République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mouloud Mammeri de Tizi-Ouzou

En vue de l'obtention du diplôme de Master Academique
Spécialité : Informatique
Option : Conduite de projets informatiques

Thème

Cohérence et disponibilité dans les Bases de données NoSQL

Étude comparative

Présenté par :
Celia IHDENE

Devant le jury composé de :
Président(e) : M^{elle} YESLI Yasmine
Examinateur(trice) :M^{r} SAIDAINI Fayçal
Promoteur(trice) : M^{me} Taouri Dalila

Année universitaire : 2019/2020

Remerciements

D'abord, je remercie mon encadreur Madame TAOURI Dalila, pour m'avoir encadré et guidé et ainsi que pour ses judicieux conseils qui ont contribué à alimenter ma réflexion.
Je remercie chaleureusement les membres du jury pour l'honneur qu'ils m'ont fait en acceptant de juger mon travail.
Mes sincères sentiments vont à mes parents qui ont tant sacrifié jusqu'aujourd'hui et leurs encouragements tout le long de mon parcours.

Celia.

Dédicaces

Je dédie ce travail aux personnes les plus importantes pour moi ,mes parents , merci pour votre aide , votre amour et votre soutien tout au long de mes études, a mes sœurs Lisa et Lydia ,A toute ma famille, à mes chers ami(e)s :Katia, Tanina, Lynda, Dania et Koceila ,et enfin à tous ceux qui ont contribué de près ou de loin pour la réalisation de ce travail.

Celia.

Table des matières

Table des figures

Liste des tableaux

Résumé

Depuis une vingtaine d'années, avec l'explosion des applications web et une croissance de leurs usages, les données générées n'ont fait que s'accroître. Des trillions d'octets de données sont générés tous les jours à travers les réseaux sociaux ou encore le cloud.

Face à cette croissance, les solutions de base de données relationnelles ont montré rapidement leurs limites, en particulier dans le domaine du web et de nouveaux besoins se sont fait sentir et les grands acteurs du Web ont dû faire face à de nouveaux enjeux concernant leurs infrastructures informatiques.

Le cloud computing est un terme général qui implique la fourniture de services hébergés sur Internet. Avec la croissance accélérée du volume de données utilisées par les applications, de nombreuses entreprises ont transféré leurs données vers des serveurs cloud pour fournir des services évolutifs, fiables et hautement disponibles. Un problème particulièrement difficile qui se pose dans le contexte des systèmes de stockage cloud avec des données répartie géographiquement est de savoir comment atteindre un état cohérent pour toutes les répliques. Plusieurs méthodes ont été proposées pour régler le compromis entre cohérence et disponibilité des données au niveau des systèmes distribués.

Mots clés

Big Data, Internet des Objets (IoT), Bases de données NoSQL, Cohérence des données , Disponibilité des données , modèles centré sur les donnés , modèles centré sur le clients.

Abstract

For the past twenty years, with the explosion of web applications and the growth of their uses, the data generated has only increased. Trillions of bytes of data are generated every day through social networks and the cloud.

Faced with this growth, relational database solutions quickly showed their limits, particularly in the web domain and new needs arose and the major web players had to face new challenges concerning their infrastructures. computer science.

Cloud computing is a general term that involves the provision of hosted services over the Internet. With the accelerating growth in the volume of data used by applications, many companies have moved their data to cloud servers to provide scalable, reliable and highly available services. A particularly difficult problem that arises in the context of cloud storage systems with geographically distributed data is how to achieve a consistent state for all replicas. Several methods have been proposed to resolve the trade-off between consistency and availability of data at the level of distributed systems.

Keywords

Big Data, Internet of Things (IoT), NoSQL DataBases, Data consistency, Data availability, data-centric models, client-centric models

Présentation du travail effectué

Dans le cadre de notre projet de fin d'études, nous avons réalisé le présent mémoire qui se divise en deux (2) grandes parties majeures.

1. **Première partie État de l'art**

   Dans cette partie qui est un état de l'art , nous allons parler des trois premiers chapitres qui nous permettront de mettre en évidence le contexte de ce mémoire.

   — **Chapitre 1** : Big Data.

   — **Chapitre 2** : Internet des Objets (IoT).

   — **Chapitre 3** : Les bases de données NoSQL.

2. **Deuxième partie Synthèse**

   Après l'étude de l'état de l'art qui est une partie indispensable afin de bien comprendre la deuxième partie qui est une synthèse des des derniers travaux de recherche et qui se divise elle même en deux parties.

   — **Chapitre 4** : Disponibilité et cohérence des données dans les BD NoSQL.

   — **Chapitre 5** : Étude comparative entre les modèles data-centric et client-centric.

Première partie

Etat de l'art

Chapitre 1

BIG DATA

Introduction

Un flux massif de données dans un format structuré, non structuré ou hétérogène a été accumulé en raison de l'augmentation continue du volume et des détails des données saisies par les organisations, telles que les médias sociaux, le gouvernement, l'industrie et la science. Ces quantités massives de données sont produites en raison de la croissance du Web, de l'essor des médias sociaux, de l'utilisation du mobile et de l'Internet des Objets (IoT) par et au sujet des personnes, des choses et de leurs interactions. L'ère du Big Data est arrivée.

Le Big Data devient la force la plus influente de la vie quotidienne. Selon les rapports d'IDC, l'univers numérique double de taille tous les 2 ans. Comment stocker d'énormes quantités de données n'est plus le plus gros problème. Mais comment concevoir des solutions pour comprendre cette grande quantité de données est un défi majeur. Les opérations telles que les opérations analytiques, les opérations de traitement et les opérations de récupération sont très difficiles et prennent énormément de temps à cause de cet énorme volume de données. Une solution pour surmonter ces problèmes est l'utilisation de techniques d'exploration de données dans la découverte de connaissances à partir de Big Data.

Dans cette section, nous allons énumérer quelques statistiques ainsi que ces concepts et définitions se rapportant au domaine du "Big Data", avantages, inconvénients et caractéristiques, ensuite nous parlerons de l'internet des objets et son lien avec les mégadonnées.

1.1 Concepts généraux

1.1.1 Historique et quelques statistiques sur le Big Data

L'expression " Big Data " apparaît fréquemment dans la presse et dans les revues universitaires, et des programmes de "Data Science» ont vu le jour dans le monde universitaire au cours des six dernières années. Le 29 mars 2012, WHOSTP [1] a annoncé la "Big Data Research and Development Initiative" (Kalil 2012) qui s'appuie sur des initiatives fédérales "allant de l'architecture informatique et des technologies de mise en réseau aux algorithmes, à la gestion des données, à l'intelligence artificielle , apprentissage automatique, développement et déploiement de cyber infrastructures avancées"[1].

1. WHOSTP : fait référence au bureau de la politique scientifique et technologique de la Maison Blanche

Les "Big data" sont apparues environ 560 fois par an dans JSTOR [2] de 2014 à 2017, même si elles ont été mentionnées moins d'une fois par an au siècle avant 2000 et seulement en moyenne environ huit fois par an entre 2001 et 2010.

Au cours des six dernières années, au moins 17 programmes de science des données ont commencé dans les principales universités de recherche américaines et internet regorge de publicités pour des livres et des cours de science des données, souvent avec le lancement de " Devenez Data Scientist ".

Selon l'étude Data Age 2025 4 La sphère de données mondiale passera de 33 zettaoctets en 2018 à 175 Zo d'ici 2025. Près de 30% des données mondiales devront être traitées en temps réel et le stockage réalisé sur le Cloud public représentera 49% du volume total de données.[2]

Près de 75% des entreprises pourraient implémenter le Big Data et l'intelligence artificielle à leurs activités en 2020. C'est ce que révèle un sondage mené par *The Economist* [3] auprès de 203 responsables d'entreprises.
En 2020, les entreprises se servant du Big Data seraient largement avantagées et gagneraient au total 1,2 billion de dollars par an de plus que les entreprises qui ne s'en servent pas. Telle est la prédiction des analystes de Forrester [4].

Puisque l'on parle d'objets connectés, il faut savoir que leurs nombres ne cessera d'augmenter de façon impressionnante. En 2018 intel [5] avait prédit qu'en 2020 un total de 200 milliards d'appareils connectés en circulation aussi bien pour le grand public que pour les entreprises.

La figure ci-dessous permet de présenter la quantité de données générées en 60 secondes d'Internet en 2019 et 2020.

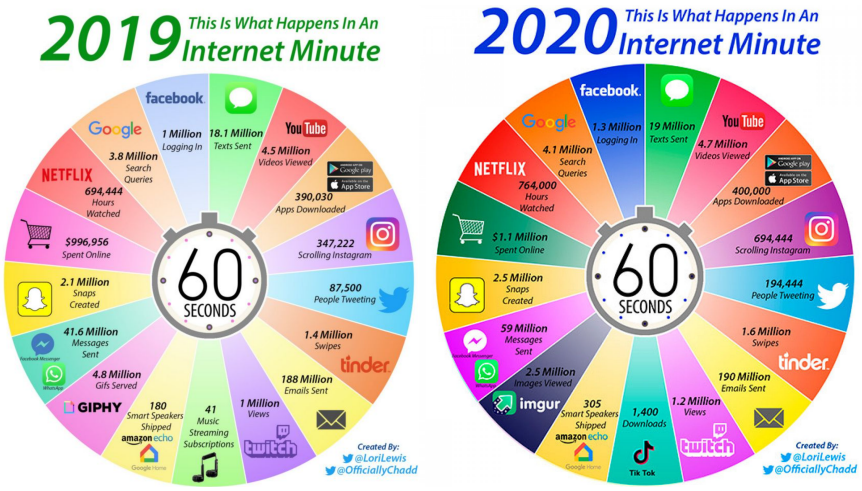

FIGURE 1.1 – 1 minute d'internet.

2. JSTOR : désigne à la fois un système d'archivage en ligne de publications universitaires et scientifiques et une bibliothèque numérique payante. Fondé en 1995, JSTOR est aujourd'hui une société américaine à but non lucratif basée à New York

3. The Economist est un magazine d'actualité hebdomadaire britannique. Imprimé simultanément dans six pays, il est l'un des hebdomadaires de référence à l'échelle mondiale, ciblant une population hautement éduquée. Il est publié à la fois sur papier et le Web, sa diffusion est supérieure à 1 500 000 exemplaires

4. Forrester Research est une entreprise américaine indépendante qui fournit à ses clients des études de marché sur l'impact des technologies dans le monde des affaires. Forrester Research posède dix-neuf implantations géographiques à travers le monde, dont huit centres de recherche

5. Intel Corporation est une entreprise américaine fondée le 18 juillet 1968 par Gordon Moore, Robert Noyce et Andrew Grove. Elle est le premier fabricant mondial de semi-conducteurs

Si nous nous intéressons de plus près à la **Figure 1.1**, nous pouvons bien constater que nous nageons dans un océan de données où le niveau de l'eau augmente rapidement. En effet, à chaque minute d'Internet, sont envoyés plus de 47 millions de messages en 2019 et 59 millions en 2020. Chaque minute, Google [6] enregistre près de 3.8 millions de requêtes différentes sur son moteur de recherche en 2019 et 4.1 millions en 2020. Facebook [7] enregistre 1.3 millions de logins.

Aujourd'hui les entreprises sont confrontées à certaines problématiques qui sont celles de savoir comment collecter, stocker, analyser et exploiter ces grands volumes de données pour pouvoir créer de la valeur ajoutée. Tout l'enjeu, pour ces dernières consiste à ne pas passer à côté d'informations précieuses noyées dans la masse. C'est là qu'intervient la technologie du "Big Data", qui repose sur une analyse très fine de masse de données.

1.1.2 Quelques définitions liées au Big Data

Définition 1 : Les big data ou mégadonnées désignent l'ensemble des données numériques produites par l'utilisation des nouvelles technologies à des fins personnelles ou professionnelles. Cela recoupe les données d'entreprise (courriels, documents, bases de données, historiques de processeurs métiers ...) aussi bien que des données issues de capteurs, des contenus publiés sur le web (images, vidéos, sons, textes), des transactions de commerce électronique, des échanges sur les réseaux sociaux, des données transmises par les objets connectés (étiquettes électroniques, compteurs intelligents, smartphones ...), des données géolocalisées ...[3]

Définition 2 : Le "Big data" désignent les technologies et les initiatives qui impliquent des données trop diverses, en évolution rapide ou massives pour que les technologies, les compétences et les infrastructures conventionnelles puissent être traitées efficacement. Autrement dit, le volume, la vitesse ou la variété des données est trop important.[4]

Définition 3 : Le terme big data fait référence aux données dont le coût de stockage, de gestion et d'analyse dans des systèmes de base de données traditionnels (relationnels et/ou monolithiques) serait généralement trop élevé. Habituellement, ces systèmes ne sont pas rentables, car ils ne disposent pas de la flexibilité nécessaire pour stocker des données non structurées (comme des images, du texte et des vidéos), pour accommoder des données "à haute vélocité" (en temps réel) ou pour s'adapter automatiquement à de très gros volumes de données (de l'ordre du pétaoctet).[5]

> **Remarque :** *Bien qu'il existe différentes définitions du big data aucune n'est précise et universelle car étant un objet complexe et polymorphe sa définition varie entre les communautés qui s'y intéressent en tant qu'usager ou fournisseur de service.*

6. Google : c'est une entreprise américaine de services technologiques fondée en 1998 dans la Silicon Valley, en Californie, par Larry Page et Sergey Brin, créateurs du moteur de recherche Google. C'est une filiale de la société Alphabet depuis août 2015.

7. Facebook : C'est une société américaine créée en 2004 par Mark Zuckerberg. Elle est un des géants du Web, regroupés sous l'acronyme GAFAM, aux côtés de Google, Apple, Amazon et Microsoft.

1.1.3 Avantages du Big Data

Plusieurs avantages peuvent être associés au Big Data, il permet par exemple :[6]

— **Prendre les bonnes décisions :** Toute partie, qu'elle soit gouvernementale ou privée, rentable ou volontaire, peut économiser beaucoup d'argent en exploitant les données. En planifiant des décisions saines et correctes et des plans pour l'avenir.
— **Augmentation des ventes :** Les producteurs et les commerçants peuvent bénéficier d'informations stockées sur des réseaux sociaux tels que Facebook et Twitter pour voir la réactivité de leurs offres, campagnes publicitaires et autres éléments les aidant à planifier leurs produits et à augmenter leurs ventes.
— **Meilleurs services de santé :** Les données des patients enregistrés à l'hôpital, telles que les dossiers pré-médicaux, peuvent être utilisées pour fournir des services plus rapides. Et mieux détaillé pour les patients.
— **Prévention des maladies :** Les médecins peuvent éviter les données du génome humain (trois milliards de caractères contenant des informations humaines) Nombreuses maladies et traitement des maladies incurables.
— **Réduire le coût de la publicité :** Toute entreprise peut envoyer aux clients des publicités personnalisées en fonction de leurs intérêts, directement via le système appelé(Microtargeting [8]).

1.1.4 Les contraintes du Big Data

Cette technologie étant en plein essor, promis à un avenir radieux, offre un nombre considérable d'avantages et marque une avancée révolutionnaire, en plus d'être pleine de promesses pour les entreprises. Mais, comme toute avancée technologique, le Big Data a ses limites et cela nous amène à nous questionner sur les risques majeurs potentiels de vulnérabilité et de sécurité liée au Big Data. On en citera parmi ces risques :

— **L'automatisation de la discrimination :** Il y a trois ans, l'EPIC [9] estimait que l'utilisation des analyses prédictives dans le secteur public et le secteur privé peuvent permettre aux gouvernements et aux entreprises d'évaluer la capacité d'une personne à obtenir un emploi où un crédit. Or, cette utilisation nui directement à la liberté d'association.
— **Une augmentation des fuites de données :** Suite aux nombreuses fuites de données des géants du commerce comme Target [10], Home Depot [11], ou de sites comme eBay [12], ayant impacté des dizaines de millions d'individus, le public est très alerté quant aux fraudes de cartes de crédit et aux usurpations d'identité.
— **La fin de l'anonymat et de la confidentialité :** De nos jours, il est de plus en plus difficile de faire quoi que ce soit sans que notre identité soit associée à nos actions. Même les données « désidentifiées » représentent un risque pour la confidentialité. Les standards de sécurité utilisés il y a encore un an ou deux ne

8. Le Microtargeting : est une stratégie marketing qui utilise les données des gens - sur ce qu'ils aiment, à qui ils sont connectés, quelles sont leurs données démographiques, ce qu'ils ont acheté, etc. - pour les segmenter en petits groupes pour le ciblage de contenu

9. EPIC : c'est le fournisseur de services de sécurité le plus grand et le mieux équipé de la région métropolitaine de NY / NJ.

10. Target Corporation, située à Minneapolis dans le Minnesota est une entreprise de grande distribution. Elle est le deuxième plus gros distributeur discount et le cinquième distributeur général aux États-Unis, derrière Wal-Mart, Home Depot, Kroger et Costco

11. Home Depot, Inc. est une entreprise de distribution américaine pour l'équipement de la maison dont le siège social est situé dans la ville d'Atlanta en Géorgie. The Home Depot est le n° 1 mondial de la distribution de détail de produits de rénovation résidentielle

12. eBay est une entreprise américaine de courtage en ligne, connue par son site web de ventes aux enchères du même nom. Elle a été créée en 1995 par le Français Pierre Omidyar

sont plus suffisants. Les entreprises qui souhaitent rendre les données anonymes sont confrontées à une difficulté croissante. Il sera bientôt impossible d'empêcher les données de pouvoir être à nouveau associées aux individus.

— **La vente aux enchères des données :** De nombreuses entreprises collectent et vendent les données des utilisateurs, permettant d'établir des profils d'individus. Les entreprises peuvent désormais savoir si une femme est enceinte, si une personne est homosexuelle ou si elle est atteinte d'un cancer avant même qu'elle ne le révèle à ses proches. Aucune loi ne protège réellement les consommateurs contre de tels agissements.

1.1.5 Les caractéristiques du Big Data

Ce volume vertigineux de données ne peut plus être collecté, stocké, géré et exploité par les solutions informatiques traditionnelles combinant infrastructures matérielles et bases de données relationnelles. En vue de trouver les solutions technologiques adéquates, une première phase de clarification conceptuelle du Big Data s'est imposée. Ainsi, des cabinets d'étude et d'analyse ont proposé la règle des 3V : Volume, Vélocité, Variété.

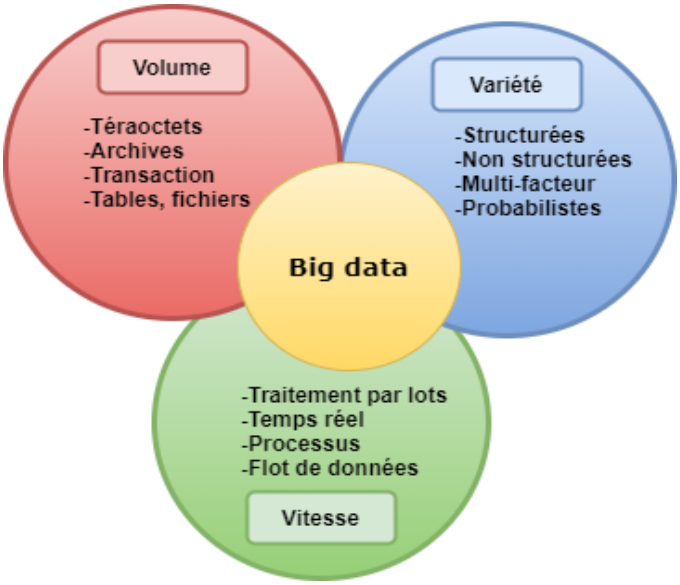

FIGURE 1.2 – Les 3V du big data.

Certaines personnes attribuent encore plus de V aux Big Data, les scientifiques des données et les consultants ont créé diverses listes avec entre sept et 10 V. Les caractéristiques des mégadonnées basées sur la définition « 10V » peuvent être détaillées comme suit :

1.1.5.1 Le Volume

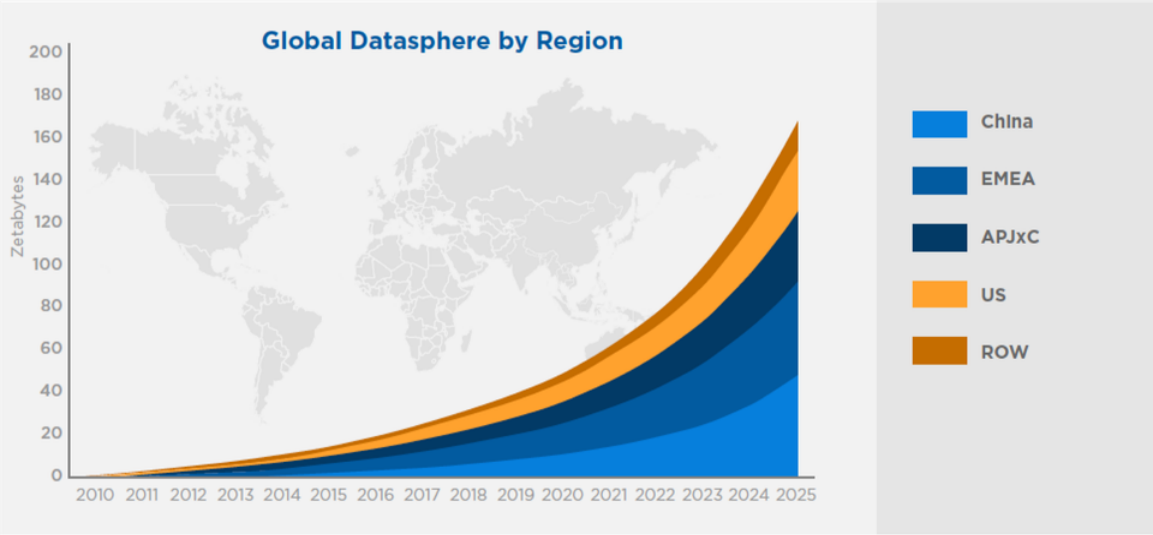

FIGURE 1.3 – Taille des données mondiales.

Le volume fait bien évidemment référence à la masse des données d'une part auxquelles nous avons accès, mais également que nous générons. Ce volume augmente à un rythme exponentiel, c'est à dire que la création de données ne cesse de s'accélérer. Ce n'est pas surprenant, si l'on considère que plus de 95% de toutes les données actuelles ont été créées ces cinq dernières années. La masse actuelle de données peut être réellement stupéfiante. En résumé, le volume est la dimension du big data liée à sa taille et à sa croissance exponentielle.

Exemple : *300 heures de vidéo sont mises en ligne sur YouTube chaque minute.*

1.1.5.2 La Variété

Elle fait référence aux différentes formes toujours croissantes que les données peuvent prendre, la variété des données est une mesure de la richesse de la représentation des données - texte, images vidéo, audio, etc. Les données produites ne sont pas de catégorie unique car elles comprennent non seulement les données traditionnelles mais aussi les données semi-structurées provenant de diverses ressources comme les pages web, les sites de médias sociaux, e-mail, les documents.

Exemple : *Un projet d'analyse des mégadonnées peut tenter d'évaluer le succès d'un produit et les ventes futures en corrélant les données de ventes passées, les données de retour et les données de révision des acheteurs en ligne pour ce produit.*

1.1.5.3 La Vitesse

La vitesse décrit la fréquence à laquelle les données sont générées, capturées et partagées. Du fait des évolutions technologiques récentes, les consommateurs mais aussi les entreprises génèrent plus de données dans des temps beaucoup plus courts.

Exemple : *À lui seul, Google traite en moyenne plus de "80 000 requêtes de recherche par seconde", ce qui correspond à peu près à plus de 6,9 milliards de recherches par jour.*

Comme on la mentionner plus haut de plus en plus de Vs ont été introduits dans la communauté du Big Data alors que nous découvrons de nouveaux défis et de nouvelles façons de définir le Big Data. La véracité et la valence sont deux de ces V supplémentaires auxquels nous accorderons une attention particulière :

1.1.5.4 La Véracité

La véracité des données se concentre sur la qualité et l'exactitude des données et définit la manière dont on peut se fier aux données lorsqu'une décision importante doit être prise concernant les données collectées. Les données sont classées en bonnes, mauvaises ou indéfinies, ce qui peut être dû à l'incohérence, l'incomplétude, l'ambiguïté, la latence, la tromperie et les approximations des données. En ce qui concerne la sensibilité des données, l'organisation doit appliquer des stratégies efficaces afin de protéger les données et de se conformer aux exigences réglementaires.

1.1.5.5 La Valeur

La valeur des données fait référence à l'utilité des données dans la prise de décision et est l'un des facteurs les plus importants des Big Data, car elle a un impact direct sur les bénéfices des entreprises.

Exemple : *McKinsey a estimé une série d'initiatives en matière de soins de santé et a affirmé que l'impact potentiel pourrait représenter 300 à 450 milliards de dollars de réduction des dépenses de santé, soit 12% à 17% des 2,6 billions de dollars de base des coûts de santé aux États-Unis si les premiers succès étaient étendus pour créer un impact à l'échelle du système.*

Il a également souligné que les organisations s'accordent à dire que l'utilisation d'informations et d'analyses commerciales les différencie dans l'industrie en tant qu'entreprises les plus performantes et les moins performantes. Ainsi, la valeur réside dans l'analyse méticuleuse de données précises.

Certaines personnes attribuent encore plus de V aux Big Data, les scientifiques des données et les consultants ont créé diverses listes avec entre sept et 10 Vs. En ajoutant la variabilité, la validité, la virtualisation, la volatilité ainsi que la vulnérabilité.

1.1.5.6 La Variabilité

Elle fait référence à un flux de données incohérent, et peut être indiquée à certains moments. Cette propriété est devenue problématique en raison de l'utilisation croissante des médias numériques, qui est la principale cause des pics de charge de données.

Exemple du supercalculateur : *Brian Hopkins, analyste principal de Forrester, a cité le supercalculateur Watson comme un excellent exemple de cela. Pour participer au jeu télévise Jeopardy, Watson devait être capable de comprendre l'énoncé des questions, buzzer pour prendre la main, disséquer une réponse dans son sens pour déterminer quelle était la bonne question. Les mots n'ont pas de définitions statiques et leurs significations peuvent varier énormément dans le contexte.*

1.1.5.7 La Validité

La validité des données peut être le fait d'avoir les mêmes idées que la véracité des données, mais, en fait, ils n'ont pas les mêmes concepts et théories. Lorsque le statut des données passe d'exploratoire à actionnable, les données doivent être valides. Un ensemble de données peut ne pas avoir de problèmes de véracité mais peut aussi ne pas être valide si elle n'est pas correctement compréhensible. Cette propriété des Big data est essentielle pour trouver la présence de relations cachées entre les éléments au sein des énormes sources de génération de Big Data. De même, un ensemble de données peut être validé pour une application ou un usage spécifique et ne sera pas validé pour une autre application ou un autre usage. La validité des sources de production de Big Data et l'analyse qui en découle doivent être exactement correctes si les résultats de l'analyse doivent être utilisés pour le processus de prise de décision.

Exemple : *Prenons exemple sur un ensemble de données statistiques sur les achats effectués dans les restaurants et les prix de ces articles au cours des cinq dernières années. Vous pourriez vous demander : "Qui a créé cette source ? Quelle méthodologie ont-ils suivie pour collecter les données ? Seules certaines cuisines ou certains types de restaurants ont-ils été inclus ? Les créateurs des données ont-ils résumé les informations ? Les informations sont-elles été éditées ou modifiées par quelqu'un d'autre ?*

1.1.5.8 La Volatilité

Quel âge doivent avoir vos données pour qu'elles soient considérées comme non pertinentes, historiques ou obsolète ? Combien de temps faut-il conserver les données ? Avant l'ère big data, en général on stockait les données indéfiniment. Quelques téraoctets de données ne pouvaient pas engendrer de dépenses de stockage élevées.

En raison de la vitesse et du volume de ces données massives, leur volatilité doit être soigneusement prise en compte. Il est maintenant fondamental d'établir des règles pour la disponibilité et à la mise à jour des données afin de garantir une récupération rapide des informations en cas de besoin.[7]

Exemple : *Certaines entreprises peuvent ne conserver que l'année la plus récente de leurs données et transactions client dans leurs systèmes. Cela garantira une récupération rapide de ces informations si nécessaires. S'ils doivent consulter une année antérieure, l'équipe informatique peut être amenée à restaurer les données d'un stockage hors connexion pour répondre à la demande. Avec le Big Data, ce problème est amplifié. Si le stockage est limité, il faut examiner les sources de données volumineuses pour déterminer ce qu'on doit collecter et combien de temps pour le conserver.*

1.1.5.9 La Virtualisation

Une autre caractéristique du Big Data est la difficulté à les visualiser. Les logiciels de visualisation de données volumineuses actuels sont confrontés à des problèmes techniques en raison des limitations de la technologie en mémoire, de leur faible évolutivité, de leur fonctionnalité et de leur temps de réponse. Il est impossible de vous fier aux graphiques traditionnels lorsque vous essayez de tracer un milliard de points de données. Il est donc nécessaire d'avoir différentes manières de représenter des données. Telles que la mise en cluster de données ou l'utilisation de cartes d'arbres, de sunbursts, de coordonnées parallèles, de diagrammes de réseau circulaires ou de cônes. Si on associe cela avec la multitude de composante résultant de la variété et de la vélocité des données massives et des relations complexes qui les lient, il est possible de voir qu'il n'est pas si simple de créer une visualisation significative.[7]

1.1.5.10 La Vulnérabilité

Le Big Data apporte de nouveaux problèmes de sécurité. Après tout, une violation de données avec Big Data est une grande violation. Malheureusement, il y a quotidiennement des violations de données massives.

Exemple rapporté par CRN [13] **:** En mai 2016, un pirate informatique appelé Peace a posté des données sur le Dark web pour les vendre, qui auraient inclus des informations sur 167 millions de comptes LinkedIn et 60 millions d'emails et de mots de passe pour les utilisateurs de MySpace [14].[7]

1.2 Les technologies du Big Data

Avec la venue et la croissance du Big Data, le volume de données à gérer par les applications a explosé, et les systèmes de gestions de données traditionnels, relationnels et transactionnels, basés sur le langage SQL [15], ont montré leurs limites. Depuis quelques années, de nouvelles approches de stockage et de la gestion des données Sont apparues permettant de s'astreindre de certaines contraintes, en particulier de scalabilité, inhérentes au paradigme relationnel. Ces nouvelles créations technologiques peuvent globalement être catégorisées en deux Familles que nous présenterons comme suit :

— Les technologies de stockage.
— Les technologies de traitement.

1.2.1 Les technologies de stockage

Dans cette première partie nous parlerons des technologies de stockage, portées plus particulièrement sur le déploiement du Cloud Computing.

1.2.1.1 Le Cloud Computing

Nous avons vu que le phénomène Big Data, par son intitulé simple et porteur, facilement englobant, génère un intérêt manifeste et les sources de données se sont largement diversifiées et ont été surtout localisées sur Internet, ces informations sont produites de façon continue et à un rythme soutenu. De nouvelles solutions ont alors vu le jour pour pouvoir traiter ces volumes d'informations et ces flux de données, toutes ces solutions reposent sur un stockage distribue (partitionne) des données sur les clusters.

1.2.1.1.1 Définitions

Le terme « Cloud » vient du nuage représentant Internet ; « Computing » renvoie la puissance de calcul de ce nuage. Le Cloud Computing ou l'informatique en nuage est l'exploitation de la puissance de calcul où de stockage des serveurs informatiques distants par l'intermédiaire d'un réseau, généralement internet. Ces serveurs sont loués à la demande, le plus souvent par tranche d'utilisation selon des critères techniques (puissance, bande passante, etc.) mais également au forfait.

13. CRN, anciennement Computer Reseller News, est un magazine commercial américain traitant d'informatique. Il cible principalement le milieu de la revente informatique

14. Myspace est un site web de réseautage social fondé aux États-Unis en août 2003, qui met gratuitement à disposition de ses membres enregistrés un espace web personnalisé, permettant de présenter diverses informations personnelles et d'y faire un blog

15. SQL : acronyme de Structured Query Language est un langage informatique normalisé servant à exploiter des bases de données relationnelles

Il se caractérise par sa grande souplesse : selon le niveau de compétence de l'utilisateur client, il est possible de gérer soi-même son serveur ou de se contenter d'utiliser des applicatifs distants en mode SaaS [16]. Selon la définition du (NIST [17]) le cloud computing est l'accès via un réseau de télécommunications, à la demande et en libre-service, à des ressources informatiques partagées configurables. Il s'agit donc d'une délocalisation de l'infrastructure informatique. On peut y avoir accès gratuitement, comme c'est le cas avec le « Webmail ». Ou sur abonnement, avec un niveau de service garanti.

1.2.1.1.2 Les caractéristiques du Cloud

Accès réseau universel : L'ensemble des ressources doit être accessible et à disposition de l'utilisateur universellement et simplement à travers le réseau.

Libre-service à la demande : Permet à l'utilisateur d'utiliser et de libérer des ressources distantes en temps réel en fonction de ses besoins, sans nécessiter d'intervention humaine du côté fournisseur.

Ressources partagées : Les ressources matérielles du fournisseur sont partagées entre les différents utilisateurs.

Élasticité : Les ressources allouées aux utilisateurs peuvent être augmentées ou diminuées selon l'usage.

Service mesurable et facturable (pay-as-you-use) : Les utilisateurs paieront pour les ressources qu'ils ont utilisées et pour la durée de leur utilisation.

1.2.1.2 Principe de fonctionnement d'un Cloud

Le Cloud Computing est une combinaison de deux concepts qui sont la " Virtualisation " et les " Data center " :

1.2.1.2.1 Virtualisation

La virtualisation regroupe l'ensemble des techniques matérielles ou logicielles permettant de faire fonctionner, sur une seule machine physique, plusieurs configurations informatiques (systèmes d'exploitation, applications, mémoire vive...) de manière à former plusieurs machines virtuelles qui reproduisent le comportement des machines physiques.

Même s'il existe plusieurs types de virtualisation, telle que la virtualisation des postes clients ou la virtualisation des super calculateurs, la forme la plus populaire de virtualisation dans le cloud est la virtualisation des serveurs qui consiste à dématérialiser le comportement et les données d'un serveur ou d'une machine, de façon à faire tourner plusieurs de ces instances dématérialisées sur un même serveur physique. De cette façon, les différentes instances créées se partagent les ressources du serveur physique. Cela permet une plus grande modularité dans la répartition des charges, une facilité dans l'administration des serveurs et une réduction des coûts [8].

16. SaaS : acronyme de Software as a service est un modèle d'exploitation commerciale des logiciels dans lequel ceux-ci sont installés sur des serveurs distants plutôt que sur la machine de l'utilisateur.

17. NIST : acronyme de National Institute of Standards and Technology, en français " Institut national des normes et de la technologie", c'est une agence du département du Commerce des États-Unis. Son but est de promouvoir l'économie en développant des technologies, la métrologie et des standards de concert avec l'industrie

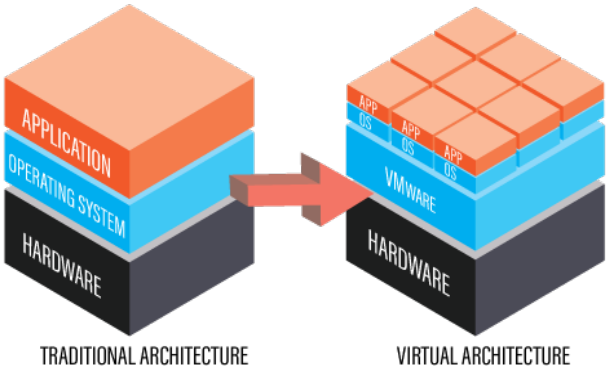

FIGURE 1.4 – La virtualisation.

1.2.1.2.2 Data center

Un data center ou centre de données, c'est un site physique qui a une infrastructure composée d'un réseau d'ordinateurs et d'espaces de stockage,Il peut être interne ou externe à l'entreprise. Ces sites sont des salle remplies de baies de stockage, utilisées par de nombreuses entreprises et autres organisations gouvernementales. [9]

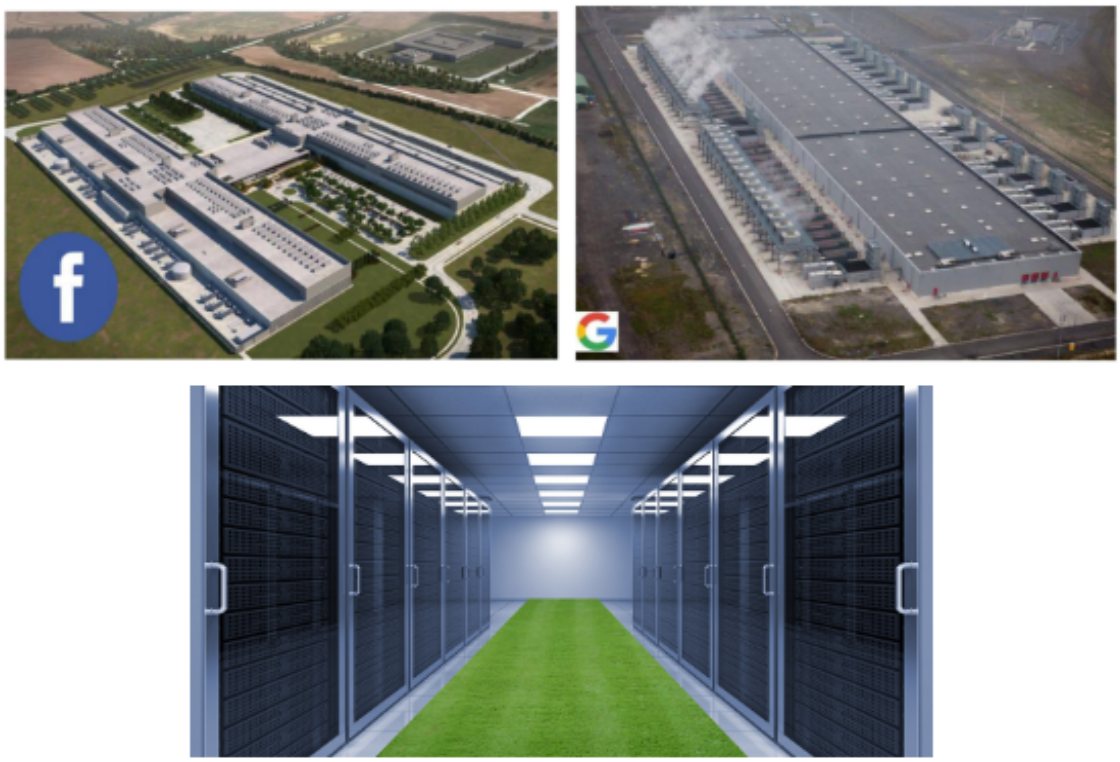

FIGURE 1.5 – Exemple de Data-center de Google et Facebook.

1.2.1.3 Modèles de service

Selon le type de ressource offerte aux utilisateurs, les services cloud ont tendance à être regroupés selon les trois modèles de base SaaS, PaaS et IaaS :

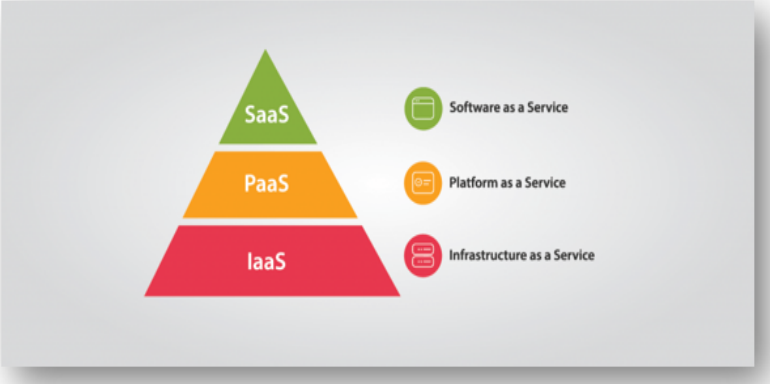

FIGURE 1.6 – Services cloud.

1.2.1.3.1 SaaS (Software as a Service)

Dans ce modèle, le logiciel est offert sous la forme d'un service. Le fournisseur de Cloud de type SaaS gère entièrement sa plateforme matérielle et logicielle, et les clients du Cloud utilisent ainsi le logiciel fourni sans s'occuper de la pile en dessous (plateforme applicative, matériel, ...) ni de l'installation du logiciel en question.

Exemples de SaaS : *La messagerie électronique, logiciels de type CRM ...*

1.2.1.3.2 PaaS (Plateforme as a Service)

Dans ce second modèle, le fournisseur offre une plateforme sous forme d'environnement complet sur laquelle des développeurs ou éditeurs de logiciels des clients peuvent déployer des applications. La pile en dessous de cette plateforme à savoir le socle applicatif, le système d'exploitation, le matériel et le réseau, sont gérés par le fournisseur de service. Notons que certaines offres PaaS exigent un langage de programmation spécifique.

