	f	AGDC	ACo	– L'état de chaque ressource figure dans le DataCenter
REP_R_DISPO	g	ACo	AFC	– Une liste contient les ressources disponibles
REP_R	h	AFC	AC	– Code de Requête. – La liste des ressources disponibles – Prix
REP_REQ_Ut	I	AC	AUC	– Les offres adéquates à la demande de l'utilisateur.
REP_Sce	k	AUC	Client final	– Résultat de recherche de service (le meilleur service avec le meilleur prix)

Table 4. Les actes échangés entre les agents.

4.2.4.2.2 Interaction entre agents

La **Figure 40** présente le protocole de l'interaction globale entre les différents agents.

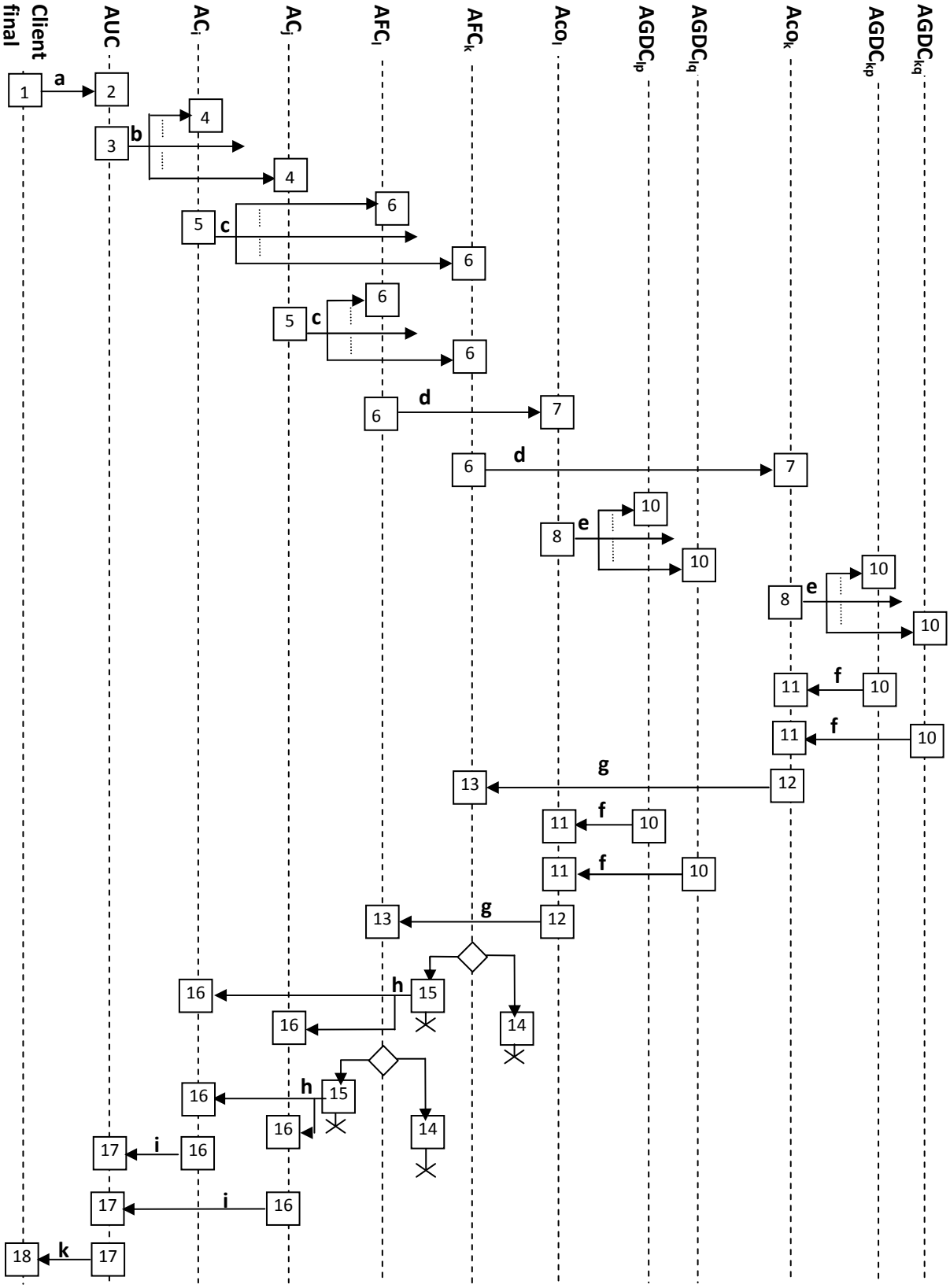


Figure 40. Interactions entre les agents.

5 Conclusion

Dans ce chapitre, nous avons présenté les principales phases et activités de nos contributions, à savoir :

- Une modélisation conceptuelle d'allocation des ressources dans le cloud computing.
- La proposition d'une approche basée agents pour l'allocation des ressources dans le cloud computing.

Au cours du chapitre, on a présenté la conception globale de l'approche proposée qui basée sur un système multi-agents, ensuite nous avons détaillée la conception de notre système. Nous avons exprimé notre démarche de conception à l'aide du langage UML, et plus particulièrement le diagramme de cas d'utilisation pour exprimer les fonctionnalités du système et les acteurs qui participent à son fonctionnement et les diagrammes de séquence pour exprimer le comportement du système proposé.

Dans le chapitre suivant, nous nous intéressons à la concrétisation de notre système proposé et cela par l'implémentation de l'approche présenté dans un environnement comporte le simulateur CloudSim et la plateforme Jade.

CHAPITRE IV

Simulation et Résultats

1 Introduction

Ce chapitre décrit la phase de validation de notre approche proposée pour l'allocation des ressources dans le cloud computing, qui consiste a exploité le paradigme agent pour concevoir un système d'allocation des ressources (en anglais RSA :- ressource allocation system). Nous avons essayé par le présent travail de répondre au cahier des charges initialement définit, qui nécessite le développement d'un système multi-agents, qui a pour objectifs : optimiser l'allocation des ressources dans le cloud computing, atteindre la satisfaction de l'utilisateur et de maximiser les bénéfices des fournisseurs de services de cloud, Au cours de ce chapitre, nous avons expérimenté nos propositions a l'aide de CloudSim et Jade. Plusieurs séries de simulation ont été menées, nous présentons quelques résultats et leurs interprétations. Nous commençons ce chapitre par la description de quelques interfaces graphiques et par la suite nous discuterons et analyserons les résultats obtenus.

2 Démarche utilisée

L'objectif de ce travail consiste à satisfaire le client en termes de qualité de traitement en minimisant le coût dans le cloud computing et par conséquent réduire le budget des clients. Afin de satisfaire ces objectifs nous avons proposé une approche basée agents pour l'allocation des ressources. Le but est d'offrir aux utilisateurs la possibilité d'acquérir les ressources demandées en un minimum de prix, pour cela une classe d'agents déjà détailles dans le chapitre précédent, a été développée :

- Des Agents Utilisateurs Cloud (AUC) ;
- Des Agents Coutiers (AC) ;
- Des Agents Fournisseurs Cloud (AFC) ;
- Des Agents Coordinateurs (ACo) ;
- Des Agents de Gestion des Data Centers (AGDC).

La **Figure 41** illustre l'idée générale de notre contribution qui consiste a

1. Crée un environnement de cloud computing: comporte les acteurs cloud (fournisseurs cloud, utilisateurs cloud, courtiers …), et les ressources cloud (ensemble DataCenter, où chacun possède un nombre de Host dans lesquels nous pouvons créer des machines virtuelles)
2. Développer un système multi-agents pour l'allocation des ressources aux utilisateurs cloud.

