République algérienne démocratique et populaire
Ministère de l'enseignement supérieur et de la recherche scientifique
Université Mouloud MAMMERI de Tizi-Ouzou
Faculté de Génie électrique et Informatique
Département Informatique

Mémoire *de fin d'étude*

En vue de l'obtention d'un Master en Informatique

Thème

Conception et Réalisation d'une base de donnée NoSQL sous Hadoop dans le cadre d'une ville intelligente (contexte : Inondation)

Réalisé par :

BELILIT Ryma

KACI Sonia

Diriger par : M^me^ TAOURI Dalila

Proposé par : Mr TAMANI Noureddine

Devant le jury composé de :

Président : Mr YACINE Younes
Examinatrice : Melle YESLI Yasmine
Examinateur : Mr RADJA Hakim

Table des matières

Liste des tableaux

Table des figures

Motivations et objectifs

”La création d’un ”Internet des Objets”, le développement et la diffusion ubiquitaire des technologies basées sur les capteurs, vont à terme brouiller les frontières entre monde virtuel et monde physique et pourraient modifier la nature même de la vie privée. Les enjeux de sécurité sur le long terme restent encore à analyser et à résoudre. . . ”[1]

Le développement de l’Internet et la multiplication des objets connectés à travers le monde s’accompagnent d’une croissance exponentielle des données crées sur la toile. La multiplication des moyens de communication et d’échange n’y est pas étrangère ; en effet, les différents écrans nous suivent partout, tout au long de la journée.
En 2011, il y avait près de 9 milliards de terminaux connectés dans le monde et ce chiffre devrait s’élever à 24 milliards en 2020 [1].

.Outre les smartphones, tablettes et télévision connectées, les nouveaux objets connectés, tels que les voitures, les appareils électroménagers ou encore les montres connectées qui déferlent sur le marché devraient remonter une quantité phénoménale d’informations dans les années à venir. Si l’on en croit les résultats d’une étude récente, le monde a manipulé en 2012 plus de 2,8 zetaoctets d’informations soit 2,8 milliards de gigaoctets, [2] **ce chiffre est colossal, mais le plus intéressant dans cette étude est de savoir que seul 0,5% de ces 2,8 zetaoctets ont été analysés d’une manière ou d’une autre alors que l’étude estime que 25% d’entre elles représentent une valeur potentielle pour les entreprises.Les informations disponibles sur Internet ne sont plus seulement volumineuses, elles sont également très diverses, non structurées au sens ’où elles ne se présentent pas sous la forme de lignes et de colonnes comme elles sont structurées sur le web.**

Ces données doivent donc être structurées avant d’être analysées et exploitées par les technologies actuelles’. Toutes leurs interactions avec les nouvelles technologies génèrent des données : téléchargement d’un fichier, consultation d’une vidéo, coup de téléphone,

1. ”Les technologies les plus profondément enracinées sont les technologies invisibles. Elles s’intègrent dans la trame de la vie quotidienne jusqu’à ne plus pouvoir en être distinguées.” Mark Weiser

2. http ://www.emc.com/collateral/analyst-reports/idc-the-digitaluniverse-in-2020.pdf

envoi de SMS, utilisation de GPS... Ce n'est pas tant les interactions en tant que telles qui génèrent autant de données, mais c'est surtout l'ensemble des informations annexes (les méta-données) et des communications cachées entre différents serveurs (publicitaires par exemple) qui ont lieu au même moment, qui génèrent un flux impressionnant de données. **L'ensemble de ces milliards de données, représente ce que l'on appelle communément les Big data.**

Les premières entreprises à avoir compris leur intérêt sont les géants du web actuel tels que Google, Yahoo, Microsoft, Facebook ou bien Amazon. Du fait de leur succès ou de leur volonté de vouloir gérer une quantité très élevée d'informations**.**

Ces technologies sont assez récentes et sont pour la plupart issues des grandes sociétés du web américain telles que Google, Yahoo ou encore Linkedin. Ces dernières ont dû créer pour leurs propres besoins un ensemble d'outils afin de traiter les masses de données qu'ils devaient analyser chaque jour. La plupart d'entre elles ont été rendues publiques via une licence Open Source ou données à la fondation Apache pour pouvoir être réutilisées et améliorées par la communauté, tout le monde bénéficie donc de ce cercle vertueux.

Des milliards de données d'activités, de ressenties, d'intentions, ou juste de comportement sont stockées via différentes applications opérationnelles, ou outils et autres médias de tous les jours. Et les trois quarts de ces données ont été créés par les utilisateurs–consommateurs–patients que nous sommes. L'exploitation de ces données peut s'avérer complexe, et ne peut se résumer à un simple projet technologique de migration de données ou d'architecture informatique.

Traiter l'ensemble de ces données requiert un travail colossal. L'atteinte de cet objectif passe par la maîtrise et l'apprentissage des données accumulées, de leurs analyses, et des retours métiers qui vont améliorer le modèle, le rendre plus apte à anticiper.

. Problématique

Etant donné l'aspect particulier des Big Data, les méthodes de traitement, de stockage et d'analyse classique ne sont pas assez efficaces, ce qui engendre des problématiques quant aux méthodes à utiliser pour parvenir à des résultats satisfaisant :

Notre problématique traite les points suivants :

1. Comment valoriser et traiter un amas de données ?

Travailler les données pour les valoriser et obtenir des résultats, nécessite de l'intelligence, l'outil et de la réflexion. Le Big Data n'a rien à voir chez une entreprise A ou avec celle de l'entreprise B et pourtant ils peuvent être clients de l'un et de l'autre. Le traitement doit être effectué de façon stratégique en fonction de la nature des données et de leur utilisation.

2. Comment stocker les données de manière à faciliter l'accès ?

Face à l'évolution informatique dans ces dérnières années et notamment les grandes volumes de données échangés, les SGBDR classiques ont montré de très grandes faiblesses en matière de scalabilité et en montée en charge, ce qui a obligé les grands acteurs du web à chercher d'autre solutions. Les systèmes NoSQL viennent, justement, pour accomplir cette mission.

Donc les SGBD classiques ne sont pas adaptés pour ce genre de stockage, il est préférable d'utiliser les **BDD non relationnelle.**

3. Efficacité des accès aux données

Les performances en termes de débit (nombre de transactions types exécutées par seconde) et le temps de réponse (temps d'attente moyen pour une requête type) sont un problème clé des SGBD. L'objectif de débit élevé nécessite un temps de latence

minimal dans la gestion des tâches accomplies par le système.Les systèmes NoSQL viennent, justement,apporter une réponse pour ce probléme**.**

4. La Structure de donnée

Entre le relationnel et le NoSQL, il y a des conceptions fondamentalement différentes de la structure des données. En fait, dans le NoSQL lui-même, il existe différentes façons de considérer la donnée. Le modèle relationnel se base sur une structuration importante des données. Les métadonnées sont fixées au préalable, les attributs sont fortement typés, et selon les principes édictés par Codd, la normalisation correcte des structures implique qu'il n'y ait aucune redondance des données. Les entités ont entre elles des relations, ce qui permet d'avoir un modèle totalement interdépendant. Dans le modèle relationnel, la métaphore de la structure, c'est le tableau : une suite de lignes et de colonnes avec sur la première ligne, le nom des colonnes en en-tête. Cela signifie que les métadonnées font partie de l'en-tête et ne sont bien sûr pas répétées ligne par ligne. Cela veut dire également que chaque donnée atomique est stockée dans une cellule, et qu'on peut la retrouver en indiquant le nom de la colonne et l'identifiant de la ligne. Nous allons voir dans les chapitres qui suit les différentes structures de données utilisées dans le monde NoSQL, comme les paires clé-valeur ou les documents JSON .

5. La Sémantique

Au niveau de la haute disponibilité des données : dans un système distribué, assurer la cohérence des données est très contraignant et pénalise les performances du système. Pour venir à bout de cette limitation, les systèmes NoSQL optent pour un relâchement complet des contraintes d'intégrité référentielles et sémantiques, qu'on retrouve dans l'approche relationnelle. Ceci implique une redondance de données et une absence de schéma (ou de modèle) au sein la base de données.

La première chose qui vient à l'esprit quand on parle de **Big Data** et d'**IoT** est l'augmentation du volume de données **qui va frapper le cadre de stockage de données** .

Compte tenu de l'impact considérable de l'IoT sur l'infrastructure de stockage de don-

nées, les entreprises ont commencé à se tourner vers le modèle **P**latform-**a**s-**a**-**S**ervice, une solution basée sur le **cloud,** plutôt que de maintenir leur propre infrastructure de stockage. Contrairement aux systèmes de données internes qui doivent être constamment mis à jour à mesure que la charge de données augmente, PaaS offre flexibilité, évolutivité, conformité et architecture sophistiquée pour stocker toutes les données IoT de grande valeur.

Les options de stockage dans le cloud incluent les modèles publics, privés et hybrides. Si une entreprise possède des données sensibles soumises à des exigences de conformité réglementaire qui nécessitent une sécurité accrue, l'utilisation d'un cloud privé constitue la meilleure solution. Pour les autres entreprises, un cloud public ou hybride peut être utilisé pour le stockage des données IoT.

Pour réaliser cela il nous faut des machines de caractéristique de 16 giga de RAM et que l'acces est cloud necessite un prix

Pour notre part on essaye d'apporter une modeste participation en essayant d'apporter des solutions à cette problématique est ce à travers un exemple d'application qui est : la surveillance du phénomene environmental dans le contexte d'une inondation qui sera confronté au problèmes suivant :

- Surveillance de la montee du niveau d'eau.
- Evacuation de la population vers des unités de secours.

• Objectifs

C'est dans le contexte de cette problèmatique a plusieurs facettes qu'ils nous a été demandés d'essayer de trouver ou au moins d'y réflechir.
Pour pouvoir travailler dans un contexte cohérent avec cette problematique il nous a été demandé de travailler et de réflechir au données véhiculés par une ville inteligente dans un contexte d'inondation. Pour ce faire il nous a fallu imaginé un système dit "Surveillance d'un Phénomene Envirenemental ".
On s'est fixé donc les objectifs suivants pour rester dans le cadre de la problematique énoncée.

1. Imaginer la ville intelligente dans le contexte de l'inondation.
2. Récolter les differentes données nécessaire pour faire face a une inondation Avant et Apres la montée des eaux.
3. Structurer les données recoltés dans une base de données NoSQL.
4. Etudier et analyser l'outil qui nous a été demandé d'utiliser à savoir Hadoop,et en présentant une critique .(**Annexe1)**
5. Imaginer et implémenter des scénarios qui peuvent se présenter avant ou pendant l'inondation.
6. Apporter une conclusion concernant les problémes de :

   - Scalabité .
   - Temps de réponse .
   - Sémantique .

liés à l'outil Hadoop et dans la mesure du possible entrevoire des perspectives.

Organisation du mémoire

Ce mémoire sera donc diviser en trois parties :

Partie 1 : Renférme trois chapitres

• **Chapitre 1 :** donne une vision globale sur le Big data , ses caractéristiques, son fonctionnement ainsi que les différents domaines dans lesquels il est utilisées,On a aussi recensé les différents modèles de bases de données NoSQL qui existent, actuellement, dans le marché.en accentuant sur les solutions les plus populaires

•**Chapitre 2 :** présente l'IoT (Internet of things) définition, domaine d'application , son architecture ,le fonctionnement de l'IoT, ainsi que les vulnérabilités et les menaces relatives à son déploiement, le reste du chapitre est consacré à la définition de bases ,et quelques notions utilisées dans le domaine de la sécurité.

• **Chapitre 3 :** consistere à expliquer les bases de données relationnelles et leurs limites,ensuite nous avons présenté les bases de .données Nosql, ainsi que les avantages qu'une base de données de type NoSQL peut avoir en comparaison à une base de données relationnel standard.

• **Chapitre 4 :** présente les différents composants d'Hadoop, principalement, HDFS et le modèle de programmation MapReduce. on a présenté également l'outil d'entreposage de données Hive et ce pour comprendre réellement l'implémentation de la plate-forme Hadoop et de son outil Hive.

Partie 2 : Analyse et conception

• **Chapitre 5 :** présente dans la premiére partie les concept de base de la ville intelligente et définit la gestion des données spatio-temporelles issues de capteurs dans le domaine de la surveillance environnementale et nous avons décrit les aspects que nous avons pris en considération dans notre étude (le cas d'innodation), dans la deuxieme partie nous avons présenté les démarches suivie pour realiser la conception de notre systéme de surveillance envirenmental .

Partie 3 : Réalisation

- **Chapitre 6 :** nous avons aborder la partie pratique de notre projet qui consiste à la mise en œuvre complète de l'environnement Hadoop Dans une distribution Hadoop qui est cloudera et nous avons presenté les differents outils implémentés dans notre projet
- **Chapitre 7 :** présente la mise en oeuvre d'une solution de modélisation de données qui garantie une performance des requêtes, et détaille l'implémentation de la base de données sous la plate-forme Hadoop avec l'outil Hive.

Première partie

Etat de l'art

Chapitre 1

Le Big Data

Introduction

Le terme anglo-saxon ”Big data” n'a pas d'équivalent en français. On parle parfois de ”données massives”, ou de ”données de grande dimension”. Mais on parle également de concept, de phénomène, ou encore de discipline Big data. Ce terme désigne le traitement automatisé de grandes quantités de données pour en extraire des informations. Pour cela, on recourt à des nouvelles procédures de transfert, de stockage et d'analyse.

Le terme ”Big data” est aussi devenu synonyme de ”data analysis” (analyse des données). **Le Big data correspond donc à la fois à une masse considérable de données, mais aussi aux technologies, processus et techniques mis en œuvre pour gérer des données à grande échelle dans le but d'en extraire des connaissances .**

Le **Big Data** est un phénomène qui a vu le jour avec l'émergence de données volumineuses qu'on ne pouvait pas traiter avec des techniques traditionnelles. Les premiers projets de Big Data sont ceux des acteurs de la recherche d'information sur le web ”moteurs de recherche” tel que Google et Yahoo.

En effet, ces acteurs étaient confrontés aux problèmes de la **scalabilité** (passage à l'échelle) des systèmes et du temps de réponse aux requêtes utilisateurs. Très rapidement, d'autres sociétés ont suivis le même chemin comme Amazon et Facebook. Le Big Data est devenu une tendance incontournable pour beaucoup d'acteurs industriels du fait de l'apport qu'il offre en qualité de stockage, traitement et d'analyse de données.[1]

• **La mise en donnée du monde**

Au delà d'un volume gigantesque, c'est la **diversité** des sources de données qui donne au Big Data toute son ampleur.

Deux leviers principaux soutiennent cette croissance de la production de données :

1.L'effacement de la frontière entre comportements online et offline

2.La mise à disposition des données publiques.

On identifie aujourd'hui **quatre grands facteurs responsables de l'explosion de la production de données** par nos comportements connectés [2]

• **Les réseaux sociaux**

A chaque minute écoulée, on compte sur internet au niveau mondial : 350 000[1] tweets, 695 000 mises à jour de statuts et onze millions de messages instantanés sur Facebook. Ce dernier s'occupe également de la gestion de 71 milliards de photos.

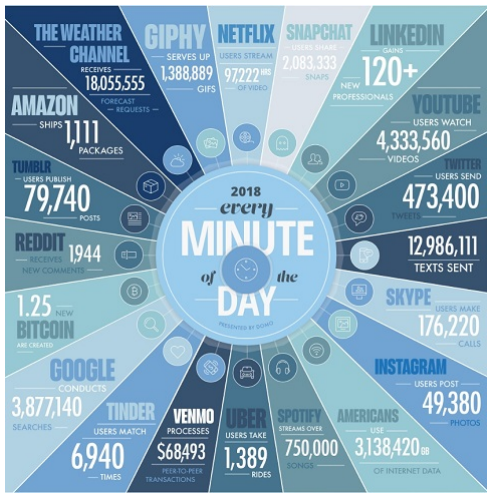


FIGURE 1.1 – Une minute sur Internet [3]

• **Les objets connectés**

L'Internet des Objets contribuent "à doubler la taille de l'univers numérique tous les deux ans, lequel devrait peser 44.000 milliards de giga-octets en 2020, soit 10 fois plus qu'en 2013" [4] [5]

1. www.planetoscope.com(janvier 2019)

• **Les technologies mobiles**

On considère qu'un smartphone génère environ 60 gigabytes chaque année. Si on multiplie ce chiffre par le nombre de smartphones dans le monde soit environ un milliard, on obtient une production de données par an de 56 exabytes soit la totalité de la bande passante consommée en 2013, dans le monde. Le terme Big Data prend alors tout son sens. En 2019, les prévisions estiment qu'il y aura 3,3 milliards de smartphones dans le monde.

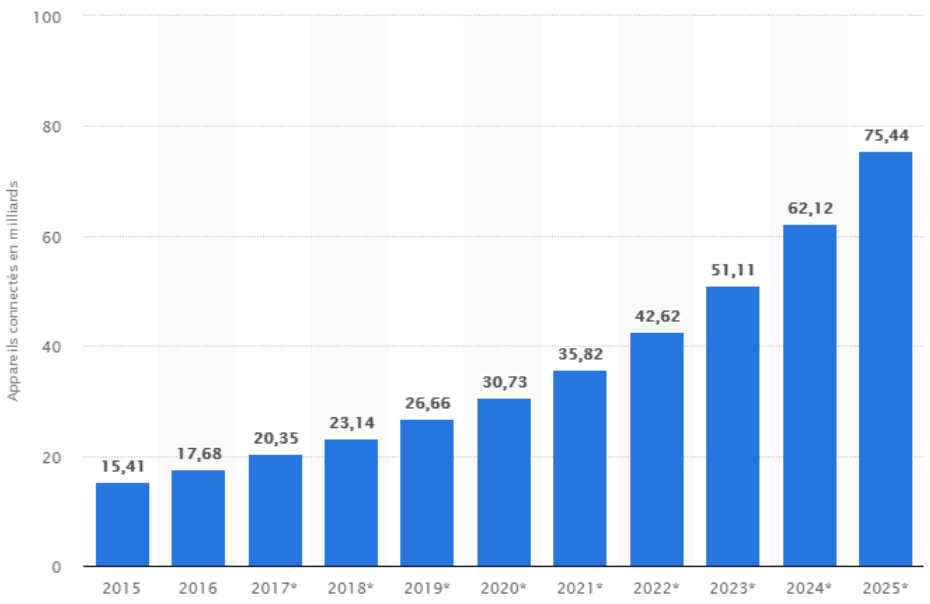


FIGURE 1.2 – nombre d'appareils connectés dans le monde de 2015 à 2025 (en milliards)

• **Les comportements numériques scrutés, analysés et stockés**

A chaque minute écoulée, on compte sur Internet 700 000 recherches Google, 12 000 annonces sur Craigslist, 600 nouvelles vidéos Youtube et 1 500 articles de blogues.

Ce chapitre présentera une vision globale sur le Big data , ses caractéristiques, son fonctionnement ainsi que les différents domaines dans lesquels il est utilisées,On introduirera les différents modèles de bases de données NoSQL qui existent, actuellement, dans le marché.en accentuant sur les solutions les plus populaires.

1.1 Caractéristiques des Big data

Les principales caractéristiques concernant les Big data retrouvées dans les articles sont leur taille **importante** et leur **complexité**. Les Big data ne concernent pas seulement l'ampleur et l'étendue des nouveaux jeux de données mais aussi leur complexité croissante. Pour décrire la complexité des Big data, une approche largement utilisée est celle des trois "V" [5], [6], [7], [8],[1] ,[9] : **Le volume, la variété et la vélocité (voir la Figure 1.3).**
"Big data est un terme utilisé pour décrire les données dont le traitement est problématique du fait de leur taille (volume), de la fréquence de leur mise à jour (vélocité), ou de leur diversité (variété)". La véracité est un quatrième "V" parfois ajouté pour décrire un challenge(inclue le stockage et l'analyse de grands magasins de données divers, en croissance rapide puis la déterme précisement la meilleure façon de gérer ces données) posé par les Big data. Certains auteurs mentionnent même un cinquième "V" : la valorisation [7].

FIGURE 1.3 – La formule des 5V du Big data

1. Le Volume

Le volume est la principale caractéristique du Big Data. Le terme est en effet directement tiré de l'immense masse de données générées au quotidien.Selon IBM, une moyenne de 2,5 quintillions de bytes de données sont créés chaque jour, soit environ 2,3 trillions de gigabytes.D'année en année, la quantité de data augmente considérablement. Sur l'ensemble

de l'année 2020, 40 zettabytes de données seront créés, soit 43 trillions de gigabytes. Ceci représente une quantité 300 fois plus importante qu'en 2005. [10]

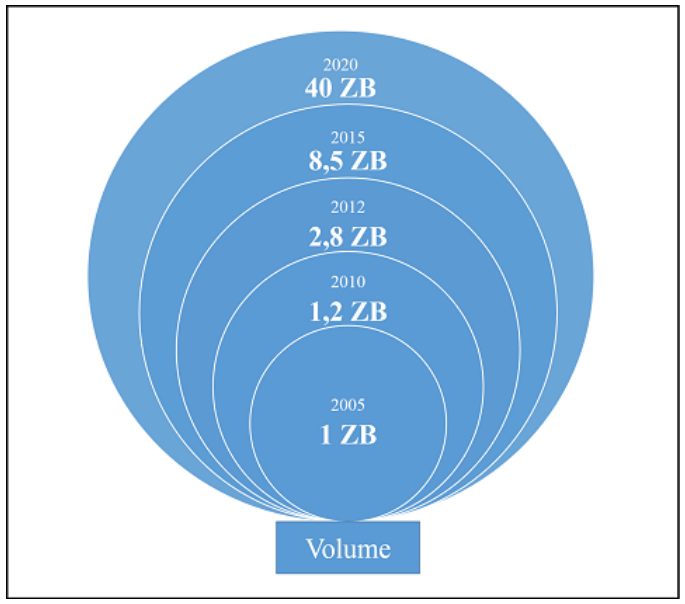


FIGURE 1.4 – Volume

2. La Variété

La variété se traduit en l'agrégation des données provenant des sources très diverses ou le regroupement de données provenant de sources indépendantes. On désigne l'origine variée des sources de données qui sont soit structurées ou non structurées (images, mails, tweets, données de géolocalisation....) [8].

Exemple

Prenons l'exemple des messages électroniques : Un processus de découverte légale peut nécessiter de passer au crible des milliers, voire des millions de messages électroniques d'une collection. Aucun de ces messages ne sera exactement comme un autre. Chacun se composera de l'adresse électronique de l'expéditeur, d'un destinataire et d'un horodatage. Chaque message comportera un texte écrit par un être humain et éventuellement des pièces jointes.

Les photos, vidéos, enregistrements audio, messages électroniques, documents, livres, présentations, tweets et tracés ECG sont tous des données, mais ils ne sont généralement pas structurés et sont incroyablement variés.

Toute cette diversité des données constitue le vecteur de la variété du big data.

3. La Vélocité

L'augmentation rapide des données est une autre caractéristique des Big data. Il s'agit de données en temps réel ou en temps quasi-réel, La vélocité correspond à la vitesse à laquelle les données d'aujourd'hui sont générées et traitées simultanément [8].

Exemple

IBM prend des voitures modernes, équipées en moyenne de 100 capteurs capables de mesurer en temps réel le niveau d'essence, la pression des pneus et bien d'autres données. Le phénomène est également illustré par l'exemple du New York Stock Exchange, qui enregistre environ 1 terabyte de données durant chaque session.Chaque activité réalisée sur internet est désormais traquée avec précision, grâce à un total de 18,9 milliards connections en réseau dans le monde, soit environ 2,5 pour chaque individu sur Terre. Internet n'a jamais été aussi rapide.