Exemple de PaaS : *Google AppEngine (PaaS où les développeurs peuvent réaliser leurs programmes en Python ou Java), Engine Yard (avec Ruby on Rails), force.com de-Salesforce.com (langage propriétaire) et Coghead SAP (langage propriétaire).*

1.2.1.3.3 IaaS (Infrastructure as a Service)

Infrastructure as a Service (IaaS) est la proposition d'un ensemble de services informatiques comprenant le matériel, le réseautage et le stockage. Le fournisseur offre une plateforme sur laquelle les clients vont pouvoir déployer de ressources d'infrastructures dont la plus grande partie est localisée à distance dans des Data Centers. Le client acquiert une ressource et est facturé pour cette ressource en fonction de la quantité utilisée et de la durée d'utilisation. On trouve des versions publiques et privées de IaaS. Dans le modèle IaaS publique, l'utilisateur utilise une carte de crédit pour acquérir ces ressources, dès que l'utilisateur cesse de payer, la ressource disparaît. Dans un service IaaS privé, c'est généralement la structure informatique qui crée l'infrastructure conçue pour fournir des ressources à la demande pour les utilisateurs internes et parfois les partenaires commerciaux. Ce modèle considéré comme une évolution des Data Centers virtualisés,

permet au client de faire abstraction du modèle physique (gestion des serveurs physiques, l'électricité, la climatisation, la sécurité physique des centres de données). Dans ce modèle, le fournisseur contrôle le matériel et la couche de virtualisation. Sur l'aspect de données, le contrôle est effectué au niveau de la machine virtuelle qui est stockée et sauvegardée par le fournisseur de Cloud IaaS.

Exemples d'IaaS : Amazon EC2 (Web Services Elastic Compute Cloud) et Amazon S3(Secure Storage Service), CloudSigma, RackSpace (Cloud Server et CloudFiles).

Pour simplifier : : Avec le SaaS on utilise une application, avec le PaaS on construit ses applications et l'IaaS permet d'héberger le tout.

1.2.1.3.4 DBaaS (DataBase as a service)

Un tel modèle fournit des mécanismes transparents pour créer, stocker, accéder et mettre à jour des bases de données. De plus, le fournisseur de services de base de données assume l'entière responsabilité de l'administration de la base de données, garantissant ainsi la sauvegarde, la réorganisation et les mises à jour de version.

L'utilisation de ce service permet aux fournisseurs de répliquer et de personnaliser leurs données sur plusieurs serveurs, qui peuvent être physiquement séparé [10]. Ce faisant, ils peuvent répondre aux demandes croissantes en dirigeant les utilisateurs vers le serveur le plus proche ou le plus récemment consulté.

1.2.1.4 Stockage dans le cloud

Le stockage dans le cloud englobe l'archivage, l'organisation et la distribution de données à la demande entre plusieurs volumes de stockage virtualisés, regroupés à partir d'équipements matériels physiques distincts. Plus simplement, le stockage dans le cloud est l'organisation des données stockées dans un emplacement accessible depuis Internet par toute personne qui dispose d'une autorisation. Nous n'avons pas besoin de nous connecter à un réseau interne (on parlerait alors de serveur de stockage en réseau) et on accède pas aux données depuis notre propre disque dur ni à partir de matériel directement relié à notre ordinateur.

Nous distinguons deux formes principales de stockage du Cloud Computing suivant le réseau dans lequel les services sont disponibles : Cloud public et Cloud privé, ainsi deux autres formes dérivées, à savoir Cloud communautaire et hybride [11], comme sur la **Figure 1.7**

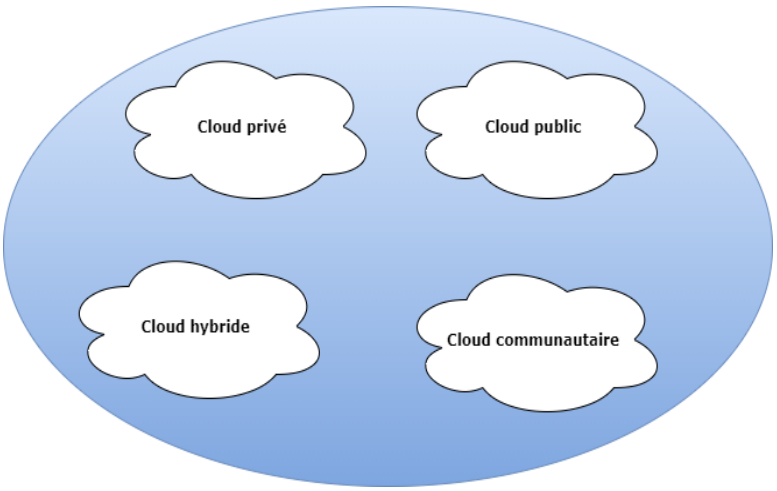

FIGURE 1.7 – Modèles de déploiement de cloud.

1.2.1.4.1 Cloud Public

Le Cloud publique est un ensemble de matériel, réseautage, stockage, services, applications et interfaces dont un tiers est propriétaire et exploité par d'autres sociétés et individus. Ces fournisseurs externes possèdent des Data Center hautement extensibles masquant les détails de l'infrastructure au client. Les services offerts sont accessibles via internet à tout le monde qui paye les services. Le principe est d'héberger des applications, en général des applications Web, sur un environnement partagé avec un nombre illimité d'utilisateurs. Les Clouds publiques sont viables parce qu'ils gèrent généralement des charges de travail relativement simples et répétitives comme par exemple, le courrier électronique. En outre, les entreprises peuvent choisir de stocker leurs des données où le coût par gigaoctet est relativement peu coûteux par rapport au stockage acheté.

Ce modèle offre un maximum de flexibilité pour le client et requiert de lourds investissements pour le fournisseur de services. Néanmoins, les contraintes principales de ces plateformes sont les exigences de sécurité et un niveau de latence acceptable. Les services offerts par les Cloud publiques ne sont pas identiques. Certains sont très extensibles avec un niveau de sécurité élevé et une meilleure qualité de services. D'autres sont moins robustes et moins sûrs, mais ils sont beaucoup moins chers à utiliser. Le choix dépendra de la nature et l'importance des données et le niveau du risque qu'on peut assumer. Les fournisseurs du Cloud publique les plus connus sont Google, Microsoft et Amazon.

1.2.1.4.2 Cloud Privé

Le Cloud privé est un ensemble de matériel, réseautage, stockage, services, applications et d'interfaces appartenant à une organisation et exploité par lui pour l'usage de ses employés, partenaires et clients. Il peut être créé et géré par un tiers pour l'usage exclusif d'une entreprise. C'est un environnement déployé au sein d'une entreprise ou organisation qui en est la propriétaire, en utilisant ses infrastructures internes. Les ressources sont détenues et contrôlées par le département informatique de l'entreprise ou les services sont accessibles via le réseau privé.

Le Cloud privé est un environnement hautement contrôlé se trouvant derrière un parefeu qui n'est pas ouvert à la consommation publique. Si une organisation gère un projet Big Data qui exige un traitement de quantités massives de données, le Cloud privé pourrait être le meilleur choix en termes de latence et de sécurité. Ce modèle correspond aujourd'hui à une évolution des Data Centers virtualisés et à l'émergence de l'IT as a

Service : le système d'information et les équipements informatiques qui se transforment en centre de services pour le reste de l'entreprise. Dans cette optique d'IT as a Service, on comprend parfaitement le principe du Cloud Computing en tant que service mesurable et facturable aux différentes divisions de l'entreprise.

Eucalyptus, OpenNebula et OpenStack sont des exemples de solutions proposées pour la mise en place d'un Cloud privé.

1.2.1.4.3 Cloud Communautaire

Le Cloud Communautaire est une extension du Cloud privé. Dans ce modèle, les ressources, services et la propriété sont partagés à l'échelle d'une communauté (exemple : à l'échelle d'un Etat, d'une ville, d'une académie, etc.). Les services sont accessibles via l'interconnexion de réseaux appartenant aux organisations de cette communauté.

1.2.1.4.4 Cloud hybride

Un Cloud hybride est une combinaison d'un Cloud privé combiné à l'utilisation de services de Cloud public. Certains services sont accessibles via un réseau privé et d'autres via Internet. Le futur devrait confirmer l'émergence du Cloud hybride qui consiste à la cohabitation et la communication entre un Cloud privé et un Cloud publique dans une organisation partageant des données et des applications.

1.2.1.5 Quelques exemples de Cloud Computing

Le Cloud est désormais ancré dans presque toutes les tâches que nous accomplissons que ce soit sur ordinateur ou smartphone. Ci-dessous quelques exemple :

— **Dropbox :** Cette application est devenue l'outil de stockage grand public fondée en 2007 par les étudiants du MIT Drew Houston and Arash Ferdowsi. Depuis maintenant 10 ans, la firme développe un service de stockage en ligne sur le cloud. Au même titre qu'Amazon et Box, Dropbox se présente comme un précurseur dans le domaine du stockage cloud. Au total, le service fédère aujourd'hui plus de 500 millions d'utilisateurs.

— **Amazon Photos :** C'est un service de stockage en ligne sécurisé pour les photos et vidéos. Chaque client d'Amazon bénéficie de 5 Go de stockage gratuit pour sauvegarder, partager et accéder à ses photos sur son ordinateur de bureau, son téléphone portable et sa tablette. Les membres d'Amazon Prime bénéficient d'un stockage de photos illimité. Ils bénéficient également de fonctions de recherche et d'organisation améliorées.

— **Google play music :** Elle a été lancé par Google et est totalement gratuite. On peut lire ses chansons depuis des terminaux Android, iOS, via un Chromecast, un PC sous Windows, un Mac ou encore directement sur des enceintes Sonos. Il n'y a pas d'application native pour les ordinateurs, ce qui pourra gêner certains utilisateurs. Outre le fait de proposer gratuitement un stockage plus important qu'Apple, Google Play Musique offre une qualité audio et une taille maximale de fichiers supérieures et prend en charge plus de formats.

1.2.2 Les technologies de traitement

Dans cette section nous parlerons de l'arrivée des technologies de traitement ajustées, plus spécialement sur la mise au point de modes de calcul à haute performance (MapReduce), nous parlerons de (Hadoop) une solution de big data très largement utilisée pour effectuer des analyses sur de très grands nombres de données et enfin nous clôturons cette section avec le développement de nouvelles bases de données adaptées aux données non structurées (NoSQL) et nous verrons qu'est-ce que le NewSQL. [12]

1.2.2.1 Hadoop

Hadoop est une plateforme open source de la fondation Apache, ayant une capacité de gérer des données volumineuses, qui sont structurées et non structurées. Elle est conçue pour trouver une solution aux problèmes liés à la volumétrie et la variété des données en les traitants sur des différents serveurs simultanément. Cette architecture va offrir une puissance, et un stockage importants, les données vont être par la suite répliquées et réparties sur les machines du cluster, grâce à un système de réplication de façon à garantir une très haute disponibilité des données en cas de défaillance d'un ou de plusieurs serveurs.[13]

Dans son principe, Hadoop se compose essentiellement de :

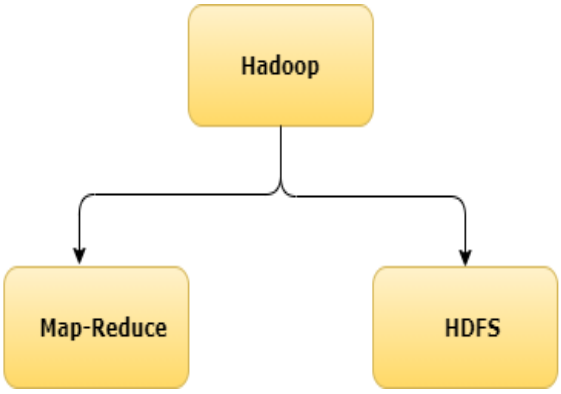

FIGURE 1.8 – Les composants d'Hadoop.

1.2.2.1.1 Système de gestion de fichiers HDFS (Hadoop Distributed File System)

HDFS est un système de fichiers aide à au stockage des données structurées ou non sur des machines distribués (cluster). Il s'appuie sur le système de fichier natif de l'OS pour présenter un système de stockage unifié reposant sur un ensemble de disques et de systèmes de fichiers hétérogènes. Ce système est basé sur la redondance dont une donnée est stockée sur au moins N volumes différents.[14]

1.2.2.1.2 Modèle de programmation Map-reduce

Grâce au framework MapReduce,il permet de traiter les immenses quantités de données. Plutôt que de devoir déplacer les données vers un réseau pour procéder au traitement, MapReduce permet de déplacer directement le logiciel de traitement vers les données. Ce modèle fera l'objet de la deuxième technologie que nous allons présenter.

1.2.2.1.3 Une collection d'outils spécifiques pour HDFS et Map Reduce comme des API et des frameworks.

1.2.2.2 Map-reduce

« MapReduce » a émergé comme modèle de programmation et une implémentation associée pour le traitement et la génération de grands volumes de données avec un algorithme distribué parallèle sur un cluster.
MapReduce est à l'origine une technique de programmation connue de longue date en programmation fonctionnelle s'inspirant des langages fonctionnels et plus précisément du langage Lisp.
MapReduce est un paradigme de programmation, dans lequel sont effectués des traitements parallèles et distribués de données potentiellement très volumineuses, supérieures en taille à 1 téraoctet.

Ce Framework popularisé par Google en 2004, est actuellement intégré dans plusieurs solutions Big Data.

1.2.2.2.1 Principe de MapReduce

Le principe de MapReduce est simple comme ci-illustré dans la **Figure 1.9** il s'agit de découper une tâche manipulant un gros volume de données en plusieurs tâches traitant chacune un sous-ensemble de ces données.

L'exécution des jobs se fait à l'aide d'un JobTracker et de TaskTrackers : lors de son exécution le job est soumis au JobTracker qui s'occupe de le distribuer au TaskTracker qui tourne sur chaque noeud. Le JobTracker choisit toujours les TaskTrackers qui sont les plus proches de l'emplacement de la donnée à traiter [15].
MapReduce est composé de deux fonctions Map() et Reduce() qui opèrent sur des couples (clés, valeurs) en entrée et en sortie :

Map

Transforme les données d'entrée en une série de couples clef /valeur et distribue le traitement sur les différents serveurs pour aboutir à un premier niveau de résultats de synthèse intermédiaires. Le parallélisme est la caractéristique de cette première phase : le nœud analyse le problème, le découpe en sous-problèmesou fragments et les délègue à d'autres nœuds (qui peuvent en faire de même récursivement) pourfaire exécuter l'opération MAP à chaque machine du clustersur un fragment distinct. Chaque nœud traite un sous-ensemble des données [16].

La fonction Map s'écrit de la manière suivante :

$$Map(cl1, valeur1) \rightarrow Liste(cl2, valeur2).$$

Reduce

Applique un traitement à toutes les valeurs de chacune des clés distinctes produites par l'opération MAP. Reduce consolide sur un nombre plus réduit de serveurs, jusqu'à un seul point pour simplifier, l'ensemble des résultats des différentes tâches de Map, et réalise une synthèse des résultats intermédiaires pour répondre à la requête du client : les nœuds les plus bas font remonter leurs résultats au nœud parent qui les avait sollicités. Celui-ci calcule un résultat partiel à l'aide de la fonction Reduce (réduction) qui associe

toutes les valeurs correspondantes à la même clé à une unique paire (clé, valeur). Puis il remonte l'information à son tour [16].

La fonction Reduce s'écrit de la manière suivante :

$$Reduce(cl2, Liste(valeur2)) \rightarrow Liste(valeur2).$$

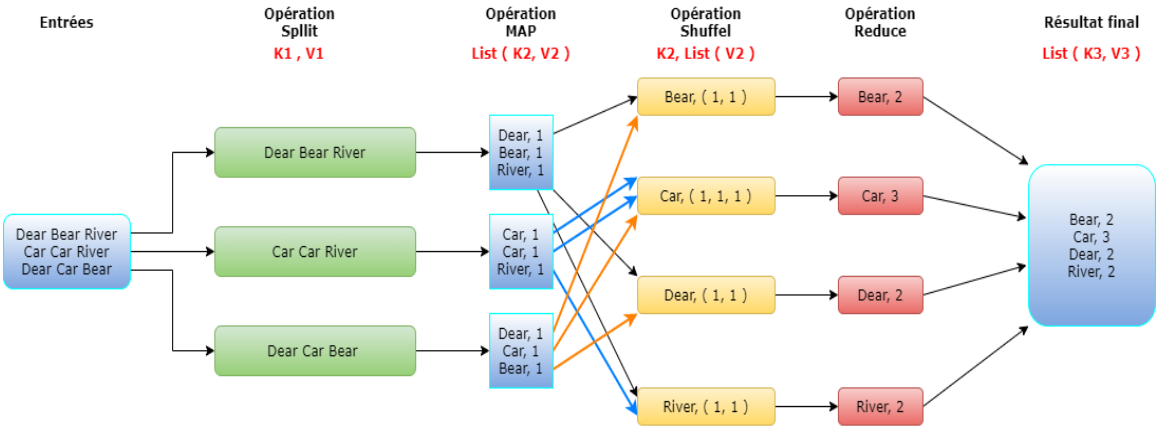

FIGURE 1.9 – Principe Map-Reduce.

1.2.2.2.2 Architecture fonctionnelle

L'architecture maître-esclave de MapReduce s'articule sur les 2 composants suivant (**Figure 1.10**) :

- Le maître MapReduce : le **JobTracker** ;
- Les esclaves MapReduce : les **TaskTracker**.

JobTracker [17]

- Gère l'ensemble des ressources du système ;
- Reçoit les jobs des clients ;
- Ordonnance les différentes tâches des jobs soumis ;
- Assigne les tâches aux TaskTrackers ;
- Réaffecte les tâches défaillantes ;
- Maintien des informations sur l'état d'avancement des jobs.

TaskTracker [17]

- Exécute les tâches données par le JobTracker ;
- Exécution des tâches dans une autre JVM (Child) ;
- A une capacité en termes de nombres de tâches qu'il peut exécuter ;
- Heartbeat avec le JobTracker : Le processus Heartbeat se charge de passer un message-aller vers le JobTracker indiquant son état.

L'exécution d'un job MapReduce, concerne quatre entités indépendantes [18] :

- Le client, qui soumet le job MapReduce ;
- Le JobTracker, qui coordonne l'exécution et le partage du job en sous tâches. Le JobTracker est une application Java dont la classe principale est JobTracker ;
- Les TaskTrackers, qui gèrent les tâches que le JobTracker lui a confiées. Les Tasktrackers sont des applications Java dont la classe principale est TaskTracker ;

— Le système de fichiers distribué, qui est utilisée pour le partage de fichiers de travail entre les autres entités (TaskTrackers).

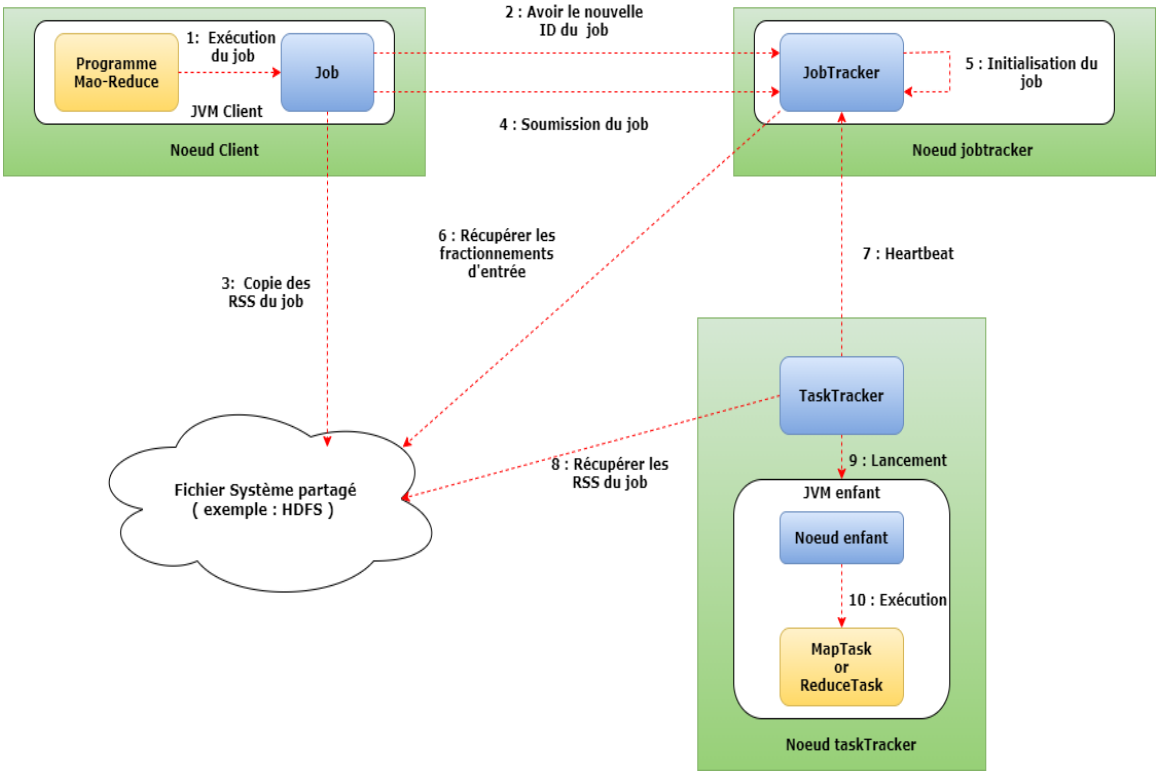

FIGURE 1.10 – Structure fonctionnelle de MapReduce.

Exemple d'utilisation de MapReduce : *Une anagramme est une transposition des lettres qui composent un mot, de telle sorte qu'elles forment un ou plusieurs autres mots ayant un autre sens. Les mots nacre, rance et ancre sont des anagrammes les uns des autres. A partir d'une liste de mots, on cherche à déterminer lesquels sont des anagrammes. On imagine qu'on souhaite exécuter le problème sur l'encyclopédie Wikipédia, il est donc nécessaire d'avoir une approche MapReduce pour facilement paralléliser l'exécution.*

1.2.2.3 Bases NoSQL

De nos jours, l'ubiquité de la connexion Internet est une réalité (les voitures que nous conduisons, les montres que nous portons, nos petits appareils médicaux domestiques, nos réfrigérateurs et congélateurs, nos Smartphones et ordinateurs portables). De plus, les données numériques produites par les êtres humains, dont les séquences vidéo, les photos et autres, atteignent des volumes importants de plusieurs EO par jour. Ces données actuellement stockées dans des bases qui leur ont été conçues spécifiquement sont gérés par des logiciels de gestion de bases de données volumineuses, jouant le rôle d'intermédiaires entre les bases de données d'un côté et les applicatifs et leurs utilisateurs de l'autre. On parle ici des bases de données non-relationnelles, dites NoSQL.

Concrètement une base de données NoSQL est une approche de la conception des bases et de leur administration particulièrement utile pour de très grands ensembles de données distribuées. Elle englobe une gamme étendue de technologies et d'architectures, afin de résoudre les problèmes de performances en matière d'évolutivité et de Big Data que les bases de données relationnelles ne sont pas conçues pour affronter. De plus elle

est particulièrement utile lorsqu'une entreprise doit accéder, à des fins d'analyse, à de grandes quantités de données non structurées ou de données stockées à distance sur plusieurs serveurs virtuels du Cloud.

Pour mieux comprendre les bases de données NoSQL et leur application dans le Big Data, nous allons consacrer par la suite un chapitre où nous détaillons cette technologie.

1.3 Les modes de stockage du Big Data

A l'ère du Big Data, la question qui revient le plus souvent est de savoir quelles sont les architectures de stockage les mieux adaptées pour soutenir des processus analytiques à grande échelle, plusieurs technologies rivalisent aujourd'hui dans ce domaine, certaines parfois Mieux adaptées à certains types de traitements que d'autres. Et en général, l'usage de l'une n'exclut pas celui d'une autre. En l´état actuel des développements technologique, quatre de ces dernières sont en lice pour les applications Big Data :

- Le stockage massivement distribué.
- Le NAS en mode Scale-out -En français la scalabilité horizontale- (ou NAS 21 en cluster).
- Les baies de stockage 100% Flash.
- Le stockage en mode objet.

1.3.1 Stockage distribué

Les architectures de stockage distribuées sont souvent associées au « grid computing », car elles évoluent en parallèle de l'environnement de calcul. Dans ces architectures, les capacités de traitement sont co-résidentes des capacités de stockage sur les noeuds de la grille. C'est par exemple le cas pour les clusters HDFS (le file system d'Hadoop) ou pour certains clusters à base de la technologie Red Hat Storage. Ceph est aussi un bon exemple dans le monde Open Source.

Le principal avantage de ces architectures est leur faible coût, mais aussi la proximité du stockage des capacités de calcul. Cette affinité peut permettre d'accélérer considérablement les traitements si l'on parvient à localiser, les traitements pertinents à certaines données sur les noeuds où sont stockées ces données (ce qui est l'objectif de mapreduce dans Hadoop).

Dans certains cas, les déploiements se font en utilisant les emplacements de disques internes aux serveurs. Dans d'autres cas, on s'appuie sur des JBOD connectés aux différents noeuds.

1.3.2 Stockage Scale-out NAS

C'est d'une certaine façon une variante du stockage distribué, à ceci près qu'à quelques rares exceptions, aucun traitement de calcul n'est effectué sur les noeuds de stockage distribués. La capacité CPU est intégralement dédiée à la gestion des fonctions de stockage. Les NAS en mode Scale-out ont des caractéristiques de performance uniques puisque l'ajout d'un noeud additionnel ajoute à la fois de la capacité, mais aussi de la connectivité additionnelle, donc de la bande passante, ce qui les rend très attractifs pour le stockage de quantités massives de données avec des contraintes de performances fortes.

Le Scale-Out NAS est ainsi massivement utilisé dans le monde du HPC et du calcul scientifique.

Un autre avantage par rapport aux systèmes de stockage distribués est que les NAS en cluster disposent en général d'un environnement logiciel riche avec des fonctions avancées de gestion du stockage (snapshots, réplication, compression, déduplication de données) qui font encore défaut aux technologies de stockage distribué open source. Autant de fonctions importantes lorsque l'on manipule de grandes quantités de données avec des contraintes de performances, de capacité et de disponibilité fortes.

Les NAS Scale-out modernes, chez NetApp, EMC-Isilon, IBM mais aussi chez des spécialistes comme Data Direct Networks ou Panasas peuvent gérer des dizaines de peta-octets, parfois sur un seul fichier système ; ce qui est suffisant pour bien des applications d'entreprises. Leur support de protocoles ouverts, comme CIFS ou NFS, permet aussi l'ingestion facile de données depuis des sources variées. Petit bémol, la performance de ces systèmes se dégrade en général lorsqu'ils doivent stocker des millions de petits fichiers. Ils tendent en général à être mieux adaptés au stockage de fichiers de grande taille.

1.3.3 Les baies de stockage 100% Flash

Elle souffre de leur réputation de cherté. De ce fait, elles ne paraissent pas, au premier abord, des solutions bien adaptées pour les applications Big Data. En fait, dans certains cas bien précis, où il est important d'obtenir des résultats d'analyse en quasi temps-réel, les baies 100% Flash sont incontournables. Elles sont par exemple un complément de choix pour les architectures In-Memory. Qui plus est, les architectures 100% Flash ne sont pas nécessairement monolithiques.

Il est possible dans certains cas d'assembler un cluster Hadoop avec des SSD. De même, les architectures Flash en cluster se multiplient. Un autre point à ne pas ignorer est que le prix des SSD et de la mémoire Flash en général ne cesse de baisser ; ce qui fait que l'écart de prix entre SSD et disques durs SAS se réduit, alors que les SSD ont un avantage majeur en matière d'IOPS. D'ailleurs, pour toutes les applications analytiques requérant un nombre massif d'IOPS et un faible temps de latence, les SSD sont aujourd'hui sans équivalent.

1.3.4 Stockage en mode objet

Le stockage en mode objet implique d'attribuer à chaque donnée des identifiants uniques, que l'on appelle les métadonnées. Compte tenu du fait que les objets ne sont ni compressés ni chiffrés, ils sont rapidement accessibles à très grande échelle. Il s'agit donc d'une solution idéale pour les applications cloud-native. Un objet est composé de :

- La donnée.
- Un identifiant unique : Cet identifiant correspond au chemin d'accès de la donnée. L'utilisateur dispose donc d'une base de référence unique quel que soit l'endroit où se trouve son objet.
- Des métadonnées : Il s'agit des informations liées au contexte de l'objet comme le type de données (vidéo, fichier, etc...), sa structure, etc... Ces métadonnées permettent de faire le lien entre plusieurs objets qui comportent ces mêmes métadonnées.

1.4 Les sources du Big Data

Le principale but des entreprises c'est analysés tout le volume de données qu'il ont a disposition.Cependant, avant de procéder à l'extraction des informations et des informations précieuses de ces mégadonnées, elles doivent avoir la connaissance d'où vient tout ce Big Data disponibles ?. Comme nous le savons, les données sont massives et existent sous diverses formes. S'il elles ne sont pas bien classé ou bien approvisionné, cela peut induire à gaspiller un temps et des ressources précieux sans avoir de résultats concrets.

Pour réussir avec le Big Data, il est important que les entreprises disposent du savoir faire pour passer en revue les différentes sources de données disponibles et classer en conséquence leur convivialité et leur pertinence [19].

Dans ce qui va suivre on classera les sources du big data en 2 classement différents, dans le premier classement on dira si les données sont internes ou externes à l'entreprise. le second classement est plus précis il est résumée par la figure ci-dessous :

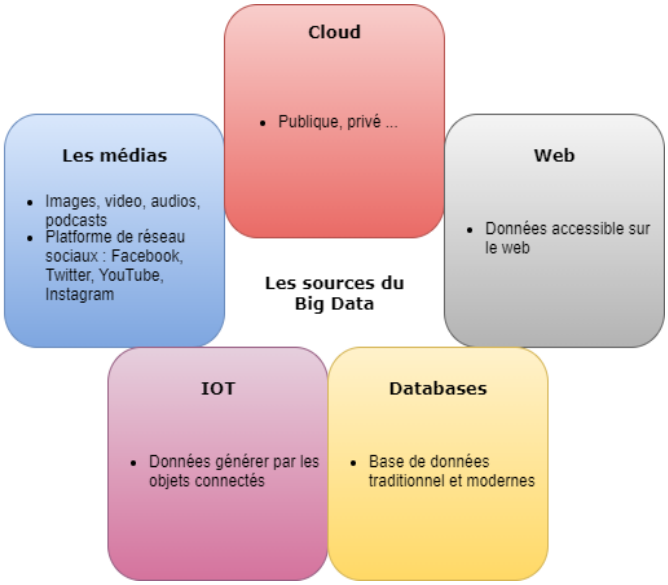

FIGURE 1.11 – Les sources du Big Data.

1.4.1 Données internes ou externes

Les données sont internes si une entreprise les génère, les possède et les contrôle. [20]

Exemple :

- *Module ERP d'entreprise.*
- *Documents internes.*
- *La flexibilité.*
- *Capteurs, contrôleurs.*
- *Centres d'appels internes.*
- *Journaux de site Web.*

Les données externes sont des données publiques ou des données générées en dehors de l'entreprise ; en conséquence, la société ne la possède ni ne la contrôle. [20]

Exemple :

- *Médias sociaux.*
- *Statistiques officielles.*
- *Prévisions météorologiques.*
- *Ensembles de données accessibles au public pour l'apprentissage automatique.*

1.4.2 Autres sources du big data

1.4.2.1 Les médias

Les médias sont considérés comme la source la plus populaire de données importantes, car ils fournissent des informations précieuses sur les préférences des consommateurs et l'évolution des tendances. Étant donné qu'ils sont auto-diffusés et qu'ils franchissent toutes les barrières physiques et démographiques, ils constituent le moyen le plus rapide pour les entreprises d'obtenir une vue d'ensemble approfondie de leur public cible, de tirer des schémas et des conclusions et d'améliorer leur prise de décision.

Exemple : Par media nous allons parler dès les médias sociaux et les plateformes interactives, comme Google, Facebook, Twitter, YouTube, Instagram, ainsi que les médias génériques comme les images, les vidéos, les audios et les podcasts qui fournissent des informations quantitatives et qualitatives sur chaque aspect de l'interaction avec l'utilisateur.

1.4.2.2 Le Cloud

Les entreprises ont pris de l'avance sur les sources de données traditionnelles en transférant leurs données dans le cloud. Le stockage dans le nuage permet de stocker des données structurées et non structurées et de fournir aux entreprises des informations en temps réel et des aperçus à la demande. Le principal attribut du cloud computing est sa flexibilité et son évolutivité. Comme les données volumineuses peuvent être stockées et obtenues sur des nuages publics ou privés, via des réseaux et des serveurs, le nuage constitue une source de données efficace et économique.

1.4.2.3 Le Web

Le web public constitue une source de données importante, répandue et facilement accessible. Les données sur le Web ou "Internet" sont généralement accessibles aux particuliers comme aux entreprises. De plus. L'énormité du web garantit la diversité de ses possibilités d'utilisation et est particulièrement bénéfique aux jeunes entreprises et aux PME, car elles n'ont pas à attendre de développer leur propre infrastructure et leurs propres référentiels de données avant de pouvoir exploiter des données volumineuses.

1.4.2.4 L'Internet Des Objets

Le contenu généré par les machines ou les données créées à partir de l'IoT constituent une source précieuse de données importantes. Ces données sont généralement générées par les capteurs qui sont connectés aux appareils électroniques. La capacité d'approvisionnement dépend de la capacité des capteurs à fournir des informations précises en temps réel. L'IoT prend maintenant de l'ampleur et comprend des données importantes

générées non seulement par des ordinateurs et des smartphones, mais aussi, éventuellement, par tout appareil pouvant émettre des données.

Grâce à l'IoT, les données peuvent désormais être obtenues à partir d'appareils médicaux, de processus automobiles, de jeux vidéo, de compteurs, d'appareils photo, d'appareils électroménagers ...

1.4.2.5 Les bases de données

Aujourd'hui es entreprises préfèrent utiliser diverses bases de données traditionnelles et modernes pour acquérir des données importantes et pertinentes. Cette intégration ouvre la voie à un modèle de données hybride et nécessite de faibles coûts d'investissement et d'infrastructure informatique. En outre, ces bases de données sont également déployées à plusieurs fins de veille économique. Ces bases de données peuvent ensuite permettre l'extraction d'informations qui sont utilisées pour générer des profits. Les bases de données les plus populaires comprennent une variété de sources de données, telles que MS Access, DB2, Oracle, SQL et Amazon Simple, entre autres.

Le processus d'extraction et d'analyse des données parmi de grandes sources de données est complexe et peut être frustrant et long. Ces complications peuvent être résolues si les organisations englobent toutes les considérations nécessaires relatives aux grandes données, prennent en compte les sources de données pertinentes et les déploient d'une manière bien adaptée à leurs objectifs organisationnels.

1.5 Les nouveaux métiers du Big Data

Les métiers du Big Data promettent des carrières florissantes pour les meilleurs spécialistes des données et parmi tous ces derniers on retrouve 5 métiers qui émergent [21] :

1.5.1 Architecte Big Data

Tout comme les architectes créent des structures physiques, les architectes de données élaborent des schémas pour des systèmes de gestion de données. Le rôle de l'architecte Big Data est d'agréger les données internes et externes, pour y parvenir L'architecte Big Data collabore avec les équipes informatiques et les managers pour déterminer une stratégie de données correspondants aux besoins de l'industrie. Il développe un inventaire des données nécessaires pour implémenter l'architecture, et recherche de nouvelles opportunités d'acquisition de données. Les métiers du Big Data demandent de l'organisation. Cet architecte est chargé d'identifier, d'évaluer les technologies de gestion actuelles des données, et de créer une vision fluide de la façon dont les données seront transmises au sein de l'entreprise.

Il développe ensuite des modèles de données pour les structures de bases de données, et va ensuite dessiner, documenter, construire et déployer des architectures et des applications de base de données. Les fonctionnalités techniques comme la scalabilité, la sécurité, la performance, la data recovery sont ensuite intégrées. Des mesures de précision et d'accessibilité des données sont implémentées. Parmi les métiers du Big Data, le rôle de l'architecte Big Data est enfin de surveiller, de raffiner en permanence les systèmes de gestion de données.

Exemple : *Le géant du réseau Cisco recrute les meilleurs architectes Big Data pour fournir des solutions et des Plateformes en tant que service pour le Big Data, les analytics et l'internet des objets. Selon Indeed, le salaire d'un architecte Big Data à Cupertino, en Californie, oscille entre 170 000 et 200 000 dollars par an.*

1.5.2 Ingénieur Big Data

L'ingénieur Big Data collabore avec des entreprises pour développer, entretenir, tester et évaluer des solutions Big Data. La plupart du temps, ils utilisent leurs compétences en technologies Hadoop comme MapReduce, HiveMongo DB, ou Cassandra. L'ingénieur Big Data crée des systèmes de traitement de données à grande échelle. Il est également expert en solutions de Data Warehousing et en technologies de bases de données.

1.5.3 Ingénieur Data Scientist

Le Data Scientist est chargé de développer les algorithmes nécessaires pour poser les bonnes questions et obtenir les bonnes réponses afin de tirer des informations pertinentes de vastes ensembles de données. Il est donc nécessaire d'avoir de solides connaissances en statistiques, en mathématiques, en modèles prédictifs et en stratégie d'entreprise.

Exemple : *Les universités américaines les plus prestigieuses tel que l'Applied Physics Laboratory de la Johns Hopkins University. recherchent actuellement les meilleurs Data Scientist avec les aptitude a diriger un service de sciences des données et d'analyse des données, et d'appliquer son expertise en analytiques, en analyses quantitatives, en techniques de Data Visualization et modélisation pour trouver des solutions pour les divers besoins techniques des entreprises.*

1.5.4 Ingénieur Big Data architecte de données

Il s'agit de l'un des métiers du Big Data qui combine les compétences en design et en implémentation de systèmes de gestion de données.

Exemple : *Verizon, le plus grand opérateur américain, cherche des personnes capables de prendre en main son flux de données massif. L'entreprise recherche des ingénieurs architecte de données pour gérer les solutions techniques, l'architecture de flux de données, et diriger les équipes dans l'implémentation de multiples projets complexes supportant les opérations et initiatives stratégiques sans fil des entreprises.*

1.5.5 Ingénieur Big Data Analytics

Ce poste est l'évolution du métier de Data Scientist. Il est destiné à une personne capable de superviser le flot de données de sa source à sa destination.

Exemple : *les entreprises tel que Slack recherche activement de tels Ingénieurs en Data Analytics.*

Bien évidemment la quantité de données ne cesse de croître, de ce fait les entreprise en de plus en plus de besoin, pour gérer traiter et mettre en place l'architecture ainsi que tout l'environnement nécessaire afin d'avoire les résultats escomptés, les tâches sont divisées en plusieurs partie ce qui pousse à voir l'apparition de nouveau metier pour le big data comme :

— **Data miner :** Son rôle assez proche de celui de Data Analyst, il est responsable principalement de la fouilleur de données.

— **Data protection officer :** C'est le chef d'orchestre de la conformité. Les responsables le consultent lors du traitement de données à caractère personnel, aussi bien en ce qui concerne la sécurité informatique que la sécurité juridique.
— **Business intelligence manager :** Il gère une équipe de développeurs et d'analystes,Ce gestionnaire est dans un premier temps chargé d'identifier les besoins de l'entreprise en matière de Business Intelligence. Il doit analyser les informations pertinentes, et fournir des rapports détaillés basés sur ces analyses pour permettre à l'entreprise d'agir.

Bien d'autre métier liée au big data existent, et d'autre feront sûrement leurs apparitions au cours des années à venir.

1.6 Les domaines d'application du Big Data

Dans ce qui suit on va citer brièvement quelques domaines d'utilisation du Big Data. Ce dernier trouve sa place dans de nombreux domaines :[22]

— **Les Banques :** La sanctuarisation de données anciennes dues à des contraintes réglementaires.
— **La Télécommunication :** L'analyse de l'état du réseau en temps réel.
— **Les Médias Numériques :** Le ciblage publicitaire et l'analyse de sites web.
— **Les Marchés Financier :** L'analyse des transactions pour la gestion des risques et la gestion des fraudes, ainsi que pour l'analyse des clients.

Dans une deuxième catégorie de secteur qui est plus hétérogène, les besoins, mais aussi l'utilisation qui est faite du Big Data, peuvent être très différents. On y trouve :[22]

— **Les Services Publics :** L'analyse des compteurs (gaz, électricité ...) et la gestion des équipements.
— **Le Marketing :** Le ciblage publicitaire et l'analyse de tendance.
— **La Santé :** L'analyse des dossiers médicaux et l'analyse génomique.

1.7 Conclusion

Au cours des dernières années un nombre important de sources de données est devenu disponible et à la portée de tous. Le Big Data en tant que discipline a déjà traversé son cycle d'implantation et est aujourd'hui transversal pour toute entreprise axée sur les données. Ainsi, les solutions et technologies Big Data représentent le moyen actuel d'atteindre la compétitivité et la croissance de notre société, où les appareils connectés augmentent rapidement, générant des données à des taux toujours croissants. Le volume de données générées par l'internet des objets (IoT) prend de plus en plus d'ampleur. Ainsi, pour pouvoir les prendre en charge et les analyser en temps réel, il est nécessaire de s'en remettre aux outils analytiques Big Data. C'est la raison principale de la relation étroite entre l'internet des objets et le big data, mais avant d'aller plus loin sur cette relation, il convient de revenir sur la présentation de cette technologie qu'est l'IoT qui fera l'objet du chapitre suivant.

