République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université A/Mira de Béjaia
Faculté des Sciences Exactes
Département d'Informatique

MÉMOIRE DE MASTER RECHERCHE

En
Informatique

Option
Réseaux et Systèmes Distribués

Thème

Réplication des données dans le Cloud computing

Présenté par : Mlle FRISSOU Yasmine

Soutenu le .. Juin 2017 devant le jury composé de :

Président		Maître de conf. A	U. A/Mira Béjaia.
Rapporteur	Mme YAICI Malika	Maître de conf. A	U. A/Mira Béjaia.
Examinateur	M. AISSANI Sofiane	Maître de conf. B	U. A/Mira Béjaia.
Examinateur	M. SAADI Mestapha	Maître de conf. A	U. A/Mira Béjaia.
Président	M. "	Doctorante "LMD"	U. A/Mira Béjaia.

Béjaia, Juin 2016.

⁕ *Remerciements* ⁕

Louange à dieu, le miséricordieux, sans lui rien de tout cela n' aurai pu être.

Nous tenons à exprimer toute notre reconnaissance à notre Directeur de mémoire Madame Yaici Malika. nous la remercions de nous avoir encadré, orienté, aidé et conseillé.

Nous adressons nos sincères remerciements aux membres du jury d'avoir accepté d'évaluer ce travail. Leurs remarques et suggestion nous seront très utiles pour la finalisation de ce mémoire.

Nous remercions nos très chers parents, qui ont toujours été là pour nous, « vous avez tout sacrifié pour vos enfants n'épargnant ni santé ni efforts. Vous nous avez donné un magnifique modèle de labeur et de persévérance. Nous somme redevables d'une éducation dont nous sommes fières ».

Nous exprimons notre profonde gratitude à nos frères et soeurs qui nous ont soutenu et épaulé tout au long de la réalisation de ce mémoire.

Enfin, nous tenons à remercier nos amis, nos grandes familles et toute personne qui a participé de près ou de loin dans la réalisation de ce modeste travail.

⁕ *Dédicaces* ⁕

A mes très chers parents qui ont toujours été là pour moi, et qui m'ont donné un magnifique modèle de labeur et de persévérance. J'espère qu'ils trouveront dans ce travail toute ma reconnaissance et tout mon amour.
A la mémoire de mes grands-parents.
A mes chers frères et soeurs : Samia, Sabrine, Mohand.
A mon mari Chabane.
A mes tantes et à mes oncles.
A chaque cousins et cousines.
A mes amis et à tous ceux qui m'ont encouragés et soutenu .
Je dédie ce mémoire

Yasmine FRISSOU

⁂ *Dédicaces* ⁂

A mes chers parents, pour tous leurs sacrifices, leur amour, leur soutien et leurs prières tout au long de mes études.
A la mémoire de mon grand-père qui reste présent dans nos pensées et nos coeurs.
A ma très chère grand-mère pour son dévouement et son soutien.
A mon chère frère, Amine, pour son appui et son encouragement.
A mes oncles, mes tantes, mes cousins et mes cousines qui ont toujours été présents.
A toute ma famille et mes amis pour leurs soutien tout au long de mon parcours universitaire.
Que ce travail soit l'accomplissement de vos voeux, et le fruit de votre soutien indéfectible.
Merci d'être là pour moi.

Nassim HOUARI

Table des matières

Table des figures

Liste des tableaux

Liste des algorithmes

Notations et symboles

ACID **A**tomicité **C**ohérence **I**solation **D**urabilité.
BD **B**ase de **D**onnées .
CCMP **C**ommon **C**loud **M**anagement **P**latform.
DN **D**egré **N**égatif.
EC2 **E**lastic **C**ompute **C**loud.
FP **F**acteur **P**ositif.
FR **F**acteur de **R**eplication.
FSM **F**og **S**erver **M**anager.
HA **H**igh **A**vailability.
HAS **H**euristic **A**audit **S**trategy.
HDFS **H**adoop **D**istributed **F**ile **S**ystem.
HEC **H**igh-end **C**omputing systems.
HLES **H**olt **l**inear and **E**xponential **S**moothing.
HPC **H**IGH de **P**erformance **C**omputing systems.
PRM **P**artial **R**eplication **M**odel.
QoS **Q**uality of **S**ervice.
SGBD **S**ystème de **G**estion de **B**ases de **D**onnées.
UOT **U**ser **O**peration **T**able.
VMM **V**irtual **M**onitor **M**achine.

Introduction générale

Au cours de la dernière décennie, les recherches et les évolutions autour des technologies de réseaux, des microprocesseurs et des supports de stockage de données ont fortement contribué à l'émergence de l'informatique distribuée. Les performances de calcul des microprocesseurs et les capacités de stockage des données ne cessent de croitre, alors que les nouvelles technologies de réseaux ouvrent de nouvelles potentialités pour les communications. Ce contexte a motivé la communauté informatique à s'intéresser aux architectures distribuées et plus précisément le Cloud computing, afin de tirer profit de sa capacité en puissance de calcul et en stockage de données. Ce type d'infrastructure constitue un environnement privilégié pour le partage de ressources et de services. Les techniques de partage, utilisées par le Cloud computing, sont le plus souvent basées sur le principe de réplication pour assurer une disponibilité très élevée des données et accélérer le temps d'accès aux données.

Dans ce contexte, un même objet (fichier, programme, etc.) peut être répliqué en autant de copies que nécessaire. Chaque copie étant détenue et accédée par un ou plusieurs utilisateurs, leur gestion pose le problème du maintien de leur cohérence.
La complexité de gestion de cette réplication est due à plusieurs facteurs, parmi lesquels : Trouver le bon placement des répliques ; décider pour un objet donnée, du nombre minimal (ou maximal) de répliques et gérer la mise à jour et le contrôle des répliques. Les approches existantes pour la gestion de la réplication offrent un ensemble limité de garanties d'un point de vue performance et qualité de service.
Le but principale de ce travail et de trouver une solution pour résoudre le problème

de réplication dans le Cloud computing, avec comme objectif la nécessité de garantir les performances et la qualité des services comme : la disponibilité, le temps d'accès et la cohérence des répliques.
Notre contribution consiste à proposer un modèle pour la gestion de la réplication dans le Cloud computing, c'est une solution qui permet de garantir la cohérence des données en nous basons sur les travaux effectués dans [9].

Notre mémoire est structuré autour de quatre chapitres avec une introduction et une conclusion générale.
Le premier chapitre est une initiation au terme de l'informatique en nuage plus connu sous le nom de Cloud computing. Le Cloud computing fait référence à l'utilisation des capacités de calcul des ordinateurs distants, où l'utilisateur dispose d'une puissance informatique considérable sans avoir à posséder des unités puissantes. Par ailleurs, les applications utilisent et génèrent des quantités toujours plus importantes de données souvent répliquées et partagées entre plusieurs utilisateurs.
Le deuxième chapitre est intitulé gestion des données dans le Cloud computing, il introduit la notion de disponibilité et présente les concepts de cohérence et de réplication qui sont fondamentaux dans notre travail.
Dans le troisième chapitre nous avons présenté quelques méthodes développées dans la littérature traitant la réplication des données dans le Cloud.
Le quatrième chapitre contient notre contribution concernant le problème de réplication de données dans les environnements Cloud, nous avons proposé une stratégie de réplication qui permet de garantir la disponibilité la cohérence des données, la fiabilité et la tolérance aux pannes.

Chapitre 1

Généralités sur le Cloud computing

De nos jours les systèmes d'informations sont constamment à la recherche de nouvelles technologies afin de rendre leurs systèmes informatiques performants, économiques, et écologiques. Des études ont révélé que la plupart des serveurs dans nos salles machines utilisent que 10% [] des capacités de leurs processeurs ; ce qui représentait une grosse perte en terme de ressources financières, humaines, et de stockage. Pour pallier à ce problème de grandes entreprises comme Amazone, Microsoft... ont décidé de mettre à la disposition des entreprises et des particuliers leurs infrastructures informatiques ce qui a donné naissance au Cloud computing.
Cependant beaucoup d'entreprises au capital réduit peinent à s'introduire dans le domaine et utilisent des technologies obsolètes. Dans ce chapitre nous présenterons le Cloud (nuage) computing, ses caractéristiques et ses domaines d'application, ainsi que son architecture et les services qu'il offre.

Définition 1.0.1. [1] Le Cloud computing est un modèle d'accès, à travers le réseau Internet, à un ensemble de ressources numériques, pouvant être allouées et libérées à la demande et pour lesquelles le fournisseur du service assure l'ensemble des activités de maintenance, de support et d'exploitation. C'est une forme particulière de gestion de l'informatique, dans laquelle l'emplacement et le fonctionnement du nuage ne sont pas portés à la connaissance des clients.
Ce modèle offre des services de différentes natures, allant des services d'infrastructure

(location de capacités de stockage ou de calcul), des services de plateforme (location d'environnements de développement pré-configurés) ou de services d'applications (location d'applications).

1.1 Caractéristiques

Le Cloud computing se distingue des solutions traditionnelles par les caractéristiques suivantes [1] :

- **Large accessibilité via le réseau :** Les services sont accessibles en ligne et sur tout type de support (ordinateur de bureau, portable, smartphone, tablette). Tout se passe dans le navigateur Internet.

- **Mesurabilité du service :** L'utilisation du service par le client est supervisée et mesurée afin de pouvoir suivre le niveau de performance et facturer le client en fonction de sa consommation réelle.

- **Solution multiclient :** Une même instance d'un logiciel est partagée par l'ensemble des clients de façon transparente et indépendante. Tous les clients utilisent la même version du logiciel et bénéficient instantanément des dernières mises à jour. Chaque client dispose d'un paramétrage utilisateur qui lui est propre.

- **Disponibilité à la demande :** Le service peut être souscrit rapidement et rendu opérationnel automatiquement avec un minimum d'interaction avec le fournisseur.

- **Élasticité immédiate des ressources :** Des ressources supplémentaires peuvent être allouées au service pour assurer la continuité du service en cas

de pic de charge, ou bien être ré-allouées à un autre service dans le cas inverse.

– **Mutualisation des ressources :** Des ressources utilisées pour exécuter le service sont mutualisées pour servir à de multiples clients. Les multiples serveurs sollicités, totalement inter-connectés, ne forment plus qu'une seule ressource virtuelle puissante et performante.

1.2 La virtualisation

La visualisation est l'utilisation de logiciels et de matériel pour créer la perception qu'une ou plusieurs entités existent, bien que les entités ne soient pas physiquement présentes. En utilisant la virtualisation, un serveur peut apparaitre comment étant plusieurs, un ordinateur de bureau semble faire fonctionner plusieurs systèmes d'exploitation simultanément ou bien une grande quantité d'espace disque ou de lecteurs sont disponibles. Les formes de virtualisation les plus courantes sont : la virtualisation du serveur, virtualisation du bureau, virtualisation des réseaux et virtualisation du stockage. La figure 1.1 représente le schéma du processus de virtualisation dans le Cloud computing [3].

L'hyperviseur ou le moniteur de la machine virtuelle (VMM) est un logiciel qui crée et exécute des machines virtuelles. L'hyperviseur exécute une ou plusieurs machines virtuelles sur une machine baptisée la machine hôte. Cette dernière peut être un ordinateur aussi bien qu'un serveur. Chacune des machines virtuelles est appelée machine cliente. Les systèmes d'exploitations clients sont représentés par l'hyperviseur avec une plateforme opératoire virtuelle. Il gère l'éxécution du système d'exploitation client.

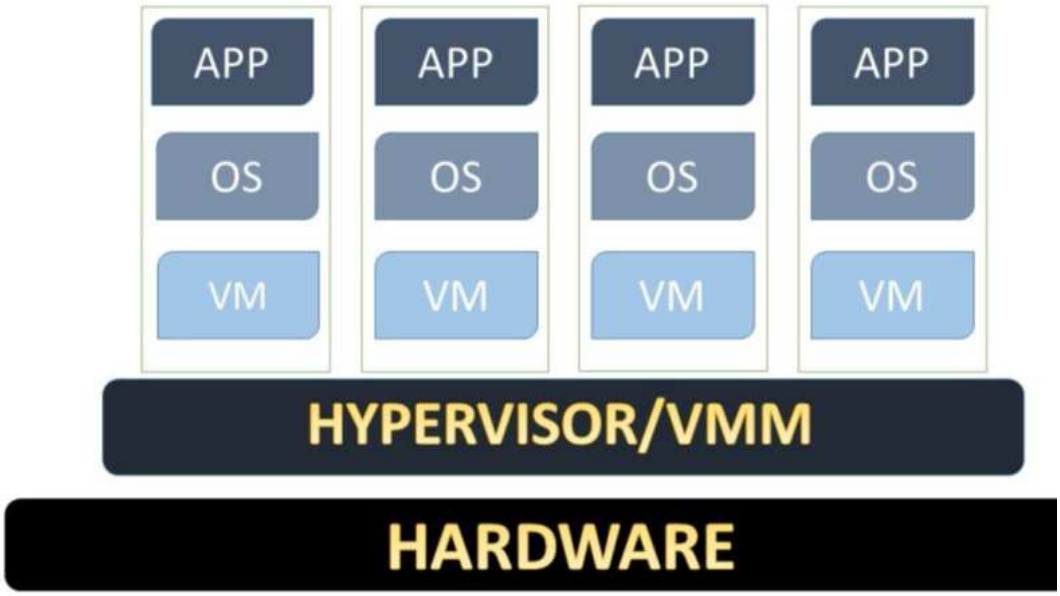

FIGURE 1.1 – Processus de virtualisation dans le Cloud computing [3].

1.3 Architecture du Cloud computing

1.3.1 Architecture globale du Cloud computing

L'architecture globale du Cloud computing comporte essentiellement [4] :

1. **Clients :** Un client Cloud se compose de matériel informatique et/ou de logiciels qui s'appuient sur le Cloud computing pour la livraison des applications, ou qui est spécifiquement conçu pour la fourniture de services Cloud et qui, dans les deux cas, est essentiellement inutile sans elle.