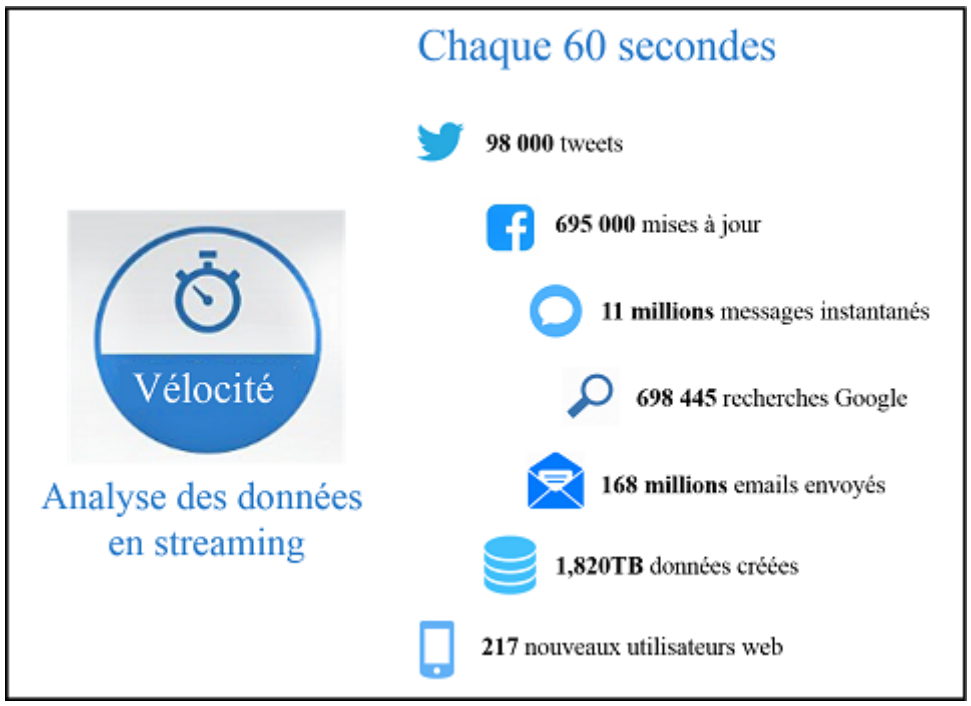


FIGURE 1.5 – La Vélocité

4. La Véracité

La véracité, l'exactitude des données demeurent aujourd'hui le principal défi du Big Data. À l'heure actuelle, ces données ne sont pas encore suffisamment maîtrisées, et la précision des analyses s'en trouve affectée. Ainsi, dans le cadre d'un sondage réalisé par IBM, 27% des entreprises interrogées avouent ne pas être certaines de l'exactitude des données qu'elles collectent. De même, un chef d'entreprise sur trois utilise les données pour prendre des décisions, mais n'a pas vraiment confiance. Ce manque de véracité et de qualité des données coûte environ 3,1 trillions de dollars par an aux États-Unis . [10]

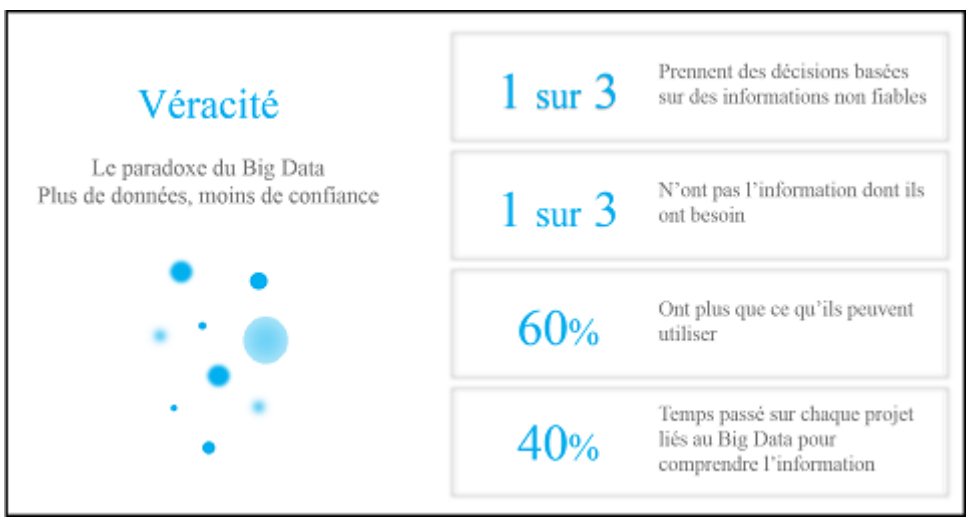


FIGURE 1.6 – La Véracité

5. La valeur

La valorisation signifie que quelque part à l'intérieur de ces données, il y a quelques précieuses informations, quelques données d'or à extraire, si la plupart des morceaux de données, individuellement, peuvent sembler sans valeur .[6]

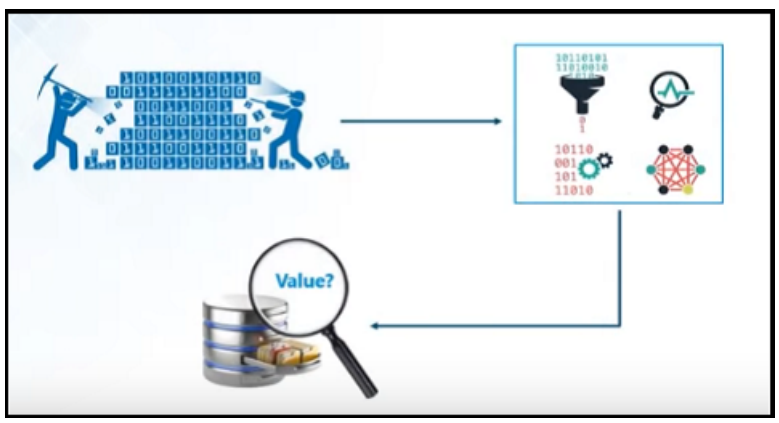


FIGURE 1.7 – La Valeur

1.2 Big data et l'internet des objets

Outre le "Big Data", l'Internet des objets - en anglais, "Internet of Things" (IoT) est l'une des principales questions d'avenir de l'informatique. Dans cette section, on va expliquer le lien entre Big Data et IoT et comment ces deux phénomènes interagissent.

1.2.1 Des évolutions technologiques favorisent la nouvelle ère de l'Internet des objets

Trois évolutions technologiques accompagnent le développement de l'Internet des objets et du Big Data et permettront de transformer la donnée en information utile pour prendre les bonnes décisions, au bon moment et en toute sécurité :

- L'essor des Smartphones a engendré une baisse du coût des capteurs et favorisé l'essor de tout un ensemble d'objets connectés qui intègrent des myriades de capteurs produisant des données.
- L'mélioration des performances des réseaux en termes de débit et/ou de consommation d'énergie pour véhiculer ces données.
- L'amélioration des algorithmes dans le traitement des données et pour des volumes fortement croissants.[11]

1.2.2 Relation Big data et IOT

"Quand on parle **Big Data**, on va tout de suite parler **volume de données**. Mais au delà du volume, rien que la variété de ces dernières va constituer un enjeu crucial, ce phénomène est amplifié par l'avènement des **objets connectés**. Selon Tania Aydenian directrice de datavenue (Orange Technocentre) [9]

L'Internet de Objets est devenu un réel phénomène en 2014 et 2015. Si les technologies sont apparues depuis quelques années, le sujet s'est trouvé au coeur des discussions tout au long de l'année. Et pour cause, l'internet des objets contribuera "à doubler la taille de l'univers numérique tous les deux ans, lequel devrait peser 44.000 milliards de gigaoctets en 2020,

soit 10 fois plus qu'en 2013"[2]. On estime qu'actuellement seulement 22% des données sont exploitables pour le Big Data, chiffre qui sera porté à 35% grâce aux données numériques issues de l'internet des objets .[4]

Le Big data et les objets connectés représentent un important relais de croissance économique selon de nombreuses études[3]. Ils ouvrent la possibilité de :

1. connecter les personnes ou les objets de manière plus pertinente.
2. fournir la bonne information au bon destinataire et au bon moment.
3. ou encore de faire ressortir les informations utiles à la prise de décision [5].

On vit aujourd'hui une révolution numérique globale, une nouvelle révolution industrielle, alimentée par l'essor des objets connectés associé à l'exploitation du Big data, qui est appelée parfois Internet of Everything (IoE), l'Internet du Tout connecté."l'offre d'objets connectés est très en avance sur les usages"[4] .

Le flot grandissant d'objets connectés soutient la croissance du Big data qui, à son tour, facilite l'explosion des usages [5].

1.2.3 Quelques exemples de l'intégration de l'Internet des Objets et du Big Data dans des macrostructure

La société multinationale américaine IBM "International Business Machines Corporation" a attribué la quantité croissante de Big data vers un monde instrumenté, interconnecté et intelligent qui est envisagé par l'Internet des Objets [6]**.** Et aujourd'hui beaucoup de conversations qui ont lieu autour de l'Internet des Objets sont incomplète sans une mention de Big data. "Le succès ou l'échec de l'Internet des Objets repose sur le Big data"[5], analyste chez Forrester . [12] Comme les organisations entrent dans l'Internet des Objets, elles doivent comprendre la relation symbiotique entre elle et le Big data. Voici quelques exemples de l'Internet des Objets et Big data fonctionnent bien ensemble pour fournir l'analyse et la perspicacité.

2. Selon une étude EMC-IDC
3. Cisco, McKinsey, Idate, Inspection générale des finances, Gartner, Boston Consulting Group, A.T. Kearney
4. Selon Yannick Lacoste et Jean-François Vermont
5. Brian Hopkins

- **Entreprise postale**

FIGURE 1.8 – : Les véhicules intelligentes de UPS

United Parcel Service "UPS" est une entreprise postale qui utilise des capteurs sur ses véhicules de livraison **(voir la Figure 1.8)** pour surveiller la vitesse, le kilométrage, le nombre d'arrêts, et le moteur...pour le but d'économiser de l'argent, d'améliorer l'efficacité et de réduire son impact environnemental.

Selon une infographie publiée par UPS, les capteurs enregistrent plus de 200 points de données pour chaque véhicule dans une flotte de plus de 80 000 chaque jour [13]. Ceux-ci aident l'entreprise à réduire le temps de ralenti, la consommation de carburant, et les émissions nocives.

Autrement UPS utilise le Big data dans son projet ORION qui signifie On-Road Integrated Optimization and Navigation (optimisation et navigation intégrées sur la route). ORION connaît le chemin et les adresses de livraison des clients, les lieux et les heures de livraison et de transport ainsi que les règles syndicales pour les chauffeurs. Il garde également en mémoire 250 millions d'adresses de livraison. ORION analyse toutes ces données et prépare des instructions de transmission optimisées jusqu'à la minute même du départ du chauffeur. [14]

- **Le tourisme**

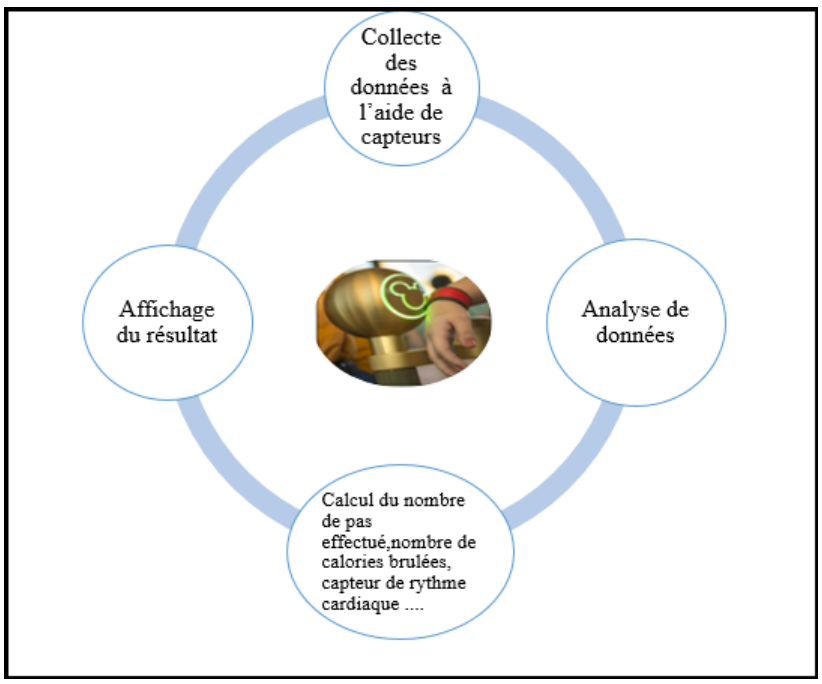


FIGURE 1.9 – Fonctionnement du bracelet connecté [22]

La société Disneyland Resort aux Etats-Unis a investi plus d'un milliard de dollars pour déployer son bracelet connecté connu sous le nom de MagicBand **(voir la Figure 1.9)**.

Un bracelet qui emporte l'influence des visiteurs puisque 90% d'entre eux se sont déclarés satisfaits des bénéfices du port de ce bracelet, suite à une enquête. Le bracelet permet aux parents de géolocaliser leurs enfants, de payer dans les boutiques et restaurants ou encore de récupérer les photos prises dans les attractions. La maison Disney recueille ces données et les emploie en vue d'améliorer l'expérience des visiteurs [15], [12]**.**

- **Le sport**

FIGURE 1.10 – l'application Nike + iPod / iPhone [5].

Big data est partout, même dans le jogging, Un exemple d'application de technologies de détection dans notre vie quotidienne est l'application Nike + iPod / iPhone **(voir la Figure 1.10)** C'est une application qui recueille et suit les informations telles que les détails d'entraînement, la distance, les calories brûlées, etc. d'un jogger en utilisant des chaussures Nike + iPhone / iPod. Une autre application similaire est iSmoothRun (www.ismoothrun.com). En outre, il permet le téléchargement des données sur les réseaux sociaux de fitness tels que (www.RunKeeper.com). Les données deviennent "Big data" lorsque l'on considère des millions d'utilisateurs [6].

1.3 La sémantique dans le Big data

Le Big data se réfère ainsi à ce qui peut être accompli à grande échelle et ne peut pas l'être à une échelle plus petite. Le Big data s'appuie sur le développement d'applications à visée analytique, **qui traitent les données pour en extraire de la sémantique**. Une chaîne de transformations bien connue existe dans le domaine du management de l'information, c'est la chaîne [16] :
"donnée → information → connaissance → sagesse" que représente le modèle DIKW [6].

La représentation graphique la plus populaire pour DIKW est une pyramide, avec les données à la base et la sagesse à son sommet **(Figure 1.11)**. Cette représentation suppose implicitement que les éléments les plus hauts dans la pyramide nécessitent les éléments inférieurs pour être définis, et qu'ils peuvent être atteints après un processus de transformation des éléments inférieurs.

Le modèle DIKW est alors une chaîne où l'information est le résultat du traitement des données, la connaissance est le résultat du traitement de l'information et la sagesse est le résultat du traitement de la connaissance.

6. Data, Information, Knowledge, Wisdom

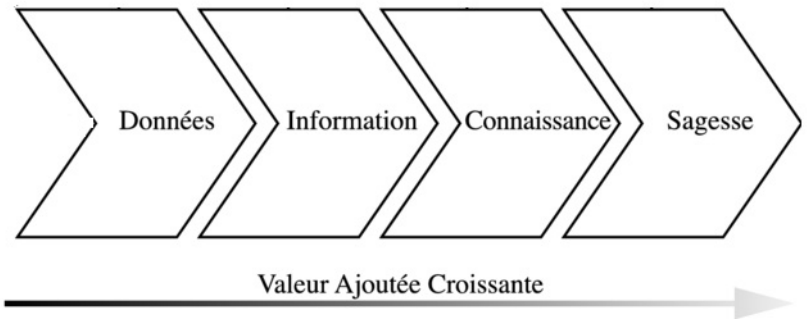


FIGURE 1.11 – Le model DIKW

1.4 La révolution de Big data par la technologique

Les progrès techniques et la baisse des prix associée dans la gestion de la donnée sont les premiers facteurs d'émergence du Big Data. Ces progrès concernent à la fois les logiciels de traitement de données et l'architecture informatique nécessaire à son transit et à son stockage. Les données massives existent déjà depuis très longtemps car nous avons toujours stocké les données.

Et parmis ces technologies on peut citer :

— Map Reduce

— Hadoop

— Bases NoSQL

— Stockage "In-Memory"

— Cloud Computing

Ce qui fait un projet "Big Data", c'est la technologie que l'on utilise. Avec ces technologies, ce qui change, c'est la puissance et la rapidité de gestion de ces données [2].

Exemple

Une analyse approfondie reliant les données comptables aux systèmes de repérage et de gestion des commandes peut fournir des informations stratégiques précieuses qui ne seraient pas disponibles avec les outils classiques. Afin de les identifier, une masse importante de données doit être traitée presque en temps réel à partir de

sources multiples et hétérogènes. Cette fonction permettant de puissants calculs peut maintenant être effectuée par le biais des technologies Big Data.

1.5 Le Big Data et le Cloud

Jusqu'à l'émergence du concept de "big data", les données étaient principalement traitées de façon locale, dans des entrepôts de données (ou data warehouse) constituées de plusieurs bases de données structurées.

Peu à peu les sources de données se sont largement diversifiées, sont devenues relativement hétérogènes (format des données très variable) et ont été surtout localisées sur Internet. Autre particularité, ces informations sont produites en permanence, avec une cadence soutenue [1]

Pour pouvoir exploiter ces mines d'informations et ces flux de données, d'importantes capacités de calcul sont nécessaires, souvent uniquement disponibles dans de grands data centers. Le cloud computing[7]permet donc de "louer" une puissance de calcul et un espace de stockage adaptés pour un traitement big data. En effet, seuls peu d'acteurs sont en mesures d'effecteur ce traitement avec leurs propres infrastructures, au vu des équipements informatiques nécessaires. Le cloud va donc mettre le big data à la portée des PME et des acteurs non experts du traitement des données .

Exemple :

Le cloud computing est une solution très adaptée pour les besoins de type paie , messagerie ou éditique(création massive et ponctuelle de documents).De plus le cloud peut permettre de tirer parti au maximum du partage du matériel (processeurs , disques.....).[14]

7. ***Le Cloud (ou cloud computing)*** *est une technologie qui permet de mettre sur des serveurs localisés a distance des données de stockage ou des logiciels qui sont habituellement stockés sur l'ordinateur d'un utilisateur, voire sur des serveurs installés en réseau local au sein d'une entreprise.*

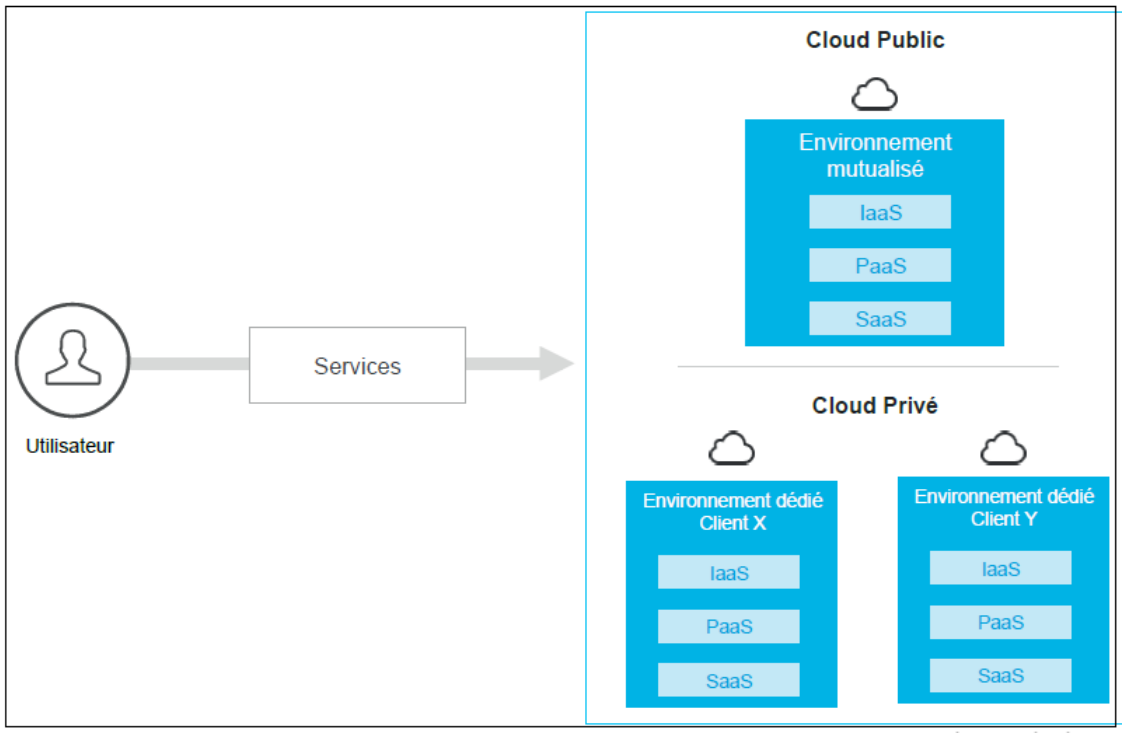


FIGURE 1.12 – Cloud computing

1.6 Big data et Hadoop

La vague du Big Data n'a été rendue possible que par une démocratisation des outils rendant de plus en plus accessible le traitement massif de données. [9]

Doug Cutting. Un grand nom dans la sphère Big Data . Un homme qui inspire respect et administration pour avoir été l'un des fondateurs de la fabuleuse éposée du Big Data.

FIGURE 1.13 – Doug Cutting le créateur de Hadoop[4]

Hadoop a été créé en 2004 et le nom "Hadoop" était initialement celui d'un éléphant en peluche, jouet favori du fils de Doug Cutting l'inventeur de Hadoop **(qui apparaît dans la Figure 1.13)** [4], qui voulait agrandir la taille de l'index de son moteur Open Source Nutch. Le terme ne désigne pas un logiciel particulier mais un environnement technologique dont le but est de réaliser des traitements sur des volumes massifs de données.

Son fonctionnement se base sur le principe des grilles de calcul : répartir l'exécution d'un traitement sur des grappes de serveurs c'est-à-dire plusieurs ordinateurs indépendants.

La grande innovation de Hadoop réside dans cette distribution de l'information. Les architectures plus traditionnelles adossent le traitement de données à une grappe unique. L'étude de l'institut IDC16 souligne que l'écrasante majorité (98 %) des entreprises portant des projets Big Data ont recourt à Hadoop. Néanmoins, le prix pour la migration de ses bases de données sur Hadoop reste un frein : 45 % des entreprises interrogées ont dû dépenser entre 100.000 $ et 500.000 $ et 30 % d'entre elles, plus de 500.000 $.

Troquer une architecture basée sur un entrepôt de données pour un projet Hadoop représente donc un coût elevé. Néanmoins, cette dernière technologie est en moyenne cinq fois moins chère qu'un datawarehouse classique. Ce chiffre comprenant le matériel, le logiciel et le déploiement de l'infrastructure. Sans compter qu'une plateforme Big Data stocke environ cinq fois plus d'informations qu'un datawarehouse traditionnel [2].

1.7 Cycle de vie du Big Data

Dans un projet Big Data ces étapes constituent normalement la majeure partie du travail dans un projet réussi. Dans cette section, nous allons jeter un peu de lumière sur chacune de ces étapes du cycle de vie des données volumineuses.

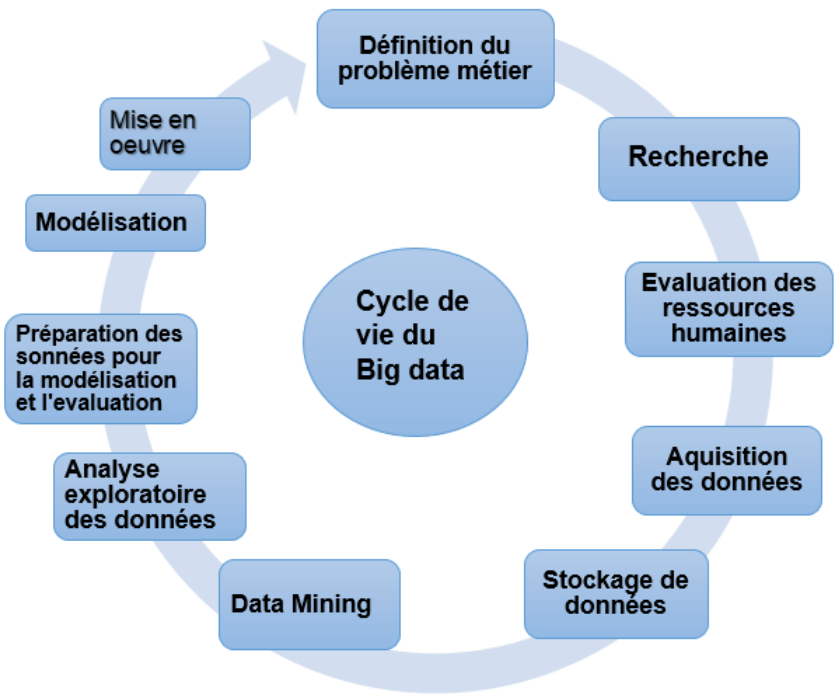


FIGURE 1.14 – Cycle de vie du Big data

• **Définition du problème métier :** c'est une étape non triviale d'un grand projet de données de définir le problème et d'évaluer correctement combien de gain potentiel il peut avoir pour une organisation. Il semble évident de le mentionner, mais il faut évaluer quels sont les gains et les coûts attendus du projet.

• **Recherche** : Analyser ce que d'autres entreprises ont fait dans la même situation. Cela implique de rechercher des solutions raisonnables pour une entreprise, même si cela implique d'adapter d'autres solutions aux ressources et aux exigences de l'entreprise. À ce stade, une méthodologie pour les étapes futures devrait être définie.

• **Évaluation des ressources humaines :** Une fois le problème défini, il est raisonnable de poursuivre l'analyse si le personnel actuel est capable de mener à bien le projet. Les équipes BI traditionnelles ne sont peut-être pas en mesure de fournir une solution optimale à toutes les étapes, il faut donc envisager de démarrer le projet s'il est nécessaire d'externaliser une partie du projet ou d'embaucher plus de personnes.

• **L'acquisition des données :** Cette étape est la plus importante , définit quel type de profils serait nécessaire pour fournir le produit de données résultant. La collecte de données est une étape non-triviale du processus ; cela implique normalement la collecte de données non structurées provenant de différentes sources.