Chapitre 2

L'Internet des Objets (IoT)

L'Internet des objets (IoT) est un domaine émergent de la technologie moderne qui a un impact sur les cas d'utilisation dans la gouvernance, l'éducation, les affaires, la fabrication, le divertissement, les transports, les infrastructures, les soins de santé, etc. D'année en année, l'Internet des objets continue à se développer. De nouvelles technologies voient le jour et le nombre d'appareils se multiplie de façon incommensurable. Aujourd'hui tous les objets sont presque connectable à Internet, que ça soit les lunettes, les montres, les téléviseurs ou bien les voitures, ou encore les machines industrielles ce qui a donné naissance au maisons et villes intelligentes.

Dans ce chapitre nous allons présenter l'IoT, énumérer quelques statistique qui lui sont associes, ensuite nous citerons ses composants et nous présenterons son architecture, puis nous allons voir les risque liés à l'IoT. Nous verrons ensuite ses domaines d'applications et son rôle dans le Big Data, nous terminerons ce chapitre par les différents modes de stockage de l'IoT

2.1 Historique et Définitions

La définition de ce qu'est l'Internet des objets n'est pas figée. Elle regroupe des dimensions d'ordres conceptuel et technique.
D'un point de vue conceptuel, elle est caractérise des objets physiques connectés ayant leur propre identité numérique et capables de communiquer les uns avec les autres. Ce réseau crée en quelque sorte une passerelle entre le monde physique et le monde virtuel. D'un point de vue technique, l'IoT consiste en l'identification numérique directe et normalisée (adresse IP, protocoles smtp, http...) d'un objet physique grâce à un système de communication sans fil qui peut être une puce RFID [1] [2], Bluetooth ou Wi-Fi, ou sur des longues distances (GSM, 3G, GPRS, CDMA,4G,5G)[23].

Il existe des dizaines de définitions de l'Internet des Objets. Certaines sont claires, mais trop spécifiques, elles échouent à rendre comptes de la diversité des concepts qui se réclament de la terminologie (réseaux de capteurs, systèmes cyber-physiques, intelligence ambiante, informatique diffuse . . .). D'autres sont plus synthétiques, mais c'est souvents au prix de la clarté. Dans ce qui suit, on va essayer de présenter les plus pertinentes [24] :

1. Le sigle RFID signifie Radio Frequency Identification(radio-identification). Il désigne une méthode
2. utilisée pour stocker et récupérer des données à distance en utilisant des balises métalliques

L'expression « Internet des Objets » n'est pas nouvelle. Elle fut adoptée dès 1999 par Kevin Ashton [3] [4] [5] qui travaillait alors en tant qu'assistant chef de marque chez Procter & Gamble [6] [7]. En 2007, Ashton développa ce concept dans un article :

> *" Si nous possédions des ordinateurs ayant toutes les connaissances possibles sur les objets, utilisant les données qu'ils recueillent sans aucune aide de notre part, nous serions en mesure de suivre et comptabiliser chaque objet, réduisant nettement les déchets, pertes et coûts. Nous serions avertis lorsqu'un objet doit être remplacé, réparé ou rappelé, s'il est frais ou périmé."*

> *" Nous devons autonomiser les ordinateurs en leur procurant leurs propres moyens de recueil des informations afin qu'ils puissent voir, entendre et sentir par eux-mêmes le monde dans toute sa magnificence aléatoire. La technologie RFID et des capteurs permet aux ordinateurs d'observer, identifier et comprendre le monde hors du cadre limité des données saisies par les personnes."*

Plus tard, en 2012, Rand Europe [8] [9] [10] a cherché à approfondir la définition de l'« Internet des Objets » dans un rapport de recherches adressé à la Commission européenne. Ce rapport indiquait :

> *" L'Internet des objets représente une extension de l'Internet tel que nous le connaissons aujourd'hui en créant un réseau omniprésent et auto organisé d'objets physiques connectés, identifiables et adressables permettant le développement d'applications au sein de secteurs verticaux clés et entre ces secteurs par le biais de puces, capteurs, déclencheurs et dispositifs miniatures intégrés et peu onéreux."*

Ces dernières années avec la croissance de la masse de données émis par l'IoT ainsi que l'apparition des systèmes distribués a fait en sorte que la définition de l'IoT évolue, ainsi plusieurs écoles de pensées coexistent et alimentent les recherches dans les perspectives parallèles et définissent l'IoT par : *"Des objets qui ont des identités et des personnalités virtuelles, opérants dans des espaces intelligents et utilisant des interfaces intelligentes pour se connecter et communiquer au sein de contextes d'usages. "*

Dans une perspectives plus technique, d'autres caractérisent l'IoT comme : *" un réseau de réseaux qui permet via des systèmes d'identification électroniques normalisés et unifiés, et des dispositifs mobiles sans fil, d'identifier directement et sans ambiguïté des entités numériques et des objets physiques et ainsi de pouvoir récupérer, stocker, transférer et traiter, sans discontinuité entre les mondes physiques et virtuelles, des données s'y rattachant. "*

3. Kevin Ashton (né en 1968) est un pionnier de la technologie britannique qui a cofondé l' Auto-ID
4. Center du Massachusetts Institute of Technology (MIT), qui a créé un système standard mondial
5. pour la RFID et d'autres capteurs
6. Procter & Gamble est une multinationale américaine spécialisée dans les biens de consommation
7. courante (hygiène et produits de beauté). Le siège de P&G est situé à Cincinnati en Ohio.
8. RAND Europe est un institut de recherche indépendant situé à Cambridge, Angleterre, à but non
9. lucratif dont la mission est d'aider à améliorer les politiques et la prise de décision par la recherche
10. et l'analyse.

Enfin, on peut aussi se focaliser autour d'une dimension plus sociotechnique en analysant l'IoT comme un élément dans un ensemble, et qui pourrait-être formulée comme : *" Des objets interconnectés ayant un rôle actif dans ce qui pourrait être appelé l'Internet du futur. "*

Selon l'UIT [11] l'Internet des Objets est une *" infrastructure mondiale pour la société de l'information, qui permet de disposer de services évolués en interconnectant des objets (physiques ou virtuels) grâce aux technologies de l'information et de la communication interopérables existantes ou en évolution. "*.[23]

La définition d'Ashton et celle de la RAND sont toutes les deux valables. Cependant la version de la RAND reprend le concept d'origine d'« ordinateurs autonomisés » d'Ashton et l'étend pour inclure les « objets physiques ». En d'autres termes, l'« Internet des Objets » ne s'appuie pas principalement sur les ordinateurs pour exister. Plutôt, chaque objet, même le corps humain, peut devenir partie intégrante de l'IdO s'il est équipé de certains composants électroniques. Ces composants varient certainement selon la fonction que l'objet doit remplir, mais ils s'inscrivent dans deux catégories générales :

- l'objet doit être capable de recueillir des données, généralement par le biais de capteurs
- l'objet doit être capable de transmettre ces données vers un autre emplacement via l'Internet.

Un capteur et une connexion sont ainsi les deux principaux « composants » d'un objet IdO.

Remarque : IoT et IIoT

Il ne faut pas confondre **l'IoT (Internet of thing)** avec **l'IIoT (Industrial Internet of Things)**. L'Internet des Objets Industriel désigne une sous-catégorie de l'IoT utilisé par les entreprises industrielles servant de passerelle entre les actifs physiques et numériques. Elle fait référence aux nombreux appareils et machines utilisés par les entreprises qui sont désormais connectés à Internet. Il ne s'agit pas uniquement des appareils IoT industriels, mais d'un vaste écosystème au sein duquel les personnes, les applications et les appareils interagissent.

En connectant les équipements à Internet, il est possible de générer des données permettant d'améliorer les performances, de prendre de meilleures décisions et de stimuler l'innovation. Ainsi, le marché de l'IIoT est en pleine croissance et devrait atteindre une valeur de 992 milliards de dollars en 2025.

Selon Accenture [12], d'ici 2030, l'IIoT ajoutera 14 billions de dollars à l'économie mondiale. Pour cause, l'Internet des Objets va permettre l'automatisation industrielle et l'augmentation de la productivité via l'exploitation des données [25].

11. UIT (l'Union internationale des télécommunications) est l'agence des Nations unies pour le développement spécialisé dans les technologies de l'information et de la communication, basée à Genève

12. Accenture est une entreprise internationale de conseil et de technologies créée en 1989 sous le nom d'Andersen Consulting

2.2 Les enjeux de l'IoT

l'IoT met en évidence deux objectifs : rendre les objets autonomes, et servir d'outil d'aide à la prise de décision. Ce concept vient ainsi ajouter une nouvelle dimension aux technologies de l'information et de la communication. En effet, à une permanente, en tous lieux, et pour tous (anytime, any place connectivity for anyone), on vient ajouter une connexion pour chaque chose (for anything).

Cette vision du monde ultra-connecté est une vision qui a dépassé le stade conceptuel, après avoir débuté avec la réalité augmenté, la géolocalisation et les QR Code, l'IoT couvre désormais plusieurs application telle que le paiement à carte sans contact, les voitures connectées aux maisons connectées et smart City et plus encore.

Plusieurs enjeux tourne au tours de l'Internet Des Objets et cela sur tous ces aspects comme par exemple :

- Reconnaître chaque objet de façon unique et recueillir les données stockées au niveau de l'objet.
- Intégrer les systèmes pour que les données soient transmises d'une couche à une autre.
- Stocker et analyser les données pour lancer des actions et aider à la prise de décisions.
- Transférer les données dans les mondes physiques et virtuels.
- Connecter les systèmes entre eux.

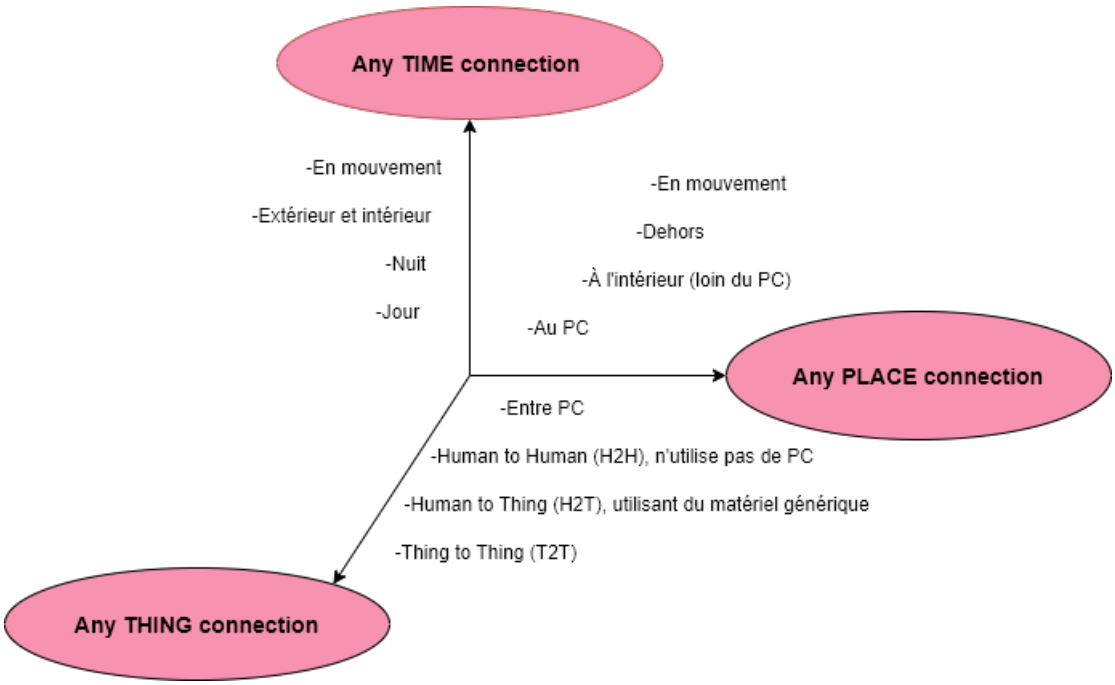

FIGURE 2.1 – Nouvelle dimension de l'IoT.

2.2.1 Les composants de l'IoT

Lier un objet ou un lieu à Internet est un processus plus complexe que la liaison de deux pages Web. Un système IoT réunit de nombreux acteurs et composants technologiques. Il est composé d'objets connectés, de réseaux de communication sans fil, de plateformes de collecte, d'hébergement, de traitement des données,

d'applications, services pour les utilisateurs finaux et d'une supervision, sécurisation de toute la chaîne [26] :

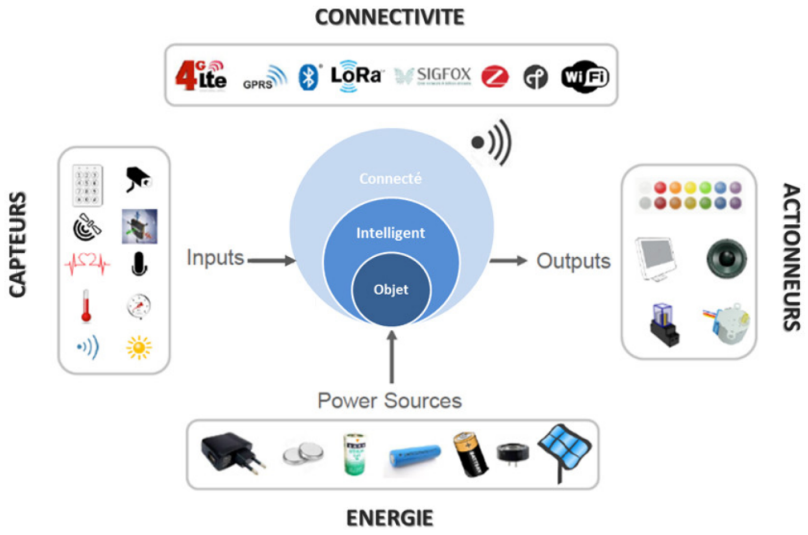

FIGURE 2.2 – Infrastructures de l'Iot.

a. **Les objets :**
L'objet connecté est d'abord un objet qui a une fonction mécanique et/ou électrique propre, il peut soit être conçu directement « connectable », soit il est déjà existant et la connectivité est rajoutée à posteriori.
L'objet connecté a pour fonction de collecter des données de capteurs, de traiter ces données et de les communiquer à l'aide d'une fonction de connectivité et de recevoir des instructions pour exécuter une action. Généralement ces fonctions de l'objet connecté nécessitent une source d'énergie, surtout quand les données sont prétraitées directement dans l'objet.

Les objets connectés (intelligents) : Ce sont des objets qui sont généralement connectés à Internet :

Exemple :

- Les objets « traditionnels » : ordinateurs, tablettes, smartphones, etc.

FIGURE 2.3 – Exemple d'objets traditionnels connectés.

- Les nouveaux objets connectés : appareils électroménagers, instruments de mesure, robots, serrures,machines-outils, bennes à ordures, drones, jouets, montres, véhicules, etc.

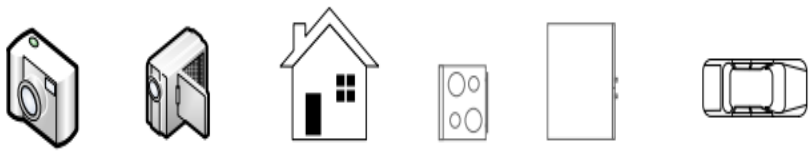

FIGURE 2.4 – Exemple de nouveaux objets connectés.

.

b. **Les capteurs :**
Les capteurs sont des dispositifs permettant de transformer une grandeur physique observée (température, luminosité, mouvement etc. . .) en une grandeur digitale utilisable par des logiciels. Il existe une très grande variété de capteurs de tous types, les objets connectés ont souvent la fonction de captation de ces grandeurs physiques sur leurs lieux d'utilisation.

Exemple de capteurs : *lumière, présence, proximité, position, déplacement, accélération, rotation, température, humidité, son, vibration, électrique, magnétique, chimique, gaz, flux, force, pression, niveau, . . . [26]*

Chaque capteur assure les fonctions suivantes :

- Acquisition des données.
- Calculer des informations à l'aide des valeurs collectées.
- Communiquer ces données à travers le réseau.

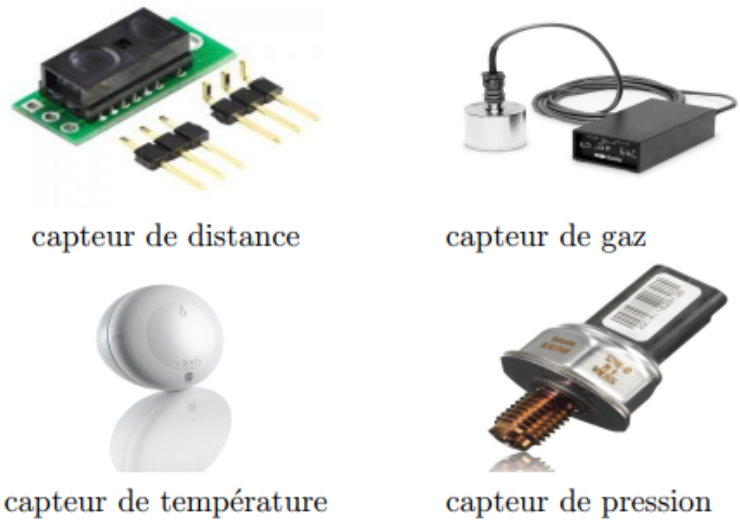

FIGURE 2.5 – Exemples de capteurs.

.

c. **Les données :**
Dans un projet d'IoT, les données sont nos diamants bruts. Il s'agit surtout des éléments bruts que nous récoltons depuis les objets ou les outils de process industriels pour l'IoT. Afin de créer de la valeur pour les utilisateurs de ces données, il est absolument nécessaire de les stocker, archiver et sauvegarder dans des bases de données et de correctement structurer cette dernière. En effet une base de données correctement structurée améliorera la performance des services IoT d'exploitation.

d. **Les sources d'énergie :**
Les sources d'énergie sont de 4 types : alimentation filaire pour les objets ayant accès à une prise de courant, piles ou batteries pour ceux qui n'y ont pas accès ou de manière occasionnelle (recharge), capteurs d'énergie ou « energy harvesting »(photovoltaïque, piezoélectrique, thermoélectrique, cinétique. . .) pour rallonger la durée de vie des objets à très faible consommation, et enfin les objets passifs sans piles qui sont alimentés par les ondes électromagnétique des lecteurs (RFID . . .).

e. **Les actionneurs :**
Les actionneurs sont des dispositifs qui transforment une donnée digitale en phénomène physique pour créer une action, ils sont en quelque sorte l'inverse du capteur, Ils permettent d'agir dans le monde physique, c'est-à-dire, changer son état.

> **Exemple :** *Afficheurs, Alarmes, Caméras, Haut-parleurs, Interrupteurs, Lampes, Moteurs, Pompes, Serrures, Vannes, Ventilateur, Vérins ...*

f. **La connectivité :**

Grace a une connectivité adapté les objets pourront d'une part remonter des informations telles que leur identité, leur état, une alerte ou les données de capteurs, et d'autre part recevoir des informations telles que des commandes d'action et des données. Le module de connectivité permet aussi de gérer le « cycle de vie de l'objet », c'est-à-dire, l'authentification et l'enregistrement dans le réseau, la mise en service, la mise à jour et la suppression de l'objet du réseau.

Toutes les technologies ne sont pas adaptées à tous les cas d'usages et leur déploiement [27] , le tableau suivant donne un a perçu des différents technologies permettant cette connectivité :

Communication à courte porté	
Technologie	**Caractéristiques**
Wifi	-Transmission sans fil à moyenne portée. -Une fréquence qui varie entre 2.5 GHZ et 5.0 GHZ . -Distance de transmission qui va de 10 m jusqu'à 100 m
Bluetooth	-Transmission sans fil à courte portée. -Une bande de fréquence de 2,4 GHz. -Distance maximale jusqu'à 15 m
M2M	-Couverture mondiale importante. -Grande consommation énergétique et faible portée. -Couverture mondiale encore faible.

Table 2.1 – Exemple de Technologie des réseaux à courte porté

Communication à longue porté	
Technologie	**Caractéristiques**
satellite	-De 4 à 8 GHz(C band), de 10 à 18 GHz(Ku band) et de 18 à 31 GHz (Ka band). -Un débit de données de 16 kbps à 155 mbps . -Distance de transmission à l'échelle terrestre.
SigFox	-Fréquence de 902 à 928 MHz. -Un débit de données arrivant jusqu'à 100 bps.
LoRa	-Fréquence de 868 MHz en Europe et 915 MHz en Amérique. -Faible consommation énergétique. -Couverture mondiale encore faible.

Table 2.2 – Exemple de Technologie des réseaux a longue porté

g. **Analyses et exploitation des données :**

Les analyses font le traitement de conversion de données analogiques d'équipements intelligents connectés et de capteurs en données utilisables pouvant être traitées, interprétées et utilisées pour des analyses détaillées. L'analytique intelligente est inévitable pour la gestion et l'amélioration des technologies IoT dans un système complet.

L'un des plus grands avantages d'un système IoT efficace est l'analyse intelligente en temps réel qui aide les ingénieurs à trouver les irrégularités dans la collecte de données et la rapidité d'action préventive d'un scénario indésirable. Les fournisseurs de services peuvent se préparer à de Nouvelles étapes si les informations sont collectées correctement et au bon moment. Aussi Les données collectés, sont transmises dans des réseaux virtuel tel que le Cloud. Leur traitement et stockage sont assurés par les opérateurs

du Cloud dans le but de produire des services.

2.2.2 Évolution l'IoT

Les objets connectés se développent très vite sans que tout le monde n'ait pris conscience du mouvement. La concurrence accélère le phénomène. Dès qu'un produit marche, toutes les sociétés se précipitent pour sortir leur version de celui ci. Mais concrètement combien existe-t-il aujourd'hui d'objets connectés et quelle évolution est prévue ? Quels sont les chiffres ?

Il y a 10 ans, l'IoT était encore une technologie confidentielle. Selon Statista [13], on comptait à peine 900 millions d'objets connectés à l'échelle mondiale. D'ici la fin de 2025, ce nombre pourrait atteindre entre 60 et 80 milliards [28].

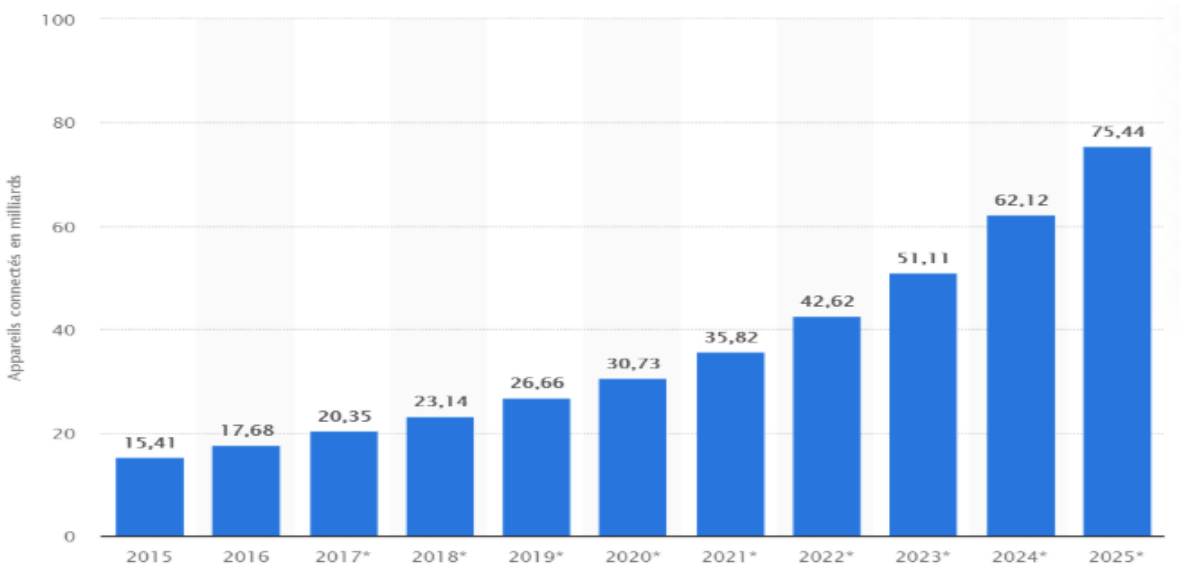

FIGURE 2.6 – Nombre d'appareils connectés dans le monde de 2015 à 2025.

Selon le rapport annuel de prévisions sur les tendances du networking *"Cisco Annual Internet Report"* :

- Les appareils et applications de l'Internet des Objets représentent une part toujours plus importante des connexions réseau. D'ici 2023, les communications M2M pourraient représenter 50% des 14,7 milliards de connexions. Ces communications M2M représentaient 33% (6,1 milliards) des connexions en 2018 et seulement 3,1% en 2017.Cette croissance rapide est liée à une variété d'applications M2M en plein essor, telles que les compteurs connectés, la surveillance vidéo, le monitoring de santé, le transport intelligent ou le tracking de colis.

- Le nombre de connexions augmente, mais le trafic croît encore plus rapidement du fait de l'utilisation d'applications vidéo sur les connexions M2M. D'autres applications nécessitant beaucoup de bande-passante et sensibles à la latence, comme la télémédecine et les systèmes de navigation intelligents pour les automobiles, entrent aussi en jeu.

13. Statista est un portail en ligne allemand offrant des statistiques issues de données d'instituts, d'études de marché et d'opinion ainsi que de données provenant du secteur économique

- L'apparition de la 5G va permettre de nouvelles applications IoT avec une portée étendue et des vitesses plus importantes. Ce réseau va aussi permettre une connexion d'accès pour des applications exigeant plus de bande-passante et moins de latence. Selon un billet de blog publié par Thomas Barnett de Cisco, la 5G va donc permettre des innovations considérées comme impossibles auparavant.

- À travers ce document, Cisco émet aussi des prédictions sur l'évolution de l'usage du réseau d'ici 2023. Selon les analystes, la vitesse moyenne de bande-passante devrait passer de 46 à 100Mbps. De plus, 45% de tous les appareils en réseau seront connectés à des réseaux mobiles : 3G, 4G, 5G, LPWA. Néanmoins, 55% seront connectés via le WiFi. La vitesse de connexion de ce protocole devrait tripler pour passer de 30 Mbps en 2018 à 92 Mbps en 2023.

- L'apparition de la 5G va permettre de nouvelles applications IoT avec une portée étendue et des vitesses plus importantes. Ce nouveau réseau va aussi permettre une connexion d'accès pour des applications exigeant plus de bande-passante et toujours moins de latence. Selon un billet de blog publié par Thomas Barnett de Cisco, la 5G va donc permettre des innovations considérées comme impossibles auparavant.

- D'ici 2023, la vitesse des réseaux cellulaires moyenne à l'échelle mondiale sera quant à elle de 43,9 Mbps contre 13,2 Mbps en 2018. Ceci représente une multiplication par 3,3. La vitesse de bande passante fixée moyenne doublera quant à elle pour passer de 46 Mbps en 2018 à 110 Mbps en 2023 [29].

Cependant, alors que la technologie progresse, les menaces de cybersécurité se font toujours plus pesantes. La fréquence des attaques DDoS [14] a augmenté de 39% en 2019, et la taille du pic d'attaques a augmenté de 63% en un an. Entre 2018 et 2019, les attaques entre 100Gbps ont augmenté de 776% et le nombre total d'attaques DDoS pourrait doubler passant de 7,9 millions en 2018 à 15,4 millions en 2023...

2.2.3 Architecture d'un système IoT

L'architecture d'un système IoT est composée de plusieurs niveaux qui communiquent entre eux, bien que chaque système IoT soit différent, la base de chaque architecture de l'Internet des Objets ainsi que son flux de processus de données général sont à peu près les mêmes [30].

14. Une attaque DDoS (Distributed Denial of Service) ou en français « attaque par déni de service distribuée » est une attaque informatique consistant à prendre pour cible un système informatique en l'inondant de messages entrants ou de requêtes de connexion afin de provoquer un déni de service

Les niveaux de communication entre les composants de l'IoT :

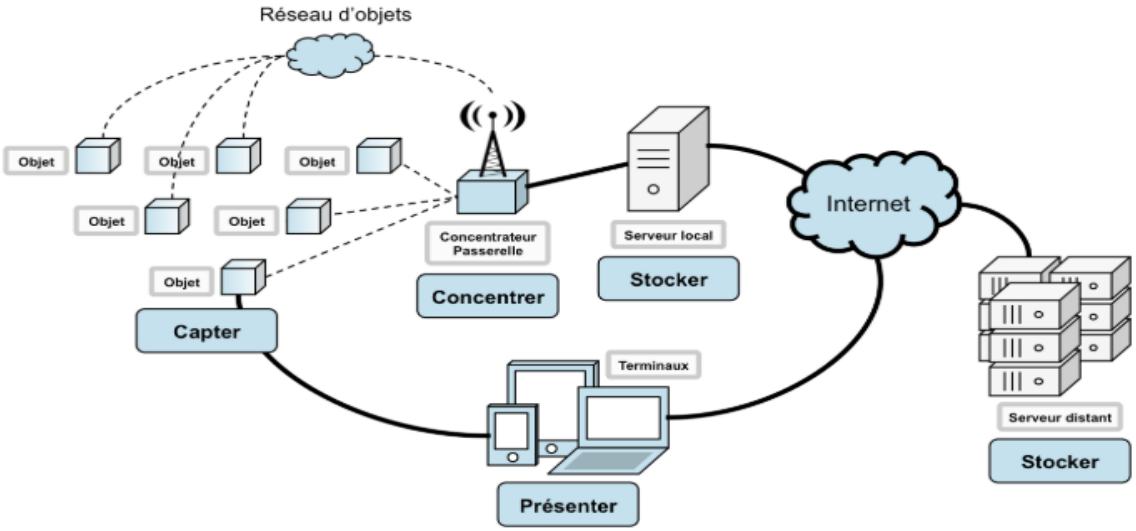

FIGURE 2.7 – Architecture de l'IoT.

1. **Capter :**
   Désigne l'action de transformer une grandeur physique analogique en un signal numérique.

2. **Concentrer :**
   Permet d'interfacer un réseau spécialisé d'objet à un réseau IP standard (e.g. WiFi) ou des dispositifs grand public.

3. **Stocker :**
   Qualifie le fait d'agréger des données brutes, produites en temps réel, méta taguées, arrivant de façon non prédictible.

4. **présenter :**
   Indique la capacité de restituer les informations de façon compréhensible par l'homme, tout en lui offrant un moyen d'agir et/ou d'interagir.

Deux autres processus n'apparaissent pas sur le schéma, car ils sont à la fois transverses et omniprésents :

5. **Le traitement des données :**
   C'est un processus qui peut intervenir à tous les niveaux de la chaîne, depuis la capture de l'information jusqu'à sa restitution. Une stratégie pertinente, et commune quand on parle d'Internet des objets, consiste à stocker l'information dans sa forme intégrale. On collecte de manière exhaustive, « Big Data », sans préjuger des traitements qu'on fera subir aux données. Cette stratégie est possible aujourd'hui grâce à des architectures distribuées type NoSQL, capables d'emmagasiner de grandes quantités d'information tout en offrant la possibilité de réaliser des traitements complexes en leur sein (Map/Reduce par exemple).

6. **La transmission des données :**
   C'est un processus qui intervient à tous les niveaux de la chaîne. Deux réseaux, supports des transmissions, cohabitent généralement :

   - **Réseau local de concentration :**On utilise alors des technologies comme wifi, ZigBee [15] [16] [17],Z-Wave [18] [19] [20] Bluetooth
   - **Réseau WAN :** Permettant d'interconnecter les réseaux spécialisés et de les interfacer avec des fermes de serveur [21]. On utilise alors les réseaux cellulaires (GSM, UMTS, LTE) ou encore les connexions physiques standard (Ethernet, fibre optique). Ces réseaux sont généralement connectés à Internet.

   Remarque : *Les technologies de transmission utilisées dépendent essentiellement de l'application et du contexte. La transmission peut par exemple exploiter le Push reposant sur Comet* [22] [23] [24] *ou WebSocket* [25] [26]*. Les canaux peuvent être bidirectionnels si l'application autorise une rétroaction. Dans certains cas, ces canaux devront transmettre les données en temps réel, dans d'autres cas, le temps ne sera pas un facteur déterminant.*

Pour mieux représenter le parcours des données au seins d'une architecture IoT on propose le schémas suivant :

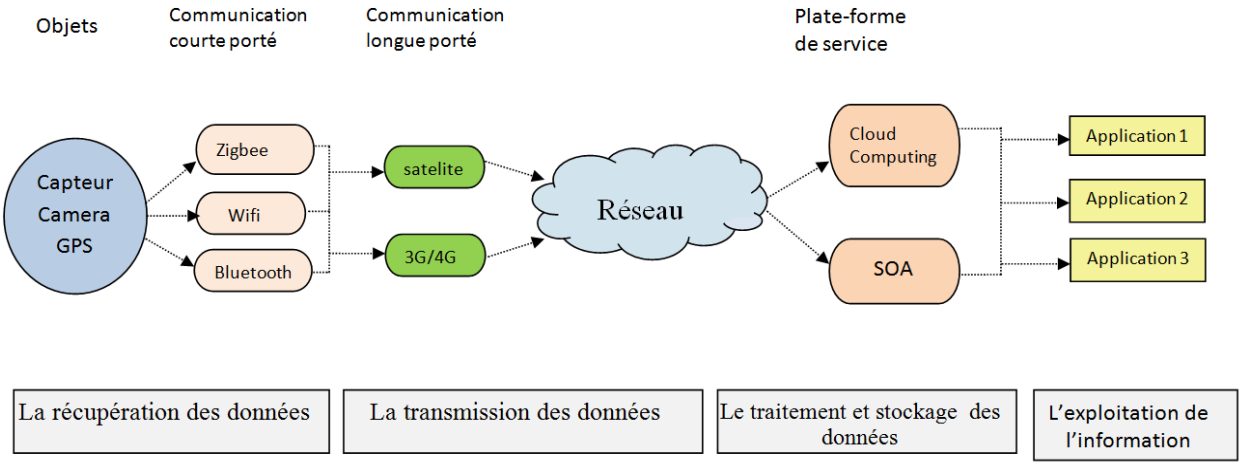

FIGURE 2.8 – Parcours des données IoT.

15. Zigbee est une spécification pour une suite de protocoles de communication de haut niveau utilisés
16. pour créer des réseaux personnels avec de petites radios numériques de faible puissance , telles que
17. pour la domotique ...
18. Z-Wave est un protocole de communication sans fil utilisé principalement pour la domotique .
19. Il s'agit d'un réseau maillé utilisant des ondes radio à faible énergie pour communiquer d'un appareil
20. à un autre
21. Fermes de serveur : Centre de données regroupant un très grand nombre d'ordinateurs (serveurs)
22. Comet est un modèle d' application Web dans lequel une demande HTTPS de longue date permet
23. à un serveur Web de transmettre des données à un navigateur , sans que le navigateur le demande
24. explicitement
25. WebSocket est un protocole de communication informatique différent de HTTP, fournissant
26. des canaux de communication en duplex intégral sur une seule connexion TCP

2.2.4 Les risques lies à l'Internet des objets

Ces dernières année nous on montrer que les possibilités de l'IoT ne sont limitées que par notre imagination. En particulier, lorsque l'on considère toutes les données échappant à l'enregistrement, toutes les bribes d'informations qui nous glissent entre les doigts et la manière dont l'IoT nous permettra de finalement recueillir ces données et de les utiliser comme l'humanité ne les a jamais utilisées. Il est facile d'ignorer le côté sombre du nouveau monde de l'IoT. Or les entreprises ne peuvent pas se permettre d'investir dans leurs systèmes IoT sans d'abord comprendre les risques non négligeables inhérents à tout système connecté à Internet [31].

Depuis le jour où nous avons allumé le premier ordinateur, nous savons que notre dépendance vis-à-vis de la technologie peut conduire à des perturbations de toutes ampleurs. Il ne s'agit pas de dissuader les entreprises d'adopter l'IoT ; les possibilités de cette technologie surpassent largement ses risques. Cependant toutes les entreprises doivent comprendre que pour chaque problème résolu par l'IoT, un autre problème est engendré. Dans cette section, nous allons présenter quelques risques inhérents à l'IoT ainsi que les recommandations techniques pour les minimiser :

1. **La vie privée :**
   Lorsque des milliards de capteurs dans le monde recueillent en permanence des données sur leur environnement, qui inclut les êtres humains, les préoccupations s'agissant de la vie privée dans le monde de l'IoT prennent une place centrale. La majorité du monde développé a tenté de protéger les consommateurs de l'utilisation illicite des informations confidentielles, mais dans bien des cas la législation n'est pas adéquate pour répondre aux multitudes de nouveaux modes d'acquisition et d'utilisation des informations, La capacité de l'IoT à suivre et enregistrer les actions humaines soulève de nombreuses questions éthiques qui n'ont pas encore trouvé toutes leurs réponses.

   Exemple :
   - *Un employé peut-il être puni sur la foi de données recueillies par un objet de l'IoT ?*
   - *Un employeur doit-il informer ses employés au sujet des capteurs qui suivent leur comportement ?*

2. **La cybersécurité :**

   Avec l'évolution des nouvelles technologies et de l'IoT, ce n'est plus seulement les entreprises et les banques qui risquent de connaître de nouvelles formes de cyber-attaques, car les individus sont également vulnérables.
   Les choses deviennent d'autant plus sérieuses lorsque l'on connaît l'ampleur de l'évolution des attaques spécialement conçues pour compromettre les dispositifs médicaux. Elles concernent à la fois la santé et les données personnelles des patients [32].
   Aujourd'hui avec l'apparition de la 5G et l'Intelligence artificielle, bien qu'elles présentent de nombreux avantages mais présentent aussi de nouveaux risques de cybersécurité pour l'Internet des Objets. C'est du moins l'avis d'une immense majorité de spécialistes interrogés par l'entreprise Information Risk Management [27] dans le cadre d'un rapport intitulé Risky Business [33].

27. Entreprise britannique spécialisée détenue par Altran qui est une entreprise de conseil en ingénierie fondée en France en 1982

Exemple d'une attaques regroupant l'IoT et l'IA à traver le réseau : *En août 2019, des criminels sont parvenus à utiliser un logiciel IA pour imiter la voix d'un CEO*[28] *allemand et persuadé sa filiale britannique d'effectuer un virement de 220 000 euros vers un compte bancaire frauduleux*

3. **La responsabilité :**

S'agissant des véhicules autonomes tels que les voitures sans conducteur, nous sommes confrontés à un dilemme éthique : durant les secondes précédant un accident, un véhicule autonome devrait il faire tout ce qu'il peut pour protéger ses passagers, même si cela implique de causer du tort à d'autres automobilistes ou piétons ? Lorsqu'un être humain est derrière le volant, les dommages collatéraux, aussi terribles soient-ils, ne posent pas vraiment de problème éthique. Un être humain en danger ne peut pas être tenu responsable lorsque son instinct de survie lui fait faire une embardée vers un piéton. Mais lorsque des machines prennent les décisions, un piéton blessé lors d'un accident est-il en droit de s'en prendre au constructeur automobile ? Un conducteur est-il en droit de poursuivre un constructeur automobile suite à un accident lors duquel ce conducteur a été blessé ? Comme le souligne un rapport de la commission européenne sur les dilemmes éthiques inhérents à la technologie de l'IoT, « Les gens ne sont pas habitués aux objets ayant leur propre identité ou agissant par eux-mêmes, en particulier s'ils agissent de manière imprévisible. »
D'autres questions de responsabilité émergent s'agissant de la propriété des données. Avec des milliards de dispositifs recueillant des données, il devient plus difficile de savoir qui est responsable de quelles données [31].

2.2.5 Les recommandations techniques :

a. **Les appareils IoT devraient utiliser les meilleures pratiques logicielles actuelles :**

- Les appareils IoT doivent être livrés aux clients ou aux points de vente avec des logiciels raisonnablement à jour qui ne contiennent pas de vulnérabilités graves connues.
- Ces apparils devraient disposer d'un mécanisme de mise à jour logicielle sécurisée et automatisée.
- Les appareils IoT doivent utiliser une authentification forte par défaut.

  Exemple : *Les protéger par mot de passe et qu'ils n'utilisent pas de noms d'utilisateur et de mots de passe courants ou facilement devinables (par exemple, «admin», «mot de passe» ...).*

- Les configurations des appareils IoT doivent être testées et renforcées.