Figure 41. Idée générale de notre contribution

Généralement, chaque infrastructure a ses propres caractéristiques, protocoles, architecture, ainsi que le simulateur que permet de tester les performances de celles-ci sur le plan théorique avant de l'appliquer dans le monde réel. Nous avons constaté que l'infrastructure Cloud utilise un nouveau paradigme qui n'existe pas les infrastructures précédentes. Ce paradigme est la virtualisation des ressources (ressources de calcul, de stockage …), qui a présenté beaucoup d'avantages dans le traitement des requêtes venant des utilisateurs.

Pour cela nous avons simulé en premier l'environnement cloud computing à l'aide de CloudSim. Les concepteurs de simulateurs Cloud ont proposé des méthodes pour facturer l'utilisation des ressources commercialisées dans le Cloud. Dans l'outil CloudSim les ressources ont un prix unitaire qui constant, définit par les concepteurs au préalable ou peut être modifiée par l'utilisateur. Par exemple :

Le cout de transfert des données en GB (Giga Byte) : DataCost=DataGB ×CostPerDataGB.

3 Métriques utilisées

Tout au long de notre étude nous nous sommes intéressés à deux métriques qui nous semblent importantes dans le processus d'allocation des ressources dans le cloud computing qui sont :

1. Mesurer la performance de l'approche d'allocation des ressources proposée dans le cloud computing.
2. Mesurer l'aspect de qualité de service (QoS) pour les clients et pour les fournisseurs.

3.1 Mesure de la performance

L'étude des performances d'un système est mesurée généralement par la rapidité d'envoyer la réponse a un utilisateur.

Le plus souvent. Cette rapidité est calculée à l'aide de temps de réponse effectué pour satisfaire une requête d'un client. Dans notre étude, nous avons calculé la durée écoulée par chaque agent utilisateur cloud pour répondre à la demande de l'utilisateur.

$$TempsEcoulé\_AUC_i = DateArrivéRequête - DateSatisfactionRequête$$

3.2 Mesures de la qualité de service

La qualité du service (QoS) dans notre étude peut se résumer en deux catégories : Assurer la qualité de service pour les fournisseurs de ressources. Ou pour les clients, nous nous sommes intéressés a la satisfaction de la clientèle qui consiste à diminuer le cout des ressources commercialisées afin d'essayer d'utiliser le maximum de service a un cout minimal.

Une des améliorations apportées dans notre travail est le paramètre propre à l'utilisateur qui est le **budget** afin de lui permettre de gérer son budget selon le nombre de services ou d'applications qu'il possède. L'agent utilisateur cloud (AUC) va s'enchérir avec la couche de courtage (plus précisément les Agents Courtier (AC)), où l'utilisateur souhaite traiter l'ensemble de ses services ou bien applications au moindre coût. Dans notre projet on suppose que le coût service ou bien application égale a la somme des coûts de consommation chaque ressource utilisée dans le traitement: $Coût\_Sce/App = \sum CoûtConsommationChaqueRessource$

4 Structure de système proposé

La structure statique de notre système est un ensemble de classes qui peuvent être divisés en deux parties :

1ére partie : elle englobe les classes utilisées pour la simulation de l'environnement cloud (**Figure 42**), nous citons : ENVCC, VM, REQ_USER.

- La classe *ENVCC* : permet de générer l'environnement de cloud computing (les Data center de chaque fournisseur cloud ainsi que les nœuds physiques de chaque Data Center). Elle contient comme principaux attributs :
  - VMList : contient la liste des machines virtuelles (VM).
  - NBFC : décrit le nombre des fournisseurs cloud.

  Cette classe contient comme fonction : créationDC().

- La classe *VM* : permet de générer les machines virtuelles dans les machines physique (Host), elle contient comme principaux attributs :
  - Id_VM : décrit l'identifiant de la VM.
  - BW, MIPS, RAM, STORAGE…: les caractéristiques de la VM.

  Cette classe contient comme fonctions: get_caractéristique(), set_caractéristique().

- La classe *REQ_USER* : permet de générer les requêtes des utilisateurs, elle contient comme principaux attributs :
  - ID_USER : décrit l'identifiant de l'utilisateur.
  - ID_REQ : décrit l'identifiant de la requête d'utilisateur.
  - Requête : décrit la requête de l'utilisateur.
  - Budget : c'est la variable constante qui détermine le budget de l'utilisateur.

  Cette classe contient comme fonctions : initialiser_requête().

- La classe VAR : permet de modifier les variables de l'environnement cloud, elle contient comme attribut :
  - NBAC : décrit le nombre des agents courtiers.
  - NBUSERS : décrit le nombre des utilisateurs.

  Cette classe contient comme fonctions: get_var(), set_var().

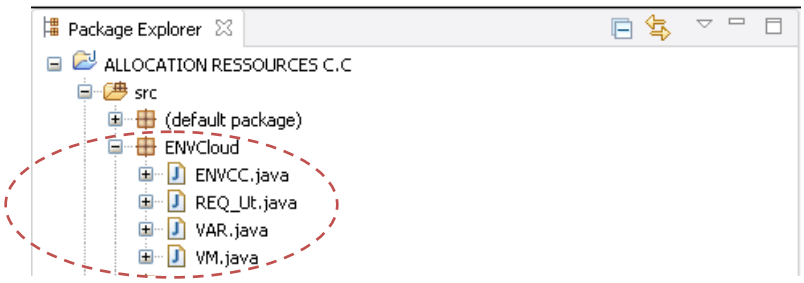


Figure 42. Simulation de l'environnement cloud computing

2éme partie : elle englobe les classes utilisées pour le développement du système multi-agents (**Figure 43**), nous citons : AUC, AFC, AC, ACo, AGDC.

- La classe *AUC* : permet de générer les agents utilisateurs cloud dans notre système, elle contient comme principaux attributs :
    - ID_AUC : décrit l'identifiant de l'agent utilisateur cloud.
    - ID_REQ : décrit l'identifiant de la requête d'utilisateur.
    - ALL_R : décrit l'allocation des ressources trouvée au profil de l'utilisateur.
    - R_AUC : décrit le registre des agents courtier détenu par l'agent utilisateur cloud.

  Cette classe contient comme fonctions : générer_requête(), sélectionner_AC(), envoyer_requête(), recevoir_ALL_R().

- La classe *AFC* : permet de générer les agents fournisseurs cloud dans notre système, elle contient comme principaux attribut :
    - ID_AFC : décrit l'identifiant de l'agent fournisseur cloud.
    - ID_REQ : décrit l'identifiant de la requête d'utilisateur reçu.
    - Requête : décrit la requête de l'utilisateur.
    - Proposition_ALL_R : décrit la proposition d'allocation des ressources en réponse de la requête reçue.

  Cette classe contient comme fonctions : recevoir_requête(), envoyerDEM_ACo(), RecevoirRep_ACo(), AjouterAttributs_CC().

- La classe *ACo* : permet de générer les agents coordinateurs cloud dans notre système, elle contient comme principaux attribut :
    - ID_ACo : décrit l'identifiant de l'agent coordinateur cloud.
    - ID_DEM : décrit l'identifiant de la demande de placement reçu.
    - Demande : décrit la demande de placement reçu.
    - REP_ACo : décrit la réponse de l'agent coordinateur sur la demande reçu.