2. **Services :** Un service Cloud comprend des produits, des services et des solutions livrés et consommés en temps réel sur Internet. Par exemple, les ser-

vices Web accessibles par d'autres composants et logiciels de Cloud computing.

3. **Applications :** Une application Cloud exploite le Cloud dans l'architecture logicielle, ce qui élimine souvent la nécessité d'installer et d'exécuter l'application sur son propre ordinateur, Ce qui allège le fardeau de la maintenance logicielle, du fonctionnement continu et du support.

4. **plateforme :** Une plateforme Cloud facilite le déploiement d'applications sans coût, la complexité d'achat, de gestion du matériel et des logiciels sous-jacents.

5. **Le stockage :** Le stockage dans le Cloud implique la livraison de stockage de données en tant que service, y compris des services de base de données, souvent facturés sur une base de calcul utilitaire.

6. **L'infrastructure :** L'infrastructure Cloud, en tant que service, est la fourniture d'infrastructures informatiques, généralement un environnement de visualisation de plates-formes.

1.3.2 Architecture de référence du Cloud computing

L'architecture de référence du Cloud computing comme fournie dans [5] détermine les éléments architecturaux constituant un environnement Cloud computing, elle définit trois rôles principaux : Cloud Service Consumer, Cloud Service Provider et Cloud Service Creator. Chaque rôle peut être rempli par une seule personne ou par un groupe de personnes ou une organisation. L'architecture de référence du Cloud computing est illustrée dans la figure 1.2.

1. **Cloud Service Consumer :** Un Cloud Service Consumer est une organisation, un être humain ou un système informatique qui consomme (i.e., demande,

utilise et gère, par exemple, modifie la capacité d'UC affectée à une machine virtuelle) des instances de service fournies par un service Cloud particulier. Le Cloud Service Consumer peut être facturé pour toutes (ou une partie) ses interactions avec le service Cloud et les instances de service provisionnées.

2. **Cloud Service Provider :** Un Cloud Service Provider a la responsabilité de fournir des services Cloud pour des Cloud Service Consumer. Un Cloud Service Provider est défini par la propriété d'une plateforme commune de gestion du Cloud (CCMP). Cette propriété peut être réalisée soit en exécutant réellement une CMPP en elle-même ou en consommant une comme un service.

3. **Cloud Service Creator :** Tout comme le Cloud Service Consumer et le Cloud Service Provider, le Cloud Service Creator peut être une organisation ou un être humain. Il est responsable de la création d'un service Cloud, qui peut être géré par un Cloud Service Provider et exposé au Cloud Service Consumers. En général, les Cloud Service Creators créent leurs services Cloud en exploitant les fonctionnalités exposées par un Cloud Service Provider.

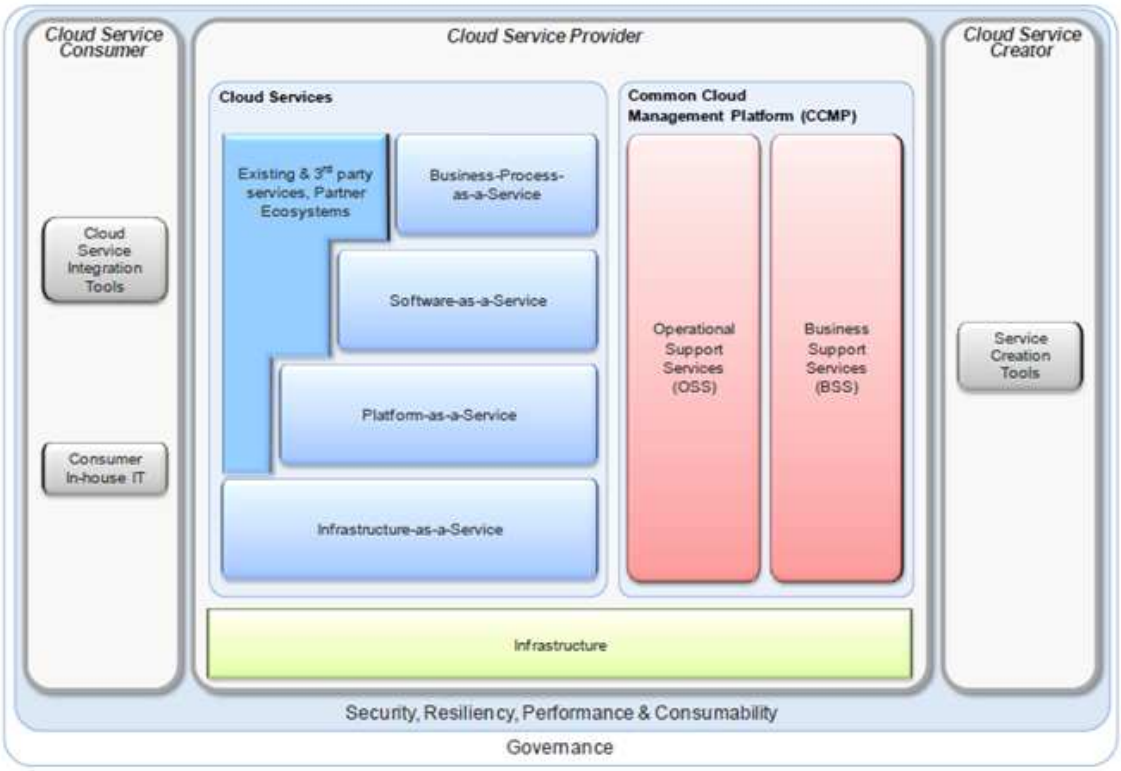

FIGURE 1.2 – L'architecture de référence du Cloud computing [5].

1.4 Services du Cloud computing

1.4.1 Modèles de déploiement

Le Cloud computing peut être déployé en quatre types différents [1] :

1. **Cloud public :** Dans le cas du Cloud public, le fournisseur propose un environnement informatique avec une mutualisation optimale des ressources : l'environnement est ainsi virtuellement partagé avec un nombre illimité de clients.
   Une telle offre privilégie l'élasticité et la flexibilité, et propose, avec la

massification des clients, des prix plus attractifs.
Cependant, la mutualisation généralisée entraîne une difficulté plus grande pour le fournisseur de garantir un niveau de service spécifique à un client.

2. **Cloud privé ou Cloud dédié :** Le Cloud privé vient répondre aux exigences des clients souhaitant des garanties exclusives sur un contrat de service. Le prestataire met en place des ressources dédiées à un client donné, sur un environnement hébergé à demeure chez le client ou à distance chez le fournisseur.
   Ainsi, l'infrastructure réservée au client bénéficie des technologies de visualisation afin de pouvoir être mutualisée et les ressources disponibles automatiquement allouées en fonction des besoins des applications internes.

3. **Cloud communautaire :** Le Cloud communautaire correspond à un Cloud privé partagé par un groupement d'acteurs. Ce mode de déploiement vise à abaisser la barrière à l'entrée du Cloud privé tout en bénéficiant d'un niveau de service spécifique. Il est adapté à la mutualisation au sein d'un écosystème où la co-gouvernance est envisageable (regroupement de collectivités ou d'acteurs publics, communautés d'universités).

4. **Cloud hybride :** Le Cloud hybride est un mode de déploiement combinant l'utilisation d'un Cloud public avec un environnement en Cloud privé. Il s'agit dans ce cas de faire cohabiter ces environnements afin de pouvoir absorber plus facilement des pics de charge ponctuels : l'environnement privé est réservé aux systèmes courants, tandis que les capacités du Cloud public sont utilisées pour absorber ponctuellement les montées en charge.

1.4.2 Modèles de services

Trois grands modèles d'usage du Cloud se dégagent actuellement, tous présentent des caractéristiques différentes comme illustré dans la figure 1.3 [5] :

1. **IaaS (Infrastructure as a Service, des services virtuels disponibles à la demande)**
   Il s'agit de l'offre la plus basique dans le portefeuille Cloud computing correspondant à la location de capacités de calcul et de stockage.
   Dans un service IaaS, le fournisseur met à disposition et administre les ressources matérielles virtualisées comprenant :
   – La puissance de calcul ;
   – Les unités de stockage des données ;
   – Les réseaux ;
   – Les couches de virtualisation ;
   – Les systèmes d'exploitation ;
   Le client prend en charge la gestion et l'exploitation de toutes les couches supérieures, middlewares, bases de données et applications.
   La souscription à une offre IaaS permet au client d'externaliser son parc matériel serveur et de s'affranchir des compétences de conception et d'exploitation des infrastructures techniques. Les prestataires des solutions IaaS les plus connus sont : Amazone avec Amazone Elastic Compute Cloud (EC2), ou Orange Business Service avec Flexible Computing.

2. **PaaS (Platform as a Service, des plateformes de développement prêtes à l'emploi)**
   Il s'agit de l'offre intermédiaire dans le portefeuille Cloud computing.
   Le fournisseur met à disposition une plateforme middleware opérationnelle, incluant des serveurs d'applications, des bases de données et les outils permettant au client de développer et de déployer ces propres applications.
   Cette configuration est très employée pour disposer de plateformes de développement ou de tests disposant de l'ensemble des outils et middleware nécessaires,

en évitant ainsi les tâches de construction et de maintenance de ces plateformes non critiques. Elle se destine donc naturellement avant tout aux développeurs. Des services comme Google App Engine, Bungee Connect et Force.com sont des exemples PaaS. Les principaux fournisseurs de PaaS sont : Microsoft avec Azure, Google avec Google App Engine et Orange Business Services.

3. **SaaS (Software as a Service, des services métiers à la demande)**
   Il s'agit d'une offre « tout compris ». Le prestataire met à disposition une application qu'il administre et configure en majeure partie. Le client externalise ainsi ses applications auxquelles il accède à la demande. Il paie à l'usage, selon le nombre d'utilisateurs et/ou le temps d'utilisation du logiciel. Les prestataires de solutions SaaS les plus connus sont : Google avec Gmail et Youtube ou encore les réseaux sociaux Facebook et Twitter.

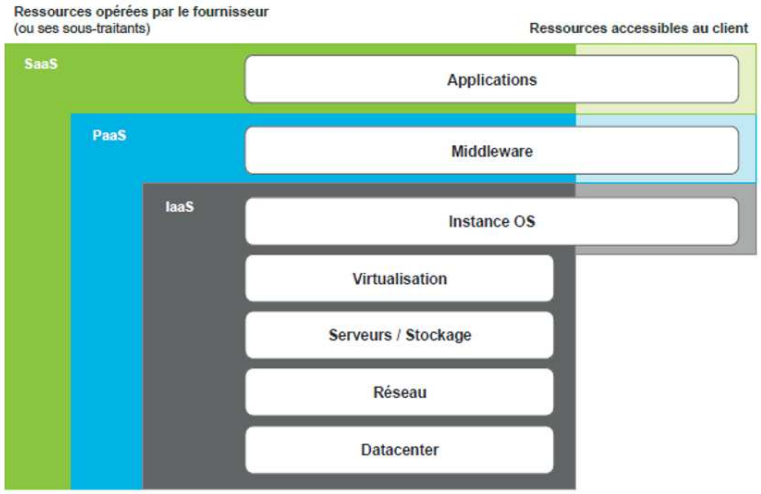

FIGURE 1.3 – Catégories d'offres Cloud computing et couches techniques [1].

1.5 Challenges

Le Cloud présente plusieurs défis [2] :

- **Sécurité :** L'utilisation des réseaux publics, dans le cas du Cloud public, entraîne des risques liés à la sécurité du Cloud. En effet, la connexion entre les postes et les serveurs applicatifs passe par le réseau Internet, et expose à des risques supplémentaires de cyber attaques, et de violation de confidentialité.

- **Disponibilité :** Le client d'un service de Cloud computing devient très dépendant de la qualité du réseau pour accéder à ce service. Aucun fournisseur de service Cloud ne peut garantir une disponibilité de 100%.
  .
- **Piratage :** Tout comme les logiciels installés localement, les services de Cloud computing sont utilisables pour lancer des attaques (craquage de mots de passe, déni de service..). En 2009, par exemple, un cheval de Troie a utilisé illégalement un service du Cloud public d'amazone pour infecter des ordinateurs.

- **Cadre légal :** Des questions juridiques peuvent se poser, notamment par l'absence de localisation précise des données du Cloud computing.

- **Réversibilité :** En cas de rupture de contrat ou de changement de fournisseur, le client doit s'assurer de la récupération et de la destruction de ses données sur l'infrastructure du fournisseur après sa migration.

1.6 Domaines d'application du Cloud computing

Actuellement et sans que les utilisateurs se rendent compte, de nombreuses applications sont passées dans le Cloud :

- **Domaine militaire :** La technologie employée dans un environnement militaire doit être opérationnelle en toute circonstance. Qu'il s'agisse de garantir l'exécution correcte des opérations quotidiennes, le fonctionnement constant des communications internationales ou la gestion réussie de situation

de crise, la solution militaire Camera manager (vidéo protection dans le Cloud) de Panasonic permet de garder un oeil constant sur les organisations militaires. Il ne se limite pas à l'enregistrement et l'affichage en temps réel, mais il embarque également de nombreuses fonctions courantes de gestion vidéo-programmation, alarme, utilisation d'une API, gestion des utilisateurs, et est capable de supporter tout ce qu'on fait, quelles que soient les difficultés à surmonter [26].

- **Domaine médical :** Le Cloud a la capacité d'améliorer la collaboration dans lindustrie des soins de santé. En permettant aux professionnels de stocker et accéder aux données à distance. Les professionnels de santé aux quatre coins du monde peuvent accéder aux données des patients et appliquer les soins nécessaires au plus vite. De plus, la téléconférence, les mises à jour des dernières innovations en soins de santé et les conditions du patient en temps réel, permettent d'économiser du temps pour sauver des vies [19].
  Followmed est une solution Cloud proposant de multiples fonctionnalités ayant pour objectif de simplifier le quotidien des professionnels de santé, gérez leur activité médicale, innovez leur communication sur le réseau social dédié aux professionnels de la santé, utilisez ce logiciel médical en Cloud pour stocker et accéder à l'ensemble de leurs données où qu'ils soient [**?**].