Exemple :

écrire un robot d'exploration pour récupérer des avis sur un site Web. Cela implique de traiter du texte, peut-être dans différentes langues nécessitant normalement beaucoup de temps à compléter.

• **Data Minging** : Une fois les données récupérées, par exemple, sur leWeb, elles doivent être stockées dans un format facile à utiliser. Pour continuer avec les exemples de commentaires, supposons que les données sont extraites de différents sites où chacun a un affichage différent des données.

• **Stockage de données :** Cette étape du cycle est liée à la connaissance des ressources humaines en termes de leurs capacités à mettre en oeuvre différentes architectures. Les versions modifiées des entrepôts de données traditionnels sont

toujours utilisées dans des applications à grande échelle.

Exemple :

Tradata et IBM offrent des bases de données SQL capables de gérer des téraoctets de données.

Des solutions Open Source telles que PostgreSQL et MySQL sont toujours utilisées pour des applications à grande échelle.

• **L'analyse exploratoire des données :** Une fois les données nettoyées et stockées de manière à pouvoir en extraire les informations, la phase d'exploration des données est obligatoire. L'objectif de cette étape est de comprendre les données, ce qui est normalement fait avec des techniques statistiques et de tracer les données. C'est une bonne étape pour évaluer si la définition du problème est logique ou faisable.

• **Préparation des données pour la modélisation et l'évaluation :** Cette étape consiste à remodeler les données nettoyées récupérées précédemment et à utiliser un prétraitement statistique pour l'imputation des valeurs manquantes, la détection des valeurs aberrantes, la normalisation, l'extraction des caractéristiques et la sélection des caractéristiques.

• **La modélisation :** L'étape précédente aurait dû produire plusieurs ensembles de données pour la formation et les tests, par exemple, un modèle prédictif. Cette étape implique d'essayer différents modèles et d'anticiper la résolution du problème commercial(faible productivité). En pratique, il est normalement souhaitable que le modèle donne un aperçu de l'activité. Enfin, le meilleur modèle ou la meilleure combinaison de modèles est sélectionné pour évaluer ses performances sur un ensemble de données abandonné.

• **la mise en oeuvre :** A ce stade, le produit de données développé est implémenté dans le pipeline de données de l'entreprise. Cela implique la mise en place d'un schéma de validation pendant que le produit de données fonctionne, afin de suivre ses performances. Par exemple, dans le cas de la mise en oeuvre d'un modèle prédictif, cette étape impliquerait l'application du modèle à de nouvelles données et, une fois la réponse disponible, évaluer le modèle.

1.8 Bases de données NoSQL (Not Only SQL)

Le terme NoSQL désigne une catégorie de systèmes de gestion de base de données destinés à manipuler des bases de données volumineuses pour des sites de grande audience. Les bases données NoSQL sont scalables, elles permettent de traiter les données d'une façon distribuée. Parmi les avantages du NoSQL on trouve :

— Leurs performances ne s'écroulent jamais quel que soit le volume traité. Leur temps de réponse est proportionnel au volume ;
— Elles se migrent facilement. En effet, contrairement aux SGBDR classiques, il n'est pas nécessaire de procéder à une interruption de service pour effectuer le déploiement d'une fonctionnalité impactant les modèles des données ;
— Elles sont facilement scalable. A titre d'exemple, le plus gros cluster de NoSQL fait 400 To, tandis qu'Oracle sait traiter jusqu'à une vingtaine de Téraoctet (pour des temps de réponse raisonnable).

1.8.1 Typologie des base de données NoSQL

Il en existe 4 types distincts qui s'utilisent différemment et qui se prêtent mieux selon le type données que l'on souhaite y stocker.[17]

a- **BDD Clé/Valeur**

Les BD NoSQL fonctionnant sur le principe Clé-Valeur sont les plus basiques que l'on peut trouver.

• Elles fonctionnent comme un grand tableau associatif et retourne une valeur dont elle ne connaît pas la structure ;

• Les données sont simplement représentées par un couple clé/valeur ;

• La valeur peut être une simple chaîne de caractères, ou un objet sérialisé.

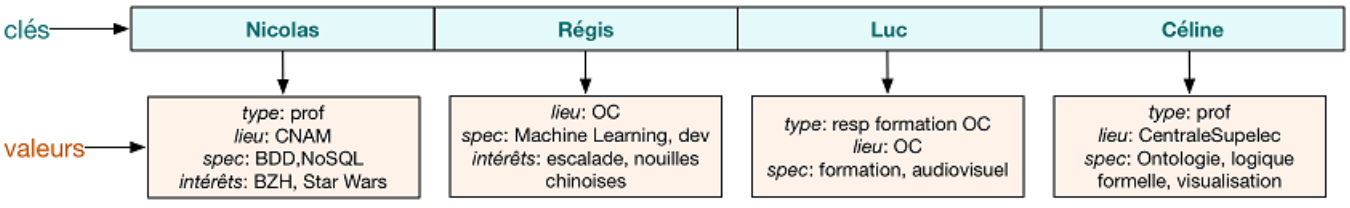


FIGURE 1.15 – Base de donnée Clé/Valeur

b- **BDD orienté Document**

Elles sont basées sur le modèle "clé-valeur" mais la valeur est un document en format semi-structuré hiérarchique de type JSON ou XML (possible aussi de stocker n'importe quel objet, via une sérialisation). Elles stockent une collection de "documents" .Les systèmes NoSQL orientés documents les plus connus sont **CouchDB** d'Apache, **RavenDB** (destiné aux plateformes .NET/Win-dows avec la possibilité d'interrogation via LINQ) et **MongoDB**.

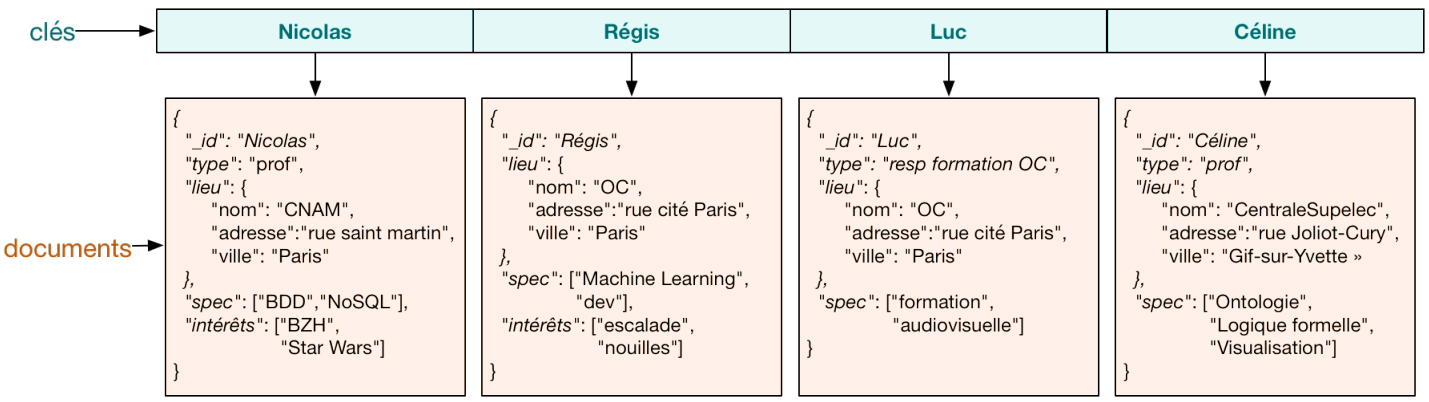


FIGURE 1.16 – Base de donnée orienté Document

c- **BDD orienté Colonnes**

Les données sont stockées par colonne, non par ligne, on peut facilement ajouter des colonnes aux tables, c'est un modèle proche d'une table dans un SGBDR mais ici le nombre de colonnes :

- Est **dynamique .**
- Peut **varier d'un enregistrement à un autre**, ce qui évite de retrouver des colonnes ayant des valeurs NULL.

• Les systèmes NoSQL orientés colonnes les plus connus sont principalement **HBase**, implémentation Open Source du modèle BigTable développé par Google, et **Cassandra**, projet Apache qui respecte l'architecture distribuée de Dynamo d'Amazon, et le modèle **BigTable** de Google.

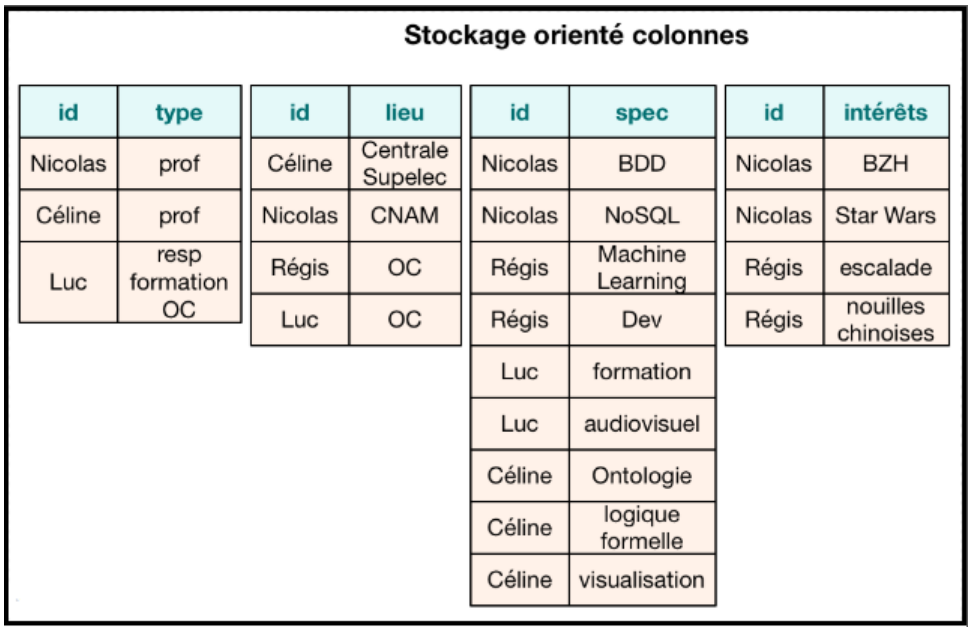


FIGURE 1.17 – BDD orienté colonnes

d- BDD orienté Graphe

Elles permettent la modélisation, le stockage et la manipulation de données complexes liées par des relations non-triviales ou variables

• modèle de représentation des données basé sur la théorie des graphes

• s'appuie sur les notions de noeuds, de relations et de propriétés qui leur sont rattachées.

• Les systèmes NoSQL orientés graphe les plus connus sont : **Neo4J, Infinite Graph, OrientDB**

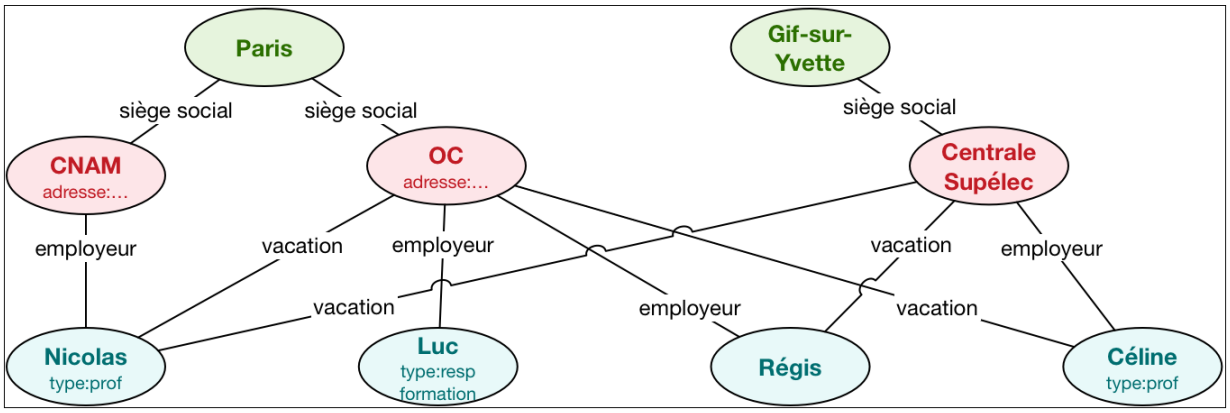


FIGURE 1.18 – BDD orienté graphe

1.9 Les principales bases de données pour les Big data

Le tableau ci dessous présente les principales base de donnée NoSQL [18]

Schéma de données	**Bases**	**Année de lancement**	**Editeur / prestataire de support**	**Positionnement**
Orienté colonnes	***Cassandra***	*2008*	DataSax (ex Riptano)	Adoptée par les géants du web et les start-up, Cassandra permet de gérer de gros volumes de données.
	HBase	*2006*	*Hortonworks*	*Souvent comparé à Cassandra, HBase joue la carte de la très forte volumétrie. Un produit complexe qui exige un gros travail de structuration.*
Orienté documents	**Couchbase**	2010	Couchbase	A la différence de MongoDB, Couchbase dispose d'un outil de requêtage normalisé SQL qui facilite sa prise en mains par des développeurs rompus aux bases SQL.
	MongoDB	2007	MongoDB	Base NoSQL la plus populaire, MongoDB est saluée pour la souplesse de sa structure et sa capacité à répondre à un grand nombre de besoins.
Orienté graphe	**Neo4j**	2010	Neo4j	permettent de modéliser, stocker et requêter en temps réel les données connectées. Ici, on ne parle plus de table ou de document, mais de nœud et de relation.
Clé-valeur	**Redis**	2009	Redis Labs	Base de données en mémoire, Redis privilégie la vitesse d'exécution. En contrepartie, ses capacités de requêtage sont limitées.
	Riak	2009	Basho Technologies	Riak se présente comme une sorte de Redis évolué en étendant les capacités de requêtage via des index secondaires.

TABLE 1.1 – Comparatif des bases de données NoSQL

1.9.1 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", estime Rudi Bruchez. Au-delà des géants du web, Cassandra peut, selon lui, convenir aux jeunes pousses en raison de ses capacités de dimensionnement. "Les start-up n'ont pas de visibilité sur le succès que rencontrera ou non leur application et donc sur la taille de leur base de données", pointe l'expert.

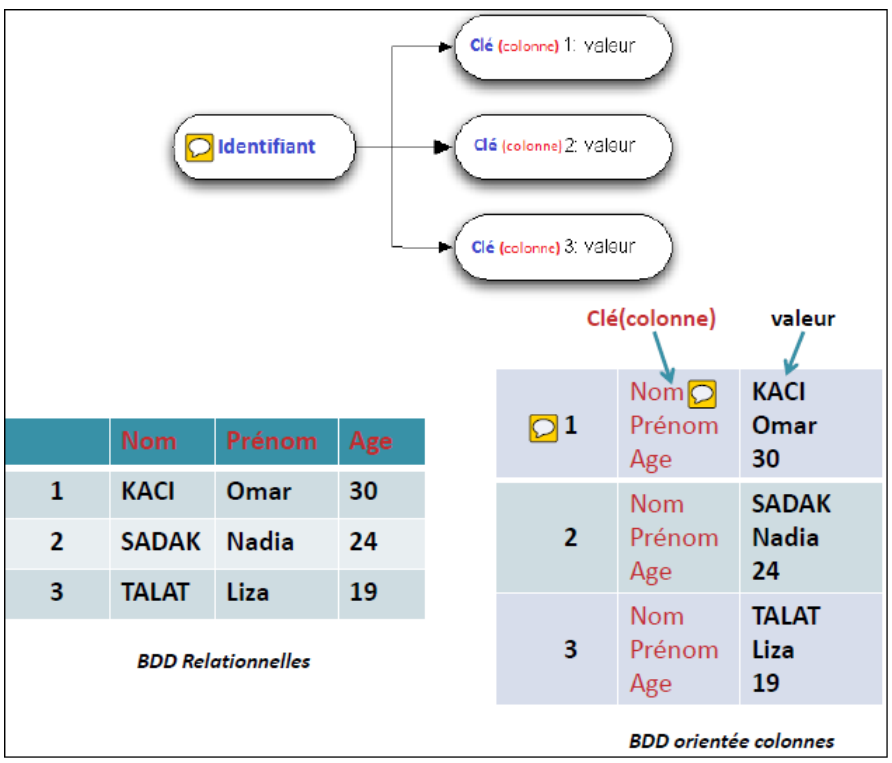


FIGURE 1.19 – Cassandra

1.9.2 Couchbase, un outil de requêtage "SQL like"

Lancée 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).

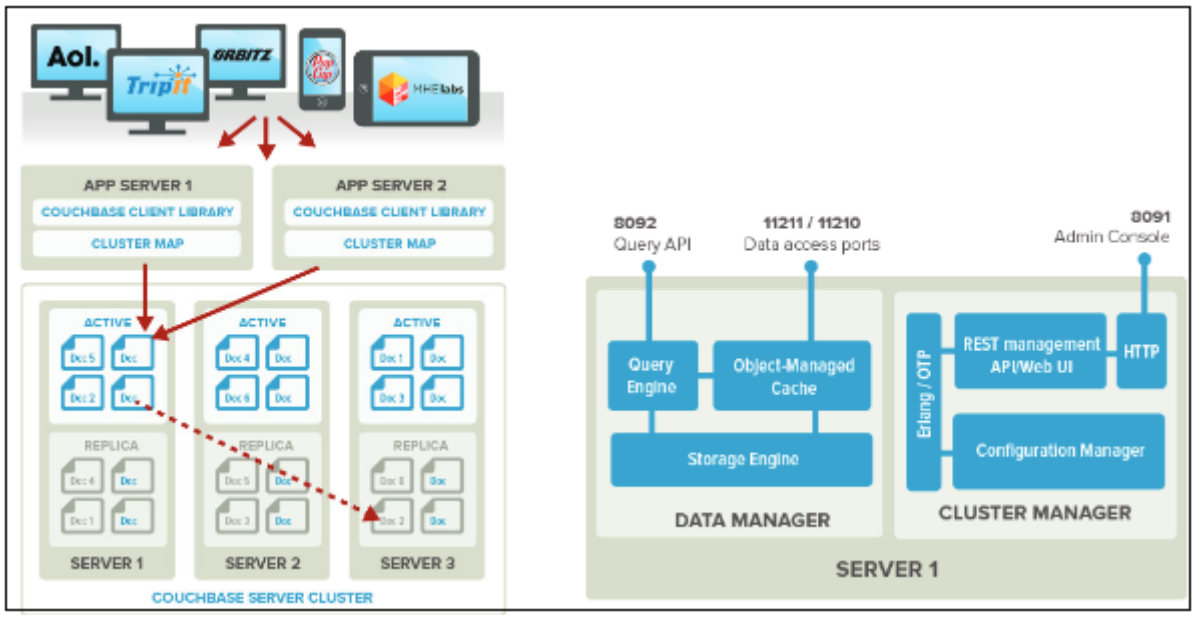


FIGURE 1.20 – Couchbase

1.9.3 Elasticsearch, 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.

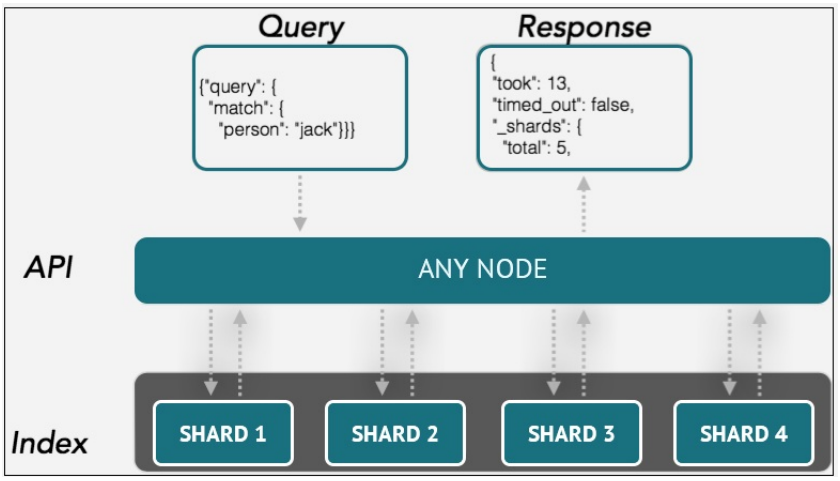


FIGURE 1.21 – Le fonctionnement d'une requête Elasticsearch

1.9.4 HBase, pour les très gros volumes

.Autre rejeton 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. Pour Christophe Parageaud, 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.

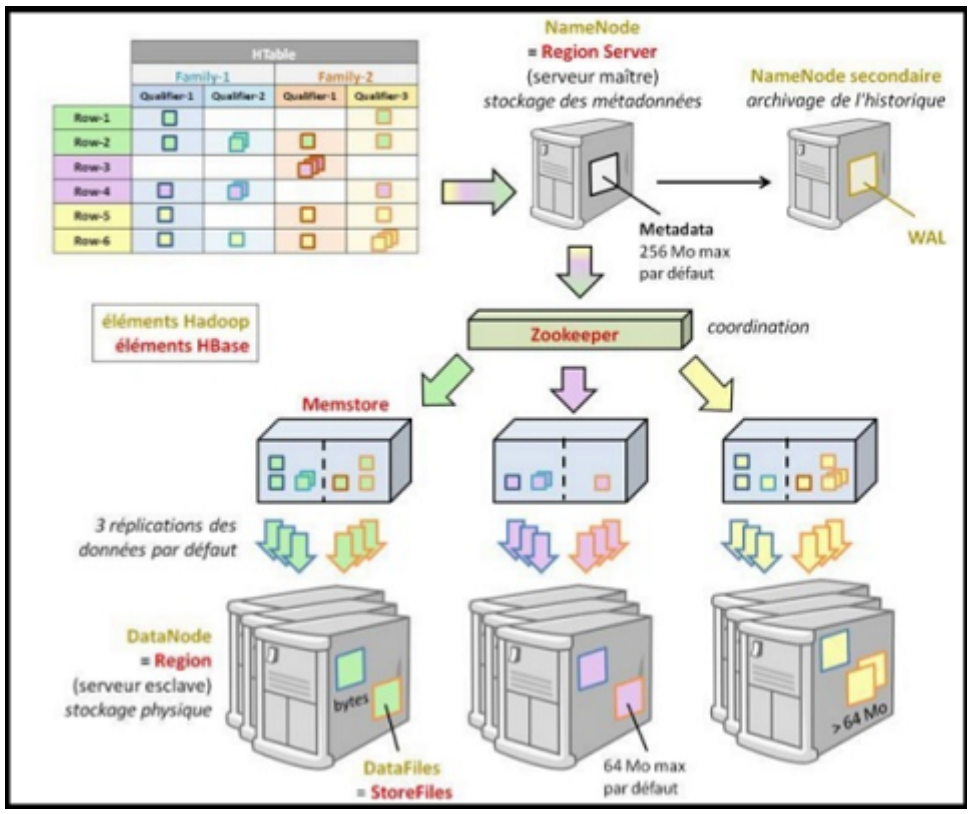


FIGURE 1.22 – Architecture HBase

1.9.5 MongoDB, la plus populaire

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 préstructurer les données, il suffit de créer des collections d'y mettre des éléments Json sans avoir besoin de dire comment les organiser", observe Rudi Bruchez. Pour Christophe Parageaud, "MongoDB est une base générique qui répond à 80% des besoins couverts par une base relationnelle traditionnelle, à l'exception du transactionnel."

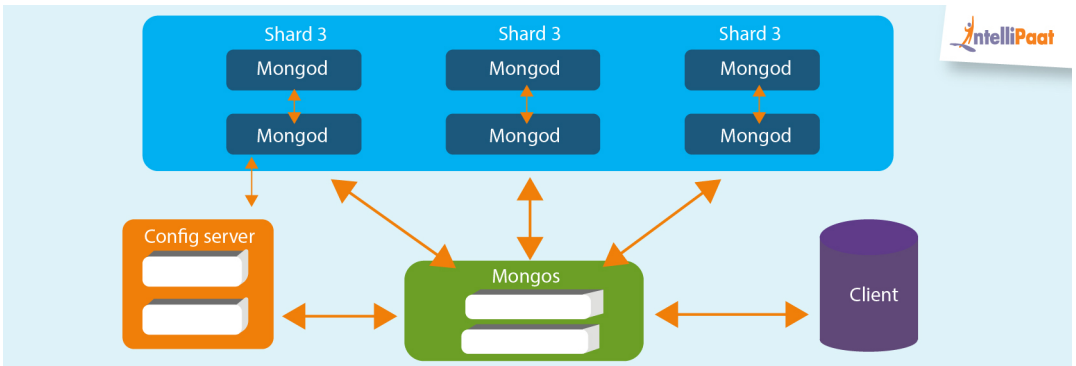


FIGURE 1.23 – L'architecture de la base de données MongoDB

1.9.6 Redis, pour la vitesse

Sorte de Memcached en plus évolué, Redis est une base de données en mémoire, 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 cette sélection à gérer le transactionnel. Lancé en 2009, le projet est sponsorisé par VMware.

```
1   # Assignation de chaîne "bonjour" à la clé "cleTexte"
2   redis> SET cleTexte bonjour
3   OK
4   # Récupération de la valeur de la clé "cleTexte"
5   redis> GET cleTexte
6   "bonjour"
7
8   # Incrémentation d'un compteur
9   redis> INCR compteur
10  (integer) 42
11  # Suppression du compteur
12  redis> DEL compteur
13  (integer) 1
14
15  # Création d'une liste avec insertion en fin de liste
16  redis> RPUSH liste "Hello"
17  (integer) 1
18  # Insertion en fin de liste
19  redis> RPUSH liste "World"
20  (integer) 2
```

FIGURE 1.24 – Quelques commandes Redis en console

1.9.7 Riak, pour la 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 ses chevaux de bataille. 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).

