الجمهورية الجزائرية الديمقراطية الشعبية

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

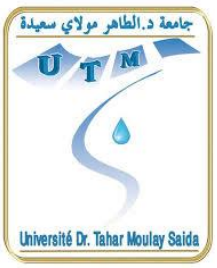
وزارة التعليم العالي و البحث العلمي

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

Université Dr. Tahar Moulay SAIDA — **جامعة د. الطاهر مولاي سعيدة**

Faculté : Technologie — **كلية : التكنولوجيا**

Département : Informatique — **قسم : الإعلام الآلي**



MEMOIRE DE MASTER

Option : RISR

THEME

Une stratégie de réplication dynamique de données dans les Cloud Computing

Présenté par :

-MEGLALI Oussama Djamel Eddine

-HADDI Mohamed Reda

Encadré par :

-LIMAM Saïd

2015 - 2016

Remerciements

En préambule à ce mémoire, on souhaite adresser nos remerciements les plus sincères aux personnes qui nous ont apporté leur aide et qui ont contribué à l'élaboration de ce mémoire.

On tient à remercier sincèrement Monsieur LIMAM Saïd, qui en tant qu'encadreur de mémoire, s'est toujours montré à l'écoute et très disponible tout au long de la réalisation de ce mémoire, ainsi pour l'inspiration, l'aide et le temps qu'il a bien voulu nous consacrer et sans qui ce mémoire n'aurait jamais vu le jour.

On n'oublie pas nos parents pour leur contribution, leur soutien et leur patience.

Enfin, on adresse nos plus sincères remerciements à tous nos proches et amis, qui nous ont toujours soutenu et encouragé au cours de la réalisation de ce mémoire.

On remercie enfin tous ceux qui n'ont pas été cités dans ces quelques lignes et qui ont contribué de près ou de loin par leur aide au bon déroulement de ce travail.

Merci à tous et à toutes.

Dédicace Reda Haddi

Je dédie ce travail à :

Ma très chère mère qui a toujours et qui a été toujours présente à mes côtés, dans les moments les plus difficiles que le bon Dieu me la garde.

A la mémoire de mon père, qu'il repose en paix.

A mes très chers frères Zakaria, Abdelillah, Yacine, Amine, Oussama, Seddik et Samir
Mes très chères sœurs.

A tous mes Oncles, Mes tantes.

A mes meilleur amis : Habbaz, Alaa, Idrici, Zoheir, Abdelkrim et Bourass

A tout mes amies qui mon beaucoup aidé même si nous avons oublié le nom de quelqu'un soyez sur que vous êtes dans notre cœur et sans oublier mes collègues de la promotion 2016.

Dédicace Oussama Meglali

Je dédie ce travail à :

A mes chers parents, je vous dédie ce modeste travail qui est le fruit de vos interminables conseils, assistance et soutient moral, en témoignage de ma reconnaissance et mon affection, dans l'espoir que vous en serez fiers.

A mes très chers frères Abdelmoumen et Abdelrahmane.

A tous mes Oncles, Mes tantes.

A tout mes amis : Abbes, Sidahmed, Mohamed, Kamel, Samir, Habbaz, Alaa, Idrici, Zoheir, Abdelkrim et Bourass.

A tout mes amies qui mon beaucoup aidé même si nous avons oublié le nom de quelqu'un soyez sur que vous êtes dans notre cœur et sans oublier mes collègues de la promotion 2016.

Table des matières

Introduction générale

L'informatique dans les nuages (Cloud Computing en anglais) s'est imposée ces dernières années comme un paradigme majeur d'utilisation des infrastructures informatiques. Celui-ci répond à des besoins et demandes croissantes en terme de disponibilité et flexibilité. Le développement remarquable du Cloud Computing, ces dernières années, suscite de plus en plus l'intérêt des différents utilisateurs de l'Internet et de l'informatique qui cherchent à profiter au mieux des services et des applications disponibles en ligne à travers le Web en mode services à la demande et facturation à l'usage.

L'approche du Cloud Computing s'appuie principalement sur le concept de virtualisation. Ce concept est un ensemble de techniques permettant de faire fonctionner sur une seule machine plusieurs systèmes d'exploitation et/ou plusieurs applications, isolés les uns des autres. Un Cloud est constitué d'un ensemble de machines virtuelles qui utilisent la même infrastructure physique.

La fiabilité d'une Cloud computing est assurée par la disponibilité et l'accessibilité des données par rapport à l'utilisateur. Comme les Nœuds peuvent tomber en panne et les données se déplacent à travers le réseau, la fiabilité peut être diminuée. Les demandes cumulées sur une machine virtuelle ou bien sur un data center sur une donnée ne peuvent pas répondre aux besoins des utilisateurs au même temps.

Dans ce travail, l'objectif visé est de proposer une stratégie basée sur la création des répliques, en appliquant des méthodes et des algorithmes. Notre politique de création vise à optimiser le temps de réponse des Cloudlets, garantir et améliorer un certain degré de disponibilité pour les données du système.

Organisation du mémoire : Le présent mémoire est structuré autour de quatre principaux chapitres qui se résument comme suit :

Chapitre 1 : Dans le premier chapitre, nous présenterons une entrée sur les systèmes à grande échelles.

Chapitre 2 : Le second chapitre présentera quelques différentes techniques de données tel que la réplication de donnée et la gestion de cohérence dans les systèmes distribué.

Chapitre 3 : Le troisième chapitre sera réservé à la description détaillée de la conception de la stratégie utilisée que nous avons proposé.

Chapitre 4 : Ce dernier chapitre présentera les étapes de l'implémentation de l'approche proposée. Nous y détaillerons la réalisation de certaines fonctionnalités ainsi que l'étude d'évaluation de cette stratégie. Les résultats d'expérimentation seront interprétés. Enfin, Une synthèse et un ensemble de perspectives viendront pour clôturer notre travail.

CHAPITRE I :

Les systèmes distribués à large échelle

Chapitre 1

Les systèmes distribués à large échelle

1.1 Introduction

L'informatique du début du 21 ème siècle, qu'elle soit visible (système d'information d'une entreprise) ou enfoui dans un processus industriel (voiture, avion, etc.) est répartie "par nature". Les systèmes informatiques de nos jours sont par essence distribués. Ils sont souvent composés de "sites" (processeurs, capteurs, ordinateurs, émetteurs, etc.) reliés en réseaux [1]. Ces sites sont caractérisés par une :
- distribution géographique étendue.
- hétérogénéité et mobilité des composants (PC, PDA, téléphones, capteurs, etc.).
- volatilité et une disponibilité partielle.

La gestion des données dans un environnement à grande échelle est indispensable pour prendre en compte les besoins des nouvelles applications. Si la gestion des données dans les systèmes distribués a été largement étudiée ces dernières années, des solutions efficaces et à bas coût tardent à voir le jour à cause de la complexité des problèmes introduits par la largeur de l'échelle et le caractère hétérogène et dynamique de l'environnement.

1.2 Caractéristiques des systèmes distribué à large échelle

Un système réparti doit assurer plusieurs propriétés pour être considéré comme performant : la transparence, le passage à l'échelle, la disponibilité et l'autonomie [2].

1.2.1 Transparence

La transparence permet de cacher aux utilisateurs les détails techniques et organisationnels d'un système distribué et complexe. L'objectif est de pouvoir faire bénéficier aux applications une multitude de services sans avoir besoin de connaître exactement la localisation ou les détails techniques des ressources qui les fournissent. Ceci rend plus simple, le développement des applications mais aussi leur maintenance évolutive ou corrective. Selon la norme (ISO, 1995) la transparence a plusieurs niveaux :

1. Accès : cacher l'organisation logique des ressources et les moyens d'accès à une ressource.

2. Localisation : l'emplacement d'une ressource du système n'a pas à être connu.

3. Migration : une ressource peut changer d'emplacement sans que cela ne soit aperçu.

4. Réplication : les ressources sont dupliquées mais les utilisateurs n'ont aucune connaissance de cela.

5. Panne : si un nœud est en panne, l'utilisateur ne doit pas s'en rendre compte et encore moins de sa reprise après panne.

6. Concurrence : rendre invisible le fait qu'une ressource peut être partagée ou sollicitée simultanément par plusieurs utilisateurs. Le parcours de cette liste montre qu'il n'est pas évident d'assurer une transparence totale. En effet, masquer complètement les pannes des ressources est quasi impossible aussi bien d'un point de vue théorique que pratique. Ceci est d'autant plus vrai qu'il n'est pas trivial de dissocier une machine lente ou surchargée de celle qui est en panne.

1.2.2 Passage à l'échelle

Le concept de passage à l'échelle désigne la capacité d'un système à continuer à délivrer avec un temps de réponse constant un service même si le nombre de clients ou de données augmente de manière importante. Le passage à l'échelle peut être mesuré avec au moins trois dimensions :

1- Le nombre d'utilisateurs et/ou de processus (passage à l'échelle en taille) ;

2- La distance maximale physique qui sépare les nœuds ou ressources du système (passage à l'échelle géographique) ;

3- Le nombre de domaines administratifs (passage à l'échelle administrative).

1.2.3 Disponibilité

Un système est dit disponible s'il est en mesure de délivrer correctement le ou les services de manière conforme à sa spécification. Pour rendre un système disponible, il faut donc le rendre capable de faire face à tout obstacle qui peut compromettre son bon fonctionnement. En effet, l'indisponibilité d'un système peut être causée par plusieurs sources parmi lesquelles nous citons :

- Les pannes qui sont des conditions ou événements accidentels empêchant le système, ou un de ses composants, de fonctionner de manière conforme à sa spécification ;
- Les surcharges qui sont des sollicitations excessives d'une ressource du système entraînant sa congestion et la dégradation des performances du système ;
- Les attaques de sécurité qui sont des tentatives délibérées pour perturber le fonctionnement du système, engendrant des pertes de données et de cohérences ou l'arrêt du système.

Pour faire face aux pannes, deux solutions sont généralement utilisées :

1. La première consiste à détecter la panne et à la résoudre, et ce dans un délai très court. La détection des pannes nécessite des mécanismes de surveillance qui s'appuient en général sur des timeouts ou des envois de messages périodiques entre ressources surveillées et ressources surveillantes.

2. La deuxième solution consiste à masquer les pannes en introduisant de la réplication. Ainsi, quand une ressource est en panne, le traitement qu'elle effectuait est déplacé sur une autre ressource disponible. La réplication peut être aussi utilisée pour faire face à la seconde cause d'indisponibilité d'un système (surcharge du système). Pour réduire la surcharge d'une ressource, les tâches sont traitées parallèlement sur plusieurs répliques ou sur les différentes répliques disponibles à tour de rôle. Une autre technique qui permet de réduire la surcharge d'une ressource consiste à distribuer les services et/ou les données sur plusieurs sites et donc de les solliciter de manière parallèle.

1.2.4 Autonomie

Un système ou un composant est dit autonome si son fonctionnement ou son intégration dans un système existant ne nécessite aucune modification des composants du système hôte. L'autonomie des composants d'un système favorise l'adaptabilité, l'extensibilité et la réutilisation des ressources de ce système. Par exemple, une ressource autonome peut être remplacée avec une autre ressource plus riche en termes de fonctionnalités, ce qui étend les services du système. Une solution pour garder l'autonomie d'une

application est d'intégrer toute nouvelle fonctionnalité supplémentaire et spécifique à une application sous forme d'intergiciel.

1.3 Quelques systèmes distribués

Dans cette section, nous étudions trois catégories de systèmes distribués à savoir les systèmes pair-à-pair (P2P), les grilles informatiques et le cloud. Le choix d'étudier ces trois catégories est fortement tributaire de leur caractère très hétérogène et leur besoin de passage à l'échelle. L'étude met l'accent sur les architectures et les mécanismes permettant d'assurer la disponibilité, le passage à l'échelle, la transparence et l'autonomie.

1.3.1 systèmes P2P

Le terme P2P fait référence à une classe de systèmes distribués qui utilisent des ressources distribuées pour réaliser une tâche particulière de manière décentralisée. Les ressources sont composées d'entités de calcul (ordinateur ou PDA), de stockage de données, d'un réseau de communication, etc. La tâche à exécuter peut être du calcul distribué, du partage de données (ou de contenu), de la communication et collaboration, d'une plateforme de services, etc. La décentralisation, quant à elle, peut s'appliquer soit aux algorithmes, soit aux données, soit aux méta données, soit à plusieurs d'entre eux. Le paradigme pair à pair de la communication comble cette lacune et a été clairement identifiée comme un moyen pertinent de construire de grands systèmes distribués. Alors que les réseaux de pairs structurés ont dominé dans un premier temps, les réseaux non structurés sont maintenant reconnus comme des infrastructures efficaces pour de nombreuses applications distribuées. Dans ce contexte, les systèmes basés sur la communication sont devenus un outil puissant pour construire et entretenir les réseaux de pairs distribués qu'ils soient structurés ou non structurés, et peuvent être utilisés pour soutenir de nombreuses applications distribuées. Le principe de cette technique, en analogie avec la propagation d'une rumeur parmi les gens, est que les entités participantes échangent des informations en continu, afin de l'étendre progressivement dans le système.

Les réseaux pair-à-pair sont considérés aujourd'hui comme l'une des plus importantes sources de partage de données et leur intérêt ne cesse de croître au fur et à mesure qu'ils sont utilisés dans de nombreux domaines. L'une des particularités des systèmes P2P est que tous les nœuds (pairs) sont en général symétriques, c'est à dire qu'ils jouent

à la fois le rôle de client et de serveur. En particulier, les systèmes de partage de fichiers permettent de rendre les objets d'autant plus disponibles qu'ils sont populaires, en les répliquant sur un grand nombre de nœud. Cela permet alors de diminuer la charge (en nombre de requêtes) imposée aux nœuds partageant les fichiers populaires, ce qui facilite l'augmentation du nombre de clients et donc le passage à l'échelle en taille des données. Un système est dit P2P lorsqu'il autorise la communication directe entre entités d'un réseau, sans passer par une autorité centrale, telle qu'un serveur. Dans un réseau P2P, chaque entité se comporte à la fois comme un client et un serveur. L'architecture des systèmes P2P est donc généralement décentralisée. Les Pair-à-Pair sont en général utilisés pour partager des données entre les utilisateurs ou les applications sont réparties géographiquement. Les données partagées sont souvent distribuées et répliquées pour plus d'évolutivité et de disponibilité. La réplication des données sur un système à grande échelle est très difficile à cause de la dynamicité des nœuds qui peuvent compromettre la cohérence et la disponibilité[3]. Le succès des systèmes P2P est dû aux " bonnes propriétés " de ces systèmes (dynamicité, passage à l'échelle et autonomie). Une classification des systèmes P2P est donnée par la Figure 1.1

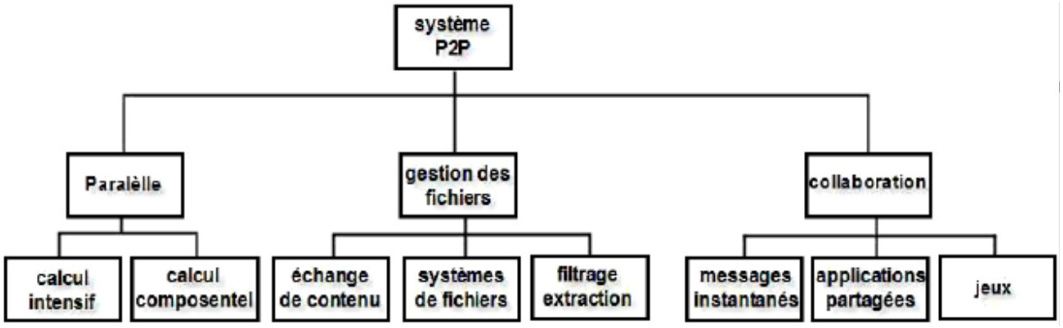


FIGURE 1.1 – Classification des systèmes P2P

1.3.2 Grilles de calculs

Le terme Grille a été introduit pour la première fois aux état Unis durant les années 1990 pour décrire une infrastructure de calcul distribuée, utilisée dans les projets de recherche scientifiques et industriels. Une grille mutualise un ensemble de machines géographiquement distribuées sur plusieurs sites. Un site peut être vu comme un ensemble de clusters, composé d'un ensemble de machines situées généralement à la même localité et qui forment un domaine d'administration local, uniforme et coordonné. La vision des inventeurs de ce terme est qu'il sera possible, à terme, de se brancher sur une grille informatique pour obtenir de la puissance de calcul et/ou de stockage de

données sans savoir ni ou ni comment cette puissance est fournie, à l'image de ce qui se passe pour l'électricité. Cependant la mise en oeuvre de cette transparence n'est pas triviale vue les caractéristiques spécifiques aux grilles. Une grille est caractérisée par sa répartition sur différents sites qui ne sont pas sous administration commune. Cela conduit à une grande hétérogénéité tant au niveau matériel que de l'environnement logiciel. Chaque site d'une grille admet sa propre politique d'administration, son propre protocole d'accès et d'authentification. Les politiques de sécurité peuvent aussi être différentes d'un site à l'autre. Les sites ne partagent pas non plus un même système de fichiers. La grille n'a pas une structure statique. Que ce soit du fait de pannes matérielles, de remplacements ou d'ajouts, des ressources peuvent apparaitre ou disparaitre à tout instant. Une grille peut comporter une grande variété de technologies d'interconnexion réseau, et toute une hiérarchie de réseaux en termes d'étendue géographique, et en termes de performances des communications (débit, latence, etc.). Des réseaux longue distance (Wide-Area Network,WAN) relient les sites de la grille. Les nœuds à l'intérieur de chaque site peuvent être inter-connectés par des réseaux locaux (LAN) ou par des réseaux haute performance (SAN) au sein d'un cluster. La Figure 1.2 montre un exemple d'une grille informatique [2]

FIGURE 1.2 – Les composants de la grille informatique [4]

Il est important de savoir quels avantages une grille est en mesure d'offrir et que les infrastructures et les technologies actuelles ne sont pas capables d'assurer. Nous exposons par la suite quelques unes des raisons pouvant amener à déployer une grille de calcul :

- Exploiter les ressources sous utilisées : les études montrent que les ordinateurs personnels et les stations de travail sont inactifs la plupart du temps. Le taux d'utilisation varie entre 30% pour les milieux académiques et industriels et 5% pour les machines de grand public. Les grilles de calcul permettront ainsi d'utiliser les cycles processeurs durant lesquels les machines sont inactives afin de faire tourner une application nécessitant une puissance de calcul importante et que les machines qui lui sont dédiées n'arrivent pas à assurer. Les cycles processeurs ne sont pas la seule ressource sous utilisée, souvent les capacités de stockage le sont aussi. Ainsi il est possible qu'une grille agrégé toutes ces ressources afin de les partager entre les différents utilisateurs (on parle alors de Grille de Données ou Data Grid). Une autre conséquence d'une telle utilisation est la possibilité de faire du partage de charge entre les différentes ressources d'une grille.

- Fournir une importante capacité de calcul parallèle : le fait de pouvoir fournir une importante capacité de calcul parallèle constitue une caractéristique importante des grilles de calcul. En plus du domaine académique, le milieu industriel bénéficiera énormément d'une telle capacité : bioinformatique, exploration pétrolière, industrie cinématographique, etc. En effet les applications sont écrites d'une façon à pouvoir exploiter parallèlement des ressources (clusters, machines multiprocesseur,. .). Les grilles de calcul peuvent de la même manière fournir des ressources dont l'utilisation pourra se faire en parallèle.