- **Domaine de l'éducation :** Le Cloud est un excellent levier technologique pour fournir des services collaboratifs extrêmement riches permettant de créer des sessions extrêmement interactives et de réaliser des opérations d'apprentissage à distance qui vont être complémentaires à l'enseignement traditionnel. Par exemple des usages de type blending-learning (ou formation mixte) avec de la vidéo en apprentissage seul, du cours en présentiel et du cours virtuel connecté qui va permettre aux étudiants d'affranchir des limites de l'espace et offrir un panel de possibilité plus important [8].
  Virtual Computing Lab (VCL) est une infrastructure Cloud qui fédère des ressources informatiques de plusieurs sites (serveurs, systèmes de stockage, instances de logiciels). Elle permet aux élèves des différents établissements

primaires et secondaires de l'Etat ainsi qu'aux étudiants des différents campus de l'université d'accéder, où qu'ils se trouvent (en ce compris chez eux), à un pool de ressources techniques et pédagogiques évoluées et à jour [26].

– **Domaine industriel :** Pour pérenniser la croissance de l'industrie manufacturière, les fabricants doivent sans cesse améliorer la précision et la rapidité de leurs procédés, et optimiser chaque interaction avec les fournisseurs, distributeurs et prestataires de services. On assiste ainsi à la démocratisation des architectures technologiques "just in time" (ou dynamiques), basées sur des Clouds publics et privés, qui aident les fabricants à tenir leurs objectifs et à maintenir leur compétitivité [11]. Salesforce est une plateforme Cloud complète permettant de gérer les interactions entre les fournisseurs et leurs clients, et les aider à prospérer et réussir [25].

1.7 Conclusion

Dans cette partie, nous avons défini le Cloud computing, nous avons aussi donné ses caractéristiques, ses challenges ainsi que ses domaines d'application. Le Cloud computing marque une réelle avancée vers l'infrastructure informatique dématérialisée. Il fournit des ressources informatiques, logicielles ou matérielles, accessible à distance en tant que service. L'adoption de ce modèle soulève un certain nombre de défis, notamment au sujet de la disponibilité des ressources. Dans le prochain chapitre nous introduirons la notion disponibilité ainsi que celle de la cohérence dans les environnements Cloud.

Chapitre 2

Gestion des données dans le Cloud computing

Le Cloud computing est basé sur le concept de la visualisation qui consiste à faire fonctionner plusieurs systèmes sur un même serveur physique. Les principaux avantages étant d'offrir une large accessibilité, une grande fiabilité et une meilleure qualité de service, donnent au même temps naissance à certaines contraintes parmi elle devoir offrir une haute disponibilité aux clients. La réplication peut être une solution à ce problème, toutefois celle-ci soulève la question de la cohérence des données entre les répliques.
Dans ce chapitre, nous allons introduire la notion de disponibilité et présenter les concepts de cohérence et de réplication qui sont fondamentaux dans notre travail.

2.1 Disponibilité dans le Cloud computing

Définition 2.1.1. [6] Un service est dit hautement disponible (Highly available en anglais) si toute requête reçue par un noeud n'étant victime d'aucune défaillance retourne un résultat.
La haute disponibilité (HA pour « High Availability ») d'un service caractérise sa capacité à assurer son effective accessibilité durant une période donnée. Le service devra donc pouvoir détecter les points de défaillances et réduire l'impact de leur potentielle défaillance grâce à la mise en place de techniques de redondance et/ou réplication.

2.1.1 Importance de la haute disponibilité dans le Cloud Computing

La disponibilité d'un service est donnée par la quantité de temps pendant laquelle le service est utilisé par les clients dans des conditions normales et anormales. La haute disponibilité découle du fait que ses clients ont besoin d'un accès permanent au service [7].
L'indisponibilité de certains services a un impact négatif pour leurs clients, c'est le cas des institutions bancaires, des entreprises de télécommunications, des applications militaires ou des hôpitaux. Les infrastructures Cloud doivent offrir une disponibilité supérieure à 99,9% ; Donc la dégradation des performances est une préoccupation plus grave que les défaillances de ressources dans de tels environnements [8].
L'augmentation de la demande pour la disponibilité continue des systèmes de calcul haute performance (HPC : High-performance computing systems) est évidente. C'est une étape majeure vers l'informatique de capacité, où les applications scientifiques demandent des quantités considérables de temps (semaines et mois) sans interruption sur les machines HPC disponibles les plus rapides [9].
Ces systèmes de calcul haut de gamme (HEC : High-end computing systems) doivent pouvoir fonctionner en cas de défaillance de manière à ce que leur potentiel ne soit pas rigoureusement dégradé. La haute disponibilité (HA) a longtemps joué un rôle important dans les applications critiques, comme dans les secteurs militaire, bancaire et des télécommunications.

2.1.2 Les solutions de disponibilité dans le Cloud

L'indisponibilité est principalement due à des pannes matérielles et logicielles, au réseau de communication, dénis de service ou catastrophes naturelles. Les solutions de disponibilité existantes se basent soit sur la réplication et la redondance matérielle, soit sur le rajeunissement.
L'une des solutions basées sur le rajeunissement utilise le mécanisme de migration en live, elle consiste à répliquer au bon moment le contenu des machines virtuelles dégradées sur un noeud de veille qui joue le rôle du noeud principal en attendant

la fin des mises à jour correctives. Cette technique a pour objectif de réduire la consommation de ressources et de produire également des améliorations significatives, car elle permet de rétablir le fonctionnement initial des machines virtuelles et donc la disponibilité prévue au départ est maintenue à condition que la migration se fasse au bon intervalle de temps avant la procédure de rajeunissement [41].
L'une des solutions basées sur la réplication décrit et surveille la disponibilité des ressources du Cloud, il est également utilisé par les clients pour décrire leurs besoins en terme de qualité de service (QoS) orienté disponibilité. Ce modèle QoS combine les caractéristiques du Cloud avec les attentes des clients pour surveiller la disponibilité du service.
L'idée est de faire correspondre une requête d'un client au noeud adéquat qui offre la meilleure QoS. Dans cette approche un algorithme de sélection du noeud de service offrant la meilleure QoS en terme de disponibilité parmi plusieurs noeuds est proposé et modélisé par un système d'équation.
La fonction objective représente le plus grand taux de QoS, tandis que les contraintes sont issues du modèle de QoS des demandes des clients. A la résolution de cette fonction, le noeud de service ayant ce score maximum sera attribué au service demandé par le client [42].

2.2 Les bases de données distribuées

Les données d'une application sont de plus en plus souvent distribuées entre plusieurs serveurs, cela permet de conserver l'intégrité des données et accorder un traitement rapide des requêtes (concurrence) ainsi qu'une souplesse d'utilisation.

Définition 2.2.1. [12] Une base de données (BD) est un ensemble cohérent de données très structurées. Une entité de base de données se caractérise par un nom et une valeur, des opérations de lecture et d'écriture peuvent être appliquées sur cette dernière.
Un système de gestion de base de données (SGBD) peut être perçu comme un ensemble de logiciels qui manipulent le contenu des bases de données. Il sert à

effectuer les opérations ordinaires telles que rechercher, ajouter ou supprimer des enregistrements (Create, Read, Update, Delete), manipuler les index, créer ou copier des bases de données.

Définition 2.2.2. [21] Une base de données répartie est constituée d'un ensemble de bases de données logiquement reliées et distribuées sur un réseau.

Définition 2.2.3. [18] Un SGBD réparti permet aux utilisateurs de gérer une base de données répartie de manière transparente comme s'ils avaient accès à une seule base de données. Chaque site contient alors soit une copie de la base de données répartie (réplication totale) ou une partie de la base de données répartie (réplication partielle).

2.2.1 Cohérence d'une base de données

Une base de données est dite cohérente si elle satisfait un certain nombre de contraintes. Une transaction est une unité de travail définie par [23] regroupant une suite d'opérations qui doivent être exécutées sur la base de données. Une transaction est considérée comme valide par le SGBD si elle ne remet pas en cause la cohérence de la base de données. Pour être valide, une transaction doit présenter les propriétés ACID c'est-à-dire [18] :

1. **Atomicité :** Elle assure qu'en cas de succès, toutes les opérations sont validées et qu'en cas d'échec aucune opération n'est appliquée.
2. **Cohérence :** La transaction ne doit pas remettre en cause la cohérence de la base de données.
3. **Isolation :** Une transaction n'a pas d'effet visible tant qu'elle n'a pas été validée.
4. **Durabilité :** Une transaction validée ne peut pas être remise en cause. Les conséquences sur la base sont persistantes et ne peuvent pas être supprimées. L'application d'une transaction valide modifie de ce fait l'état de la base de données.

2.2.2 Réplication dans les systèmes de gestion des bases de données

Une base de données répliquée est composée d'un ensemble de répliques stockées sur différents noeuds d'un SGBD réparti, chaque réplique étant la copie d'une donnée de référence [24].
Dans un SGBD réparti, la mise à jour de la base de données répartie s'effectue soit selon une approche dite de copie-primaire (la mise à jour est centralisée sur un site puis propagée aux autres sites), ou selon une approche distribuée (la mise à jour peut être effectuée sur n'importe quel site). L'utilisation d'une copie primaire permet d'éviter les transactions concurrentes mais pose le problème de la tolérance aux pannes. Autoriser la mise à jour sur tous les sites permet d'obtenir l'information de mise à jour plus rapidement mais nécessite un ordonnancement des transactions provenant des différentes copies. Par ailleurs, la propagation des mises à jour est gérée par le SGBD réparti et peut être effectuée de manière synchrone ou asynchrone [18].

1. **Propagation synchrone :** Consiste à valider une transaction après l'avoir propagée aux autres répliques. Cette approche assure la cohérence des données de manière simple et garantit la propriété d'atomicité [**?**] mais le retardement de la validation des transactions provoque un surcoût des temps de réponse et engendre de ce fait des problèmes de performance et de disponibilité des données.
2. **Propagation asynchrone :** Consiste à valider localement la transaction, avant de la propager aux autres répliques. Cette validation locale permet aux utilisateurs de pouvoir continuer à travailler en mode déconnecté. Une validation définitive est alors effectuée la reconnexion lors d'une étape de synchronisation. Cependant, les opérations effectuées sur le site local pendant la déconnexion peuvent être en déconnexion avec d'autres opérations qui ont déjà été validées par le SGBD réparti, il faut alors vérifier la cohérence mutuelle des bases de données. La cohérence mutuelle est définie comme étant le fait qu'à un instant donné, toutes les copies d'une donnée ont la même valeur.
   Un autre inconvénient de la propagation asynchrone est que la propriété d'ato-

micité ne peut être garantie. En effet, si une panne se produit avant qu'une transaction validée localement puisse être propagée aux autres sites alors cette transaction est perdue.

2.3 Réplication des données dans le Cloud computing

La réplication crée plusieurs copies d'une entité existante [20]. Elle permet d'assurer la disponibilité des ressources, elle assure également la fiabilité en créant plusieurs copies des mêmes données sur différents sites. La réplication fournit un coût d'accès minimal et une utilisation partagée de la bande passante en répliquant les données. La valeur de la réplication est de fournir un accès transparent et sans faille aux ressources en cas de défaillance du système [13].

2.3.1 Techniques de réplication

Pour proposer une stratégie de réplication, il va falloir répondre aux problèmes suivants [10] :

1. **Sélection de la donnée à répliquer :** Les ressources distribuées devraient être répliquées selon la popularité, afin de maximiser la probabilité de satisfaction des requêtes des utilisateurs.
2. **La position des nouvelles répliques sur les centres de données disponibles :** Pour raison de sécurité, de coût, de facilité de gestion, de capacité de stockage, etc. Certains sites ont une copie et d'autres pas. Nous citons alors deux critères nous permettant de préciser l'emplacement des répliques :
   - **Distance :** Les répliques sont placées sur les sites qui ont plus de chances de les demander, cela leur permet de chercher et de trouver les ressources demandées, et de réduire les délais de recherche et de téléchargement.
   - **Coût :** Le coût de placement des répliques est la somme de trois termes :
   * Le coût de stockage des répliques.
   * Le coût des lectures.

* Le coût des écritures.

La multiplication des répliques réduit le coût des lectures et augmente celui du stockage et des écritures, alors que la diminution du nombre de répliques a l'effet inverse. Néanmoins, la réplication implique une gestion plus complexe des mises à jour et nécessite une vérification de la cohérence entre les données répliquées et la donnée de référence.

3. **Mise à jour et contrôle :** Le processus de réplication a lieu lorsqu'une donnée est mise en ligne par un client et en crée en plus une réplique. Il existe trois approches de réplication :
   * **Réplication instantanée :** Les données sont copiées du site principal vers les sites secondaires. Les changements au niveau des sites secondaires doivent provenir du principal. Ainsi, seuls les secondaires sont interrogés, mais leurs données ne peuvent être modifiées par les clients.

   * **Réplication par fusion :** Les données sont combinées de deux ou plusieurs sites vers un troisième site. Cette méthode est plus difficile à mettre en oeuvre qu'une réplication instantanée.

   * **Réplication transactionnelle :** La donnée est copiée complètement, suivie des mises à jour qui sont copiées périodiquement des sites principaux vers les sites secondaires.

2.4 Types de réplication

2.4.1 Réplication synchrone

Elle garantit que lorsqu'une transaction met à jour une réplique primaire, toutes ses répliques secondaires sont mises à jour dans la même transaction. L'avantage essentiel de la mise à jour synchrone est de garder toutes les données au même niveau de mise à jour. Le système peut alors garantir la fourniture de la dernière version des données quel que soit la réplique accédée. Les inconvénients sont cependant multiples,

ce sont d'une part, la nécessité de gérer des transactions multiples coûteuses en ressources et d'autre part, la complexité des algorithmes de gestion de panne d'un site, etc. C'est pour cela que l'on préfère souvent le mode de mise à jour asynchrone [10].