28. Chief executive officer :la personne occupant le poste le plus important dans une entreprise

b. **Les appareils IoT doivent suivre les meilleures pratiques de sécurité et de cryptographie :**

Par exemple, sécuriser les communications à l'aide de TLS [29] ou de la LWC [30]. Les meilleures pratiques de chiffrement supplémentaires incluent :

- Chiffrer les communications de configuration (commande et contrôle) par défaut.
- Communications sécurisées vers et depuis les contrôleurs IoT.
- Crypter le stockage local des données sensibles.
- Authentifier les communications, les modifications logicielles et les demandes de données.
- Utiliser des informations d'identification uniques pour chaque appareil.
- Utiliser des informations d'identification pouvant être mises à jour.
- Fermer les ports inutiles et désactiver les services inutiles.
- Utiliser des bibliothèques activement gérées et prises en charge.

c. **Les appareils IoT doivent être restrictifs plutôt que permissifs dans la communication :**

Lorsque cela est possible, les appareils ne doivent pas être accessibles via les connexions entrantes par défaut. Les appareils IoT ne doivent pas compter uniquement sur le pare-feu du réseau pour restreindre la communication, car certaines communications entre les appareils de la maison peuvent ne pas traverser le pare-feu.

d. **Les appareils IoT doivent continuer à fonctionner si la connectivité Internet est interrompue :**

Un appareil IoT il faut qu'il puisse exécuter sa ou ses fonctions principales (par exemple, un interrupteur d'éclairage ou un thermostat doit continuer à fonctionner avec des commandes manuelles), même s'il est n'est pas connecté à Internet car la connectivité Internet peut être interrompue pour des raisons allant d'une mauvaise configuration accidentelle à une attaque intentionnelle. Les dispositifs IoT qui ont des implications pour la sécurité des utilisateurs devraient continuer à fonctionner en mode déconnecté pour protéger la sécurité des consommateurs.

e. **Les appareils IoT devraient continuer à fonctionner si le back-end cloud échoue :**

De nombreux services qui dépendent ou utilisent un back-end cloud peuvent continuer à fonctionner, même dans un état dégradé ou partiellement fonctionnel, lorsque la connectivité au back-end cloud est interrompu ou le service lui-même échoue.

f. **Les appareils IoT doivent prendre en charge les meilleures pratiques d'adressage et de dénomination :**

29. La Transport Layer Security ou « Sécurité de la couche de transport », et auparavant son prédécesseur la Secure Sockets Layer ou « Couche de sockets sécurisée », sont des protocoles de sécurisation des échanges sur Internet

30. La cryptographie LWC ou Lightweight est principalement un compromis entre sécurité et légèreté (charge de calcul) et en tant que telle pour différentes applications IOt

De nombreux appareils IoT peuvent rester déployés pendant plusieurs années après leur installation. La prise en charge des derniers protocoles d'adressage et de dénomination garantira que ces appareils resteront fonctionnels pour les années à venir :

- **IPv6** Les appareils IoT doivent prendre en charge la version la plus récente du protocole Internet, IPv6.
- **DNSSEC** Les appareils IoT doivent prendre en charge l'utilisation ou la validation des extensions de sécurité DNS (DNSSEC [31]) lorsque des noms de domaine sont utilisés.

g. **Les appareils IoT doivent être livrés avec une politique de confidentialité facile à trouver et à comprendre :**

Cette politique doit être facile à trouver et à comprendre pour un utilisateur type.

h. **La chaîne d'approvisionnement IoT devrait jouer son rôle pour résoudre les problèmes de sécurité et de confidentialité IoT :**

Les utilisateurs finaux des appareils IoT dépendent de la chaîne d'approvisionnement IoT, du fabricant au détaillant, pour protéger leur sécurité et leur confidentialité, et certaines ou toutes les parties de cette chaîne d'approvisionnement IoT joue un rôle critique tout au long du cycle de vie du produit. En plus des autres recommandations il faut que la chaîne d'approvisionnement IoT prenne des mesures en se qui concerne par exemple les mécanisme de réinitialisation et signaler la découverte et la correction des vulnérabilités ... [34]

2.3 Domaines d'application de l'Internet des objets

Au fil des deux dernières années, l'Internet des Objets a connu un essor particulièrement fulgurant. À l'heure actuelle, même notre brosse à dents peut être connectée à internet tel que la brosse à dent **Aras** munie d'une intelligence artificielle [35] Le phénomène a pris une telle envergure que l'utilité réelle de l'IoT est parfois remise en question.

Sans nul doute, dans quelques années absolument tous les secteurs seront affectés par l'IoT. Le rythme d'adoption, associé aux attentes et demandes des consommateurs, fera de tout secteur non tourné vers l'IoT ainsi que les entreprises individuelles, une pièce de musée. Cela dit, beaucoup de secteurs ont du temps pour comprendre l'IoT et la manière dont ce dernier pourrait les aider à atteindre leurs objectifs stratégiques à long terme.
Dans ce qui va suivre nous allons montrer des exemples actuels de la manière dont certains secteurs ont commencé à utiliser l'IoT. Sachant que les usages de l'IoT couvrent d'innombrables secteurs [31] :

31. DNSSEC est un protocole standardisé par l'IETF permettant de résoudre certains problèmes de sécurité liés au protocole DNS

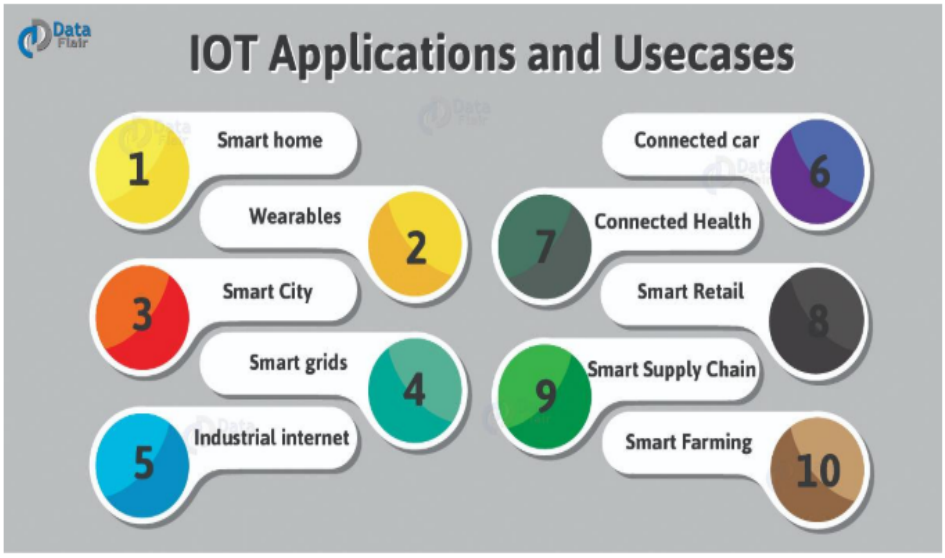

FIGURE 2.9 – Domaines d'application de l'IoT.

a. **Applications grand public :**

Les clients bénéficient de l'évaluation et de l'optimisation des informations de l'IoT. La technologie IoT se comporte comme une équipe de conseillers, d'assistants et de sécurité. Les applications Internet des objets dans le domaine des consommateurs peuvent varier de celles assez simples et bon marché qui incluent des gadgets de santé privés à des programmes intelligents d'automatisation domestique de haute qualité. Ainsi, les instances d'utilisation de l'IoT, les appareils et les packages pour les consommateurs sont également très variés [36].

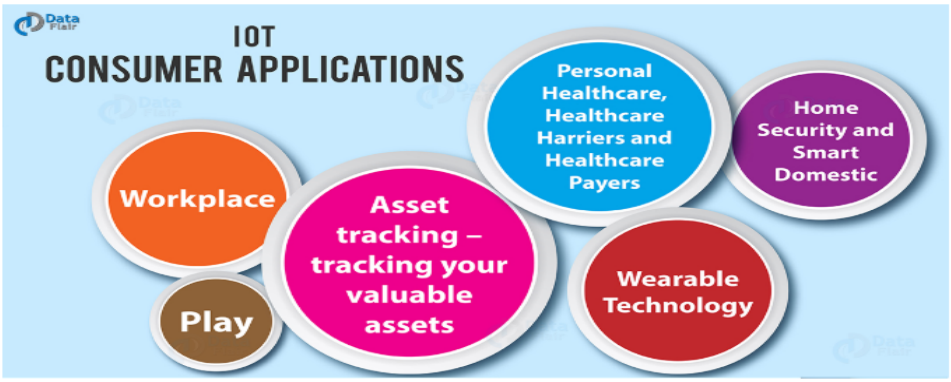

FIGURE 2.10 – Application grand public de l'IoT.

Exemple :

- *Sécurité domestique et domestique intelligent : Une cuisine IoT prépare des repas ou vous aide vraiment à les préparer.L'IoT peut vraiment agir en tant que maman ou papa en contrôlant l'accès, en fournissant des fournitures et en alertant les bonnes personnes en cas d'urgence.l'IoT peut aussi veillez sur nous 24/24. Il pourrait jeter un œil sur des individus suspects à des kilomètres. . .*
- *Permet le suivi des actifs : suivi intelligent des téléphones portables à la surveillance GPS des animaux de compagnie et au suivi de tout actif que vous souhaitez.*

Nous utilisons plusieurs autres objets connectés au quotidien, ça peut aller de la simple utilisation de smartwatch à la vérification de notre état de santé.

b. **Application de l'IoT dans l'éducation :**

Au niveau du secteur de l'éducation l'IoT est très importante car elle permet entre autre de pouvoir se passer des méthodes classique d'apprentissage et propose maintenant des cours sur les plateforme e-learning diverses et variées qui produit une acquisition interactive de connaissances avec la diversité de contenu. Dans les écoles primaires on peut suivre la position des enfants lors des randonnées par exemple et savoir leurs états de santé à tout moment. Aujourd'hui il existe des établissements qui n'utilisent plus les tableaus noires ou blancs pour véhiculer l'information mais plutôt des tableaux intelligents interconnectés avec l'ordinateur ou la tablette du professeur qui peut même faire place à un échange interactif avec les étudiants qui peuvent partager leurs devoirs sur ce dernier [36].

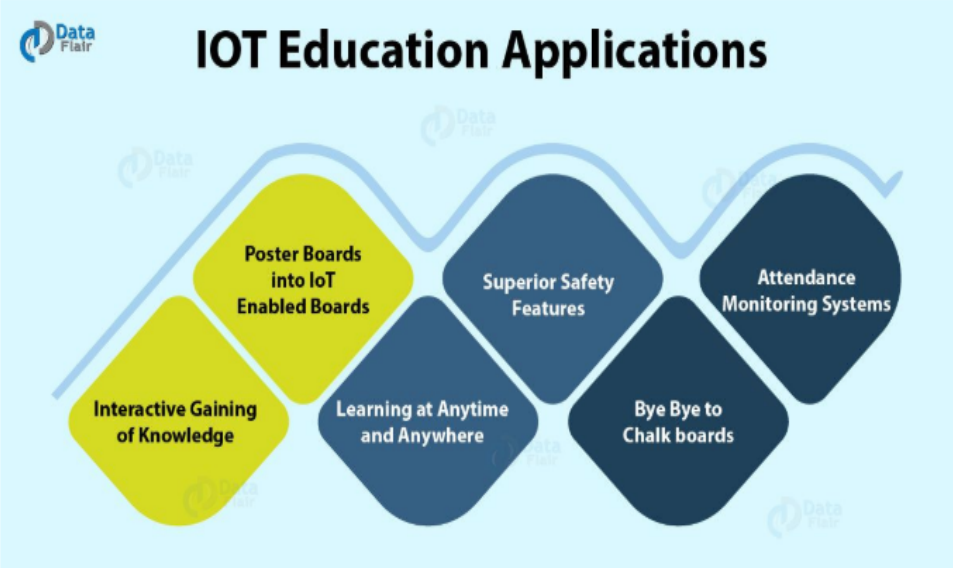

FIGURE 2.11 – Application de l'IoT dans l'éducation.

c. **Applications gouvernementales :**

Les organes directeurs et les ingénieurs peuvent utiliser l'IoT pour analyser les aspects souvent complexes de la planification et de la gestion des villes. L'IoT simplifie l'examen de divers facteurs tels que la croissance démographique, la cartographie, l'approvisionnement en eau, les modèles de transport, l'approvisionnement alimentaire, les services sociaux et l'utilisation des terres. Il rassemble des données détaillées dans ces domaines et produit des informations plus précieuses et plus précises que les analyses actuelles étant donné sa capacité à «vivre» réellement avec les gens d'une ville.

L'IoT contribue également à l'amélioration urbaine en sautant des tests ou des recherches médiocres, et en fournissant des données fonctionnelles sur la façon dont la ville peut être optimisée. Cela conduit à des changements plus rapides et plus significatifs [36].

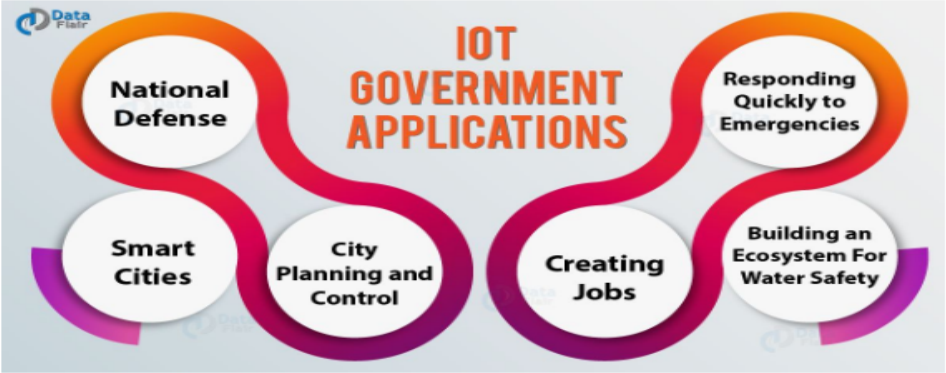

FIGURE 2.12 – Application de l'IoT dans le secteur gouvernementale.

Exemple :*Les poubelles intelligentes de New York indiquent aux éboueurs quand ils doivent être vidés. Ils optimisent le service des ordures en s'assurant que les conducteurs ne font que les arrêts nécessaires, et les conducteurs modifient leur itinéraire pour réduire la consommation de carburant.*

d. **L'industrie :**

L'Internet des Objets Industriel (IIoT) est un secteur en croissance constante et rapide qui représente un grand pourcentage des dépenses sur le marché mondial.

Les industriels et les fabricants de presque tous les secteurs ont une formidable opportunité non seulement de surveiller, mais d'automatiser également de nombreux processus complexes impliqués dans la fabrication. Pendant longtemps, les industries et les usines ont eu des capteurs et des systèmes pour suivre leurs produits, mais l'IoT va plus loin et fournit des facilités aux problèmes les plus infimes.

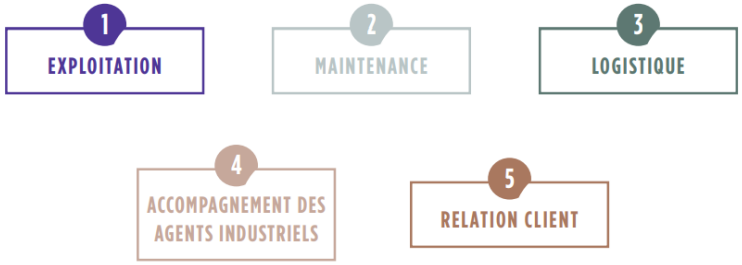

FIGURE 2.13 – Application de l'IoT dans le secteur de l'industrie.

- **Exemple.** *Les capteurs IoT collectent un mélange de données sur les produits et d'autres données synchronisées à travers les étapes d'un cycle de fabrication d'un produit. Ces données contiennent des informations sur la composition des matières premières utilisées dans la fabrication d'un produit, la température et l'environnement de travail, les différents déchets, l'importance du transport, etc... De plus, l'appareil IoT peut également fournir des données sur les sentiments du client pendant qu'il utilise le produit. Toutes ces entrées provenant de différentes sources et via les systèmes IoT peuvent être analysées pour identifier et corriger les problèmes potentiels de qualité.*

e. **La santé :**

L'Internet des Objets a le potentiel de transformer les soins de santé, en changeant en profondeur la manière dont les hôpitaux, les cliniques et d'autres établissements de soins collectent et utilisent des données en réunissant les principales tendances techniques et commerciales liées à la mobilité. Les diverses données collectées à partir de grands ensembles de cas réels augmentent à la fois la précision et la taille des données médicales et améliore la prestation des soins médicaux par l'intégration de technologies plus sophistiquées dans le système de santé.
Concrètement les objets connectés permettent, notamment, un meilleur suivi des maladies chroniques (diabète,asthme, hypertension artérielle, etc.) et une meilleure observance des traitements.

FIGURE 2.14 – Application de l'IoT dans le secteur de medical.

f. **L'agriculture :**

La population mondiale en constante augmentation toucherait environ 9,6 milliards d'ici 2050. Ainsi, pour nourrir cette immense population, l'industrie agricole doit adopter l'IoT. La demande de plus de nourriture doit répondre à des défis tels que l'augmentation du changement climatique, les conditions météorologiques extrêmes et l'impact environnemental résultant des pratiques agricoles intensives.L'agriculture intelligente grâce à l'utilisation des technologies IoT aidera les agriculteurs à réduire les déchets générés et à améliorer la productivité. Cela peut provenir de la quantité d'engrais utilisée et du nombre de déplacements effectués par les véhicules agricoles.

Ainsi, l'agriculture intelligente est fondamentalement un système de haute technologie. C'est l'induction ainsi que l'application des TIC modernes (technologies de l'information et de la communication) dans l'agriculture.

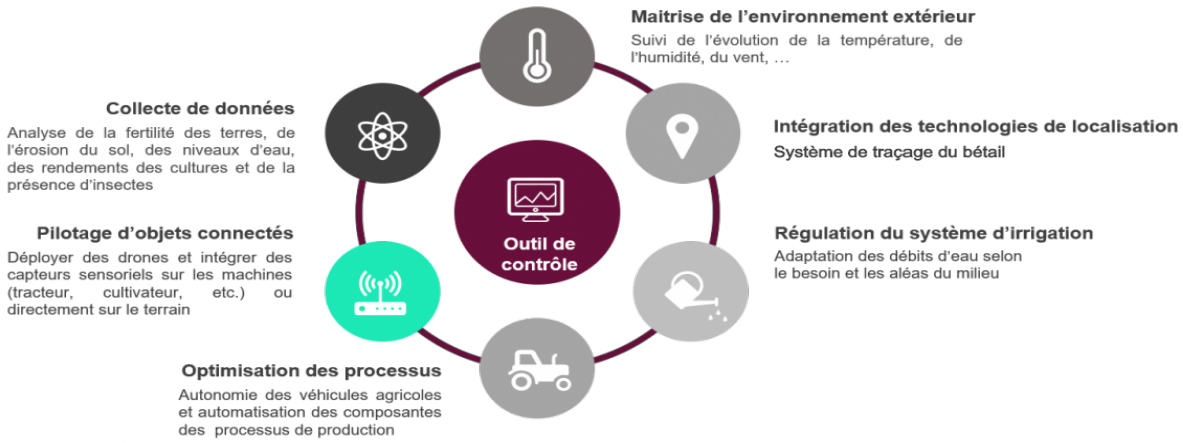

FIGURE 2.15 – Application de l'IoT dans le secteur agricole.

g. **Les smart cities :**

Cette technologie est réellement révolutionnaire. Elle se révèle particulièrement pertinente avec les assistants vocaux comme Amazon Echo et Google Home permettant de contrôler sa maison connectée au son de sa voix. Le confort s'en trouve véritablement accru au quotidien, et cette innovation a rapidement conquis le grand public. En 2018, plus de 47 millions d'américains utilisaient une enceinte connectée.

De même, grâce à l'IoT, les Smart Cities peuvent désormais mieux gérer le trafic routier et la consommation d'électricité. Il résulte des villes plus agréables à vivre, plus respectueuses de l'environnement et plus sûres. Les voitures autonomes, embarquant différents capteurs IoT pour détecter les obstacles et se repérer sur les routes, sont aussi une preuve de l'utilité réelle de cette technologie. Ces véhicules sans pilote vont rendre les routes plus sûres, et nous permettre de gagner un temps fou puisque nous n'aurons plus à conduire.

L'IoT s'est aussi frayé une place dans les entreprises par le biais de l'Internet des objets industriels (IIoT). Les équipements industriels peuvent être équipés de capteurs à des fins de maintenance prédictive et de contrôle centralisé, et c'est une véritable révolution...

h. **L'IoT est présente dans plusieurs autres secteurs :**

- Application énergétique.
- Application de surveillance environnemental.
- Application de transport.
- Application publicitaires.
- Application bâtiments et maisons jusqu'à la smart city (ville intelligente).
- Le secteur bancaire.
- Les activités maritimes.
- L'aérospatiale.

2.3.1 Mode de stockage des données de l'Internet des objets

Lorsqu'un projet IoT et opérationnel, de nombreux appareils produisent beaucoup de données. On a alors besoin d'un moyen efficace, évolutif et abordable pour gérer ces appareils et toutes ces informations afin de les exploiter au mieux. Lorsqu'il est question de stockage, de traitement et d'analyse de données, en particulier de Big Data, le cloud est la solution idéale [37]. Dans cette section nous allons aborder les modes de stockage utilisées à savoir le Cloud computing, le Edge computing, le Fog computing et le Roof computing.

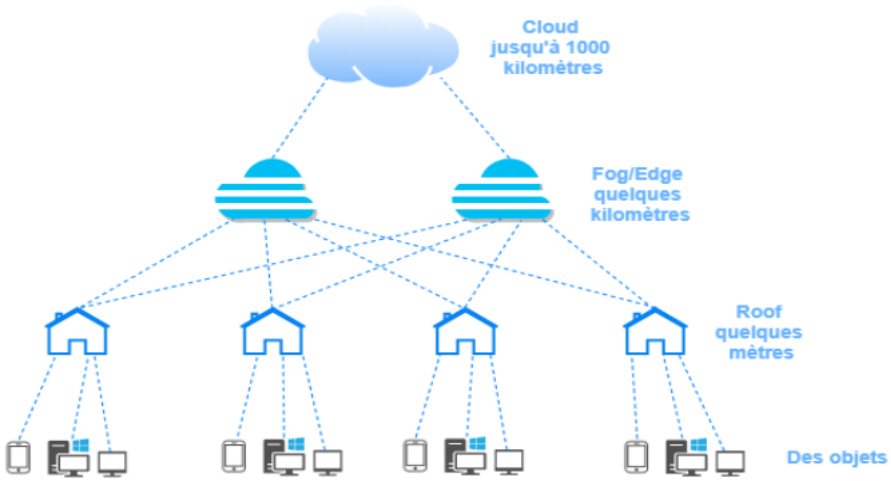

FIGURE 2.16 – Portés du cloud.

2.3.1.1 Le cloud Computing

Comme spécifié au niveau du chapitre précédent Le cloud computing est un modèle qui permet un accès omniprésent, pratique et à la demande à un réseau partagé et à un ensemble de ressources informatiques configurables (comme par exemple : des réseaux, des serveurs, du stockage, des applications et des services).

2.3.1.2 Le Edge Computing

L'edge computing se définit comme une architecture informatique destinée aux environnements IoT, dans laquelle les ressources informatiques, la capacité de stockage et la puissance de calcul sont maintenues au plus près des équipements terminaux et des capteurs qui génèrent les données. Le terme « edge » fait allusion au fait que le traitement des données ne se fait plus dans le Cloud, mais il est décentralisé, en périphérie du réseau. L'edge computing peut ainsi offrir une option que le Cloud n'est pas capable de proposer, à savoir des serveurs capables d'interpréter sans délai les données de masse générées par des usines, des réseaux de distributions ou des systèmes de circulation « intelligents », et de prendre immédiatement les mesures nécessaires en cas d'incidents [38].

En résumé l'edge computing se définit comme une architecture informatique destinée aux environnements IoT, dans laquelle les ressources informatiques, la capacité de stockage et la puissance de calcul sont maintenues au plus près des équipements terminaux et des capteurs qui génèrent les données. Le mot « edge » vient de l'anglais et signifie bord ou périphérie.

Plus précisément le mot « edge » vient de l'anglais et signifie bord ou périphérie. Voici un schéma illustrant le lien entre l'IoT et le Edge Computing :

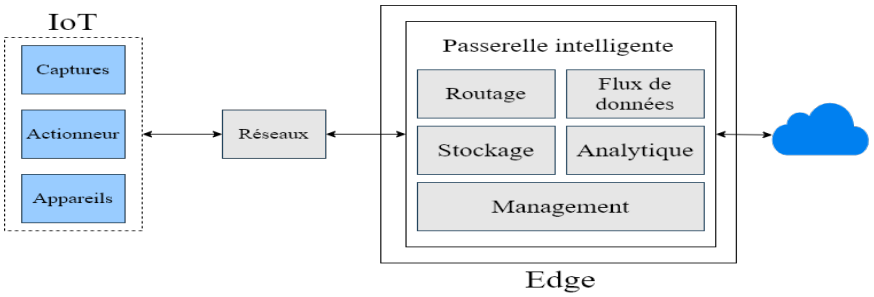

FIGURE 2.17 – l'IoT et le Edge Computing.

2.3.1.3 Le Fog computing

Le fog computing fait référence à une structure informatique décentralisée, où les ressources, y compris les données et les applications, sont placées à des emplacements logiques entre la source de données et le cloud ; il est également connu sous les termes de «brouillard» et de «réseau de brouillard».
L'objectif est d'amener les services analytiques de base à l'edge computing, en améliorant les performances en plaçant les ressources informatiques plus près de l'endroit où elles sont nécessaires, réduisant ainsi la distance que les données doivent être transportées sur le réseau, améliorant ainsi l'efficacité et les performances globales du réseau, il crée une interface supplémentaire que l'on peut situer entre le Edge Computing et le Cloud Computing. Le fog computing peut également être déployé pour des raisons de sécurité, car il a la capacité de segmenter le trafic de bande passante et d'introduire des pare-feu supplémentaires sur un réseau pour une sécurité accrue [39].

2.3.1.4 Le Roof computing

Roof Computing est placé à quelques mètres (ou un saut) des objets et bien en dessous du Fog et de Cloud. Les passerelles IoT et autres appareils qui implémentent le Roof seront en mesure de prendre en charge la connectivité du dernier mètre pour les objets. L'architecture du réseau Roof [Fig. 2.17] reflète l'architecture de réseau fédérée traditionnelle qui est pratiquée depuis l'utilisation généralisée d'Internet. Roof peut être implémenté dans les passerelles IoT, les routeurs domestiques, les téléphones mobiles, les plates-formes informatiques personnelles et d'autres plates-formes intégrées qui procurent les objets pour la connectivité au réseau et au Cloud.

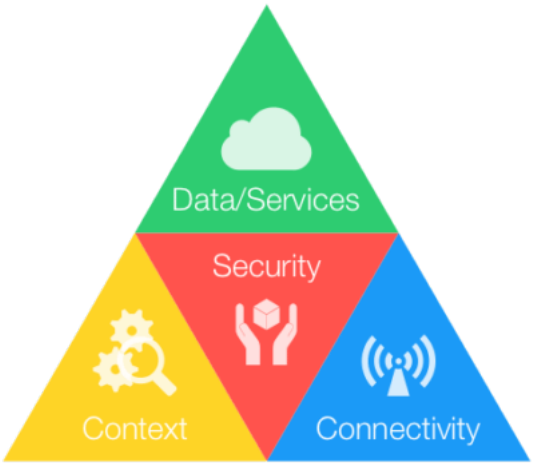

FIGURE 2.18 – Les principales fonctionnalités du Roof Computing.

Le Roof propose quatre fonctionnalités principales à savoir :

- Connectivité interopérable.
- Mise en contexte et décisions en temps réel.
- Gestion des informations et connectivité efficace au Cloud et aux fournisseurs de services
- Sécurité et confidentialité dès la conception.

Conclusion

Il n'est pas exagéré de dire que l'IoT annonce une nouvelle ère économique pour l'ensemble de la planète. Les espoirs portés par l'IoT ne portent pas simplement sur l'amélioration des processus et des modèles économiques existants, ils présentent en outre une dimension transformationnelle dans leur portée. L'économie de l'IoT va révolutionner la manière dont les entreprises produisent, fonctionnent et amènent des résultats. Et ce bouleversement se produit plus rapidement que toute autre révolution industrielle précédente.

Parallèlement, l'IoT pose des défis conséquents dans l'ensemble des secteurs et industries en apportant des solutions à des problèmes préoccupant les entreprises depuis des décennies, voire des siècles. Il va également poser des dilemmes entièrement nouveaux, tant sur les plans procédural qu'éthique. Les préoccupations en matière de respect de la vie privée, de cybersécurité, de propriété et de responsabilité quant aux produits gagneront en ampleur aussi rapidement que les opportunités de l'IoT se présenteront. Si les entreprises ont intérêt à commencer à mettre en place la technologie de l'IoT pour espérer survivre à long terme, elles doivent également mettre en oeuvre des stratégies tenant compte des nombreux risques associés à l'IoT.

Cependant, rien de tout cela n'est possible sans une base de données fiable capable de gérer les énormes quantités de données générées par les appareils IoT. Dans le prochain chapitre nous allons voir ou stocker ces quantités immenses de données produites par l'IoT, ainsi que les différentes technologies participant à se stockage.

Chapitre 3

Base de données NoSQL

Introduction

Plusieurs modèles de stockage de l'information sont explorés à la naissance de l'informatique. Mais c'est le modèle relationnel qui a eu le dessus en 1970 grâce à un modèle théorique puissant et simple qui permet d'assurer au mieux l'intégrité des données . Les bases de données NoSQL, signifiant « Not only SQL » ont commencé à émerger depuis 2009, notamment par les géants du WEB pour répondre aux nouveaux besoins en performance lors de traitement de gros volumes de données ,Ce dernier ne vient pas remplacer les bases de données relationnelles mais proposer une alternative ou compléter les fonctionnalités des SGBDs relationnels pour donner des solutions plus intéressantes dans certains contextes. En effet, le langage SQL a atteint ses limites dans certaines situations que nous verrons plus en détail dans ce chapitre. Tout au long de ce chapitre nous allons présenter d'une part les SGBDR, leurs contraintes, leurs points forts ainsi que leurs limites qui ont été le point de départ du mouvement NoSQL que nous allons aussi détailler. D'autre part, la comparaison entre ces deux notions s'impose afin de mieux comprendre l'apport du Nosql par rapport au SQL. Enfin, nous achevons ce chapitre en citant quelques exemples des bases de données NoSQL les plus utilisées avec leurs caractéristiques et avantages.

3.1 Systèmes de gestion de base de données relationnels

Les bases de données ont pris aujourd'hui une place essentielle dans l'informatique, plus particulièrement en gestion. Au cours des trente dernières années, des concepts, méthodes et algorithmes ont été développés pour gérer des données sur mémoires secondaires ; Ils constituent l'essentiel de la discipline « Bases de Données ». Cette discipline est utilisée dans de nombreuses applications. Il existe un grand nombre de Systèmes de Gestion de Bases de données (SGBD) qui permettent de gérer efficacement de grandes bases de données. Les bases de données constituent donc une discipline s'appuyant sur une théorie solide et offrant de nombreux débouchés pratiques. Les SGBD ont bientôt quarante ans d'histoire. Les années 60 ont connu un premier développement des bases de données sous forme de fichiers reliés par des pointeurs. Les fichiers sont composés d'articles stockés les uns à la suite des autres et accessibles par des valeurs de données appelées clés. Les systèmes IDS.I et IMS.I développés respectivement à Honeywell et à IBM vers 1965 pour les programmes de conquête spatiale, notamment pour le programme APOLLO qui a permis d'envoyer un homme sur la lune, sont les précurseurs des SGBD modernes. Ils permettent de constituer des chaînes d'articles entre fichiers et de parcourir ces chaînes.

Un système de gestion de base de données relationnels est un type particulier des

SGBD base sur le modèle relationnel introduit par Edgar Frank Codd Il permet en particulier de synthétiser n'importe quel lot d'informations en exploitant le contenu des différentes tables de la base de données par application des opérations de l'algèbre relationnelle telles que la jointure, la sélection et la projection.

3.1.1 Les bases de données relationnelle

Une base de données relationnelle est un type de base de données où les données sont liées à d'autres informations au sein des bases de données. Les bases de données relationnelles sont composées d'un ensemble de tables qui peuvent être accessibles et reconstruites de différentes manières, sans qu'il soit nécessaire de réarranger ces tables de quelque façon que ce soit. Le langage de requête structuré (SQL) est l'interface standard pour une base de données relationnelle. Les instructions SQL sont utilisées à la fois pour interroger de façon interactive les données contenues dans la base de données relationnelle et pour collecter les données dans le cadre de rapports.

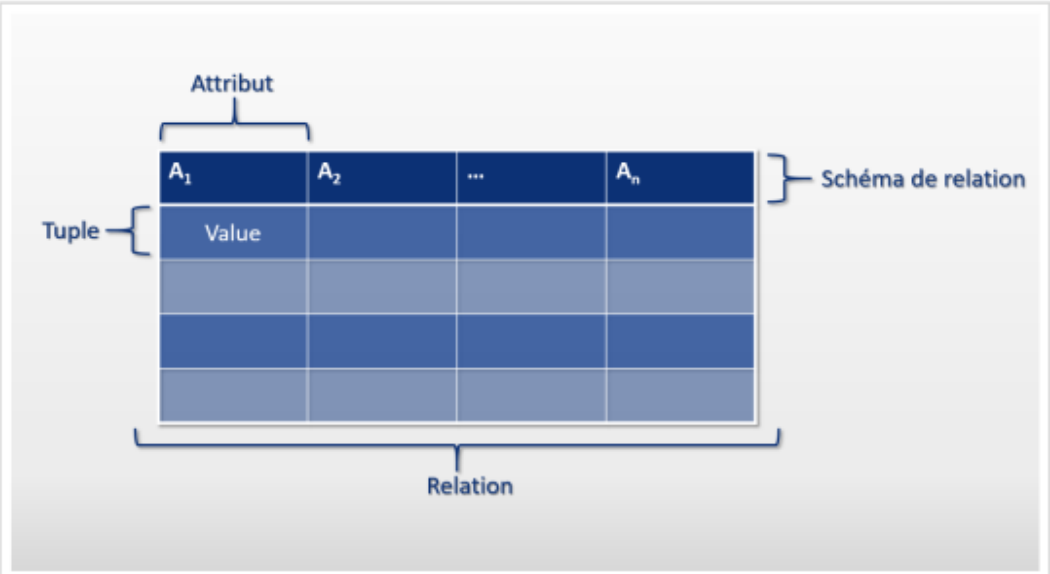

FIGURE 3.1 – Elements d'une table d'une BDDR.

3.1.2 Les règles CODD

Les douze règles de Codd (numérotées de 0 à 12, donc en fait treize...) formalisent la notion de SGBD relationnel. C'est le docteur Codd, père fondateur de l'algèbre relationnelle qui grave ainsi dans le marbre les 13 commandements de la Loi pour tout SGBD se prétendant relationnel. Ces règles sont basées sur ses travaux originaux effectués à partir de 1970, et ont été publiées dans deux articles de vulgarisation du magazine Computerworld (octobre 1985) : Is your DBMS really relational ? et Does your DBMS run by the rules ? Ces règles constituent les fondements des bases de données relationnelles et permettent de connaître le niveau relationnel du produit d'un éditeur. Ne pas s'y contraindre implique plus d'inconvénients que d'avantages. En un sens elles gravent dans le marbre "la Loi" que tout bon SGBD relationnel devrait suivre [40] :

— **Règle 0 :** Toutes les fonctionnalités du SGBDR doivent être disponibles à travers le modèle relationnel et le langage d'interrogation.

— **Unicité :** Toutes les informations de la base de données sont représentées d'une et d'une seule façon, c'est-à-dire par la valeur dans le champ de la

colonne de table.

— **Garantie d'accès :** Toutes les données doivent être clairement accessibles. La règle est essentiellement un ajustement des conditions de base de la clé primaire. Cela signifie qu'en spécifiant le nom de la table contenant, le nom de la colonne contenant et la valeur principale de la ligne contenante, chaque valeur scalaire dans la base de données doit être logiquement accessible.

— **Traitement des valeurs nulles :** Le système de gestion de la base de données doit permettre à chaque champ de rester nul (ou vide). Plus précisément, il doit prendre en charge la représentation systématique des « informations manquantes et des informations inappropriées » différentes de toutes les valeurs conventionnelles (par exemple, pour les valeurs numériques, « différentes de zéro ou de tout autre nombre »), indépendamment de Quel est le type de données. Cela implique également que ces représentations doivent être gérées par le système de gestion de base de données de manière systématique.

— **Catalogue lui-même relationnel :** Le système doit prendre en charge un catalogue en ligne intégré et associé auquel les utilisateurs autorisés peuvent accéder via des langages de requête réguliers. Par conséquent, l'utilisateur doit pouvoir accéder à la structure de la base de données (répertoire) en utilisant le même langage de requête que les données de la base de données.

— **Sous-langage de données :** Le SGBDR doit implémenter un langage relationnel qui supporte la manipulation des données et Métadonnées, définir les contraintes de sécurité et gérer les transactions

— **Mise à jour des vues :** En théorie, toutes les vues pouvant être mises à jour doivent pouvoir être mises à jour par le système.

— **Insertion, mise à jour, et effacement de haut niveau :** Le système doit prendre en charge les opérations par lots pour l'insertion, la mise à jour et la suppression. Cela signifie que les données peuvent être extraites d'une base de données relationnelle et que l'ensemble de données se compose de données provenant de plusieurs tuples et / ou de plusieurs tables. Cette règle stipule que les tuples par lots dans plusieurs tables et les tuples uniques dans une seule table doivent prendre en charge les opérations d'insertion, de mise à jour et de suppression.

— **Indépendance physique :** Les modifications au niveau physique (comment les données sont stockées, qu'il s'agisse de lignes ou de listes liées, etc.) ne nécessitent pas de modifications des applications basées sur la structure.

— **Indépendance logique :** Les modifications logiques (tableaux, colonnes, lignes, etc.) n'ont pas besoin d'être modifiées dans les applications basées sur la structure. L'indépendance des données logiques est plus difficile à atteindre que l'indépendance des données physiques.

— **Indépendance d'intégrité :** Les contraintes d'intégrité doivent être indiquées séparément de l'application et stockées dans le répertoire. Au fil du temps, il devrait être possible de modifier ces contraintes sans affecter inutilement les applications existantes.

— **Indépendance de distribution :** Attribuer ou partitionner Les données ne devraient avoir aucun effet sur le programme client.

— **Règle de non-subversion :** Si le système fournit une interface de bas niveau, l'interface ne doit pas permettre le contournement du système (par exemple, la sécurité des relations ou les contraintes d'intégrité). L'ensemble de ces règles indique la voie à suivre pour les systèmes de gestion de bases de données relationnelles. Elles ne sont jamais totalement implémentées, à cause des difficultés techniques que cela représente.

L'ensemble de ces règles indique la voie à suivre pour les systèmes de gestion de bases de données relationnelles. Elles ne sont jamais totalement implémentées, à cause des difficultés techniques que cela représente.

3.1.3 Les contraintes des SGBDs relationnels

La plupart des moteurs de SGBD relationnels sont basés sur le concept de transaction Il s'agit d'une unité d'action qui les oblige à se conformer aux contraintes ACID, acronyme : Atomicité Consistance Isolation Durabilité et qu'on nomme donc acidifiante de la transaction :[41]

- **Atomicité :** Tout ou rien. Soit l'opération se fait en entier, soit elle ne se fait pas du tout. La notion d'atomicité sous-entend la possibilité de défaire une opération avortée qui consiste à interdire l'exécution partielle de la transaction. Quand une transaction en cours d'exécution est interrompue, le SGBD doit annuler toutes les opérations déjà exécutées.[42]

  Exemple : *sur 5000 lignes devant être modifiées au sein d'une même transaction, si la modification d'une seule échoue, alors la transaction entière doit être annulée. C'est primordial, car chaque ligne modifiée peut dépendre du contexte de modification d'une autre, et toute rupture de ce contexte pourrait engendrer une incohérence des données de la base.*

- **Cohérence :** L'opération doit assurer que la base de données sera dans un état valide après l'opération.[42]

- **Isolation :** L'opération doit se faire en toute autonomie sans dépendance à une autre opération Cohérence.[43]

  Exemple *si une requête est lancée alors qu'une transaction est en cours, le résultat de celle-ci ne peut montrer que l'état original ou final d'une donnée, mais pas l'état intermédiaire. De fait, les transactions doivent s'enchaîner les unes à la suite des autres, et non de manière concurrentielle.*
- **Durabilité :** Elle garantit que les transactions réussies survivront de façon permanente et ne seront pas affectées par d'éventuelles pannes ou problèmes techniques. Les changements apportés aux données doivent être permanents. Plus précisément, ce sont les effets logiques des données modifiées sur les futures transactions qui doivent être permanents [44].

3.1.4 Les forces des SGBDs relationnels

Le succès des SGBDs relationnels est dû essentiellement aux 12 règles de CODD et le modèle relationnel qui a permis de développer une vision de ce qu'est ou doit être une base de données. Parmi ces nombreux avantages nous citons [45] :

1. Cohérence transactionnelle forte : Assurer la gestion de la concurrence, l'isolation entre les utilisateurs, la reprise sur panne.
2. Optimisation très fine des requêtes, index permettant un accès rapide aux données.
3. Logiciels mûrs, stables, efficaces, riches en fonctionnalités et en interfaces.
4. Contraintes d'intégrité permettant d'assurer des invariants sur les données.
5. Séparation logique et physique : justifié par une forte indépendance entre :
   — Modèle de données et structures de stockage.
   — Requêtes déclaratives et exécution.
6. Capacité de gestion de requêtes complexes.