- Meilleure utilisation de certaines ressources : en partageant les ressources, une grille pourra fournir l'accès à des ressources spéciales comme des équipements spécifiques (microscope électronique, bras robotique, . . .) ou des logiciels dont le prix de la licence est élevée. Ainsi ces ressources exposées à tous les utilisateurs seront mieux utilisées et partagées et ainsi on évitera d'avoir recours à installer du nouveau matériel ou acheter de nouvelles licences.

- Fiabilité et disponibilité des services : du fait que les ressources fédérées par une grille de calcul soient géographiquement dispersées et disponibles en importantes quantités permet d'assurer la continuité du service si certaines ressources deviennent inaccessibles. Les logiciels de contrôle et de gestion de la grille seront en mesure de soumettre la demande de calcul à d'autres ressources.

1.3.3 Cloud computing

Le "cloud computing" est un néologisme utilisé pour décrire l'association d'Internet (" cloud ", le nuage) et l'utilisation de l'informatique ("computing"). C'est une manière d'utiliser l'informatique dans laquelle tout est dynamiquement couplé et évolutif et dans laquelle les ressources sont fournies sous la forme de services au travers d'Internet. Les utilisateurs n'ont ainsi besoin d'aucune connaissance ni expérience en rapport avec la technologie derrière les services proposé[5].

Figure 1.3 – Cloud Computing

Le Cloud Computing est un nuage de services et de données. Plus précisément, c'est un paradigme, et à ce titre, il est difficile de lui donner une définition exacte et de dire avec certitude s'il s'agit ou non de Cloud.

Il faut donc être vigilant, car de nombreux fournisseurs de services utilisent le mot "Cloud" des fins marketings. Sur Internet, il n'y a pas de définition exacte du Cloud Computing et donc pas de certification pour dire si nous avons faire un "vrai Cloud". Nous tenterons toutefois, au travers de ce mémoire, de donner les principales clés pour comprendre le Cloud Computing.
-Pour Wikipédia, il s'agit d'un concept de déportation sur des serveurs distants et traitements informatiques traditionnellement localisés sur le poste client [6].

-Pour le Syntec, cela consiste en "une interconnexion et une coopération de ressources informatiques, situées au sein d'une même entité ou dans diverses structures internes, externes ou mixtes, et dont le mode d'accès est basé sur les protocoles et stan-

dards Internet [7].
Pour vulgariser, L'informatique dans le nuage s'appuie sur une infrastructure (le nuage) 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 nuage prennent typiquement la forme de contrats de niveau de service [8].
Le Cloud Computing n'impose aucune dépense en immobilisation puisque les services sont payés en fonction de l'utilisation. Cela permet aux entreprises de ne plus se focaliser sur la gestion, la maintenance et l'exploitation de l'infrastructure ou des services applicatifs.
Les fortes avancées dans le domaine de la virtualisation ont rendu possible le Cloud Computing. Cette virtualisation permet d'optimiser les ressources matérielles en les partageant entre plusieurs environnements (time-sharing). De même, il est possible de coupler l'application (et son système d'exploitation) et le matériel (en capsulé dans la machine virtuelle), cela assure également un provisionning , c'est-à-dire la capacité de déploiement d'environnement, de manière automatique.
Le Cloud Computing couplé, aux technologies de virtualisation, permet la mise à disposition d'infrastructures et de plate-forme à la demande. Mais le Cloud Computing ne concerne pas seulement l'infrastructure, il bouleverse la plate-forme d'exécution et les applications [8].
Le cloud computing correspond au développement et à l'utilisation d'applications accessibles uniquement via Internet. Les utilisateurs dépendent ainsi uniquement d'Internet pour utiliser leurs logiciels, ils ont la possibilité d'accéder à des services sans installer quoique ce soit d'autre qu'un simple navigateur Internet. Aujourd'hui, le cloud computing est exploité par la quasi-totalité des grandes entreprises car il fournit une analyse sophistiquée des données de la manière la plus rapide possible [5].

Historique

Techniquement, le concept de cloud computing est loin d'être nouveau, il est même présent depuis des décennies. On en trouve les premières traces dans les années 1960, quand John McCarty [9] affirmait que cette puissance de traitement informatique serait accessible au public dans le futur. Le terme en lui-même est apparu plus couramment

aux alentours de la fin du XXe siècle et il semblerait que Amazon.com soit l'un des premiers à avoir assemblé des data centers et fournit des accès à des clients. Les entreprises comme IBM et Google ainsi que plusieurs universités ont seulement commencé à s'y intéresser sérieusement aux alentours de 2008, quand le cloud computing est devenu un concept "à la mode" [5]. Réalisant ce qu'ils pourraient faire de toute cette puissance, de nombreuses compagnies ont ensuite commencé à montrer un certain intérêt à échanger leurs anciennes infrastructures et applications internes contre ce que l'on appelle les "pay per-use service" (services payés à l'utilisation) [5]. Auparavant, seuls les super-ordinateurs permettaient de fournir cette puissance et étaient principalement utilisés par des gouvernements, des militaires, des laboratoires et des universités pour réaliser des calculs aussi complexes que prédire le comportement d'un avion en vol, les changements climatiques ou la simulation d'explosions nucléaires. Désormais, des entreprises comme Google fournissent des applications qui exploitent le même type de puissance et sont accessibles à tout moment, de n'importe où et par tout via Internet. Quelques universités prestigieuses ont également lancé leurs propres programmes de cloud computing en fournissant des accès à des maillages de centaines ou milliers de processeurs ; des entreprises comme IBM ont récemment annoncé leur intention d'utiliser massivement le cloud computing a l'avenir. Ces derniers ont récemment dévoilé un système ultra-performant connu sous le nom de "Blue Cloud" qui permettra d'aider les banques et diverses entreprises à distribuer leurs calculs sur un très grand nombre de machines sans posséder d'infrastructure interne. Le 24 mars 2008, Yahoo ! a même annoncé avoir débuté un partenariat avec la Carnegie Mellon University de Pittsburgh afin de leur mettre à disposition, à des fins de recherche, un ordinateur doté de 4 000 processeurs situe dans les locaux de la firme [10].

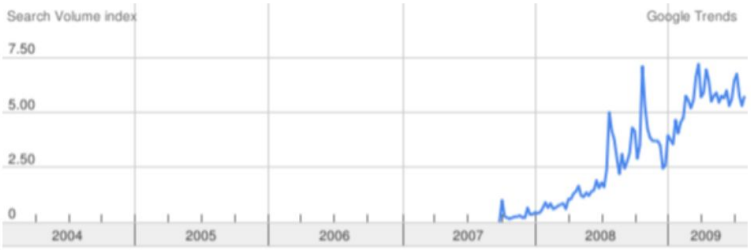


FIGURE 1.4 – Interet pour le terme "cloud computing" sur Internet

Actuellement les experts sont convaincu que bientôt, nous utiliserons le cloud computing de la même manière que nous utilisons l'électricité, c'est à dire en payant unique-

ment ce que nous consommons sans même nous soucier des aspects techniques nécessaires au bon fonctionnement du système. Le principal facteur de développement restant le fait que toute cette puissance est à tout moment partagée par plusieurs utilisateurs et évite ainsi de perdre du "temps machine" à ne rien faire. Cela devrait également drastiquement réduire les coûts de développements et donc les prix [5].

La virtualisation

La virtualisation a été la première pierre vers l'ère du Cloud Computing. En effet, cette notion permet une gestion optimisée des ressources matérielles dans le but de pouvoir y exécuter plusieurs systèmes "virtuels" sur une seule ressource physique et fournir une couche supplémentaire d'abstraction du matériel. Les premiers travaux peuvent être attribués à IBM, qui dans les années 60, travaillait déjà sur les mécanismes de virtualisation en développant dans les centres de recherche de Cambridge et de Grenoble, CMS (Conversation Monitor System), le tout premier hyperviseur.
C'est donc depuis presque 50 ans que l'idée d'une informatique à la demande est présente dans les esprits même si les technologies n'étaient jusqu'alors pas au rendez-vous pour pouvoir concrétiser cette idée. Avec les différents progrès technologiques réalisés durant ces 50 dernières années, tant sur le plan matériel, logiciel et conceptuel, aux avancées des mécanismes de sécurité, à l'élaboration de réseaux complexes mais standardisés comme Internet, et à l'expérience dans l'édition et la gestion de logiciels, services, infrastructures et stockage de données, nous sommes maintenant prêts à entrer dans l'ère du Cloud Computing, telle que rêvait par John McCarthy en 1961 [11].

1.4 Composants du Cloud

La Figure 1.5 illustre les composants communs de cloud computing [12].

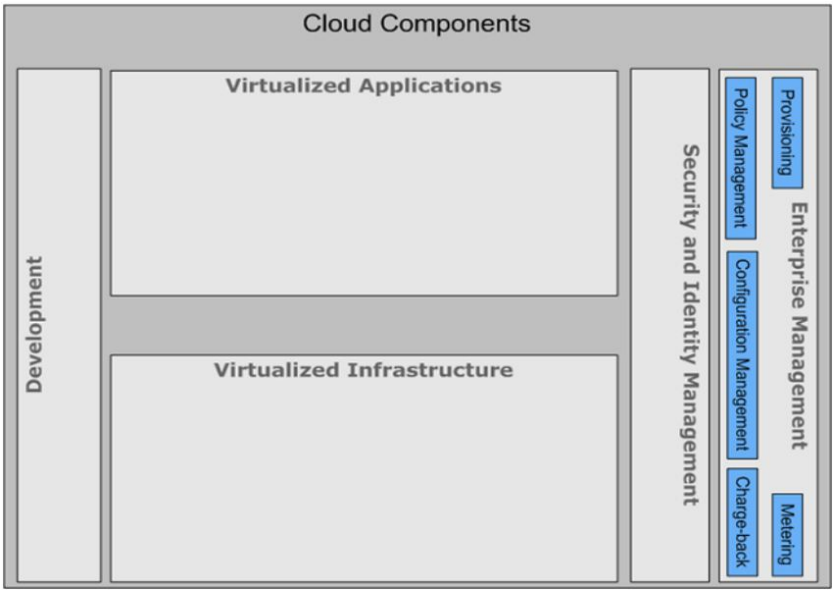


FIGURE 1.5 – Composants du cloud

a. Applications virtualisées : Les applications virtualisées rendent compatible les applications de l'utilisateur avec les hardwares, les systèmes d'exploitations, le réseau et le stockage pour permettra la flexibilité du déploiement.

b. Infrastructure virtualisée : L'infrastructure virtualisée fournit l'abstraction nécessaire pour s'assurer qu'une application ou un service ne soit pas directement attachée à l'infrastructure matérielle (serveurs, stockage ou réseaux). Ceci permet au service de se déplacer dynamiquement à travers les ressources virtualisées d'infrastructure.

c. Gestion de sécurité et d'identité : Le système de gestion de sécurité fournie les commandes nécessaire pour assurer les informations sensibles (les protéger) et rependre au exigence de conformité.

d. Développement : Les infrastructures de développement facilitent non seulement l'orchestration de service mais permettent également aux processus d'être développés. C'est les outils de développement comme le compilateur, SDK (Software Development Kit) et l'environnement de développement.

e. Gestion d'entreprise : La couche de gestion d'entreprise manipule le cycle de vie des ressources virtualisées et fournit les éléments additionnels d'infrastructure com-

mune pour la gestion de taux de disponibilité, utilisation dosée, gestion de politique, gestion de permis, et recouvrement des pertes.

1.5 L'informatique en tant que service

On distingue trois sous-ensembles de services (Figure 1.6) au sein de l'informatique dans le nuage : le logiciel en tant que service (SaaS), la plateforme en tant que service (PaaS) et l'infrastructure en tant que service (IaaS). Chacun de ces types de service correspond à un niveau d'abstraction logiciel précis par rapport aux ressources informatiques matérielles accessibles via Internet, et donc hébergées au sein du nuage du point de vue de l'utilisateur [13].

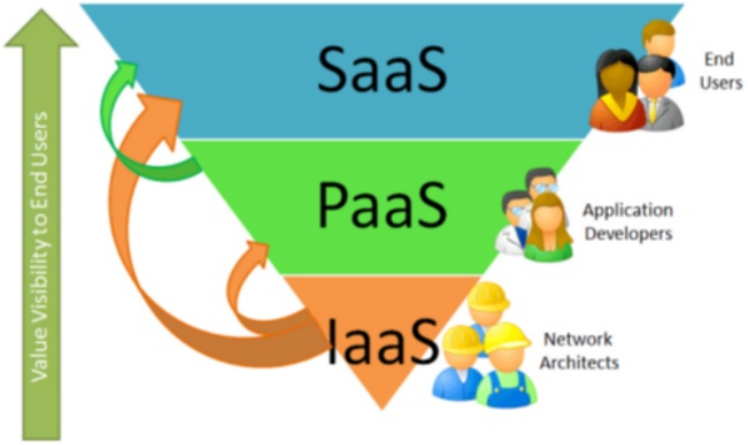


FIGURE 1.6 – Les différents types de services dans le Cloud

C'est l'infrastructure en tant que service qui correspond au plus faible niveau d'abstraction que l'on peut obtenir par rapport aux ressources informatiques partagées via le nuage. En utilisant ce type de service, les utilisateurs peuvent directement administrer les ressources informatiques qu'ils consomment. Un exemple de service appartenant à cette catégorie est la location de serveurs virtuels proposée par Amazon EC2 [14]. Avec cette solution, les clients peuvent installer le système d'exploitation et les composants logiciels de leur choix au sein des espaces d'exécution virtualisés distribués par Amazon, comme ils le feraient sur une grappe de machines privées. A un niveau d'abstraction supérieur, on trouve la plateforme en tant que service qui étend la gamme de services proposés par l'infrastructure en tant que service avec des outils de développement d'applications Web qui sont totalement hébergés chez le fournisseur. Finalement, au plus

haut niveau d'abstraction, on trouve les logiciels en tant que services, qui correspondent à des services logiciels classiques (bureautique, gestion de base de données simplifiée, etc.), dont la particularité est d'être hébergé au sein du nuage et non sur une infrastructure privée.

1.5.1 Infrastructure as a Service (IaaS)

L'infrastructure en tant que service est plus connue sous le nom d'IaaS pour Infrastructure as a Service . Ce type de service consiste à distribuer des ressources informatiques telles que de la capacité de calcul, des moyens de stockage et de communication, de façon publique via Internet et sous une forme de paiement à l'utilisation. Les clients de l'infrastructure en tant que service peuvent donc exécuter et héberger leurs applications informatiques dans le nuage et ne paient que les ressources qu'ils consomment. Ces services d'approvisionnement en infrastructure peuvent servir, d'une part à héberger des logiciels ou plateformes en tant que services et, d'autre part, à être utilisés de façon plus générique en tant que ressources informatiques pour des applications très variées, allant de la sauvegarde de données jusqu'au calcul haute performance, en passant par l'analyse statistique de données. La variété d'utilisation des services proposés par l'infrastructure en tant que service en fait la technologie de l'informatique dans le nuage la plus populaire. Aussi lorsque l'on parle d'informatique dans le nuage, on fait souvent référence à l'infrastructure en tant que service.
La particularité de l'infrastructure en tant que service est de fournir un approvisionnement en ressources informatiques de qualité, qui soit extensible sur commande et dont la capacité dépasse généralement la demande. Le nuage est alors vu par ses utilisateurs comme une source infinie de capacité de calcul, de stockage et de communication. Internet devient alors une place de marché ou l'infrastructure informatique est distribuée, et consommée en tant que marchandise, selon le modèle illustré sur la Figure 1.7.

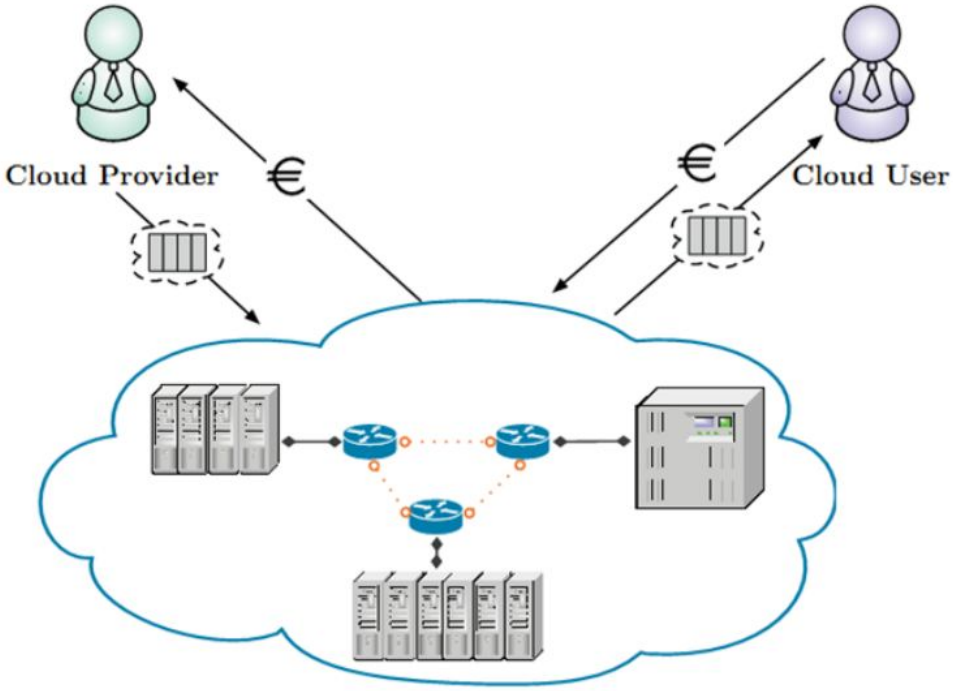


FIGURE 1.7 – Modèle de distribution de l'infrastructure en tant que service

Ce modèle de distribution d'infrastructure assouplit le mode d'investissement en ressources matérielles et logicielles des industries, ouvrant des perspectives jusqu'alors inconnues pour les sociétés et organismes ayant besoin de disposer d'un accès direct à une infrastructure informatique. En effet, d'après les chercheurs de l'Université de Berkeley [15], les trois raisons majeures du succès de l'infrastructure en tant que service sont les suivantes :

1. L'illusion d'une capacité de calcul infinie de la part des utilisateurs du nuage, permettant aux clients de l'informatique dans les nuages de se reposer sur un unique service pour l'approvisionnement de ressources de calcul à long terme.

2. L'assouplissement du mode d'investissement des utilisateurs du nuage, permettant aux entreprises de commencer petit, puis d'augmenter les ressources informatiques matérielles seulement si le besoin s'en fait sentir.

3. La possibilité de payer pour l'utilisation de ressources de calcul sur une base à court terme (c-à-d. accès aux serveurs virtuels à l'heure), permettant une utilisation économe des capacités informatiques, en relâchant les ressources de calcul dés qu'elles deviennent inutilisées.

1.5.2 Platform as a Service (PaaS)

La plateforme en tant que service, plus connue sous l'appellation anglophone Plateforme as a Service (PaaS) se trouve à mi-chemin entre le logiciel en tant que service et l'infrastructure en tant que service. Aussi, il est difficile de définir les frontières entre plateforme et infrastructure, aussi bien qu'entre plateforme et logiciel en tant que service [15]. Cependant, nous proposons de caractériser ce type de service comme suit : la plateforme en tant que service offre un environnement de développement et de déploiement pour les logiciels en tant que service, qui est accessible et hébergé au sein du nuage. Les services offerts par cette technologie facilitent le déploiement des applications (souvent déployées comme logiciels en tant que service), en abstrayant à ses utilisateurs les coûts et la complexité de maintenance de l'infrastructure sous-jacente, et ce, pendant l'intégralité du cycle de vie des applications. Ainsi, ces services sont généralement adressés à des développeurs de logiciels souhaitant utiliser une même plateforme pour les cycles de développement et de déploiement de leurs applications. Les plateformes en tant que services incluent généralement des services d'aide au développement tels que des applications de conceptions, de versionnement, de test, d'intégration de service Web ou de base de données, etc. Elles incluent également des services d'aide au déploiement tels que l'hébergement d'application, la surveillance des applications, le stockage de données, l'allocation dynamique de ressources, la gestion de la persistance des données, etc.