2.4.2 Réplication asynchrone

C'est le mode de distribution dans lequel certaines sous-opérations locales effectuées suite à une mise à jour globale sont accomplies dans des transactions indépendantes en temps différé. Le temps de mise à jour des copies peut être plus au moins différé : les transactions de report peuvent être lancées dès que possible ou à des instants fixes, par exemple le soir ou en fin de semaine [15]. L'avantage est la possibilité de mettre à jour en temps choisi des données, tout en autorisant l'accès aux versions anciennes avant la mise à niveau. L'inconvénient est que l'accès à la dernière version n'est pas garanti.
La réplication asynchrone est basée sur deux approches [10] :

- **Approche basée sur l'état :** La réplique source est mise à jour, ensuite le sous- système transmet l'état sur les répliques par la fusion de l'état livré à l'état local.
- **Approche basée sur les opérations :** Le sous-système envoie l'opération de mise à jour et ses paramètres à toutes les répliques.

2.4.3 Avantages de la réplication dans le Cloud

La réplication présente plusieurs avantages parmi lesquels [14] :

- **Disponibilité des données :** Permettre à un serveur secondaire de prendre rapidement la place du serveur principal lorsque celui-ci échoue, ou alors les répliques répondent aux requêtes qui sollicitent une même donnée. S'il y a une perte de toute connectivité réseau, avoir une réplique complète signifie qu'il y a toujours accès à toutes les données.

– **Amélioration des performances d'accès aux données** (meilleur temps de réponse, diminution du temps de transfert des données, etc.).

– **Autonomie accrue :** Les utilisateurs peuvent manipuler des copies de données hors connexion, puis propager leurs modifications aux autres bases de données lorsqu'ils sont connectés.

2.5 Cohérence des données dans le Cloud

Définition 2.5.1. [10] Un stockage distribué cohérent se réfère souvent au fait que chaque mise à jour soit appliquée à tous les noeuds concernés en même temps logique. Une conséquence de cette définition est que, si une donnée est répliquée sur des noeuds différents, elle serait toujours la même sur chaque noeud à chaque instant.

2.5.1 Modèles de cohérence

Un modèle de cohérence pour des données réparties spécifie un contrat entre un client et le système de gestion de données (avec engagement mutuel).
Les données réparties sont un ensemble d'éléments (un élément peut avoir une granularité quelconque : octet, enregistrement, fichier, etc.). Les opérations sont lire et écrire (éventuellement des opérations plus complexes, mais toujours divisées entre consultation et modification). Les modèles de cohérence les plus courants sont [15] :

Modèles sans synchronisation

1. **Cohérence forte** : Dans les systèmes traditionnels de stockage et de bases de données distribuées, la manière instinctive et correcte de traiter la cohérence des répliques était d'assurer un état de cohérence forte de toutes les répliques dans le système à tout moment. La cohérence forte garantie que toutes les répliques sont dans un état cohérent immédiatement après une mise à jour, avant de retourner un succès. Ce modèle exige des mécanismes très couteux en termes de performance et de disponibilité et limitent l'involutivité du système. Ce ne

fut pas un problème dans les premières années des systèmes de stockage distribués, les performances nécessaires à l'époque n'étaient pas aussi importantes. Cependant, à l'ère du Big Data et Cloud computing, ce modèle de cohérence peut être pénalisant en particulier si une telle cohérence forte n'est en réalité pas requise par les applications [10].

2. **Cohérence séquentielle** : Un système est cohérent séquentiellement si le résultat de n'importe quelle exécution est le même si les opérations de tous les opérations de tous les processus sont exécutées dans un ordre séquentiel, et les opérations de chaque processus individuel apparaissent dans cette séquence dans l'ordre spécifié dans son programme. Dans un système essentiellement cohérent, tous les processus doivent s'entendre sur l'ordre des effets observés (on impose un ordre total sur les écritures). Les lectures locales à chaque processus lisent la dernière valeur écrite [16].
3. **Cohérence causale** : Modèle plus faible que la cohérence séquentielle, car on ne considère que des événements reliés par une relation de causalité.
   Exemple : Soit deux événements E1 et E2 tels que E1 E2 (précédence causale). Alors tout processus doit "voir" E1 avant E2 [16].
4. **Cohérence FIFO** : Des écritures réalisées par un même processus doivent être vues par tous les processus dans leur ordre de réalisation. Des écritures réalisées des processus différents peuvent être vues dans un ordre différent [16].
5. **Cohérence stricte** : On parle de cohérence atomique, indivisible, stricte ou non-interruptible quand une lecture renvoie toujours le résultat de la dernière écriture effectuée, ce qui implique l'existence d'une base de temps globale. Ce modèle de cohérence est le plus fort qui soit [17].
6. **Cohérence cache** : Pour chaque emplacement d'une donnée, il existe un ordre total des écritures sue cette donnée. Les écritures sur différents emplacements peuvent être vues dans un ordre différent par différents processeurs [16].
7. **Cohérence éventuelle** : Ce modèle est devenu très populaire dans les dernières années, car il assure une meilleure disponibilité, et de plus grandes performances. On peut tolérer des données incohérentes pendant un certain temps T, si la condition suivante est respectée : si aucune mise à jour n'a eu lieu pen-

dant T, toutes les répliques deviendront progressivement la dernière version de la valeur. Ce type de cohérence peut poser un problème lorsqu'un client accède successivement à plusieurs répliques différentes d'une donnée. D'où une classe de modèles de cohérence définis à partir de la vue du client [10].

- **Lectures monotones** : Si un processus a lu la valeur d'une donnée, toute lecture ultérieure par ce même processus doit rendre la même valeur ou une valeur plus récente. Ce modèle garantie qu'un client ne reviendra pas en arrière [10].
- **Écritures monotones** : Si un processus exécute deux écritures successives sur une même donnée, la deuxième écriture ne peut être réalisée que quand la première a été terminée [10].
- **Cohérence écriture-lecture** : Si un processus écrit la valeur 'x' sur une donnée 'd', toute lecture ultérieure par ce processus doit retourner 'x'.
- **Cohérence lecture- écriture** : Si un processus lit une donnée 'D', récupère sa version v, puis écrit sur cette même donnée la valeur 'y', l'écriture modifiera partout la donnée 'D' ayant une version au moins aussi récente que v [10].

Modèles avec synchronisation

- **Cohérence faible :** La mise en oeuvre du modèle de cohérence forte impose, dans de nombreux cas, des limitations dans les choix de conception de systèmes et des performances de l'application. Pour surmonter ces limitations, le modèle de cohérence faible a été introduit. L'accès aux données en lecture et en écriture est considéré comme faible s'il remplit les trois conditions suivantes :

1. Tous les accès à une variable de synchronisation partagée sont fortement (séquentiellement) commandés. Tous les processeurs perçoivent le même ordre des opérations.
2. Aucun accès à une variable de synchronisation n'est fait par un processeur avant que tous les accès aux données précédentes aient été réalisés.
3. Aucune lecture ou écriture ne peut être effectuée par un processeur tant qu'un accès à une variable de synchronisation n'est pas terminé [10].

2.6 Challenges dans l'environnement de réplication dans le Cloud

Les défis posés pour la réplication dans le Cloud se résument en [15] :

a. **Cohérence des données :** Le maintien de l'intégrité et de la cohérence des données dans un environnement répliqué est une importance majeure. Les applications de haute précision peuvent nécessiter une cohérence stricte des mises à jour effectuées par les transactions.

b. **Temps d'inactivité lors de la création d'une nouvelle réplique :** Si une cohérence stricte des données doit être maintenue, Les performances sont gravement affectées si une nouvelle réplique doit être créée. Les sites ne seront pas en mesure de répondre à la demande en raison des exigences de cohérence.

c. **Coût :** Si les fichiers sont répliqués sur plusieurs sites, ils occupent plus d'espace de stockage. Cela nécessite une maintenance supplémentaire et devient une surcharge en stockant plusieurs fichiers.

d. **Performances :** La performance des opérations d'écriture peut être considérablement inférieure dans les applications nécessitant des mises à jour élevées dans un environnement répliqué, car la transaction peut avoir besoin de mettre à jour plusieurs copies.

2.7 Conclusion

La réalisation de ce chapitre nous a permis de comprendre l'importance de la disponibilité, les avantages de la réplication, ainsi que les modèles de cohérence avec leur importance dans la synchronisation des échanges entre les processus. Dans le chapitre qui suit, nous allons étudier les différents travaux qui ont été consacrés à la réplication dans le Cloud.

Chapitre 3

Etat de l'art sur les méthodes de réplication dans le Cloud computing

Le besoin croissant des applications à accéder simultanément à des données réparties sur plusieurs sites, a conduit la communauté informatique à s'intéresser à l'étude de la réplication des données dans le Cloud.
Ce chapitre comprend une présentation des méthodes développées dans la littérature traitant la réplication des données dans le Cloud.

3.1 Les méthodes de réplication existantes dans la littérature

3.1.1 Approche basée sur les réseaux de neurones

Al Ridhawi et al.[27] proposent une méthode de prédiction de localisation pour identifier des emplacements potentiels de densité élevée d'utilisateurs. Un algorithme partiel de réplication de données est utilisé pour reproduire à l'avance les données demandées par l'utilisateur. La méthode proposée a pour objectif de minimiser le temps d ?accès aux données.
La solution utilise une technique de prédiction de localisation qui s'appuie sur la théorie de Dempster-Shafer [28], c'est une théorie probabiliste mathématique qui

combine des éléments de preuve pour atteindre les décisions dans des situations d'incertitude, pour fournir des résultats de prévision rapides et précis. Par la suite, une nouvelle technique de réplication partielle des données sera utilisée pour répliquer temporairement les données les plus consultées à partir du site de stockage d'origine à des sites tiers situés dans les futurs emplacements prévus où les abonnés au service seront disponibles. Cela garantira que les données les plus accessibles seront facilement disponibles aux utilisateurs à leur arrivée, améliorant ainsi le temps d'accès aux données et maintenant un certain niveau de QoS.
Le modèle de sélection de répliques proposé fournit une solution efficace pour accéder aux fichiers locaux et distants. La solution suppose la présence de plusieurs utilisateurs répartis sur différents sites distants qui soumettront un certain nombre de jobs. Ces jobs nécessitent un ou plusieurs fichiers. Les fichiers sont situés dans des ressources locales et distantes (c'est-à-dire le site de stockage original ou des sites de stockage tiers).
Le modèle de réplication partielle (PRM) proposé permet de répliquer une partie d'un fichier au lieu du fichier entier. Un ensemble ou une partie de fichiers est sélectionné pour la réplication partielle en fonction des requêtes de soumission de jobs exécutées par les utilisateurs.
L'approche proposée utilise un réseau de neurones artificiels (RNA) qui est formé sur l'historique passé des utilisateurs pour prédire l'emplacement d'un fichier soit dans le cache, les ressources locales ou des ressources à distance. Les RNA sont des programmes informatiques qui sont formés pour reconnaître les modèles d'entrée et les associer à des sorties particulières [40].

3.1.2 Approche basée sur l'économie d'énergie

Les auteurs de l'article [29] présentent des modèles basés sur la consommation d'énergie et la demande de la bande passante, pour l'accès à la base de données dans le datacenter du Cloud computing. En outre, ils proposent une stratégie de réplication à faible consommation d'énergie induits des modèles proposés, ce qui se traduit par une amélioration de la qualité de service (QoS) avec des délais de communication réduits entre les centres de données répartis géographiquement et à l'intérieur

de chaque datacenter.
Afin d'assurer l'efficacité énergétique et la performance des applications Cloud, les auteurs proposent un algorithme de réplication qui prend en considération la consommation d'énergie et la bande passante requises pour l'accès aux données. Chaque objet de données est disponible sur la base de données DB centrale et, basé sur des observations de l'historique des fréquences d'accès aux données ; celles-ci peuvent être répliquées vers la base de données du datacenter et les bases de données de la grille. Un module appelé Replica Manager (RM) localisé à la DB centrale analyse ces métas données pour identifier les objets de données qui doivent être répliqués et dans quel emplacement. RM calcule les taux d'accès et de mise à jour dans les intervalles précédents et fait une estimation de leurs futures valeurs.
Avec ces paramètres, il devient possible de calculer la consommation d'énergie et la demande de la bande passante dans les intervalles à venir. En outre, les niveaux de congestion dans le réseau du datacenter sont surveillés afin de déterminer les candidats les mieux adaptés à la réplication.