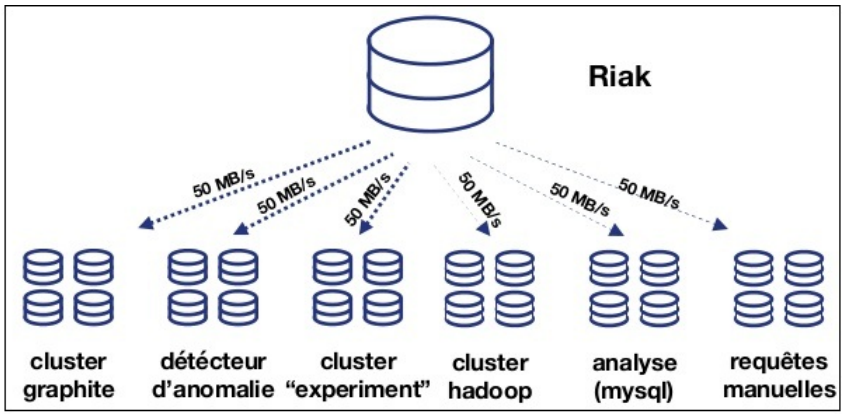


FIGURE 1.25 – Riak

Conclusion

Nous avons abordé dans ce premier chapitre les principes des Big Data, ces caractéristiques, son fonctionnement ainsi que les différents domaines dans lesquels elles sont utilisées.

On a aussi recensé les différents modèles de bases de données NoSQL qui existent, actuellement, dans le marché en accentuant sur les solutions les plus populaires.
L'Internet of Things (IoT ou Internet des Objets) constitue une nouvelle source de données pour le Big Data. elles sont deux technologies interconnectées et indissociables.Nous présenterons l'IoT plus en détails dans le prochain chapitre.

Chapitre 2

L'Internet des objets

Introduction

Internet des objets est un réseau mondial d'objets qui repose sur l'idée que tous les objets peuvent être connectés un jour à Internet, ces objets sont adressables de maniére unique. Tout objet, y compris (des ordinateurs, des capteurs, des RFID [1] et des téléphones mobiles) seront en mesure d'émettre de l'information et éventuellement de recevoir des commandes. IdO ouvre la voie vers une multitude de scénarios basés sur l'interconnexion entre le monde physique et le monde virtuel , Cependant, comme d'autres concepts , celui-ci fait face à un nombre de problématiques qui nécessitent d'être étudiées pour permettre àl'Internet des objets d'atteindre son plein potentiel .[19]

Dans ce chapitre, nous présentons l'IDO ou bien IoT (Internet of things) définition, domaine d'application , son architecture ,le fonctionnement de l'IoT, ainsi que les vulnérabilités et les menaces relatives à son déploiement, et nous consacrons par la suite le reste du chapitre à la définition de bases , et quelques notions utilisées dans le domaine de la sécurité.

2.1 Évolution de l'Internet

L'évolution d'Internet a connu quatre phases distinctes : La connectivité, l'économie en réseau, l'expérience en collaboration et l'internet des objets : [20]

1. RFID (l'identification par radio-fréquence)est une technologie qui permet de mémoriser et de récupérer des informations à distance grâce à une étiquette qui émet des ondes radio.

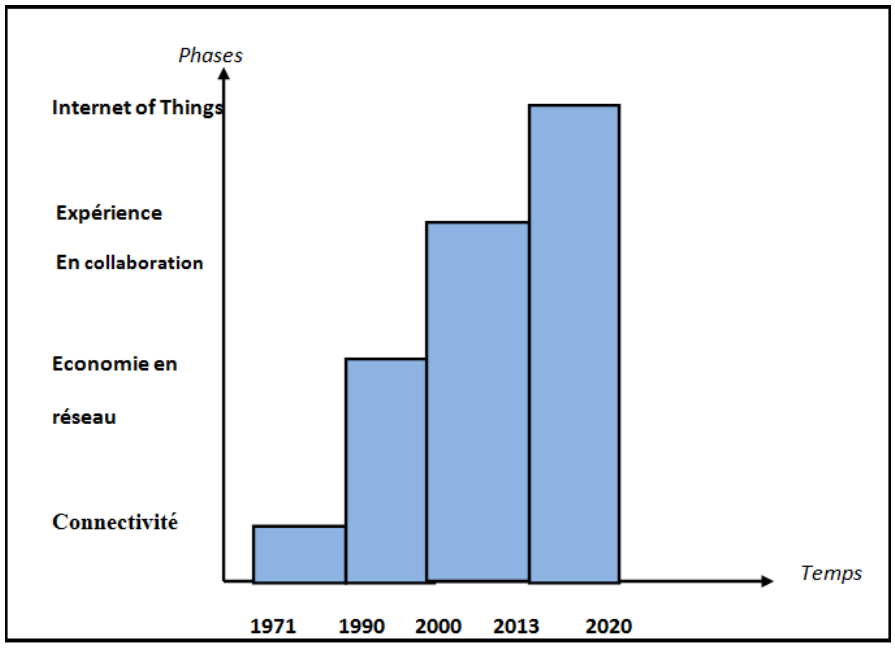


FIGURE 2.1 – Les phases de l'évolution de l'internet

• **Phase 1 :Connectivité :** accés numérique aux informations ,cette phase se caractérise par :

- l'apparition de la messagerie éléctronique et l'envoi du 1er courriel par RAM TOMLINSON qui était (QWERTYUIOP).

- Début de la recherche sur le web.

• **Phase 2 : Economie en réseau :**numérisation des processus métiers commerce éléctronique chaine d'approvionement.

• **Phase 3 :Experience en collaboration :**Elle se caractérise par la génération des medias sociaux, de la mobilité, de la vidéo et de Cloud Computing.

• **Phase 4 :Internet of Things :** cela consiste à connecter les personnes, les processus, les données et les objets.

Concepts de base et définitions de l'internet des objets (IoT)

Dans cette section nous allons donner quelques concepts se rapportant à l'IoT à savoir :

1**. l'internet des objet**,

2. **l'internet of everything**

3. **Ses pilliers.**

2.1.1 l'Internet des objet (IoT)

Définition 1

L'internet des objets (IdO) est une infrastructure dynamique d'un réseau global. qui permet d'interconnecter des objets (physiques ou virtuels) grâce aux technologies de l'information et de la communication interopérables.

- **D'un point de vue conceptuel,** l'Internet des objets affecte, à chaque objet une identification unique sous forme d'une étiquette lisible par des dispositifs mobiles sans fil, afin de pouvoir de communiquer les uns avec les autres. Ce réseau crée une passerelle entre le monde physique et le monde virtuel
- **D'un point de vue technique,** l'IdO consiste l'identification numérique directe et normalisée (adresse IP, protocole http...) d'un objet physique grâce à un systéme de communication sans fil (puce RFID, Bluetooth ou WiFi) . [21]

Définition 2 :

Selon l'Union internationale des télécommunications," l'Internet des objets (IdO) 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 ". [21]

2.1.2 Ido vers l'internet of everything (IoE)

D'aprés la société cisco la convergence entre les réseaux des personnes ,des processus,des données et des objets ,l'Ido va vers l'internet of Everything (IoE) ,ou (Internet du Tout connecté) c'est un internet multidimensionnel qui combine les champs de l'IdO et du Big data.

2.1.3 Les pilliers de l'IoE

L'IoT repose sur quatre piliers visant à rendre les connexions réseau plus efficaces et plus utiles qu'auparavant, à savoir **(les personnes, les processus, les données et les objets.)** Les informations issues de ces connexions conduisent à des décisions et des actions qui créent de nouvelles possibilités, des expériences plus riches et des opportunités économiques sans précédent, et ce, pour les utilisateurs, les entreprises et les pays.[22]

FIGURE 2.2 – Les piliers de l'IoE

a- Les objets :

• **Définition** :

Les objets connectés sont dotés d'une technologie intégrée qui leur permet d'interagir avec des serveurs internes et leur environnement externe.

Ces Objets sont capables de communiquer avec un autre objet (souvent un smartphone, une tablette ou un ordinateur) par l'intermédiaire d'une plate-forme réseau sécurisée, fiable et disponible. Cette communication permet à l'objet d'envoyer ou de recevoir des informations via une connexion Internet(d'ou l'internet des objets ou l'Internet of Things IoT). L'intérêt de cette interactivité est de pouvoir récupérer des informations, d'en tirer des statistiques, de créer des règles ...etc. [23]

Exemple :

La montre connectée : *Elle relève des informations (nombre de pas, rythme cardiaque, ...etc.) pour les envoyer sur un smartphone. Ce dernier nous montre ensuite*

les résultats sous forme de statistiques et nous donne des conseils personnalisés en fonction de nos performances.

- **Exemples d'objets connectés (les objets connectés du marché)**

Les gens utilisent chaque jour de plus en plus des périphériques mobiles pour communiquer entre eux et effectuer des tâches quotidiennes, comme se renseigner sur la météo ou effectuer des opérations bancaires en ligne. À l'avenir, de nombreux objets présents dans la maison seront également connectés à Internet et il sera par conséquent possible de les contrôler et de les configurer à distance.Il existe également de nombreux périphériques connectés dans le monde extérieur et qui peuvent fournir des informations pratiques, utiles, voire même essentielles. Nous présentons dans ce qui suit quelques exemples d'obje connectés :

D\penalty \@M :/d/Desktop/ryma/ryma/image/smart.PNG ***Les Smartphones*** *peuvent se connecter à Internet presque partout. Ils réunissent les fonctions de nombreux autres appareils différents :téléphone, appareil photo, récepteur GPS, lecteur multimédia, ordinateur à écran tactile.*	D\penalty \@M :/d/Desktop/ryma/ryma/image/montre.PNG *Une* ***smartwatch*** *peut se connecter à un Smartphone pour transmettre à l'utilisateur les alertes et les messages qu'il reçoit. D'autres fonctions, telles que la prise du pouls et le comptage des pas, comme un podomètre,calories dépensées, la tension artérielle,ils peuvent aider à contrôler l'état de santé.*
D\penalty \@M :/d/Desktop/ryma/ryma/image/lunette.PNG ***Les lunettes Google***, *véritable ordinateur miniature, sont dotées d'un tout petit écran qui renseigne la personne qui les porte et d'un pavé tactile pour naviguer dans l'interface. Elles permettent de recevoir des notifications pour les messages électroniques et les SMS ; de prendre des photos ou de filmer avec la caméra intégrée. Les informations sont projetées sur l'écran transparent et viennent se superposer à l'environnement réel.*	D\penalty \@M :/d/Desktop/ryma/ryma/image/appareil.PNG ***Les appareils électroménagers*** *tels que les réfrigérateurs, les fours et les chauffe-eau peuvent être connectés à Internet. Ainsi, le propriétaire de la maison peut les allumer ou les éteindre, contrôler l'état de l'appareil et également recevoir des alertes en fonction des conditions prédéfinies.*
D\penalty \@M :/d/Desktop/ryma/ryma/image/voiture.PNG ***Les voitures*** *modernes peuvent se connecter à Internet pour accéder aux cartes,à du contenu audio et vidéo ou aux informations sur une destination. Elles peuvent même envoyer un SMS ou un e-mail en cas de tentative de vol ou appeler les secours en cas d'accident. Ces voitures peuvent également se connecter aux Smartphones et aux tablettes pour afficher des informations sur les différents systèmes du moteur, fournir des alertes relatives à l'entretien ou indiquer l'état du système de sécurité.*	D\penalty \@M :/d/Desktop/ryma/ryma/image/dispositif.PNG ***Les dispositifs médicaux*** *tels que les pacemakers, les pompes à insuline et les systèmes de monitoring des hôpitaux renseignent ou alertent les patients ou le personnel médical lorsque les signes vitaux atteignent des niveaux spécifiques.*

TABLE *2.1 – Exemples des objets connectés*

b- Les Données

Les données sont une valeur affectée à tout ce qui nous entoure.Toutefois, les données seules peuvent être sans signification. Lorsqu'on interprète des données, par exemple en les corrélant ou en les comparant, elles deviennent alors plus utiles. Ces données utiles constituent désormais des informations. Une fois ces informations appliquées ou comprises, elles deviennent de la connaissance.[23]

• Gestion des données :

Les ordinateurs ne disposent généralement pas de la sensibilité contextuelle et de l'intuition des êtres humains. Par conséquent, il est important de tenir compte des trois états de données suivants : structurées ,semi structurée et non structurées.[24]

1. Données structurées

Les données structurées sont des données qui sont entrées et mises à jour dans des champs fixes au sein d'un fichier ou d'un enregistrement. Elles sont facilement entrées, classées, interrogées et analysées par un ordinateur.

Par exemple, *lorsqu' on entre un nom, une adresse et des données de facturation sur un site Web, on crée des données structurées. Afin de minimiser les erreurs et de faciliter l'interprétation des données par l'ordinateur, un format spécifique est requis lors de l'acquisition de ces données.*[24]

ID_PERSONNE	NOM	PRENOM	AGE
1	Einstein	Albert	64
2	Eiffel	Gustave	65
3	Cage	Nicolas	45

FIGURE 2.3 – Exemple Données Relationnelles

2. Données semi-structurées

On appelle donnée semi-structurée une donnée dont le schéma n'est pas défini a priori. Par exemple, il peut s'agir d'une page HTML, d'un site Web tout entier ou encore d'un document XML . [25]

```xml
<?xml version = "1.0" encoding="UTF-8" standalone="yes"?>
<document>
  <employee>
      <name>Alex</name>
      <age>22</age>
  </employee>
  <employee>
      <name>Bob</name>
      <age>24</age>
  </employee>
  <employee>
      <name>Emily</name>
      <age>32</age>
  </employee>
</document>
```

3. **Les données non-structurées**

Les données non structurées ne possèdent pas l'organisation que l'on retrouve dans les données structurées. Elles sont des données brutes. Il n'est par conséquent pas possible d'identifier leur valeur. Les données non structurées ne présentent pas de moyen défini permettant de les saisir, de les regrouper et de les analyser. [24]

Exemple : *Word, PDF, texte....*

c- Les personnes

Les personnes doivent être connectées,les données seules ne servent à rien. Un grand nombre de données auxquelles personne ne peut accéder ne sert à rien. L'organisation de ces données et leur transformation en informations utilisables permettent aux personnes de prendre des décisions en meilleure connaissance de cause et d'adopter les mesures appropriées. Cela crée de la valeur économique dans une économie activée par l'IoT, c'est pourquoi les personnes en constituent l'un des quatre piliers. Elles sont au coeur de tout système économique. Elles interagissent en tant que producteurs et consommateurs.

L'objectif étant d'améliorer le bien-être par la satisfaction des besoins des êtres humains, qu'il s'agisse de connexions de personne à personne (P2P), de machine à personne (M2P) ou de machine à machine (M2M), toutes les connexions, ainsi que les données générées à partir de celles-ci, sont utilisées pour créer de la valeur ajoutée pour les personnes.

Remarque :

Les informations transforment le comportement d'après John Chambers ' L'IoT n'est pas du tout une question de technologie. Il concerne la manière avec laquelle nous modifions les vies des gens. ' , John Chambers, PDG de Cisco Systems

La valeur est une mesure des avantages offerts par un système économique, ce sont les personnes qui déterminent la valeur des offres par le biais d'un système d'échange. Il est important de souligner que si les données et les analyses sont importantes, c'est le jugement des personnes qui transforme les données en opinions, et les opinions en valeur de l'IoT.

L'IoT permet l'obtention d'informations précises et opportunes, susceptibles de provoquer une modification du comportement humain, et ce, au bénéfice de l'ensemble des personnes.,Il facilite les commentaires qui permettent aux individus de prendre des décisions en toute connaissance de cause, diminuant ainsi les différences entre les résultats souhaités et réels. Cela porte le nom de boucle de rétroaction. Une boucle de rétroaction fournit des informations en temps réel, basées sur le comportement actuel, puis délivre des informations exploitables afin de modifier ce comportement. [26]

d- Les processus

Le quatrième pilier concerne les processus. Les processus jouent un rôle important dans la manière dont les autres piliers, à savoir les objets, les données et les personnes, interagissent afin de fournir de la valeur dans le monde connecté de l'IoT. Internet a révolutionné la manière avec laquelle les entreprises gèrent leurs chaînes d'approvisionnement ainsi que celle avec laquelle les consommateurs effectuent leurs achats. Bientôt, nous disposerons d'une visibilité sur les processus jamais atteinte auparavant. Cela fournira des opportunités permettant de rendre ces interactions plus rapides et plus simples. Avec le processus adéquat, les connexions deviennent pertinentes et ajoutent de la valeur, car la bonne information est livrée à la bonne

personne au bon moment, et de la manière la plus appropriée.Les processus facilitent les interactions entre les personnes, les objets et les données.

Aujourd'hui, l'IoT rassemble tous ces éléments en combinant des connexions de machine à machine (M2M), de machine à personne (M2P) et de personne à personne (P2P). [27]

2.2 Architecture de l'IoT

IoT est un concept technologique et une architecture qui regroupe des technologies déjà disponibles. L'architecture IoT peut également être appelée en tant que modèle piloté par les événements. Les visionnaires ont également réalisé que cet écosystème IoT a des applications commerciales dans les domaines de l'automatisation des lignes d'usine et d'assemblage, du commerce de détail, des soins médicaux / préventifs, de l'automobile et plus encore.

À partir du niveau inférieur, le flux de données est généré à partir de n'importe quelle objets grâce à des capteurs qui sont envoyés vers le Cloud via la passerelle de communication pour l'analyse, ce qui s'avère être une information utile, comme le montre la figure suivante :

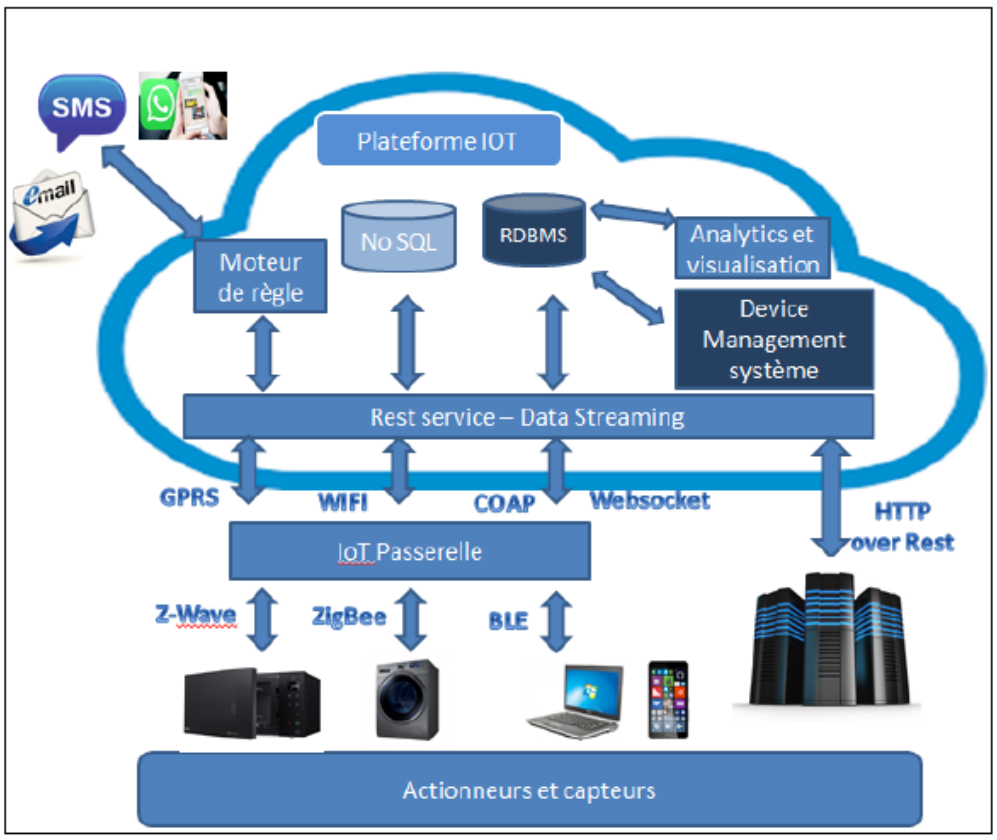


FIGURE 2.4 – Architecture de l'IoT

1. Les capteurs ou Actionneurs :

Un capteur est un dispositif capable de transformer une grandeur physique (telle que la température, la pression, la lumière, etc.) en une autre grandeur physique manipulable. [28]

Exemple :

un microphone est un capteur qui permet de transformer une onde sonore en un signal électrique .

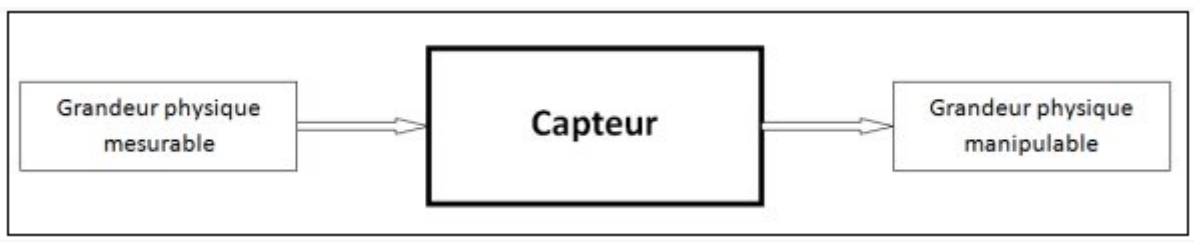


FIGURE 2.5 – Schéma d'un capteur

Les capteurs de l'IoT sont considérés comme un noeud qui va collecter des informations et les envoyer au monde extérieur, via des protocoles de communication - Bluetooth, BLE, ZigBee, Z-Wave, Wi-Fi ou par communication filaire. Ces noeuds transmettront les données à un périphérique appelé **Gateway** .[28]

2. La passerelle (Gateway) IoT :

La passerelle agit comme un pont entre ces objets IoT et Internet. Les passerelles peuvent se connecter aux périphériques IoT qui communiquent via des protocoles spécifiques, stocker et analyser les informations, puis les envoyer aux serveurs cloud pour traitement et analyse.[29]

Les passerelles IoT non seulement permettent d'abstraire le moyen de communication mais fournissent également le canal sécurisé nécessaire à la transmission de ces données.

Les passerelles exécutent généralement des systèmes d'exploitation en temps réel (RTOS) ou une forme de Linux pour piloter leurs systèmes. Le cryptage au niveau matériel et logiciel est intégré directement dans la passerelle pour fournir un canal sécurisé pour la communication.

3. La plateforme cloud et Big Data Analytics :

Les protocoles supportant la plate-forme cloud sont : GPRS, Wi-Fi, CoAP, Websocket, RESTful, etc. Le cloud permettra à IoT de fournir une puissance de calcul, un stockage et une mise en réseau élastiques. Les données massives générées par IoT peuvent être analysées dans le cloud avec des solutions Big Data pour obtenir des informations et des schémas d'utilisation et de comportement des machines et des humains.

Cette intelligence d'entreprise nous permet à son tour de prédire la croissance prochaine de la demande de données et de déployer des ressources supplémentaires en conséquence. Ces modèles sont ensuite analysés, et s'ils ne sont pas pertinents, les informations sont envoyées à l'utilisateur, pour contrôler et surveiller leurs appareils (allant du thermostat d'ambiance aux moteurs à réaction et lignes d'assemblage) à distance. Ces applications transmettent les informations importantes sur des appareils portables et aident à envoyer des commandes à des appareils intelligents.
Cependant, le flux de communication peut également être inversé lorsque l'utilisateur ou bien fabricant souhaite actionner un objet :

- Utilisateur ou fabricant donnera une entrée sous la forme de SMS, Push, Email, Call, ...etc.Cette information est transmise à Internet, c'est-à-dire le Cloud .
- Le Cloud traite ensuite l'information identifie l'objet particulier à travers l'adresse IP et pousse l'information à travers les protocoles de communication à la passerelle.
- La passerelle déclenchera l'actionneur qui sera responsable du contrôle et du déplacement du système ou de l'objet.[18]

2.3 Domaines d'applications de l'Internet des Objets

le cabinet McKinsey [4] évalue l'impact de l'Internet des Objets à 6,2 trillions (milliards de milliards) de dollars en 2025. Le cabinet estime qu'il y aura 9 sujets touchés par l'Internet des Objets :

1. La domotique (automatisation et sécurité de la maison).
2. L'automobile (autonomie, maintenance et assurance).
3. La ville (santé publique et transports).
4. L'externe (transports, logistique et navigation).
5. L'humain (sport et santé).
6. La construction (optimisation des travaux, sécurité et santé).
7. Le commerce de grande consommation (marketing).
8. Les usines (gestion des opérations et des équipements) .
9. Le lieu de travail (sécurité et énergie) .