Enfin, les services proposés par une plateforme en tant que service sont généralement délivrés sous la forme d'une solution intégrée accessible via des interfaces Web publiques. On peut distinguer deux types principaux de plateformes en tant que services : les plateformes de développement d'extension et les plateformes de développement d'applications autonomes. Les plateformes de développement d'extension sont généralement mises à disposition par les grands éditeurs de logiciels en tant que services, dans le but de permettre à leurs utilisateurs d'ajouter des fonctions personnalisées aux services classiques. Les plateformes de développement d'extension les plus connues sont proposées par de grands éditeurs de logiciels en tant que services tels que Salesforce [16] et Netsuite [17]. Par exemple, Salesforce propose des outils de création assistée de base de données ou encore de personnalisation d'interface graphique qui sont conçu pour que les utilisateurs puissent personnaliser leur environnement de travail au sein de la plateforme en tant que service. Les plateformes de développement d'applications autonomes sont, dans leur principe, plus proches de l'infrastructure en tant que service. Elles proposent généralement, en plus de la distribution d'infrastructure, des services de développement permettant de se reposer sur le Nuage pour l'intégralité du cycle de

vie des applications clientes. C'est le cas, par exemple, du service Google Code [18] qui prend en charge l'hébergement du code d'une application ainsi que son versionnement. Les plateformes en tant que service favorisent généralement l'utilisation d'une technologie propriétaire ou de services ciblés, comme c'est le cas pour les offres Windows Azure [19] ou Google App Engine [20]. Par exemple, la plateforme Google App Engine met à disposition de ses clients des interfaces de programmation qui facilitent l'intégration des services logiciels distribués par Google, comme la gestion des comptes utilisateurs ou encore la gestion de partage des documents. La plateforme Windows Azure, elle favorise l'utilisation des technologies et services appartenant à Microsoft, comme l'environnement de développement ".Net" ou la gamme de services "Live" .

1.5.3 Software as a Service (SaaS)

Le logiciel en tant que service est un logiciel accessible à la demande, via Internet. Il est également connu sous l'appellation "SaaS" , dérivée de l'expression anglophone "Software as a Service" . Le logiciel en tant que service est un concept apparu au début du siècle. Les logiciels en tant que service sont les services du nuage qui visent le plus grand nombre d'utilisateurs, car contrairement à l'infrastructure et à la plateforme en tant que service, leur utilisation ne demande aucune connaissance particulière en technologie de l'information et des télécommunications. Ces services sont accessibles via Internet, c'est-à-dire hébergés dans le nuage du point de vue de l'utilisateur, et sont généralement utilisables via un simple navigateur web. Une autre particularité est d'être facturé par abonnement plutôt que par licence logicielle. Ils ont été initialement déployés pour automatiser les forces de ventes des entreprises, ainsi que la gestion de leur clientèle, comme ce fut le cas avec la solution Salesforce [16], considérée comme pionnier du logiciel en tant que service. Aujourd'hui, les logiciels en tant que services sont largement utilisés par les entreprises pour différentes tâches telles que la comptabilité, la facturation en ligne, la gestion de ressources humaines et les suites bureautiques de gestion de documents.

L'avantage des logiciels en tant que services est multiple pour les utilisateurs. En premier lieu, ils bénéficient d'un accès nomade et multi plateforme à leurs applications, grâce à l'hébergement dans le nuage et grâce à l'utilisation d'accès standardisés (accès via interface Web). Ensuite, la distribution de logiciel à la demande et le mode de facturation par abonnement permettent aux entreprises clientes d'assouplir leur mode d'investissement dans les technologies informatiques : ils peuvent dynamiquement adapter leur consommation logicielle en fonction de leur besoin. Finalement, ils jouissent d'une uti-

lisation plus simple des services logiciels, sans avoir à se soucier de leur installation ou de leur mise à jour.
Il existe une catégorie particulière de service logiciel distribué via Internet qui n'est pas systématiquement considérée comme faisant partie du logiciel en tant que service, il s'agit des services gratuits à l'utilisation. Ces services sont indirectement financés par la publicité, ou par des produits dérivés de l'analyse statistique à grande échelle.
La distribution du logiciel en tant que service est principalement freinée par deux problématiques : la dépendance technologique vis-à-vis du fournisseur et la confidentialité des données produites par les clients. Ces deux problèmes se généralisent à tous les types de services accessibles via le nuage. Cependant, il est important de comprendre que la confiance accordée au prestataire qui fournit le service logiciel est une composante importante du marché de la distribution logiciel, en particulier lorsque celle-ci est réalisée via le Nuage. Cette notion de confiance explique, en partie, la polarisation du marché vers un faible nombre de grands distributeurs tels que Salesforce [16] ou encore Net Suite [17].
En résumé, le logiciel en tant que service peut être gratuit ou payant, intégrer des notions de rèseautage social ou encore de diffusion de média. Les services ainsi proposés, en particulier lorsqu'ils sont payants, sont régis par des contrats de niveau de service. Ces contrats définissent typiquement le dédommagement prévu pour les clients en cas d'indisponibilité du service vendu. Par exemple, les contrats de niveau de service fournis par Salesforce [16] prévoient de dédommager les clients sous forme de remise forfaitaire en cas d'indisponibilité du service. Le mode de facturation de ce service étant mensuel, lorsque l'abonnement d'un client arrive à échéance, les deux partis (le client et le prestataire) établissent un bilan au sein duquel le taux d'indisponibilité du service durant le mois écoulé est mesuré. Des remises forfaitaires, inscrites au contrat de niveau de service, sont alors appliquées en fonction de cette mesure.

1.5.4 Avantages et Inconvénients des services

Le tableau 1.1 illustre les services de Cloud Computing qui ont été décrites dans la section précédente tout en montrant les avantages et les inconvénients de chaque service [21].

	Avantage	inconvenient
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 1.1 – Les avantages et les inconvénients des différents services

1.6 Modèles de déploiement

Un nuage correspond à une infrastructure distante, dont on ne connait pas les détails architecturaux, et qui est connue pour les services informatiques qu'elle offre. Aussi, il est courant d'utiliser le terme un nuage pour désigner l'infrastructure gérée par un prestataire donné. On peut distinguer trois types principaux de modèles de déploiement pour ces nuages : le nuage privé, le nuage public et le nuage hybride, (Voir Figure 1.8).

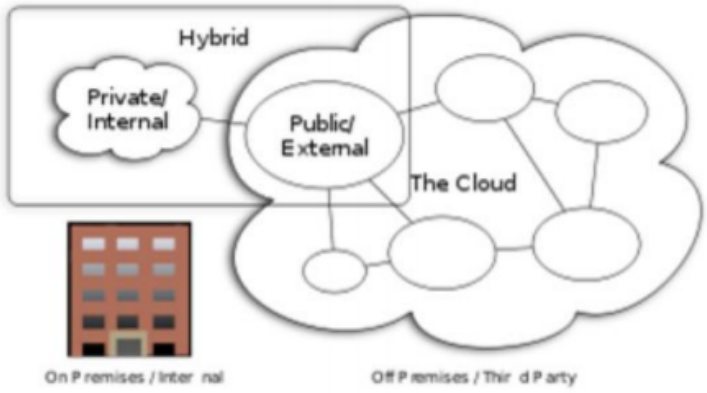


FIGURE 1.8 – Type de cloud computing

1.6.1 Le nuage privé

L'infrastructure d'un nuage privé n'est utilisée que par un unique client. 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 nuage privé permet de garantir, par exemple, que les ressources matérielles allouées ne seront jamais partagées par deux clients différents.

1.6.2 Le nuage public

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

1.6.3 Le nuage hybride

L'infrastructure d'un nuage hybride est une composition de deux types de nuages précédemment cités. Les différents nuages 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 nuages. Une utilisation type de nuage hybride est la répartition de charge entre plusieurs nuages pendant les pics du taux d'utilisation [22].

1.6.4 La différence entre le cloud privé et le cloud public

Dans le cas du cloud public, votre cloud ne vous appartient pas entièrement. Un grand nombre de ressources informatiques sont partagées avec de nombreuses entreprises à travers l'ensemble du réseau Internet. Si ce modèle possède de nombreux avantages en termes de réduction des coûts, de collaboration et d'agilité, pour certaines entreprises, en revanche, il soulève, parfois à juste titre et parfois tort, certaines questions sur la sécurité et la confidentialité des données.
De son coté, le cloud privé propose des ressources informatiques dont l'usage est uniquement réservé à votre entreprise. Vous pouvez héberger votre cloud privé soit sur site, dans votre centre de données (en utilisant une virtualisation et une automatisation à grande échelle), soit hors site chez un fournisseur de services de cloud. Le cloud privé possède la plupart des avantages du cloud public (options de libre-service, relative capacité de montée en charge et facturation interne, par exemple) mais permet davantage

de contrôle et de personnalisation du fait que des ressources dédiées sont à votre disposition. Il peut offrir encore plus de flexibilité, ce qui peut rendre son coût prohibitif et amoindrir les économies d'échelle pour certaines entreprises [22].

1.7 Vers la fédération de nuages ou Intercloud

Un nuage correspond à une infrastructure et à son domaine d'administration. De façon plus simple, il est courant d'associer un nuage à l'entreprise qui est responsable de la gestion de l'infrastructure associée. On parlera alors du nuage d'Amazon, de celui de Google, du nuage de Microsoft,... Cependant, du point de vue d'un utilisateur, cet ensemble de nuages accessibles publiquement via Internet peut être vu comme un méta nuage, au sein duquel un certain nombre de services et de ressources informatiques sont disponibles. De la même façon qu'internet est le réseau des réseaux. Ce méta-nuage est le nuage des nuages et on l'appelle "Intercloud" , son principe est illustré sur la Figure 1.9.

FIGURE 1.9 – Intercloud : le nuage des nuages

Intercloud est un ensemble de services et de ressources informatiques qui sont publiquement disponibles via Internet. Comme décrit au précédent, on retrouve trois grandes

catégories de services : le logiciel, la plateforme et l'infrastructure en tant que service. Le modèle de déploiement des applications distribuées sur une fédération de nuages correspond au nuage hybride. Il a pour particularité d'utiliser plusieurs nuages au sein d'un même environnement de déploiement.

1.8 Avantages et inconvénients du Cloud Computing

Les principaux avantages et inconvénients associés au cloud computing sont les suivants [21] :

1.8.1 Avantages

- Un démarrage rapide : Le cloud computing permet de tester le business plan rapidement, à coût réduits et avec facilité.
- L'agilité pour l'entreprise : Résolution des problèmes de gestion informatique simplement sans avoir à vous engager à long terme.
- Un développement plus rapide des produits : Réduisons le temps de recherche pour les développeurs sur le paramétrage des applications.
- Pas de dépenses de capital : Plus besoin des locaux pour élargir vos infrastructures informatiques.

1.8.2 Inconvénients

- La bande passante peut faire exploser votre budget : La bande passante qui serait nécessaire pour mettre cela dans le Cloud est gigantesque, et les coûts seraient tellement importants qu'il est plus avantageux d'acheter le stockage nous-mêmes plutôt que de payer quelqu'un d'autre pour s'en charger.

- Les performances des applications peuvent être amoindries : Un Cloud public n'améliorera définitivement pas les performances des applications.

- La fiabilité du Cloud : Un grand risque lorsqu'on met une application qui donne des avantages compétitifs ou qui contient des informations clients dans le Cloud.

- Taille de l'entreprise : Si votre entreprise est grande alors vos ressources sont grandes, ce qui inclut une grande consommation du cloud. vous trouverez peut- être plus d'intérêt à mettre au point votre propre Cloud plutôt que d'en utiliser un externalisé. Les gains sont bien plus importants quand on passe d'une petite consommation de ressources à une consommation plus importante.

1.9 La sécurité

Les exigences de sécurité constituent souvent un problème majeur avec des solutions de Software ou de Platform as a Service. Une étude menée par l'Université de Darmstadt révélé que 22 % des personnes interrogées perçoivent les exigences de sécurité comme le principal obstacle à la mise en oeuvre du cloud computing [23]. Les exigences juridiques, de confidentialité et de conformité arrivent en deuxième et troisième positions avec 19,8 % et 11,9 % respectivement. étonnamment, les problèmes techniques sont bien moins perçus comme des obstacles : seulement 7,9 % de personnes expriment des doutes quant à la fiabilité de la solution et un pourcentage très bas (3,4 %) fait référence à des coûts potentiels de performances comme argument contre le cloud computing. Certains s'inquiètent principalement que les données ou l'identité tombent entre de mauvaises mains. Cela a été confirmé dans une étude menée par IBM [23] concluant que 80 % des entreprises craignent pour leur sécurité avec l'introduction du cloud computing.

En dépit des problèmes de sécurité existants, le triomphe du cloud computing sera à peine perturbé. L'aspect financier joue toujours un rôle central dans le processus de prise de décision lorsqu'il s'agit de sélectionner la bonne solution. Mais le cloud le devance : vous ne payez que pour les services que vous utilisez vraiment. Si le nombre d'utilisateurs augmente, il suffit simplement d'ajouter de la capacité pour satisfaire cet accroissement des demandes ; si le nombre d'utilisateurs en ligne diminue, il suffit simplement de réduire la capacité afin de ne pas laisser une infrastructure informatique inutilisée. Au lieu de faire face à de forts investissements initiaux, les entreprises choisissent des coûts opérationnels flexibles et déductibles d'impôt. L'étude menée par l'université de Darmstadt stipule que 22,4 % des répondants considèrent la réduction des coûts comme argument principal dans le choix d'une solution de cloud. L'évolutivité(20,4 %) et la flexibilité accrue (19,9 %) sont les deuxième et troisième raisons [23]. Peu importe que vous pensiez qu'il soit bon ou mauvais, et malgré toutes les préoccupations liées à la sécurité, le cloud computing est la tendance informatique des

années à venir. Vu cette tendance, il convient d'accorder une grande importance aujourd'hui et à l'avenir pour gagner la confiance des entreprises et de s'y tenir.

1.10 Conclusion

L'informatique dans les nuages est un paradigme qui offre un nouveau modèle de distribution et de consommation de ressources informatiques à grande échelle. Les technologies associées à cette discipline permettent aux propriétaires de grands centres de traitement de données de louer les ressources inutilisées dont ils disposent, et de ce fait d'augmenter la rentabilité de leur investissement matériel. Les clients de l'informatique dans le nuage bénéficient également de ce modèle de distribution de ressources, car il leur permet d'assouplir leur mode d'investissement en ressources informatiques, par exemple en ajustant la capacité de traitement de leur infrastructure informatique au fur et à mesure que leurs besoins évoluent.
L'informatique dans le nuage est un concept jeune et en constante évolution. Une des évolutions les plus prometteuses d'après la communauté scientifique est le déploiement d'applications distribuées sur une fédération de nuages. En effet, ce mode de déploiement permet, entre autres, d'atténuer le risque de verrouillage propriétaire, de stimuler la concurrence des différents fournisseurs d'infrastructure dans le nuage, et de contrôler une partie de la confidentialité des données utilisées par les applications déployées dans le nuage. Néanmoins, la gestion des ressources informatiques provenant de plusieurs nuages est un défi technologique et scientifique d'actualité. En effet, la grande taille des fédérations de nuages et l'hétérogénéité des ressources qui les composent sont des aspects difficilement pris en compte par les solutions de l'informatique dans le nuage d'aujourd'hui.

CHAPITRE II :

Réplication et cohérence dans les systèmes distribués

Chapitre 2

Réplication et cohérence dans les systèmes distribués

2.1 Introduction

L'utilisation des techniques de réplication de données permet de mettre en place des solutions, avec plus ou moins d'efficacité, à des catégories de problèmes. Ainsi, la réplication peut contribuer à réduire la latence, à tolérer des fautes et à améliorer les performances. Cependant, malgré les bénéfices qu'elle peut procurer, la réplication pose de nombreux problèmes quant à sa mise en oeuvre : placement, recherche et accès aux répliques, gestion de la cohérence, etc.
Plusieurs travaux de recherches sont actuellement menés pour répondre à ces problèmes. Par exemple, pour déterminer quand et où créer une réplique, un certain nombre de stratégies de réplication ont été proposées.

2.2 Principe de réplication

La réplication met en œuvre un processus qui est chargé de la création, du placement et de la gestion de copies d'entités physiques et/ou logicielles. Les entités répliquées peuvent être des données, du code, des objets, des composants physiques ou une combinaison de tous ces éléments.
La création des copies ou répliques d'une entité consiste à reproduire la structure et l'état des entités répliquées. La copie d'un fichier est un autre fichier de même contenu. La copie d'un programme est un autre programme qui exécute le même code et dont

l'état d'exécution est celui du programme initial. L'intérêt premier de cette réplication est que, si une donnée n'est plus disponible, le système peut continuer à assurer ses fonctionnalités en utilisant une donnée répliquée, ce qui permet d'augmenter la disponibilité des données et la tolérance aux pannes. D'autre part, l'utilisation de cette technique va générer un coût supplémentaire à cause de l'augmentation du travail à fournir, la difficulté principale de la réplication est où faut il placé la réplique ?

2.3 La réplication dans les grilles informatiques

Dans les grilles informatiques la réplication peut être statique ou dynamique [24].
a - Dans la réplication statique les répliques sont manuellement crées, gérées ou supprimées. La réplication statique a donc le problème de ne pas pouvoir être adaptée aux changements suivant le comportement de l'utilisateur. Dans un réel scénario où les données se mesurent par de péta-octets et où existent des centaines de communautés d'utilisateurs du monde entier la réplication statique ne peut être faisable. Elle est par contre utilisée dans les systèmes orientés partage de données et non stockage de données. Exemple : les systèmes pair à pair (P2P).
b - Dans la réplication dynamique la création, la gestion et la suppression se font automatiquement. Les stratégies de réplication dynamiques ont la capacité de s'adapter aux changements suivant le comportement de l'utilisateur.

2.4 Avantages et inconvénients de la réplication

La réplication présente des avantages différents selon le type de réplication et les options choisis, mais l'intérêt général de la réplication est la disponibilité des données à tout moment et en tout lieu. Mais malgré tous les avantages qu'elle procure, la technique de réplication soulève un certain nombre de problèmes. [1]

2.4.1 Avantages :

. Permettre un parallélisme dans la consultation de la même donnée ;
. Améliorer la tolérance aux pannes : la réplication permet les accès aux données même en cas de défaillance d'un support puisque la donnée se trouve sur plusieurs en-

droits ;

.Améliorer les performances : La réplication permet d'améliorer le temps de réponse des requêtes et l'accès aux données pour deux raisons essentielles :

i) les requêtes sont traitées sur un serveur local sans accès à un réseau étendu qui nécessite de la communication ;

ii) le traitement local allège la charge globale des serveurs.

2.4.2 Inconvénients :

- Placement des répliques : Ce problème consiste à choisir, en fonction des objectifs des applications et de la réplication, des localisations physiques pour les répliques, qui réduisent les coûts de stockage et d'accès aux données ;

- Choix d'une réplique : Il s'agit ici de sélectionner, parmi toutes les répliques d'une donnée, celle qui est la meilleure du point de vue de la consistance ;

- Degré de réplication : Ce problème concerne la recherche du nombre minimal de répliques qu'il faut créer pour une donnée, sans réduire les performances des applications ;

- Cohérence des répliques : les techniques de réplication n'assurent pas une cohérence des données de l'ensemble des répliques. Ainsi, il est possible d'avoir, à un instant donnée, des copies différentes d'un même ensemble de données sur différents nœuds.