  Cette classe contient comme fonctions : recevoir_demande(), envoyerDEM_AGDC(), RecevoirRep_AGDC(), Rechercher_Placement(), EnvoyerRep_AFC().

- La classe *AC* : permet de générer les agents courtier cloud dans notre système, elle contient comme principaux attribut :
    - ID_AC : décrit l'identifiant de l'agent courtier cloud.
    - ID_REQ : décrit l'identifiant de la requête d'utilisateur reçue.
    - Requête : décrit la requête de l'utilisateur.
    - ALL_R : décrit l'allocation des ressources trouvée au profil de l'AUC.
    - R_AC : décrit le registre des agents fournisseurs détenu par l'agent courtier cloud.

  Cette classe contient comme fonctions: Recevoir_requête(), Sélectionner_AFC(), RecevoirRep_AFC(), Evaluer_Rep_AFC().

- La classe *AGDC* : permet de générer les agents de gestion des Data Center dans notre système, elle contient comme principaux attribut :
    - ID_AGDC : décrit l'identifiant de l'agent de gestion de Data Center.

- ID_DEM : décrit l'identifiant de la demande reçue concernant l'état des ressources de data center.
- REP_DEM : décrit la réponse de la demande reçue.

Cette classe contient comme fonctions: Recevoir_DEM(), Editer_EtatDC(), EnvoyerRep().

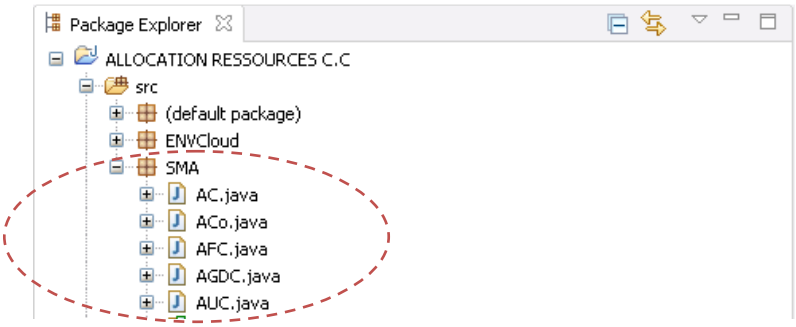


Figure 43. Le système multi-agents développé

5 Langage et environnement de développement

Nous avons réalisé ce travail sur une machine avec un processeur Intel(R) Pentieum(R) Core 2 Duo CPU T5300@ 1.73GHz, doté d'une capacité mémoire de 1 GB, sous Windows XP SP 2 de 32 bits. Nous avons utilisé l'environnement de développement Eclipse.

5.1 Le langage de programmation :- Java

Le langage Java est un langage de programmation orienté objet créé par James Gosling et Patrick Naughton employés de Sun Microsystems avec le soutien de Bill Joy (cofondateur de Sun Microsystems en 1982), présenté officiellement le 23 mai 1995 au SunWorld. Créé à ses débuts pour s'exécuter surtout sur environnements hétéroclites (embarqués principalement), son succès est surtout dû à son intégration aux navigateurs internet grand public **[48]**.

Java à la particularité principale d'être portable, c'est-à-dire, que les logiciels écrits avec ce dernier sont très facilement réutilisables sur plusieurs systèmes d'exploitation tels que Unix, Microsoft Windows, Mac OS, ou linux avec un peu ou pas de modification. C'est la plate-forme qui garantit la portabilité des applications développées en Java.

Le langage reprend en grande partie la syntaxe du langage C++, très utilisé par les informaticiens. Néanmoins, Java a été épuré des concepts du C++ et a la fois les plus déroutants, tels que l'héritage multiple remplacé par l'implémentation des interfaces. Les concepteurs ont privilégié l'approche orientée objet de sort qu'on Java, tout est objet à l'exception des type primitifs (nombres entiers, nombres à virgule flottante, ….).

Java possède plusieurs avantages dont **[49]**:

- Un langage interprété puisque la source est compilée en pseudo code ou Byte code puis exécuté par un interpréteur Java « Java Virtuel Machine (JVM)». Son principe est WORA (Write One Run Anywhere ; écrire une fois et exécuter partout). En effet le Byte code, s'il ne contient pas de code spécifique a une plate-forme particulière peut être

exécuté et obtenir quasiment les mêmes résultats sur toutes les machines disposant d'une JVM ;

- Un langage orienté objet dérivé du C, mais plus simple à utiliser et plus «pur» que le C++. On entend par «pur» le fait qu'en Java, on ne peut faire que de la programmation orientée objet contrairement au C++ qui reste un langage hybride, c'est-à-dire, autorisant plusieurs styles de programmation ;
- Simple, en effet le choix de ses auteurs a été d'abandonner des éléments mal compris ou mal exploités des autres langages tels que la notion de pointeurs (pour éviter les incidents en manipulant directement la mémoire), l'héritage multiple et la surcharge des opérateurs ;
- Doté, en standard, de bibliothèques de classes très riches comprenant la gestion des interfaces graphiques (fenêtres, boites de dialogue, contrôles, menus, graphisme), la programmation multithreads, la gestion des exceptions, les accès eu fichiers et au réseau, etc. ;
- L'utilisation des ces bibliothèques facilites grandement la tache du programmeur lors de la construction d'application complexes ;
- Doté, en standard, d'un mécanisme de gestion des erreurs (les exceptions) très utile et très performant. Ce mécanisme, inexistant en C, existe en C++ sous la forme d'une extension au langage beaucoup moins simple à utiliser qu'en Java ;
- Il est multi plates-formes : les programmes tournent sans modification sur tous les environnements ou Java existe (Windows, Unix et Mac).

Pour utiliser un programme Java, on installe un JRE, (Java Runtime Environment) qui correspond à l'environnement d'exécution. Il contient l'ensemble des outils nécessaires à l'exécution du code Java (JVM, bibliothèque standard, Just-In-Time Compiler, . . .). C'est un sous ensemble du JDK (*Java Development Kit*). Lors de l'exécution, une partie des classes est compilée en natif pour des questions de performance par le JIT (Just-In-Time Compiler). Le code compilé nativement est choisi automatiquement en fonction de sa fréquence d'utilisation. Pour les portions très utilisées, des optimisations sont aussi réalisées pour accélérer davantage l'exécution. La détection des « points chauds » se fait généralement par comptage du nombre de passages dans les méthodes, voire dans les têtes de boucle. Le JIT est présent dans tous les JRE depuis la version 1.2 de Java **[48]**.

5.2 Environnement de développement :- Eclipse

Eclipse **[52]** est un environnement de développement intégré extensible, universel et polyvalent, permettant potentiellement de créer des projets de développement mettant en œuvre n'importe quel langage de programmation. L'application est écrite en Java (à l'aide de la bibliothèque graphique SWT, d'IBM), et ce langage, grâce à des bibliothèques spécifiques, est également utilisé pour écrire des extensions.

Sa spécificité vient du fait de son architecture, toutes les fonctionnalités de cet atelier logiciel sont développées en tant que plug-in. Deux raisons principales ont amené à considérer Eclipse comme socle pour des applications clientes :

- La qualité d'Eclipse : l'environnement de développement Eclipse s'est imposé par sa fiabilité et la qualité de ses interfaces graphiques. L'utilisation massive de l'environnement de développement Eclipse par la communauté des développeurs Java ainsi que son utilisation comme base de produits commerciaux (WebSphere Studio, SAP NetWeaver Studio...) ont permis d'éprouver le framework Eclipse.
- La disponibilité en open source du framework Eclipse.