3.1.5 Limite des base de données relationnels

Bien que le SGBD relationnel soit utilisé avec succès depuis plus de 20 ans. Cependant, il existe de nombreuses restrictions importantes. Les premiers acteurs à atteindre ces limites sont les géants du web. Le constat est simple : les SGBD relationnels ne sont plus la solution nécessaire pour un environnement distribué requis pour l'énorme quantité de données n'est pas tant le langage SQL en lui-même qui est inadapté mais les grands principes qui l'ont établi (Le modèle de relation et de transaction que nous avons vu en haut). En effet, en Big Data, nous devons non seulement gérer de grandes quantités de données, mais nous espérons également obtenir des résultats rapides. Mais Les SGBDs relationnels traditionnels atteignent ici leurs limites à savoir

— Problème d'application des propriété ACID en milieu distribué : Une base de données relationnelle est construite en respectant les propriétés ACID, ses propriétés bien que nécessaires à la logique du relationnel nuisent fortement aux performances et en particulier la propriété de cohérence En effet, la cohérence est très difficile à mettre en place dans le cadre de plusieurs serveurs (environnement distribué), car pour que celle-ci soit respectée tous les serveurs doivent être des miroirs les uns des autres, de ce fait deux problèmes apparaissent :
   — Le coût en stockage est énorme car chaque donnée est présente sur chaque serveur.
   — Le coût d'insertion/modification/suppression est très grand, car on ne peut valider une transaction que si on est certain qu'elle a été effectuée sur tous les serveurs et le système [46].
— Problème de requête non optimale dû à l'utilisation des jointures : Imaginons une table contenant toutes les personnes ayant un compte sur Instagram, soit 1 milliards d'utilisateurs actifs par mois, les données dans une base de données relationnelle classique sont stockées par lignes, ainsi si on effectue une requête pour extraire tous les amis d'un utilisateur donné, il faudra effectuer la jointure entre la table des usagers et celle des amitiés (chaque usager ayant au moins un ami) puis parcourir le produit cartésien de ces

deux tables. De ce fait, on perd énormément en performances en raison du temps consommé pour stocker et parcourir une telle quantité de données [46].

— Problème de gestion des objets hétérogène : Au fur et à mesure du temps, les structures de données manipulées par les systèmes sont devenues de plus en plus complexes en contrepartie les moteurs de stockage évoluant peu. Le principal point faible des modèles relationnels est l'absence de gestion d'objets hétérogènes ainsi que le besoin de déclarer au préalable l'ensemble des champs représentant un objet [46].

— Surcharge sémantique : Le modèle relationnel s'appuie sur un seul concept (la relation) pour modéliser à la fois les entités et les associations entre ces entités. Il existe donc un décalage entre la réalité et sa représentation abstraite [47].

— Scalabilité limitée : On peut distinguer deux types de scalabilité que les SGBDs relationnels ne puissent gérer :

  — La scalabilité des traitements : représente la capacité de distribution des traitements sur un nombre de machines important afin d'être en mesure d'absorber des charges très importantes.
  — La scalabilité des données : représente la capacité de répartition des données entre un nombre important de machines pour être en mesure de stocker de très grands volumes de données.

— Contraintes d'intégrité : Les SGBDs relationnels sont limités à des contrôles de cohérence simples. Il est donc difficile de modéliser des contraintes réelles correspondant aux données d'une entreprise. Ce problème n'est que partiellement résolu avec SQL-2 qui permet d'exprimer des contraintes dans la partie langage de définition [47].

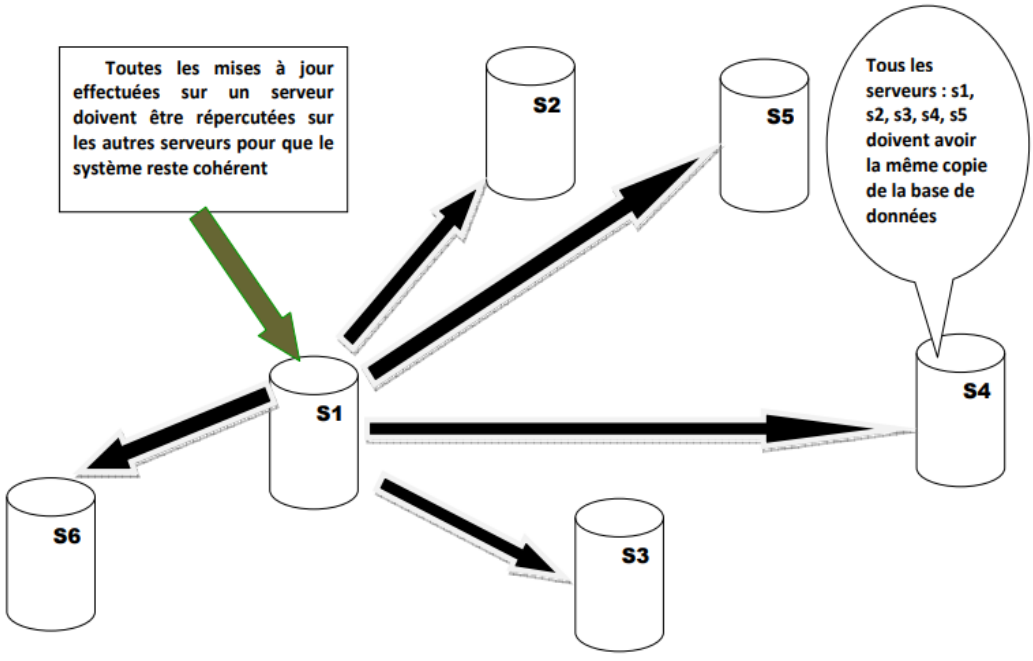

FIGURE 3.2 – Limites liées aux propriétés ACID.

3.1.6 Exemple des base de données relationnels

Nous présentons ci-dessous les 4 systèmes de gestion de bases de données relationnelles (SGBDR) les plus utilisé en 2020 :[48]

- **Oracle Database :** Oracle Database est un système de gestion de base de données relationnelle (SGBDR) qui depuis l'introduction du support du modèle objet dans sa version 8 peut être aussi qualifié de système de gestion de base de données relationnel-objet (SGBDRO). Fourni par Oracle Corporation, il a été développé par Larry Ellison, accompagné entre autres, de Bob Miner et Ed Oates.
- **MySQL :** C'est un système de gestion de bases de données relationnelles (SGBDR). Il est distribué sous une double licence GPL et propriétaire. Il fait partie des logiciels de gestion de base de données les plus utilisés au monde, autant par le grand public (applications web principalement) que par des professionnels.
- **Microsoft SQL Server :** Microsoft SQL Server est un système de gestion de base de données (SGBD) en langage SQL incorporant entre autres un SGBDR (SGBD relationnel) développé et commercialisé par la société Microsoft. Il fonctionne sous les OS Windows et Linux (depuis mars 2016), mais il est possible de le lancer sur Mac OS via Docker, car il en existe une version en téléchargement sur le site de Microsoft1.
- **PostgreSQL :** PostgreSQL est un système de gestion de base de données relationnelle et objet (SGBDRO). C'est un outil libre disponible selon les termes d'une licence de type BSD. Comme les projets libres Apache et Linux, PostgreSQL n'est pas contrôlé par une seule entreprise, mais est fondé sur une communauté mondiale de développeurs et d'entreprises.

3.2 Les bases de données NoSQL

Depuis les années 2000, avec l'explosion des applications web et une croissance de leurs usages, les grands acteurs du Web ont dû faire face à de nouveaux enjeux concernant leurs infrastructures informatiques. Face à cette croissance, les solutions de base de données relationnelles Open Source ont montré rapidement leurs limites, en particulier dans le domaine du web.

3.2.1 Définition :

Définition 1 : NoSQL est un mouvement très récent, qui concerne les bases de données. L'idée du mouvement est simple : proposer des alternatives aux bases de données relationnelles pour coller aux nouvelles tendances et architectures du moment, notamment le Cloud Computing. Les axes principaux du NoSQL sont une haute disponibilité et un partitionnement horizontal des données, au détriment de la cohérence. Alors que les bases de données relationnelles actuelles sont basées sur les propriétés ACID (Atomicité, Consistance ou Cohérence, Isolation et Durabilité).

Définition 2 : Le NoSQL, désignant Not Only SQL , représente une catégorie de bases de données apparue au courant de l'année 2009 qui se différencie du modèle relationnel que l'on retrouve dans les systèmes de bases de données connues tels que MS SQL Server, Oracle ou PostgreSQL. Cette catégorie de produits fait le compromis d'abandonner certaines fonctionnalités classiques des SGBD relationnels au portfie de la simplicité, la performance et une forte scalabilité. La scalabilité est la capacité d'un système à répondre à une demande toujours grandissante de la part des utilisateurs en termes de requêtes. Il s'agit d'une capacité de montée en charge.

Remarque : *NoSQL est constitué de No et SQL. Le No est un acronyme qui signifie Not only, ce qui veut dire en français ce n'est pas seulement du SQL. Cela signifie donc qu'il y a plus que les bases de données relationnelles.*

3.2.2 Le théorème CAP

CAP est un acronyme qui signifie Consistency, Availability and Partition Tolerance , dans la langue français cela donne Cohérence, Disponibilité et Résistance au morcellement . Ce théorème est aussi connu sous le nom de théorème de Brewer , du nom d' Eric Brewer qui l'a inventé en 2000 Ce théorème explique qu'un système d'information distribué ne peut satisfaire à la fois plus de 2 contraintes suivantes :

1. **Consistency (Cohérence) :** Une donnée n'a qu'un seul état visible quel que soit le nombre de réplicas.

2. **Availability (Disponibilité) :** Tant que le système tourne (distribué ou non), la donnée doit être disponible.

3. **Partition Tolerance (Tolérance au partitionnement) :** Quel que soit le nombre de serveurs, toute requête doit fournir un résultat correct.

Comme nous l'avons mentionner ci-dessus les trois propriétés a savoir la cohérence, la disponibilité et la tolérance au partitionnement ne peuvent être obtenues simultanément c'est pour cela que des couples assurant deux propriétés se sont formés :

— **Soucieux de la cohérence et de la disponibilité (CA)**
Consistency Availability, soit le couple AC : cluster de site unique, donc tous les nœuds sont toujours en contact. Lorsqu'une partition se produit, le système s'arrête. Ce cas est possible que dans le cadre de bases de données transactionnelles telles que les SGBDR.

— **Soucieux de la tolérance de partition et la cohérence (PC)**
Consistency Partition Tolerance, soit le couple PC : Certaines données peuvent ne pas être accessibles, mais les données restent cohérentes et le système est tolérant au panne.

— **Soucieux de la disponibilité et la tolérance de partition (AP)**
Availability Partition Tolerance, soit le couple AP : Le système est toujours disponible même sous partitionnement, mais certaines des données retournées peuvent être inexactes.

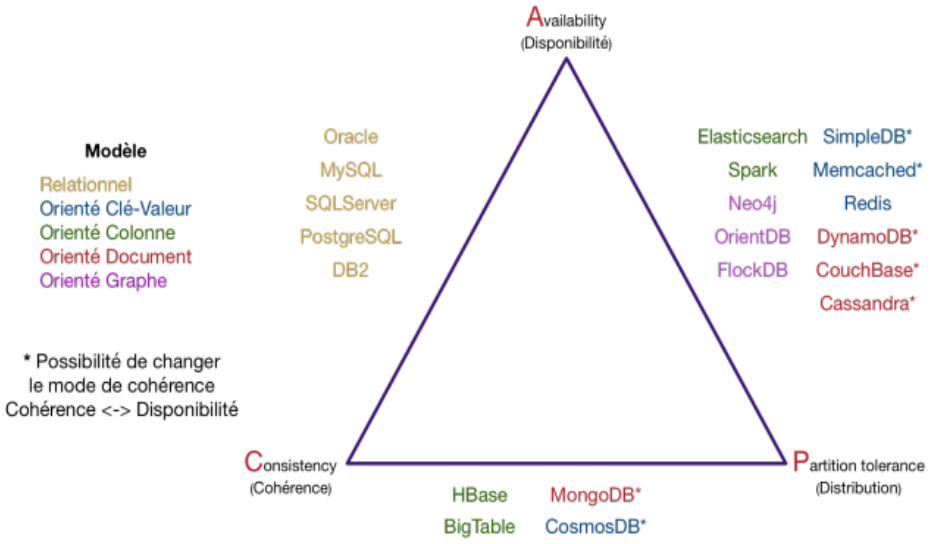

FIGURE 3.3 – Le théorème CAP.

3.2.3 Les types des base NoSQL

Le modèle de données de la base de données traditionnelle est principalement relationnel, spécifiquement pour prendre en charge les opérations de classe associées et les transactions ACID, mais dans les champs de la base de données NoSQL, le modèle de données traditionnel est le suivant :

3.2.3.1 Les bases de données orientées clé-valeur

Les données sont représentées par un couple clé valeur, ce type est le plus simple et est très adapté aux caches ou aux accès rapides aux informations. Son principe de base est le stockage d'une valeur associée à une clé unique. Celle-ci représente la seule manière de solliciter l'objet. La valeur pouvant être une simple chaîne de caractères ou un objet Elles ont de très bonnes performances car les lectures et écritures sont réduites à un accès disque simple. Il n'y a pas de schéma et les données ne sont pas liées entre elles. On les utilise quand le modèle de données est simple.[49]

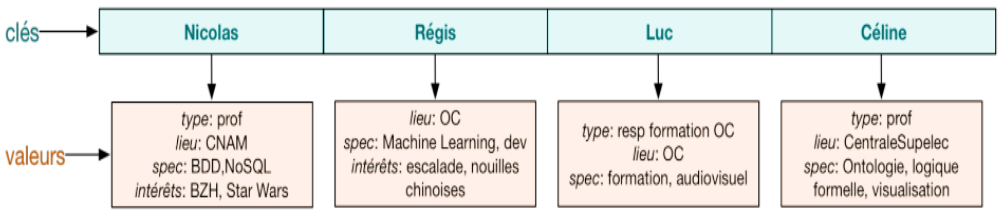

FIGURE 3.4 – Base de donnée orientée clé-valeu.

Ces bases permettent 4 opérations La création en effet seules opérations de type CRUD peuvent être utilisées :

- Create (key,value). Création d'un nouveau couple (clé, valeur).
- Read (key). La lecture : lire un objet en sachant sa clé
- Update (key,value). La modification : modifier et mettre à jour l'objet associé à une clé

- Delete (key). La suppression : supprimer un objet sachant clé.

Exemple de base de données clé-valeur

- Redis (VMWare) : Vodafone, Trip Advisor, Nokia, Samsung, Docker.
- Memcached (Danga) : LiveJournal, Wikipédia, Flickr, Wordpress.
- Azure Cosmos DB (Microsoft) : Real Madrid, Orange tribes, MSN, LG, Schneider Electric.
- SimpleDB (Amazon).

Avantage

— Leur simplicité.
— scalabilité.
— disponibilité.
— Très bonnes performances car les lectures et écritures sont réduites à un accès disque.

Inconvénient

— Pas de requêtes sur le contenu des objets stockés.
— Pas de relations entre les objets.

3.2.3.2 Les bases de données orientées colonnes

Dans ce type les BDD ne sont plus stockées en lignes mais en colonnes. Le nombre de colonnes est dynamique. On pourrait se dire que ce type ressemble un peu à la représentation des tables dans les bdd relationnelles car on y retrouve le principe de table qui contient des lignes et aussi des colonnes, mais elles ont deux principales différences : Des colonnes dynamiques. Dans la même table deux éléments peuvent ne pas avoir le même nombre de colonnes car les valeurs nulles ne sont pas stockées (contrairement au SGBDR relationnel). Cette propriété permet un gain d'espace de stockage et amélioration des performances de traitement L'historisation des données se fait à la valeur et non pas à la ligne comme dans les SGBDR cela empêche le stockage d'informations en doublon et de ce fait allège considérablement la base de données et les temps de calcul [49].

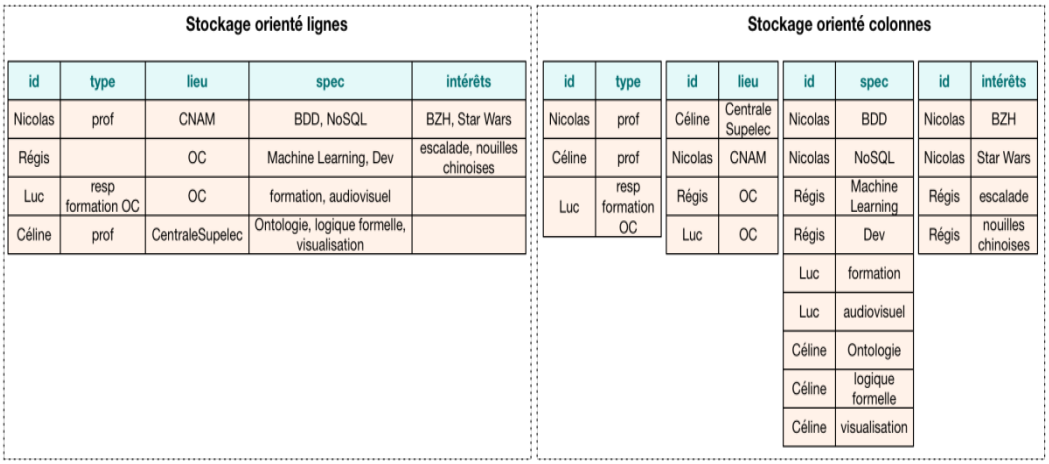

FIGURE 3.5 – Base de donnée orientée colonne.

Exemple de base de données orienté colonne :

— Hbase (Apache, Hadoop).
— Cassandra (Facebook , Apache)
— Big Table (Google).
— Hypertable.
— Apache Parquet.

Avantage

— Un temps de traitement réduit .
— Un gain d'espace de stockage grâce au stockage des valeurs nulles.

Inconvenant

— Pas adaptée aux données interconnectées.
— Pas adaptée pour les données non-structurées.

3.2.3.3 Les bases de données orientées documents

C'est une évolution de la base clé valeur, elle repose également sur l'association d'un couple [clé, valeur], et la valeur, dans ce cas, est un document. Ce document a une structure arborescente : Un document se compose d'un ensemble de couple clés/ valeurs. Les documents sont souvent de type JSON ou bien XML. Ces bases sont particulièrement adaptées au stockage de données semi structurées. Elles conviennent aux applications web pour leur simplicité d'utilisation et de déploiement [49].

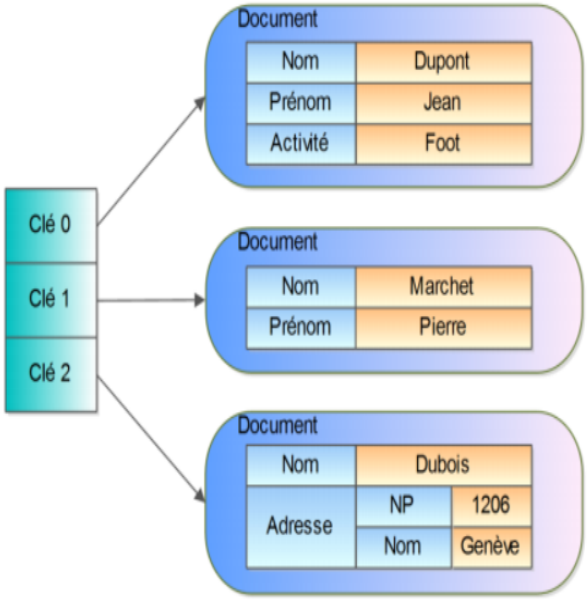

FIGURE 3.6 – Base de donnée orientée document.

Exemple de base de données orienté document

— MongoDB (MongoDB).
— RavenDB par Hibernating Rhinos.
— CouchBase (Apache, Hadoop).
— RethinkDB.
— DynamoDB (Amazon).

,

Avantage

- Les documents sont structurés mais aucune définition de structure préalable n'est nécessaire.
- On peut requêter et manipuler ces documents, et notamment récupérer, grâce une clé unique, l'ensemble des informations structurées de manière hiérarchique.

Inconvénient

- Pas de relations entre les documents Non adapter pour les données non-structurées.

3.2.3.4 Les bases de données orientées graphe

Ce sont des bases de données qui utilisent la théorie des graphes pour représenter et stocker les données , avec des nœuds et des arcs. Elles s'appuient sur les notions de : Noeuds qui ont chacun leur propre structure, Relations entre les noeuds ,Propriétés (de noeuds ou de relations). Le type orienté graphe est particulièrement bien adapté au traitement de certaines données comme celle des réseaux sociaux, les données géographiques, et de toutes les données fortement connectées de manière générale, par exemple la représentation des relations entre les abonnés sur Twitter ou Instagram.

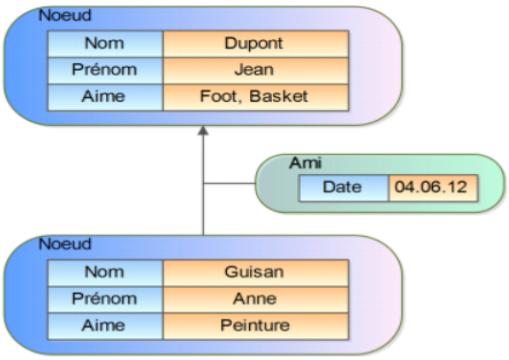

FIGURE 3.7 – Base de donnée orientée graphe.

Exemple de bases de données orienté graphes :

- Neo4j
- OrientDB (Apache)
- FlockDB (Twitter)
- inniteGraphe
- Titan
- ArangoDB

Avantage

- Adaptées aux objets complexes organisés en réseaux, aux données présentant des dépendances fortes.
- Permettent d'appliquer les algorithmes de théorie des graphes et la mise en place de visualisation de graphes nativement.
- Beaucoup plus rapides que les autres systèmes de stockage pour manipuler les données fortement connectées.

Inconvénient

— Non adaptées pour tous les autres contextes que celui des données fortement connectées.

3.2.3.5 Autres base de données

L'approche non relationnelle ne se limite pas aux base de données NOSQL, Il existe plusieurs autres suivant une approche non relationnelle, elles ne sont pas considérées comme des bases de données NoSQL de base, mais plutôt comme des bases de données NoSQL logicielles :

Bases de données d'objets : Les modèles de bases de données orientées objet regroupent des paquets de données très proches : un ensemble de données est regroupé avec tous ses attributs pour former un objet. Ainsi, toutes les informations sont immédiatement disponibles. Au lieu de tout répartir dans différentes tables, les données sont interrogeables par ensemble.

Bases de données XML : Une base de données XML Native (NXD en anglais) est une base de données qui s'appuie sur le modèle de données fourni par XML. Elle utilise typiquement des langages de requête XML comme XPath ou XQuery.

L'indexation dans une base de données XML nécessite d'indexer non seulement le contenu des éléments mais aussi la structure, les relations entre éléments pour que des requêtes XPath comme /foo/bar utilisent l'index.

3.2.4 Avantage du NoSQL

L'intérêt des systèmes de stockage NoSQL réside surtout dans les choix d'architecture logicielle qui ont été pris lors de leurs conceptions. Parmi les raisons principales qui ont mené à la création de ces systèmes.Le NoSQL possède un ensemble de fonctionnalités, notamment :

— La plupart des systèmes NoSQL sont open source.
— Les SGBD NoSQL évoluent horizontalement. Ainsi, pour gérer une grande quantité de données, il faut ajouter un serveur et non en augmenter sa capacité ou ses performances. La flexibilité et l'évolutivité horizontale offrent aux bases de données NoSQL une grande capacité d'adaptation au changement sans affecter le système ou les performances.
— Rapidité : flexibilité dans la gestion des données et rapidité , car ils n'utilisent pas de schéma de base avec les contraintes sur les champs.
— Les systèmes de gestion bases de données NoSQL peuvent s'adapter à tous les types de données car ils supportent les données structurées comme les non structurées.
— La réplication des bases de données NoSQL est pensée autrement que pour les bases de données classiques. Les réplications sont effectuées automatiquement dans les clusters (ensemble de plusieurs serveurs) et dans les centres de données
— Peut fonctionner sur des appareils de faible spécification.
— peut s'utiliser avec différents types de données.
— Permet de manipuler de grand volume de données
— Schéma flexible.

3.2.5 Inconvénients du NoSQL

Il faut néanmoins être conscient que les avantages apportés par ces systèmes ne sont pas sans contreparties, aucun système n'étant parfait. Les principaux inconvénients apportés par les choix de design des NoSQL sont les suivants :

— Fonctionnalités réduites : Les bases de données NoSQL sont principalement conçues pour stocker de la meilleure des façons qui soit des ensembles donnée mais en contrepartie nous avons très peu de fonctionnalités aussi bien faite que celle ci notamment le langage permettant d'effectuer des requêtes vers le système NoSQL est beaucoup moins riche.

— Normalisation et Open Source : Étrangement, le critère d'open source pour les bases de données NoSQL est à la fois sa plus grande force et sa plus grande faiblesse. La raison est qu'il n'existe pas encore de normalisation fiable pour NoSQL.

— Performances et évolutivité au détriment de la cohérence : En raison de la façon dont les données sont gérées et stockées dans ces bases, la cohérence des données pourrait bien être une préoccupation. Comme déjà vu précédemment, les bases de données NoSQL font l'impasse sur les propriétés dites ACID afin de mieux répondre aux besoins de performances et d'évolutivité. La cohérence des données est donc un facteur moins important. Selon le besoin, ces caractéristiques peuvent être un sérieux atout comme un paramètre irrecevable. S'il faut faire face à de très grosses montées en charge de très gros volumes de données, NoSQL est le système adéquat. Si on traite des données sensibles (transactions bancaires ou autres), la cohérence des données est indispensable. Et comme les systèmes NoSQL ne respectent pas l'ensemble des propriétés ACID, comme le font les systèmes relationnels classiques, cela se traduit en pratique par un effort supplémentaire du développeur dans certains cas pour s'assurer de la cohérence des données .

— Manque général de maturité : Bien que les bases de données NoSQL soient présentes depuis bon moment, leurs technologies sont encore immatures par rapport à celles des bases relationnelles. Cela se traduit également par un manque d'administrateurs et de développeurs ayant les compétences dans ce système. Les bases de données ont beau être « administrator-friendly » (simples à administrer), cela n'a pas de sens si les administrateurs en question ne savent pas comment les utiliser. A l'heure actuelle, les bases de données relationnelles sont beaucoup mieux implémentées dans les entreprises. Elles disposent d'un nombre plus grand de fonctionnalités et de professionnels qui comprennent comment les gérer. Les bases de données NoSQL ne sont donc, pas la solution miracle pour répondre à toutes les problématiques de stockage sur le web ou ailleurs. Il est surtout très important, de bien comprendre, ce que le choix d'une base de données de ce type va avoir comme conséquences en termes d'architecture logicielle et complexité de développement.

— difficulté d'administration : Un des buts des bases de données NoSQL était d'en simplifier l'administration, mais la réalité est différente. Aujourd'hui, les solutions NoSQL demandent beaucoup de compétences pour leur installation ainsi que des efforts conséquents lors de leur maintenance.

— Aucune contrainte et validation à exécuter dans la base de données.

— Interopérabilité : La passage d'une base de données NoSQL vers une autre n'est pas transparente pour une application.

— Peu de mises à jour sont prises en charge : Le problème qui se pose est que, dans NoSql, les données peuvent ne pas être cohérentes, ce qui signifie que les deux nœuds peuvent avoir des données différentes pour le même identifiant, tandis que la base de données SQL vous donne des propriétés ACID par lesquelles nous pouvons les résoudre .

3.3 Comparaison entre le SQL et NoSQL

— Les bases de données SQL sont des bases de données basées sur des tables tandis que les bases de données NoSQL sont des bases de données basées sur des paires clé-valeur, graphe , colonne ou encore Document . Cela signifie que les types de données représentées sont différents et que les relations entres elles sont aussi différentes
— Les requêtes SQL sont pour les données structurées alors que NoSQL traite les données non structurées.
— Les bases de données SQL sont destinées au traitement de données dont le volume est petit à moyen ou élevé par contre les bases de données NOSQL peuvent traiter un volume de données élevé.
— Les bases de données SQL conviennent parfaitement à l'environnement exigeant de nombreuses requêtes complexes , tandis que les bases NoSQL ne conviennent pas aux requêtes complexes
— Les bases de données SQL évoluent verticalement. Vous pouvez gérer l'augmentation de la charge en augmentant le processeur, la RAM, le SSD, etc. sur un seul serveur. D'autre part, les bases de données NoSQL évoluent horizontalement. Vous pouvez simplement ajouter quelques serveurs supplémentaires facilement dans votre infrastructure de base de données NoSQL pour gérer le trafic important.
— Dans les systèmes relationnels nous devons garantir l'acidité contrairement au NOSQL , les garanties ACID ne sont pas requises mais suivent le théorème Brewers CAP (Cohérence, Disponibilité et Tolérance de partition).

— Pourquoi utiliser les BDD SQL
SQL protège activement l'intégrité de votre base de données en fournissant la conformité ACID. En raison de ses données structurées, vous n'avez besoin d'aucun support système intégré pour utiliser différents types de données. De plus, SQL est l'option la plus recommandée par de nombreuses entreprises en raison de sa structure et de ses schémas prédéfinis.
— Pourquoi utiliser les BDD NOSQL
Le NOSQL vous permettant de stocker différents types de données ensemble et vous pouvez facilement évoluer en répartissant plusieurs serveurs. Si vous avez besoin de développer une application dans un délai imparti, vous pouvez opter pour NoSQL, qui améliorera vos performances grâce à sa phase de développement rapide

3.4 Comparaison entre les différentes base de données NOSQL

Comment faire son choix ?

— Apache Cassandra pour les acteurs du web : Initialement développée par Facebook (qui l'a publiée en open source en 2008), Apache Cassandra fait figure de star dans le monde web. Cette base NoSQL orientée colonnes a été adoptée par Apple, Netflix ou Spotify. Cassandra peut gérer de grands volumes de données et privilégie les performances notamment en lecture. Elle nécessite toutefois un réel travail de normalisation , mais elle peut également convenir aux jeunes pousses en raison de ses capacités de dimensionnement.

— Couchbase un outil de requêtage "SQL like" : Couchbase, un outil de requêtage "SQL like" lancé en 2010 par des anciens du projet d'infrastructure distribuée Memcached, Couchbase se caractérise par une architecture type maître-maître. Cette base orientée documents privilégie la cohérence des données aux performances pures. Couchbase dispose d'un outil de requêtage normalisé SQL baptisé N1QL (qui se prononce Nickel). Ce qui peut faciliter sa prise en mains par des développeurs rompus aux bases SQL. Avec une solution de cache intégrée, Couchbase vise les applications web. C'est la seule solution NoSQL de notre comparatif à proposer une offre pour mobiles (Couchbase Lite).

— Elasticsearch, pour la force du moteur de recherche : Elasticsearch est avant tout connu pour son moteur de recherche distribué. Il utilise la bibliothèque d'indexation open source Lucene tout comme Apache Solr à qui il est souvent comparé. Cet outil permet de stocker et analyser des données, structurées ou non, comme des textes libres ou des logs systèmes. Projet open source développé en Java sous licence Apache, Elasticsearch est associé à deux autres produits open source : le visualiseur de données Kibana et l'outil d'extraction, de transformation et de chargement de données (ETL) Logstash.

— HBase , pour les très gros volumes : Issue de la famille Apache, HBase est souvent opposé à Cassandra. Il s'agit là encore d'une base orientée colonnes. Basée sur une architecture maître-esclave et s'inspirant de BigTable de Google, elle peut gérer d'énormes quantités de données, plus encore que Cassandra. HBase est une base un peu à part car intimement liée à Hadoop dont elle est un sous-projet. Elle s'installe d'ailleurs sur son système de fichiers distribué HDFS. "Destinée aux fortes volumétries, HBase privilégie d'avantage les possibilités de requêtage à la cohérence des données", ajoute Christophe Parageaud. Produit complexe, HBase exige un gros travail de structuration.

— MongoDB : C'est l'une des plus populaires, développée depuis 2007 par la société du même nom, MongoDB est la base NoSQL la plus populaire selon le palmarès de DB-Engines. Basé sur une architecture de type maître-esclave, ce moteur orienté documents est reconnu pour la souplesse de sa structure. pas besoin de pré structurer les données, il suffit de créer des collections et d'y mettre des éléments Json sans avoir besoin de dire comment les organiser. MongoDB est une base générique qui répond à 80% des besoins couverts par une base relationnelle traditionnelle, à l'exception du transactionnel."

— REDIS, pour sa vitesse : Lancé en 2009, le projet est sponsorisé par VMware, Redis est une base de données reposant sur le principe de clé-valeur. En rendant les données facilement accessibles, elle privilégie la vitesse d'exécution. Revers de

la médaille, cette base n'est pas taillée pour les gros volumes et ses capacités de requêtage sont limitées. Elle ne gère pas la recherche multicritères. Redis se destine aux applications nécessitant une haute disponibilité et une faible latence, notamment dans l'e-commerce. Distribué sous licence BSD et écrit en code C, c'est le seul moteur de notre comparatif à gérer le transactionnel.

— Riak , pour sa tolérance aux pannes : Distribué sous licence Apache et inspiré de Dynamo, Riak est un système de gestion de base de données orienté clé-valeur. Sorte de version évoluée de Redis, il développe les capacités de requêtage en offrant la possibilité, via des index secondaires, d'aller requêter dans la valeur. Basho Technologies, l'éditeur qui développe Riak, fait de la tolérance aux pannes et de la haute disponibilité . Il propose des solutions taillées spécifiquement pour le stockage dans le cloud (Riak CS) et pour l'internet des objets avec la prise en compte des time series (Riak TS).

Deuxième partie

Synthèse

Problématique

Avec la croissance du volume de données qui doivent être enregistrés, traités et mis à jour de plus en plus vite. Le NoSQL s'est imposé ces dernières années comme étant le système de base de données le plus adéquat afin de stocker de grosses quantités de données, cependant les avantages apportés par ce système ne sont pas sans contreparties.

Quand ont parle de bases de données NoSQL en parles d'immenses masse de données hétérogènes qui proviennent de plusieurs endroits ce qui nous ramène au théorème de CAP qui dit qu'entre la cohérence des données et leurs disponibilité ainsi que le partitionnement du réseau, seul **trois** de ces contraintes peuvent être satisfaite en même temps, mais dans le cas des bases de données NoSQL le partitionnement du réseau est primordiales, ce qui nous obligent à choisir entre la disponibilité des données et leurs cohérences.[10]

Afin de bien illustrer cette situation nous proposons la figure ci-dessous :

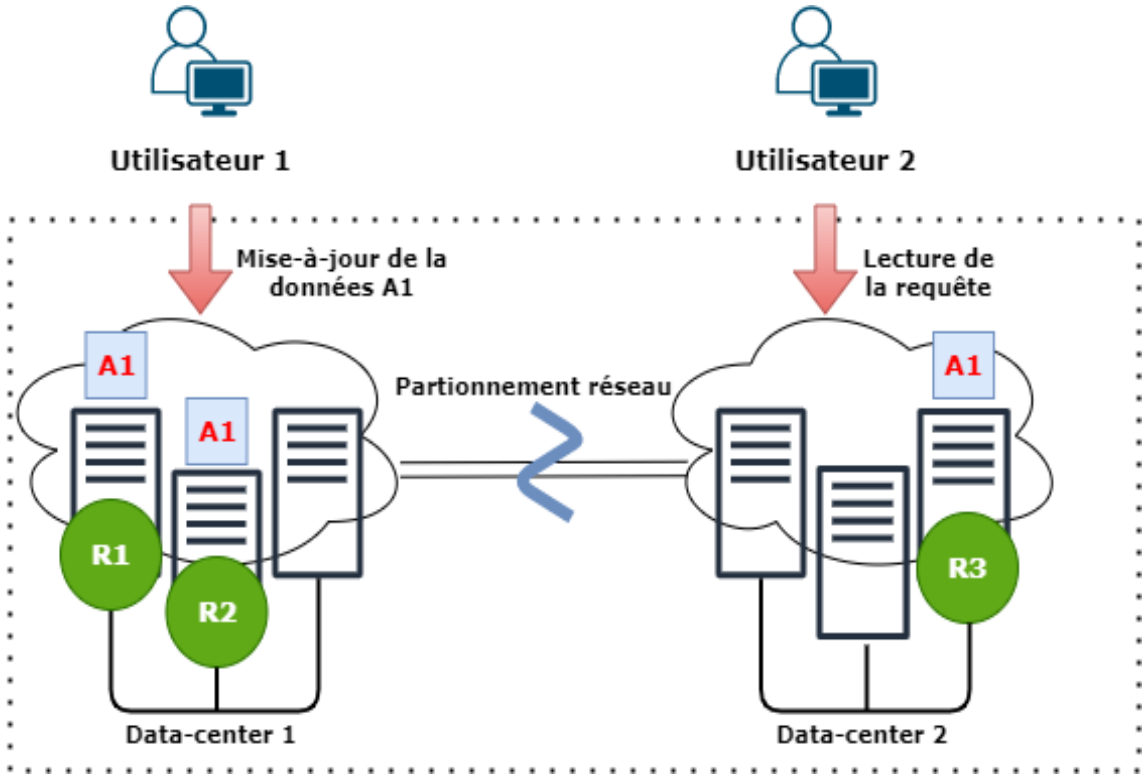

FIGURE 3.8 – Cohérence et disponibilité dans les systèmes distribuées.

Au niveau de la **figure 3.8** nous Pouvons observer que l'utilisateur 2 effectue une demande de lecture pour l'élément de données **A1** dans la réplique **R3** (Datacenter 2), après que l'utilisateur 1 a mis à jour l'élément de données **A1** dans la réplique **R1** (Datacenter 1) en présence d'une partition réseau qui isole les deux centres de données.

Étant donné la partition réseau, il existe deux scénarios possibles :

— Soit toutes les répliques peuvent être disponibles sans que la mise à jour de la donnée **A1** soit propagée vers la réplique **R3**, ce qui fait que l'utilisateur **2** lira des données obsolètes, violant ainsi la cohérence.
— Soit l'utilisateur **2** doit attendre que la partition réseau soit corrigée et que la mise à jour ait été propagée à la réplique **R3**, violant ainsi la disponibilité.

Pour aider à résoudre ce problème, des systèmes de base de données NoSQL ont vu le jour. Ces systèmes ont été créés avec une exigence standard à l'esprit, l'évolutivité. Les compromis causés par le théorème de **CAP** ont conduit à la prolifération de systèmes **non-ACID** pour la construction d'applications basées sur le cloud, appelés **BASE** [1] qui sont des systèmes qui sont essentiellement disponibles, reposent sur le maintien d'un état souple qui peut être reconstruit en cas d'échecs et ne sont finalement cohérents que pour survivre aux partitions réseau.[50]

Approche BASE

L'approche BASE selon E.Brewer [2] perd les propriétés ACID de cohérence et d'isolement au profit de "la disponibilité, la dégradation gracieuse et les performances". L'acronyme **BASE** est composé des caractéristiques suivantes :[51]

— Fondamentalement disponible.
— État doux.
— Finalement cohérent.

En ce qui concerne les bases de données, Brewer conclut que les bases de données actuelles sont plus cohérentes que disponibles et que les bases de données étendues ne peuvent pas avoir les deux, une notion qui est largement adoptée dans la communauté NoSQL. Les systèmes qui peuvent être caractérisés par les propriétés BASE incluent le Dynamo d'Amazon, qui est disponible et tolérant aux partitions mais pas strictement cohérent, c'est-à-dire que les écritures d'un client ne sont pas vues immédiatement après avoir été validées pour tous les lecteurs. BigTable de Google ne choisit ni ACID ni BASE mais la troisième alternative CAP étant un système cohérent et disponible et par conséquent incapable de fonctionner pleinement en présence de partitions de réseau.

Brewer souligne des traits et des exemples des trois choix différents qui peuvent être faits selon le theorem CAP. De plus, il oppose ACID à BASE tout en considérant les deux concepts.[51]

On peut résumé les propriétés BASE de la manière suivante : *une application fonctionne essentiellement tout le temps (essentiellement disponible) ; n'a pas besoin d'être cohérent tout le temps (soft-state) ; mais sera éventuellement dans un état connu (cohérence éventuelle)*

Le problème de cohérence et de disponibilité de données n'est pas le seul frein que rencontrent les bases de données NoSQL, bien qu'elles soient présentes depuis un bon moment, leurs technologies sont encore immatures par rapport à celles des bases de don-

1. Acronyme : Basically Available Soft-state Eventually-consistent
2. Eric Brewer est un scientifique américain, professeur émérite d'informatique à l'Université de Californie à Berkeley et vice-président de l'infrastructure de la société Google. Ses domaines de recherche comprennent les systèmes d'exploitation, l'informatique distribuée, le cloud computing et les réseaux de capteurs.

nées relationnelles qui résiste à l'épreuve du temps et qui continue d'ajouter de nouvelles innovations, De plus, elles sont maintenant capable d'absorber les nouveaux types de données (spatiales, semi-structurées, et les modèles de cohérence afin qu'ils puissent coexister dans un seul système). Il n'y a pas de problèmes d'évolutivité inhérents au modèle relationnel ou à la syntaxe de requête SQL. Il fallait juste une implémentation de stockage différente pour tirer parti d'une architecture évolutive. Cela a prouvé que, pour la majorité des cas d'utilisation, les bases de données relationnelles sont plus faciles à utiliser et généralement plus performantes que les systèmes NoSQL. Un autre problème confronté par les BDD NoSQL est la difficulté d'administration qui est un des points à ne pas négliger, si à la base elles ont été créées afin de simplifier l'administration, la réalité est différente. Aujourd'hui, les solutions NoSQL demandent beaucoup de compétences pour leur installation ainsi que des efforts conséquents lors de leur maintenance, et une infrastructure de stockage immense qui garantit la répartition de ce volume immense de donnée (tolérance au pannes),elles ont beau être « administrator-friendly » (simples à administrer), cela n'a pas de sens si les administrateurs en question ne savent pas comment les utiliser.