2.5 Technique de réplication des données

La réplication est aujourd'hui largement utilisée dans les Clouds. Elle consiste à créer plusieurs copies d'un même fichier sur des ressources de stockage différentes en mettant en œuvre un processus de création et de placement des copies d'entités logiciels. La phase de création consiste à reproduire la structure et l'état des entités répliquées, tandis que la phase de placement consiste à choisir, en fonction des objectifs de la réplication, le bon emplacement de cette nouvelle reproduction. Cette technique permet d'améliorer la fiabilité, la tolérance aux pannes, l'accessibilité et d'augmenter la disponibilité des données, la charge étant alors répartie sur les différents nœuds possédant une réplique [25] [24] [26].

2.5.1 Création des répliques :

Ranganathan et Foster définissent les quatre questions auxquelles une stratégie de création de répliques doit répondre : [24] [26]
-Quand créer les répliques ? moment de la réplication.
-Quels fichiers doivent être répliqués ? choix de l'entité à répliquer.
-Où les répliques doivent-elles être placées ? placement des répliques.
-Comment une copie est-elle crée ? manière de répliquer une entité.

1. **.Moment de la réplication :** Pour répondre à la question quand ? deux solutions sont possibles [27] :
   - Réplication statique : les répliques persistent jusqu'à ce qu'elles soient effacées par l'utilisateur du nœuds sur lequel elles sont hébergées ou que leurs durées de vie respectives expirent. L'avantage de ce schéma est sa simplicité, son inconvénient est sa non-adaptabilité aux changements de comportement des participants.
   - Réplication dynamique : contrairement à la réplication statique, la réplication dynamique crée et supprime automatiquement les copies selon l'évolution des demandes des utilisateurs. L'avantage est la réduction des points d'engorgements et l'équilibrage de la charge. L'inconvénient observé est l'induction de coûts supplémentaires causés par l'évaluation en temps-réel du trafic réseau pour prendre les décisions de réplication.
   Selon le moment de la réplication, on distingue :
   o Réplication à la demande : la réplique est créée suite à la demande d'un client.
   o Réplication périodique : elle est indépendante des requêtes des clients. Son but est de permettre la gestion automatique de répliques avec des stratégies adaptées aux comportements des clients. Le processus de réplication est déclenché à chaque intervalle de temps (période).

2. **.Choix de l'entité à répliquer :** Pour répondre à la question quoi : les données répliquées sont généralement de deux types : des fichiers ou des objets. Les objets peuvent être composés d'un ensemble de fichiers distribués (on les appelle aussi collection). Selon les stratégies de réplication, les données à répliquer, peuvent être les plus populaires ou encore les plus fréquemment accédées.

3. **.Placement des répliques :** Pour répondre à la question où : les stratégies de

placement de répliques doivent tenir compte du fait que les sites potentiels :
a. ne possèdent pas déjà de réplique de la donnée ;
b. possèdent l'espace de stockage suffisant ;
c. sont à une distance raisonnable en termes de temps de transfert.

4. **.Manière de répliquer une entité :** Pour répondre à la question comment : Le processus de création de copie dépend de la structure et de l'état de l'entité à répliquer. La structure de l'entité peut être indivisible ou composée, alors que l'état peut être constitué de données, de code et éventuellement d'un état d'exécution. Les problèmes de coûts sont au centre des stratégies de réplication. Un enjeu majeur de la réplication est la réduction de la latence d'accès ainsi que la consommation de bande passante [36].

2.6 Protocoles de réplication

Trois principaux protocoles sont utilisés pour la gestion des répliques dans les systèmes distribués :

2.6.1 Protocole de réplication passive

Dans ce protocole, une seule copie reçoit une requête d'un client et l'exécute. Cette copie est désignée sous le nom de copie primaire (primary copy). Elle a la tâche d'effectuer tous les traitements, alors que les copies secondaires ne font aucune action (voir Figure 2.1). En cas de défaillance de la copie primaire, une copie secondaire devient (par un protocole d'élection) la nouvelle copie primaire [32]. Pour assurer la cohérence, la copie primaire diffuse régulièrement son nouvel état à toutes les copies secondaires

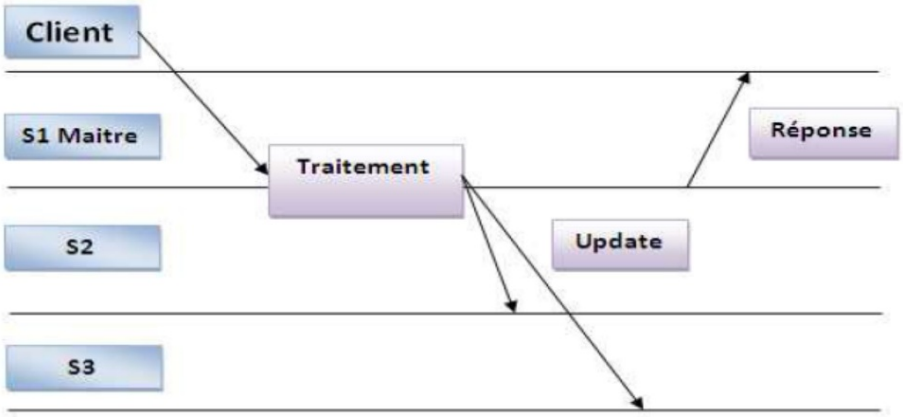


FIGURE 2.1 – Protocole de réplication passive

2.6.2 Protocole de réplication active

Dans un protocole de réplication active, chaque copie joue un rôle identique à celui des autres copies. Toutes les copies reçoivent la même séquence, totalement ordonnée, des requêtes des clients, les exécutent puis renvoient la même séquence, totalement ordonnée, des réponses(voir Figure 2.2).

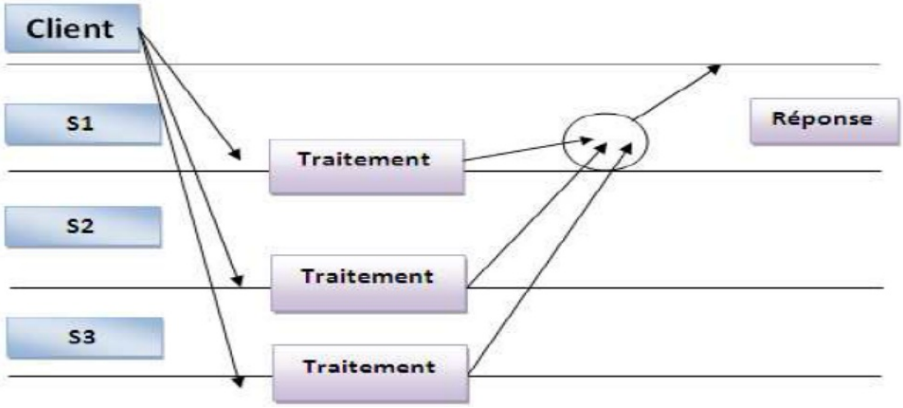


FIGURE 2.2 – Protocole de réplication active

2.6.3 Protocole de réplication semi-active

C'est un protocole hybride qui se situe entre les deux protocoles précédents, où toutes les copies exécutent en même temps la requête du client, mais une seule copie (leader) d'entre elles émet la réponse, les autres copies (suiveurs) mettent à jour leur

état interne et sont donc étroitement synchronisées avec le leader(voir Figure 2.3).

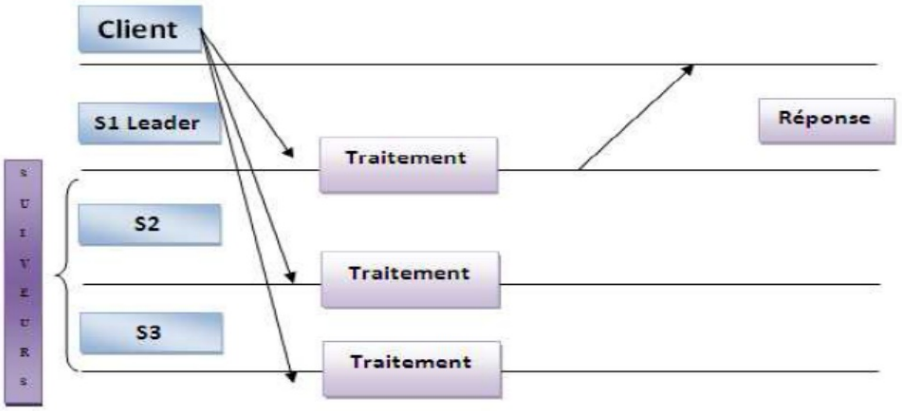


FIGURE 2.3 – Protocole de réplication semi-active

Les principales caractéristiques d'un protocole de réplication sont :

1. Le nombre de copies concernées par une lecture ou une écriture :
Pour pouvoir répondre à une requête externe de lecture ou d'écriture, chaque protocole de réplication suit ses propres contraintes sur le nombre de copies à consulter.
Par exemple, certains protocoles font une écriture sur toutes les copies avant de valider une requête externe d'écriture. Dans ce cas, ils n'ont besoin de consulter qu'une copie pour répondre à une requête externe d'écriture, alors que d'autres protocoles valident une requête externe d'écriture lorsque n/2+1 copies sont mises à jour (n est le nombre total de copies). Lors d'une requête externe de lecture, il suffit alors de consulter n/2 copies, pour pouvoir répondre au demandeur[1].

2. Les droits d'accès : Il existe deux approches selon lesquelles la détermination des copies qui peuvent être modifiées est faite :
l'approche maître-esclaves ou primaire-secondaire (master-slave ou primary-secondary) et l'approche copies identiques (update anywhere ou peer to peer)[28].

i) Approche maître-esclaves : Chaque objet répliqué possède une copie dite maîtresse, les autres étant des copies esclaves. Toutes les mises à jour (requêtes d'écriture) sont d'abord exécutées sur la copie maîtresse (primaire), ensuite les modifications sont diffusées aux copies esclaves (voir la Figure 2.4).

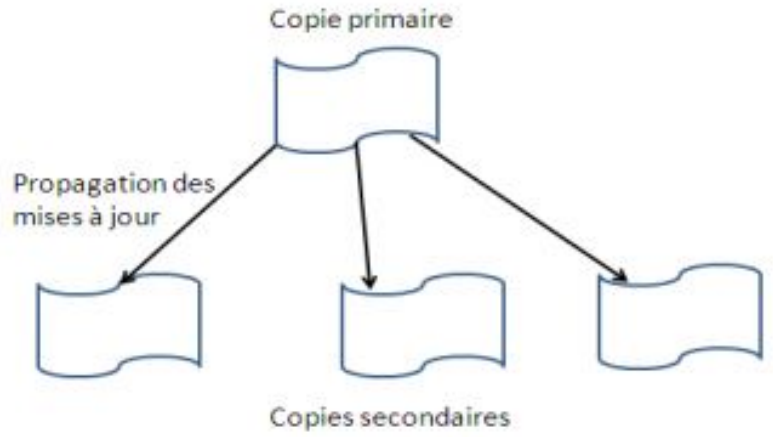


FIGURE 2.4 – Approche maître-esclaves

ii) Approche copies identiques : Toutes les copies sont des maîtresses, c'est-a dire, chaque requête de type lecture ou écriture peut être traitée sur n'importe quelle réplique en concurrence. A chaque fois qu'une copie traite une requête d'écriture, elle propage les mises à jour aux autres copies (voir la Figure 2.5)[29].

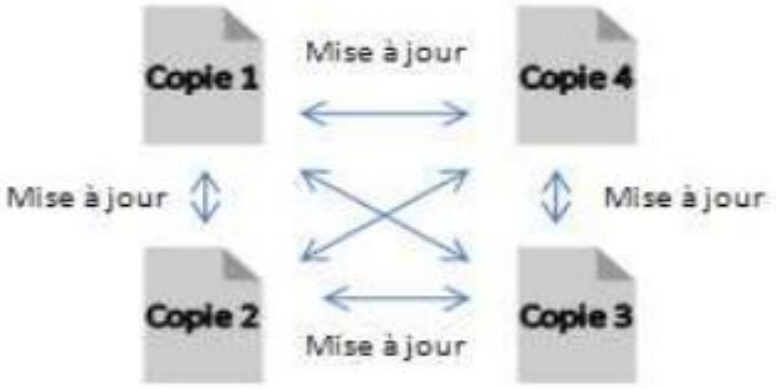


FIGURE 2.5 – Approche copies identiques

3. Synchronisation des répliques La mise à jour des différentes copies peut se faire simultanément sur toutes les copies ou d'abord sur une et ensuite sur les autres. Nous pouvons classifier les moments de synchronisation en sept instants de

déclenchement[30] :

a. Conditions sur le délai : Ces conditions portent sur le temps. Elles expriment la durée maximale que le protocole peut attendre avant la propagation d'une mise à jour. Par exemple, un délai maximum de 60 secondes signifie que, toutes les mises à jour d'une copie doivent être propagées vers une autre copie, avant l'expiration de ce délai.

b. Conditions sur la périodicité : Elles expriment qu'une copie d'un objet doit être mise à jour avec la dernière valeur de l'objet, toutes les n unités de temps (périodes), que l'objet ait été modifié ou non.

c. Conditions sur le moment : Ces contraintes expriment le fait qu'une copie d'un objet doit être mise à jour avec la dernière valeur de l'objet à un instant donné.

d. Conditions sur la version : Elles spécifient le nombre de modifications pouvant avoir lieu sur une copie avant de propager les mises à jour sur une autre copie.

e. Conditions numériques : Si la donnée est numérique, ces conditions permettent de limiter l'écart "d" entre les valeurs des différentes copies d'un objet.

f. Conditions sur les objets : Ces conditions portent sur la structure des objets. Il est possible de spécifier qu'une copie X d'un objet O doit être mise à jour avec la dernière valeur de O.

g. Conditions d'événements : Ces conditions portent sur les événements de déclenchement des mises à jour des copies.

4. L'initiative de la mise la jour des répliques

Les politiques de propagation des mises à jour peuvent être classifiées en deux approches[29] :

i) Approche Push : Lorsque une copie reçoit une mise à jour, elle initie l'opération de mise à jour des autres copies.

ii) Approche Pull : Chaque copie demande la mise à jour aux autres, c'est-a-dire, que si une copie reçoit une mise à jour, elle n'informe pas les autres, mais ce sont les autres qui initient la propagation de mise à jour.

5. La nature des mises à jour

Les mises à jour utilisées peuvent être soit un transfert d'état soit un transfert d'opérations[29] :
i) Un transfert d'état : C'est de transférer l'état (le contenu en entier) de la copie source vers les autres copies.
ii) Un transfert d'opération : C'est de diffuser l'opération exécutée sur la copie source vers les autres copies pour y être exécutée.

6. Le cheminement (ou la topographie) des mises à jour
La propagation des mises à jour peut suivre plusieurs chemins. Le choix du chemin se justifie, soit par le fait que certaines copies doivent être mises à jour avant d'autres, soit par la topologie du réseau qu'elles utilisent. La propagation des mises à jour peut utiliser des protocoles de communication des réseaux tels que Unicast, Multicast ou Broadcast[30].
7. La capture des mises à jour
Le mécanisme utilisé, pour détecter et sélectionner les changements sur une copie, afin de les propager aux autres copies est appelé la capture. Il peut s'implanter de diverses façons. Une manière de faire, consiste simplement à consulter la copie, afin de connaître son dernier état. Une deuxième façon de faire, consiste à enregistrer les modifications sur un support particulier : un journal (log sniffing) ou une copie ombre (shadow)[30].
8. La gestion des conflits
Dans certains protocoles, deux copies peuvent être modifiées de manière concurrente. Lorsque le protocole désire synchroniser les copies (pas forcément immédiatement), il se trouve face à des conflits, donc, il détecte le conflit et réconcilie les différentes copies, afin de ne pas perdre de modifications[1].
- Détection d'un conflit : lors du fonctionnement normal d'un protocole de réplication, la détection des conflits est l'action à postériori des accès conflictuels sur différentes copies.
- Réconciliation d'un conflit : suite à la détection d'un conflit, la réconciliation est l'action de résoudre un conflit [1].

2.7 Notion de cohérence

La cohérence est une relation qui définit le degré de similitude entre les copies d'une entité répliquée. Dans le cas idéal, cette relation caractérise des copies qui ont des comportements identiques. Dans les cas réels, où les copies évoluent de manière différente, la cohérence définit les limites de divergence autorisées entre copies. La relation de cohérence est assurée par synchronisation entre copies. [31]

Traditionnellement on dit que la mémoire est cohérente si la valeur retournée par une opération de lecture d'une donnée correspond toujours à la dernière valeur écrite de cette même donnée (cohérence stricte). Il existe des modèles de cohérence plus faibles qui permettent l'implémentation de protocoles moins coûteux en nombre et en taille des messages, en imposant en contrepartie plus de contraintes au programmeur.

2.7.1 Modèles de cohérence

La notion centrale dans un système distribué est le modèle de cohérence utilisé. Un modèle de cohérence s'affiche comme un contrat passé entre le système et le programmeur. Il définit les critères déterminant la valeur retournée par une lecture en fonction des écritures précédentes. Il existe plusieurs modèles de cohérence appartenant aux classes de cohérence forte et relâchée.
I) Modèles de cohérence forte : Les modèles de cohérence forte sont caractérisés pas des contraintes fortes entre la dernière écriture et la prochaine lecture.

1)Le modèle strict (atomic consistency) : est un modèle idéal où chaque lecture rend la dernière valeur écrite dans la donnée. Dans les systèmes distribués, le protocole associé nécessite l'utilisation d'une horloge globale, ce qui rend son implémentation impossible.

2)Le modèle séquentiel (sequential consistency (SC)) : Ce model est formalisé par Lamport en 1979 assure que chaque site voit toutes les opérations dans le même ordre. Les premières MVP(Minimum Viable Product), comme IVY(Integrated shared Virtual memory at Yale) , utilisent ce modèle de cohérence et propose de combiner les modèles strict et séquentiel en offrant deux primitives de lecture en fonction du modèle à utiliser pour la donnée considérée.

3)Le modèle causal (causal consistency) : Il se base sur la relation de causalité introduite pour déterminer un ordre entre les écritures. De nombreuses applications tolèrent que deux évènements ne soient pas vus dans le même ordre sur tous les sites. Le modèle causal permet alors de lier certains évènements entre eux par un ordre bien fondé tout en relâchant les contraintes sur les évènements indépendants.

II) Modèles de cohérence relachée : Les modèles de cohérence relâchée ont été introduits afin de diminuer le nombre d'échanges réseau induit par les protocoles de

cohérence forte. Ils tirent parti du fait quelles applications distribuées imposent déjà un ordre sur les accès mémoire par l'utilisation explicite de mécanismes de synchronisation. Ce modèle est composé de :

1)La cohérence faible (weak consistency) Il fait la distinction entre les accès ordinaires à la mémoire et les accès synchronisés. Seuls les accès synchronisés garantissent la cohérence de la mémoire partagée par l'utilisation d'objets de synchronisation comme les verrous ou les barrières. Ce modèle garantit que la mémoire est cohérente à chacun des points de synchronisation pendant le quels toutes les informations sont mises à jour.