3.1.3 Approche basée sur le Fog computing

Fog computing est un scénario où un grand nombre d'appareils ubiquitaires, décentralisés et hétérogènes (sans fil et parfois autonomes) communiquent et coopèrent potentiellement entre eux et avec le réseau pour accomplir des tâches de stockage et de traitement sans l'intervention de tierces parties. Ces tâches peuvent être destinées à prendre en charge les fonctions réseau de Fog computing de base ou les nouveaux services et applications qui s'exécutent dans un environnement en bac à sable (sandboxed environment) [30].
Verma et al. [31] Proposent un algorithme d'équilibrage de charge efficace pour une architecture Fog-Cloud. L'algorithme utilise une technique de réplication pour le maintien des données dans les réseaux Fog qui réduit la dépendance globale sur les grands centres de données. Le but ultime est d'équilibrer la charge par les réseaux Fog et de rendre l'Internet moins dépendante du Cloud en ayant les données disponibles plus proches de l'utilisateur. Tout client crée un paquet IP via un réseau Internet avec l'adresse du serveur Fog voisin. Les serveurs du niveau

Fog fonctionnent comme un middleware et envoient les données de l'utilisateur au serveur Cloud le plus proche pour l'authentification.
Une fois que l'utilisateur s'est authentifié sur le serveur Cloud, une connexion sécurisée est établie à travers laquelle les paquets cryptés sont distribués entre le client et les niveaux supérieurs. Les serveurs présents au niveau du Fog doivent être surveillés de manière cohérente par leurs Fog Server Managers (FSM)respectifs. Les FSMs interagissent systématiquement avec les FSMs de tous les autres serveurs des réseaux en pointe (Edge networks) comme le montre la figure 3.1. Le serveur capable de gérer la charge, au bon moment est choisi, s'il n'est pas capable de fournir les données et les ressources elles-mêmes, la requête est transmise au serveur suivant pour traiter la demande dans le réseau Edge respectif ou un réseau Edge différent.
Lorsque le serveur Fog ne parvient pas à fournir les données à l'utilisateur, le paquet migre vers le niveau supérieur à travers la connexion établie avec l'adresse du serveur Fog actif. Le serveur Cloud actif gérera la demande et la fournira avec les données disponibles. Dans le dernier scénario possible, si le serveur Cloud ne peut pas non plus traiter la requête, il diffuse des paquets de requêtes dans tout le niveau Cloud. Le serveur avec les données requises pour traiter la demande accusera réception du serveur Cloud initialement actif. Le serveur Cloud nouvellement actif au lieu d'envoyer des données aux serveurs Cloud précédents envoie des données directement au serveur Fog actif au milieu. Par conséquent, la surcharge d'envoi de données vers un autre serveur Cloud est ignorée et les données sont également répliquées dans le niveau Fog pour des requêtes futures similaires simultanément. Les serveurs Fog atteignent enfin le client authentifié avec le paquet de réponse, et la connexion sécurisée entre les trois niveaux est terminée.

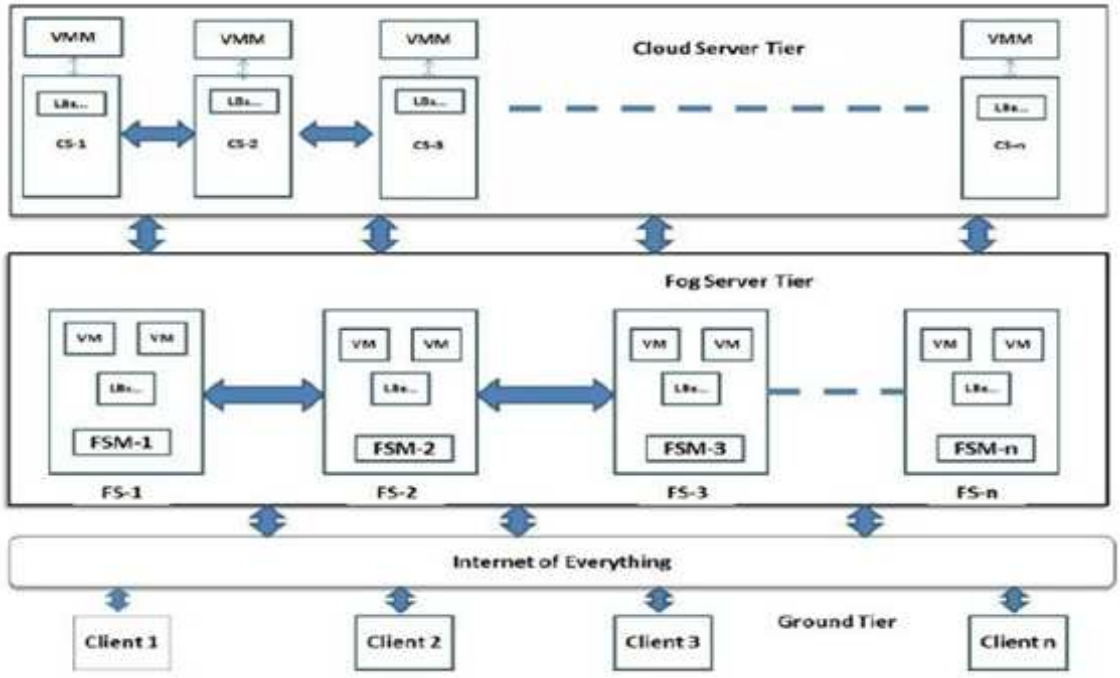

FIGURE 3.1 – Architecture proposée utilisant le Fog computing [3].

3.1.4 Approche basée sur la réplication par bloc de fichiers (HDFS)

Dans l'article [32] les auteurs présentent une politique de réplication de données pour un Cloud de calcul avec des données bio-informatiques, de sorte que l'exécution de l'application de la bio-informatique est réduite, en utilisant l'Apache avec le système d'archives distribuées.
Le système de fichiers distribué Hadoop (HDFS) [33], permet aux ordinateurs communs de faire partie du cluster, ce qui permet de calculer avec des machines hétérogènes à faible coût.
Le HDFS utilise la réplication par blocs, ou plutôt, en interne un fichier est divisé en un ou plusieurs blocs et chaque bloc dans le HDFS est copié sur d'autres ordinateurs. La politique de réplication des données HDFS met en priorité l'équilibrage et répartit aléatoirement les copies des blocs de données dans le système de fichiers en fonction

de la capacité de stockage de chaque noeud de données.
Pour respecter les critères d'organisation du fichier et pouvoir utiliser correctement la politique de réplication, il a fallu compléter quelques adaptations dans les systèmes de fichiers qui peuvent expliquer ce besoin, ces adaptations ont été divisées en quatre phases :
Phase 1 : Les blocs de données dans le système de fichiers ne sont pas placés directement dans le HDFS, car si les fichiers avaient été stockés directement, il n'aurait pas été possible de prévenir la fin du bloc de données, ce qui pourrait entraîner une rupture de la séquence de filtrage. Au lieu de cela, les blocs sont traités par une procédure avant d'être insérés dans la base de données. Cette procédure détermine dans quelle partie du fichier chaque bloc doit être inséré.
Phase 2 : Comme chaque partie du fichier est devenue un bloc distinct, il est nécessaire de connaitre l'emplacement de chaque bloc et, par conséquent, le choix des ordinateurs qui participeront au traitement du fichier. Dans ce but, un mappage est appliqué dans les ordinateurs possédant le fichier pendant la phase de création des copies. Ce mappage est simplement constitué d'un fichier contenant l'identificateur de bloc et le lieu où chacun a été copié. Une fois l'identification terminée, l'exécution des tâches est initiée en se référant aux blocs par diffusion.
Phase 3 : Malgré que le Hadoop dispose d'un système de fichiers multiplateformes, le programme d'allocation de ressources a toujours une difficulté de connaître l'état actuel du système. Pour remédier à cela, un programme d'allocation de ressources a été utilisé, par exemple, un ordinateur qui a quatre coeurs peut être fixé pour exécuter au plus huit tâches en même temps, de façon standard, mais selon l'activité planifiée dans le nuage, une tâche créée peut utiliser différents noyaux du processeur et le traitement de deux tâches peut être mis en péril. De même, huit tâches qui n'exigent pas toute la capacité de traitement de l'ordinateur peuvent être exécutées et il peut y avoir une tâche qui peut utiliser les ressources disponibles sans compromettre certains des autres processus, mais cette tâche devra attendre dans la ligne d'exécution.
Phase 4 : La phase finale est l'allocation des fichiers nouvellement générés après le traitement. Cette phase complète la première étape, pour les fichiers créés, générant les journaux de traitement et complétant le mappage décrit dans la deuxième phase.

3.1.5 Approche basée sur la réplication et le placement dynamique des répliques

Karuppusamy et al. [34] proposent un algorithme de réplication et de placement dynamique pour améliorer les performances du gestionnaire du Cloud. Ils calculent le degré de popularité et le facteur de réplication pour identifier le fichier approprié à répliquer. Ensuite, l'algorithme proposé est utilisé pour placer les répliques à l'endroit approprié. L'approche proposée se compose de trois phases : sélection de la réplique en utilisant le degré de popularité (DP), Création de réplique en utilisant le facteur de la réplique (FR) et mise en place de la réplique. Dans la première phase ; le fichier à répliquer est trouvé en calculant le DP. Dans la deuxième phase ; les répliques sont créés à l'aide du DP qui dépend de la fréquence d'accès d'un utilisateur, de plusieurs utilisateurs et des documents demandés auxquels sont associés des poids suivant leur importance. Le degré négatif (DN) dépend de l'espace mémoire utilisé par ce document, du nombre de réplique et du temps de réponse à une requête vers ce document.

a. **Sélection des répliques :**
   Pour sélectionner une réplique, un fichier de données populaire est identifié en analysant les historiques d'accès et en définissant des poids différents pour différentes données accédées. Fondamentalement, les données les plus récemment consultées sont plus pertinentes à l'analyse, et sont donc placées à une priorité plus élevée.
   Le nombre de répliques est calculé à l'instant t comme suit :

   $$\text{Le nombre de répliques} = \begin{cases} RF_{t+2} * RF_{t-1} & si\ RF_t > RF_{t-1} \\ 0 & si\ RF_t < RF_{t-1}. \end{cases}$$

b. **Création des répliques :**
   Pour créer la réplique, un facteur de réplication (FR) est utilisé. Le FR est déterminé en calculant le facteur positif (FP) qui lui-même dépend des degrés de popularité courant, minimal et le facteur négatif (FN) qui est calculé en fonction des degrés courant, minimal et maximal. Le but de FP est d'identifier l'importance d'un fichier pour la réplication. FN d'un fichier spécifie si ce

dernier ne doit pas être répliqué.

$$PF = \frac{DP_{current} - DP_{min}}{DP_{max} - DP_{min}} \tag{3.1}$$

$$NF = \frac{ND_{current} - ND_{min}}{ND_{max} - ND_{min}} \tag{3.2}$$

c. **Placement des répliques :**
Le placement des répliques (obtenues de l'étape précédente) a pour objectif d'assurer la disponibilité souhaitée avec des répliques minimales sans dégrader les performances du système. Cet objectif est possible avec une stratégie de placement des répliques qui prend en compte : la disponibilité souhaitée, la stabilité des noeuds dans le système et les échecs. Pour cela, ils doivent d'abord considérer la liste des noeuds dans le centre de données. Ensuite, ces noeuds sont disposés en ordre décroissant en utilisant les critères de classification. Le placement de plusieurs répliques de mêmes données dans le même noeud n'améliore pas la disponibilité ou la tolérance aux pannes. Pour cette raison, il sera utile de stocker une seule réplique des mêmes données dans un noeud. Dans le système étudié, il est possible de prédire les échecs dans le noeud. Ainsi, en cas de défaillance suspecte, le noeud place toutes ses données dans un autre noeud pour assurer la disponibilité souhaitée.

Après avoir calculé les critères de classification et les avoir classés par ordre décroissant, les auteurs vérifient si le fichier répliqué peut être placé dans le noeud ou non. Si possible, stocker le fichier répliqué dans le noeud correspondant ou passer au noeud suivant et vérifier si ce noeud satisfait la condition. Ce processus se poursuit jusqu'à ce que toutes les répliques soient placées. Pour chaque noeud, il vérifie dans la liste des noeuds liste si l'ajout de la réplique augmente la non-similitude du noeud. Si c'est le cas, il stocke les données dans le noeud sinon il teste le noeud suivant dans la liste.

3.1.6 Approche basée sur la réplication adaptative

Dans l'article [35] une stratégie adaptative de réplication de données, qui est basé principalement sur une sélection dynamique des fichiers de données répliquées a été présentée, de façon à améliorer la fiabilité globale du système et de répondre aux exigences de qualité des services. De plus, la stratégie proposée détermine dynamiquement le nombre de répliques ainsi que le nombre de noeuds à utiliser.
La méthode de réplication analyse les fichiers de données populaires et le modèle récent des demandes puis en s'appuyant sur l'utilisation d'une technique de série chronologique légère (une série chronologique permet d'étudier l'évolution d'une variable dans le temps). On présente une série chronologique au moyen d'un tableau dans lequel la première colonne représente le temps et les autres colonnes la variable qui évoluent en fonction du temps. L'approche proposée identifie à chaque fichier de données les besoins de réplication parmi lesquels on cite : le nombre de répliques pour chaque donnée sélectionnée, la position des nouvelles répliques sur les centres de données disponibles et enfin la surcharge de la stratégie de réplication sur l'infrastructure du Cloud.
La réplication de données adaptative proposée comporte trois phases importantes : 1) Quels fichiers de données doivent être sélectionnés pour la réplication ? ; 2) Définir le nombre de répliques appropriées pour répondre aux QoS spécifiées ; 3) Déterminer le meilleur emplacement des répliques. La première étape consiste à déterminer quelles données répliquer et le temps de réplication. La phase de sélection analyse l'historique des demandes d'accès aux données et utilise une technique de série chronologique légère pour prédire la future fréquence d'accès des données. Si les demandes futures prévues de fragments de données dépassent un seuil adaptatif, les blocs de données seront sélectionnés pour la réplication. Soit pd_k le degré de popularité d'un bloc b_k. pd_k est défini comme la fréquence d'accès future basée sur le nombre de demande d'accès, $an_k(t)$ à l'instant t, le degré de popularité pd_k d'un bloc b_k peut être calculé à l'aide du Lissage Linéaire et exponentielle de Holt (HLES). Le lissage linéaire et exponentiel de Holt (HLES) est une technique de prédiction de la série chronologique à faible coût. HLES est sélectionné pour sa capacité de lissage et fournit des prévisions à court terme pour les taux d'arrivée des demandes mesurées et les

taux de demande de service. Par conséquent, HLES permet au framework proposé de surveiller les taux d'arrivée et les taux de service et de prévoir à court terme les taux d'arrivée futurs et les taux de service avec un faible temps de calcul. HLES lisse la série chronologique et fournit une prévision à court terme basée sur la tendance qui existe sur les séries chronologiques [13]. Supposons que $an_k(t)$ soit une valeur de série chronologique à l'instant t. La prévision linéaire pour les m étapes à venir est la suivante :

$$pd_k = an_k(t+m) = L_t + b_t * m \tag{3.3}$$

Où L_t et b_t sont des estimations exponentiellement lissées du niveau et de la tendance linéaire de la série à l'instant t :

$$L_t = \alpha an_k(t) + (1-\alpha)(L_{t-1} + b_{t-1}) \tag{3.4}$$

Avec α est un paramètre de lissage, $0 < \alpha < 1$,

$$b_t = \beta(L_t - L_{t-1}) + (1-\beta)b_{t-1}) \tag{3.5}$$

Où β est un coefficient de tendance, $0 < \beta < 1$.
Une grande valeur de α ajoute plus de poids aux valeurs récentes plutôt qu'aux valeurs précédentes afin de prédire la valeur suivante. Une grande valeur de β ajoute plus de poids aux variations du niveau que la tendance précédente.
Le facteur de réplication est défini comme la moyenne du ratio du degré de popularité et de la disponibilité moyenne des répliques sur les différents noeuds de données de tous les blocs l du fichier de données f_i. Il sert à déterminer si le fichier de données f_i doit être répliqué.