Dans notre travail nous avons utilisé la version 4.4 appelée Luna, sortie le 25 juin 2014.

5.3 Outil d'observation et de visualisation d'SMA :- Jade

JADE **[51]** propose deux outils graphiques pour déboguer les systèmes multi-agents réalisés: l'agent Dummy et l'agent Sniffer. Dummy est un outil pour inspecter les échanges de messages entre agents, et offre une interface graphique pour l'édition, l'écriture et l'envoi des messages ACL vers les agents, permet de recevoir et lire les messages des autres agents, et éventuellement de sauvegarder ces messages. Sniffer trace l'échange de messages et donne une interface graphique pour afficher les échanges de messages entre les différents groupes d'agents on utilisant une notation proche d'UML.

Dans notre travail nous avons utilisé la version JADE 4.3.2.

5.4 Outil de simulation:- Cloudsim

Cloudsim **[52]** l'un des principaux représentants des simulateurs de clouds. C'est un projet permet de simuler entièrement une infrastructure de type cloud en écrivant un programme utilisant les objets mis à disposition par l'interface de programmation fournie par la librairie. C'est un utilitaire principalement destiné à la recherche en matière de conception et d'évaluation de l'architecture sous-jacente des plateformes de services informatiques à la demande. Il permet d'obtenir un modèle du comportement du Data Center dans son ensemble. Par exemple, des travaux menés avec CloudSim par Beloglazov et Buyya ont permis d'analyser l'influence de différentes stratégies d'allocation de ressources sur la qualité de service rendue aux utilisateurs et la consommation électrique d'un centre de données.

Dans notre travail nous avons utilisé la version CloudSim 3.0.3.

6 Description de l'application

Nous avons crée une interface qui facilite a l'utilisateur l'accès et la manipulation de simulateur CloudSim, car le CloudSim n'offre pas une interface graphique qui permettre a l'utilisateur de profiter pleinement du simulateur (changer les paramètres de simulation, l'affichage graphique des résultats, visualiser le déroulement d'une simulation) [le CloudSim utilise le mode console]. Principalement une simulation se déroule en 3 étapes:

- ✓ Paramétrage de la simulation.
- ✓ Lancement de la simulation.
- ✓ Visualisation des résultats.

6.1 Interface principale

Le lancement de notre application débute par une interface qui permettre a l'utilisateur d'introduire ces propres paramètres, cette interface contient trois anglets principaux qui vont être détaillés dans la section suivante (6.2):

- ✓ Un anglet pour la configuration des paramètres globaux de la simulation.
- ✓ Un anglet pour la configuration de l'environnement de cloud computing.
- ✓ Un anglet pour la configuration des requêtes des utilisateurs.

Afin de visualiser et d'observer graphiquement le déroulement de processus d'allocation des ressources a base d'agents, l'application charge automatiquement la plateforme JADE (**Figure 44**) dès a sa lancement.

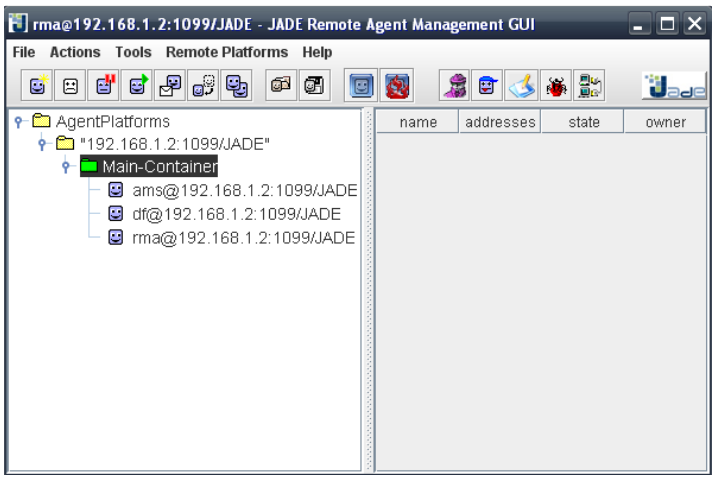


Figure 44. Chargement de JADE au lancement de l'application

6.2 Paramétrage de la simulation

Le paramétrage de la simulation est introduit par l'utilisateur, il contient trois parties :

- ➢ Paramètres globaux: Cette partie (**Figure 45.a**) permet d'initialiser le nombre des fournisseurs cloud, le nombre des utilisateurs, le nombre des agents courtiers dans la couche de courtage, ainsi que l'initialisation des répertoires des AUCs et ACs.

- ➢ Cloud Computing: Cette partie (**Figure 45.b**) permet d'introduire les paramètres propres au cloud computing à savoir: nombre de Data Center, d'hôtes, PE (Processing Element) ou bien CPU/Core, la puissance de chaque PE, la taille de la mémoire, la taille de stockage, la bande passante, ainsi que les coûts de traitement, de la mémoire, de stockage, et de bande passante.

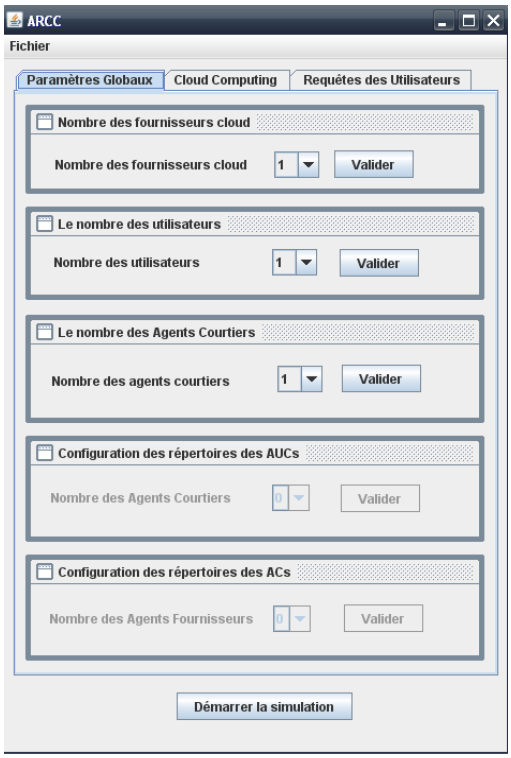


Figure 45.a. Paramètres globaux

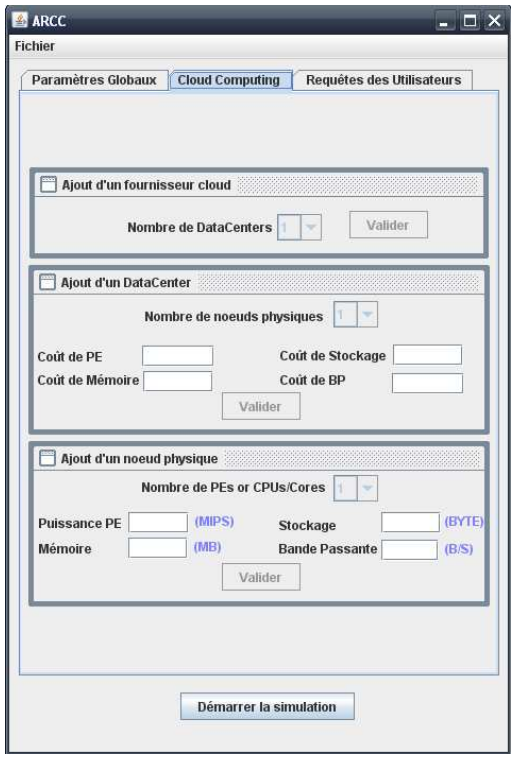


Figure 465.b. Paramètres du C.C

- Les requêtes des utilisateurs: Cette partie (**Figure 45.c**) détermine les requêtes des utilisateurs à savoir : nombre des services ou applications, ces besoins en ressources, ainsi que le budget de chaque utilisateur.