Sur les quantités de données à stocker, aucune contrainte et validation n'est faite sur elles en général, le principe c'est de sauvegarder toute donnée, car cette dernière est susceptible d'être utilisée, si ce n'est pas le cas à cet instant il le sera à un instant "t".

A l'heure actuelle, les bases de données relationnelles sont beaucoup mieux implémentées dans la majorité des entreprises, ces dernières ne veulent pas passer au base NoSQL en vu des problèmes liée à la migration d'une base relationnels à une base NoSQL et de l'interopérabilité (passage d'une base de données NoSQL vers une autre), la cause de cela c'est que ces bases de données on été crée par les géant du web (facebook, google, amazon, microsoft etc), et chacun à créer un système adapter pour ces propres besoins, y'a des modèle qui sont plus focalisé sur la contrainte de disponibilité de donnée et y'a ceux qui sont plus focalisé sur la contrainte de cohérence de données etc.À cause de cela malgré le critère d'open source pour les BD NoSQL qui est à la fois sa plus grande force et sa plus grande faiblesse il n'existe pas encore de normalisation fiable pour les BD NoSQL.

Si la migration ou l'interopérabilité est un problème liée à l'architecture des base NoSQL, l'extraction d'un modèle de donnée via une base de données Nosql n'en reste pas moindre, actuellement, les SGBD NoSQL ne disposent pas d'une fonctionnalité permettant d'afficher dynamiquement le modèle de la BD. Or, les utilisateurs (développeurs, Data scientists, Data analysts, etc.) ont besoin du modèle de données pour exprimer des requêtes d'interrogation et de mise à jour des données. En vu de la jeunesse de ce type de BD ce dernier n'est pas encore très utilisé dans toutes les entreprises, il y a donc un support limité (Documentation) ainsi qu'un support très faible de la communauté car le plus grand des travaux est consacré à la recherche sur le sujet. Interroger de tel système de BD n'est vraiment pas facile en vue de la quantité, les types de données à traiter ainsi que l'énergie nécessaire (CPU) à fin de retourner le résultat attendu en minimum de temps possible (idéalement en temps réel).

Dans les systèmes "not only SQL", le problème le plus critique est avant tout le partitionnement réseau c'est à dire que le système est distribuée et partagée sur différents réseau, ce dernier doit assurer doit pouvoir fonctionner même en cas de panne du réseau ce qui n'est pas évident à mettre en place dans de tel systèmes qui sont généralement temps réels .

Les bases de données NoSQL ne sont donc, pas la solution miracle pour répondre à tous les besoins . Il est surtout très important, de bien comprendre, ce que le choix

d'une base de données de ce type va avoir comme conséquences en termes d'architecture logicielle et complexité de développement, et ainsi en cas d'adoption de cette solution, il faudra savoir saisir quel est le l'élément le plus important entre la disponibilité et la cohérence de données.

Ce problème de cohérence et de disponibilité qu'illustre le théorème de CAP est particulièrement délicat dans le contexte des systèmes de stockage avec réplication de données, répartie géographiquement, car le but est de savoir comment atteindre un état cohérent dans toutes les répliques. Garantir une forte cohérence dans un tel environnement entraîne des surcoûts de performance importants en raison de la latence accrue du réseau entre les centres de données et du fait que les partitions réseau peuvent entraîner une indisponibilité du service . En conséquence, des modèles spécifiques ont été proposés pour offrir des garanties de cohérence plus faibles ou assouplies .Obtenir le juste équilibre entre des niveaux plus élevés de cohérence et de disponibilité est l'un des défis ouverts de cette décennie. Dans ce but, nous nous concentrerons sur les méthodes de pointe pour la cohérence et la disponibilité de données dans les environnements cloud.[10]

Plusieurs entreprises optent pour assurer un niveau de disponibilité et de performances élevés même en présence de partitions réseau plutôt que d'offrir un modèle de cohérence plus solide. Les environnements de stockage de données basés sur NoSQL fournissent des propriétés de cohérence en mode éventuel, ce qui signifie que toutes les modifications apportées à un élément de données répliquées atteignent finalement toutes ses répliques. Cependant, l'utilisation de ce type de cohérence augmente la probabilité de lire des données obsolètes, car les répliques auxquelles on accède peuvent ne pas avoir reçu les écritures les plus récentes. Cela a conduit au développement de solutions de cohérence adaptative, qui permettent d'ajuster le niveau de cohérence au moment de l'exécution afin d'améliorer les performances ou de réduire les coûts, tout en maintenant le pourcentage de lectures obsolètes à de faibles niveaux.

Chapitre 4

Disponibilité et cohérence des données dans les BD NoSQL

Introduction

La prolifération de sources de données allant des médias sociaux et de l'Internet des objets (IoT) aux données générées industriellement (par exemple, les transactions) a conduit à une demande croissante d'applications basées sur le cloud à forte intensité de données et a créé de nouveaux défis pour les bases de données de l'ère du Big Data.

Les bases de données NoSQL sont utilisées de nos jours par les développeurs d'applications cloud pour exploiter ces grandes quantités de données sans schémas spécifique. Ils ont un schéma flexible, impliquant la possibilité d'un schéma évoluant dynamiquement. Ces bases de données ne nécessitent pas de définition de schéma avant d'insérer des données et ne nécessitent pas de modification de schéma lorsque les besoins de gestion des données évoluent.[51]

L'Un des problème difficile qui se pose dans le contexte des systèmes de stockage en nuage avec réplication des données géographiquement distribuées est de savoir comment atteindre un état cohérent dans toutes les répliques. La mise en œuvre de la réplication synchrone pour garantir une cohérence élevée dans un tel environnement entraîne des surcharges de performances importantes en raison de la latence réseau accrue entre les centres de données et du fait que les partitions réseau peuvent entraîner une indisponibilité du service, Afin d'y remédier à ce problème plusieurs modèles spécifiques ont été proposés pour offrir des garanties de cohérence plus faibles ou assouplies.[10]

Dans ce chapitre on va voir tout d'abord les concepts généraux liée à la gestion de BDD Cloud puis nous allons étudier le conflit entre la cohérance et la disponibilité dans les systèmes réparti ainsi que les différentes méthodes de cohérences proposés afin de palier à ce compromis.

4.1 Concepts généraux

4.1.1 Réplication et cohérence de donnée

La réplication des données sur plusieurs nœuds est une solution efficace afin d'avoire de meilleurs performances et une disponibilité des données élevé. Le traitement distribuée des demandes d'accès améliore les performances en termes de temps de réponse et de Charge acceptable par le système (traitement parallèle).

Remarque *Il faut savoir qu'une donnée ne devient indisponible que si tous les nœuds qui en possèdent une copie tombent en panne simultanément.*

Le problème général lié à la réplication de ces données sur plusieur noeuds est celui de la cohérence des données au niveau des différentes répliquent. Afin de gérer cette dernière en a deux points de vue essentielle :

- Du point de vue des demandes d'accès, la gestion de la concurrence locale se préoccupe de leur isolation, c'est-à-dire de les exécuter en parallèle en évitant qu'une mise à jour soit visible par d'autres demandes d'accès tant qu'elle n'a pas été validée.
- Du point de vue des données, la gestion des copies doit assurer leur cohérence mutuelle, c'est-à-dire que toutes les copies d'une donnée soient identiques.

Sans contrôle, l'exécution simultanée des demandes d'accès peut conduire à des phénomènes indésirables. Néanmoins, accepter de vivre avec certaines imperfections peut améliorer la performance du système en améliorant la concurrence.

4.1.2 La réplication

La duplication des données est une solution très efficaces pour faciliter l'accès aux données tout en augmentant leur disponibilités, soit parce qu'un nœud voisin peut prendre la relève lorsque le serveur principal s'écroule (cas de panne), soit parce que les données sont copiées sur différents nœuds permettant de répartir les requêtes et de les traités en parallèles. Cela fournit une meilleur tolérance aux pannes et un meilleur temps de réponse.

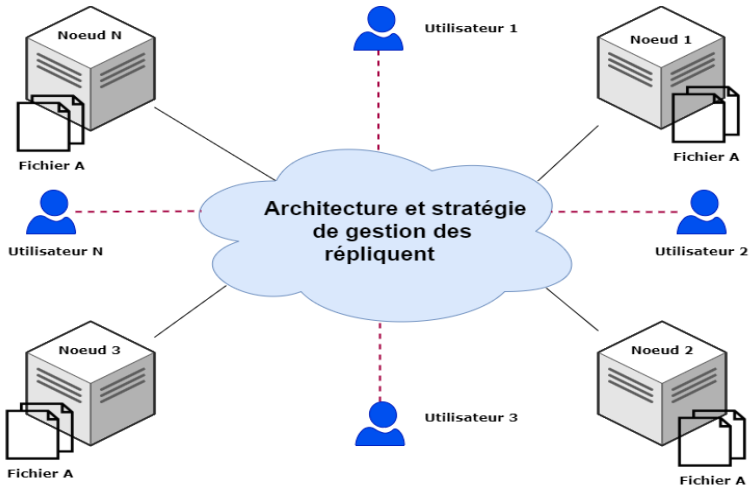

FIGURE 4.1 – Environnement d'utilisation de la technique de réplication.

La réplication peut se faire à plusieurs niveaux et prendre plusieurs formes comme suit :

- **Aucune réplication :** Les données ne sont pas dupliquées entre les sources du système (cas de la centralisation).
- **Réplication partielle :** Les données sont répliquées sur quelques sources du système.
- **Réplication totale :** Les données sont répliquées sur toutes les sources du système.

Cependant Il faut bien distinguer entre la réplication et les autre type de partage de données tel que :

La sauvegarde : Les données sauvegardées ne changent pas dans le temps (état fixe des données), tandis que les données répliquées évoluent sans cesse à mesure que les données sources changent.

La copie : À la différence de copier, la réplication gère la cohérence des répliques, donc une réplique est plus qu'une copie.

4.1.2.1 Avantage de la réplication

— Une amélioration de la fiabilité ou bien une sûreté de fonctionnement.

Exemple :
— Si une copie tombe en panne, il est toujours possible d'obtenir les données à partir d'une autre copie.
— La redondance permet une meilleure protection contre la corruption de fichiers.

— Amélioration des performances.
Exemple :
— Diviser la charge de travail entre plusieurs serveurs.
— Extensibilité géographique en rapprochant les serveurs des clients.

— Une disponibilité de données.
Les données sont disponibles (dans le réseau locals) même en l'absence de toute connexion à un serveur central,de sorte que l'utilisateur n'est pas coupé de ses données en cas de défaillance d'une connexion réseau longue distance.

— Temps de réponse.
Exemple :
— Les requêtes sont traitées sur un serveur local sans accès à un réseau étendu, ce qui accélère le débit.
— Par ailleurs, le traitement local allège la charge du serveur de bases de données central, ce qui permet de moins solliciter le processeur.

4.1.2.2 Inconvénients de la réplication

Malgré tous les avantages qu'elle procure, la réplication soulève un certain nombre de problèmes que nous allons aborder dans ce qui suit :

- **Placement des répliques :** Ce problème consiste à choisir, des localisations physiques pour les répliques, qui réduisent les coûts de stockage et d'accès aux données en fonction des objectifs des applications et de la réplication.
- **Choix d'une réplique :** Ce problème consiste à sélectionner, parmi toutes les répliques d'une donnée, celle qui est la meilleure du point de vue de la consistance.
- **Degré de réplication :** Ce problème concerne la recherche du nombre minimal de répliques qu'il faut créer pour une donnée, sans pour autant réduire les performances des applications. Cela peut être déterminé d'une manière prédictive ou adaptative.
- **Cohérence des répliques :** Parmi les problèmes liés à la réplication, le problème de la cohérence des données est sans doute celui qui est le plus complexe, c'est sur ce dernier que nous allons nous intéresser dans la suite de ce chapitre.

4.1.3 La cohérence

Les avantages de la réplication des données sont contraints par le problème de cohérence mutuelle des copies, car cette réplication doit être transparente vis à vis de l'utilisateur se qui offre la garantie d'une vue cohérente des données dispersées.

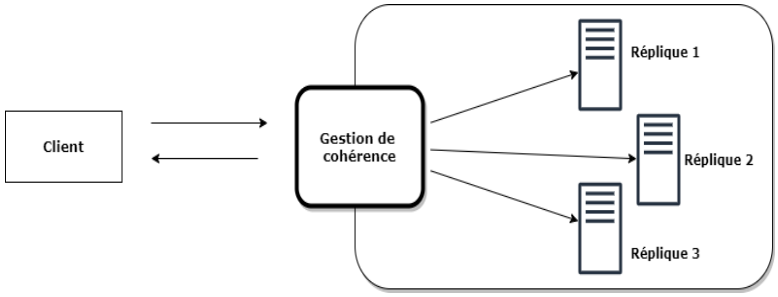

FIGURE 4.2 – Vue cohérente des donnée.

La gestion de cohérence est donc définie comme étant le contrôle des accès, afin de fournir une vision qui fait abstraction de la réplication, de la distribution et de la concurrence des accès aux données partagées.

4.1.3.1 Types de cohérence

La gestion des copies en termes de propagation des mises à jour est ainsi nécessaire. La charge induite par cette cohérence peut avoir un impact significatif sur le système notamment du point de vue performance. Il s'agira donc de garantir la cohérence mutuelle d'un ensemble de répliques dans des délais acceptables, on distingues deux type de cohérence :

- **Cohérence forte :** Une cohérence de répliques est forte lorsque toute interrogation d'une copie quelconque reflète le résultat de toutes les modifications antérieures.

- **Cohérence faible :** Une cohérence de répliqués est dite faible, si on tolère qu'une interrogation ne reflète pas toutes les modifications antérieures, avec la garantie que celles-ci seront toutes répercutées au bout d'un temps fini.

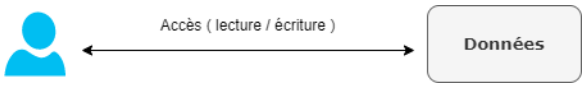

FIGURE 4.3 – Vue idéale des données.

FIGURE 4.4 – Vue réelle des données.

4.1.4 Exigences de stockage des données dans le cloud

Le stockage de données dans le cloud exige une infrastructure fiable et appropriée, afin que toutes les ressources puissent être utilisées et partagées efficacement et cela pour réduire les problèmes liées à la cohérence de données.

Pour garder la bonne gestion des bases de données dans les systèmes distribuées il faut veiller au respect de certain critères :[10]

Automatisation : Le stockage des données doit être automatisé pour pouvoir exécuter rapidement les changements d'infrastructure nécessaires pour maintenir la cohérence des répliques et cela sans intervention humaine.

Disponibilité : Le stockage des données doit garantir que les données continuent d'être disponibles à un niveau de performance requis dans des situations allant de normales à défavorables.

Élasticité : Non seulement le stockage de données doit pouvoir évoluer avec une charge croissante, mais il doit également pouvoir s'adapter aux réductions de charge en libérant des ressources cloud, tout en garantissant la conformité avec un accord de niveau de service (SLA [1]).

Tolérance aux pannes : Le stockage de données doit pouvoir récupérer en cas de panne, par exemple en fournissant une instance de sauvegarde de l'application qui sera prête à prendre le relais sans interruption.

Faible latence : Le stockage de données doit gérer les problèmes de latence en mesurant et en testant la latence du réseau, avant d'enregistrer les données modifiées

1. Service- Level Agreement : est un document qui définit la qualité de service, prestation prescrite entre un fournisseur de service et un client

par une application et avant de mettre ces données à la disposition d'autres applications.

Tolérance de partition : Le système doit continuer à fonctionner malgré les partitions réseau.

Performance : Le stockage de données doit fournir une infrastructure qui prend en charge un accès, une mise à jour et une récupération de données rapides et robustes.

Fiabilité : Le stockage des données doit garantir que les données peuvent être récupérées en cas de catastrophe.

Évolutivité : Le stockage de données doit évoluer rapidement pour répondre aux demandes de charge de travail, offrant ainsi une évolutivité horizontale et verticale.

Afin de garantir l'intégrité des données, la plupart des systèmes de bases de données classiques sont basés sur des transactions. Cela garantit la cohérence des données dans toutes les situations de gestion des données. Ces caractéristiques transactionnelles sont également appelées propriété ACID (atomicité, cohérence, isolation et durabilité).

Cependant, il n'est pas trivial d'assurer les propriétés ACID dans un stockage de données cloud, précisément parce que les données sont répliquées sur plusieurs serveurs. Malgré cette difficulté, des stratégies ont été proposées pour tenter d'émuler les propriétés ACID pour les transactions d'applications Web.

Exemple :
- *l'atomicité peut être garantie en implémentant le protocole de validation en deux phases (2PC).*
- *l'isolement peut être obtenu par un contrôle d'accès simultané multi-versions ou par un horodatage global.*
- *la durabilité en appliquant des stratégies de mise en file d'attente telles que FIFO (First-In, First-Out) aux transactions d'écriture simultanées, afin que les anciennes mises à jour ne remplacent pas les dernières.*

Cependant, la réplication représente un obstacle important pour garantir la cohérence, à fin de garantir cette dernière des stratégies de mise à jour et de propagation des données doivent être mises en œuvre, c.à.d si une copie est mise à jour, toutes les autres doivent également être mises à jour.

De ce fait plusieurs conflits surgissent entre les différents aspects de la haute disponibilité, de la cohérence ainsi que la tolérance de partition dans les systèmes distribués - connus sous le nom de théorème CAP.

Bien que la mise à l'échelle horizontale puisse sembler préférable, le théorème de CAP montre qu'étant donné les partitions réseaux qui sont inévitables dans un scénario géographiquement distribué, nous mettons en évidence le compromis entre cohérence et disponibilité.

De tels systèmes non ACID offre des modèles de cohérence distincts, qui seront discutés ci-après.

4.1.5 Modèles de cohérence

Comme discuté précédemment, le problème de cohérence est l'un des problèmes les plus importants dans la conception des systèmes de stockage à grande échelle. En règle générale, la cohérence des données peut être divisée en deux catégories :[10]

— Du point de vue des données, le système de stockage de données distribué synchronise les opérations d'accès aux données de tous les processus pour garantir des résultats corrects.
— Du point de vue du client, le système synchronise uniquement les opérations d'accès aux données du même processus, indépendamment des autres, pour garantir leur cohérence.
.

Remarque

Cette perspective est justifiée car il arrive souvent que les mises à jour partagées soient rares et accèdent principalement à des données privées.

Cependant, aujourd'hui, les chercheurs cherchent les moyens de présenter un hybride des modèles de cohérence par rapport aux exigences des applications. L'un des principaux objectifs de cette recherche est d'introduire certains des modèles de cohérence qui, contrairement aux modèles hybrides, sont non seulement capables d'assurer la cohérence des systèmes de stockage distribué, mais sont également capables de couvrir les besoins des applications qui sont généralement répondu par les modèles de cohérence hybride.[53]

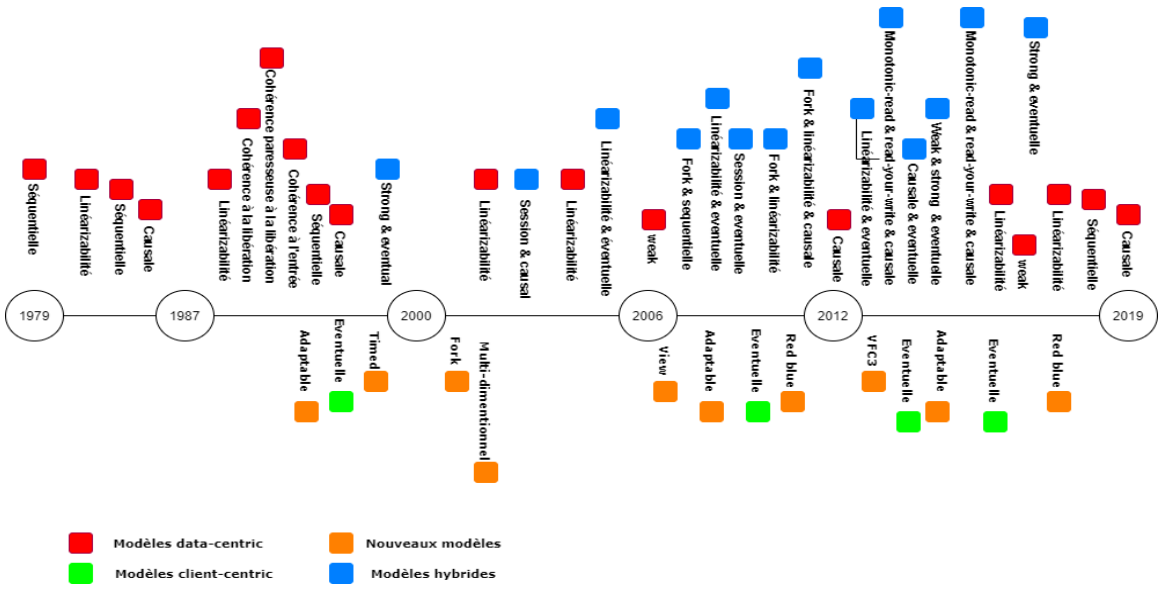

FIGURE 4.5 – Classification des différents modèles de cohérence utilisés dans les systèmes distribués.

4.1.5.1 Modèles de cohérence centrés sur les données (Data-Centric)

Dans cette perspective, les modèles de cohérence cherchent à garantir que les opérations d'accès aux données suivent certaines règles qui garantissent le bon fonctionnement du système de stockage. Ces règles sont basées sur la définition des résultats attendus après les opérations de lecture/écriture (même si ces opérations sont effectuées simultanément).

Cependant, l'absence d'une horloge globale rend l'identification de la dernière opération d'écriture difficile, ce qui nécessite certaines restrictions sur les valeurs de données qui peuvent être renvoyées par une opération de lecture.

Les modèles de cohérence les plus utilisées et qui entrent dans cette catégorie sont :

— La cohérence stricte (strict consistency).
— La cohérence séquentielle (sequential consistency).
— Linéarisabilité (Linearizability model).
— La cohérence causale (causal consistency).
— La cohérence PRAM (PRAM consistency).
— La cohérence faible (weak consistency).
— La cohérence à la libération (Release consistency).
— La cohérence paresseuse à la libération (Lazy release consistency).

a. **Cohérence stricte**

La cohérence stricte est le modèle de cohérence le plus solide qui nécessite une synchronisation globale permanente [55]. Cette synchronisation se fait en utilisant une heure globale absolue (un timestamp). La création de cette synchronisation par le temps physique entre les serveurs est dans une certaine mesure impossible. En d'autres termes, les répliques doivent être synchronisées globalement et en permanence. Ce modèle est si coûteux, alors que le système n'a pas besoin d'être toujours synchronisé globalement. Cependant, dans les systèmes distribués, cette simplicité impose de nombreuses dépenses. En ce qui concerne le comportement du modèle de cohérence sur les répliques.[52][53][55]

Dans un tel système il y a un chevauchement entre les sous ensembles de répliques qui sont mis à jours dans le cadre de l'écriture, et les sous ensembles de répliques consultés lors d'une lecture, c'est ainsi qu'un système strictement cohérent peut toujour garantir une lecture des données les plus récentes.[54]

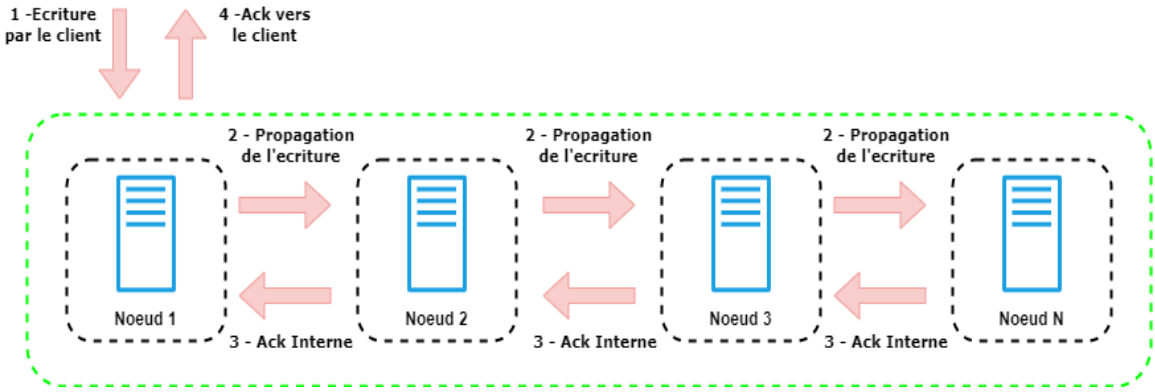

FIGURE 4.6 – Système strictement cohérent.

à travers la **figure f.ff** nous pouvons observer que lors d'une écriture(mise à jour) au niveau d'un système distribuée strictement cohérent au niveau d'un noeud (serveur), cette écriture est synchroniser juste après vers toutes les autres répliquent simultanément, ainsi il n'y a pas d'incohérence lors de la lecture des données.[55]

Pour résumer un tel système :

- *Renvoie toujours la dernière écriture : pour toute opération d'écriture entrante, une fois qu'une écriture est acquittée au client, la valeur mise à jour est visible lors de la lecture depuis n'importe quel nœud.*
- *Garantie la résilience des données : pour toute opération d'écriture entrante, une fois qu'une écriture est acquittée vers le client, la mise à jour est protégée contre la défaillance du nœud avec redondance.*

Ce modèle de cohérence fournit le niveau de cohérence le plus élevé. Il indique que, si une opération d'écriture est effectuée sur un élément de données, le résultat doit être instantanément visible pour tous les processus, quelle que soit la réplique sur laquelle l'opération d'écriture a été exécutée. Pour y parvenir, un ordre de temps global absolu doit être maintenu.[10]

Inconvénient :
Un tel modèle de cohérence n'est pas toujours utilisé Principalement parce que la mise en œuvre d'une cohérence stricte peut avoir un impact significatif sur les performances. Plus précisément, la latence et le débit seront affectés. L'ampleur de l'impact dépend du scénario.

b. **Cohérence séquentielle**

Une autre variante des modèles de cohérence est la cohérence séquentielle. Ce modèle a d'abord été défini par Lamport [2] en 1979. Dans la discipline de la mémoire partagée pour les systèmes multiprocesseurs. Ce modèle est une variante plus simple du modèle de cohérence stricte mais reste assez strict car c'est un modèle qui requiert que : [52][53][55]

- Toutes les opérations soient sérialisées dans le même ordre dans toutes les répliques.[10]
- Toutes les opérations du même processus soient exécutées dans l'ordre où le système de stockage les a reçues.[10]

Dans ce modèle, l'opération d'écriture est vue entre les processus avec un ordre égal, cependant l'opération de lecture effectuée par les autres processus n'est pas observable.
Afin de bien comprendre ce modèle on va se focaliser sur le cas ou il y'a qu'une seul réplique à laquelle plusieurs processus peuvent accéder :[53]

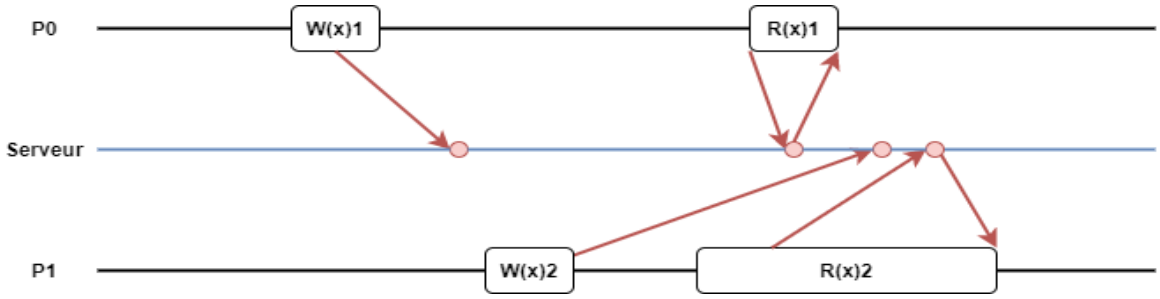

FIGURE 4.7 – Cohérence séquentielle.

Afin de décrire le comportement attendu d'un objet partagé on exige que ce dernier imite l'existence d'une seule copie de l'objet dans le réseau, à laquelle tous les processus accèdent séquentiellement. Une façon de comprendre cette propriété est que cette copie partagée est stockée sur un serveur qui trie les opérations selon ses propres ordres séquentiels.

Par exemple, considérons l'historique donné sur la **figure f.ff**, dans lequel deux processus accèdent à un registre se trouvant au niveau d'un serveur qui gère la copie unique de l'objet ; le processus P0 écrit 1 dans le registre à fenêtre glissante (opération W(x)1), le deuxième processus écrit 2. puis P0 et P1 veulent tout les deux lire le registre (opération R). le résultat de la lecture de P0 n'est pas mit à jours (plus précisément elle ne va pas avoir lieu avant que la l'écriture de P1 ne soit terminé), mais une fois l'écriture terminée les deux processus auront accès à la dernière mise à jour de l'objet partagée.en outre pour écrire, un processus

2. Leslie B. Lamport, né le 7 février 1941 à New York, est un chercheur en informatique américain, spécialiste de l'algorithmique répartie. Il a obtenu le prix Turing 2013. Il est le concepteur du logiciel libre de composition de documents LaTeX

envoie simplement un message au serveur et attend sa réponse. Puisque le serveur traite séquentiellement les messages, les opérations sur l'objet partagé apparaîtront totalement ordonnées.

Exemple
Pour simplifier si on a une donnée A au niveau d'un serveur et que l'on a 4 processus qui exécute successivement les opérations suivante au niveau du serveur ou il y a l'objet partagée A :

P1 : Write A' à la place de A
P2 : Load B
P3 : Load C
P4 : Load D

Un modèle de cohérence séquentielle dit que les chargement de B,C et D n'auront pas lieu avant que l'écriture de A' soit terminée alors que la valeur de A n'est même pas consulter au niveau du reste du programme ce qui induit à une perte performance.

Le but de la cohérence séquentielle (SC) est d'imiter ce comportement. Il est à noter que le serveur n'est pas nécessaire pour la mise en œuvre de la cohérence séquentielle,un objet partagé séquentiellement cohérent ne doit se comporter que de la même manière. Autrement dit dans un système distribuée il faut garantir cette manière de fonctionner au niveau de chaque répliques ce qui provoque une perte de performance significative.

Mise en œuvre de la cohérence séquentielle dans PNUTS

PNUTS est l'un des des outils qui utilise la cohérence séquentielle, c'est un SGBD massivement parallèle et géographiquement distribué développé par Yahoo. Les développeurs de PNUTS ont observé que les applications Web manipulent généralement un enregistrement à la fois, alors que différents enregistrements peuvent être situés dans des localités géographiques différentes. Par conséquent, une méthode de cohérence basée sur le séquencement des événements établit que toutes les répliques d'un enregistrement donné reçoivent toutes les mises à jour appliquées à cet enregistrement dans le même ordre. Cette stratégie est mise en œuvre en désignant l'une des répliques comme maître pour chaque enregistrement, de sorte que ce maître reçoive toutes les écritures envoyées à cet enregistrement par les autres répliques. Si un enregistrement a la majorité de ses écritures envoyées à une réplique particulière, cette réplique devient le maître de cet enregistrement.

c. **Linéarisabilité**

Un autre type de modèles de cohérence centrés sur les données est la linéarisation, également appelée modèle de cohérence forte. Ce modèle est nettement meilleur que la cohérence séquentielle introduite en 1990 par Morris Herlihy [3] et M.Wing [4]. Ce modèle nécessite également une horloge de synchronisation globale.

3. Maurice Peter Herlihy, né le 4 janvier 1954, est un informaticien américain. Il travaille sur les aspects théoriques et pratiques des systèmes concurrents et distribués.

4. Jeannette M. Wing, née le 4 décembre 1956, est professeur d'informatique à l'Université Carnegie-Mellon. Elle est directrice du département d'informatique.

Comme cette horloge n'est pas aussi digne de confiance, elle est remplacée par une horloge logique dans ce modèle, qui est appelée l'heure logique globale. Le comportement de ce modèle est comme le modèle de cohérence séquentielle. Cependant, l'ordre des opérations est défini par l'ensemble des processus en fonction de leur moment d'occurrence. Dans ce cas, en mettant une limitation sur le modèle de cohérence séquentielle, par exemple, le temps de l'événement, ce modèle change la linéarisation, ce qui a un effet significatif sur l'opération d'écriture sur la mémoire partagée. Par conséquent, l'ensemble des processus basés sur cette cohérence ont une vision solide des opérations.[53]

Afin de garantir la linéarisation, les protocoles réseau sécurisent la répétition automatique de l'opération qui a échoué et les protocoles de validation en deux phases sont utilisés. Ce modèle de cohérence exige que chacune des opérations de lecture ou d'écriture soit effectuée dans un intervalle entre la demande d'exécution de l'opération de réponse.

Pour résumer :

— Il n'y a pas de référence à l'opération d'écriture «la plus récente».
— Le résultat de toute exécution est le même que si les opérations par tous les processus de la mémoire de données ont été exécutés dans un ordre séquentiel et les opérations de chaque processus individuel apparaissent dans cette séquence dans l'ordre spécifié par son programme. De plus, si tepmps(operation1) (x) < temps(operation2) (y), alors operation1 (x) doit précéder operation2 (y) dans cette séquence.

Exemple :

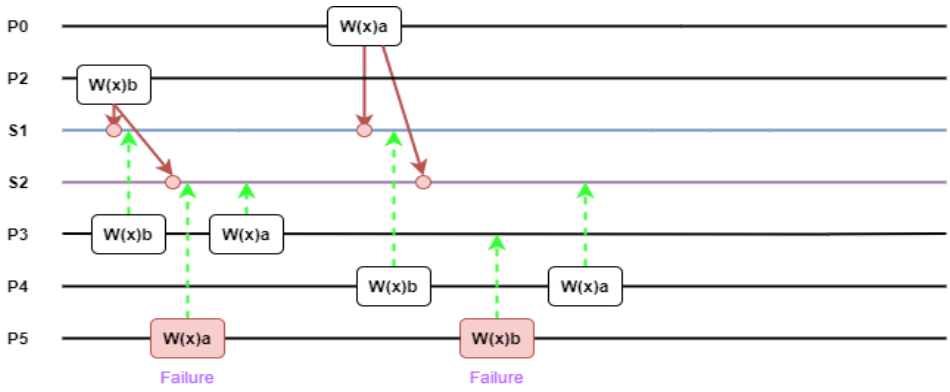

FIGURE 4.8 – Linéarizabilité.

L'ordre et la priorité de l'exécution sont égaux pour l'ensemble des processus et sont basés sur leur heure d'événement logique. Lorsque l'opération d'écriture de a est exécutée dans le temps, après l'opération d'écriture de b sur la mémoire partagée dans le serveur S(i), on s'attend à ce qu'après avoir accès au stockage de données, chaque processus par l'exécution du processus de lecture lise le valeur b , et par la suite, la valeur a. Si au cours du processus de mise à jour du stockage de données, on essaye de lire nouvelle valeur de a, puis la nouvelle valeur de b, ce qui est contradictoire au principe du modèle de linéarisation.

Si le processus P(s) agit en contradiction avec la priorité du processus d'écriture par les processus P1 et P2, d'abord par l'exécution de l'opération de lecture de b puis a, alors le stockage des données a violé le modèle de linéarisation.

d. **Cohérence causale**

Ce modèle a d'abord été proposé pour les systèmes distribués partagés, il est plus faible que le modèle de cohérence séquentielle. Ce modèle de cohérence fait la distinction entre les événements qui ont une relation de cause à conséquence et ceux qui n'en ont pas. Dans le cas où l'opération de lecture est le résultat de plusieurs opérations d'écriture sur la réplique, l'opération de lecture ne s'exécute pas tant que l'opération d'écriture n'est pas terminée. Ce modèle garantit qu'un client ne lit pas deux opérations d'écriture liées dans un ordre incorrect et si le client a lu la dernière valeur, il ne lit pas la valeur périmée [53][55][60].
Plus précisément le modèle de cohérence causale ... assouplit l'exigence du modèle séquentiel pour une meilleure concurrence. Contrairement au modèle de cohérence séquentielle, dans le modèle de cohérence causale, tous les processus ne voient que les opérations de référence mémoire dans le même ordre (correct) qui sont potentiellement liées de manière causale. Les opérations de référence de mémoire qui ne sont pas potentiellement liées de manière causale peuvent être vues par différents processus dans des ordres différents.[52]

> **Remarque :[10]** *Deux requêtes A et B ont une dépendance causale si au moins l'une des deux conditions suivantes est remplie :*
>
> (a) *A et B sont tous deux exécutés sur un seul thread et l'exécution de l'une précède l'autre dans le temps.*
> (b) *B lit une valeur qui a été écrite par A.*
>
> *De plus, cette dépendance est transitive, en ce sens que si A et B ont une dépendance causale et que B et C ont une dépendance causale, alors A et C ont aussi une dépendance causale.*

Ainsi, dans un scénario d'un système de stockage toujours disponible dans lequel les demandes ont des dépendances causales, un niveau de cohérence plus strict que celui fourni par le modèle causal ne peut être atteint en raison des compromis du théorème CAP.

Si un processus exécute une opération d'écriture A, et qu'un processus (le même ou un autre) qui a observé A effectue alors une opération d'écriture B, alors il est possible que A soit la cause de B ; nous disons que A «cause potentiellement» ou «précède causalement» B. La cohérence causale garantit que si A précède causalement B, alors chaque processus du système observe A avant d'observer B. Inversement, deux opérations d'écriture C et D sont dites concurrentes, ou causalement indépendant, si aucun des deux ne précède causalement l'autre. Dans ce cas, un processus peut observer soit C avant D, soit D avant C.[60]

> **Exemple 1 :**
> Voici un exemple de cohérence causale.
> Les relations causales sont respectées dans la séquence d'événements suivante :

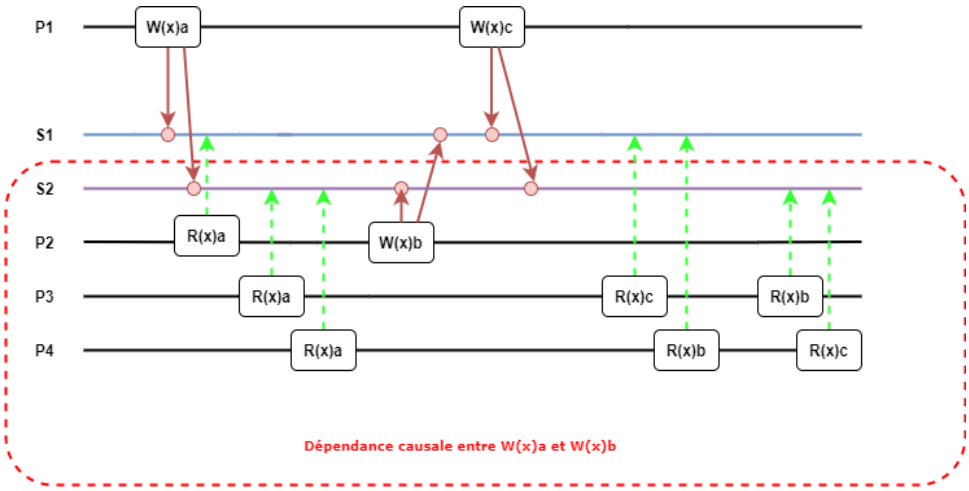

FIGURE 4.9 – Une séquence correcte d'événements qui satisfait la cohérence causal.

Le processus P2 observe, lit, l'écriture antérieure W(x)a effectuée par le processus P1. Par conséquent, les deux écritures W(x)a et W(x)b sont liées de manière causale. Sous cohérence causale, chaque processus observe d'abord W(x)a, avant d'observer W(x)b. Notez que les deux opérations d'écriture W (x)b et W(x)c, sans opérations de lecture intermédiaires, elles sont simultanées et les processus P3 et P4 les observent (lisent) dans des ordres différents.

Exemple 2 : Une séquence d'événements qui viole la cohérence causal

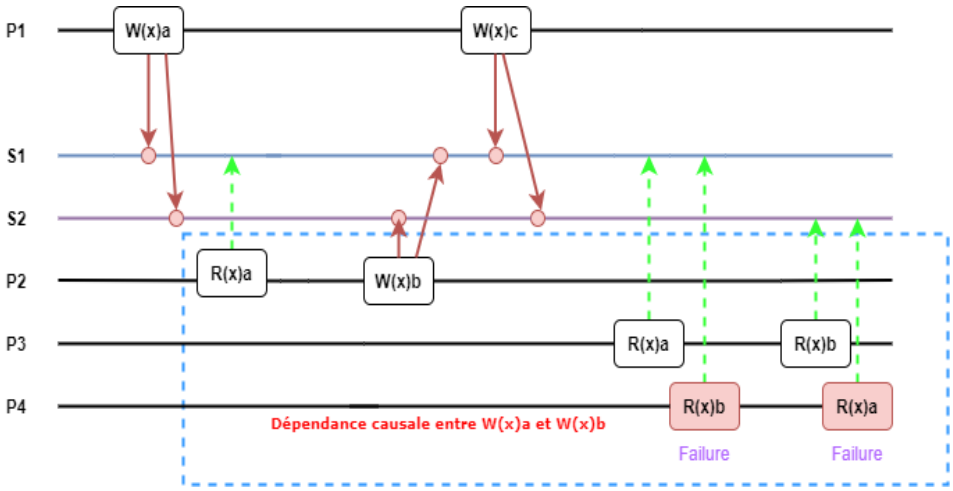

FIGURE 4.10 – Une séquence d'événements qui viole la cohérence causal.