3.1.7 Approche basée sur les graphes

Les auteurs de l'article [36] présentent un projet qui implémente la répartition et la réplication partielle des données dans le Cloud, afin d'optimiser la performance et la sécurité des fragments de données qui sont téléchargés par leurs propriétaires, et mettre en oeuvre une approche graphique qui calcule les distances pour prédire les

emplacements des répliques. L'approche proposée est très utile au propriétaire pour protéger ces données des attaquants.
Ensuite, ils étendent leur approche pour vérifier la cohérence dans le système Cloud lors de la mise à jour du fichier. Les auteurs proposent également une stratégie d'audit heuristique (HAS : Heuristic Audit Strategy) qui ajoute des lectures appropriées pour révéler autant de violations que possible, en utilisant la table d'opération de l'utilisateur. Chaque utilisateur conserve une UOT (User Operation Table) pour enregistrer les opérations locales.
La méthode de T-Coloring [37] est utilisée pour placer les fragments dans divers fournisseurs. Ceci est utilisé pour l'attribution de canal en assignant des canaux aux noeuds, de sorte que les canaux sont séparés par une distance pour éviter l'interférence. Cette méthode interdit le stockage du fragment dans le voisinage d'un noeud stockant un fragment, aboutissant à l'élimination d'un certain nombre de noeuds à utiliser pour le stockage. Dans ce cas, uniquement pour les fragments restants, les noeuds qui ne contiennent aucun fragment sont sélectionnés pour être stockés de manière aléatoire.

3.2 Critères de comparaisons des méthodes de réplication étudiées

- **Réplication partielle :** Il s'agit de répliquer une partie des données au lieu de la donnée entière.
- **Réplication totale :** Réplication entière de la données.
- **Réplication synchrone :** Garantir que lorsqu'une transaction met à jour une réplique primaire, toutes les répliques secondaires sont mises à jour dans la même transaction.
- **Réplication active :** Lors de réplication active, les calculs effectués par la source (ou maître) sont répliqués, c'est à dire que le noeud principal propage le calcul vers les noeuds secondaires.
- **Réplication dynamique :** Les copies de données sont crées et supprimées automatiquement selon l'évolution des demandes des utilisateurs.

– **Disponibilité :** C'est la capacité d'assurer l'accessibilité au données durant une période donnée.

3.3 Comparaison des méthodes de réplication étudiées

	Approche basée sur les graphes [36]	Approche basée sur la réplication et le placement dynamique des répliques [34]	Approche basée sur la réplication par bloc de fichiers [32]	Approche basée sur les réseaux de neurones [27]	Approche basée sur la réplication adaptative [35]	Approche basée sur l'économie d'énergie [29]	Approche basée sur le Fog computing [31]
Réplication partielle	√	×	√	√	×	×	×
Réplication totale	×	√	×	×	√	√	√
Hétérogène	√	√	√	√	√	√	√
Réplication synchrone	-	-	×	-	-	×	-
Réplication active	×	-	√	√	-	√	-
Réplication dynamique	-	√	-	√	√	-	-
Disponibilité	√	√	√	√	√	√	√

TABLE 3.1 – Tableau comparatif entre les méthodes de réplication étudiées.

3.4 Synthèse

Dans les environnements Cloud, pour fournir et gérer un service extrêmement disponible, Santhi et al. [36] Présentent une stratégie de réplication partielle basée sur les distances pour prédire les emplacements des répliques, la méthode proposée a un moindre coût et plus efficace pour cause de réplication partielle et l'ajout

d'un système d'audit heuristique qui lance plusieurs lectures pour révéler des incohérences éventuelles des répliques. Da Silva et Araujo [32] ont développé une politique de réplication en utilisant Hadoop, ce dernier permet aux ordinateurs communs de calculer avec des machines hétérogènes ce qui permet de réduire le coût et d'assurer une efficacité élevée. Hadoop utilise la réplication par blocs, et chaque bloc est copié sur d'autres ordinateurs et ces copies sont réparties aléatoirement dans le système de fichiers en fonction de la capacité de stockage de chaque noeud de données, ce qui permet d'acroitre la disponibilité de ces dernières. Dans ce contexte, Vaquero et Merino [30] ont proposé un algorithme d'équilibrage de charge qui utilise une technique de réplication pour le maintien des données dans les réseaux Fog afin d'équilibrer la charge et rendre Internet moins dépendante du Cloud en ayant les données disponibles plus proche des utilisateurs. L'approche proposée assure la disponibilité avec un moindre coût. Hussein et Moussa [35] ont adopté une stratégie adaptative de réplication basée sur une sélection dynamique des fichiers de données répliquées, pour améliorer la fiabilité et la disponibilité globale du système, dans cette méthode la réplication analyse les fichiers de données populaires et les demandes récentes, puis en s'appuyant sur les séries chronologiques légères, elle sélectionne les données à répliquer. Karuppusamy et Muthaiyan [34] proposent un algorithme de réplication et de placement dynamique, cet algorithme sert à placer les répliques à l'endroit approprié, leur approche se compose de trois phases : une phase de sélection de la réplique, une phase de création, et une phase de placement de la réplique. Boru et al. [29] Ont présenté une stratégie de réplication à faible consommation d'énergie pour améliorer la qualité de service et réduire les délais de communication entre les centres de données, ils ont également proposé un algorithme de réplication qui prend en considération la consommation d'énergie et la bande passante requises pour l'accès aux données.

3.5 Conclusion

Ce chapitre représente un diaporama des méthodes de réplication existantes dans la littérature. Le chapitre qui suit est notre contribution à la gestion du problème de réplication dans les environnements Cloud computing. Nous adressons le problème

de la sélection des données à répliquer, la stratégie mise en oeuvre pour la création de ces répliques, ainsi que leur placement stratégique. Nous allons ensuite procéder à l'évaluation des performances et la simulation des résultats obtenus.

Chapitre 4

Proposition pour la réplication des données dans le Cloud computing

Pour de nombreuses applications scientifiques ayant recours aux systèmes à large échelle, les quantités de données nécessaires aux traitements de ces applications sont très importantes. De telles quantités de données impliquent de très grands délais de transfert si seule une copie de ces données est stockée. De là, le temps de réponse et la disponibilité des données deviennent les défis principaux à adresser. Afin de répondre à ces défis, une technique importante est de répliquer les données dans différents sites, de sorte qu'un utilisateur puisse accéder aux données d'un site de sa proximité.
Dans ce chapitre, nous allons proposer une stratégie pour la gestion de la réplication, qui permet de garantir la cohérence des donnée dans le Cloud computing en nous basons sur les travaux effectués dans [9], nous allons définir des protocoles d'écriture, de lecture, de stockage et de réplication qui garantissent également la disponibilité et améliorent le temps d'accès aux données.

4.1 Problématique

Le Cloud computing exige la gestion d'un nombre massif de données afin de les rendre accessible aux clients qui les réclament. Les stratégies de réplication de

données représentent un facteur important vis-à-vis des performances des services proposés le Cloud. Ces stratégies doivent répondre à des questions essentielles tellesque : 1) Quoi répliquer ? La sélection des données à répliquer dépend de leur degré depopularité et du taux de sollicitation par les utilisateurs. 2) Où répliquer ? Placer les nouvelles répliques sur le site de l'emplacement approprié peut favoriser la réduction de la consommation de bande passante du réseau et réduit donc le temps de réponse.
L'idée principale est de conserver les données à proximité de l'utilisateur afin de rendre l'accès efficace et rapide. Nous nous sommes basés sur des travaux ultérieurs qui traitent de la cohérence de donnée [9] ainsi que de la disponibilité [10] pour proposer un protocole de réplication qui permet de répondre aux questions précédentes et de garantir la cohérence de données, la disponibilité, la fiabilité et qui améliore le temps d'accès aux données.

4.2 Architecture du Cloud Proposé

L'architecture du Cloud est composée de clusters disjoints de datacenters. Chaque cluster est supervisé par un ClusterHead (CH) et est composé des quorums de lecture et d'écriture et illustrée sur la figure 4.1.

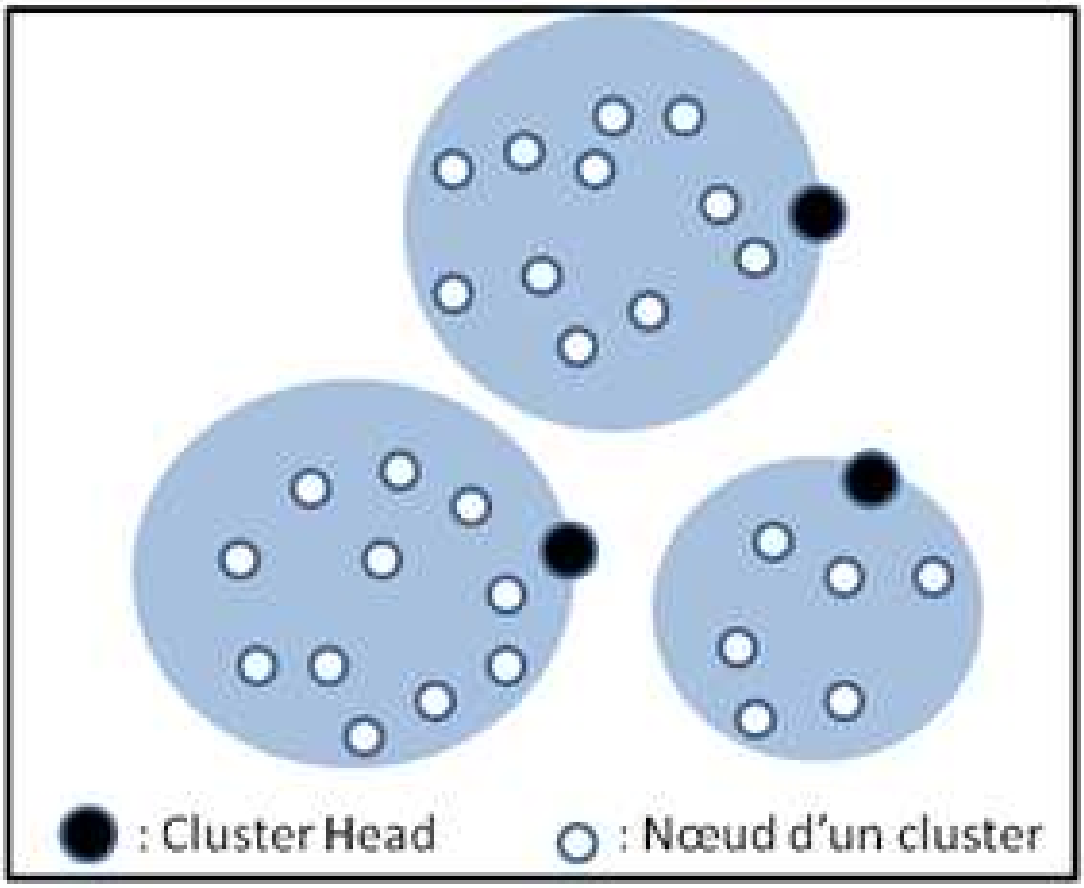

FIGURE 4.1 – Architecture du Cloud proposée.

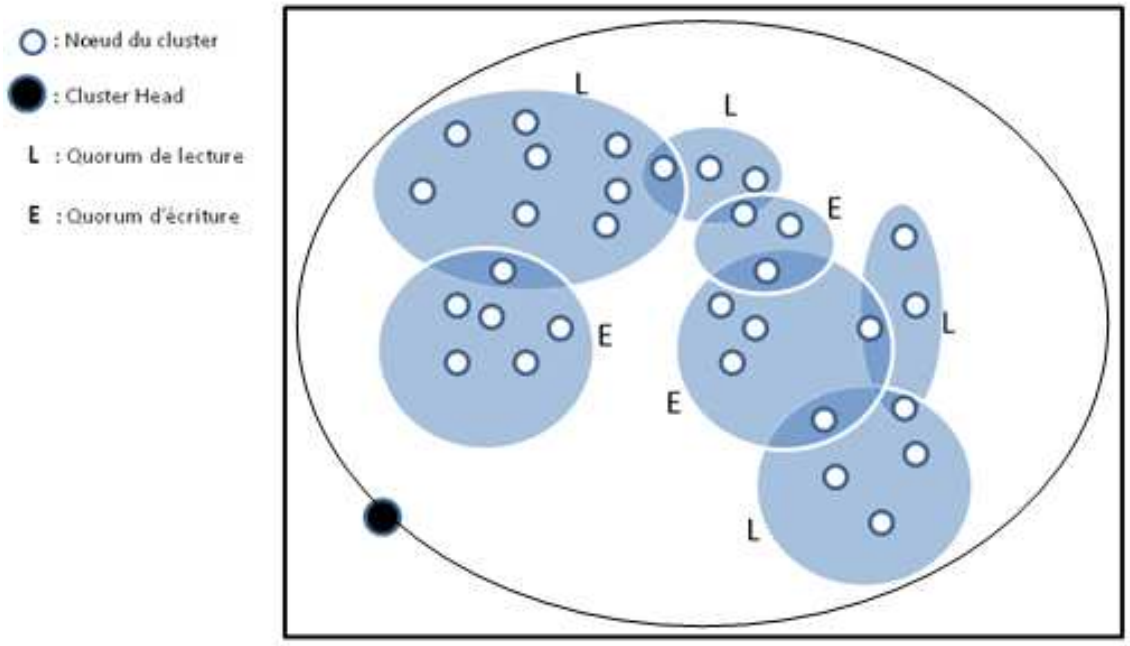

FIGURE 4.2 – Zoom sur un cluster.