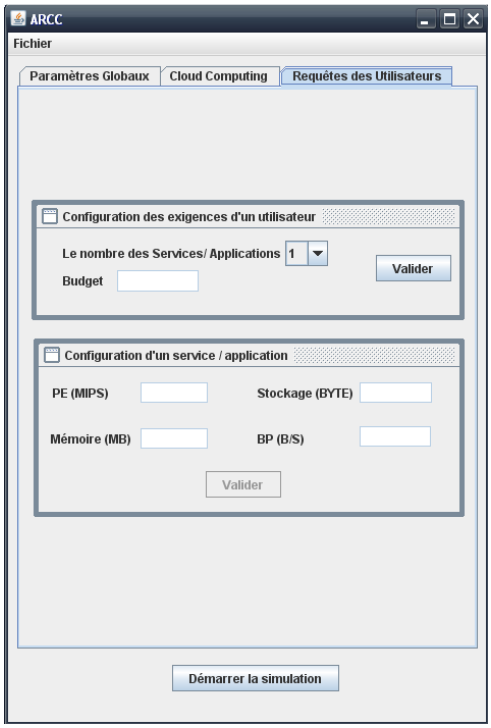


Figure 475.c. Les requêtes des utilisateurs

6.3 Lancement de la simulation

Une fois les paramètres de la simulation introduite, l'utilisateur peut démarrer la simulation en cliquant sur le bouton *Démarrer la simulation* (**Figure 46**).

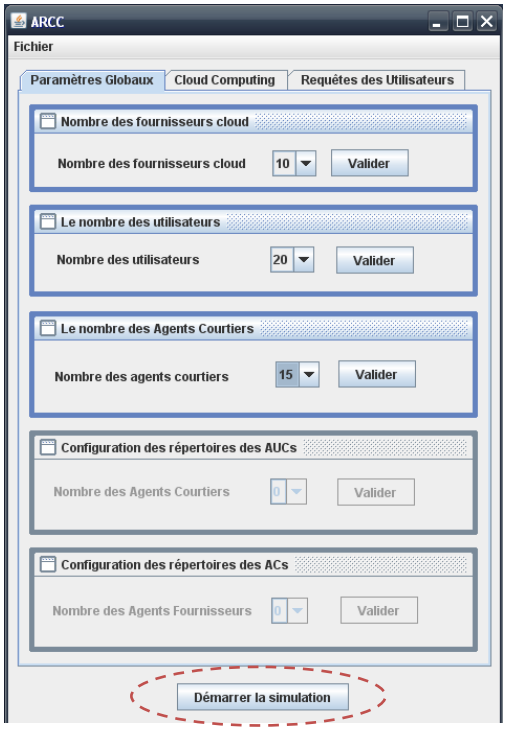


Figure 486. Interface de lancement de la simulation

6.4 Déroulement et Résultats

Nous allons présenter dans cette partie le déroulement d'une simulation de l'approche d'allocation des ressources à base d'agents dans le cloud computing que nous avons proposée, ensuite nous allons présenter quelques résultats obtenus en appliquant notre approche.

Afin de démontrer le déroulement d'une simulation, nous avons pris l'exemple suivant:

- Les paramètres globaux: nombre des fournisseurs cloud à trois, nombre des utilisateurs à deux, nombre des courtiers à quatre, et nous avons initialisé les répertoires des agents utilisateurs cloud (AUCs) et les répertoires des agents courtiers (ACs) comme suite :

- Les répertoires des agents utilisateurs cloud (AUCs) :

L'ID de l'AUC	1	2	3	4	5
Nombre des ACs	3	2	4	1	4

Table 5. Exemple d'initialisation des répertoires des agents utilisateurs cloud

- Les répertoires des agents courtier (ACs) :

L'ID de l'AC	1	2	3	4
Nombre des AFCs	3	1	2	3

Table 6. Exemple d'initialisation des répertoires des agents utilisateurs cloud

- Les paramètres de l'environnement cloud computing: soit la configuration suivante :

		Nombre des nœuds physiques	Capacités des nœuds physiques					Coût du processing element (U)	Coût de la mémoire (U)	Coût de stockage (U)	Coût de la Bande Passante (U)
			Nombre des PE	Puissance de chaque PE (MIPS)	Mémoire (MB)	Stockage (GO)	Bande Passante (MB/S)				
Fournisseur Cloud 1	**Data Center 1**	2	2	1000	2	500	128	3.0	0.5	0.1	0.01
	Data Center 2	1	4	400	1	250	256	4.0	1.2	0.3	0.01
	Data Center 3	4	3	2000	4	750	1024	1.5	0.8	0.5	0.03
Fournisseur Cloud 2	**Data Center 1**	1	5	5000	8	750	512	2.0	0.5	0.4	0.05
Fournisseur Cloud 3	**Data Center 1**	4	4	2500	4	750	512	2.5	0.7	0.1	0.06
	Data Center 2	3	2	3200	4	500	2048	1.0	1.1	0.8	0.02

Table 7. Exemple de paramétrage de l'environnement cloud computing

NB: Dans cet exemple en suppose que les nœuds physiques de la même Data Center sont homogènes.

- Les requêtes des utilisateurs: soit R = (PE en MIPS, Mémoire en MB, stockage en GO et Bande passante en MB/S) la description de ressource utilisée pour décrire les requêtes des utilisateurs. Et soit les requêtes des utilisateurs suivantes:

- Requête utilisateur 1 = {R1 (970, 2, 150, 32); R2 (520, 1, 50, 64); R3 (390, 5, 75, 16)} avec un budget = 2000U.
- Requête utilisateur 2 = {R (450, 4, 200, 128)} avec un budget = 500U.

Une fois les paramètres de simulation introduite, le mode console affiche l'initialisation de l'application (**Figure 47**) :

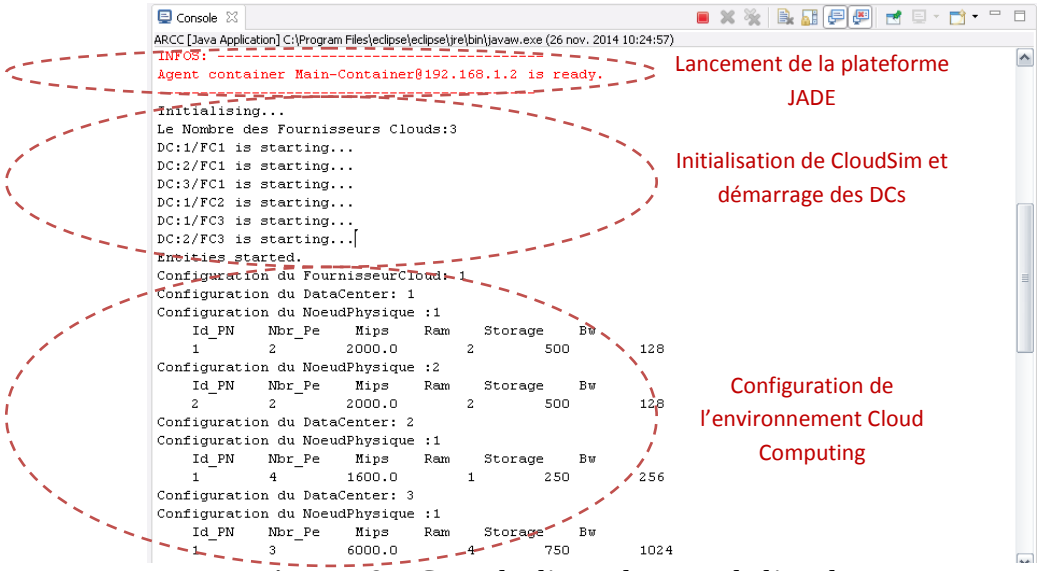


Figure 497. Console d'initialisation de l'application

Grâce à la plate-forme JADE, l'utilisateur de l'application peut suivre tous les agents de notre système et cela en utilisant l'agent *Sniffer* qui peu être lancé automatiquement avec la lancement de l'application.