2)La cohérence à la libération (Eager Release Consistency (ERC)) : Ce modèle améliore la cohérence faible en ne mettant à jour que les données modifiées entre deux synchronisations. Ce modèle utilise le principe de section critique gérée par un verrou et délimitée par les primitives acquire et release. La cohérence de la mémoire est assurée par la propagation vers les autres sites des modifications effectuées sur la donnée dans la section critique. La particularité de ce modèle est que la mise à jour intervient lors de l'appel à la primitive release. Ce protocole génère des communications inutiles vers des sites n'effectuant par la suite aucun accès sur les données mises à jour. Le protocole associé est mis en œuvre dans le système à MVP Munin .

3)La cohérence à la libération paresseuse (Lazy Release Consistency (LRC)) : est une version plus relachée d'ERC qui tente de réduire les communications inutiles de ce dernier. Une liste des données modifiées (write notice (wn)) est envoyée au site effectuant le prochain appel à la primitive acquire.
Les modifications ne s'appliquent alors que sur ce site et uniquement lors de l'accès en lecture ou écriture à une donnée déclarée modifiée dans wn.

4)La cohérence à l'entrée (Entry Consistency (EC)) : Elle a été proposée par le système à MVP Midway et elle tente de limiter les effets du faux-partage apparaissant dans LRC en associant à chaque variable partagée un objet de synchronisation. L'établissement de cette relation est laissée à la charge du programmeur.

5)La cohérence de portée (Scope Consistency (ScC)) : Elle reprend le principe de l'EC et tente d'éviter au programmeur d'effectuer lui-même l'association entre verrous et données. Cette technique se base sur les instructions de synchronisation déjà présentées dans le programme. Lors de l'acquisition d'un verrou par un site, seules les

modifications effectuées dans les portées correspondant à ce verrou sont visibles.

2.8 Conclusion

Ce chapitre a été axé sur la technique de la réplication de données. Nous avons signalé également que cette technique peut améliorer la disponibilité de données et la performance de l'accès aux données. A travers ce chapitre, nous avons présenté dans un premier lieu les principales techniques et stratégies de réplication et puis nous avons terminé ce chapitre par la description des protocoles de cohérence qui permettent la convergence des répliques. Elle peut nécessiter la technique de réplication de données afin d'augmenter la disponibilité de données localement. Nous pensons que cette démarche pourra améliorer les performances.

CHAPITRE III :

Description et modélisation de

l'approche proposée

Chapitre 3

Description et modélisation de l'approche proposée

3.1 Introduction

Il y a Plusieurs stratégies pour la réplication . Certaines d'entre elles utilisent un seuil (si le nombre de demandes d'un fichier dépasse ce seuil, alors une réplique du fichier est créée). La plupart des travaux utilisent un seuil fixe pour tous les fichiers et pour tous les Hosts du Data Center. Nous proposons dans ce chapitre une stratégie utilisant un seuil dynamique qui varie, d'une part, selon le comportement de l'utilisateur, et d'autre part selon les niveaux du Cloud à travers la largeur de la bande passante.

3.2 Création et placement de répliques

La distribution de répliques dans le Cloud computing peut prendre plusieurs formes. Dans ce travail, nous avons utilisé une distribution sur une topologie réseau d'architecture hiérarchique multi-niveau.

3.2.1 Topologie du Cloud

La topologie du Cloud utilisée dans le modèle et inspirée de l'architecture de grille de données CERN et de l'Article .Elle est composée de trois niveaux : les Data Centers (niveau 1), deux niveaux intermédiaires contenant des nœuds (niveau 2 à 3) . Tous les nœuds y compris les Data Centers, représentant des serveurs, peuvent avoir des répliques des données, sauf le dernier niveau, celui des VM et les feuilles qui représente les clients (Cloudlets) d'où les requêtes sont émises.

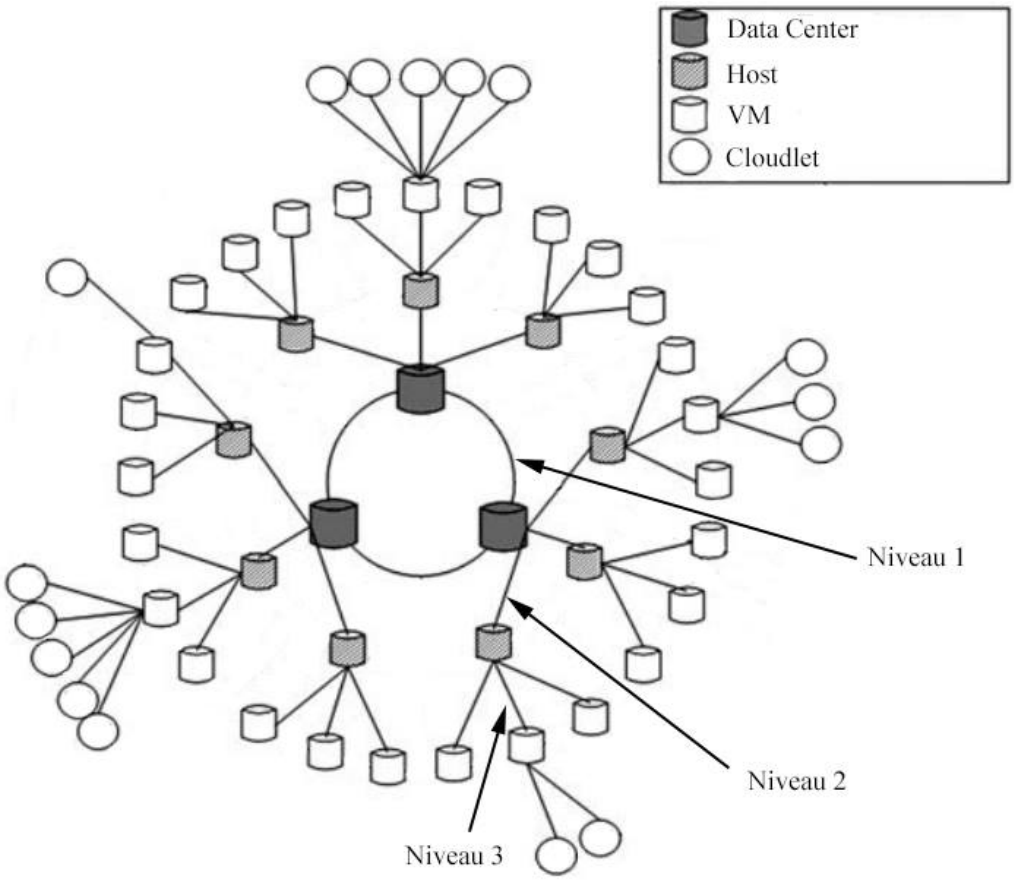


FIGURE 3.1 – Topologie du Cloud utilisée

L'architecture hiérarchique multi-niveau du système du cloud supporte une méthode efficace pour le partage de données, calculs et autres ressources, comme la représentation dans la Fig.3.1. Elle se compose typiquement de plusieurs niveaux différents avec des tailles différentes. Les Data Centers qui se trouvent dans le niveau 1 vont s'occuper de l'analyse des données dans l'intra domaine et l'échange d'information de données entre les inters domaines. Les Hosts sont dans le niveau 2, les VMs sont dans le niveau 3 et les Cloudlets sont les feuilles. L'architecture minimise le temps d'accès au données et la charge du réseau en créant et déployant des répliques depuis les Data Centers vers des Data Centers différents ou bien des Hosts. Les Data Centers collectionnent et diffusent l'information global périodiquement.

3.2.2 Modèle de coût

La stratégie de placement proposée est basée sur un modèle de coût qui utilise les paramètres décrits dans le tableau suivant :

T_{req}	Taille de la requête
Td	Taille de la donnée
T_{res}	Taille de la réponse (résultat de la requête)
BW_n^{n+1}	Largeur de bande entre le niveau n et le niveau n+1
S	Le seuil
C_{req}	Coût de transmission de la requête
C_{res}	Coût de réponse
C_{rep}	Coût de réplication de la donnée du niveau n au niveau n-1
$C_{acc}(C)$	Coût d'accès à la donnée d'une Cloudlet C
CA	Le nombre de demande d'accès au fichier

TABLE 3.1 – Paramètres utilisés dans le modèle

La décision de réplication se fait en comparant deux coûts :

. coût d'accès à la donnée.

. coût de la réplication.

Le coût d'accès à la donnée est composé de deux coûts :

$$C_{acc} = C_{req} + C_{res}$$

Notons que c'est le coût d'un seul accès.

. Coût de transmission de la requête Creq :

$$C_{req} = T_{req} * \sum_{n=3}^{i+1} \frac{1}{BW_n^{n-1}}$$

. Coût de réponse Cres :

$$C_{res} = T_{res} * \sum_{n=1}^{2} \frac{1}{BW_n^{n+1}}$$

i : le niveau où se trouve la donnée demandée par les clients (initialement égal à 0 : le niveau de la racine). Le coût de réplication de la donnée d'un niveau i au niveau i+1 est calculé comme suit :

$$C_{rep} = T_d * \frac{1}{BW_i^{i+1}} + C_{acc}^{i+1}$$

C_{acc}^{i+1} : Le coût d'accès au nouveau niveau où la donnée est répliquée.

3.2.3 Algorithme de base

Pour prendre une décision de réplication, il faut comparer entre le coût d'accès à la donnée et le coût de la réplication.
Plusieurs stratégies utilisent un seuil afin de prendre une décision de réplication. Quand le nombre de d'accès dépasse un seuil, on fait répliquer le fichier demandé. Le seuil est prédéfini et fixé pour tous les niveaux de l'architecture hiérarchique, et pour n'importe quelle donnée.
Plusieurs facteurs peuvent influer sur la précision du seuil tels que la largeur de la bande passante entre les Cloudlets et les sources de données, et la taille de la donnée demandée par une Cloudlets.
Notre algorithme est composé de trois phases :

a. calcul des coûts et nombres d'accès : À travers les enregistrements d'accès historiques des fichiers au niveau des Cloudlets, nous calculons les coûts d'accès C_{acc} d'une donnée pour chaque Cloudlet :

$$C_{acc} = 1 * (C_{req} + C_{res}\ )$$

Note : le coût d'accès est calculé par rapport à une seule demande de chaque Cloudlet (on ne tient pas compte du nombre d'accès), puisque par la suite nous allons calculer le seuil, qui est expliqué par le nombre de demande maximal.

Ensuite, le calcul additionne simplement les nombres d'accès et les coûts d'accès pour les enregistrements dont les nœuds sont des enfants de mêmes parents et qui se rapportent aux mêmes fichiers, étape par étape jusqu'au Data Center.

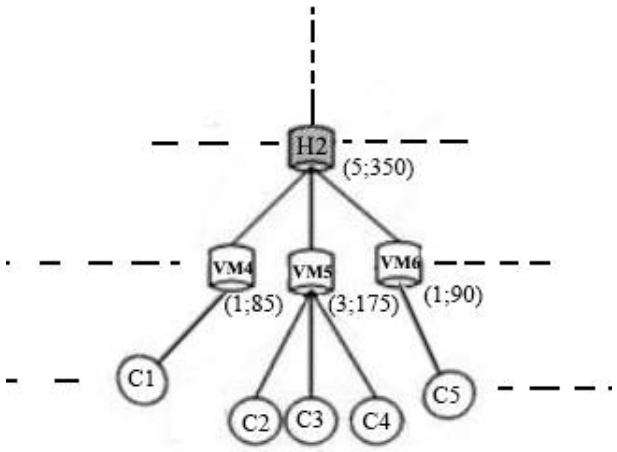


FIGURE 3.2 – Calcul des coûts et nombres d'accès de bas en haut.

Pour l'exemple de la figure 3.2, le coût d'accès d'une donnée D, qui se trouve dans le Data Center, par la Cloudlet C1 est calculé comme suit :

$$C_{acc}(c1) = 1 * (C_{req} + C_{res})$$

Supposons que c'est égal à 85 ms pour C1, 73 ms pour C2, 52 pour C3, 50 pour C4 et 90 pour C5.
Au niveau du Nœud VM5, le coût d'accès et le nombre d'accès sont égales à la somme de celles des Cloudlets C2,C3 et C4 ; au niveau du nœud H2, le coût d'accès et le nombre d'accès sont égales à la somme de celles des VMs VM1,VM2 et VM3 ; de la même manière on peut calculer les coûts et les nombres d'accès de tous les nœuds de l'architecture.

b. calcul du seuil : le calcul se fait au niveau de chaque nœud du Cloud.
Commençons par le Data Center (niveau 0 de l'architecture) ; à partir d'un seuil S_0 on doit toujours trouver que le coût d'accès est supérieur au coût de réplication, on aura la formule suivante :

$$S_0 * C_{acc} > C_{rep}$$

Ce qui implique

$$S_0 > \frac{C_{req}}{C_{acc}}$$

Le coût de réplication initialement est calculé en répliquant la donnée du niveau 0 au niveau 1. Donc :

$$S_0 > \frac{T_d * \frac{1}{BW_0^1} + C_{acc}^1}{C_{acc}} = S_0'$$

L'équation précédente donne les résultats suivants :

$$S_0 \in]S_0', +\infty[$$

Comme solution optimale on prend le plus petit seuil, on aura :

$$S_0 = Int(S_0') + 1$$

Int(S_0') : La partie entière de S_0' , puisque le seuil est un nombre entier positif.
Note : Concernant le calcul du seuil des autres nœuds, nous devons refaire la phase a, tout simplement parce que le coût d'accès tient compte du nouveau placement de la donnée. Donc le seuil dépend de la taille de la donnée, de la réponse et dépend aussi du niveau de l'architecture, qui est expliqué par la largeur de la bande passante entre les niveaux.

c. Placement de réplique et Suppression : en utilisant les nombres d'accès calculés dans la première phase et les seuils déterminés dans la deuxième phase, nous pouvons définir la stratégie de placement de réplique comme suit :
En commençant par le Data Center, nous traversons le fond de la hiérarchie tant que le nombre d'accès de l'un des fils est supérieur ou égal au seuil du père, jusqu'à arriver au dernier niveau. Une réplique est placée sur le nœud le plus populaire (le Host qui contient le plus de VM) ceci est dans le cas où le fichier demandé se trouve dans le même Data Center, Si le fichier demandé se trouve dans un autre Data Center la réplique est créer dans le Data Center dont la requête a était émise .
La suppression s'effectue seulement si l'espace de stockage est plein, dans ce cas là on tri les fichiers par leurs taille et on supprime le fichier le plus volumineux, on refait la même chose jusqu'à ce qu'il ya un espace suffisant pour créer la réplique .

Remarque : On supprime seulement les répliques et non pas les fichiers originaux.
Un exemple de démonstration est donné dans la section 4.

Note : puisque le niveau des Cloudlets n'est pas concerné par le placement de répliques, alors le dernier niveau intermédiaire (niveau 3) ne possède pas de seuils. L'algorithme de réplication proposé avec ses trois phases est répété dans chaque intervalle de temps dt en réinitialisant les nombres d'accès des Cloudlets au 0.

3.3 Algorithme de l'approche proposée

Notre algorithme permet de déterminer quand et où placer la réplique.
N.B : On suppose que chaque Data Center a son propre serveur de données qui est représenté dans les Caractéristiques du Data Center en tant qu'espace de stockage.
Le Host le plus populaire est celui qui possède le plus de VMs et de Cloudlets, vue qu'on va utiliser le Binding afin d'exécuter les Cloudlets dans des VMs spécifique.
On supprime seulement les répliques et non pas les fichiers originaux.

Algorithm 1 Approche proposée
Début
Lancement de la Cloudlet ;
Obtention de la liste des fichiers demandés par la Cloudlet ;
Comparer et localiser les fichiers du Data Center ;
Calculer les seuils seulement des fichiers demandés ;
Si le fichier demandé se trouve dans le même Data Center où la Cloudlet est exécutée et que le nombre d'accès dépasse le seuil **Alors**
Déterminer le Host le plus populaire ;
Si l'espace de stockage est suffisant alors créer la réplique ;
Sinon trier les fichiers par taille et supprimer le fichier le plus volumineux **Jusqu'à** ce que l'espace disponible permet de créer la réplique ;
Si le fichier demandé se trouve dans un autre Data Center où la Cloudlet est exécutée que le nombre d'accès dépasse le seuil **Alors**
Si l'espace de stockage est suffisant alors créer la réplique ;
Sinon trier les fichiers par taille et supprimer seulement le fichier le plus volumineux et qui est une réplique (n'est pas un fichier original) ;
Répété le processus de suppression **Jusqu'à** ce que l'espace disponible permet de créer la réplique
Fin.

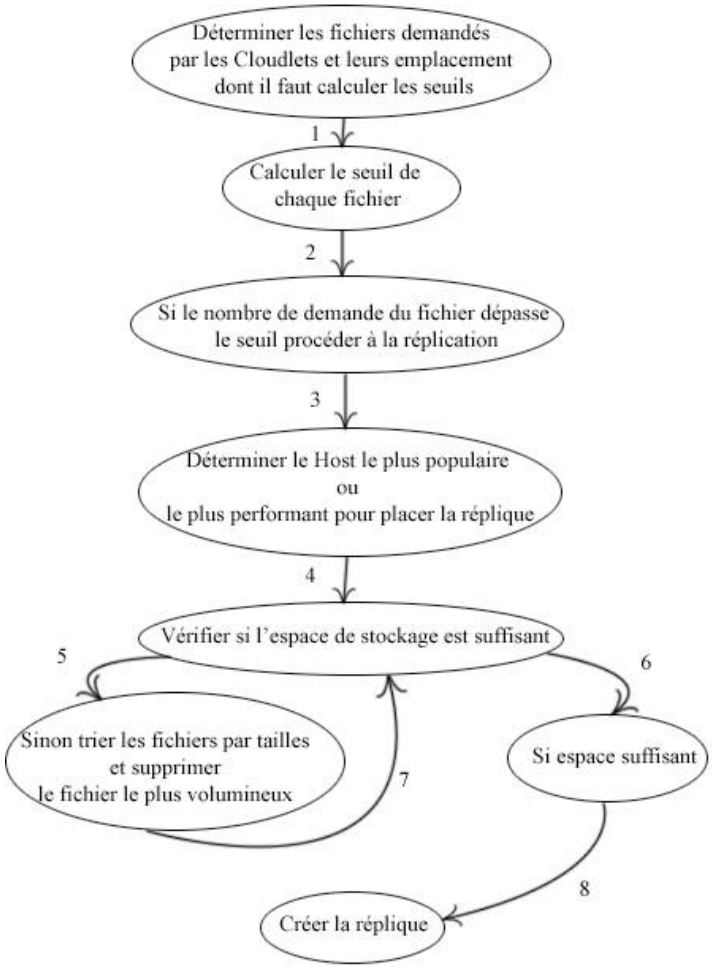


FIGURE 3.3 – Les étapes d'algorithme

3.4 Exemple de démonstration

Le but de cette partie est de donner une petite démonstration de l'algorithme proposé.

Nous avons deux cas :

1- Fichier se trouvant dans le même Data Center.

2- Fichier se trouvant dans un autre Data Center.

On commence par le premier cas.