Désormais, nous pouvons interagir avec des objets réels. Pour la première fois, nous pouvons vivre dans les villes intelligentes pleines de capteurs qui nous aident à améliorer notre mode de vie et des machines qui parlent à d'autres machines. Il existe de nombreux domaines d'application inhérents à l' Internet des Objets. Nous nous contentons de décrire seulement quelques domaines d'applications de l'Internet des Objets [30] :

1. La domotique ou maison connectée

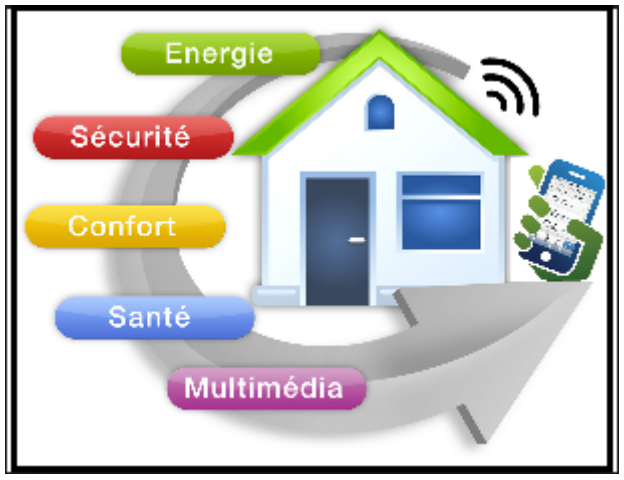


FIGURE 2.6 – La maison connectée

Appelée également domotique, la maison intelligente est en train de se normaliser. Une étude du cabinet Juniper Research prévoit d'ailleurs un accroissement de 200 % du nombre d'objets connectés à l'intérieur des habitations d'ici fin 2021.

Outre les objets de divertissement comme les télés intelligentes ou les enceintes connectées, la domotique a pensé également la sécurité et l'économie d'énergie au sein de l'habitat :

— centrale domotique : contrôle et programmation de différentes interventions à l'intérieur du foyer.
— capteurs d'informations (système d'alarme, variations de température, etc.).
— actionneurs, qui permettent la programmation et le contrôle des différents appareils électroniques du foyer, même à distance. [31]

2. L'automobile

la voiture connectée participe grandement au renforcement de la sécurité routière. La révolution numérique a offert à l'industrie automobile :

— Boitier d'appel d'urgence autonome.
— tableau de bord synchronisé avec le smartphone.
— développement d'applications sur les plateformes dédiées. . ...

La voiture d'aujourd'hui se transforme en véritable ordinateur qui conduit peu à peu à la voiture autonome, comme celle actuellement en essai chez Google. Si nos véhicules ne sont pas encore capables de se conduire tout seuls, ils n'en deviennent pas moins de plus en plus autonomes grâce à un système d'automatisation de certaines tâches de conduite (allumage des phares, park assist, freinage automatique).[31]

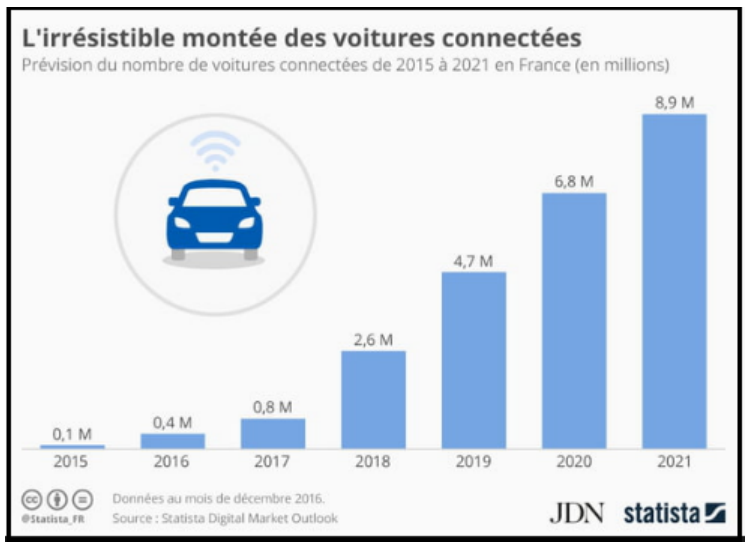


FIGURE 2.7 – Prévisions du nombre de voitures connectées de 2015 à 2021 en France(en millions) [29]

3. Textile connecté (Smart sensing)

Un autre domaine d'application est celui du textile connecté où un textile est doté de micro-capteurs intégrés, capables d'effectuer le monitoring d'individus : température, fréquence cardiaque, vitesse et accélération, géolocalisation [4].

FIGURE 2.8 – le textile connecté chez Cityzen ***[6]***

Exemple

l'entreprise française CityzenSciences ***(voir sa publicité la Figure 2.8)*** *qui est*

spécialisée dans la conception, la création, le développement de textiles connectés , est le pilote d'un projet industriel textile, Smart Sensing, menée par un consortium d'entreprises, fortement soutenu par la BPI5 France. Son but : créer, concevoir et développer l'industrie française du vêtement connecté. Le sport professionnel et amateur est la cible première du projet Smart Sensing .[32]

4. Réseau électrique intelligent (Smart grid)

Le smart grid ou (réseau de distribution et de gestion d'énergie intelligent) est un réseau électrique communicant qui intègre les NTIC (Nouvelles Technologies de l'Information et de la Communication) dans son fonctionnement. Cela permet d'établir des interactions entre les réseaux d'électricité et les bâtiments auxquels ils sont raccordés afin d'améliorer la gestion de l'énergie et de rationnaliser la consommation.

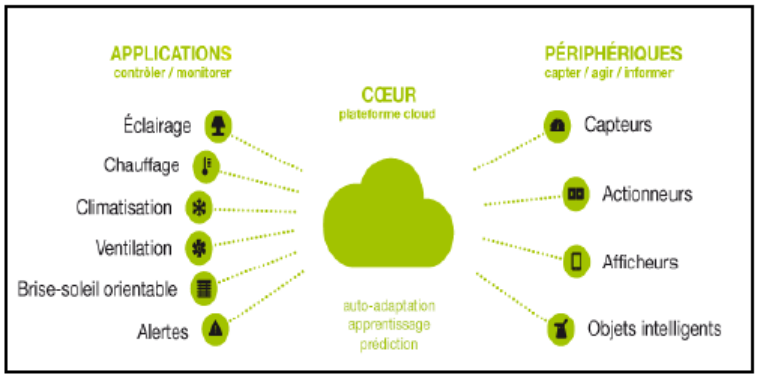


FIGURE 2.9 – L'écosystème Hemis [21]

A titre d'exemple : Ubiant la start-up Française qui équilibre le confort et l'énergie dans la maison, a développé Hemis **(voir la Figure 2.9)** qui permet de réduire la consommation d'énergie des bâtiments tout en maximisant le bien-être de leurs occupants.

Hemis est un écosystème qui comprend son environnement en analysant en temps réel un grand nombre d'informations collectées : température, humidité, luminosité, CO2, présence humaine. . ., il équilibre entre l'efficacité énergétique et le confort des personnes. Hemis est une solution adaptée à la gestion de l'électricité, du gaz, de l'eau et des énergies renouvelables. [4] [33]

5. Ville intelligente (Smart city)

La ville intelligente (ou Smart city) cherche à concilier les piliers sociaux, culturels et environnementaux à travers une approche systémique qui allie gouvernance participative et gestion éclairée des ressources naturelles afin de faire face aux besoins des institutions, des entreprises et des citoyens. [34]

FIGURE 2.10 – Ville Intelligente

Songdo, en Corée du Sud **(voir la Figure 2.10)**, est présentée comme le parfait exemple : une ville construite de toute pièce où tous les bâtiments se trouvent dans un rayon de 6 km^2. Des capteurs et des ordinateurs sont également placés le long des routes et des édifices pour évaluer et ajuster la consommation d'énergie. Songdo a coûté 35 milliards de dollars (25 milliards d'euros environ) et forme le plus grand projet immobilier privé du monde. Elle devrait être achevée en 2017.

Parmi les autres villes sur la liste, on compte Masdar aux Émirats Arabes Unis [35], où 500 foyers sont alimentés en énergie grâce à des panneaux solaires et aux sources renouvelables et où les voitures sont interdites. A la place, les habitants se déplaceront à vélo, à pieds ou emprunteront les transports en commun. La ville sera aussi le quartier général de l'Agence internationale de l'énergie renouvelable. Vienne s'est également fixé des objectifs pour devenir une ville neutre en carbone d'ici 2020 [4].Il existe beaucoup d'autres Smart-X Applications comme le smart grid, smart environment, smart agriculture...etc.

2.4 La sécurité dans l'IoT

L'augmentation du nombre d'appareils connectés et de la quantité de données qu'ils génèrent accroît la demande de sécurité de ces données.Les attaques des pirates informatiques sont quotidiennes et il semble qu'aucune organisation ne soit à l'abri. Étant donné la facilité avec laquelle il est aujourd'hui possible de voler et d'utiliser de façon malveillante des informations dans le monde connecté, il est naturel de se préoccuper de ce problème, car les personnes, les processus, les données et les objets seront à l'avenir tous connectés au sein de l'IoT. Mais trop souvent, les objets de l'IoT sont la cible de cyber attaques, et il est impératif d'adopter une approche globale de la sécurité de ces objets. On nous aide à :

- Renforcer la sécurité du réseau de capteurs.
- Mettre en oeuvre une authentification renforcée .

2.4.1 Les vulnérabilités les plus fréquemment rencontrées sur les objets connectés

Dans le monde de l'informatique et dans celui des objets connectés plus particulièrement les pirates informatiques tentent d'accéder aux systèmes en utilisant les failles et les vulnérabilités de ce derniers et parmi les vulnérabilités les plus répandues on a :

- **Mises à jour non sécurisées** : absence de chiffrement et de signature pour les mises à jour des micrologiciels.
- **Utilisation de secrets par défaut** : définition de clés et de mots de passe connus en environnement de production.
- **Communications non-sécurisées** : absence ou faiblesse du chiffrement et du contrôle d'intégrité par signature numérique sur les communications.
- **Stockage de données en clair** : absence de chiffrement sur le stockage local des données,
- **Présence des interfaces de débogage** : possibilité de prendre le contrôle

des composants matériels de l'objet.[37]

2.4.2 Les solutions pour se prémunir des attaques ciblant l'IoT [40]

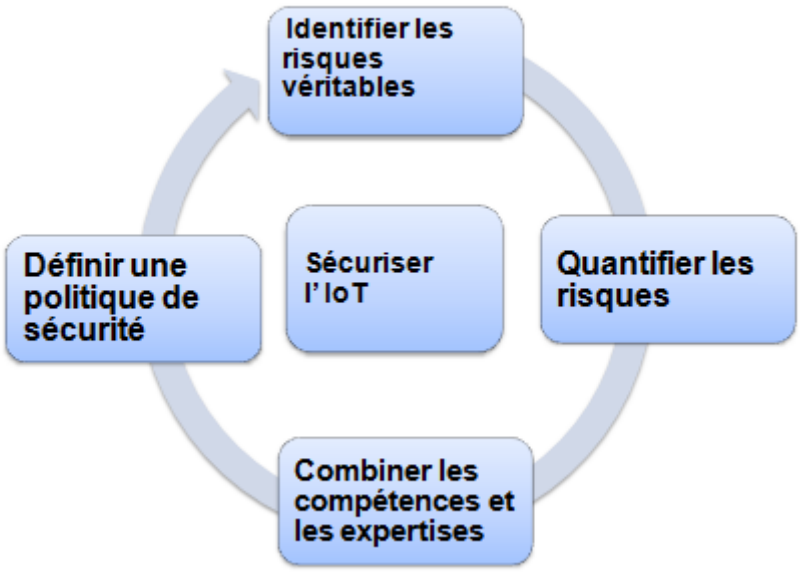


FIGURE 2.11 – Solutions pour se prémunir des attaques ciblant l'IoT

Sécuriser une solution liée à l'Internet des Objets (IoT) exige de nouveaux réflexes dans la façon d'envisager le risque, d'évaluer le coût associé, de constituer une équipe pour les anticiper et y répondre.

- **Identifier les risques véritables**

  La cybersécurité pour les entreprises a longtemps consisté à **protéger ses infrastructures**. Mais avec l'IoT, elle ne porte plus seulement sur l'information mais sur le contrôle des actions pilotées par les objets connectés. Le détournement de ces fonctions peut avoir **des conséquences graves pour l'entreprise** ou la collectivité qui les a mises en place.

  Exemple :

  En juillet 2015, le groupe Fiat Chrysler Automobiles a été contraint de rappeler 1,4 million de véhicules aux Etats-Unis pour effectuer une mise à jour de leurs systèmes informatiques embarqués, deux chercheurs en sécurité avaient piraté un modèle de véhicule connecté.

• **Quantifier les risques**

Tout risque n'est pas forcément à anticiper : il faut définir le rapport **bénéfice-risque** pour juger de ce qui sera acceptable. Lorsque de nombreux **capteurs sont déployés à des fins de collecte d'information**, pour établir des statistiques

Exemple

On peut décider que l'altération des données transmises par un seul capteur n'a pas d'impact majeur. En revanche, si ces mêmes capteurs servent à piloter des actions concrètes, il faudra ***évaluer le coût d'une attaque****, la perte potentielle associée pour l'entreprise et l'investissement nécessaire pour l'éviter afin de décider ou non de sécuriser la solution.*

• **Combiner les compétences et les expertises**

Chaque solution IoT est spécifique et unique. Pour identifier les risques, il est nécessaire de recourir à des experts, des hackers éthiques et des consultants qui envisageront d'abord tous les risques théoriques. En adoptant une démarche similaire à celle des pirates, un **hacker éthique tentera toutes les cyberattaques possibles** : capter l'information transmise, s'attaquer physiquement à un capteur pour altérer ses composants ou comprendre comment en prendre le contrôle... Le consultant quant à lui pourra évaluer les conséquences de chaque type d'attaque sur les données et leur confidentialité ou toutes les conséquences potentielles (dégradation de la production, arrêt de l'activité,. . .). L'entreprise aura ainsi en sa possession tous les éléments pour prendre une décision et définir sa politique. Lorsqu'un risque est identifié, l'entreprise peut avoir deux postures différentes :

Définir une politique de sécurité

Lorsqu'un risque est identifié, l'entreprise peut avoir deux postures différentes :

1. apporter une réponse technique, en protégeant la solution IoT pour supprimer

le risque.

2. apporter une réponse légale, en se déchargeant de toute responsabilité.

2.4.3 Categorie des menaces dans la sécurité de l'Internet des objets (IoT) [39]

Nous pouvons classer les attaques potentielles contre l'Internet des objets dans trois catégories principales, en fonction de leur cible : les attaques contre un appareil, les attaques contre la communication entre les appareils et les maîtres d'opération, et les attaques contre les maîtres d'opération. Pour protéger les utilisateurs finaux et leurs appareils connectés, nous devons faire face à ces trois types d'attaques de l'IoT.

- **Attaques contre les appareils de l'Internet des objets**

Pour un agresseur potentiel, un appareil représente une cible idéale pour plusieurs raisons. D'abord, un grand nombre de ces appareils auront une valeur inhérente en raison de la simple nature de leur fonction. Une caméra de sécurité connectée, par exemple, pourrait divulguer des informations stratégiques sur la sécurité d'un emplacement donné si elle était compromise.

Solution :

Sécuriser l'Internet des objets nécessite que des certificats d'identité soient attribués à tous les appareils dans leur centre de fabrication, afin de pouvoir établir leur provenance et faciliter l'authentification avec les services et les autres appareils.

- **Attaques contre les communications**

Une méthode d'attaque courante consiste à surveiller et à modifier les messages pendant leur transmission. Le volume et la nature confidentielle des données qui traversent l'environnement de l'Internet des objets (IoT) rendent les attaques de ce type particulièrement dangereuses, puisque les messages risquent d'être interceptés, capturés ou manipulés en transit. Toutes ces menaces mettent en péril la fiabilité des informations et des données transmises, ainsi que la fiabilité générale de l'infrastructure dans son ensemble.

Solution :

Lorsque des données confidentielles sont transmises par l'intermédiaire du Cloud et de l'environnement IoT, elles doivent être chiffrées pour éviter les interceptions. De la même manière, les données stockées doivent être chiffrées en toute transparence et de façon homogène pour empêcher leur vol.

- **Attaques contre le maître d'opération des appareils**

Pour chaque appareil ou service dans l'Internet des objets (IoT), il doit y avoir un maître d'opération. Le rôle du maître d'opération consiste à distribuer et à gérer les appareils, ainsi qu'à faciliter l'analyse des données. Les attaques contre les maîtres d'opération (qui incluent les fabricants, les prestataires de services Cloud et les fournisseurs de solutions IoT), ont le potentiel d'infliger le plus de dommages. Ces entités auront la responsabilité de gérer de vastes quantités de données, dont certaines seront d'une nature extrêmement confidentielle. Ces données ont également de la valeur pour les fournisseurs IoT investis dans l'analytique, qui représente un actif professionnel stratégique vital, et une vulnérabilité concurrentielle grave en cas de divulgation.

Solution :

signature du code des mises à jour du logiciel/micrologiciel, à l'aide de certificats numériques. Toutes les communications avec les appareils sur le terrain doivent également utiliser des certificats SSL.

Conclusion

Dans ce chapitre nous avons présenté d'une façon générale la technologie l'IoT. Nous avons défini c'est quoi l'IoT, Ensuite, nous avons présenté un modéle d'architecture, ses domaines d'application ainsi que son importance. Aussi, nous avons décrit en détail les problémes relatifs à son déploiement, puis nous avons mis l'accent sur quelques concepts liés à la sécurité. Le chapitre suivant sera consacré à l'étude des bases de données NoSQL.Les données issue de l'internet des objets sont tellement vaste que les structure de données traditionnelles ne peuvent traiter, d'ou le passage vers le Nosql,ce dernier sera l'objet du chapitre suivant.

Chapitre 3

Les Bases de Données NoSQL

Introduction

Les bases de données en général et le modèle relationnel en particulier existent depuis plusieurs décennies. Ce modèle bien très puissant, représentait la solution parfaite pour les différents acteurs dans le domaine de gestion des données, néanmoins ces architectures ont atteints leurs limites pour certains services ou sites manipulant de grandes masses de données, tels que Google, Yahoo, Facebook. En effet ce genre de sites possède plusieurs millions voire des milliards d'entrées dans leurs bases de données distribuées sur plusieurs machines, produits par des millions d'utilisateurs éparpillés partout dans le monde. Pour des raisons de fiabilité, ces bases de données sont dupliquées pour que le service ne soit pas interrompu en cas de panne. Pour répondre à ces nouveaux besoins, plusieurs réflexions de conception de nouvelles architectures ont été imposées, et qui ont émergé plusieurs modèles et solutions pour la gestion de données, principalement le NoSQL .

La première partie de ce chapitre consistera à expliquer les bases de données relationnelles et leurs limites,ensuite nous allons voir les base de données de type Nosql, Nous allons également voir de quelle manière ces type de base de données fonctionnent et sur quelles fondements elles s'appuient, ainsi que les avantages qu'une base de données de type NoSQL peut avoir en comparaison à une base de données relationnel standard.

3.1 Les systèmes relationnels et leurs limites atteintes

Dans cette section nous allons donner quelques concepts se rapportant aux bases de données relationnelles à savoir : **sa définition,propriété ACID ainsi que leur limites .**

3.1.1 Définition des bases de données relationnelles

Le modèle relationnel a été inventé par Edgar Codd [41], un scientifique d'IBM, dans les années 1970 et a été utilisé par IBM, Oracle, Microsoft, et d'autres. Il était, et reste encore largement utilisé aujourd'hui en raison de sa simplicité et de ses solides fondations mathématiques.

Dans ce modèle, les données sont stockées dans une table, l'ensemble de ces tables constituent une base de données relationnelle ayant un schéma. Ce dernier est une représentation structurelle du contenu de la base de données, définissant les tables, les champs des tables et les relations entre les deux. Les données sont stockées sous forme de tableaux ou chaque colonne correspond à un attribut spécifique et chaque ligne correspond à un enregistrement ou un tuple. Les données de chaque table peuvent être lues, supprimées et mises à jour en utilisant le langage standard pour les bases de données relationnelles SQL (Structured Query Language). La gestion, le stockage, la mise à jour, le partage, la cohérence et la sécurité des données sont assurés par un SGBD Relationnel (RDBMS).

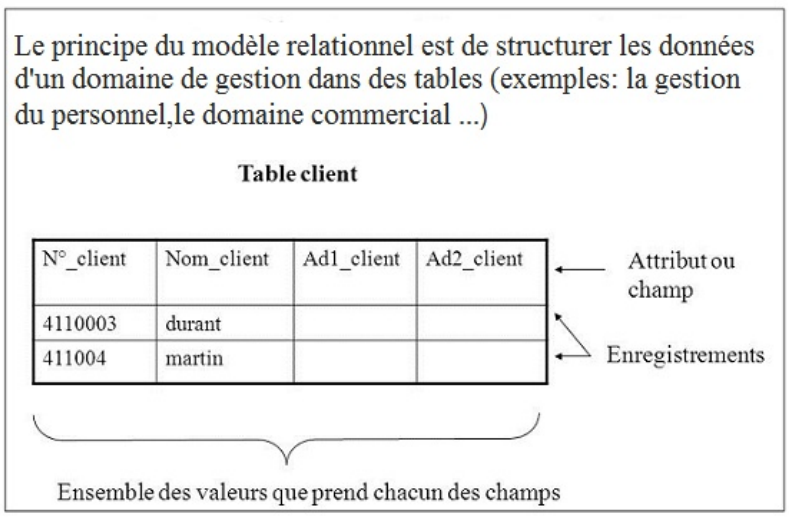


FIGURE 3.1 – Base de donnée relationnelle

Face à l'évolution informatique dans la dernière décennie et notamment les grands volumes de données échangés, les SGBDR classiques ont montré de très grandes faiblesses en matière

de scalabilité et en montée en charge, principalement à cause du respect des propriétés ACID, ce qui a obligé les grands acteurs du web à chercher d'autre solutions.

3.1.2 Les propriétés ACID

Selon la théorie des bases de données, les propriétés ACID sont les quatre principaux attributs d'une transaction de données. Il s'agit là d'un des concepts les plus anciens et les plus importants du fonctionnement des bases de données, il spécifie quatre buts à atteindre pour toute transaction. Ces buts sont les suivants :

- **L'Atomicité (Atomicity)**

  Lorsqu'une transaction est effectuée, toutes les opérations qu'elle comporte doivent être menées à bien : en effet, en cas d'échec d'une seule des opérations, toutes les opérations précédentes doivent être complètement annulées, peu importe le nombre d'opérations déjà réussies. En résumé, une transaction doit s'effectuer complètement ou pas du tout. Voici un exemple concret : une transaction qui comporte 3000 lignes qui doivent être modifiées, si la modification d'une seule des lignes échoue, alors la transaction entière est annulée. L'annulation de la transaction est toute à fait normale, car chaque ligne ayant été 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.

- **La cohérence (Consistancy)**

  Avant et après l'exécution d'une transaction, les données d'une base doivent toujours être dans un état cohérent. Si le contenu final d'une base de données contient des incohérences, cela entraînera l'échec et l'annulation de toutes les opérations de la dernière transaction. Le système revient au dernier état cohérent. La cohérence est établie par les règles fonctionnelles.

- **L'isolation (Isolation)**

  La caractéristique d'isolation permet à une transaction de s'exécuter en un mode isolé. En mode isolé, seule la transaction peut voir les données qu'elle est en train de modifier, c'est le système qui garantit aux autres transactions exécutées en parallèle une visibilité sur les données antérieures. Ce fonctionnement est obtenu grâce aux verrous système posés par le SGBD. Prenons l'exemple de deux transactions A et B : lorsque celles-ci s'exécutent en même temps, les modifications effectuées par A ne sont ni visibles, ni modifiables par B tant que la transaction A n'est pas terminée et validée par un **commit** .

- **La Durabilité (Durability)**

  Toutes les transactions sont lancées de manière définitive. Une base de données ne doit pas afficher le succès d'une transaction pour ensuite remettre les données modifiées dans leur état initial. Pour ce faire, toute transaction est sauvegardée dans un fichier journal afin que, dans le cas où un problème survient empêchant sa validation complète, elle puisse être correctement terminée lors de la disponibilité du système [42].