Chaque message partant ou allant vers un agent est capté et affiché sur l'interface du Sniffer. Nous allons illustrer par la **Figure 48** le déroulement d'allocation des ressources afin de satisfaire la requête de *deuxième utilisateur cloud*.

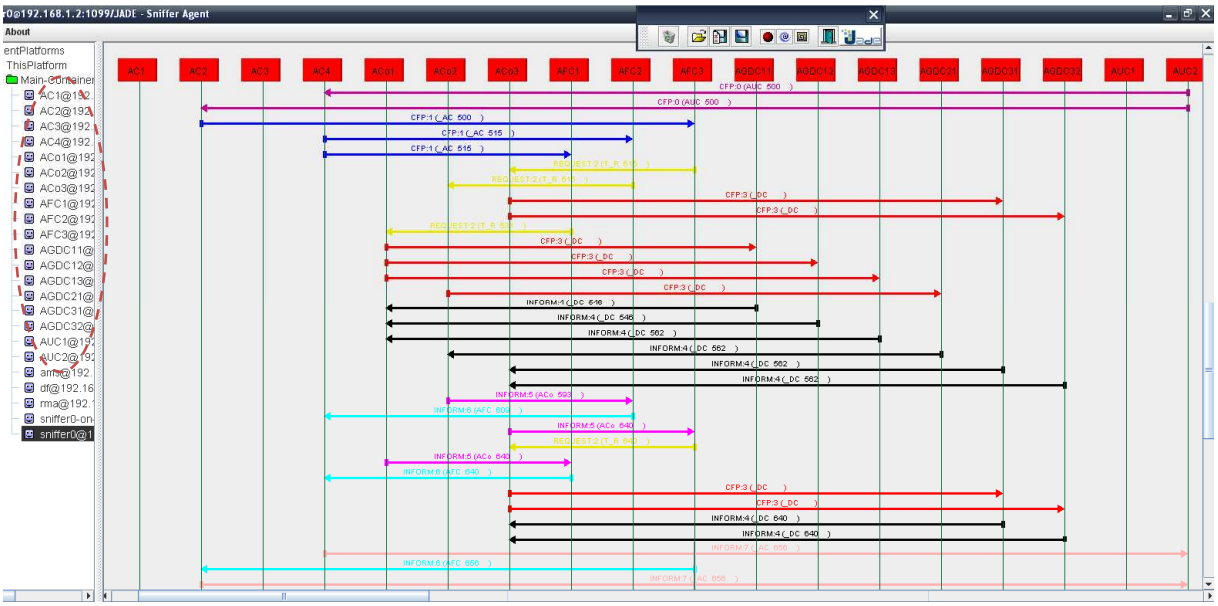


Figure 48. Résultats sur le Sniffer.

Pour suivre de plus prés le déroulement de l'allocation des ressources (la coopération entre les agents, le comportement de chaque agent…), nous avons affiché dans un journal les traces de coopération qui se produisent lors de l'exécution de chaque agent (voir les illustrations des **Figures 49 a, 49.b, 49.c**).

Nous allons illustrer par la **Figure 49.a** le journal des traces de coopération [diffusion de la demande de l'utilisateur, réception des propositions d'allocation …] de l'agent utilisateur (AUC2) avec les agents courtiers (AC2 et AC4).

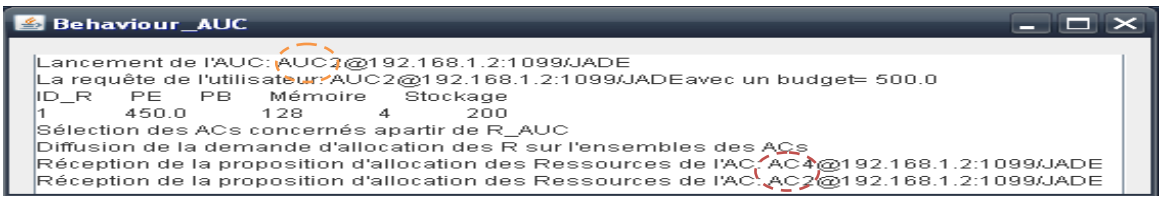


Figure 49.a. Le journal des agents utilisateurs cloud.

Nous allons illustrer par la **Figure 49.b** le journal des traces de coopération des agents courtiers; D'une part, l'agent courtier AC4 avec les agents fournisseurs cloud (AFC1 et AFC2) [diffusion d'une demande d'allocation, réception des propositions d'allocation…], et d'autre part avec l'agent utilisateur cloud (AUC2) [envoi de la meilleure allocation des ressources trouvée…] (la même chose pour l'agent courtier AC2 avec l'AFC3 et l'AUC2).

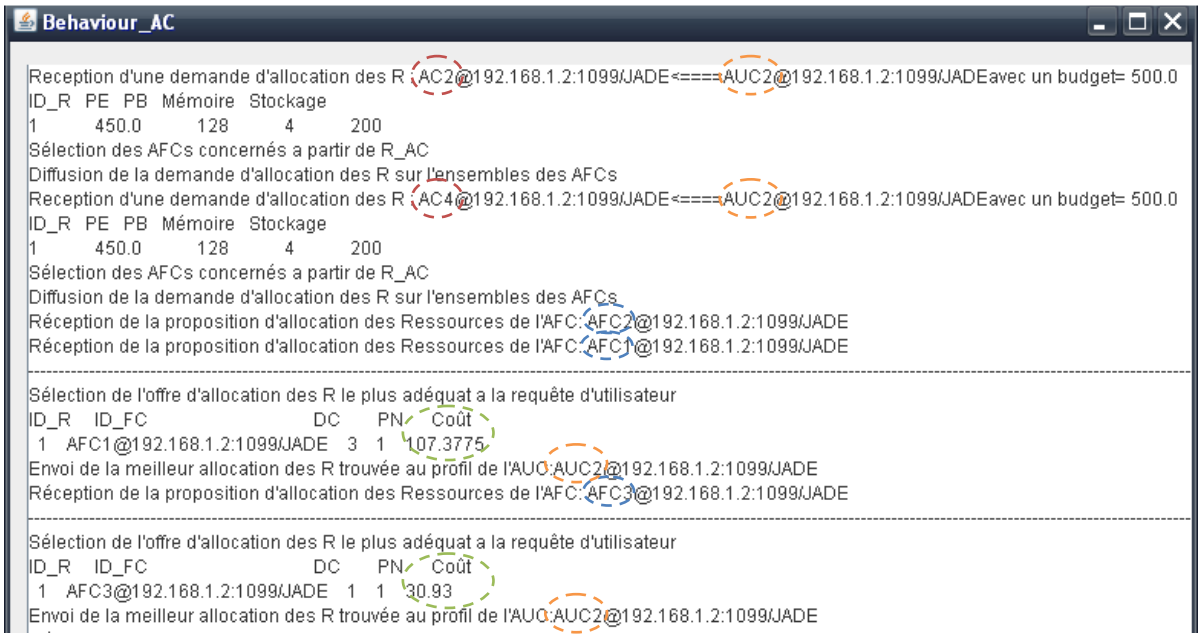


Figure 49.b. Le journal des agents courtiers.