Si un processus exécute une opération d'écriture a, et qu'un processus (le même ou un autre) qui a observé A effectue alors une opération d'écriture b, alors il est possible que a soit la cause de b, donc il faut que chaque processus du système observé a avant d'observer b, or P4 observe b puis observe a et cela ne doit pas se produire dans un système causalement cohérent.

Si on enlève la lecture de P2 de a (R(x)a) alors le système devient causalement cohérent.

e. **Cohérence PRAM**

The first-in, first-out (FIFO), également connu sous le nom de pipelined random access memory (PRAM) est un autre type de modèles de cohérence centrée sur les données. Ce modèle a été proposé par Lipton [5] et Sandberg en 1988 pour la mémoire partagée, c'est un modèle qui garantit que la vue de chaque processus est cohérente avec l'ordre dans lequel les écritures ont été effectuées par chaque processus. Chaque processus doit être capable d'expliquer l'histotique par une linéarisation de ses propres connaissances. La cohérence pipelinée est plus faible que la cohérence séquentielle, pour laquelle il est en outre nécessaire que les linéarisations vues par différents processus soient identiques. La cohérence PRAM est locale à chaque processus.[52][53]

En d'autres termes, il n'y a aucune garantie sur l'ordre dans lequel les écritures sont vues par différents processus, bien que les écritures d'une seule source doivent conserver leur ordre comme si elles étaient dans un pipeline.

Le raisonnement qui a conduit à ce modèle était le suivant : considérons un multiprocesseur où chaque processeur possède une copie locale de la mémoire partagée. Pour que la mémoire soit évolutive, un accès doit être indépendant du temps nécessaire pour accéder aux mémoires des autres processeurs. Ils ont proposé que lors d'une lecture, une PRAM renverrait simplement la valeur stockée dans la copie locale de la mémoire. Lors d'une écriture, il mettrait d'abord à jour la copie locale et diffuserait la nouvelle valeur aux autres processeurs. En supposant un temps constant pour lancer une opération de diffusion, le but de rendre constant le coût d'une lecture ou d'une écriture est ainsi atteint. En termes de contraintes d'ordre, cela équivaut à exiger que tous les processeurs observent les écritures d'un seul processeur dans le même ordre alors qu'ils peuvent être en désaccord sur l'ordre des écritures par différents processeurs. L'historique d'exécution suivant est légal sous PRAM mais pas sous CC (cohérence causale) :

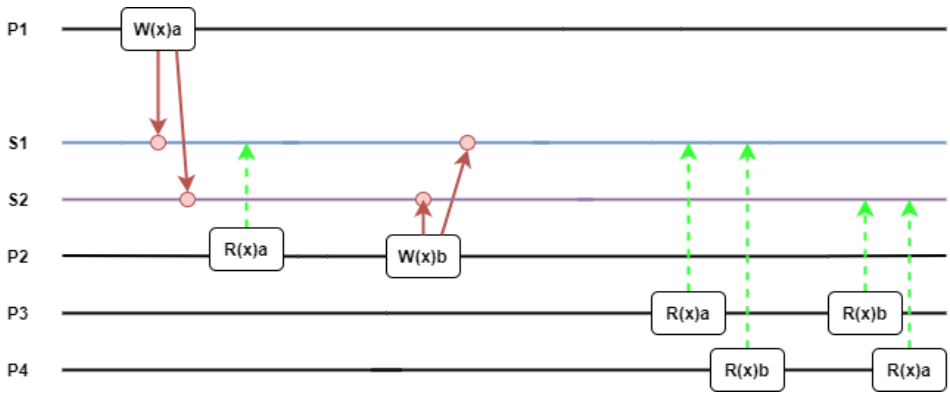

FIGURE 4.11 – Une séquence d'événements qui satisfait la cohérence PRAM.

P3 et P4 observent les écritures de P1 et P2 dans des ordres différents, bien que W(x)a et W(x)b soient potentiellement liés de manière causale. Ainsi, ce ne serait pas un historique légale pour CC.

5. Richard J. Lipton, naissance le 6 septembre 1946, est un chercheur anglo-américain en informatique reconnu notamment pour son travail en algorithmique et en cryptographie. Il a reçu le prix Knuth en 2014.

L'un des points importants de ce modèle de cohérence est son indépendance du temps dans la hiérarchisation des opérations. D'après ce qui est illustré sur la figure suivante, la séquence d'écriture des valeurs b et c par le processus P2 est d'abord l'opération d'écriture de b, puis l'opération sur le serveur Si. Le but est d'observer cette séquence entre l'ensemble des processus. Dans le cas comme le processus P4, cette séquence est différente avec l'observation des autres processus, alors il y aurait des violations dans le stockage des données en termes de cohérence FIFO.[53]

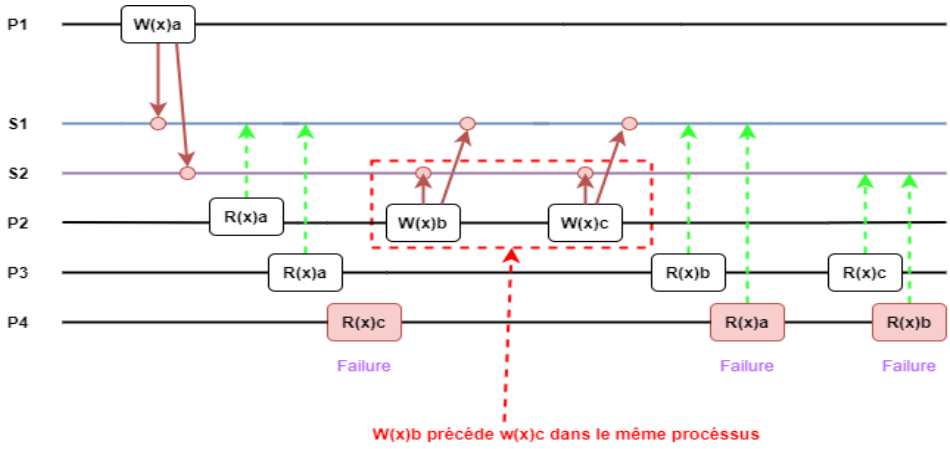

FIGURE 4.12 – Le comportement des processus sur les éléments de données en fonction de la cohérence FIFO.

Dans la **Figure 4.12** ci-dessus on remarque que le processus P4 viole cette cohérence PRAM car la lecture de P4 des deux écriture w(x)b ainsi que w(x)c du même processus P2 ont été lus différemment que leur ordre d'émission.

f. Cohérence faible

Ce modèle de cohérence est une autre variante des modèles data-centric, il a été proposés en 1986 par Micheal Dubois, L'idée de base derrière le modèle de cohérence faible consiste à appliquer la cohérence à un groupe d'opérations plutôt qu'à des opérations individuelles, cela évite la synchronisation globale. Pour ce faire, il définit une variable de synchronisation qui joue le rôle de token (jeton).

Le processus qui possède ce jeton peut effectuer les opérations de lecture ou d'écriture sur la ressource partagée. Dans ce modèle, l'accessibilité à la ressource se fait par la cohérence séquentielle au sein de la variable de synchronisation. Si le processus ne possède pas ce jeton, aucune des opérations de lecture ou d'écriture n'est privilégiée sur la ressource partagée. Ce modèle est établi dans les conditions suivantes :

- Les accès aux variables de synchronisation sont cohérents de manière séquentielle.
- Aucune opération sur une variable de synchronisation ne peut être effectuée tant que toutes les écritures précédentes n'ont pas été effectuées partout.
- Aucune opération de lecture ou d'écriture sur des éléments de données n'est autorisée tant que toutes les opérations précédentes sur les variables de synchronisation n'ont pas été effectuées.

Il faut notez que le sens de «précédent» est bien défini car il fait référence à l'ordre des programmes. C'est-à-dire qu'un accès précède l'accès B si et seulement si le processeur qui a exécuté l'accès B a précédemment exécuté l'accès A. La synchronisation des accès fonctionne comme des clôtures. Au moment où un accès de synchronisation est effectué, tous les accès antérieurs par ce processeur sont garantis comme ayant été effectués et tous les accès futurs par ce processeur sont garantis de ne pas avoir été effectués. Le modèle de synchronisation correspondant à ces contraintes d'ordre d'accès est relativement simple. Un programme s'exécutant sur un système faiblement cohérent apparaît séquentiellement cohérent si les deux contraintes suivantes sont observées [57] :

— Il n'y a pas d'accès concurrents.
— La synchronisation est visible par le système de mémoire.

Dans ce modèle de cohérence, si plusieurs processus veulent simplement effectuer l'opération de lecture de la mémoire partagée, alors tous peuvent avoir la variable de synchronisation à leur service. Cependant, si un processus veut effectuer l'opération d'écriture sur la mémoire partagée, alors jusqu'à la fin de l'opération d'écriture par le processus, les autres processus ne peuvent pas avoir la variable de synchronisation jusqu'à la fin de l'opération d'écriture.

plus précisément afin d'exécuter tout type d'opération, soit de lecture, soit d'écriture sur la réplique, la possession de la variable de synchronisation est nécessaire. Cette variable n'est accessible qu'après la fin de l'opération d'écriture par le processus ainsi la variable de synchronisation est libre.

Exemple :

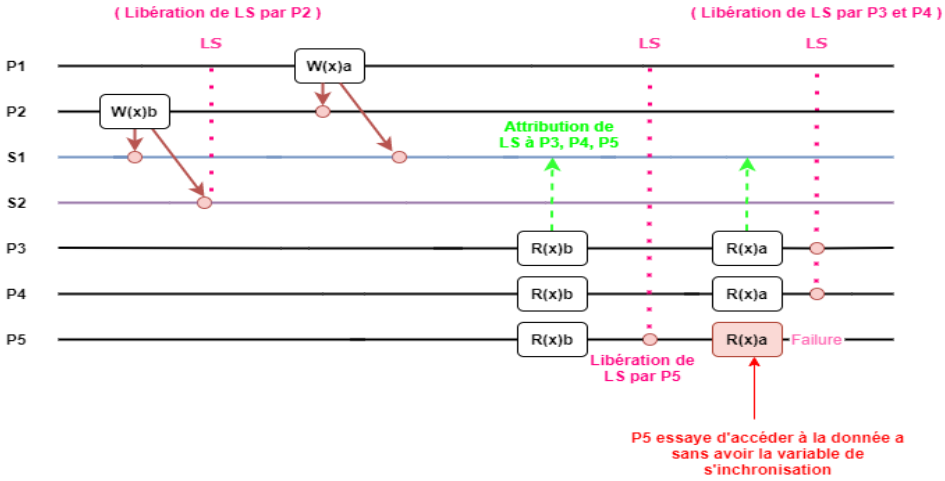

FIGURE 4.13 – Le comportement des processus sur les éléments de données en fonction de la cohérence faible.

La **Figure 4.13** nous montre un cas de violation de la cohérence faible, effectivement afin d'effectuer toute opération sur la mémoire, avoir la variable de synchronisation est nécessaire, la violation se produit dans le stockage de données si le processus n'a pas reçu cette dernière. Lorsque le processus demande au serveur S(i) de lire une valeur sans la variable de synchronisation, la valeur renvoyée peut ne pas être valide.

La **figure 4.13** illustre le comportement de ce modèle de cohérence. Les points violets sur la figure indiquent la libération de la variable de synchronisation après

la fin de l'opération sur la réplique existant dans le serveur S(i). Le point sur lequel cette figure se concentre est que la variable de synchronisation pendant l'opération de lecture est partagée entre plusieurs processus simultanément. Cependant, alors qu'un processus utilise la variable de synchronisation pour le processus d'écriture, aucun autre processus ne peut y avoir accès. Si un processus comme P5 libère la variable après l'opération de lecture, alors s'il veut lire a sans recevoir la variable, par la suite la valeur de lecture de a n'est pas valide et il y aurait violation en termes de cohérence faible dans la mémoire partagée.

g. **Cohérence à la libération**

La cohérence à la libération est l'un des modèles faibles centrés sur les données qui a été proposé en 1992 par K. Gharachorloo. Il assouplit le modèle de cohérence faible en distinguant l'opération de synchronisation d'entrée de l'opération de synchronisation de sortie. Dans un ordre faible, lorsqu'une opération de synchronisation doit être vue, toutes les opérations dans tous les processeurs doivent être visibles avant que l'opération de synchronisation ne soit effectuée et que le processeur continue. Cependant, dans le modèle de cohérence à la libération, lors de l'entrée dans une section critique, appelée «acquisition [6]», toutes les opérations relatives aux variables de mémoire locale doivent être terminées. Lors de la sortie, appelée "release [7]", toutes les modifications apportées par le processeur local doivent être propagées à tous les autres processeurs. La cohérence est toujours maintenue.[52][53][58][59]

Ce modèle de cohérence utilise un verrou ou une variable de synchronisation. Pour avoir accès à la variable de synchronisation, deux étapes doivent être pavées. Dans un premier temps, il est nécessaire de demander au verrou d'avoir accès à la mémoire. Par conséquent, le processus attend d'avoir accès à la réplique stockée dans la mémoire. Après réception du verrou symbolisé par L dans, le verrou est positionné sur toute la réplique en mémoire.[53]

Si le verrou n'est pas reçu par le processus lors de l'exécution, les résultats obtenus ne sont pas valides. Dans la deuxième étape, le verrou reçu par le processus qui l'a demandé est libéré et les valeurs mises à jour dans la réplique sont envoyées aux autres répliques dans d'autres serveurs afin de mettre à jour leurs opérations et leurs valeurs.

Le problème qui émerge de la weak consistency est que dans le temps d'accès à la variable de synchronisation, la mémoire partagée distribuée n'a aucune idée de l'opération (c'est-à-dire lecture ou écriture) sur la réplique. Ce problème a été résolu par la cohérence à la libération. Dans ce modèle, le type d'opération est déterminé en recevant le verrou et après sa libération. Ce modèle n'est pas garanti pour les systèmes géo-distribués à haute disponibilité.[58][59]

6. L'opération d'acquisition est un chargement / lecture effectué pour accéder à la section critique.
7. Une opération de libération est un stockage / écriture effectué pour permettre à d'autres processeurs d'utiliser les variables partagées.

exemple :

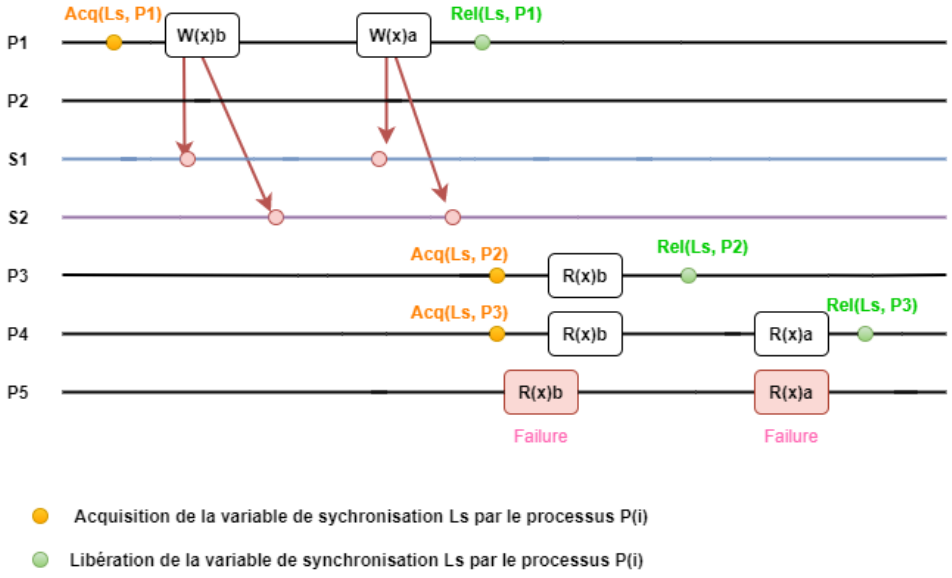

FIGURE 4.14 – Le comportement des processus sur les éléments de données en fonction de la cohérence de versions.

Si un processus prend la variable de synchronisation, alors son opération est valide. Par conséquent, en recevant la variable de synchronisation, le processus prend son réplica demandé et le met à jour. L'opération de lecture comme l'opération d'écriture doit recevoir la variable de synchronisation et en la maintenant et en exécutant l'opération de lecture, elle lit une valeur valide. Sinon, sans réception de la variable de synchronisation, la valeur écrite ou lue n'est pas valide. Les conditions dans lesquelles la cohérence de publication pourrait être exécutée sont les suivantes :

— Le processus doit réussir à placer le verrou sur la mémoire partagée avant d'effectuer les opérations de lecture ou d'écriture.
— Avant de libérer le verrou sur la mémoire, l'opération de lecture ou d'écriture doit être terminé par le processus qui détient le verrou.
— L'accessibilité à la variable de synchronisation doit être effectuée en utilisant le modèle de cohérence premier entré, premier sorti.

Dans ce modèle, le processus qui a reçu le verrou est également déterminé et après la fin de l'opération, il libère le verrou et les caractéristiques du processus sont également éliminées. Dans le cas où un processus tel que P4 ne reçoit pas la variable de synchronisation, la mémoire est confrontée à des violations en termes de cohérence de publication.[53]

h. Cohérence des paresseuse à la libération

Le défi dans la cohérence de versions est qu'après la fin de l'opération d'écriture et la libération de la variable de synchronisation, elle doit propager l'ensemble des modifications survenues sur les données de la réplique aux autres répliques de la mémoire.[52][53]

Cependant, si la mise à jour se produit dans toutes les répliques, il se peut que certaines répliques n'aient pas besoin de la mise à jour. Puis avec la propagation de

la mise à jour l'augmentation de la surcharge et par conséquent les performances de ce modèle ne serait pas efficace.

Le modèle de cohérence de lazy release est une extension de la cohérence release qui a été afin d'améliorer l'efficacité et l'optimalité du modèle de cohérence release. Dans ce modèle, la mise à jour n'a lieu dans l'autre réplica que lorsqu'elle doit vraiment être mise à jour. Si nécessaire, il enverra un message aux répliques dans lesquelles les données ont déjà été modifiées et mises à jour. Il est important de noter que le timestamp est d'une grande aide pour déterminer si les données sont obsolètes ou conformes à la dernière mise à jour.

Dans ce modèle de cohérence donc toutes les acquisitions en attente (par exemple, une opération de verrouillage) doivent être effectuées avant qu'une libération (par exemple, une opération de déverrouillage) ne soit effectuée. Les dépendances locales au sein du même processeur doivent toujours être respectées.[52]

Remarque : *"La cohérence de la libération est un assouplissement supplémentaire de la cohérence faible sans perte significative de cohérence."*

Plusieurs autres modèles de cohérence data-centric en étaient proposées comme le modèle de cohérence d'entrée (Entry consistency) qui est une variante des modèles de cohérence faible proposé en 1991 par Bershad, et le modèle de cohérence continue (Continous consistency) ...

4.1.5.2 Modèles de cohérence centrés sur le client

Dans les systèmes distribués, le maintien de la cohérence séquentielle afin de contrôler les opérations simultanées est essentiel. Un magasin de données réparties se caractérise par une relative absence de mises à jour simultanées. Le but est alors de maintenir une vue cohérente des éléments de données pour un processus client individuel qui fonctionne actuellement sur le magasin de données.

Les modèles de cohérence les plus utilisées et qui entrent dans cette catégorie sont :

- La cohérence éventuelle (eventual consistency)
- La cohérence de lecture monotone (monotonic reads consistency)
- La cohérence d'écriture monotone (monotonic writes consistency).
- La cohérence de lecture-écriture (read-your-writes consistency).
- La cohérence écritures aprés lectures (writes-follow-reads consistency).

Pour mieux comprendre par la suite ces modèles de cohérence :

- considérons un système de dénomination mondial tel que DNS. L'espace de noms DNS est partitionné en domaines, où chaque domaine est attribué à une autorité de dénomination, qui agit en tant que propriétaire de ce domaine. Seule cette autorité est autorisée à mettre à jour sa partie de l'espace de nom. Par conséquent, les conflits résultant de deux opérations qui souhaitent toutes deux effectuer une mise à jour sur les mêmes données (c'est-à-dire des conflits d'écriture / écriture) ne se produisent jamais. La seule situation qui doit être gérée est celle des conflits de lecture-écriture, dans lesquels un processus souhaite mettre à jour un élément de données tandis qu'un autre tente simultanément de lire cet élément. En fin de compte, il est souvent acceptable de propager une mise à jour de manière paresseuse, ce qui signifie qu'un processus de lecture ne verra une mise à jour qu'après un certain temps depuis la mise à jour.

— Le World Wide Web est un autre exemple. Dans pratiquement tous les cas, les pages Web sont mises à jour par une seule autorité, telle qu'un webmaster ou le propriétaire réel de la page. Il n'y a normalement aucun conflit d'écriture / écriture à résoudre. D'autre part, pour améliorer l'efficacité, les navigateurs et les proxys Web sont souvent configurés pour conserver une page récupérée dans un cache local et pour renvoyer cette page à la demande suivante.
— Un aspect important des deux types de caches Web est qu'ils peuvent renvoyer des pages Web obsolètes. En d'autres termes, la page mise en cache qui est renvoyée au client demandeur est une version plus ancienne par rapport à celle disponible sur le serveur Web réel. En fait, de nombreux utilisateurs trouvent cette incohérence acceptable (dans une certaine mesure).

Ces exemples peuvent être considérés comme des cas de bases de données distribuées et répliquées (à grande échelle) qui tolèrent un degré relativement élevé d'incohérence. Ils ont en commun que si aucune mise à jour n'a lieu pendant une longue période, toutes les répliques deviendront progressivement cohérentes. Cette forme de cohérence est appelée cohérence éventuelle.

Les magasins de données qui sont finalement cohérents ont donc la propriété qu'en l'absence de mises à jour, tous les répliquas convergent vers des copies identiques les uns des autres. La cohérence finale exige essentiellement que les mises à jour soient garanties de se propager à tous les répliquas. Les conflits d'écriture-écriture sont souvent relativement faciles à résoudre si l'on suppose que seul un petit groupe de processus peut effectuer des mises à jour. La cohérence finale est donc souvent peu coûteuse à mettre en œuvre.

Les éventuels magasins de données cohérents fonctionnent tant que les clients accèdent toujours à la même réplique. Cependant, des problèmes surviennent lorsque différentes répliques sont accessibles sur une courte période de temps. Ceci est mieux illustré en considérant qu'un utilisateur mobile accède à une base de données distribuée, comme le montre la figure suivante :

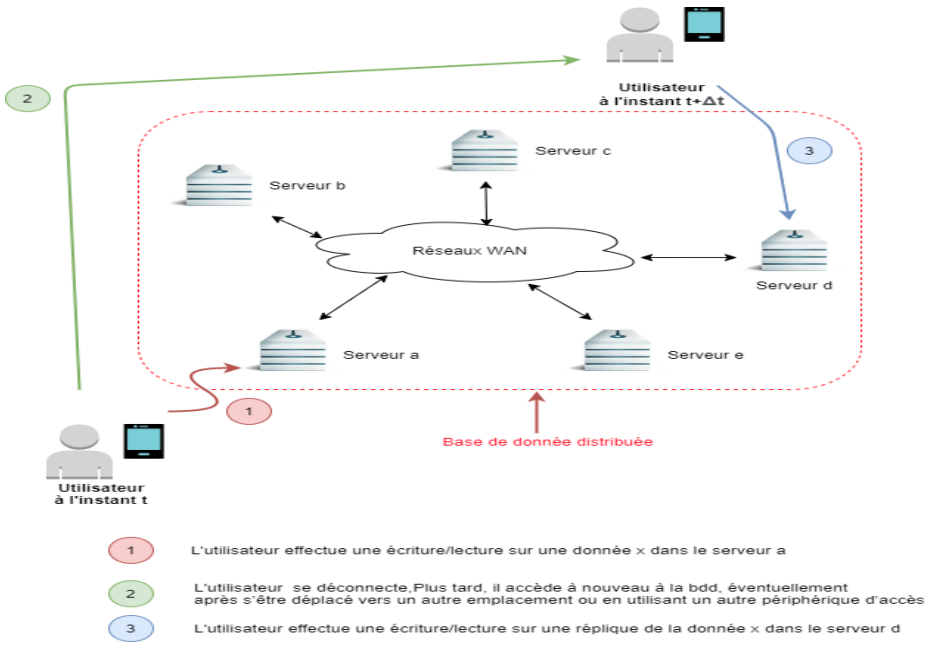

FIGURE 4.15 – Le principe d'un utilisateur mobile accédant à différentes répliques d'une base de données distribuée.

L'utilisateur mobile accède à la base de données en se connectant à l'une des répliques de manière transparente. En d'autres termes, l'application exécutée sur l'ordinateur portable de l'utilisateur ne sait pas sur quelle réplique elle fonctionne réellement. Supposons que l'utilisateur effectue plusieurs opérations de mise à jour, puis se déconnecte à nouveau. Plus tard, il accède à nouveau à la base de données, éventuellement après s'être déplacé vers un autre emplacement ou en utilisant un autre périphérique d'accès. À ce stade, l'utilisateur peut être connecté à une réplique différente de celle d'avant, comme le montre la figure.

Cependant, si les mises à jour effectuées précédemment n'ont pas encore été propagées, l'utilisateur remarquera un comportement incohérent. En particulier, il s'attendrait à voir toutes les modifications apportées précédemment, mais à la place, il semble que rien du tout ne s'est produit.

Cet exemple est typique des magasins de données cohérents à terme et est dû au fait que les utilisateurs peuvent parfois opérer sur des répliquas différents. Le problème peut être atténué en introduisant une cohérence centrée sur le client. En substance, la cohérence centrée sur le client fournit des garanties pour un seul client concernant la cohérence des accès à un magasin de données par ce client. Aucune garantie n'est donnée concernant les accès simultanés par différents clients.

a. **Cohérence éventuelle**

Un grand nombre d'incohérences pourraient être négligées grâce à ce modèle de manière relativement peu coûteuse. Le niveau de cohérence entre les processus et leur confiance est variable dans ce modèle. La plupart des processus préforment à peine la mise à jour sur la réplique, c'est pourquoi un petit nombre de processus préforme la mise à jour. Sur cette base, le seul cas qui doit être analysé plus fréquemment est celui des conflits lecture / écriture où un processus qui veut mettre à jour une donnée alors qu'un autre processus veut lire la même donnée simultanément [53][60].

Dans ce cas, la mise à jour par ce modèle est préformée par un mode paresseux comme le modèle Lazy release consistency. À cet égard, le système tolère un niveau élevé d'incohérence. S'il n'y a pas de nouvelles mises à jour après une longue période, le système ou les répliques deviendront progressivement cohérents [53][55].

La cohérence éventuelle vient renforcer le modèle de cohérence faible (Weak Consistency) . Les réplicas ont tendance à converger vers le même état de données. Pendant l'exécution de ce processus de convergence, il est possible pour les opérations de lecture de récupérer une version plus ancienne au lieu de la dernière. La fenêtre d'incohérence dépendra des délais de communication entre les répliques et ses sources, la charge sur le système et le nombre de répliques impliquées.[52][60]

Ce modèle est à mi-chemin entre un modèle de cohérence forte et un modèle de cohérence faible. La cohérence éventuelle est une fonctionnalité populaire offerte par de nombreuses bases de données NoSQL. Cassandra est l'un d'entre eux, et il peut offrir une disponibilité et une partition du réseau à un niveau tel qu'il ne compromet pas la convivialité des sites Web les plus consultés au monde qui utilisent Cassandra. L'un d'eux est Facebook, la société qui a initialement développé Cassandra.[50]

Exemple : *Amazon Dynamo choisit la cohérence éventuelle dans laquelle*

chaque élément de données dans les réplicas est synchronisé progressivement et diminue la surcharge de synchronisation. En cas d'absence des nouvelles mises à jour pour des données spécifiques, alors ils recevront des mises à jour de l'endroit le plus proche où les données sont accessibles, la dernière mise à jour est lue. Cependant, l'opération de lecture sur plusieurs objets peut renvoyer une combinaison d'anciennes et de nouvelles valeurs[53]

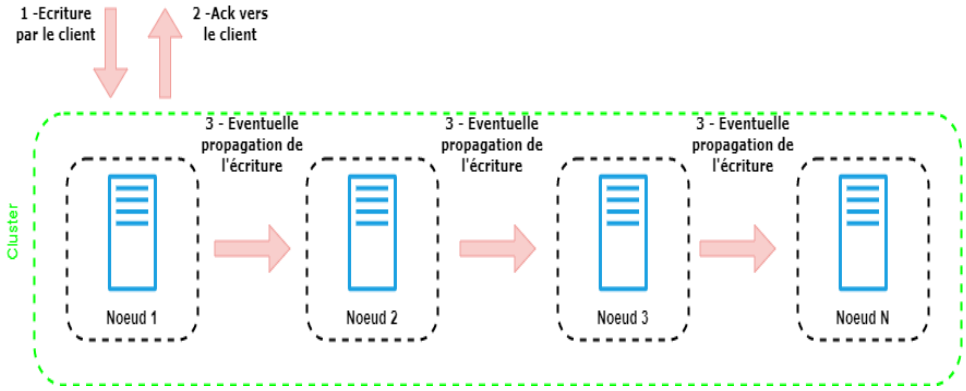

FIGURE 4.16 – Système éventuellement cohérent.

à travers la **Figure 4.16** nous pouvons observer que lors d'une écriture(mise à jour) au niveau d'un système distribuée éventuellement cohérent au niveau d'un noeud (serveur), cette écriture est synchroniser juste après ou bien après un temps donnée vers toutes les autres répliquent éventuellement.

Pour résumer un tel système :

- *Le système retourne finalement la dernière écriture : affaiblit les conditions de cohérence en ajoutant le mot «éventuellement».*
- *Risque de perte de données en cas de défaillance du nœud : ajoute la condition «à condition qu'il n'y ait pas de panne permanente».*

Pourquoi une cohérence éventuelle et pas une cohérence strict ?

Une cohérence stricte n'est pas toujours requise et une cohérence éventuelle peut suffire dans certains cas d'utilisation. Par exemple, dans un panier, disons qu'un article est ajouté et que le centre de données a échoué. Ce n'est pas un désastre pour le client d'ajouter à nouveau cet article. Dans ce cas, une cohérence éventuelle serait suffisante.

Cependant, vous ne voudriez pas que cela se produise sur votre compte bancaire avec un dépôt que vous venez d'effectuer. Faire disparaître votre argent parce qu'un nœud a échoué est inacceptable. Une stricte cohérence est requise dans les transactions financières.

Pourquoi une cohérence éventuelle par rapport aux autres modèles :

- Plus d'opportunités d'accès concurrentiel qu'une cohérence stricte, séquentielle ou causale
- La cohérence séquentielle nécessite des connexions hautement disponibles (Beaucoup de bavardages entre clients / serveurs).
- La cohérence séquentielle n'est pas adaptée à certains scénarios

    1- Clients déconnectés (par exemple, votre ordinateur portable se déconnecte, mais vous souhaitez toujours modifier votre document partagé).

    2- Partitionnement du réseau entre les centres de données.

    3- Les applications peuvent préférer une incohérence potentielle à une perte de disponibilité.

b. **Cohérence de lecture monotone**

Lorsqu'un réplica nécessite la cohérence des données, les dernières modifications apportées à l'élément de données sont envoyées à la copie exigeante par le réplica. Le modèle de cohérence garantit que si un processus observe une valeur dans un certain temps, il ne verra jamais sa valeur précédente.

Par exemple, un utilisateur peut lire le courrier entrant tout en se déplaçant. Chaque fois que l'utilisateur se connecte à un serveur de messagerie différent, ce serveur récupère toutes les mises à jour du serveur que l'utilisateur a précédemment visité. Monotonic Reads garantit que l'utilisateur voit toutes les mises à jour, quel que soit le serveur à partir duquel la lecture automatique a lieu.[53][60]

Ces opérations de lecture séquentielle reflètent un grand nombre d'opérations d'écriture . Si l'opération de lecture séquentielle est effectuée par un client, le contenu mis à jour sera renvoyé de manière significative.
Dans le modèle de cohérence de lecture monotone, si le client a lu une valeur de la mémoire, alors il ne verrait aucune autre valeur qui a été écrite avant cela. En d'autres termes, en effectuant l'opération de lecture par le client, une valeur valide est lue à partir de la réplique sur laquelle toutes les modifications antérieures ont été affectées. Il en résulte que l'opération de lecture est valide lorsque le client en ayant accès à la réplique existant dans le serveur a exécuté toutes les modifications sur la réplique d'objet. Sinon, il fera face à des violations dans la mémoire partagée en utilisant la cohérence de lecture monotone. Pour mieux comprendre ce modèle, son comportement est illustré sur la **figure 4.17** suivante.[53]

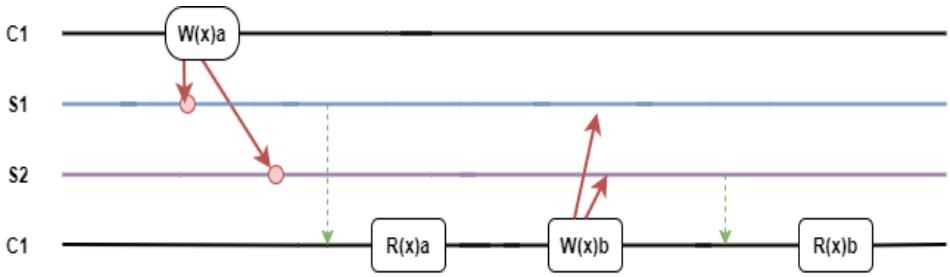

FIGURE 4.17 – Le comportement des processus sur les éléments de données basé sur la cohérence de lecture monotone.

Le comportement du modèle de cohérence de lecture monotone sur la mémoire partagée est introduit comme le client Ci, peut lire la valeur valide b, quand en ayant accès à chacune des répliques dans les serveurs S1 et S2 toutes les opérations antérieures ont été exécutées sur le réplique.[53]

Pour résumer la lecture monotone reflète les lectures continues de l'ensemble des opérations d'écriture. Si la lecture continue est effectuée par un client, la mise à jour sera effectuée par le même client. La lecture de votre écriture nécessite que chaque opération d'écriture soit visible dans l'ordre d'exécution. En d'autres termes, chaque opération de lecture reflète le résultat de son opération d'écriture précédente.

c. **Cohérence d'écriture monotone**

L'opération d'écriture par le processus sur l'élément de données est acceptable lorsque l'opération d'écriture précédente est exécutée par le même processus sur la réplique. Comme la bibliothèque du logiciel qui doit remplacer une ou plusieurs fonctions pour être mise à jour. Le point important est que dans la cohérence d'écriture monotone, les opérations d'écriture sont effectuées avec la même séquence qui a été lancée. L'opération d'écriture par un client sur une réplique est assurée lorsque l'ensemble des opérations d'écriture précédentes sont enregistrées par le même client sur la même réplique. Ce modèle de cohérence propage l'opération d'écriture par rapport aux priorités entre les opérations. En fin de compte, il faut que chaque opération d'écriture soit visible dans l'ordre de présentation.[53][60]

Le comportement de la cohérence d'écriture monotone est exprimé comme suit : l'opération d'écriture 1 s'exécute correctement lorsque l'opération d'écriture de 2 est effectuée avant cela. En d'autres termes, lorsque le client écrit la valeur valide 1 sur la réplique que l'opération 2 sur la même réplique dans le serveur est effectuée avant cela .[53]

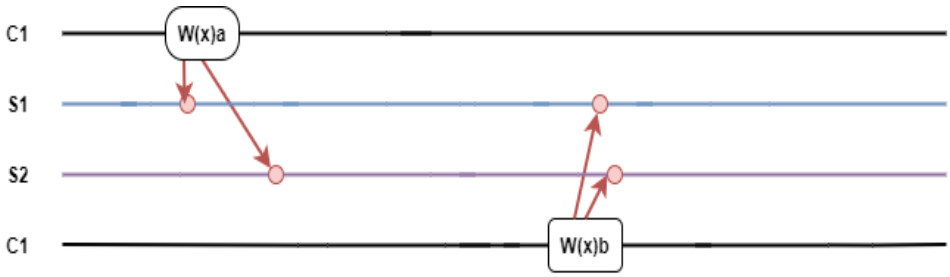

FIGURE 4.18 – Le comportement des processus sur les éléments de données en fonction de la cohérence d'écriture monotone.

Ce modèle de cohérence qui est introduit sur la **Figure 4.18** exprime que le client Ci effectue l'opération d'écriture de la valeur valide b sur l'élément de données x si toutes les opérations d'écriture précédentes sur cet élément sont exécutées sur le serveur Si.

En résumé en écriture monotone , l'opération d'écriture d'un client sur une réplique est garantie une fois que toutes les opérations d'écriture précédentes de ce client sur la même réplique sont enregistrées.

d. **Cohérence de lecture-écriture**

Dans ce modèle, l'opération d'écriture est effectuée par le client sur l'élément de données x qui est toujours visible lors des opérations de lecture successives pour le même client sur l'élément de données x. Cela signifie que les opérations d'écriture sont terminées avant que la prochaine opération de lecture ne soit effectuée par le même client. Ce modèle de cohérence pourrait être spécifié comme suit :[53][60]

— Le temps d'accès aux données est prolongé (comme le changement de mot de passe).
— Similaire à la cohérence en lecture seule, mais avec la différence que la cohérence de la dernière opération de lecture est déterminée par la dernière opération d'écriture du client.

Dans ce modèle, la nouvelle valeur écrite est lue par le client au lieu de la précédente valeur. Ce modèle nécessite que chaque opération d'écriture soit visible dans l'ordre d'exécution. Toute séquence de transactions telle que la lecture non validée est basée sur la priorité spécifiée par l'observateur global, et l'opération de lecture reflète ses opérations d'écriture précédentes. La mise à jour de la page Web peut être désignée comme l'une des applications de ce modèle, elle garantie que le navigateur Web affiche la version la plus récente au lieu de sa copie en cache. Afin d'avoir une meilleure compréhension de ce modèle, la **figure 4.19** illustre le fonctionnement de ce modèle.[53][60]

Le modèle de cohérence de lecture-écriture est exprimé comme suit : le client Ci peut lire une valeur valide b lorsque l'opération d'écriture a est préalablement exécutée sur le réplica existant dans le serveur Sj. Où b est la valeur écrite valide par le client Ci. Afin d'avoir une meilleure compréhension de ce modèle, la **figure 4.19** illustre le fonctionnement de ce modèle.[53]

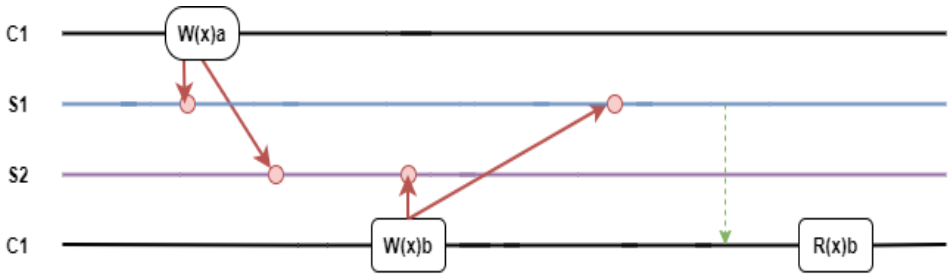

FIGURE 4.19 – Le comportement des processus sur les éléments de données en fonction de la cohérence de lecture de votre écriture.

Le comportement de ce modèle selon la **figure 4.19** est tel que le client peut lire la nouvelle valeur b comme résultat valide, avant l'exécution de l'opération d'écriture b par le client C1 sur la donnée du serveur Si, tous écrivent les opérations sur cet élément de données particulier sont exécutées dans la version existante du serveur. Sinon, il rencontrera une incohérence dans le référentiel de données partagé.

e. **Cohérence écritures après lectures**

Dans la cohérence Writes-follow-reads, également connue sous le nom de session causale, les propagations de mise à jour sont basées sur la dernière opération de lecture. Le client peut écrire sur l'élément de données x, lorsque la dernière valeur de l'élément de données x est lue par le même client. Par exemple, sur Twitter, un client peut publier un retweet sur un message qu'il / elle a déjà vu. Avec l'opération de lecture, ce modèle vérifie l'opération d'écriture de dictée précédente du client et assure la nouvelle opération d'écriture. Ce modèle est lié à la relation «happening-before» qui est proposée par Lamport.[53][60]

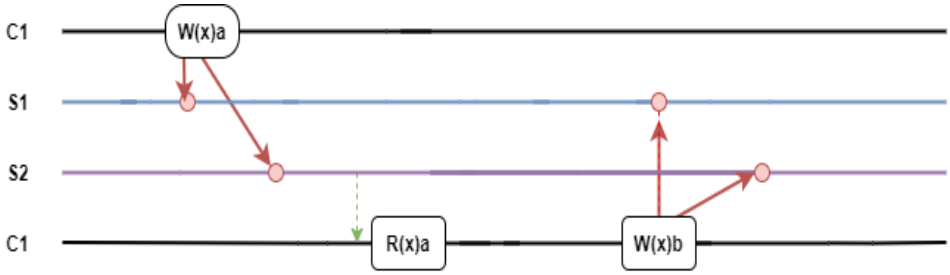

FIGURE 4.20 – Le comportement des processus sur les éléments de données en fonction de l'écriture suit la cohérence de lecture.