3.1.3 Limites des bases de données relationnelles

Le modèle relationnel est fondé sur des concepts simples qui font sa force en même temps que sa faiblesse. En effet, les systèmes de bases de données relationnelles répondent bien au besoin transactionnel grâce aux propriétés ACID (Atomicité, Cohérence, Isolation, Durabilité). De même, les extensions faites sur ce modèle (tel que les cubes, le schéma en étoile, etc.) permettent de prendre en charge le besoin d'analyse dans les grandes bases consolidées (Data Warehouse). Néanmoins, ces systèmes relationnels ne peuvent pas être déployés dans un environnement Cloud à grande échelle, en conservant les mêmes performances.

1. Problème lié à l'application des propriétés ACID en milieu distribué

Une base de données relationnelle est construite en respectant les propriétés ACID (Atomicité, Cohérence, Isolation, logique du relationnel nuisent fortement aux performances 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 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 fait patienter l'utilisateur durant ce temps [43]

2. Problème de requête non optimale dû à l'utilisation des jointures

Imaginons une table contenant toutes les personnes ayant un compte sur Facebook, soit 1.55 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.[44]

3. Problème lié à la gestion des objets hétérogènes

Le stockage distribué n'est pas la seule contrainte qui pèse à ce jour sur les systèmes relationnel.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 .[44]

D'autre part le modèle relationnel est fondé sur un modèle mathématique solide s'appuyant sur des concepts simples qui font sa force en même temps que sa faiblesse. Nous expliquerons quelques **limites** :

a - La 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 [45].

b - Les types de données

Ces modèles sont limités à des types simples (entiers, réels, chaînes de caractères), les seuls types étendus se limitant à l'expression de dates ou de données financières, ainsi que des conteneurs binaires de grande dimension (BLOB, pour Binary Large OBjects) qui permettent de stocker des images ainsi que des fichiers audio ou vidéos. Ces BLOBs ne sont toutefois pas suffisants pour représenter des données complexes (pas de structure), les mécanismes de contrôle BD sont inexistants, et le langage de requêtes (SQL) ne possède pas les opérateurs correspondant aux objets stockés dans ces BLOBs [45].

c - Le langage de manipulation

Un autre inconvénient est lié au fait que le langage SQL n'est pas un langage complet. Le développement d'une application autour d'une base relationnelle nécessite l'utilisation d'autres langages (Cobol, Java, C,...) et pose le problème de concordance des types entre le SGBD et le langage de développement [46].

d - La pauvreté sémantique

La pauvreté sémantique du modèle relationnel ne permet pas de prendre en compte efficacement les nouveaux besoins liées aux informations multimédia. En particulier, le modèle relationnel est très mal adapté à la gestion de données multivaluées complexes [46].

Et même si on arrive parfois à placer ces données dans des tables, ce n'est pas forcément efficace.

On pourrait imaginer stocker chaque pixel d'une image d'une personne dans une ligne distincte d'une table relationnelle mais il faut alors se poser la question suivante : quel code SQL écrire pour déterminer si l'image représente bien cette personne ? [47].

e - Le partitionnement de données

L'un des problèmes de la normalisation dans un SGBDR concerne la distribution des données et du traitement. S'il y a des données stockées ayant un rapport entre elles, comme des clients, des commandes, des factures, des lignes de facture, etc., dans des tables différentes, des problèmes surgiront en cas de partitionnement de ces données. Pour y remédier, il faut alors s'assurer que les données en rapport les unes avec les autres se trouvent sur le même serveur [48].

3.2 La technologie NoSQL

NoSQL est une combinaison de deux mots : No et SQL. Qui pourrait être mal interprétée car l'on pourrait penser que cela signifie la fin du langage SQL et qu'on ne devrait donc plus l'utiliser. En fait, le(No) est un acronyme qui signifie(Not only), c'est-à-dire en français,(non seulement), c'est donc une manière de dire qu'il y a autre chose que les bases de données relationnelles. [42]

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).

En effet, NoSQL ne vient pas remplacer les BD relationnelles mais proposer une alternative ou compléter les fonctionnalités des SGBDR pour donner des solutions plus intéressantes dans certains contextes. Le NoSQL regroupe de nombreuses bases de données, récentes pour

la plupart, qui se différencient du modèle SQL par une logique de représentation de données non relationnelle. Leurs principaux avantages sont leurs performances et leur capacité à traiter de très grands volumes de données .[44]

3.2.1 Pourquoi le NoSQL ?

C'est une évidence de dire qu'il convient de choisir la bonne technologie en fonction du besoin. Il existe cependant certains critères déterminants pour basculer sur du NoSQL :

- **La taille**

  Le NoSQL est conçu pour supporter un nombre important d'opérations, de données, d'utilisateurs etc... C'est à cause de cette grande masse de données que le système doit devenir distribué. Bien que tous les systèmes NoSQL ne soient pas conçus pour cette problématique, il est possible de la résoudre sans problème.

- **La performance en écriture**

  Celle-ci est l'intérêt principal du géant Google, Facebook, Twitter, gérant des masses des données qui augmentent considérablement chaque année exigeant un temps très important pour stocker ces données. En conséquence, l'écriture se doit être distribuée sur un cluster de machines, ce qui implique du MapReduce, la réplication, la tolérance aux pannes, et la cohérence. [49]

- **Les Types de données flexibles**

  Les solutions NoSQL supportent de nouveaux types de données et c'est une innovation majeure. Les objets complexes peuvent être mappés sans trop de problèmes avec la bonne solution. [49]

- **Les propriétés ACID**

  Bien que ce ne soit pas le but premier du NoSQL, il existe des solutions permettant de conserver certains (voire tous) aspects des propriétés ACID. Se référer au théorème CAP plus bas dans le manuscrit et aux propriétés BASE .[49]

- **La simplicité de développement**

  L'accès aux données est simple. Bien que le modèle relationnel soit simple pour les

utilisateurs finaux, il n'est pas très intuitif pour les développeurs puisque les données sont restituées selon la structure de la base . [49]

• **La scalabilité**

La scalabilité est le terme utilisé pour définir l'aptitude d'un système à maintenir un même niveau de performance face à l'augmentation de charge ou de volumétrie de données, par augmentation des ressources matérielles .[42]

Il y a deux façons de rendre un système extensible : la scalabilité horizontale (externe) ainsi que la scalabilité verticale (interne).

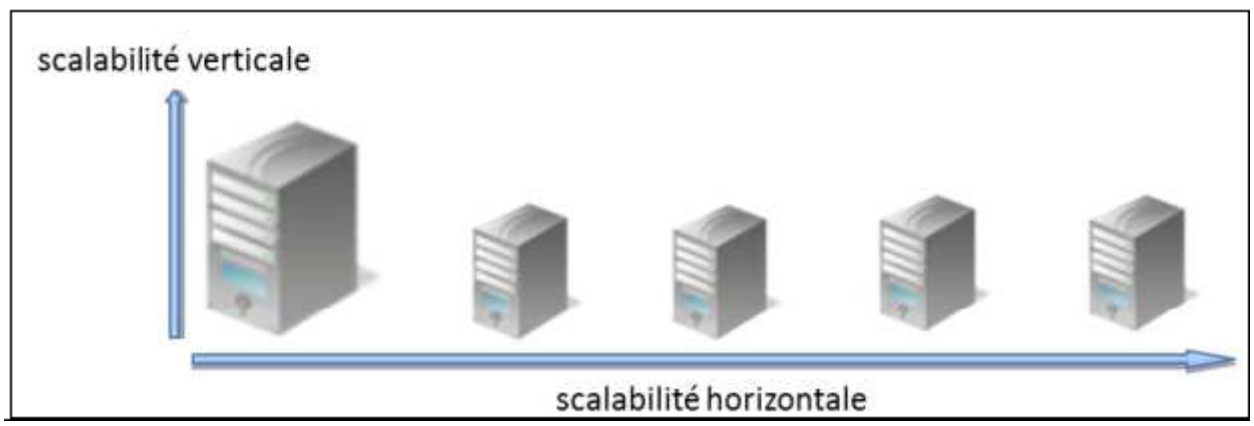


FIGURE 3.2 – Scalabilité horizontale et verticale [6]

1- Scalabilité horizontale

Le principe de la scalabilité parallèle, c'est la raison pour laquelle on parle de croissance externe. On part d'un serveur basique et on rajoute des nouveaux serveurs identiques afin de répondre à l'augmentation de la charge.

2- Scalabilité verticale

Le principe de la scalabilité serveur, comme par exemple le remplacement du CPU par un modèle plus puissant ou par l'augmentation de la mémoire RAM. une machine dans le temps.

Le but des systèmes NoSQL est de renforcer la scalabilité horizontale, les SGBD NoSQL sont basés sur des systèmes innovants permettant la scalabilité hori-

zontale et dont la plupart d'entre eux sont Open Source, ils sont conçus pour répondre à .des besoins spécifiques et assurer une extensibilité extrême sur de très grands ensembles de données .[42]

3.2.2 Le théorème de CAP

Le théorème de CAP est l'acronyme de(Coherence),(Availability) et(Partition tolerance), aussi connu sous le nom de théorème de Brewer.Ce théorème, formulé par Eric Brewer en 2000 et démontré par Seth Gilbert et Nancy Lych en 2002, énonce une conjecture qui définit qu'il est impossible,sur un système informatique de calcul distribué, de garantir en même temps les trois contraintes suivantes :[42]

- **La cohérence (Consistency)**

  Tous les noeuds (serveurs) du système voient exactement les mêmes données au même moment.

- **La disponibilité (Availability)**

Garantie que toute requête reçoive une réponse même si elle n'est pas actualisée.

- **La résistance au partitionnement (Partition tolerance)**

Le système doit être en mesure de répondre de manière correcte toutes requêtes dans toutes circonstances sauf en cas d'une panne générale du réseau.Dans le cas d'un partitionnement en sous-réseaux,chacun de ces sous-réseaux doit pouvoir fonctionner de manière autonome.

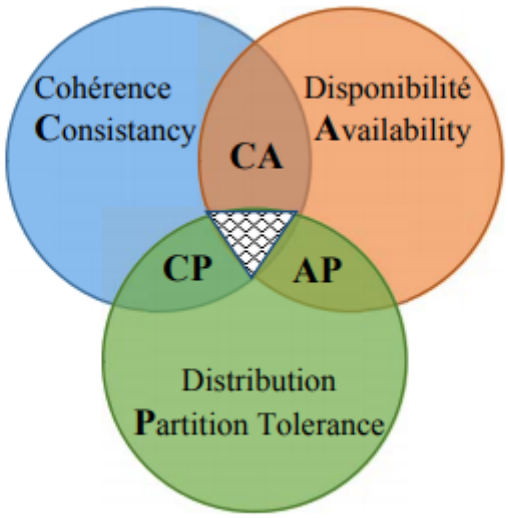


FIGURE 3.3 – Théorème CAP

Dans un système distribué, il est impossible d'obtenir ces 3 propriétés en même temps il faut en choisir 2 parmi les 3 :

1- Marginaliser la tolérance à la distribution (CA) :

le système ne prend pas en considération la distribution des données sur un réseau. C'est le cas typique des SGBDRs.

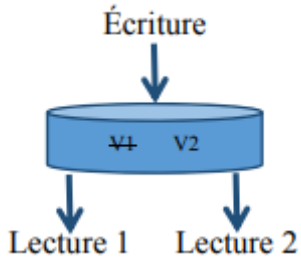


FIGURE 3.4 – CA Cohérence + Disponibilité

Dans la (**Figure 3.4**), les deux requêtes de lecture concurrente sur une même donnée, retournent le même nouveau résultat et sans délai d'attente.

2- Marginaliser la cohérence (AP) :

Dans le cas de la distribution, les données peuvent être sollicitées, mais à cause de la rupture des nœuds, la cohérence n'est pas garantie, parce que les mises à jour sont asynchrones sur le réseau. Cette option s'intéresse à fournir un temps de réponse rapide.

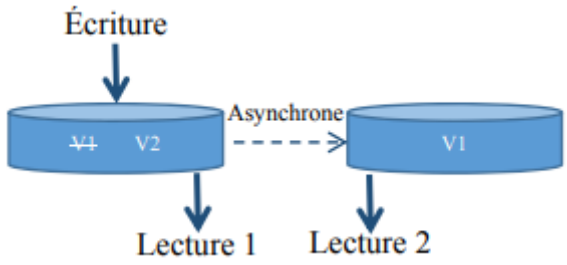


FIGURE 3.5 – AP Disponibilité + Distribution

Dans la(**Figure 3.5**), la lecture1 retourne v2 alors que la lecture2 retourne v1. Cassandra utilise cette option, avec des temps de réponse très appréciables mais avec des résultats non garantis à 100%.

3- Marginaliser la disponibilité (CP) :

Les données ne peuvent être utilisées que si leur cohérence est garantie. Une donnée mise à jour sur un nœud, doit être bloquée sur les autres nœuds jusqu'à la propagation de la nouvelle version sur tout le réseau. Dans un environnement distribué, une base de données prend un temps considérable pour avoir un état cohérent, ce qui rend la disponibilité relative.

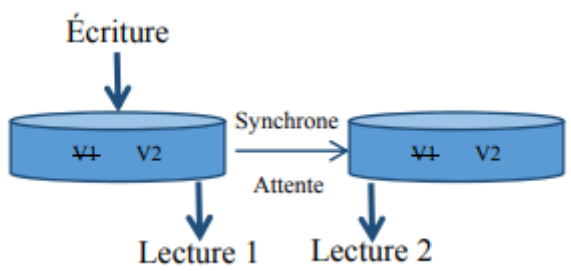


FIGURE 3.6 – CP Cohérence + Distribution

Dans la(**Figure 3.6**) : Les requêtes Lecture1 et Lecture2 attendent la synchronisation pour avoir le résultat v2. Le résultat est cohérent mais avec un délai de latence. MongoDB utilise cette option des BD NoSQL.

L'option de marginaliser la distribution n'est pas réaliste, car de nos jours, il n'est pas pratique, voire inimaginable de travailler dans un environnement non distribué.

3.2.3 Les propriétés BASE

Dans le premier chapitre consacré aux bases de données relationnelles, nous avons rappelé les propriétés ACID auxquelles doivent répondre les SGBD de type relationnel. Les SGBD NoSQL qui, selon le théorème CAP, privilégient la disponibilité ainsi que la tolérance à la

partition plutôt que la cohérence, répondent aux propriétés de BASE. Le principe de BASE est le fruit d'une réflexion menée par Eric Brewer Les caractéristiques de BASE sont fondées sur les limites que montrent les SGBD relationnelles. Voici sa description :[42]

- **Basically Available (Disponibilité basique)**

  Même en cas de désynchronisation ou de panne d'un des noeuds du cluster, le système reste disponible selon le théorème CAP. [42]

- **Soft-state (Cohérence légère)**

  Cela indique que l'état du système risque de changer au cours du temps, sans pour autant que des données soient entrées dans le système.[42]

- **Eventual consistancy (Cohérence à terme)**

  Cela indique que le système devient cohérent dans le temps, pour que pendant ce laps de temps, le système ne reçoit pas d'autres données.[42]

3.2.4 Principaux modèles de bases de données NoSQL

Il existe une diversité d'approches NoSQL classées en quatre catégories. Ces systèmes utilisent des technologies distinctes spécifiques aux différentes solutions :

- Orienté clé-valeur
- Orienté colonnes
- Orientée documents
- Orientée graphes

1- Modèle Orientées Clé / Valeur

La base de données de type clé-valeur est considérée comme la plus élémentaire. Son principe est très simple, chaque valeur stockée est associée à une clé unique. C'est uniquement par cette clé qu'il sera possible d'exécuter des requêtes sur la valeur.
La communication avec la base de données se résume aux opérations basiques qui sont :PUT, GET, UPDATE et DELETE .

Les systèmes NoSQL orientés clé-valeur les plus connus sont Memcached, Amazon's

Dynamo, Redis, Riak et Voldemort créé par LinkedIn. [50]

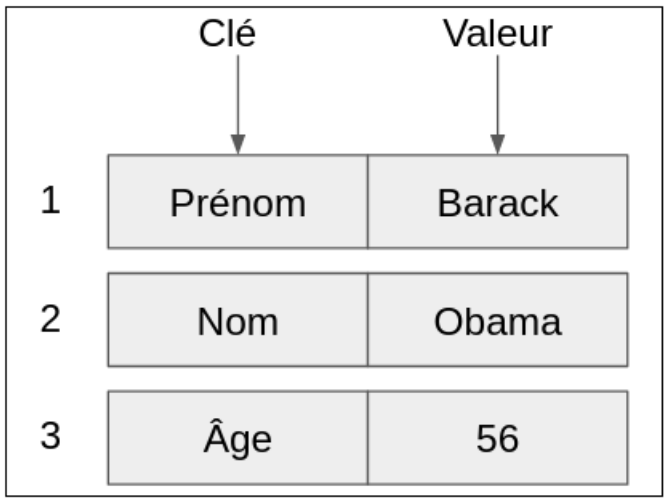


FIGURE 3.7 – Illustration d'une base de données Orientée Clé / Valeur

Ces bases de données sont caractérisées par leur non utilisation du SQL et offrent une meilleure scalabilité horizontale ainsi elles peuvent adopter une architecture distribuée et tolérante aux pannes. En ce qui suit des solutions exemples de ce type :

- **Hadoop :** Créé par Doug Cutting, salarié chez Yahoo,OPEN SOURCE (Fondation Apache) écrit en JAVA,Inspiré de publications Google (2004) (Google Map Reduce Google Filesystem).
- **DynamoDB :** Solution d'Amazon à l'origine de ce type de base. Système ultra scalable basé sur des disques électroniques SSD (Solid State Drive) rapides. Design de type AP selon le théorème de CAP mais peut aussi fournir une consistance éventuelle.
- **Voldemort** : Proposé par Dynamo, c'est une implémentation open-source qui offre une Très grande scalabilité.

2- Modèle orienté documents

Les systèmes de type documentaire sont composés de collections de documents. La représentation en document est une sorte d'extension du concept clé/valeur. La valeur est représentée sous forme de document contenant des données. Des champs et des valeurs associées composent le document. Ces valeurs associées peuvent soit être de type simple (integer, string, date, ...), soit de type composé, c'est-à-dire de plusieurs couples clé/valeur. Bien que les documents soient organisés de manière

hiérarchique à l'image de ce que permettent les fichiers de type XML ou JSON, ces bases sont dites "Schémaless", ce qui signifie sans schéma défini. Cela veut tout simplement dire qu'il n'est pas nécessaire de définir au préalable les champs dans le document : on peut très bien en rajouter en cours de développement. Les documents peuvent être très différents les uns des autres au sein de la base.[50]

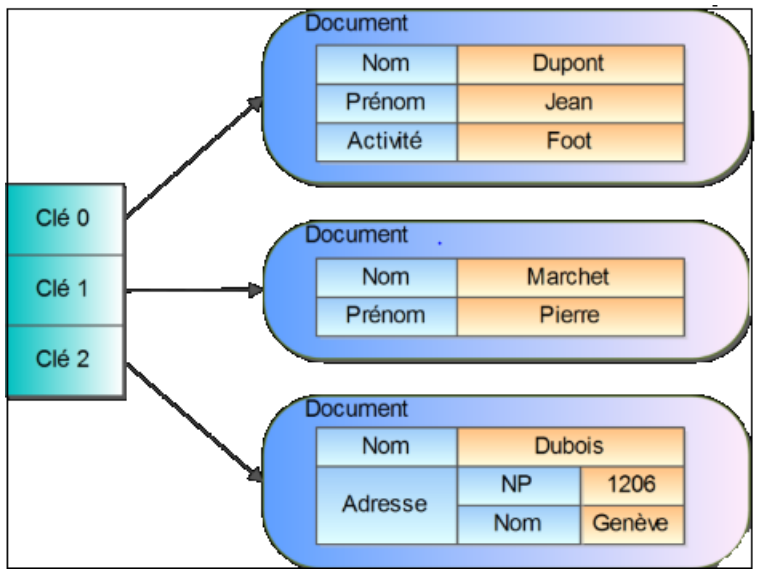


FIGURE 3.8 – illustration d'une base de donnée Orientée Document [8]

Parmi ces solutions :

- **MongoDB :** Développé en C++, API officielles pour beaucoup de langages, Protocole personnalisé BSON, Réplication master/slave et auto-sharding, Licence commerciale et gratuite.
- **CouchDB :** Développé en Erlang, Protocol http, Réplication master/master, Licence Apache.

3- Modèle orienté colonnes

Les bases de données orientées colonnes ont été conçues par les géants du web afin de faire face à la gestion et au traitement de gros volumes de données s'amplifiant rapidement. Ce modèle ressemble à première vue à une table dans un SGBDR à la différence que le nombre de colonnes est dynamique.

En effet, dans une table relationnelle, le nombre de colonnes est fixé dès la création du schéma de la table et ce nombre reste le même pour tous les enregistrements dans cette table. Par contre, avec ce modèle, le nombre de colonnes peut varier d'un enregistrement à un autre, ce qui évite de retrouver des colonnes ayant des valeurs

NULL.[50]

Les concepts de base de cette architecture sont :

- **Column** : L'entité de base qui représente un champ de données. Toutes les colonnes sont définies par un couple clé/valeur.
- **Supercolumn** : Une colonne qui contient d'autres colonnes.
- **Columnfamily** : Considérée comme un conteneur de plusieurs colonnes ou supercolonnes.
- **Row** : L'identifiant unique de chaque ligne de colonne.
- **Value** : Contenu de la colonne qui peut, elle-même, être une colonne

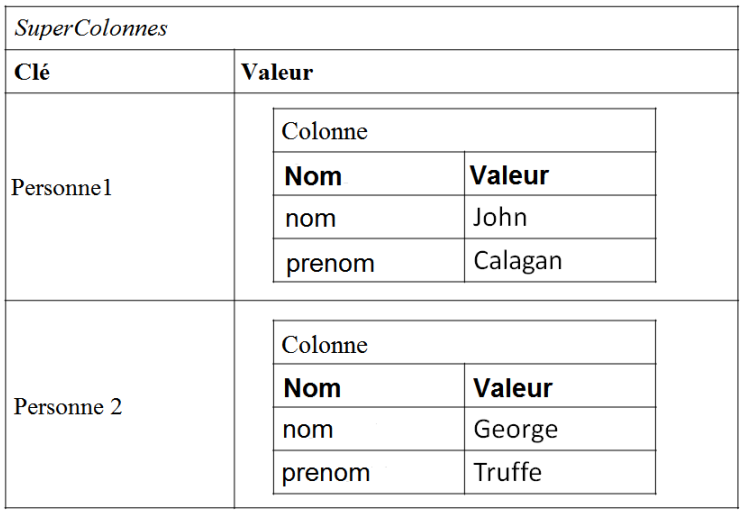


FIGURE 3.9 – Illustration d'une Base de données Orientée Colonne

Ce type de structure permet d'être plus évolutif et flexible ; cela vient du fait qu'on peut ajouter à tout moment une colonne ou une super-colonne à n'importe quelle famille de colonnes, nous citons comme exemple :

- **HBase :** Utilise un API Java. Adopte un design CA.
- **Cassandra :** Beaucoup d'API disponibles. Adopte un design AP avec consistance éventuelle.

4- **Modèle orienté(graphe)**

Ce modèle (figure 3.10) qui repose sur la théorie des graphes, permet de représenter les données sous la forme de graphes. Le modèle s'appuie sur la notion de nœuds, de relations et de propriétés qui leur sont rattachées. Les entités sont alors les nœuds du graphe et les relations que partagent les entités sont alors des arcs qui relient ces entités.

Ce modèle est notamment adapté au traitement des données des réseaux sociaux. Notons que les systèmes NoSQL orientés graphe trouvent un certain intérêt pour des applications dans le domaine du Web Sémantique, dans la gestion de bases de données de triplets RDF (triple-stores), permettant de stocker des connaissances ou ontologies, un triplet étant une arête d'un graphe. [50]

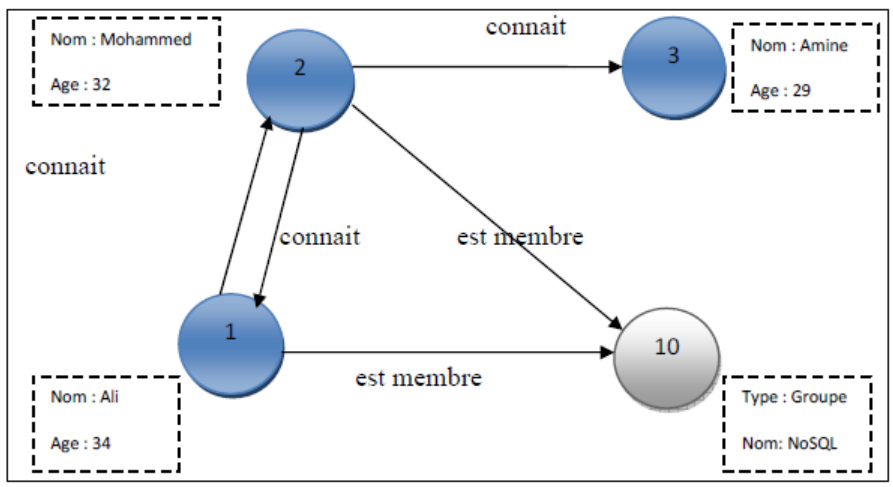


FIGURE 3.10 – Illustration d'une Base de données Orientée Graphe

Les systèmes NoSQL orientés graphe les plus connus :

- **Neo4J :** Développé en Java, supporte beaucoup de langages, propriétés ACID possibles, utilisé dans le monde des réseaux sociaux (Ex : amis sur Facebook) ; Langage de requêtes personnalisé " Cypher ".