Nous allons illustrer par la **Figure 49.c** le journal des traces de coopération de l'agent fournisseur cloud (AFC3) avec l'agent courtier (AC2), et les agents fournisseurs cloud (AFC1, AFC2) avec l'agent courtier (AC4).

Figure 49.c. Le journal des agents fournisseurs cloud.

La **figure 50** illustre les propositions d'allocation des ressources des agents courtiers (AC2, AC4) au profil de l'agent utilisateur cloud (AUC2), afin de satisfait la demande de deuxième utilisateur cloud.

- L'agent courtier AC4 propose une allocation d'une machine virtuelle héberger sur le nœud physique 1 du Data Center 3, détenue par le fournisseur cloud 1 avec un coût = **107.3775U**.
- L'agent courtier AC2 propose une allocation d'une machine virtuelle héberger sur le nœud physique 1 du Data Center 1, détenue par le fournisseur cloud 3 avec un coût = **30.93U.**

Figure 50. Les propositions d'allocation des ressources des agents courtiers.

7 Conclusion

Dans ce chapitre, nous avons essayé de mettre en œuvre l'ensemble des idées qui caractérise l'approche proposée en se concentrant sur l'implémentation de système multi-agents ainsi que l'intégration entre différents agents.

Notre architecture est bien implémentée en utilisant un environnement de développement JAVA qui comporte le simulateur CloudSim pour simuler l'environnement cloud computing et la plateforme Jade qui permet de visualiser l'interaction entre l'ensemble des agents implémentés.

Les résultats obtenus à partir de notre simulation nous confirment que l'utilisation d'un système multi-agents pour l'allocation des ressources dans le cloud computing assure La qualité du service (QoS), en assurant la satisfaction de la clientèle qui consiste à diminuer le cout des ressources commercialisées afin d'essayer d'utiliser le maximum de service a un cout minimal.

Conclusion générale et perspectives

Les récents efforts visant a 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 une approche d'allocation des ressources dans le cloud computing basée agent. 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 adoptant le paradigme d'agent, nous avons modélisé le processus d'allocation des ressources dans le cloud computing par un système multi-agent, comporte cinq classe d'agent: un agent utilisateur cloud qui représente l'utilisateur dans l'environnement cloud, un agent fournisseur cloud qui représente le fournisseur dans l'environnement cloud. Afin de trouver l'offre le plus approprié a la demande d'utilisateur parmi les offres des fournisseurs, un agent courtier joue le rôle de courtage entre les utilisateurs et les fournisseurs. Afin de préparer son offre d'allocation des ressources chaque agent fournisseur cloud consulte l'agent coordinateur pour chercher le placement des machines virtuelles faisable. Pour accomplir la tâche de coordination, l'agent coordinateur consulte les agents de gestion des Data Centers (responsables des ressources physiques, dans les différentes Data Centers).

La réalisation de cette étude nous a conduites à suivre les étapes suivantes :

- En premier lieu, Nous avons défini le cloud et ses propriétés essentielles, nous avons détaillé le paradigme utilisé (paradigme d'agent) pour développé l'approche proposée.
- Après, nous avons présenté un état de l'art sur l'allocation des ressources dans le cloud et une revue sur les travaux existants.
- Ensuite, nous avons proposé notre approche d'allocation des ressources à base d'agents (conception globale et conception détaillée ainsi que les différentes interactions entre les agents).
- En fin, dans la quatrième partie, nous avons validé notre approche par le développement d'une application, dans la quelle nous avons utilisé le simulateur Cloudsim pour simuler l'environnement cloud computing, et la plateforme Jade pour visualiser l'interaction entre les agents, avant de terminer ce chapitre par la présentation des résultats obtenus.

Ce mémoire constitue une base de travail à partir du quelle, de nouvelles activités de recherche peuvent être lancées afin d'améliorer le travail présenté. Les perspectives que nous proposons peuvent donc s'orienter vers les directions suivantes:

- Enrichir la couche d'interaction cloud-utilisateur (Accès d'utilisateur, prétraitement de la demande d'utilisateur …).

- Trouver un mécanisme pour le développement des répertoires des agents utilisateurs cloud ainsi que les répertoires des agents courtiers.
- Doter les agents courtiers par des stratégies de négociations, des techniques d'établissement d'SLA…
- Développer une méthode d'optimisation pour le placement des machines virtuelles sur les machines physiques (l'une des principales tâches de l'agent coordinateur).
- Proposition des méthodes d'optimisation pour: minimiser le nombre des nœuds physiques, réduire dans la consommation d'énergie etc. (au niveau de la couche gestion des Data Centers).
- Prendre en compte l'aspect d'adaptabilité des agents.

Bibliographie

[1]: M-M. Willy, “Vers une architecture pair-a-pair pour l’informatique dans le nuage”, Thèse de doctorat en informatique, Université de Grenoble, 2011.

[2]: B. Nabil et T. Abdellah, “Cloud-based Web Application Firewall”, Mémoire de master en informatique, Université Saad Dahleb, Algérie, 2013

[3]: P. Mell et T. Grance, “The NIST Definition of Cloud Computing”, National Institute of Standards and Technology -NIST- Special Publication, 2011.

[4]: R. Buyya, J. Broberg et M. Andrzej, “Cloud Computing: Principles and Paradigms”. Hoboken, New Jersey: John Wiley & Sons, 2011

[5]: A. Gadafi, “Gestion mémoire dans une infrastructure répartie”, Thèse de doctorat en informatique, Université de Toulouse, 2013

[6]: M. Lucas Nussbaum, “Cloud Computing”, Licence Professionnelle : Administration de systèmes, réseaux et applications à base de logiciels libres, IUT Nancy Charlemagne, 2010.

[7]: N. Degroodt, “L'élasticité des bases de données sur le cloud computing”, Mémoire de master en sciences informatiques, Université libre de Bruxelles, Université d’Europe, 2011.

[8]: E. Daubert, “Adaptation et Cloud Computing : un besoin d’abstraction pour une gestion transverse”, Thèse de doctorat en informatique, Institut National des Sciences Appliquées de Rennes, 2013.

[9]: A-B. Tchana, “Système d'Administration Autonome Adaptable: application au Cloud”, thèse de doctorat en informatique, Université de Toulouse, 2011

[10]: O. Leclère, “Mise en place d’un site type Web 2.0 sur un Cloud ”, Mémoire d’ingénieur CNAM en informatique, Centre régional associe de Saint Genis Pouilly, 2010.

[11]: S. François, “La Virtualisation”, Projet de recherche et communication scientifique, université libre de Bruxelles, Université d’Europe, 2010.

[12]: R. Pottier, “Un langage dédié a l'administration d'infrastructures virtualisées”, thèse de doctorat en informatique, Université Nantes Angers le mans, 2012.

[13]: J. Tranier, “Vers une vision intégrale des systèmes multi-agents, Contribution à l’intégration des concepts d’agent, d’environnement, d’organisation et d’institution”, Thèse de doctorat en informatique, Université Montpellier II, 2007.

[14]: J. Ferber, “Les systèmes multi-agents : vers une intelligence collective”, Informatique, Intelligence Artificielle. Inter-Editions, 1995.