Comme on peut le voir sur la **figure 4.20**, si le client Ci lit la valeur a de la donnée x sur le serveur Sj, alors cette valeur est écrite sur le même serveur puis écrit la nouvelle valeur b sur la donnée x sur serveur Sj et par conséquent la lecture de suivi d'écriture fonctionne correctement.

Nouveaux modèles de cohérences

Au fil des années, les éléments essentiels des systèmes distribués, en particulier les environnements cloud, ont changé. Les nécessités telles que la sécurité et la fiabilité,et la nécessité d'une plus grande convergence des opérations, sont devenues apparentes dans de tels systèmes. Dans cette optique, les chercheurs ont introduit de nouveaux modèles de cohérence pour répondre à ces besoins et défis dans les systèmes distribués.[53]

1. **Cohérence Fork**
   Il s'agit d'un modèle de cohérence centré sur le client, qui exprime le concept le plus fort de l'intégrité des données sans la présence de serveurs en ligne et de clients fiables.
   Les conditions de cohérence fork sont souvent encapsulées dans la notion de "Byzantine emulation"[64].

   > **Remarque :** *" On dit qu'un processus p fait une erreur byzantine s'il se ne comporte pas selon le protocole. Un tel processus est appelé processus byzantin. En particulier les fautes byzantines sont utilisées pour modéliser des comportements malicieux à partir desquels un processus va tenter de faire échouer un calcul."*

   Si le service empêche un client d'afficher la mise à jour d'un autre client, les deux clients ne verront jamais les informations de mise à jour l'un de l'autre. Ce modèle a été introduit à l'origine pour les systèmes de fichiers qui dissimulent les opérations des clients les uns aux autres. Selon ce modèle, si l'on divise les clients en deux groupes, chacun de ces groupes ne peut pas voir les opérations de l'autre et vice versa.

   La cohérence fork augmente la concurrence de l'opération sur les données partagées dans les systèmes distribués et est introduite pour protéger les informations client contre les systèmes malveillants.

   Plusieurs autre modèle découle de la cohérence fork, par exemple la cohérence séquentielle fork (FSC) et Fork Linearize Consistency (FLC) qui sont des modèles de cohérence hybride.[65]

2. **Cohérence de vue**
   Dans un système à distribuée, la politique de propagation de l'information est formée à l'avance pour avoir une connaissance approfondie de la structure complète de chaque arbre de preuve. Ce modèle garantit la cohérence dans laquelle les données sont cryptées par les autorités de certification. En utilisant ce modèle, les contraintes de cohérence pourraient être exécutées dans les systèmes d'arbres de preuve. Dans ce modèle, il est peu probable de montrer complètement les détails de la preuve pour l'émission de privilèges et de preuves. Il est probable que ce type de système distribué soit similaire à celui utilisé en informatique et dans l'environnement réseau des capteurs. Ce modèle fait partie d'une variété de modèles de cohérence basés sur les données.Le système avec la cohérence de vue serait cohérent si et seulement si le système a une vue valide de l'état des données stockées à des intervalles spécifiés. Cependant, trois niveaux de cohérence de vue liés aux protocoles de distribution éprouvés sont décrits pour être appliqués dans le système [53] :
   — Cohérence incrémentielle : elle est fréquemment utilisée que les autres niveaux de cohérence. Elle dit que toute opération lors de la construction d'un arbre de

preuve associé est valide à certains points. En fait, les protocoles de construction distribués existants utilisent la cohérence de la vue incrémentielle lors de la prise de décision concernant les privilèges. Ce phénomène conduit à diverses violations de la sécurité. Par conséquent, la cohérence des incréments n'est pas garantie en raison du chevauchement des intervalles de validation stockés dans le système.

— Cohérence de la requête : le niveau suivant de la cohérence de la vue est la cohérence de la requête où toutes les opérations utilisées pour créer la preuve distribuée sont valides lorsque la requête de déclenchement de la création de la preuve est effectuée simultanément. Si la politique de privilège est satisfaite de l'utilisation de la cohérence de vue de requête, cette politique est satisfaite dans l'environnement de création de la cohérence de preuve distribuée en utilisant un cadre de preuve centralisé pour prendre en charge les évaluations de transaction.
— - Cohérence d'intervalle : un autre niveau de cohérence de la vue est la cohérence d'intervalle, dans laquelle le fonctionnement du modèle est complètement précis si et seulement si l'état de chaque opération multiple qui consiste en la validité de deux intervalles de temps précis est chiffré.

3. **Cohérence VFC3**

De nos jours, la dépendance aux données stockées dans les centres de données cloud a considérablement augmenté dans le monde entier. Différents protocoles de réplication sont appliqués afin d'obtenir une accessibilité et des performances élevées et garantir la cohérence entre les répliques. En utilisant les modèles traditionnels, les performances de la cohérence peuvent se détériorer. Par conséquent, la plupart des centres de données à grande échelle prennent en compte la déclinaison de cohérence provoquée par la diminution du délai pour les utilisateurs finaux. En règle générale, le niveau de cohérence est réduit par les systèmes basés sur le cloud qui donnent des privilèges aux données au hasard pour rester en mémoire pendant des périodes de temps constantes. De plus, le comportement de tels systèmes entraîne une ignorance de la sémantique des données. Ce comportement, la nécessité de combiner les niveaux forts et faibles de cohérence de vue se fait sentir complètement.[53]

Le modèle de cohérence VFC3 a été spécialement conçu pour répondre aux environnements dynamiques et à très grande échelle , qui doivent synchroniser de grandes quantités de données entre plusieurs points géographiquement dispersés, tout en maintenant de fortes exigences concernant la qualité du service et des données fournie , c' est donc afin d'avoir une meilleure cohérence dans le traitement de l'accessibilité, que le modèle de cohérence VFC3 a été proposé.[66]

Ce nouveau modèle de cohérence est utilisé pour répliquer les données dans les centres de données dans le cadre de la bibliothèque afin d'augmenter le niveau de cohérence et est basé sur le vecteur tridimensionnel du temps, de la séquence et de la valeur liés aux objets de données. Chacune des dimensions est un scalaire qui montre le niveau maximum de discrétisation à partir des limitations de la cohérence. En considérant les dimensions suivantes, ce modèle offre la cohérence :[53]

— Dimension temporelle : indique la durée maximale pendant laquelle une réplique ayant la dernière valeur ne peut pas être mise à jour.
— Dimension de séquence : affiche la fréquence de mise à jour la plus élevée qui peut être accordée pour un objet sans tenir compte des répliques.
— Dimension de valeur : représente la différence proportionnelle maximale entre

le contenu de données de la réplique à une valeur constante.

4. **Cohérence adaptable**

   La cohérence adaptable ou rationnelle est exécutée dès que les éléments de données sont regroupés proportionnellement à leur importance, par exemple, dans les magasins en ligne, les cartes de crédit utilisent ce modèle de cohérence sur différents types d'éléments de données, c'est-à-dire A, B , C.

   Bien que les éléments de données A et C appliquent de manière réceptive les modèles de cohérence linéaire et éventuelle, l'élément de données B calcule les alternances continues de cette cohérence sur la base de la fonction de coût d'incohérence. Chaque fois que le coût de l'incohérence dépasse le coût de l'inaccessibilité ou de la latence élevée, la cohérence linéaire est effectuée sur l'élément de données B . Par exemple, la cohérence de rationnement basée sur le cloud est exécutée via la bibliothèque GARF.

   Ce modèle de cohérence avec auto-adaptabilité dans l'environnement cloud est sélectionné en fonction du coût de cohérence, il permet au client de spécifier le taux de lecture périmé maximal ou le coût de cohérence conformément au SLA . Ce modèle basé sur le type de coût et de données présente différents niveaux de cohérence. La cohérence RedBlue comme la cohérence adaptable fournit deux niveaux de cohérence discriminants basés sur le type d'opérations.[53]

5. **Cohérence RedBlue**

   La cohérence ReadBlue est présentée pour augmenter la vitesse de réplication dans les systèmes distribués. En d'autres termes, l'augmentation de la vitesse suggère que chaque fois que le client envoie une demande au serveur, il reçoit sa réponse dans un court laps de temps. La cohérence éventuelle, en réduisant la synchronisation entre les nœuds ou sites, traite les opérations locales avec une vitesse plus rapide. En revanche, les consistances linéaire et séquentielle car elles ont des taux de communication élevés dans leur processus de synchronisation entre les nœuds, elles interdisent l'agilité dans le traitement des opérations.

   La cohérence RedBlue catégorise les opérations en fonction du type d'exécution en deux séries d'opérations rouges et bleues. L'ordre d'exécution des opérations bleues peut être différent d'un site à l'autre, alors que les opérations rouges doivent être exécutées de la même manière pour tous les sites. Ce modèle de cohérence se compose de deux parties : [53]

   — L'ordre de RedBlue qui spécifie l'ordre des opérations.
   — Et une série d'opérations sérialisables locales qui ont relation de causalité entre eux.

   La relation de causalité dans ce modèle enregistre ces réalités dans d'autres sites en s'assurant que la dépendance des opérations est enregistrée dans le site principal et les garantit.

   En définition, la séquence des opérations dans des sites spécifiques est traitée localement. Dans un tel système, chaque opération avec une étiquette rouge est exécutée dans un ordre sérialisable tandis que chaque opération étiquetée en bleu est exécutée comme la cohérence éventuelle. Ces étiquettes spécifient la catégorie d'opération. L'exécution de ce modèle s'assure de l'absence d'incohérence dans les attributs de l'application. Enfin, toutes les répliques sont convergées. L'opération dans ce modèle n'a aucun effet sur le remplacement des opérations bleues. Sur cette base, un procédé est introduit afin d'augmenter l'espace réalisable pour

exécuter les opérations bleues en les divisant en deux phases de fonctionnement générateur et ombre.

Le générateur d'opération trouve simplement les alternances dans l'opération principale. Dans la phase de reconnaissance, les opérations bleues et rouges sont identifiées. Cependant, dans les opérations d'ombrage les alternances identifiées sont exécutées et sont répliquées sur tous les sites et les opérations sont l'ombrage, bleu ou rouge exclusivement.

Conclusion

Dans ce chapitre ont nos avons détailler les différents modèles présents dans la littérature afin d'assurer la cohérence et/ou la disponibilité des données dans le cadre d'un environnement distribuée.

FIGURE 4.21 – Résumer des modèles de cohérences.

Assurer la cohérence de données dans un cadre répartie est devenu un sujets de recherche de plus en plus important ces dernières années. Ce sujet présente de nombreux défie liée car dans un tel environnements le système doit être capable de garantir un

état cohérents pour toute les répliques. En d'autres termes il faut que ces systèmes soit capable de fournir un ensemble de stratégie et de méthodes pour la propagation et la mise à jour des données dans un environnement distribuée afin d'être sûr que quand une copie est mise à jours toutes les autres copies seront misent à jour également.

Ce chapitre n'épuise pas tout les modèles liée a la cohérence des donnée dans les systèmes réparties, plusieurs modèles cohérence sont on développement constant, les développeur essaye de voir une approche qui consiste à fusionner deux ou trois modèles de cohérences déjà présent dans le but d'assurer de plus en plus de cohérence et disponibilité de données dans les systèmes cloud distribué, on appel ce mélange de modèles des modèles hybrides, nous citons par exemple :

— Linéarisabilité et éventuelle.
— Fork et séquentielle.
— Fork et linéarisable.
— Fork et linéarisable et causale.
— Monotonic read er read-your-write et causale.

Dans le ce mémoire nous avons choisie de faire une études comparative entre les principaux modèles de cohérence, celui-ci fera l'objet du chapitre 5 afin d'avoir une idée plus précise de ces modèles là de cohérences.

Chapitre 5

Étude comparative entre les modèles data-centric et client centric

Dans les systèmes distribués afin de réduire le temps d'accès, la mise en cache des données est utilisée. L'effet de la réplication et de la mise en cache augmente la complexité et la surcharge de la gestion de la cohérence. Différents modèles de cohérence offrent différents degrés de cohérence. En fonction de l'ordre des opérations autorisé et du degré d'incohérence pouvant être toléré, les modèles de cohérence vont de fort à faible.

Dans ce chapitre on va comparer dans un premier temps les modèles data-centric entre eux puis les méthodes client-centric en eux, et juste après en fera une étude comparative entre ces deux catégories.

5.1 Étude comparative entre les modèles data-centric

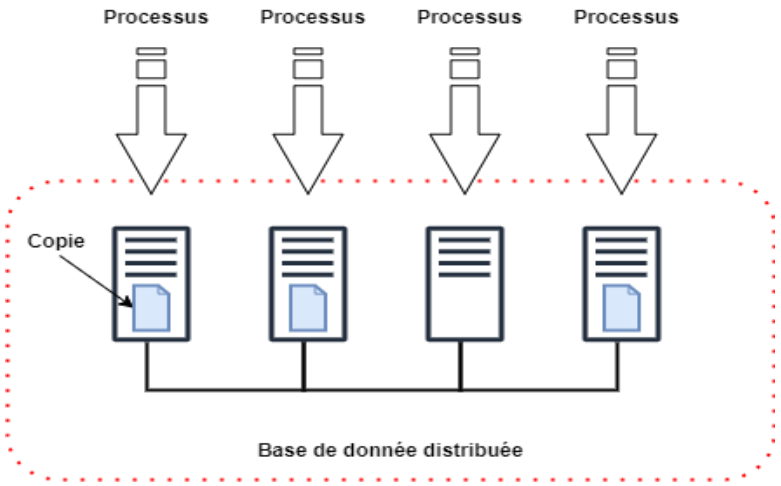

FIGURE 5.1 – Modèle centré sur les données.

Dans ces modèle, il est garanti que les résultats des opérations de lecture et d'écriture effectuées sur les données peuvent être répliqués immédiatement dans divers répliquent.[61]

Au niveau de la cohérence strict **(Strict consistency)** on utilise des mécanismes basés sur l'horloge pour contrôler les horodatages, de telles méthodes offrent la garantie que les objets arbitraires du magasin de données sont accessibles de manière atomique et isolés des accès simultanés. L'approche derrière cette méthode de cohérence est basée

sur la capacité d'un système à fournir un journal d'horodatage pour suivre l'ordre dans lequel les opérations se produisent. Selon Bravo et al, ce type de technique est mis en œuvre par les systèmes de stockage de données eux-mêmes et pourrait avoir un impact sur les garanties de cohérence qu'ils offrent.[10]

Ce type de méthodes est tolérant au partitionnement réseau et met l'accent beaucoup plus sur la disponibilité des données en contrôlant la latence de lecture et d'écriture, un des outils qui met à l'œuvre tout ça est Spanner [1]. Clock-SI est un autre outil qui respecte la cohérence strict qui prend en charge l'évolutivité, apportant des avantages en termes de performances. Il évite également un « point de défaillance unique » et, par conséquent, un goulot d'étranglement potentiel des performances.

La Cohérence séquentielle quand à elle c'est une variante de la cohérence strict **(Strict consistency)** la contrainte c'est que les opérations du même processus doivent êtres exécutées dans l'ordre où le système les a reçues, ce type de cohérence est basées sur le séquençage d'événements vise à masquer la complexité de la réplication en se basant sur le fait que la sérialisabilité des transactions est coûteuse et souvent inutile dans les applications Web. Ainsi, ils fournissent une solution simple, mais dans de nombreuses situations, efficace de garantie de cohérence.

L'un des outils qui met en œuvre ce type de cohérence c'est PNUTS [2].
Ce type de modèle offre ainsi un haut degré de disponibilité à leurs utilisateurs qui doivent être capables de lire les données en présence de pannes en répliquant les données sur plusieurs serveurs offrant ainsi plus de fiabilité et de tolérance aux pannes malgré les partitions réseau.

En ce qui concerne la cohérence linéarisable **(Linearizability model)** quand à elle c'est le modèle de cohérence séquentielles ajoutant des opérations ordonnées selon un temps global logique, les opérations ne vérifient pas que que les opérations sont sont séquentielles dans un même processus seulement mais les opérations doivent être séquentielles pour toutes les opérations de tous les processus. Offrant ainsi des mises à jour rapides et toujours disponibles. L'un des outils qui utilise cette cohérence est GEO [67], ce dernier améliore les performances en mettant en cache les états des acteurs dans un ou plusieurs centres de données. Dans le cas où un utilisateur demande la linéarisation, GEO garantit une véritable linéarisation en temps réel, entre l'appel et le retour.[10]

Quant à la cohérence causale **(Causal Consistency)** proposée pour les systèmes distribués basés sur les relations survenues avant par Lamport . Ce modèle est proposé pour faire un compromis entre la cohérence et la disponibilité des données. Certains modèles centrés sur les données tels que les modèles linéaires et séquentiels qui sont plus forts que le causal ne sont pas applicables dans la pratique car les répliques sont toujours sujettes au changement et aucune convergence intense ne se produit entre elles. La fonction de convergence spécifie que toutes les répliques doivent être dans le même état. Cependant, la cohérence causale leur permet d'être temporairement divergentes.

La cohérence causale permet non seulement de tolérer le partitionnement du réseau avec un faible degré de convergence, mais garantit également la cohérence et la disponibilité des données avec la plus grande satisfaction. De plus, la cohérence causale peut être considérée comme la cohérence la plus forte qui est une combinaison d'arbitrage causal et de visibilité causale.

1. C'est un service de base de données NewSQL évolutif et distribué à l'échelle mondiale, conçu, construit et déployé par Google
2. Un SGBD massivement parallèle et géographiquement distribué développé par Yahoo

De plus, le causal a une faible latence du réseau lors du partitionnement dans le réseau. Puisque dans ce modèle le temps physique n'est pas pris en compte et n'est pas capable de fournir une convergence par lui-même. Par conséquent, la mise en œuvre de ce modèle nécessite de définir une heure logique pour chaque événement pour résoudre ce problème. Ensuite, le temps logique est ajouté au modèle qui introduit la cohérence causale temporelle (TCC).

En ce qui concerne la cohérence PRAM **(PRAM consistency)**, les écritures issues d'un processus sont vues dans le même ordre par TOUS les autres processus tandis que Les écritures issues de processus distincts peuvent être vues dans un ordre différent L'avantage de ce type de cohérence c'est ça mise en œuvre simple. Le système le plus représentatif qui met en œuvre cette approche est VFC [3], ce dernier adopte un modèle de cohérence pour les données répliquées dans les centres de données, il prend en compte les différentes sémantiques des données et ajuste automatiquement les niveaux de cohérence en fonction des informations statistiques. Son objectif principal est d'offrir un contrôle de la cohérence pour fournir une haute disponibilité sans compromettre les performances. En outre, VFC 3 cible les magasins de données tabulaires dans le cloud, offrant une rationalisation des ressources et une amélioration de la qualité de service (QoS), réduisant ainsi la latence.[10]

Plein d'autres modèles et outils existent se basant sur la cohérence des données comme par exemple release consistency ou lazy release consistency ou encore le modèle de cohérence faible **(weak consistency)** qui applique la cohérence à un groupement d'opérations ce qui évite la synchronisation globale et cela en introduisant une variable de synchronisation (Token).

5.2 Étude comparative entre les modèles client-centric

Dans la cohérence éventuelle le niveau de cohérence entre les processus et leur confiance est variable, Ce modèle stipule que :

- Toutes les mises à jour se propageront dans le système et que toutes les répliques deviendront progressivement cohérentes.
- Grand magasin de données distribué avec presque aucun conflit de mise à jour.
- En règle générale, une seule autorité mise à jour, avec de nombreux processus read only. **Exemple***changements DNS, Web*
- Le seul conflit est un conflit de lecture-écriture où un processus veut mettre à jour un élément de données pendant qu'un autre tente simultanément de lire la même donnée.
- Des données non mises à jour peuvent être fournies aux lecteurs.
- En règle générale, si aucune mise à jour n'a lieu pendant un certain temps, progressivement toutes les répliques deviendra cohérent.

3. Versatile Framework for Consistency in Cloud Computing

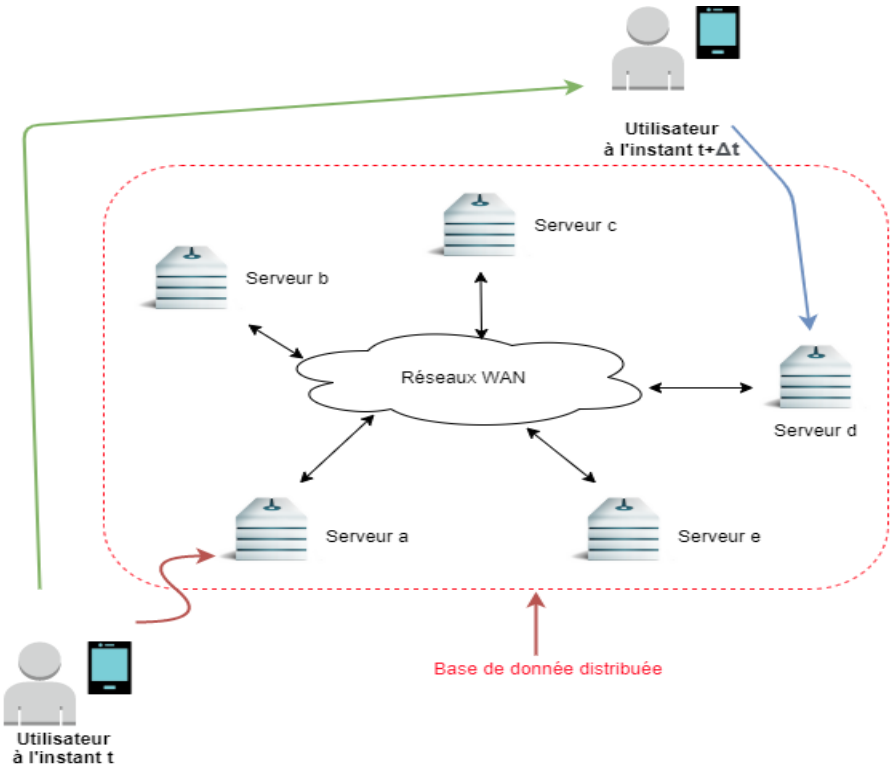

FIGURE 5.2 – Modèle centré sur le client.

Ce modèle de cohérence ne gère pas les mises à jour simultanées. Mais, pour maintenir une vue cohérente pour le processus client individuel pour accéder à différentes répliques à partir d'emplacements différents a été effectuée.

Pour la Cohérence éventuelle , modèle n'apporte pas de garanties concrètes de cohérence . et généralement une seule , une seule autorité mise à jour, avec de nombreux processus read only La cohérence éventuelle offre une disponibilité accrue et de meilleures performances, mais il est plus difficile de programmer des applications, car les données peuvent ne pas être entièrement cohérentes dans toutes les régions. Comme outils ce modèle utilise GEO.

un système d'acteurs géo-distribués open source qui améliore les performances en mettant en cache les états d'acteurs dans un ou plusieurs centres de données, tout en garantissant l'existence d'une seule dernière version grâce à un protocole de cohérence de cache distribué.Le modèle de programmation de Geo prend en charge les acteurs volatils et persistants(Dans le premier cas, la dernière version réside en mémoire et peut être perdue en cas de défaillance des serveurs, tandis que dans le second cas, la dernière version réside dans la couche de stockage.) et les protocoles de cohérence répliqués et à instance unique.[10]

changements DNS, Web mais aussi dans le nombre de retweets, de mentions J'aime ou de commentaires non liés à un thread , la cohérence éventuelle est l'un des 5 niveau de cohérence offert dans Azure Cosmos DB ,nous avons aussi DynamoDB qui adopte la cohérence éventuelle comme modèle par défaut.

Avec la cohérence des écritures monotones, une garantie est ajoutée à la cohérence cohérence que toutes les écritures par un "unique" processus vers le "même" élément de données suivront un ordre monotone et L'ordre des mises à jour est conservé sur les répliques distribuées. C'est là que la cohérence de la chronologie entre en jeu. Contrairement à la cohérence des écritures monotones, elle offre une garantie de "commande totale" sur "toutes" les écritures sur le même élément. Ainsi, les écritures de différents processus sont également sérialisées avec ce modèle de cohérence. Essentiellement, la cohérence de la chronologie est centrée sur les données tandis que les écritures monotones sont centrées sur le client. Avec les écritures monotones, le processus peut se bloquer si les écritures vont vers des répliques différentes et que ces répliques n'ont pas encore été synchronisées. Cependant, le processus ne se bloquera jamais en raison d'écritures simultanées d'un autre processus vers le même élément de données ; ce qui sera le cas avec la cohérence de la chronologie.[68]

Exemple concret d'utilisation : **Bibliothèque de logiciels en cours de développement.**

la cohérence en lecture-écriture est une grande amélioration par rapport à ce qu'on appelle la cohérence éventuelle , Si un client a modifié la valeur d'une donnée, cette modification doit être visible par toute lecture ultérieure par ce même client.

Exemple concret d'utilisation : *Mise à jour du mot de passe , mais aussi dans CassandraDB.*

5.3 Data-centric models vs client-centric

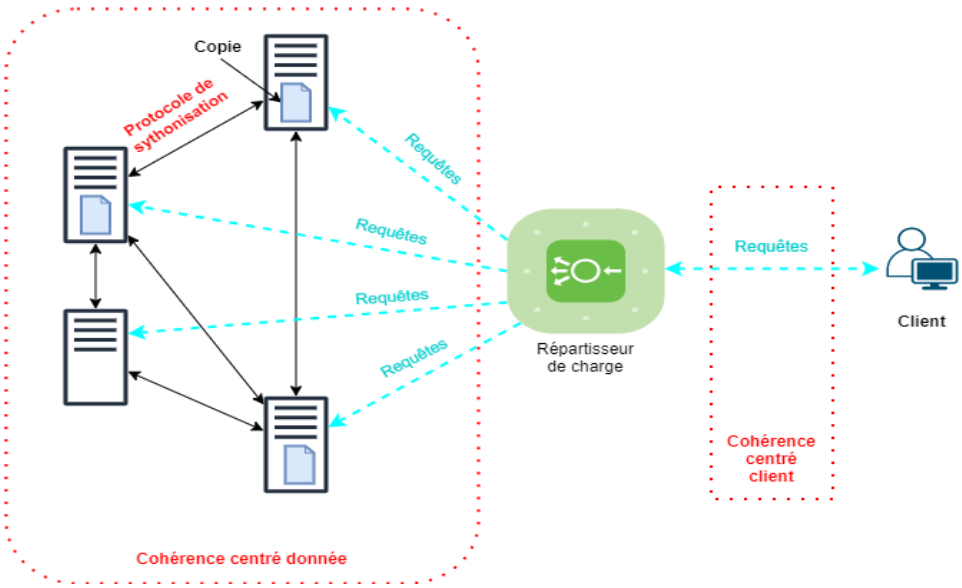

FIGURE 5.3 – Relation d'architecture entre les modèles Client-centric et Data-centric.

Les relations de cause à effet peuvent être entre les opérations des utilisateurs, les données dans le stockage de données partagé, les clés stockées dans un système, les sessions utilisateur et les fichiers journaux existant dans les répliques. Toujours dans la cohérence de session, le type de cohérence peut être différent. Parfois ces modèles sont effectués sur les données du client, parfois sur les sessions et d'autres fois sur les opérations qui sont exécutées par le client. La similitude entre ces deux modèles centrés sur les données et centrés sur le client réside dans leur type d'applicabilité. Ces deux modèles partagent la

cohérence entre les opérations et les sessions de l'utilisateur ainsi que les relations causales entre les opérations des utilisateurs dans différentes sessions. De plus, leur différence est que la cohérence causale se concentre sur l'exécution des opérations de l'utilisateur côté serveur ; cependant, les modèles centrés sur le client tels que (monotones lecture , écriture monotones , lire votre écriture et écriture suivi de lecture) sont effectuées sur les opérations de l' utilisateur au côté client.[62]

Il y a un compromis entre convergence et cohérence. Cependant, si la cohérence est faible, la convergence entre les répliques est réduite. Ensuite, la probabilité d'un taux de lecture périmé est augmentée. Par conséquent, la probabilité du taux de lecture périmé des modèles centrés sur le client est supérieure à celle des autres.[63]

Concrètement quand favoriser une approche centré sur les modèles centrés données et un modèles centrés clients au niveau des entreprise (pme) ?

Data centric concerne le traitement de l'information et centré sur le client lié à l'établissement de relations avec ce dernier. L'approche centrée sur le client a pour objectif de faire des affaires qui se concentre sur la création d'une expérience positive pour les clients afin de maximiser l'offre de produits et d'établir des relations à long terme.

> **Exemple d'utilisation :** *Dans une grande entreprise de pizzeria où la fidélisation de la clientèle est l'une des principales priorités et met fortement l'accent sur la satisfaction de la clientèle.*

L'approche centrée sur les données est celle qui met l'accent sur l'applicabilité d'une informations valide qui permet une prise de décision qui se traduit par une augmentation de la flexibilité aux coûts les plus bas possibles. Comme les entreprises veulent des fonctionnalités, le modèle centré sur les données contribue à la réussite commerciale d'une organisation.[62]

Conclusion

Certains des modèles de cohérence proposés tels que la linéarisation, cohérence fork, cohérence causale ... sont proposés par les chercheurs pour faire face à différents défis des systèmes distribués (voir la Figure ci-dessous).

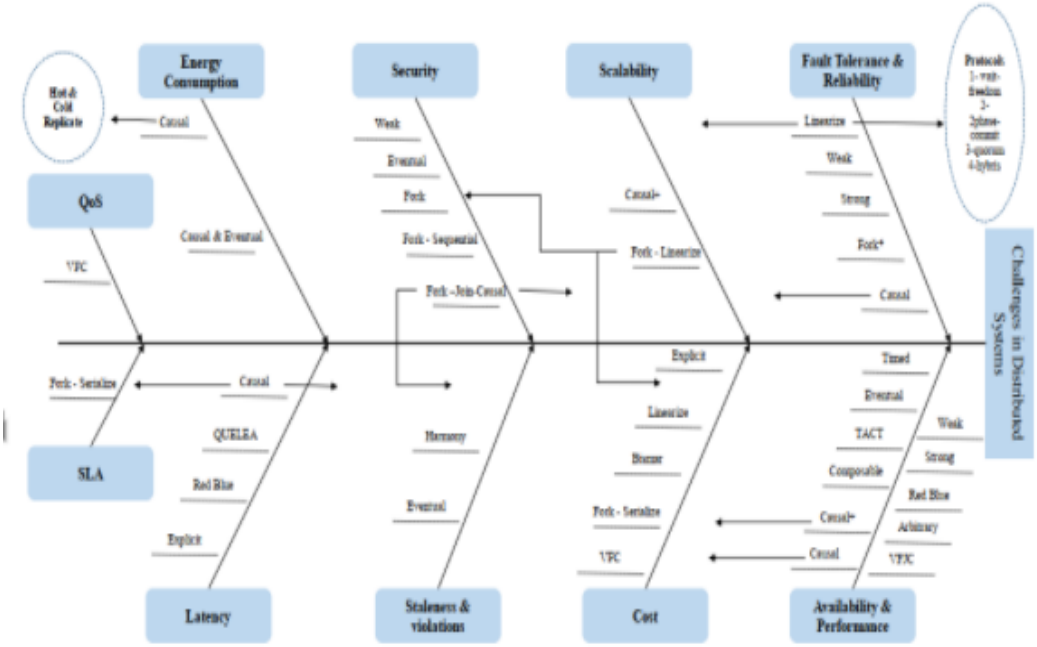

FIGURE 5.4 – Classification des différents types de cohérence en fonction des défis introduits dans les systèmes distribués.

On remarque par exemple que la linéarisation est utilisée pour augmenter la tolérance aux pannes et la fiabilité en appliquant différents protocoles.

Cependant le compromis entre cohérence et disponibilité est le défi le plus important sur lequel les chercheurs ont mené un grand nombre de recherches au cours des dernières années. Après cela, les scientifiques se sont concentrés sur la fiabilité, le coût et la sécurité des systèmes distribués. La cohérence des données entraîne une augmentation de la sécurité et de la fiabilité des utilisateurs dans le système. En plaçant la réplique au point le plus proche du client, la latence et le temps de réponse devraient être réduits. Par conséquent, cela permet une plus grande accessibilité aux données.

Webographie

Partie 1 : Etat de l'art

Chapitre 1 : Big DATA

[2] : https://www.seagate.com/fr/fr/our-story/data-age-2025/

[3] : https://www.futura-sciences.com/tech/definitions/informatique-big-data-15028/

[4] : https://www.mongodb.com/big-data-explained/

[5] : https://cloud.google.com/what-is-big-data/

[6] : http://www.tutorialspoint.com/hadoop/hadoop_quick_guide.html,consulté le 24/02/2019/

[7] : https://le-datascientist.fr/les-10-v-du-big-data/

[8] : https://www.ivision.fr/cloud-et-virtualisation-les-differences/

[9] : https://www.lebigdata.fr/definition-data-center-centre-donnees/

[12] : https://www.piloter.org/business-intelligence/technologie-big-data.htm/

[19] : https://www.allerin.com/blog/top-5-sources-of-big-data/

[20] : https://www.scnsoft.com/blog/what-is-big-data/

[21] : https ://www.ipi-ecoles.com/metiers-big-data/

Chapitre 2 : L'Internet des Objets (IoT)

[23] : https ://fr.slideshare.net/CITC-EuraRFID/flyers-analyse-et-perspectives-de-linternet-des-objets

[24] : https ://www.futura-sciences.com/tech/definitions/internet-internet-objets-15158/

[25] : https ://www.objetconnecte.com/iiot-internet-des-objets-industriel-definition

[26] : https ://www.connectwave.fr/techno-appli-iot/iot/reseaux-et-infrastructures-iot/

[27] : https ://www.digora.com/fr/blog/definition-iot-et-strategie-iot

[28] : https ://www.objetconnecte.net/objets-connectes-chiffres-etudes-2401/

[29] : https ://www.objetconnecte.com/cisco-iot-connexions-reseau-2023/

[31] : https://www.aigassurance.fr/content/dam/aig/emea/france/documents/publications/guides-rapports/aig-iot-french-repport2.pdf

[32] : https ://www.objetconnecte.com/dispositifs-medicaux-attaques-2806/

[33] : https ://www.objetconnecte.com/ia-5g-risques-cybersecurite-iot/

[34] : https ://www.bitag.org/report-internet-of-things-security-privacy-recommendations.php

[35] : https ://www.kolibree.com/fr/

[36] : https ://data-flair.training/blogs/iot-applications/

[37] : https ://www.connectwave.fr/techno-appli-iot/marches-cles-de-liot/automobile-et-iot/

[38] : https ://cloud.google.com/solutions/iot-overview ?hl=fr

Chapitre 3 : Base de données NoSQL

[40] : https://studylibfr.com/doc/2451145/l-article-sur-les-règles-de-codd-pour-les-sgbd-relationnels

[41] : http ://www.mickael-martin-nevot.com/emse/ismin/utilisation-des-basesde-donnees/

[43] : https://www.base-de-donnees.com/acid/

[44] : https://www.lebigdata.fr/acid-base-de-donnees-definition

[48] : https://fr.wikipedia.org/

[49] : https://openclassrooms.com/fr/courses/4462426-maitrisez-les-bases-de-donnees-nosql

Partie 2 : Synthèse

Chapitre 4 : Disponibilité et cohérence des données dans les BD NoSQL

[52] https://www.cs.colostate.edu/~cs551/CourseNotes/Consistency/TypesConsistency.html

[54] https://www.cohesity.com/blog/strict-vs-eventual-consistency/

Chapitre 5 : Étude comparative entre les modèles data-centric et client-centric

[61] https://www.ques10.com/p/20534/explain-the-difference-between-data-centric-and-cl/

[62] https://www.sciencedirect.com/science/article/pii/S0167739X19321053

[63] https://brainly.in/question/3664869

[68] https://svcministry.org/fr/dictionary/what-is-the-difference-between-timeline-consistencyand-monotonic-writes-consistency/

Bibliographie

Partie 1 : Etat de l'art

Chapitre 1 : Big DATA

[1] : Henry E. Brady, " The Challenge of Big Data and Data Science ", Department of Political Science and Goldman School of Public Policy, University of California, Berkeley (January 2019)

[10] Robson A. Campêlo, Marco A. Casanova, Dorgival O. Guedes and Alberto H. F. Laender, " A brief survey on replica consistency in cloud environments " Journal of Internet Services and Applications (21 february 2020)

[11] Armbrust, M, and al (2009). Above the clouds : A Berkeley view of cloud computing. UC Berkeley Technical Report

[13] Jonathan Lejeune, Hadoop:uneplate-formed'exécution de programme Map-reduce, École des Mines de Nantes, 83p, Janvier 2015

[14] Big data application and archetecteure; de himanshu et soumendramohanty

[15] Gerlier, J and Chen, S (2011). Prototypage et évaluation de performances d'un service de traçabilité avec une architecture distribuée basée sur Hadoop. http://docplayer.fr/storage/ 18/673047/673047.pdf

[16] Bejamin, R(2014). Haddop/BigData, Université de Nice Sophia-Antipoli

[17] White, T (2015). Hadoop: The definitive guide 4th edition. Publisher : O'Reilly Media. Isbn : 978-1-491-90163-2

[18] White, T (2012). Hadoop: The definitive guide 3rd edition. Publisher : O'Reilly Media. ISBN : 13-978-1449311520.

[22] M.Corinus, T.Derey, J.Marguerie, W.Techer, N.Vic, Rapport d'étude sur le Big Data, SRS Day, 2012.

Chapitre 2 : L'Internet des Objets (IoT)

[30] N. C. Guillaume Plouin, "Modèles d'architectures de l'Internet des Objets," Sept. 2019.

[39] J. P. DEMURO, "What is fog computing?," Dec. 2019.

Chapitre 3 : Base de données NoSQL

[42] : NoSQL : les meilleures solutions open source,Collaborateurs Smile, MILE,NoSQL : les meilleures solutions open source", 2011.

[45] : Fondamentaux pour le Big Data , Pierre Senellart, Telecom ParisTech, Limites des systèmes classiques de gestion de bases de données.

[46] : Mathieu Roger, Bases NoSQL, synthèse d'étude et projets d'interlogiciels, octera [AT] octera [DOT] info.

[47] : Xavier MALETRAS, Le NoSQL- Cassandra, Thèse Professionnelle, université Paris 13, 27/05/2012

Partie 2 : Synthèse

[50] : Miguel Diogo, Bruno Cabral and Jorge Bernardino, " Consistency Models of NoSQL Databases " (14 february 2019).

[51] : Chaimae .A, Karim .B, Mounir .G, " nosql databases: yearning for disambiguation ", University Mohammed V in Rabat and TicLab, International University of Rabat, Morocco. (March 2020).

Chapitre 4 : Disponibilité et cohérence des données dans les BD NoSQL

[53] Hesam N.S.A, Hossein D, Mohammad H.M, Mostafa R.G, "Consistency models in distributed systems: A survey on definitions, disciplines, challenges and applications ", Department of Computer Engineering, Ferdowsi University of Mashhad, Iran (february 2019).

[55] Zohra M, Nadia N.T " Consistency in Cloud-based Database Systems ", USTHB University, Algeria and CERIST Research Center, Algeria (July 2019).

[57] David Mosberger, " Memory Consistency Models ", Department of Computer Science The University of Arizona (1993).

[58] Vasilis G, Antonios K, Vijay N, Boris G, Arpit J, "Efficient and Available Release Consistency for the Datacenter ", University of Edinburgh (February 2020).

[59] Mahesh D, Vasilis G, Arpit J, Vijay N, " Lazy Release Persistency ", University of Edinburgh (march 2020).

[60] . Bravo, L. Rodrigues, P. Van Roy, Saturn: A distributed metadata ser-vice for causal consistency, in: Proceedings of the Twelfth European Con-ference on Computer Systems, ACM, 2017.

[64] Matthias Majuntke, Dan Dobre, Christian Cachin, and Neeraj Suri, " Fork-Consistent Constructions From Registers ", Technische Universität Darmstadt Germany and NEC Laboratories EuropeHeidelbergGermany and IBM Research ZurichRüschlikon Switzerland (2011).

[65] Paolo V, " Consistency in Distributed Storage Systems : Theoretical Foundations with Applications to Cloud Storage " (avril 2017).

[66] S érgio Esteves, Joao Silva, and Luis Veiga, " Quality-of-Service for Consistency of Data Georeplication in Cloud Computing ", Instituto Superior T ́ecnico - UTL / INESC-ID Lisboa, GSD, Lisbon, Portugal (2012).

[67] Bernstein PA, Burckhardt S, Bykov S, Crooks N, Faleiro JM, Kliot G, Kumbhare A, Rahman MR, Shah V, Szekeres A, Thelin J, " Geo-Distribution of Actor-Based Services ", (October 2017).