3.2.5 Le requêtage NoSQL

Dans le monde du NoSQL il n'y a pas de langage standard comme SQL l'est dans le monde des bases de données relationnelles. L'interrogation des bases de données NoSQL se fait au niveau applicatif à travers principalement la technique dite de(MapReduce [1]).[44] MapReduce est une technique de programmation distribuée très utilisée dans le milieu NoSQL et qui vise à produire des requêtes distribuées. Cette technique se décompose en deux grandes étapes :

1. MapReduce est un Framework de traitement de données en clusters. Composé des fonctions Map et Reduce, il permet de répartir les tâches de traitement de données entre différents ordinateurs, pour ensuite réduire les résultats en une seule synthèse

1. Etape de mapping

Chaque item d'une liste d'objets clé-valeur passe par la fonction map qui va retourner un nouvel élément clé-valeur. Exemple de la fonction map : à chaque couple (UserId, User), on assigne le couple (Role, User). A l'issue de l'étape de mapping, on obtient une liste contenant les utilisateurs groupés par rôle .[44]

2. Etape de Reduce

La fonction reduce est appelée sur le résultat de l'étape de mapping et permet d'appliquer une opération sur la liste. Exemples de fonction reduce :

— Moyenne des valeurs contenues dans la liste
— Comptabilisation des différentes clés de la liste
— Comptabilisation du nombre d'entrées par clé dans la liste L'étape de mapping peut être parallélisée en traitant l'application sur différents noeuds du système pour chaque couple clé-valeur.

L'étape de réduction n'est pas parallélisée et ne peut être exécutée avant la fin de l'étape de mapping. Les bases de données NoSQL proposent diverses implémentations de la technique MapReduce permettant le plus souvent de développer les méthodes map et reduce en Java Scriptou en Java .[44]

3.2.6 Les avantages du NoSQL

a- La scalabilité horizontal

Pendant longtemps, les administrateurs de bases de données ont opté pour la "scalabilité" verticale, au lieu d'avoir misé sur une "scalabilité" horizontale. Ceci vient du fait qu'il est difficile d'adapter cette stratégie avec une base de données

relationnelle.

Aujourd'hui, la rapidité en lecture/écriture ainsi que la haute disponibilité sont devenues des critères indispensables. C'est pourquoi les bases de données NoSQL répondent entièrement à ce besoin. Les performances qu'offre-la "scalabilité" horizontale peuvent même être atteintes avec des serveurs bas de gamme, ce qui rend l'infrastructure plus économique.

b- Gros volume de données

"Big data" Au cours de la dernière décennie, le volume de données à stocker a augmenté de manière massive. Les bases de données relationnelles ont augmenté leurs capacités afin de suivre la tendance. Mais avec cette constante augmentation de volume de données, il est devenu quasi impossible, pour un unique serveur de base de données relationnelle, de répondre aux exigences des entreprises en terme de performance.

Aujourd'hui, ces gros volumes de données ne sont plus un problème pour les SGBD de type NoSQL et même le plus grand des SGBD relationnel ne peut rivaliser avec une base NoSQL.

c- Solution économique

Les bases de données NoSQL ont tendance à utiliser des serveurs bas de gamme dont le coût est moindre afin d'équiper les "clusters", tandis que les SGBD relationnels, eux, tendent à utiliser des serveurs ultra puissants dont le coût est extrêmement élevé. De ce fait, les systèmes NoSQL permettent de revoir à la baisse les coûts d'une entreprise. Cela permet de stocker ainsi que de manipuler plus d'informations à un coût nettement inférieur

d- Plus simple

Les bases de données NoSQL ne sont pas forcément moins complexes que les bases relationnelles, mais elles sont beaucoup plus simples à déployer. La façon dont elles ont été conçues permet une gestion beaucoup plus légère .

e- Modèle de données flexible

Changer le modèle de données d'une base de données relationnelle en production, est un vrai casse-tête, même une petite modification doit être maniée avec précaution et peut nécessiter l'arrêt du serveur pendant la modification ou limiter les niveaux de services.

Les systèmes NoSQL sont plus souples en termes de modèles de données, comme dans les catégories clé/valeur et documentaire. Même les modèles un peu plus stricts comme dans la catégorie orientée colonne permettent d'ajouter une colonne assez facilement.[51]

3.2.7 Inconvénients du NoSQL

a- La maturité

Les SGBD relationnels sont là depuis plus de 30 ans, ainsi, cette pérennité est un signe de réconfort pour les responsables. Les SGBD relationnels sont de nature stable et riche en fonctionnalités. En comparaison, la plupart des SGBD NoSQL sont en pré-production et ont encore beaucoup de fonctionnalités futures à implémenter.
Les spécialistes parlent d'une attente d'une bonne dizaine d'années avant de voir ce concept utilisé fréquemment par des applications critiques.

b- Le support

Les entreprises veulent être rassurées de leurs bases de données et cherchent un support rapide et efficace. Les principaux fournisseurs de SGBD relationnels offrent un haut niveau de support pour les entreprises. Bien que certains fournisseurs NoSQL proposent des supports assez fiables, ils ne peuvent, bien évidemment, pas rivalisé avec un support mondial comme peuvent le faire Oracle, IBM ou Microsoft.

c- **Expertise**

Il existe des millions de développeurs dans le monde, opérant, dans différents businesses, qui se sont familiarisés avec les concepts ainsi que de la programmation dans un environnement de bases de données relationnelles. Dans le monde NoSQL, presque tous les développeurs sont en apprentissage avec la technologie. Avec le temps, cette situation risque de changer, mais pour le moment, il est plus simple de trouver une personne ayant une grande expérience des bases de données relationnelles qu'un expert NoSQL .[42]

Conclusion

Nous avons vu dans ce chapitre que NoSQL est une technologie principalement développée dans un contexte où la volumétrie des données rendait indispensable l'utilisation des systèmes distribués pour assurer leur stockage et leur traitement. C'est dans ce contexte qu'on a essayé de justifier le changement radical qu'apporte cette technologie aux architectures traditionnelles des bases de données que nous avons l'habitude de manipuler mais Le NoSQL ne remplace pas le SQL, c'est juste une alternative,on résumera ci dessous les differents aspets entre ces deux technologies .

NoSQL versus SQL

Le NoSQL se distingue du relationnel (SQL) sur plusieurs aspects qui sont :

- Les bases de données SQL sont principalement appelées bases de données relationnelles (SGBDR) ; alors que la base de données NoSQL est principalement appelée base de données distribuée ou non relationnelle.

- 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, des bases de données graphiques, etc. Cela signifie que les bases de données SQL représentent des données sous la forme de tables composées de n nombre de lignes de données, tandis que les bases de données NoSQL sont la collection de paires clé-valeur, de documents, de bases de données graphiques, etc. qui ne possèdent pas de définitions de schéma standard.

- Les bases de données SQL ont un schéma prédéfini alors que les bases de données NoSQL ont un schéma dynamique pour les données non structurées.

- Les bases de données SQL sont évolutives verticalement, tandis que les bases de données NoSQL sont évolutives horizontalement. Les bases de données SQL sont mises à l'échelle en augmentant la puissance du matériel. Les bases de données

NoSQL sont mises à l'échelle en augmentant le nombre de serveurs de bases de données dans le pool de ressources afin de réduire la charge.

• Les bases de données SQL utilisent SQL (structured query language) pour définir et manipuler les données, ce qui est très puissant. Dans la base de données NoSQL, les requêtes sont axées sur la collecte de documents. Parfois, il est également appelé UnQL (Unstructured Query Language). La syntaxe d'utilisation de UnQL varie d'une base à l'autre.

• **Exemples de bases de données SQL :** MySql, Oracle, Sqlite, Postgres et MS-SQL. Exemples de bases de données NoSQL : MongoDB, BigTable, Redis, RavenDb, Cassandra, Hbase, Neo4j et CouchDb

• **Pour les requêtes complexes :** les bases de données SQL conviennent parfaitement à l'environnement exigeant de nombreuses requêtes, tandis que les bases NoSQL ne conviennent pas aux requêtes complexes. À un niveau élevé, NoSQL n'a pas d'interfaces standard pour effectuer des requêtes complexes, et les requêtes elles-mêmes dans NoSQL ne sont pas aussi puissantes que le langage de requête SQL.

• **Pour le type de données à stocker :** Les bases de données SQL ne conviennent pas au stockage de données hiérarchique. Cependant, la base de données NoSQL convient mieux au stockage de données hiérarchique car elle suit la méthode de la paire clé-valeur pour stocker des données similaires aux données JSON. Les bases de données NoSQL sont hautement préférées pour les ensembles de données volumineux (c'est-à-dire pour les données volumineuses). Hbase est un exemple à cet effet.

• **Évolutivité :** dans la plupart des situations, les bases de données SQL sont évolutives 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 sont évolutives horizontalement. Vous pouvez simplement ajouter quelques serveurs supplémentaires facilement dans votre infrastructure de base de données NoSQL pour gérer le trafic important.

• **Pour les applications fortement transactionnelles** : les bases de données SQL conviennent mieux aux applications de type transactionnel à usage intensif, car elles sont plus stables et promettent l'atomicité ainsi que l'intégrité des données. Bien que vous puissiez utiliser NoSQL à des fins de transaction, il n'est toujours pas comparable et peut être utilisé suffisamment pour des applications transactionnelles complexes.

• **Pour les propriétés :** les bases de données SQL mettent l'accent sur les propriétés ACID (Atomicité, Cohérence, Isolation et Durabilité), tandis que la base de données NoSQL suit le théorème Brewers CAP (Cohérence, Disponibilité et Tolérance de partition).

Ci dessous un tableau récapitulatif qui présente les différence clé entre SQL et NoSQL :

Fonctionnalités	**SQL**	**NoSQL**
Modèle	*Relationnel* *Stockage des données est basée sur des table*	*Non-Relationnel* *Stockage des donnéesde type : paires Clé/valeur,documents, colonnes,graphe,*
Données	*stucturée ou semi-structuré*	*stucturée ou semi-structuré ou non-structurée*
Schema	*schéma prédéfini au départ*	*schéma dynamique pour les données non structurées.*
Transaction	*ACID*	*BASE,CAP*
Cohérence	*Il devrait être configuré pour une forte cohérence*	*Cela dépend du SGBD, car certains offrent une forte cohérence, comme MongoDB, alors que d'autres n'offrent qu'une cohérence éventuelle, comme Cassandra.*
Haute disponibilité	***clustering*** **:***on a une machine master,on copie la base sur des serveurs esclaves (avec sql serveur 2012)pour avoir une base de données dedié a la lecture*	***Base de données distribué*** *:* stokage des données sur differents serveurs.
Temps de réponse sur de gros volumes de données	*Perfermence limitée pour le croisement de données sur de gros volumes*	*Performant*
Scalabilité	***Scalabilité Horizontale*** *:ajout d'autant de serveur dont on a besoin pour augmenter les performence.*	***Scalabilité Verticale*** *:on augmente les performece hardware du serveur utilisé*
Schéma d'interrogation	*SELECT GROUP BY*	*MAP/REDUCE*
Langage d'interrogation	*SQL*	*Tout langage permettant de requêter une API REST (http)*
Exemple	*MySQL, Oracle, MS-SQL, SQLite.*	*MongoDB, Apache CouchDB, Redis, HBase.*

Table 3.1 – Tableau comparatif entre le SQL et le NoSQL

Chapitre 4

Hadoop et son outil d'entreposage de données Hive

Introduction

Le Big data regroupe plusieurs nouvelles technologies et d'outils pour répondre à une triple problématique :

— un **Volume** de données important à traiter,
— une grande **Variété** d'informations (structurées ou non structurées),
— et un certain niveau de **Vélocité** à atteindre.

Pour répondre à ces besoins, il s'avère que l'écosystème Hadoop serait la solution open source par excellence. Apache Hadoop (High-availability distributed object-oriented platform) est un système distribué qui répond à ces problématiques. D'une part, il propose un système de fichier distribué HDFS (Hadoop Distributed File System) pour assurer le stockage et l'intégrité des données en dupliquant plusieurs copies d'un même bloc à travers des dizaines, des centaines, voire des milliers de machines différentes, ce qui amène un cluster Hadoop à être configuré sans requérir un système RAID. D'autre part, Hadoop fournit un système d'analyse de données appelé **MapReduce** pour réaliser des traitements sur des gros volumes de données grâce à sa répartition efficace du travail sur différents nœuds de calcul.
Dans ce chapitre on présentera d'abord Hadoop et ensuite nous exposerons ces differents outils.

4.1 Vue globale sur Hadoop

Hadoop est un système matériel et logiciel distribué capable de répondre aux besoins du Big Data. Hadoop est capable de stocker et de traiter de manière séquentielle, et à des coûts ”raisonnables”, des volumes de données de plusieurs péta-octets.
Dans cette section nous allons présenté le framwork hadoop ainsi que ces différents outils.

4.1.1 Historique d'Hadoop

Hadoop trouve son origine dans le projet du moteur de recherche libre Apache Lucene lancé en 2002. Les développeurs du projet Lucene rencontrent pendant le développement du projet des problèmes de volumétrie de données, que Doug Cutting résoud en adaptant une version open source des outils propriétaires développées par Google, qui deviendra le projet Hadoop. Les dates marquantes de l'histoire du projet Hadoop sont :[52]

— En **2003**, Google publie un article qui décrit son système de gestion de fichiers distribués Google File System (GFS), ce dernier est utilisé dans le stockage des fichiers généré dans le cadre d'exploitation web et des processus d'indexation. En 2004, Apache implémente une version open source du système de fichier GFS, qui est nommé Nutch distributed File system (NDFS).

— En **2004,** Google publie un article qui présente son système d'analyse des données MapReduce.

— En **2005**, Apache intègre MapReduce avec NDFS dans le projet Nutch.

— En **2006**, Doug Cutting s'inspire du doudou de son fils de cinq ans, Hadoop un éléphant jaune, pour le logo ainsi que pour le nom de ce nouveau framework Java, qui est devenu un projet indépendant d'Apache.

— En **2008**, Doug Cutting intègre Yahoo et exploite Hadoop pour améliorer l'indexation de leur moteur de recherche

4.1.2 Présentation d'Hadoop

Hadoop est un Framework Java open source d'Apache pour réaliser des traitements sur des volumes de données massifs, de l'ordre de plusieurs pétaoctets (soit plusieurs milliers de To).

Hadoop a été conçu par Doug Cutting en 2004, également à l'origine du moteur Open Source Nutch. Doug Cutting cherchait une solution pour accroître la taille de l'index de son moteur. Il eut l'idée de créer un Framework de gestion de fichiers distribués. Yahoo ! en est devenu ensuite le principal contributeur, le portail utilisait notamment l'infrastructure pour supporter son moteur de recherche historique. Comptant plus de 10 000 clusters Linux en 2008, il s'agissait d'une des premières architectures Hadoop digne de ce nom. Crée spécialement pour les gros volumes. Facebook pour l'analyse des logs, Google pour l'analyse des requêtes, etc...
Il est caractérisé par :

- **Robustesse** : si un nœud de calcul tombe en panne , ses tâches sont automatiquement réparties sur d'autres nœuds. Les blocs de données sont également répliqués ;
- **Coût :** il optimise les coûts via une meilleure utilisation des ressources présentées ;
- **Souplesse :** car il répond à la caractéristique de variété des données en étant capable de traiter différents types de données ;
- **Virtualisation :** ne plus se reposer directement sur l'infrastructure physique (baie de stockage coûteuse), mais choisir la virtualisation de ses clusters Hadoop.

Trois principales distributions Hadoop sont aujourd'hui disponibles : **Cloudera, Hortonworks, MapR.**

Nous allons présenter deux concepts fondamentaux d'Hadoop : Sa propre version de l'algorithme **MapReduce** à savoir Hadoop MapReduce et son système de fichiers distribué **HDFS**.

4.1.3 Le système de fichier distribué d'Hadoop HDFS

Hadoop utilise un système de fichiers virtuel qui lui est propre : le HDFS (Hadoop Distributed File System). HDFS est un système de fichier distribué, extensible et portable inspiré par le

Google File System (GFS).

Il a été conçu pour stocker de très gros volumes de données sur un grand nombre de machine équipées de disques durs banalisés, il permet de l'abstraction de l'architecture physique de stockage, afin de manipuler un système de fichier distribué comme s'il s'agissait d'un disque dur unique. [53]

Toutefois, HDFS se démarque d'un système de fichiers classique pour les principales raisons suivantes : [54]

— HDFS n'est pas dépendant du noyau du système d'exploitation. Il assure une portabilité et peut être déployé sur différents systèmes d'exploitation. Un de ses inconvénients est de devoir solliciter une application externe pour monter une unité de disque HDFS ;

— HDFS est un système distribué sur un système classique, la taille du disque est généralement considérée comme la limite globale d'utilisation. Dans HDFS, chaque nœud d'un cluster correspond à un sous-ensemble du volume global des données du cluster. Pour augmenter ce volume global, il suffira d'ajouter de nouveaux nœuds. On retrouvera également dans HDFS, un service central appelé NameNode qui aura la tâche de gérer les métadonnées ;

— HDFS utilise des tailles de blocs largement supérieures à ceux des systèmes classiques. Par défaut, la taille est fixée à 64 Mo. Il est toutefois possible de monter à 128 Mo, 256 Mo, 512 Mo voire 1 Go. Alors que sur des systèmes classiques, la taille est généralement de 4 Ko, l'intérêt de fournir des tailles plus grandes permettant de réduire le temps d'accès à un bloc. Notez que si la taille du fichier est inférieure à la taille d'un bloc, le fichier n'occupera pas la taille totale de ce bloc ;

— HDFS fournit un système de réplication des blocs dont le nombre de réplications est configurable. Pendant la phase d'écriture, chaque bloc correspondant au fichier est répliqué sur plusieurs nœuds. Pour la phase de lecture, si un bloc est indisponible sur un nœud, des copies de ce bloc seront disponibles sur d'autres nœuds.

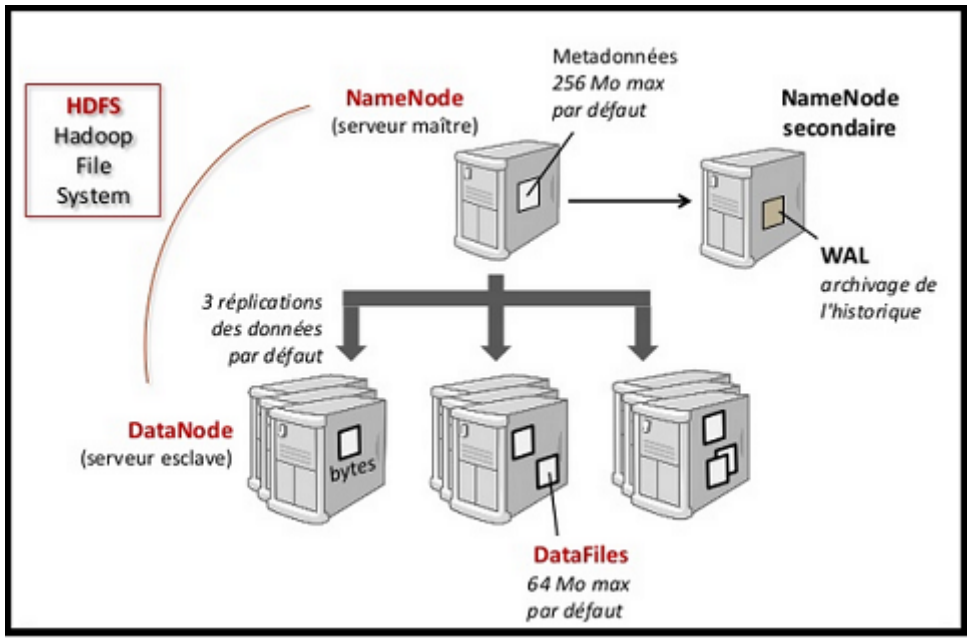


FIGURE 4.1 – L'architecture d'HDFS [55]

4.1.3.1 Les composantes d'HDFS

HDFS définit deux types de nœuds :

1. **Le nœud principal ou NameNode**

Il se caractérise par les tâches suivantes :

— Responsable de la distribution et de la réplication des blocs ;

— Serveur d'informations du Hdfs pour le client Hdfs ;

— Stocke et gère les métadonnées ;

— Comporte la liste des blocs pour chaque fichier (dans le cas de lecture) ;

— Contient la liste des DataNodes pour chaque bloc (dans le cas de l'écriture) ;

— Tenir les attributs des fichiers (ex : nom, date de création, facteur de réplication) ;

— Logs toute métadonnée et toute transaction sur un support persistant ;

— Lectures/écritures ; Créations/suppressions ;

— Démarre à partir d'une image d'HDFS (fsimage).

2. **Le nœud de données ou DataNode**

Il se caractérise par les tâches suivantes :

• Stocke des blocs de données dans le système de fichier local ;

• Maintenir des métadonnées sur les blocs possédés (ex : CRC) ;

• Serveur de bloc de données et de métadonnées pour le client hdfs ;

• Heartbeat avec le Namenode : Heartbeat est système permettant sous Linux la mise en clusters de plusieurs serveurs pour effectuer entre eux un processus de tolérance de panne. Le processus Heartbeat se chargera de passer un message-aller vers le NameNode indiquant : son identité, sa capacité totale, son espace utilisé, son espace restant.

3. Secondary NameNode

Le NameNode dans l'architecture Hadoop est un point unique de défaillance. Si ce service est arrêté, il n'y a pas moyen de pouvoir extraire les blocs d'un fichier donné. Pour résoudre ce problème, un NameNode secondaire appelé Secondary NameNode a été mis en place dans l'architecture Hadoop. Son rôle consiste à :

— Télécharger régulièrement les logs sur le NameNode ;

— Crée une nouvelle image en fusionnant les logs avec l'image HDFS ;

— Renvoie la nouvelle image au NameNode. [56]

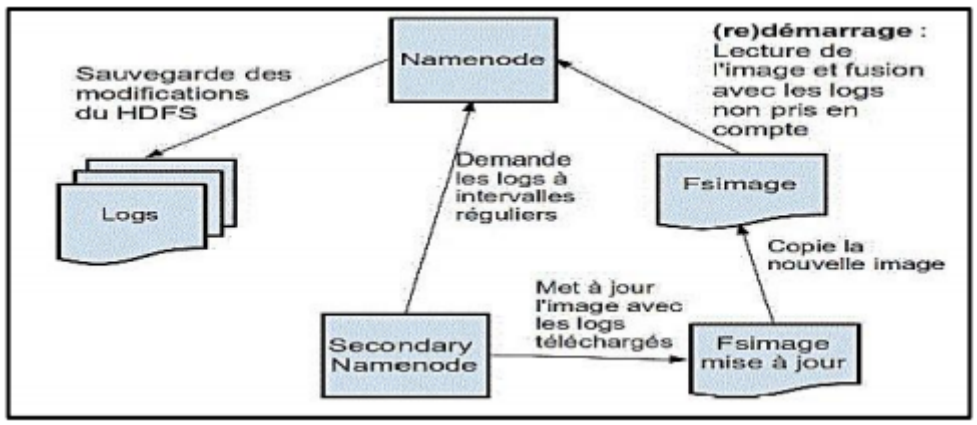


FIGURE 4.2 – Fonctionnement du Secondary NameNode

4.1.3.2 HDFS et tolérance aux fautes

Pour éviter un crash du DataNode la solution serait :

- Plus de Heartbeat (détection par le NameNode) ;
- Plus de réplication distribuée (robustesse des données).
- Dans le cas d'un crash du NameNode **(voir Figure 4.2) ;**
- Sauvegarde des logs de transaction sur un support stable ;
- Redémarrage sur la dernière image du hdfs.

4.1.3.3 Lecture d'un fichier HDFS

Pour lire un fichier au sein de HDFS, il faut suivre les étapes suivantes :

Etape 1 : Le client indique au NameNode qu'il souhaite lire le fichier data.txt

Etape 2 : Le NameNode lui indiquera la taille de fichier (nombre de blocs) ainsi que les différents Data Node hébergeant les n blocs.

Etape 3 : Le client récupère chacun des blocs à un des DataNodes.