[15]: A. Selma, “Approche dirigée par les modèles pour le développement de systèmes multi-agents”, Thèse de doctorat en informatique, université de Savoie, 2007

[16]: L. Yoann, “Architecture multi-agent pour la gestion d’objets tangibles et virtuels sur Tables Interactives ”, thèse de doctorat en informatique, Université de valenciennes et du Hainaut-Cambrésis, 2012.

[17]: M. Hanaa, “Une Approche Multi-agents à Architecture P2P pour l’Apprentissage Collaboratif », Thèse de doctorat en informatique, Université du littoral côte d’opale, 2013.

[18]: H. Mikael, “algorithmes de gestion de ressources dans une infrastructure de virtualisation de services de jeux vidéo”, Mémoire, école de technologie supérieure université du Québec, 2014.

[19]: D. Talia, “Clouds Meet Agents: Toward Intelligent Cloud Services”, IEEE computer society, IEEE internet computing, 2012.

[20]: G. Lee, “Resource Allocation and Scheduling in Heterogeneous Cloud Environments”, Electrical Engineering and Computer Sciences University of California at Berkeley, Technical Report No. UCB/EECS-2012-78, May 2012.

[21]: N. Krishnaveni ET G. Sivakumar, “Survey on Dynamic Resource Allocation Strategy in Cloud Computing Environment”, International Journal of Computer Applications Technology and Research (IJCAT), 2013.

[22]: N. Asha ET G. Raghavendra Rao, “A Review on Various Resource Allocation Strategies in Cloud Computing”, International Journal of Emerging Technology and Advanced Engineering (IJETAE), July 2013.

[23]: P. Ronak ET P. Sanjay, “Survey on Resource Allocation Strategies in Cloud Computing”, International Journal of Engineering Research & Technology (IJERT), February 2013.

[24]: V. Vinothina, S. Lecturer ET al, “A Survey on Resource Allocation Strategies in Cloud Computing”, International Journal of Advanced Computer Science and Applications (IJACSA) 2012.

[25]: V. Vivek, R. Srinivasan ET E. Blessing Rajsingh, “Resource Provisioning Methodologies: An Approach of Producer and Consumer Favorable In Cloud Environment”, International Conference on Modern Trends in Science, Engineering and Technology (ICMTSET) 2013.

[26]: P. Takako Endo, A. Vitor ET al, “Resource Allocation for Distributed Cloud: Concepts and Research Challenges”, IEEE Network 2011.

[27]: P. Sareen, “Cloud Computing: Types, Architecture, Applications, Concerns, Virtualization and Role of IT Governance in Cloud”, International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE) 2013.

[28]: G. Estácio, P. Takako Endo ET al, “Resource Allocation in Clouds: Concepts, Tools and Research Challenges”, 29° Symposium Brésilien sur les réseaux informatiques et les systèmes

distribués -XXIX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos- (SBRC) 2011.

[29]: H. Hussain ET al, "A survey on resource allocation in high performance distributed computing systems", Parallel Computing 2013.

[30]: C. Xiao Jun ET al, "Resource virtualization methodology for on-demand allocation in cloud computing systems", Springer-Verlag London 2011.

[31]: S. Jiajia ET al, "A Cloud Resource Allocation Scheme Based on Microeconomics and Wind Driven Optimization", 8th ChinaGrid Annual Conference (ChinaGrid) 2013.

[32]: Y. Xindong ET al, "ARAS-M: Automatic Resource Allocation Strategy based on Market Mechanism in Cloud Computing", Journal of computers 2011.

[33]: J. Gihun ET S. Kwang Mong, "Agent-based Adaptive Resource Allocation on the Cloud Computing Environment", International Conference on Parallel Processing Workshops 2011.

[34]: K. Clark ET al, "An intelligent cloud resource allocation service -Agent based automated Cloud resource allocation using micro-agreements", The proceedings of the 2nd International Conference on Cloud Computing and Services Science 2012.

[35]: Seokho Son ET al, "An SLA-based cloud computing that facilitates resource allocation in the distributed data centers of a cloud provider", Springer Science+Business Media New York 2013.

[36]: K. Arafat, R. Sagbo ET P. Houngue, "Quality Architecture for Resource Allocation in Cloud Computing", Service-Oriented and Cloud Computing, First European Conference (ESOCC) 2012.

[37]: X. Xin ET Y. Huiqun, "A Game Theory Approach to Fair and Efficient Resource Allocation in Cloud Computing", Mathematical Problems in Engineering 2014.

[38]: W. Yuan ET al, "Research on Optimization of Resources Allocation in Cloud Computing Based on Structure Supportiveness", Springer Science+Business Media Dordrecht 2014.

[39]: F. Xie ET al, "A Resource Allocation Strategy Based on Particle Swarm Algorithm in Cloud Computing Environment", Fourth International Conference on Digital Manufacturing and Automation (ICDMA) 2013.

[40]: F. Zongqin ET al, "Simulated-Annealing Load Balancing for Resource Allocation in Cloud Environments", International Conference on Parallel and Distributed Computing, Applications and Technologies 2013.

[41]: Y. Zhengqiu ET al, "study on cloud resource allocation strategy based on particle swarm ant colony optimization algorithm", proceedings of IEEE (CCIS) 2012.

[42]: R. Buyya, R. Ranjan and R. N. Calheiros: “InterCloud: Utility-Oriented Federation of Cloud Computing Environments for Scaling of Application Services”, Algorithms and Architectures for Parallel Processing -10th International Conference-(ICA3PP) 2010.

[43]: X. Wang, X. Liu et al., “A Decentralized Virtual Machine Migration Approach of Data Centers for Cloud Computing”, HindawiPublishing Corporation Mathematical Problems in Engineering Volume 2013.

[44]: V. Krishna Reddy, “Meta-heuristic based multi objective task scheduling for managing the cost in cloud computing environment”, thèse de doctorat – chapitre 2-, AcharyaNagarjuna University, 2012.

[45]: M. Wooldridge, “Multiagent systems: A modern approach to distributed artificial intelligence”, Éditer par G. Weiss, pp. 27-78, MIT Press, London, England, 1999.

[46]: J. Bryson, L.A. Stein, “Modularity and specialized learning in the organization of behavior”. Proceedings of the Sixth Neural Computation and Psychology Workshop, Liège, Belgium, 2000.

[47]: A. ALI, “ L’approche multi-agents pour le pilotage des systèmes complexes, Appliquée aux Systèmes du Trafic Urbain”. Thèse de doctorat en informatique, UNIVERSITE BLAISE PASCAL – CLERMONT II, 2009.

[48]: L. Christophe, ‘ De la nécessité d’une vision holistique du code pour l’analyse statique et la correction automatique des Applications Web ’, thèse de doctorat, UNIVERSITÉ DE RENNES 1, 2011.

[49]: B. Samah, ‘ Gestion des ressources dans le Cloud Computing à base des modèles économique ’, Mémoire de magister, Université d'Oran, Faculté des sciences, 2011.

[50]: J. Jonckers, ‘Réalisation d'une application eID (la carte d'identité électronique)’, Université de Mons-Hainaut, 2007.

[51]: H. Joumaa, ‘Analyse des performances d'un système multi-agents par visualisation’, thèse de doctorat, Université de Grenoble, 2010.

[52]: L. Sylvain, ‘Services de répartition de charge pour le Cloud : Application au traitement de données multimédia’, École Doctorale Informatique, Télécommunications et Électronique (Paris), thèse de doctorat, 2013.