Afin de simplifier les calculs, nous avons simplifié la topologie du Cloud comme le montre le schéma suivant :

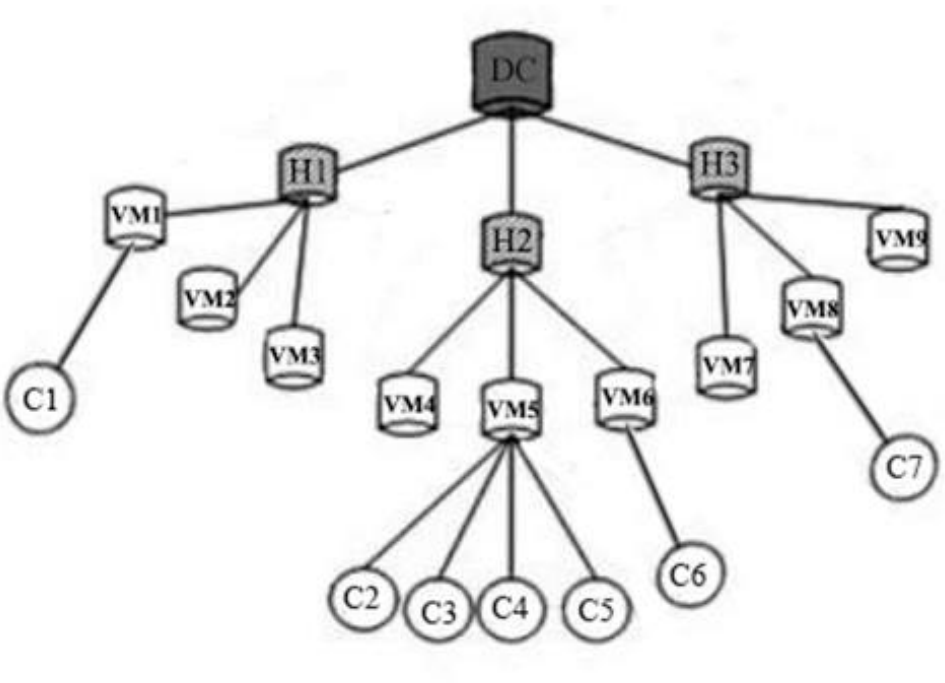


FIGURE 3.4 – Topologie du Cloud premier cas.

Dans le schéma proposé, nous avons 7 Cloudlets , qui envoient des requêtes à une donnée d qui se trouve initialement dans le Data Center DC. Initialement, prenons les données suivantes :

On suppose que toutes les Cloudlets demandent le même (F1).
Td = 30 Mo
T_{req} = 6 Ko, en supposant que les requêtes des Cloudlets ont la même taille.
T_{res1} = 2 Mo, taille de la réponse du Cloudlets C1
T_{res2} = 1,5 Mo, taille de la réponse du Cloudlets C2
T_{res3} = 3 Mo, taille de la réponse du Cloudlets C3
T_{res4} = 2 Mo, taille de la réponse du Cloudlets C4
T_{res5} = 1,25 Mo, taille de la réponse du Cloudlets C5
T_{res6} = 1 Mo, taille de la réponse du Cloudlets C6
T_{res7} = 1,5 Mo, taille de la réponse du Cloudlets C7
BW_n^{n+1} = 12 Ko /ms, largeur de la bande passante entre deux niveaux qui se suit.
La première phase de l'algorithme consiste à calculer les coûts et nombres d'accès au niveau de chaque nœud.

$C_{acc}(C1) = (6 * [\frac{1}{12}+\frac{1}{12}] + 2048 * [\frac{1}{12}+\frac{1}{12}]) = 342{,}33$ ms
$C_{acc}(C2) = (6 * [\frac{1}{12}+\frac{1}{12}] + 1536 * [\frac{1}{12}+\frac{1}{12}]) = 257$ ms
$C_{acc}(C3) = (6 * [\frac{1}{12}+\frac{1}{12}] + 3072 * [\frac{1}{12}+\frac{1}{12}]) = 513$ ms
$C_{acc}(C4) = (6 * [\frac{1}{12}+\frac{1}{12}] + 2048 * [\frac{1}{12}+\frac{1}{12}]) = 342{,}33$ ms
$C_{acc}(C5) = (6 * [\frac{1}{12}+\frac{1}{12}] + 1280 * [\frac{1}{12}+\frac{1}{12}]) = 214{,}33$ ms
$C_{acc}(C6) = (6 * [\frac{1}{12}+\frac{1}{12}] + 1024 * [\frac{1}{12}+\frac{1}{12}]) = 171{,}66$ ms
$C_{acc}(C7) = (6 * [\frac{1}{12}+\frac{1}{12}] + 1536 * [\frac{1}{12}+\frac{1}{12}]) = 257$ ms

Sur la base des nombres et coûts d'accès des Cloudlets nous calculons ceux des nœuds pères ; on aura le tableau suivant :

nœud	**Cloudlets(CA,C_{acc})**
H1	(1 ; 342,33)
H2	(5 ; 1498,32)
H3	(1 ; 257)
DC	(7 ; 2097,65)

TABLE 3.2 – Calcul des nombres et coûts d'accès des nœuds

Passons à la deuxième phase de l'algorithme qui consiste à calculer le seuil :

$$S'_0 = \frac{(30*\frac{1}{12})+C^1_{acc}}{2097{,}65} = \frac{2560+C^1_{acc}}{2097{,}65}$$

Nous avons besoin de calculer C^1_{acc} , qui est le coût du nouvel accès au niveau 1 où se trouvera la donnée d.

$C^1_{acc} = [(6 *\frac{1}{12})+(2048 *\frac{1}{12})] + [(6 *\frac{1}{12})+(1536 *\frac{1}{12})] + [(6 *\frac{1}{12})+(3072 *\frac{1}{12})] + [(6 *\frac{1}{12})+(2048 *\frac{1}{12})] + [(6 *\frac{1}{12})+(1280 *\frac{1}{12})] +[(6 *\frac{1}{12})+(1024 *\frac{1}{12})] + [(6 *\frac{1}{12})+(1536 *\frac{1}{12})]$

$C^1_{acc} = 171{,}16 + 128{,}5 + 256{,}5 + 171{,}16 + 107{,}16 + 85{,}83 + 128{,}5$
$= 1048{,}81$

$$S'_0 = \frac{2560+1048{,}81}{2097{,}65} = 1{,}72$$

$$S_0 = 2$$

Le placement de la réplique se fait au niveau du Host dont les requêtes proviennent le plus ou bien au niveau du Host le plus performant, donc dans cet exemple c'est : H2
Deuxième cas : Fichier se trouvant dans un autre Data Center.

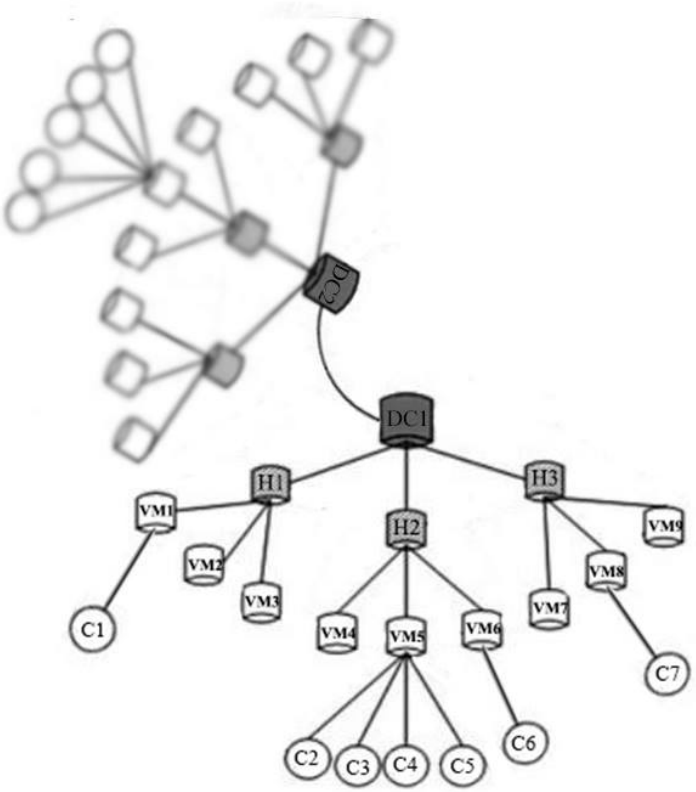


FIGURE 3.5 – Topologie du Cloud deuxiéme cas.

Supposons maintenant que les Cloudlets demandent le fichier F2 qui se trouve dans le Data Center 2 DC2.
Dans ce cas là il y a le niveau entre les DC qui va être pris en considération et la bande passante entre les DC sera inclus dans les calculs.

Td = Taille du fichier F2 = 30 Mo.

$C_{acc}(C1) = (6 * [\frac{1}{12}+\frac{1}{12}+\frac{1}{12}] + 2048 * [\frac{1}{12}+\frac{1}{12}+\frac{1}{12}]) = 513{,}5$ ms

$C_{acc}(C2) = (6 * [\frac{1}{12}+\frac{1}{12}+\frac{1}{12}] + 1536 * [\frac{1}{12}+\frac{1}{12}+\frac{1}{12}]) = 385{,}5$ ms

$C_{acc}(C3) = (6 * [\frac{1}{12}+\frac{1}{12}+\frac{1}{12}] + 3072 * [\frac{1}{12}+\frac{1}{12}+\frac{1}{12}]) = 769{,}5$ ms

$C_{acc}(C4) = (6 * [\frac{1}{12}+\frac{1}{12}+\frac{1}{12}] + 2048 * [\frac{1}{12}+\frac{1}{12}+\frac{1}{12}]) = 513{,}5$ ms

$C_{acc}(C5) = (6 * [\frac{1}{12}+\frac{1}{12}+\frac{1}{12}] + 1280 * [\frac{1}{12}+\frac{1}{12}+\frac{1}{12}]) = 321{,}5$ ms

$C_{acc}(C6) = (6 * [\frac{1}{12}+\frac{1}{12}+\frac{1}{12}] + 1024 * [\frac{1}{12}+\frac{1}{12}+\frac{1}{12}]) = 257{,}5$ ms
$C_{acc}(C7) = (6 * [\frac{1}{12}+\frac{1}{12}+\frac{1}{12}] + 1536 * [\frac{1}{12}+\frac{1}{12}+\frac{1}{12}]) = 385{,}5$ ms

L'agrégation des nombres et coûts d'accès donne le tableau suivant :

nœud	**Cloudlets(CA,C_{acc})**
H1	(1 ; 513,5)
H2	(5 ; 2247,5)
H3	(1 ; 385,5)
DC1	(7 ; 3146,5)
DC2	(7 ; 3146,5)

TABLE 3.3 – Calcul des nombres et coûts d'accès des nœuds, la donnée se trouve dans DC2

Passons au calcul du seuil :

$$S'_1 = \frac{(30*\frac{1}{12})+C^2_{acc}}{3146{,}5}$$

$C^2_{acc} = [6 * (\frac{1}{12}+\frac{1}{12})+(2048 *(\frac{1}{12}+\frac{1}{12})] + [6 * (\frac{1}{12}+\frac{1}{12})+(1536 *(\frac{1}{12}+\frac{1}{12})] + [6 * (\frac{1}{12}+\frac{1}{12})+(3072 *(\frac{1}{12}+\frac{1}{12})] + [6 * (\frac{1}{12}+\frac{1}{12})+(2048 *(\frac{1}{12}+\frac{1}{12})] + [6 * (\frac{1}{12}+\frac{1}{12})+(1280 *(\frac{1}{12}+\frac{1}{12})] + [6 * (\frac{1}{12}+\frac{1}{12})+(1024 *(\frac{1}{12}+\frac{1}{12})] + [6 * (\frac{1}{12}+\frac{1}{12})+(1536 *(\frac{1}{12}+\frac{1}{12})] = 2097{,}65$ ms

$$S'_1 = \frac{2560+2097{,}65}{3146{,}5} = 1{,}48$$
$$S_1 = 2$$

Rappelons qu'on n'a pas à calculer les seuils des autres nœuds qui se trouvent au dernier niveau intermédiaire.
Le placement de la réplique se fait au niveau du DC donc dans cet exemple c'est : DC1

CHAPITRE IV :

Implémentation

Chapitre 4

Ce chapitre est consacré à la réalisation et la concrétisation de notre approche proposée, qui consistent à la réplication des données dans les environnements de Cloud Computing. Dans un premier temps, nous présentons l'environnement de notre travail, puis nous définissons les différents services du simulateur CloudSim ainsi l'extension que nous avons réalisé pour intégrer la gestion des données, ensuite nous décrivons quelques interfaces graphiques, et finalement nous présentons une série de simulations et leurs interprétations pour mettre en évidence notre proposition.

4.1 Langage et environnement de développement

Nous avons utilisé l'environnement de développement netbeans.

4.1.1 Langage de programmation Java

Java est un langage de programmation à usage général, évolué et orienté objet dont la syntaxe est proche du C++. Il a été mis au point en 1991 par la firme Sun Microsystems [33]. Il s'agissait de concevoir un langage bien adapté aux environnements de travail en réseau et capable de gérer des informations de nature variées (données numériques, informations sonores et graphiques). Java est devenu aujourd'hui une direction incontournable dans le monde de la programmation, parmi les différentes caractéristiques qui sont attribuées à son succès :

- L'indépendance de toute plate-forme : le code reste indépendant de la machine sur laquelle il s'exécute. Il est possible d'exécuter des programmes Java sur tous les environnements qui possèdent une Java Virtual Machine.
- Java est également portable, permettant à la simulation d'être distribuée facilement sans avoir à recompiler le code pour les différents systèmes.
- Le code est structuré dans plusieurs classes, dont chacune traite une partie différente

de la simulation.
- Il assure la gestion de la mémoire.
- Java est multitâches : il permet l'utilisation de Threads qui sont des unités d'exécution isolées.
Aussi, une des principales raisons de ce choix est que le simulateur CloudSim est développé avec ce langage.

4.1.2 Environnements de développement

Netbeans [34] est l'environnement de Développement Intégré (EDI) supporté par SUN. Il est particulièrement bien adapté pour le développement d'applications WEB. Il remplace l'IDE Java Studio Creator.
C'est un IDE moderne offrant un éditeur avec des codes couleurs et un ensemble de signes, des modèles de projets multi-langage et de différents types (application indépendante, distribuée, plugin, mobiles, ...), le refactoring, l'éditeur graphique d'interfaces et de pages web pour supporter le programmeur dans son travail. Il permet d'accéder rapidement à la documentation détaillée, de naviguer dans les sources et de faire des recherches d'usage des classes, méthodes et propriétés. Netbeans indique à l'utilisateur les erreurs et fait des propositions pour y remédier. Un déboggeur permet l'exécution pas à pas. Un suivi des ressources utilisées (cpu, mémoire) par le logiciel développé peut être fait via un profiler. Un framework de test unitaire tel que Junit Fiche Junit peut être utilisé.
L'EDI NetBeans fournit des outils pour construire tous les composants Java EE, ce qui inclut les Enterprise Java Beans (EJBs), les pages web, les servlets, et les services web. Il intègre le serveur d'application Glassfish, ce qui permet de facilement développer des EJB et de les déployer.
Il intègre la norme WebService JAX-WS. Il est aisé de lier un WS avec un EJB pour faire son implémentation. CloudSim Objectif principal de simulateur CloudSim est de fournir un cadre de simulation généralisé et extensible qui permet la modélisation, la simulation et l'expérimentation des nouvelles infrastructures du Cloud Computing et les services d'application, permettant aux utilisateurs de se concentrer sur des questions de conception du système qu'ils veulent étudier, sans être préoccupé aux détails relatifs aux services et infrastructures Cloud.
Nous avons utilisé pour la réalisation de notre travail la version du simulateur Cloudsim 3.0.2 [35]

4.1.3 Architecture de CloudSim

La structure logicielle de CloudSim et ses composants est représentée par une architecture en couches comme il est montré par la Figure 4.1. Les premières version de CloudSim utilise SimJava, un moteur de simulation d'événement discret qui met en œuvre les principales fonctionnalités requises pour des structures de simulation de haut niveau comme la formation d'une file d'attente et le traitement d'événements, la création de composants système (les services, les machines (Host), le centre de données (Datacenter), le courtier (Broker), les machines virtuelles), la communication entre les composants et la gestion de l'horloge de simulation. Cependant, dans la version actuelle, la couche SimJava a été supprimée afin de permettre à certaines opérations avancées qui ne sont pas pris en charge par celle-ci.

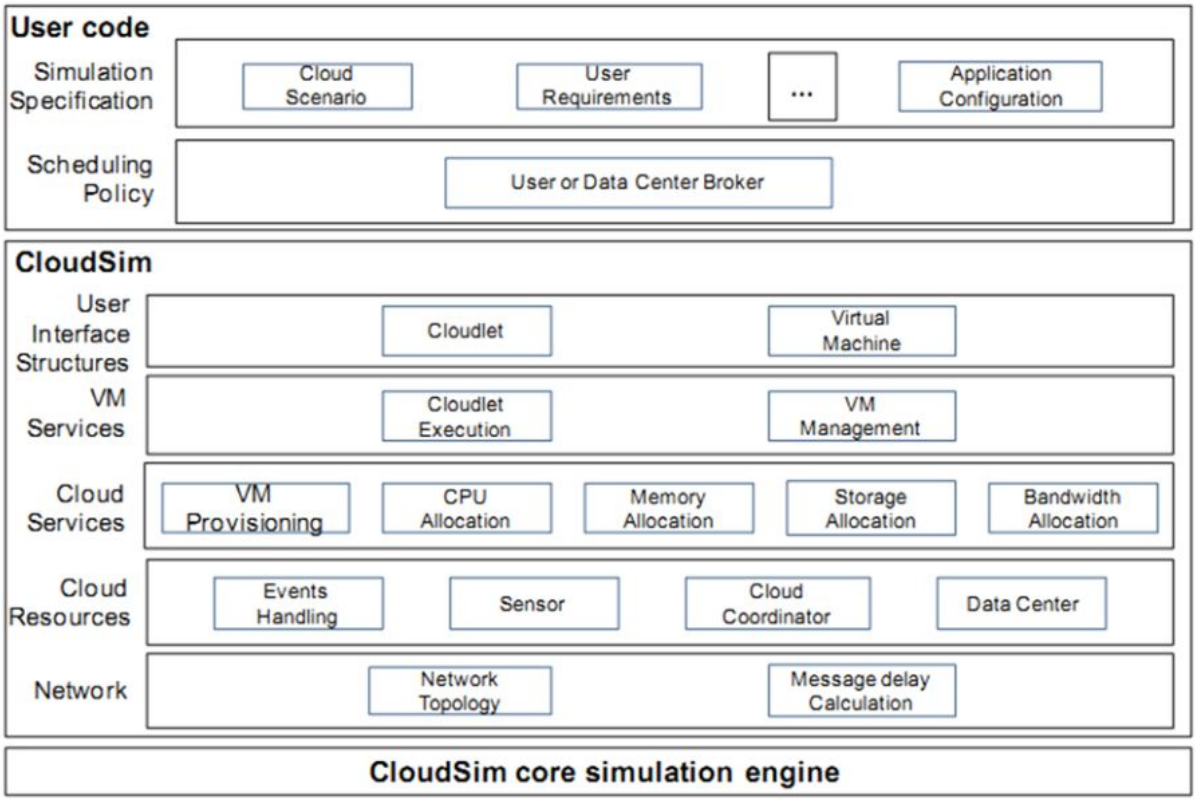


FIGURE 4.1 – Architecture de CloudSim

4.2 Description du fonctionnement de notre application

4.2.1 Interface principale

Au lancement de notre application l'interface qui s'affiche est montré dans la Figure 4.2 cette interface contient deux bouton le 1er permet de lancer la configuration pour la simulation le deuxième affiche quelques information a propos de l'application.

FIGURE 4.2 – Accueil

4.2.2 Configuration des paramètres de simulation

L'onglet DC affiche l'interface pour configurer les DCs (le nombre de DC, configurations des Hosts, le nombre et la taille des fichiers, la vitesse de chaque CPU, le coût de traitement, la taille de la mémoire, le coût de la mémoire, l'espace de stockage du serveur de données, la bande passante, le coût de stockage et le coût de la bande passante) et le choix de la stratégie avec laquelle fonctionnent les DCs.