4.3 structuration des données

Au lieu de traiter les données une à une, nous proposons de les regrouper sous une matrice dynamique i*c, tel que i représente le nombre de lignes qui croit suivant les requêtes de stockage ou de réplication, et c le nombre de colonnes qui reste fixe (mais dont la taille est dynamique). Chaque cellule de la matrice est un vecteur de 5 éléments, tels que : le premier concerne l'identifiant de la donnée, le deuxième est sa valeur, le troisième sa date, le quatrième sa popularité et la cinquième la date du dernier calcul de la popularité. La matrice se remplie ligne par ligne lors d'une requête de stockage ou de réplication. Id-Donnée est l'identifiant de la donnée, Val-Donnée, la valeur de cette donnée, Date, son estampille, Popularité, le nombre de fois ou cette donnée a était sollicité pendant ces 30 derniers jours et Date-Popularité, la date du dernier calcul de la popularité, voir Table 4.1.

Chaque CH conserve l'intégralité des requêtes qui concernent les données afin de

\	colonne 1			
id_donnée	val_donnée	estampille	popularité	date_popularité
data 1	66	765127	1	13/04/2017

TABLE 4.1 – Exemple de structure matricielle proposée.

pouvoir calculer la popularité de chaque donnée après un délais de 30 jours, au-delà de ce délais l'historique est supprimé. L'historique est sauvegardé comme un vecteur de 5 éléments, où ID-Requête, l'identifiant de la requête, Date-Requête, la date ou cette requête a été généré la première fois, Type-requête, le type de cette requête (requête de stockage, de lecture, ou d'écriture, ID-Donnée, l'identifiant de la donnée sollicité et Loc-noeud la localisation du n ?ud sollicité par le client qui introduit la nouvelle donné, voir Table 4.2.

ID_Requête	Date_Requête	Type_Reqiête	ID_Donnée	Loc_Noeud
ReqAK34	19/04/2017	ReqL	data1	noeudRsx

TABLE 4.2 – Exemple de structure proposé pour la sauvegarde de l'historique.

4.4 Construction des quorums

Les quorums sont construit de façon statique, tel que :

- L'intersection entre un quorum de lecture et un quorum d'écriture n'est jamais vide.
- Dans chaque intersection, il y a un noeud spécial appelé coordonateur de quorum.
- Chaque quorum de lecture ou d'écriture est muni d'un coordonateur.
- Chaque quorum de lecture ou d'écriture contient toutes les données au moins une fois.

Remarques

- Chaque noeud d'un quorum connait son coordonateur.
- Chaque coordonateur connait les nouds de son quorum ainsi que leurs emplacements géographiques.

– Chaque CH connait tous les coordonateurs de son cluster.

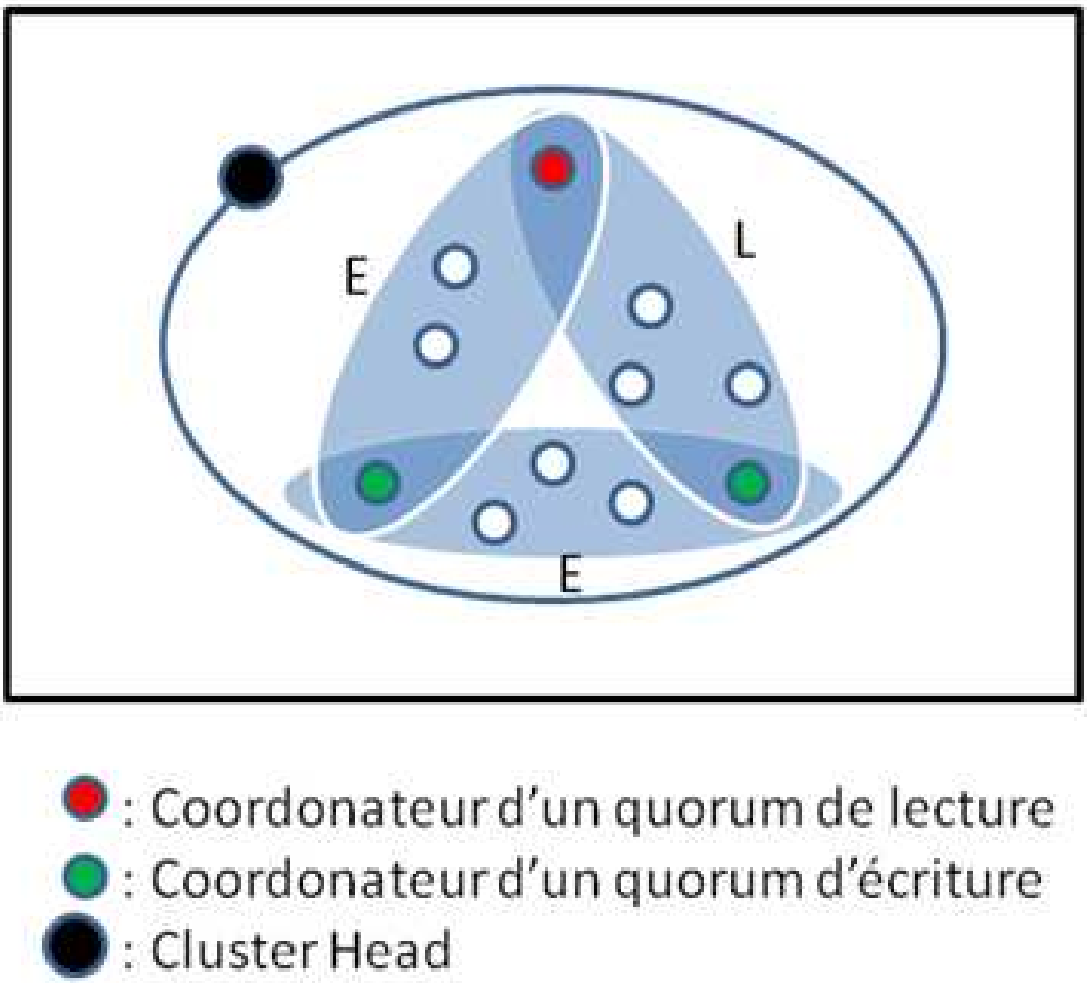

FIGURE 4.3 – Zoom sur l'intersection de quorums dans un Cluster.

4.5 Protocole de stockage

1. Quand un client introduit une nouvelle donnée x, il sollicite le n ?ud le plus proche en envoyant ReqS (Id-Donnée,Val-Donnée), où ReqS représente une requête de stockage.

2. Ce noeud redirige cette requête ReqS (ID-Donnée,Val-Donnée,Date, Loc-Client,Loc-noeud) vers CH.
3. CH envoi ReqS (ID-Donnée,Val-Donnée,Date,Loc-noeud) vers les coordonateurs des quorums et vers les autres CH.
4. En recevant ce message, les CHs diffusent ce message vers les coordonateurs de quorums de leurs clusters. Chaque coordonateur désigne le noeud le plus proche géographiquement de son quorum du pour effectuer la sauvegarde.
5. Le noeud désigné reçoit la requête de stockage, stocke la donnée et renvoi un ACK à son coordonateur.
6. Le coordonateur attend un ACK du noeud désigné sinon désigne un autre noeud géographiquement proche du client et exécute de nouveau le point (6).
7. Le coordonateur reçoit un ACK du noeud désigné le transmet au CH.
8. CH, après avoir reçu au moins un ACK, il le renvoi au client.
9. CH exécute une sauvegarde dans l'historique (X, 12/03/2017, ReqS, data1).

Remarque
L'historique contient une sauvegarde de toutes les requêtes d'écriture, de lecture et de stockage dans le réseau afin d'obtenir la popularité des données (plus une donnée est sollicitée, plus elle est populaire) afin de les répliquer pour améliorer la fiabilité, la tolérance aux pannes, la disponibilité des données ainsi que le temps de réponse. La sauvegarde s'effectue dans une base de donnée centralisée dans CH.

4.6 Protocole de lecture

1. Le client envoi une requête de lecture ReqL (Lec, ID-Donnée, estampille, ID-client) au noeud le plus proche, tel que : Lec : désigne une requête de lecture.
2. A la réception de la ReqL :
   2.1. Si le noeud récepteur n'est pas un coordonateur, alors il redirige la ReqL vers son coordonateur et passe au point (2.2).
   2.2. Si le noeud récepteur est un coordonateur, alors il exécute l'un des points

2.3, 2.4.
2.3. S'il est coordonateur d'un quorum de lecture, il redirige la ReqL vers le CH et attend une confirmation.
2.4. S'il coordonne un quorum d'écriture et n'appartient à aucun quorum de lecture, il redirige la ReqL vers le CH.

3. Lorsque CH reçoit la ReqL :
   3.1. (Dans le cas du point 2.3) : s'il n'y a pas de requête d'écriture en cours sur la donnée sollicité, CH envoie un ReqL + ACK au coordonateur sinon il attend la fin de l'écriture pour l'envoyer.
4. (Dans le cas du point 2.4) : CH désigne un quorum de lecture proche, puis envoi la ReqL accompagnée d'un ACK vers sont coordonateur. A la réception de l'ACK + ReqL le coordonateur diffuse la ReqL vers les noeuds de son quorum.
5. Chaque noeud, y compris le coordonateur, vérifie s'il dispose de la donnée sollicité.
   5.1. Si le noeud dispose de la donnée, il l'envoie vers le coordonateur un ACK, la Date de la donnée(Date-Donnée), sa Popularité, la Date-Popularité.
   5.2. Sinon, il envoi un NACK.

6. Le coordonateur, lorsqu'il reçoit la totalité des réponses des n ?uds de son quorum (ACK et NACK), il renvoi au CH les informations concernant la donnée avec la date la plus récente.
7. CH répond au coordonateur avec une requête d'affichage, pour indiquer au noeud ayant la date la plus récente et la localisation la plus proche du client de transmettre la donnée au client.
8. CH exécute le Protocole de Sauvegarde de l'historique.
   8.1. Si (Date-Popularité - date actuelle > 30) date ct alors il exécute le protocole de réplication.
9. Le Protocole de Réplication.

4.7 Protocole d'écriture

1. Le client envoi une requête d'écriture ReqE (Ecr, ID-Donnée, Val-Donnée, ID-client) au noeud le plus proche, tel que, Ecr : est une requête d'écriture.
2. Lors de la réception de la ReqE : 2.1. Si le noeud récepteur n'est pas un coordonateur, alors on passe aux points 2.1.1 et 2.1.2.
   2.1.1. Si le noeud récepteur appartient à un quorum de lecture, alors il redirige la ReqE vers son coordonateur et passe au scénario 2.2.1
   2.1.2. Si le noeud récepteur appartient à un quorum d'écriture, alors il exécute l'écriture s'il dispose de la donnée, puis redirige la ReqE suivie du résultat (ACK ou NACK) vers son coordonateur.
   2.2. Si le noeud récepteur est un coordonateur, alors il exécute l'un des points 2.2.1 ou 2.2.2.
   2.2.1. Si le noeud récepteur coordonne un quorum de lecture, il redirige la ReqE vers le coordonateur du quorum d'écriture et passe au scénario 2.2.2.
   2.2.2. Si le noeud récepteur coordonne un quorum d'écriture :
   (a) Il redirige la ReqE vers CH et vers les noeuds de son quorum, sauf le noeud récepteur.
   (b) Chaque n ?ud, y compris le coordonateur, exécutent l'écriture (la mise à jour) s'il dispose de la donnée sollicité et envoi un ACK. Sinon il envoi un NACK.
   (c) En parallèle, CH diffuse un message de supervision vers les coordonateurs de quorums de son cluster (sauf celui qui a envoyé la ReqE) et vers les autres CHs.
   (d) A la réception de ce message, les CHs le diffusent vers tous les coordonateurs et quorums de leur cluster.
   (e) A la réception de ce message, chaque coordonateur et le diffuse aux n ?uds de son quorum et tous les noeuds exécutent l'écriture.
3. CH exécute le Protocole de Sauvegarde de l'historique.
   3.1. Si (Date-Popularité - la date actuelle > 30) alors il exécute le protocole de

calcul de la popularité qui renvoi la nouvelle popularité (nouv-popularité) de la donnée.
3.2. Si (nouv-popularité > popularité) alors il exécute le protocole de réplication.

Remarques

- Si un n ?ud dispose de la donnée sollicité vérifie si la date de la nouvelle requête est supérieure à celle qui est stockée avant d'exécuteur l'écriture.
- Le coordonateur, s'il réussit à écrire la donnée envoi directement un ACK au CH, sinon il l'envoi au premier ACK reçu.
- Si un client ne reçoit pas de confirmation à sa requête après un certain délai, il sollicite un autre noeud qui lui est proche ainsi le processus est recommencé.

4.8 Protocole de réplication

C'est un protocole exécuté après la vérification des conditions du point (9) pour le protocole de lecture et (3) du protocole d'écriture.

1. C'est un protocole exécuté après la vérification des conditions du point (9) pour le protocole de lecture et (3) du protocole d'écriture.
2. CH calcul la nouvelle popularité de la donnée (nouv-popularité).
3. Si (nouv-popularité > popularité) alors il recherche le n ?ud qui a le plus sollicité la donnée et renvoi sa localisation (Loc-noeud) puis exécute le protocole de stockage ReqS (ID-Donnée, Val-Donnée, estampille, nouv-poplarité, nouv-date-popularité,Loc-noeud), le protocole de stockage se charge de stocker la nouvelle donnée au plus près des clients qui l'a le plus sollicité.
4. CH diffuse un message d'écriture en faisant appel au protocole d'écriture pour mettre à jour la nouvelle popularité de la donnée ainsi que la nouvelle date de calcul de cette dernière.
5. CH effectue une suppression de toutes les requêtes qui concernent cette requête dans ' ?historique.

Remarque

La réplication est facturée au client, ainsi ce dernier peut consulter la popularité de ses données et exiger la suppression de l'une des répliques de ses données si cette dernière n'est pas assez sollicitée.

4.9 Caractéristiques de notre proposition

Le tableau 4.3 représente les caractéristiques de notre proposition.

Réplication partielle	Réplication totale	Hétérogénéité	Réplication synchrone	Réplication active	Réplication dynamique
√	×	√	×	×	√

TABLE 4.3 – Caractéristiques de notre proposition.

4.10 Défaillances lors de la lecture

- Dans le cas où le délai d'attente de CH est expiré ou qu'il ne reçoit que des NACKs, il désigne un autre quorum de lecture et ainsi la procédure de lecture est recommencée.
- Si CH a déjà envoyé la requête à tout les coordonateurs, il envoi la requête au CH le plus proche de lui qui s'occupe de répondre au client.

4.11 Défaillances lors de l'écriture

- Lorsque le délai d'attente de CH expire, il diffuse une interrogation vers tous les coordonateurs de quorums, au premier ACK qu'il reçoit il confirme l'écriture pour le client.
- Si CH ne reçoit aucune réponse, il sollicite un autre CH qui se charge de l'écriture et de répondre au client.

4.12 Vérification des propriétés

- La solution tolère plusieurs lectures simultanées sur une même donnée, une lecture ne bloque pas une autre.
- La solution tolère plusieurs écritures simultanées sur une même donnée, une écriture ne bloque pas une autre.
- Le comportement prévu au départ est toujours assuré, si la donnée sollicitée existe, elle est toujours retournée au client.
- La solution permet de garantir la disponibilité de la donnée grâce à la réplication.
- La solution optimise le temps de réponse en stockant les répliques au plus près des clients qui la sollicitent.
- Lorsqu'une panne se produit sur un noeud de notre réseau, la donnée est récupérée dans un emplacement plus éloigné du client, le temps remettre en service le noeud défectueux.

4.13 Conclusion

Dans ce chapitre, nous avons proposé une solution qui permet de garantir la cohérence des donnée dans le Cloud computing en nous basons sur les travaux effectués dans [9], nous avons défini des protocoles d'écriture, de lecture, de stockage et de réplication qui garantissent également la disponibilité et améliorent le temps d'accès aux données.

Conclusion et perspectives

A travers ce mémoire, nous avons étudié le problème de réplication de données dans le Cloud computing. Plusieurs stratégies de réplication ont été proposées dans la littérature : elles visent principalement à créer et à placer de manière efficace les répliques afin qu'un site puisse trouver les données nécessaires à l'exécution de sa requête. Les approches sur lesquelles se base ces stratégies diffèrent pour pouvoir offrir un compromis entre les objectifs conflictuels de la réplication, à savoir, la réduction du temps d'accès aux données, l'augmentation du niveau de disponibilité et la tolérance aux pannes.

Le problème de réplication de données dans le Cloud computing est assez complexe, il présente plusieurs défis, tels que la dynamicité des sites, la capacité de stockage et la surcharge de la bande passante. Une stratégie efficace de réplication doit s'adapter aux changements du Cloud et au comportement dynamique des utilisateurs en évitant de nuire aux performances du système en le surchargeant de transfert de données superflu.

D'autre part, le choix de l'architecture du Cloud est un facteur crucial dont dépendent les performances attendues du système. Dans ce contexte nous avons proposé sur une solution qui permet de garantir la cohérence des donnée dans le cloud computing en nous basons sur les travaux effectués dans [8], nous avons défini des protocoles d'écriture, de lecture, de stockage et de réplication qui garantissent également la disponibilité et améliore le temps d'accès aux données. Ce travail a permis d'ouvrir quelques perspectives intéressantes comme la validation de la proposition et sa simulation et son évaluation de performances.

Bibliographie

[1] Pascal Faure, Gabrielle Gauthey, Marie-Caroline Bonnet-Galzy, Guide sur le Cloud Computing et les Datacentersà l'attention des collectivités locales, (Juillet 2015).

[2] https ://fr.wikipedia.org/wiki/Cloud_computing and Applications (consulté le 23/11/2016).

[3] K. C. Gouda, AnuragPatro, Dines Dwivedi, NagarajBhat, Virtualization Approaches in Cloud Computing, in International Journal of Computer Trends and Technology (IJCTT), vol 12 Issues 4, (Juin 2014).

[4] Puja Dhar, Cloud computing and its applications in the world of networking, in IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 1, No 2, (Janvier 2012).

[5] Michael Behrendt, Bernard Glasner, Petra Kopp, Robert Dieckmann, Gerd-Breiter, Stefan Pappe, Heather Kreger, Ali Arsanjani, Introduction and Architecture Overview IBM Cloud Computing Reference Architecture 2.0, CCRA.IBMSubmission.02282011.doc, V1.0, Draft,(Février 2011).

[6] Nicolas Degroodt, L'élasticité des bases de données sur le Cloud computing. Mémoire de Master en Sciences Informatiques. Bruxelles : Académie Universitaire Wallonie Bruxelles, 2011.

[7] http//www.zdnet.fr/actualites/sante-quel-cloud-computing-pour-ce-secteur-sensible-39822904.htm. (Consulté le 29/12/2016)

[8] http ://www.orange-business.com/fr/blogs/cloud computing/transformation/le-potentiel-du-cloud-dans-l-educationmstechdays. (Consulté le 29/12/2016)

[9] Ait Ali Sylia, Lahdir Meriem, Gestion de la cohérence des données répliquées dans le Cloud. Mémoire de Master en Informatique. Option : Réseaux et systèmes distribués : Université A/Mira de Bejaia, 2016.

[10] Alouache Lilia, Ait Mesbah Sofia, Solution de disponibilité dans les environnements Cloud Computing. Mémoire de Master en Informatique. Option : Réseaux et systèmes distribués : Université A/Mira de Bejaia, 2015.

[11] http ://www.usine-digitale.fr/article/les-5-grandes-tendances-qui-poussent-l-industrie-manufacturiere-a-adopter-le-cloud.N278476 (Consulté le 29/12/2016).

[12] Georges Gardarin, Bases de données.Paris : Eyrolles,2003, 826p. ISBN : 2-212-11281-5.

[13] Ciobanu (Iacob) Nicoleta ? Magdalena, Ciobano (Defta) costenelaLuminita, Synchronous partial replication case study : implementing elearning platform in an academic environment, Procedia - Social and Behavioral Sciences 46 (2012) 1522-1526 Elsevier.

[14] Roselin BILEY, Mise en place d'un système de réplication de base de données entre sites distants. [en ligne] génie logiciel. Dschang : Université de Dschang, 2009. Disponible sur http ://www.memoireonline.com/11/09/2900/m_Mise-en-place-dun-systeme-de-replication-de-base-de-donnees-entre-sites-distants5.html (Consulté le 02/01/2017).

[15] Nagamani H Shahapure, P Jayarekha, Replication, A Technique for Scalability in Cloud Computing, International Journal of Computer Applications Volume 122 ? No.5, (0975 ? 8887),(Juillet 2015).

[16] Sacha Krakowiak, Gestion répartie de données ? 1 Duplication et cohérence. Grenoble : INRIA et IMAG-LSR. École Doctorale de Grenoble Master 2 Recherche ?Systèmes et Logiciel ?, 2004.

[17] https ://fr.wikipedia.org/wiki/Coh%C3%A9rence(donn%C3%A9es)#Coh.C3.A9rence_stricte (consulté le 26 janvier 2017).

[18] C. Pierkot. Gestion de la mise à jour de données Géographiques Répliquées. Thèse de Doctorat en Informatique. Toulouse :Université de Toulouse III - Paul Sabatier, 2008.

[19] http ://www.lebigdata.fr/soins-de-sante-cloud-1412 (consulté le 27 janvier 2017).

[20] Arindam Das and Ajanta De Sarkar, ?On Fault Tolerance of Resources in Computational Grids ?, International Journal of Grid Computing and Applications, Vol.3, No.3, DOI : 10.5121/ijgca.2012.3301. Sept. 2012,

[21] Arcangeli, J., Hameurlain, A., Migeon, F. et Morvan,F. Mobile agent based self adaptative join for wide area distributedquery processing. Journal of Database Management, 15(4) :25-44, (2004).

[22] Ozsu, T. et Valduriez, P. (1999). Principles of Distributed Database Systems. Prentice Hall.

[23] Gray, J. (1980). A transaction model. Rapport technique, IBM Research Laboratory.

[24] www.renaudvenet.com/cloud-computing-avantages-et-inconvenients-2011-01-26.html (consulté le 29/12/2016).

[25] http ://www.salesforce.com/fr/crm/what-is-salesforce/ (consulté le 27 janvier 2017).

[26] http ://business.panasonic.fr/solutions/application/cameramanager-vidoprotection-dans-le-cloud-pour-le-secteur-militaire (consulté le 30 Janvier 2017

[27] I. Al Ridhawi, N. Mostafa, W. Masri, Location-Aware Data Replication in Cloud Computing SystemsWireless and Mobile Computing, Networking and Communication, Abu Dhabi, UAE, pp20-27, 19-21 Octobre 2015.

[28] G. Shafer, A mathematical theory of evidence. Princeton Univ. Press, 1975.

[29] D. Boru, D. Kliazovich, F. Granelli, P. Bouvry, A.Y. Zomaya, Energy-efficient data replication in Cloud Computing datacenters, Springer Cluster Computing, vol 18, n° 1, pp 385-402, 2015.

[30] L. M. Vaquero, L. Rodero-Merino, Finding your Way in the Fog : Towards a Comprehensive Definition of Fog Computing, 2014.

[31] S. Verma, A. K. Yadav, D.Motwani, R.S. Raw, H. K. Singh, An Efficient Data Replication and Load Balancing Technique for Fog Computing Environment. 3rd International Conference on ?Computing for Sustainable Global Development ?, 16-18 mars, New Delhi, India, pp : 5092-5099, 2016.

[32] G. da Silva, M. Holanda, A. Araujo, Data Replication Policy in a Cloud Computing Environment, 2016, URL : http ://ieeexplore.ieee.org/document/7521383/.

[33] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, ?The Hadoop distributed file system, ? in Proceedings of the IEEE 26th Symposium on Mass Storage Systems and Technologies, pp. 1 ?10, USA, 2010.

[34] S. Karuppusamy, M. Muthaiyan, An Efficient Placement Algorithm for Data Replication and To Improve System Availability in Cloud Environment. International Journal of Intelligent Engineering and Systems, Vol.9, No.4, pp : 88-97, 2016.

[35] Mohamed-K HUSSEIN, Mohamed-H MOUSA, A Light-weight Data Replication for Cloud Data centers environment, International Journal of Innovative Research in Computer, and Communication Engineering, Vol. 2, Issue 1, pp. 2392-2400, 2014.

[36] N. Santhi, R. Selva Kumar, Consistency and Privacy Based Replication System in Cloud Sharing Framework.International Journal of Current Research and Modern Education (IJCRME) 2455 ? 5428 Volume I, Issue I, 2016, (www.rdmodernresearch.com).

[37] G. J. Chang, D. D-Liu, X. Zhu, Distange graphs and T-Coloring, International journal of Combinatorial Theory, Series B 75, 259-269, 1999.

[38] Holt.C.C. Forecasting Seasonal and trends by exponentially weighted moving averages, ONR Memorandom (vol 52), Pitts burgh, PA : Carnegie Institute of Technology. Available from the Engineering Library, University of Texas at Astin, 1957.

[39] Makridakis, S.G., S.C. Wheelwright, and R.J. Hyndman, eds. Forecasting : Methods and Applications, 3rd Edition, 1998.

[40] http ://www.lebigdata.fr/soins-de-sante-cloud-1412 (consulté le 27 janvier 2017).

[41] M. Melo, P. Maciel, J. Araujo, R. Matos, C. Araujo : Availability Study on Cloud computing Environments : Live Migration as a Rejuvenationn Mechanism, Dependable Systems and Networks (DSN), 43 e Annual IEEE, 2013.

[42] W. En Dong, W. Nan, L. Xu, QoS-oriented Monitoring Model of Cloud computing Ressources Availability, Computational and Information Sciences (ICCIS) Fifth International Conference, IEEE 2013, 1537-1540, 21-23 juin 2013.

RÉSUMÉ

Le Cloud computing, possède une grand capacité en puissance de calcul et en stockage de données. Ce type d'infrastructure constitue un environnement privilégié pour le partage de ressources et de services.
Les techniques de partage, utilisées par le Cloud computing, sont le plus souvent basées sur le principe de réplication pour assurer une disponibilité très élevée des données et accélérer le temps d'accès aux données. La gestion de la réplication est une question critique pour les environnements du Cloud computing.
Dans ce mémoire nous avons parlé du Cloud computing, vu la gestion des données dans le Cloud, introduit les méthodes de réplication existantes dans la littérature, puis nous avons étudié différentes approches pour remédier au problème de la réplication. Nous avons ensuite présenté notre proposition qui permet de garantir la cohérence , la disponibilité et améliorer le temps d'accès aux données.
Mots clés : Cloud computing, réplication, cohérence, disponibilité.

ABSTRACT

Cloud computing, has a large capacity in computing power and data storage. This type of infrastructure provides a privileged environment for sharing resources and services. The sharing techniques used by Cloud computing are most often based on the principle of replication to ensure a very high availability of data and to speed up the access time to the data. Managing replication is a critical issue for cloud computing environments.
In this brief, we talked about cloud computing, considering the management of data in the Cloud, introduced the existing replication methods in the literature, and then studied different approaches to remedy the problem of replication. We then presented our proposal to ensure consistency, availability and improved data access time.
Key words : Cloud computing, replication, consistency, availability.