Remarque : avant de créer les DCs, Il faut d'abord configurer les Hosts (Figure 4.4).

FIGURE 4.3 – Onglet configuration des DC

La fenêtre de configuration des Hosts permet d'entrer le nombre de Pes, MIPS, la RAM, l'espace de stockage et la bande passante.

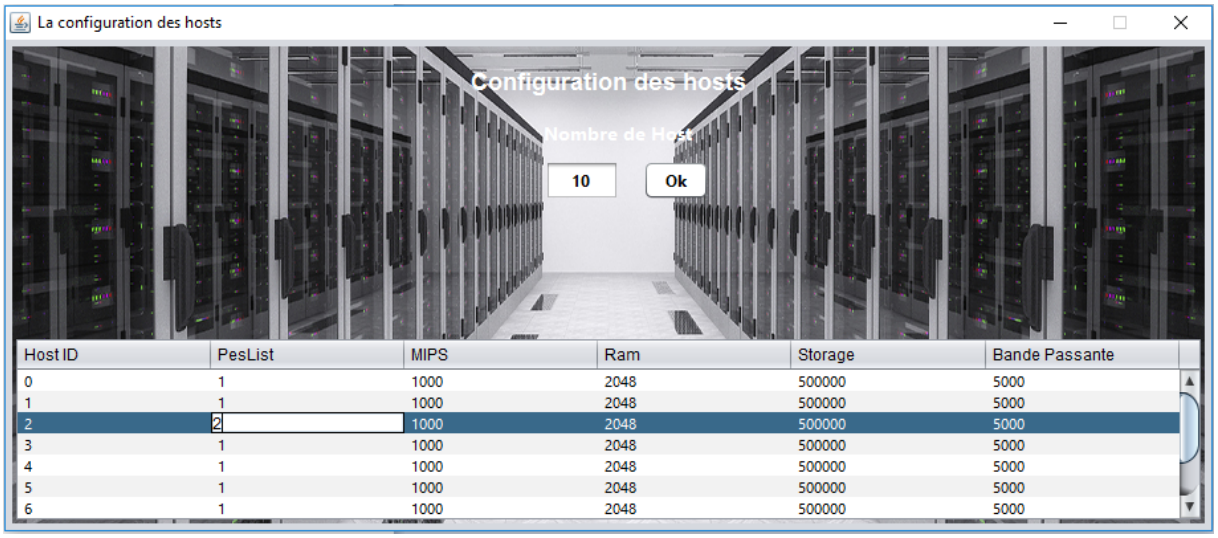


FIGURE 4.4 – Interface configuration des Hosts

L'onglet VM affiche l'interface pour configurer les VMs (le nombre de VMs, MIPS, Size, la RAM, la bande passante, le nombre de Pes nécessaire et le nom de la VM).

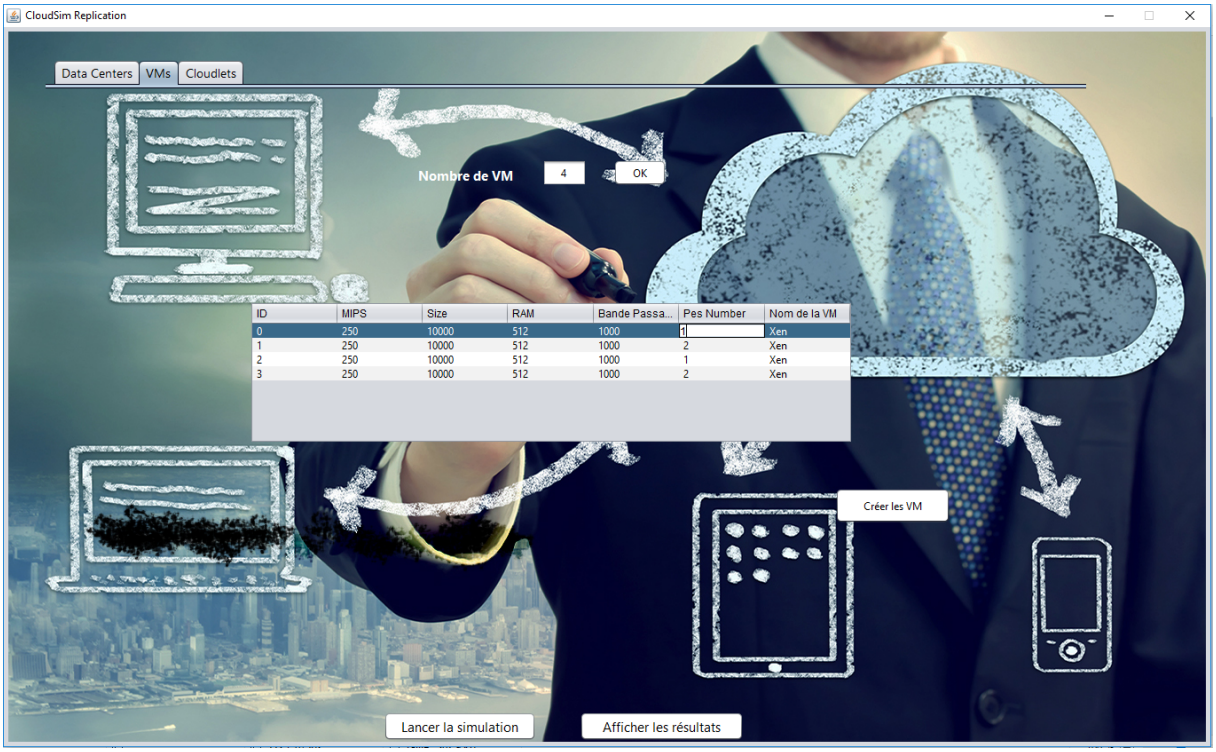


FIGURE 4.5 – Onglet configuration des VMs

L'onglet Cloudlet affiche l'interface pour configurer les Cloudlets : Length, File Size, Output Size, les fichiers demandé par la cloudlet (chaque fichier séparé entre le suivant par un point virgule ;) et l'ID de la VM a la quelle on veut affecter la Cloudlet.

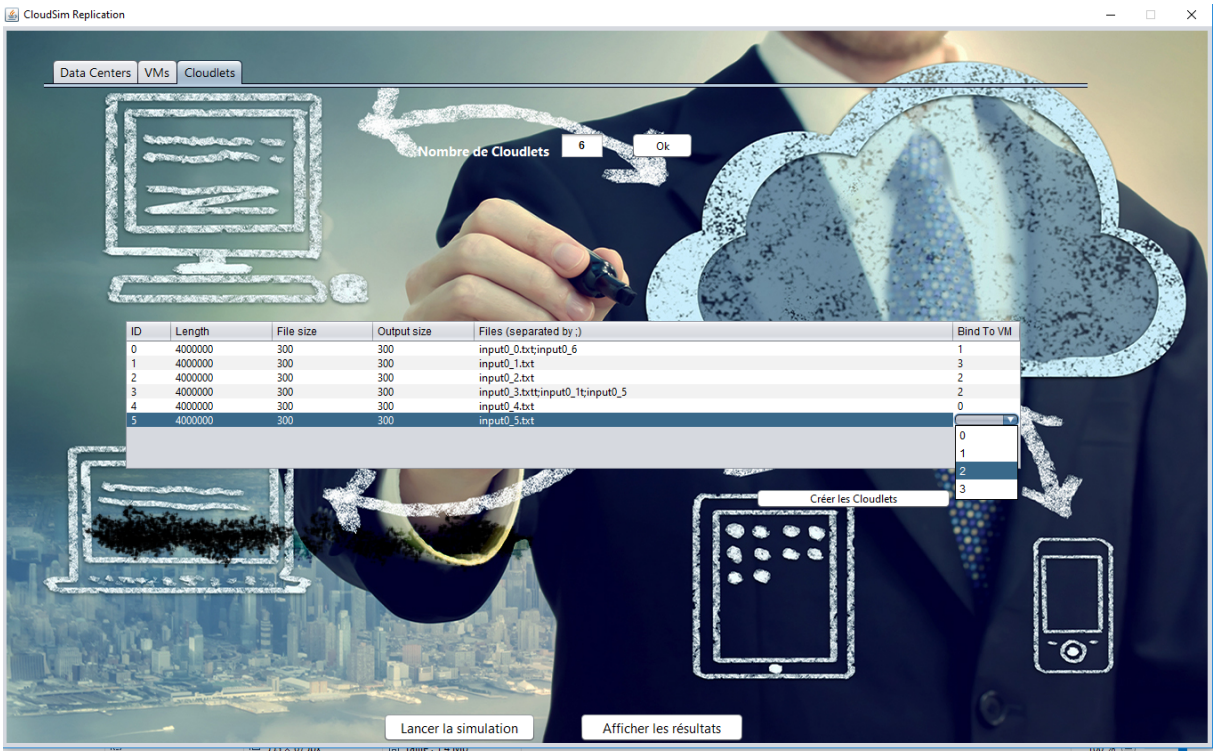


FIGURE 4.6 – Onglet configuration des Cloudlets

Pour lancer la simulation on clique sur Lancer la simulation et puis Afficher les résultats pour voir les résultats de la simulation.

FIGURE 4.7 – Fenêtre des résultats

4.3 Résultats expérimentaux

Nous allons effectué quelques expérience (Expérience 1, 2 et 3) afin de comparer notre stratégie avec une autre et déterminer le temps d'exécution moyen des Cloudlets tout en changeant quelques paramètres (Nombre DC, taille des fichiers et nombre de Cloudlets), la stratégie avec la quelle on va comparer notre approche est nommé stratégie RANDOM et a pour principe de créer des répliques aléatoirement dans le Cloud. Puis nous allons voir le nombre de réplique créer (Expérience 5, 4 et 6).
Nous avons choisis de créer 4 Hosts et 8 VMs pour cette expérience.

4.3.1 Expérience 1

A travers cette expérience, notre objectif était de voir comment l'augmentation du nombre de Datacenter pouvait avoir un effet sur le temps d'exécution des cloudlets. Dans Cette simulation nous avons varié le nombre de Datacenter par pas de 1 ; nous avons fixé le nombre de cloudlets à 20 ;

	1	2	3	4
Stratégie Randam	233577	234308	236242	239477
Stratégie Dynamique	229953	230100	230340	230500

TABLE 4.1 – Impact du nombre de DC sur le temps d'exécution

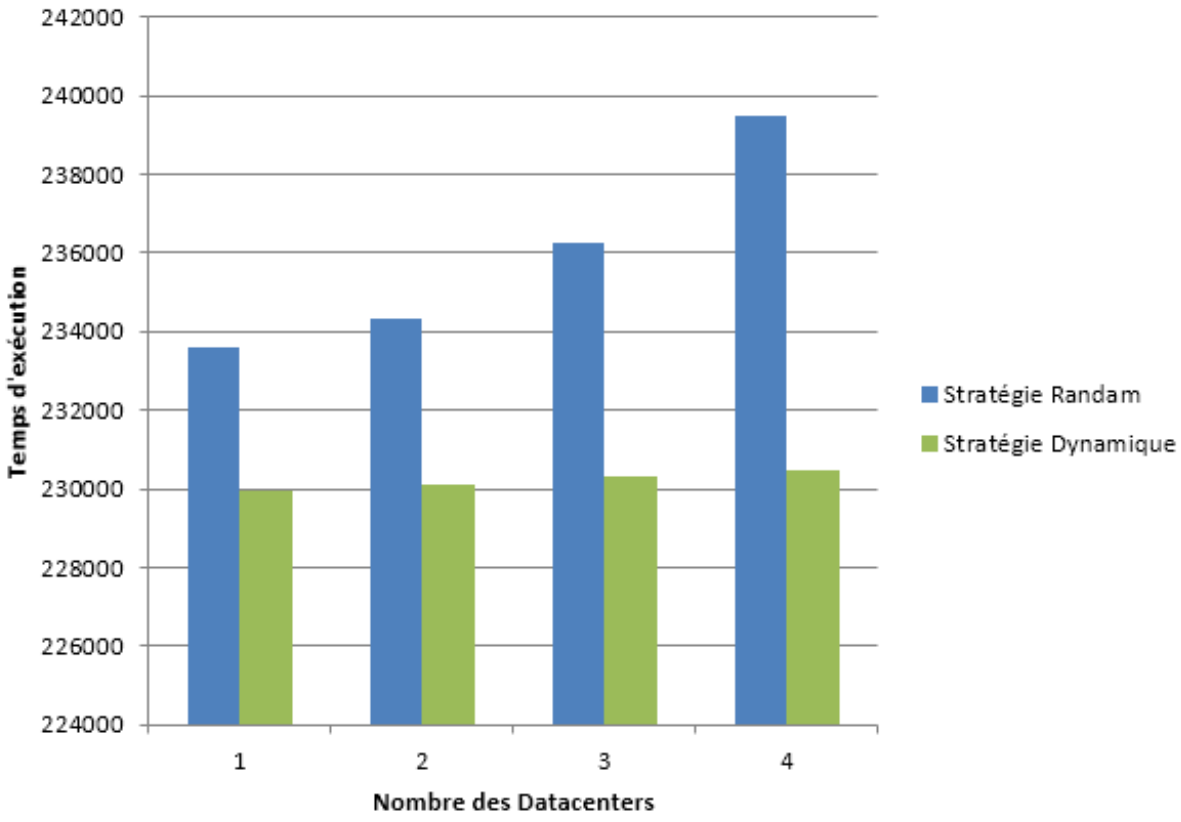


FIGURE 4.8 – Impact du nombre de DC sur le temps d'exécution

Le tableau 4.1 compare entre les deux approches selon le nombre de Datacenter, où nous pouvons déduire que le temps de réponse de notre approche reste presque constant, car notre stratégie va crées des réplique dans chaque datacenter, la différence dans le temps peut être expliqué par le temps du 1er transfert de la donnée de datacenter qui contient la donnée originale vers le nouveau datacenter ; Dans l'autre approche, nous remarquons que le temps réponse s'augmente lorsque le nombre de datacenter augmente ; parce que la stratégie randam crée de manière aléatoire les répliques, donc on peut avoir

dans un datacenter plusieurs répliques et dans un autre datacenter aucune réplique.

4.3.2 Expérience 2

Dans cette simulation, nous avons créé deux Data Center contenant 10 hôsts hétérogène, Chaque hôte posséde 1 processeur avec une vitesse variante en MIPS entre (1000, 2000), bande passante entre(100,1000). Cette simulation consiste à varier la taille de la donnée (200, 500, 1000, 1500)

	200	500	1000	1500
Stratégie Randam	233577	237913	239238	242103
Stratégie Dynamique	229953	231233	232122	232508

TABLE 4.2 – Impact de la taille du fichier sur le temps d'exécution

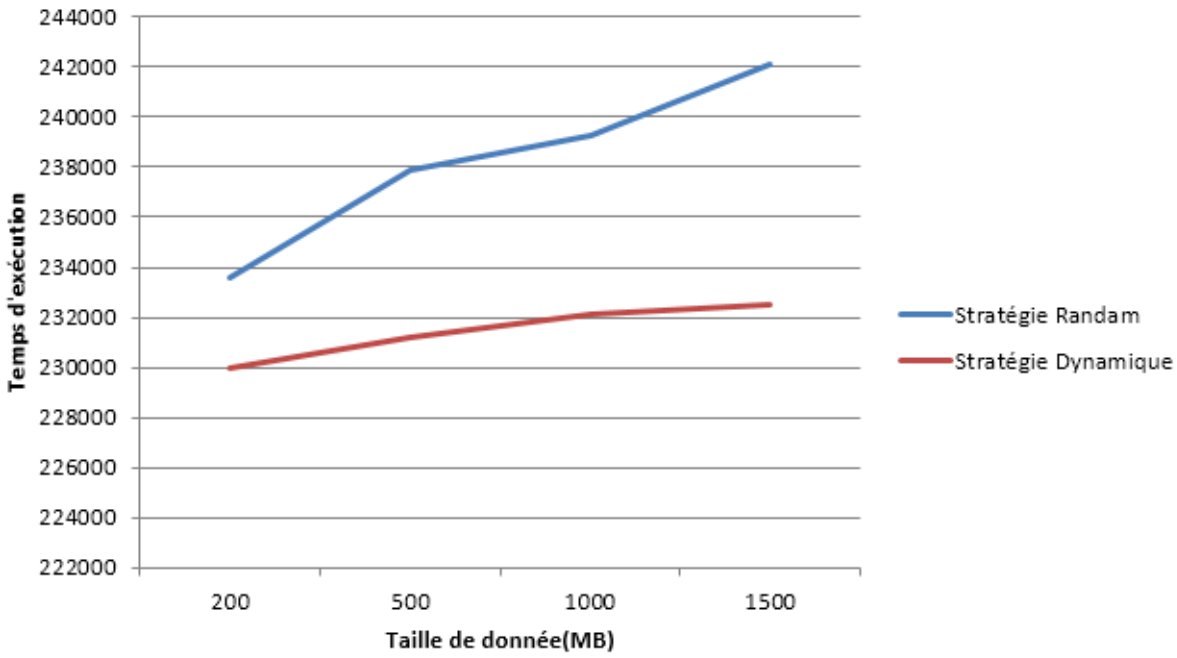


FIGURE 4.9 – Impact de la taille du fichier sur le temps d'exécution

La figure montre de manière évidente, la supériorité de l'approche dynamique par rapport à l'approche random. Nous remarquons que le temps d'exécution des cloudlets

dans la stratégie random augmente avec l'augmentation de la taille de la donnée ; La taille des fichiers détermine le temps de transfert des fichiers, la stratégie Random place les réplique aléatoirement, donc elle peut les placer dans des nœuds qui ne sont pas proche des VMs exécutant les Cloudlets, ou dans des nœuds moins populaire par conséquent le temps de transfert va être important. et ça va affecter le temps d'exécution des Cloudlets qui sera grand aussi. Par contre notre stratégie qui place les répliques aux bons endroits, a permis de diminuer le temps de transfert des fichiers ; et l'augmentation dans le temps d'exécution totale est moins significative par rapport à l'autre stratégie ;

4.3.3 Expérience 3

Dans cette simulation, nous avons créé deux Data Center contenant 10 hôsts hétérogène, chaque hôte posséde 1 processeur avec une vitesse variante en MIPS entre (1000, 2000), bande passante entre (100,1000), la taille de la donnée est 200 MB. Cette simulation consiste à varier le nombre de cloudlet par pas de 20 et voir l'impact sur le temps d'exécution ;

	20	40	60	80
Stratégie Randam	233577	239913	242538	248652
Stratégie Dynamique	229953	231233	232122	232508

TABLE 4.3 – Impact du nombre de Cloudlets sur le temps d'exécution

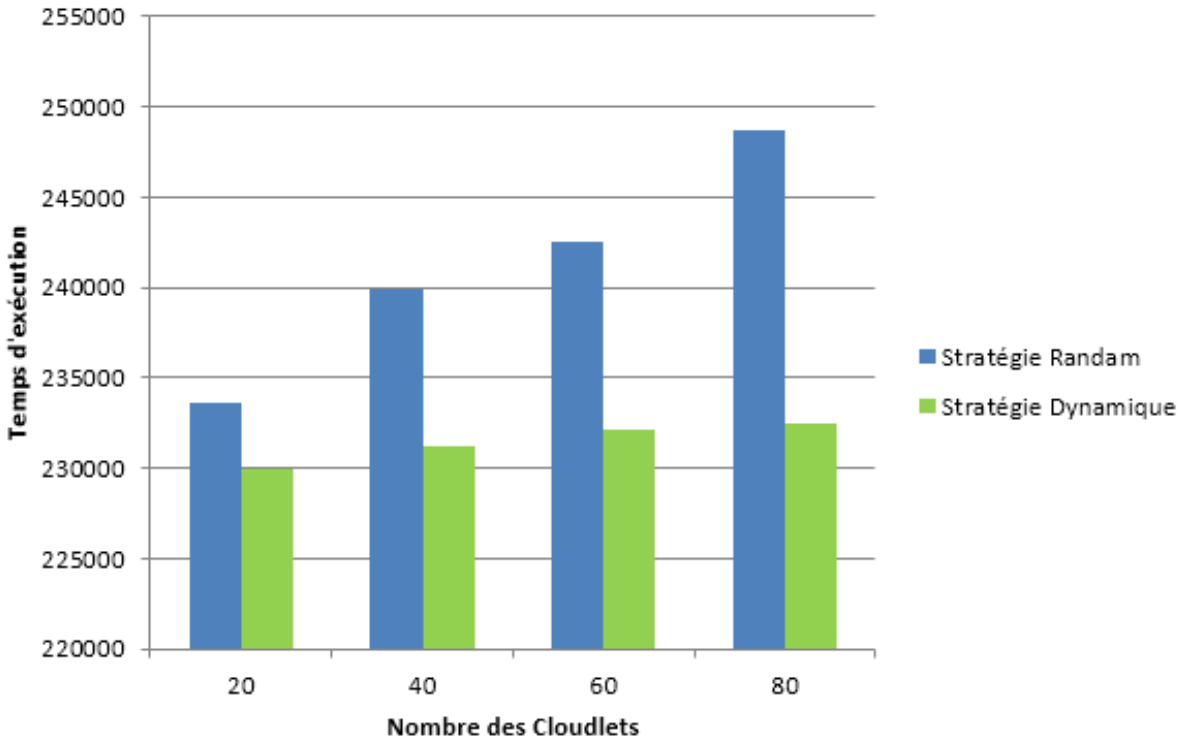


FIGURE 4.10 – Impact du nombre de Cloudlets sur le temps d'exécution

Nous remarquons une diminution significative dans le temps d'exécution des cloudlets avec l'approche proposée par rapport à la stratégie de réplication statique (Random). Parce que l'approche proposé s'adapte selon les requêtes, des nouvelle réplique seront créés si le seuil est dépassé ; Les réplique sont placées dans les hosts les plus populaires, c'est à dire les hosts qui subissent plus de requêtes. Ce qui diminue le temps de transfert et donc le temps d'exécution totale ;

4.3.4 Expérience 4

Dans cette expérience, nous allons augmenter le nombre de Data Centers et on verra l'impact sur le nombre de réplique.

NB : Une Cloudlet ou deux au maximum on étaient exécuté dans chaque nouveau Data Center.

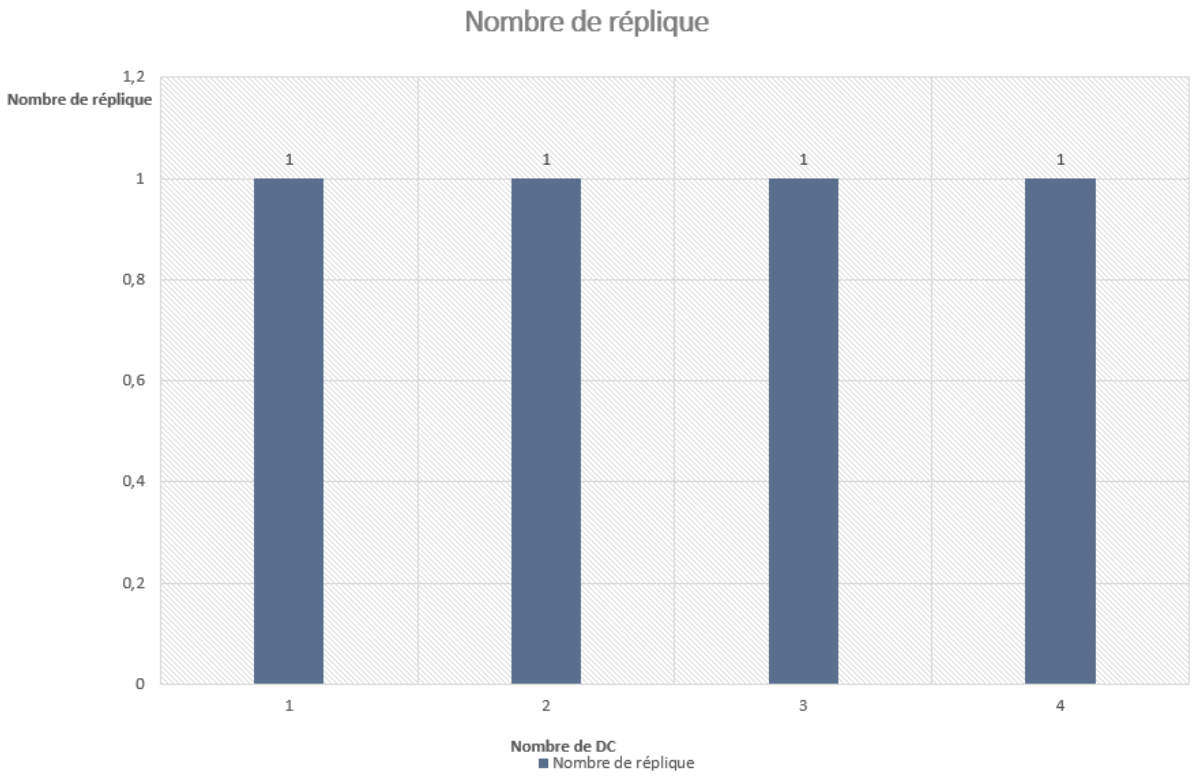


FIGURE 4.11 – Impact du nombre de DC sur le nombre de réplique

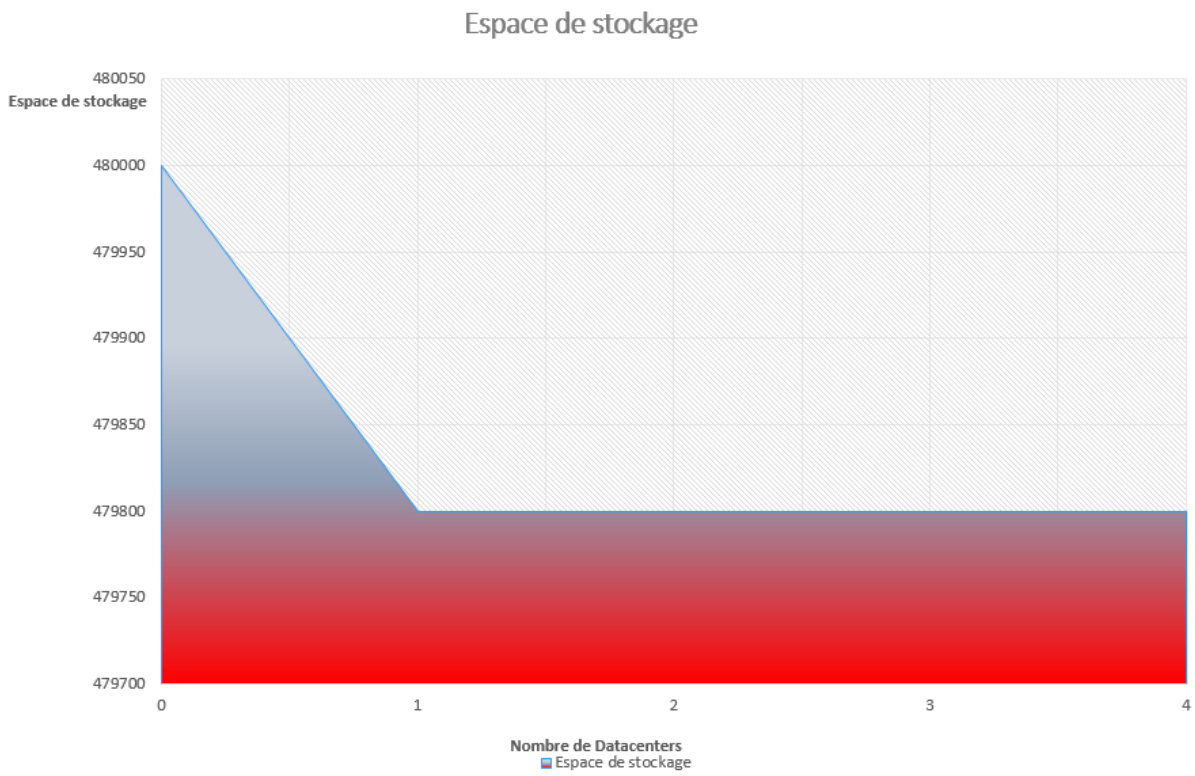


FIGURE 4.12 – Espace de stockage d'un Host après la création de réplique

4.4 Interprétation des résultats :

Nous remarquons une diminution significative dans le temps d'exécution des cloudlets avec l'approche proposée par rapport au stratégie de réplication statique. parce que l'approche proposé s'adapte selon les requêtes, et les réplique sont placées dans les hosts les plus populaires, c'est à dire les hosts qui subissent plus de requêtes.

4.5 Conclusion

Dans ce chapitre nous avons expliquer le fonctionnement de notre stratégie dans notre application et nous avons démontrer que notre stratégie lorsque le nombre de Data Centers et de fichiers est élevé permet une exécution des Cloudlets plus rapidement que la stratégie Random, par contre notre stratégie à certain cas ne s'avère pas plus rapide lorsque le nombre de Cloudlets est élevé et cela aussi dépends de la topologie (répartition de tout les noeuds).

Conclusion générale

Le cloud computing est en pleine expansion et tend à s'imposer comme un des paradigmes dominants dans l'univers informatique. Les infrastructures proposant des services de cloud computing deviennent donc de plus en plus nombreuses, et de plus en plus complexes pour répondre à cette demande croissante de services décentralisés. Aujourdhui on n'a pas un problème de stockage mais on a un problème de gestion et de récupération de l'information avec le minimum temps. Il faut donc concevoir des techniques et outils afin de répondre à ces nouveaux besoins de gestion.

La réplication de données est une technique qui permet de régler en général certains problèmes dans le Cloud tels que les problèmes de la disponibilité des données à chaque instant.

Cette technique consiste à répliquer les données dans les nœuds selon une disponibilité exigé par l'utilisateur dont le but d'améliorer la disponibilité et réduire le temps de réponse selon certains critères.

Pour bénéficier au maximum du gain que peuvent apporter la réplication. Au cours de ce projet, nous avons défini et implémenter un algorithme de réplication et placement de données dans l'environnement de Cloud Computing. Qui permet de mettre les bonnes données au bon endroit et au bon moment. Ce qui permet d'économiser le temps et l'argent tout en augmentant la protection des données.

Afin de valider et d'évaluer l'approche proposée, nous avons réalisé plusieurs séries d'expérimentations en faisant varier plusieurs paramètres. Les résultats montrent de manière évidente, la supériorité de l'approche proposée par rapport à l'approche statique. La stratégie proposée nous a permis de réduire de façon significative le temps d'exécution des Cloudlets.

Résumé

Le cloud computing ou " informatique dans les nuages " nous permet d'accéder à toutes nos applications et services de partout et à tout moment via l'Internet. Le Cloud permet une réduction des coûts pour les entreprises, ce qui est plus intéressent que d'acheter des ordinateurs plus rapides ou meilleure en termes de mémoire et espace de stockage, tout ordinateur ou smart phone peut accéder aux services du cloud à l'aide d'un navigateur ou d'une application. Aussi les entreprises n'ont plus besoin d'acheter des équipements tels que des serveurs coûteux afin de fournir un service e-mail pour leurs employés, ou de grandes unités de stockage pour effectuer des sauvegardes de données et d'informations pour la société. C'est pourquoi au cours de ce travail, nous nous sommes intéressés à la gestion des données dans le Cloud Computing, où nous avons présenté notre approche de réplication qui permet de mettre les bonnes données au bon endroit et au bon moment. L'approche proposée permet d'économiser le temps et l'argent tout en augmentant la protection des données.

The Cloud computing allows to access our applications and services from anywhere, anytime from the web ; The Cloud allows to reduce costs for companies, which is better than buying better computers in term of speed, memory and storage space, any PC or smart phone can access to the cloud services using a navigator or an application. Also the companies aren't required to buy equipment like expensive servers to provide e-mail services for the workers, or high storage units to save all data and information for society. Therefore in this work, we focused management Data in the Cloud computing, where we presented our replication approach that can put the right data in the right place at the right time. This can save time and money while increasing data protection.

الحوسبة السحابية أو الحوسبة في السحاب تسمح لنا بالوصول إلى كافة التطبيقات والخدمات لدينا في أي مكان وفي أي وقت عن طريق شبكة الإنترنت. الحوسبة السحابية تقلل التكاليف للشركات، وهذا شيء مهم جدا وأهم من شراء أجهزة كمبيوتر أسرع وأفضل من حيث الذاكرة ومساحة التخزين، أي جهاز كمبيوتر أو هاتف ذكي يمكنه الوصول إلى الخدمات السحابية باستخدام متصفح أو التطبيق وبما أن الشركات لم تعد بحاجة لشراء معدات باهظة الثمن مثل الخوادم لتقديم خدمة البريد الإلكتروني لموظفيها، أو وحدات تخزين كبيرة لتنفيذ عمليات النسخ الاحتياطي للبيانات والمعلومات للمجتمع. وبذلك وجهنا اهتمامنا في مجال إدارة البيانات في السحابة، حيث قدمنا استراتيجية حول نسخ البيانات وأين يتم وضع البيانات الصحيحة في المكان المناسب في الوقت المناسب. الاستراتيجية المقترحة توفر الوقت والمال مع زيادة لحماية البيانات.

Bibliographie

[1] S. Drapeau. RS2.7 : Canevas Adaptable de services de duplication. PhD thesis, Institut National Polytechnique de Grenoble, France, Juin 2003.

[2] N. Hadi. : Réplication et ordonnancement dans les grilles de calcul, Une approche basée sur les méthodes d'aide à la décision multicritères. PhD thesis, Département informatique, Faculté des sciences, Université Oran, juin 2013.

[3] I. SARR. : Routage des Transactions dans les Bases de Données à Large Echelle. PhD thesis, La boratoire d'informatique Paris 6, Université de Pierre et Marie Curie, France, Octobre 2010. http ://download.intel.com/ press-room/pdf/computertrendsrelease.pdf.

[4] B. Meroufel : Tolérance aux pannes dans les grilles de données. Master's thesis, Département informatique, Faculté des sciences, Université Oran, Juin 2011.

[5] N. Grevet : Le cloud computing : évolution ou révolution. Mémoire de recherche, Aout 2009.

[6] Cloud computing. http :fr.wikipedia.org/wiki/Cloud-computing

[7] Syntec informatique. : Tout ce que vous devez savoir sur l'informatique dans le nuage. Le Livre Blanc du Cloud Computing. http ://journal-ntic.fr/wp-content/uploads/2011/06/Livre-blanc-cloudcomputing.pdf.

[8] I. Foster and C.Kesselman : The grid : blueprint for a new computing infrastructure. Morgan Kaufmann, 1999.

[9] L. Alvisi and K.Marzullo : Message logging : Pessimistic, optimistic, causal, and optimal. IEEE Transactions on Software Engineering, 24(2) pp. 149ñ159, 1998.

[10] Yahoo ! Inc., Yahoo ! and CRL to Collaborate on Cloud Computing Research, 2008.

[11] S. Warin : Le Cloud Computing. Réelle révolution ou simple évolution. Livre Blanc sur le Cloud Computing, Février 2011. http ://www.wygwam.com/documents/cloud-computing.pdf.

[12] Oracle White Paper in Enterprise Architecture-Architectural Strategies for Cloud Computing.

[13] W. Malvault : Vers une architecture pair-à-pair pour l'informatique dans le nuage. Thèse de Doctorat, Université de Grenoble, France, Octobre 2011.

[14] Amazon EC2, Amazon Elastic Compute Cloud, http ://aws.amazon.com/ec2/

[15] M. Armbrust ,A. Fox ,R. Griffith ,A. D. Joseph ,R. H. Katz ,A. Konwinski ,G. Lee , D. A. Patterson ,A. Rabkin ,I. Stoica and M. Zaharia : A view of cloud computing. Commun. ACM, 53(4) pp. 50-58, 2010.

[16] Salesforce. https ://www.salesforce.com/.

[17] Netsuite. http ://www.netsuite.com/.

[18] Google code. http ://code.google.com/.

[19] Windows azure. http ://www.microsoft.com/windowsazure/windowsazure/.

[20] Google app engine. http ://code.google.com/intl/fr-FR/appengine/.

[21] A. Lefort : Cloud Computing. Projet tutoré en licence professionnelle ASRALL, 2010.

[22] J. Anderson : Choisissez votre cloud, Août 2011. http ://www.thecloudadvantage.com/downloads/frFR/CloudPublicVsPrivate-PoV.pdf.

[23] L'avenir est au cloud computing. http ://www.cfo-news.com/L-avenir-est-aucloud-computing-a20143.html.

[24] I. Foster K.Ranganathan : "Identifying dynamic replication strategies for high performances data grids", Proceedings. 3 rd IEEE/ACM International Workshop of grid computing, London, UK, 2001, pp 75-76.

[25] I. Foster and C. Kesselman : "The Grid : Blueprint for a New Computing Infrastructure", Morgan Kaufmann, edition, 1998, San Francisco, USA.

[26] D.Yang et Al : "A Comparative study of Replicas Placement Strategies in Data Grids", Proceedings of Advances in Web and Network Technologies, and Information Management, Computer Sciences vol 4537,2007, pp 135-143.

[27] Y. Nemati et Al : "A novel data replication policy in data grid", Australian Journal of Basics and Applied Sciences, Vol.6, Numéro 7, 2012, pp 339-344.

[28] L. Allal and C. Dad : Gestion de la cohérence des répliques de données orientée QoS dans les Wireless Grid. Mémoire d'ingénieur d'état en informatique, Département d'informatique, Faculté des sciences, Université d'Oran, Algérie (Juin 2009).

[29] N. Belayachi and R. Behidji : Influence de l'équilibrage de charge sur la cohérence des répliques dans les grilles de données. Mémoire d'ingénieur d'état en informatique, Département d'informatique, Faculté des sciences, Université d'Oran, Algérie (Juin 2009).

[30] L. MOINE : La gestion et la sécurité dans une architecture de ressources de calcul distribuées sur l'Internet. Mémoire d'ingénieur c.n.a.m. en informatique, UREC (Unité Réseaux du CNRS), Centre d'enseignement de Grenoble, Grenoble Cedex 9, France (Juillet 2002).

[31] G. Oster : Réplication optimiste et cohérence des données dans les environnements collaboratifs répartis. PhD thesis, Université Henri Poincaré, Nancy 1, France, Novembre 2005.

[32] S. KOUIDRI : Gestion de la coherence des répliques tolérente aux fautes dans une grille de données. Master's thesis, Département informatique, Faculté des sciences, Université Oran, Juin 2011.

[33] http ://fr.wikipedia.org/wiki/Java(langage).

[34] https ://www.projet-plume.org/fiche/netbeans

[35] https ://github.com/Cloudslab/cloudsim/releases/tag/cloudsim-3.0.2 consulté le 20-11-2015

[36] F. Z.Bellounar : "Stratégies efficaces de réplication de données sur les grilles", Thèse de doctorat, Université d'Oran, Algérie, 2014

Table des figures

Liste des tableaux