Etape 4 : En cas d'erreur/non réponse d'un des DataNode, il passe au suivant dans la liste fournie par le NameNode. [56]

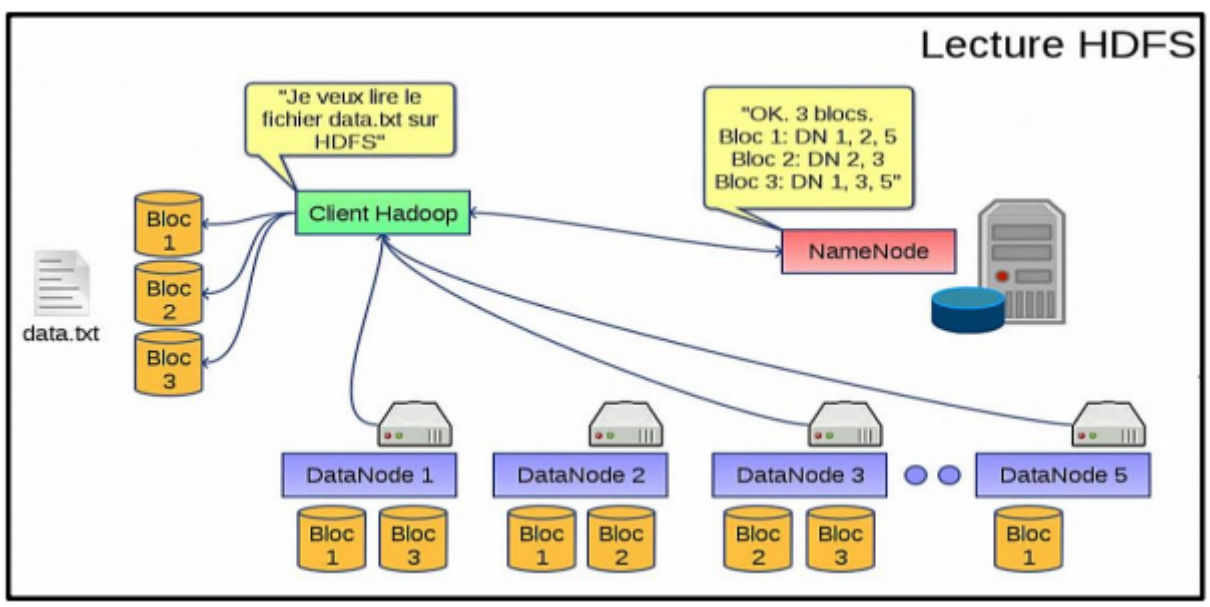


FIGURE 4.3 – Lecture d'un fichier HDFS.

4.1.3.4 Ecriture dans un fichier ou volume HDFS

Pour écrire un fichier au sein d'HDFS :

Etape 1 : On va utiliser la commande principale de gestion de Hadoop : Hadoop, avec l'option fs. Admettons qu'on souhaite stocker le fichier data.txt sur HDFS.

Etape 2 : Le programme va diviser le fichier en blocs de 64KB (ou autre, selon la configuration) – supposons qu'on ait ici 3 blocs.

Etape 3 : Le NameNode lui indique les DataNodes à contacter.

Etape 4 : Le client contacte directement le DataNode concerné et lui demande de stocker le bloc.

Etape 5 : les DataNodes s'occuperont – en informant le NameNode – de répliquer les données entre eux pour éviter toute perte de données.

Etape 6 : Le cycle se répète pour le bloc suivant.[56]

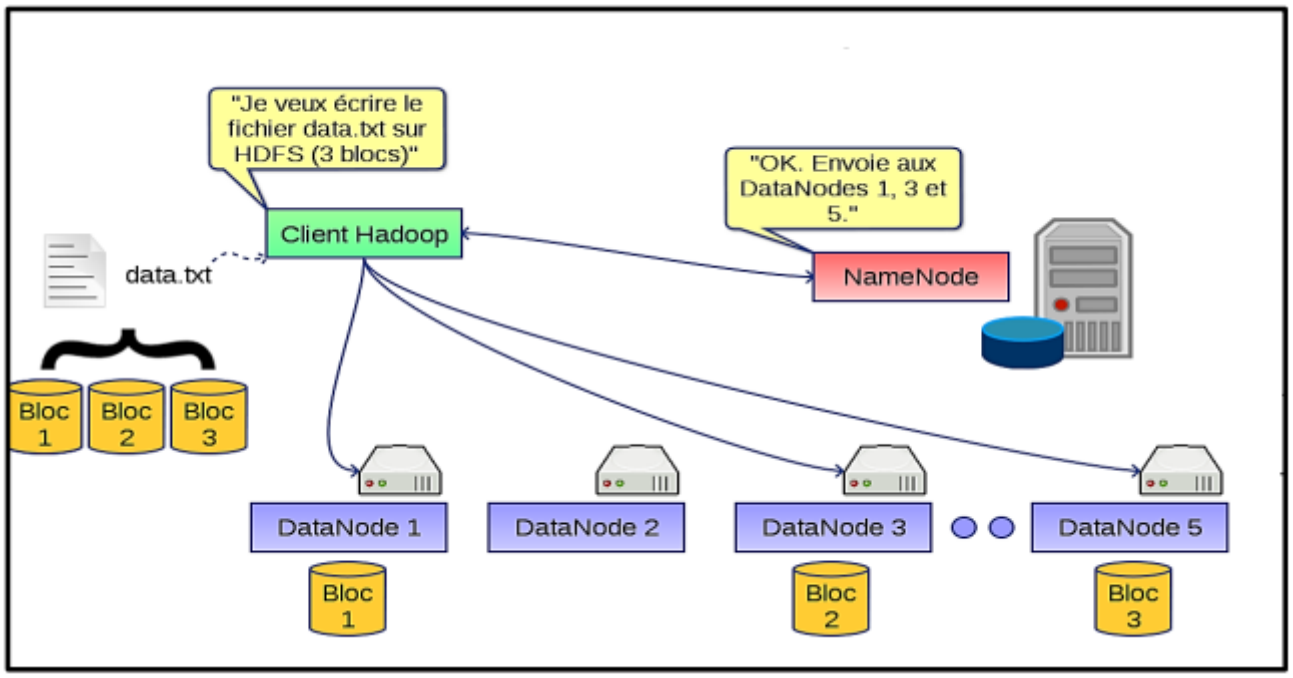


FIGURE 4.4 – Processus d'écriture dans un volume ou fichier HDFS.

4.1.4 MapReduce

MapReduce est un paradigme (modèle) de programmation parallèle proposé par Google. Il est principalement utilisé pour le traitement distribué sur de gros volumes de données aux seins d'un cluster de nœuds. Il est conçu pour la scalabilité et la tolérance aux pannes.
Le modèle de programmation fournit un cadre à un développeur afin d'écrire une fonction Map et une fonction Reduce. Tout l'intérêt de ce modèle de programmation est de simplifier la vie du développeur. Ainsi, ce développeur n'a pas à se soucier du travail de parallélisation et de distribution du travail. MapReduce permet au développeur de ne s'intéresser qu'à la partie algorithmique. [54]
Un programme MapReduce peut se résumer à deux fonctions **Map () et Reduce ()**

— La première, **MAP**, va transformer les données d'entrée en une série de couples clef /valeur. Elle va regrouper les données en les associant à des clefs, choisies de telle sorte que les couples clef/valeur aient un sens par rapport au problème à résoudre. Par ailleurs, cette opération doit être parallélisable : on doit pouvoir découper les données d'entrée en plusieurs fragments, et faire exécuter l'opération MAP à chaque machine du cluster sur un fragment distinct. La fonction Map s'écrit de la manière suivante :

Map (clé1, valeur1) → List (clé2, valeur2).

— La seconde, REDUCE, va appliquer un traitement à toutes les valeurs de chacune des clefs distinctes produite par l'opération MAP. Au terme de l'opération REDUCE, on aura un résultat pour chacune des clefs distinctes. Ici, on attribuera à chacune des machines du cluster une des clefs uniques produites par MAP, en lui donnant la liste des valeurs associées à la clef. Chacune des machines effectuera alors l'opération REDUCE pour cette clef. La fonction Reduce s'écrit de la manière suivante :

Reduce (clé2, List (valeur2)) → List (valeur2).
Exemple : (sentiment client/twitter) [58]

— *une entreprise dispose d'un compte twitter pour son service aprés vente, recevant plusieurs dizaines de milliers de tweets par jour. Elle cherche à déterminer le taux de satisfaction de ses clients à partir du compte twitter.*

— *Chaque heure, les tweets reçus sont exportés au sein d'un fichier texte. Données d'entrée :*

```
"@acme Votre service client est nul"
"@acme 30min d'attente... très insatisfait"
"Très satisfait par un produit super !! @acme"
"merci d'avoir RT @acme"
"@acme produit déjà cassé, super insatisfait !"
```

Remarque :

Dans un environnement de calcul traditionnel, on utilise généralement des Hash-tables, sous forme de : (Clef Valeur)

Si on utilise les hashtables sur 1To :

- Le traitement séquentiel de toutes les données peut s'avérer très long.
- Plus on a de commentaires plus l'ajout des valeurs à la table est long .
- Mais cela peut marcher, et le résultat sera correct .

Map-Reduce : Moyen plus efficace et rapide de traiter ces données :

— **Clef** : un descripteur de sentiment client (satisfait , insatisfait ou attente , inconcluant).

— **map** : génère un couple (clef ;valeur) par sentiment client détecté (mot correspondant à une liste prédéfinie).

— Si deux sentiments contradictoires détectés : renvoyer inconcluant.

— On renvoie un couple (clef ;valeur) pour chaque fragment des données d'entrée : chaque tweet

— Pseudo code :

```
WORDS_BAD = ["nul", "insatisfait", "bof", "incompétents", …]
WORDS_GOOD = ["satisfait", "super", "excellent", …]
BAD=0; GOOD=0
POUR MOT dans [TWEET], FAIRE:
  SI MOT PRESENT_DANS WORDS_BAD:
    BAD=1
  SINON SI MOT PRESENT_DANS WORDS_GOOD:
    GOOD=1
SI BAD==1 ET GOOD==0:
  RENVOYER("insatisfait",1)
SINON SI BAD==0 ET GOOD=1:
  RENVOYER("satisfait",1)
SINON:
  RENVOYER("inconcluant",1)
```

— Après exécution :

```
"@acme Votre service client est nul"
"@acme 30min d'attente... très insatisfait"
"Très satisfait par un produit super !! @acme"
"merci d'avoir RT @acme"
"@acme produit déjà cassé, super insatisfait !"
```

↓

```
("insatisfait";1)
("insatisfait";1)
("satisfait";1)
("inconcluant";1)
("inconcluant";1)
```

— **reduce :** additionne les valeurs associées à la clef unique ; renvoie le total pour valeur .

— Pseudo code :

```
Fonction reduce (cle, valeurs):

Result=0

Pour C dans [valeurs] faire:

Result=Result + C

Renvoyer (cle, Result)
```

— Après exécution :

```
("insatisfait";2)
("satisfait";1)
("inconcluant";2)
```

Conclusion : lors de la dernière heure, **33%** de tweets exprimant de la **satisfaction**.

4.1.4.1 MapReduce dans Hadoop

Il existe Deux versions notables du MapReduce :

— Version 0.x et 1.x : architecture purement maître-esclave ;

— Version 2.x YARN : architecture maître-esclave à deux niveaux (Stable depuis oct. 2013). [56]

4.1.4.2 Architecture du Framework MapReduce (purement maître-esclave)

Le MapReduce possède une architecture maître-esclave

— Le maître MapReduce : **le JobTracker** ;

— Les esclaves MapReduce : **les TaskTracker.**

1. Le JobTracker

- 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 ;
- Maintient des informations sur l'état d'avancement des jobs. [56]

2. **Un TaskTracker**

- Exécute les tâches données par le Jobtracker ;
- A une capacité en termes de nombres de tâches qu'il peut exécuter ;
- Heartbeat avec le JobTracker. [56]

4.1.4.3 Fonctionnement du Framework MapReduce

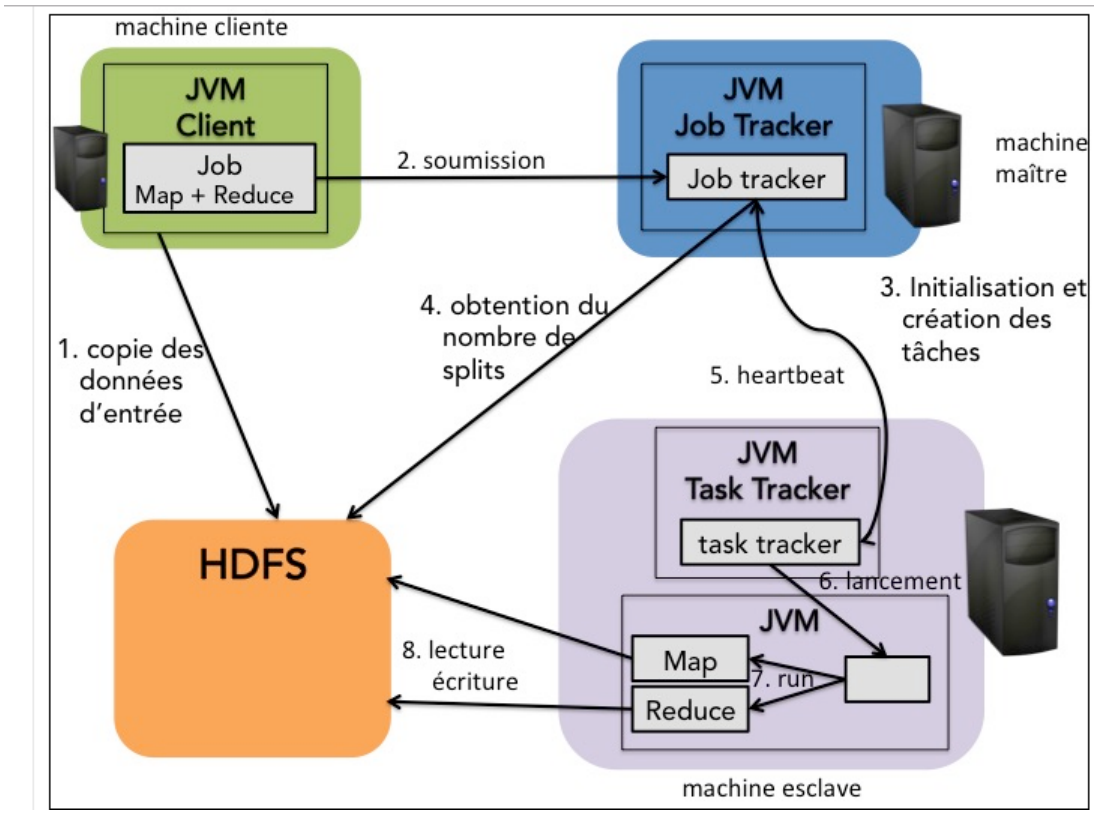


FIGURE 4.5 – soumission et exécution d'un job dans Hadoop MapReduce[59]

1. Un client hadoop copie ses données sur HDFS.
2. Le client soumet le travail à effectuer au job tracker sous la forme d'une archive.jar et des noms des fichiers d'entrée et de sortie.
3. Le job tracker demande au name node où se trouvent les blocs correspondants aux données d'entrée.
4. Il détermine alors quels sont les nœuds Task Tracker les plus appropriés pour exécuter les traitements (colocalisation ou proximité des nœuds). Il envoie alors au task tracker

selectionné et pour chaque bloc de données, le travail à effectuer (Map, Reduce ou Shuffle, fichier .jar).

5. Les task trackers envoient régulièrement un message (hearbeat) au job tracker pour l'informer de l'avancement de la tâche et de leur nombre de slots disponibles.
6. Quand toutes les opérations envoyées aux task trackers sont confirmées comme étant effectuées, la tâche est considérée comme effectuée.[59]

4.1.4.4 Hadoop MapReduce 2.x : YARN

YARN est un nouveau composant dans l'architecture maître-esclave à deux niveaux par rapport aux architectures précédentes des versions 0.x et 1.x. si on s'interesse un peu plus à l'architecture d'Hadoop, on remarque que le job tracker a une double responsabilité :

— Il doit gérer les ressources du cluster.
— Il doit ordonnancer les jobs. Que se passe-t-il si le Job tracker est défaillant ?

YARN apporte une séparation claire entre les problématiques suivantes :

— Gestion de l'état du cluster et des ressources.
— Gestion de l'exécution des jobs.

En particulier, dans YARN, les fonctionnalités du **job tracker** sont réparties entre :

— **Le resource manager** qui est le chef d'orchestre des ressources du cluster. Il ordonnance les requêtes clients et pilote le cluster par l'intermédiaire de **node managers** qui s'exécutent sur chaque nœud de calcul. Il a donc pour rôle de contrôler toutes les ressources du cluster et l'état des machines qui le constituent. Il gère donc le cluster en maximisant l'utilisation de ressources.
— **L'application master (AM)** qui est un processus s'exécutant sur toutes les machines esclaves et qui gère, en discussion avec le **resource manager**, les ressources nécéssaires au travail soumis.

De même, les fonctionnalités du **task tracker s**ont aussi réparties sur une même machine entre :

— **Des containers** qui sont des abstractions de ressources sur un nœud dédiées soit à l'exécution de tâches comme Map et Reduce, soit à l'exécution d'un application master.

— **Un node manager** qui héberge des containers et gére donc les ressources du nœud. Il est en communication via un heartbeat avec le ressource manager.

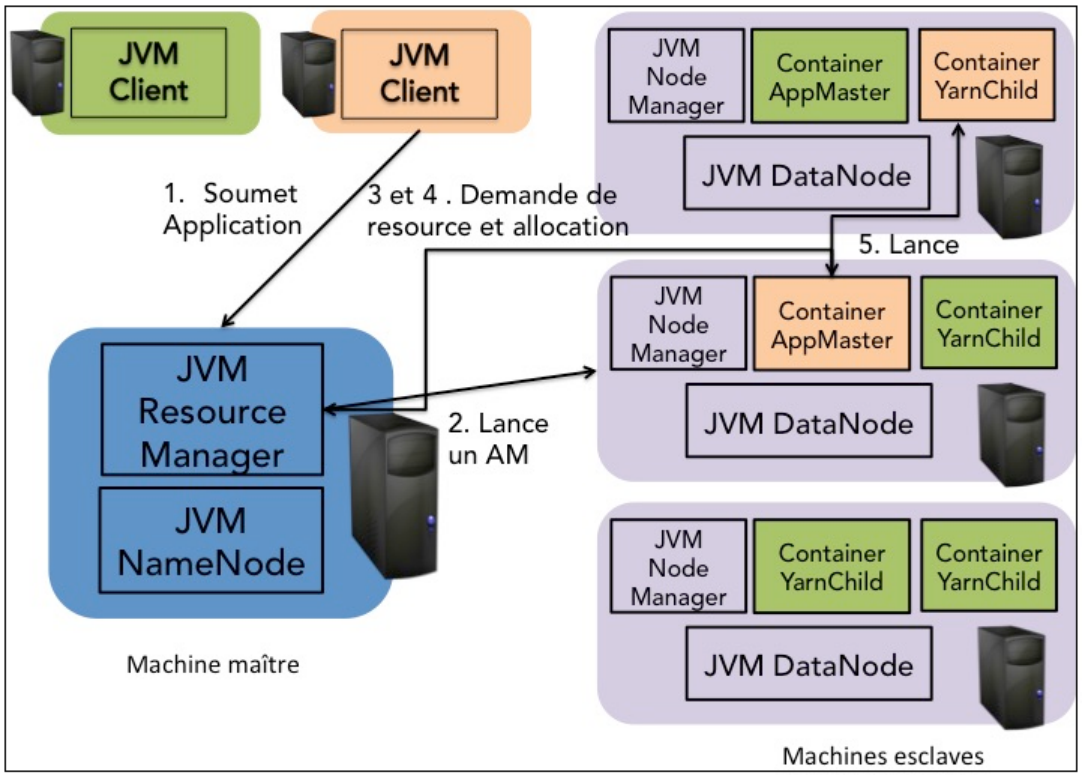


FIGURE 4.6 – Schéma simplifié de l'exécution d'un travail dans Hadoop 2.X avec YARN [59]

Le schéma de soumission et d'exécution d'un job dans cette nouvelle architecture est donc le suivant :

1. Un client hadoop copie ses données sur HDFS.
2. Le client soumet le travail à effectuer au **resource manager** sous la forme d'une archive.jaret des noms des fichiers d'entrée et de sortie.
3. **Le resource manager** alloue alors un container pour l'**application master** sur un **node manager**.
4. L'**application master** demande au **resource manager** un ou plusieurs **containers**

avec des préférences de localisation dépendant de la localité des données d'entrée du travail.

5. Le **resource manager** alloue alors un ou plusieurs **containers** (child) à l'**application master.**
6. L'**application master** choisit parmi la liste des tâches (par exemple Map et Reduce) et demarre une instance de la tâche choisie dans un des **containers** qui lui a été alloué. Il collabore alors avec le **node manager** pour utiliser les ressources acquises. Il communique aussi souvent avec le **resource manager** (message heartbeat) pour la tolérance aux pannes.

4.1.5 Écosystème d'Hadoop

L'écosystème de Hadoop comprend de nombreux outils en outre HDFS et MapReduce, tel représenté dans la FIG 5.7. Les outils d'Hadoop peuvent être classés en trois grandes catégories :

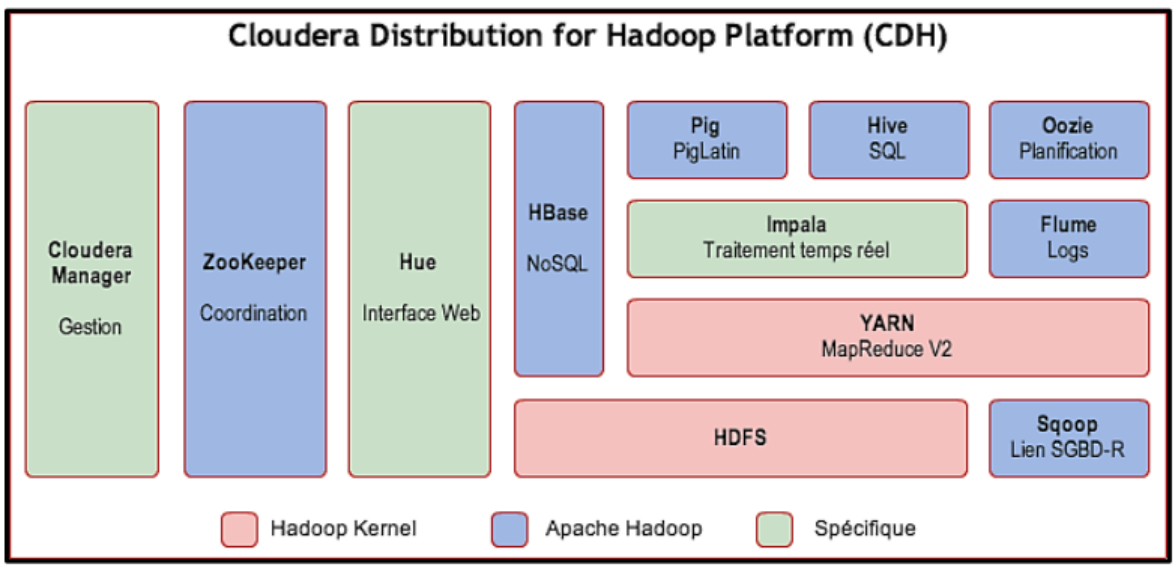


FIGURE 4.7 – Ecosystème d'Hadoop [60]

i)- Outils compsant le noyau Hadoop

1- HDFS (Hadoop Distributed File System) :

- **Définition :**

HDFS est un système de fichiers Java utilisé pour stocker des données structurées ou non sur un ensemble de serveurs distribués. HDFS 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. la consistance des données est basée sur la redondance. Une donnée est stockée sur au moins n volumes différents.

- **Principe de Fonctionnement :**

HDFS repose sur une architecture HDFS maître/esclave. Le serveur maître ne s'occupe pas de la gestion de ses propres machines. En revanche, indirectement, il procéde à la gestion des ordinateurs de ses serveurs esclaves en leur attribuant des tâches. Il fournit également des informations sur l'état des réseaux comme la sécurité par exemple. En d'autres termes, le serveur maître octroie des ordres à l'esclave qui les exécute par la suite.

Concernant le serveur esclave, celui-ci agira comme une instance autonome qui gére les ordinateurs du réseau.Chaque cluster à un Namenode et cela permettra aux clients d'accéder aux données. Ensuite, chaque noeud est constitué d'un ou d'une multitude de Datanode.

Le Namenode de chaque cluster centralise toute la gestion des dossiers et des fichiers.L'architecture de HDFS repose sur un système de hiérarchisation de fichier qui permet aux utilisateurs de créer un dossier, d'ajouter et de supprimer des fichiers, de déplacer les fichiers et de les renommer.

Enfin, étant donné que HDFS n'est pas à l'abri d'une panne éventuelle, les données

sont stockées de manière redondante.[61]

- **Structure de données sur HDFS**

Les données sur HDFS (structurées, semi-structurées ou non structurées) sont enregistrées avec le concept : écrire une fois, lire plusieurs fois. Cela veut dire que les mises à jour et les suppressions de données ne font pas partie des bonnes pratiques du HDFS .

Les données dans un HDFS sont enregistrées en blocs de grande taille de 64 MB (par défaut) ou plus pour des raisons de performance (réduire le temps d'accès).

Dans un Cluster on dispose d'un nœud maître NameNode qui gère les nœuds de données DataNodes . La haute disponibilité est assurée par la couche de données, car les blocs sont répliqués (par défaut) en 3 copies ou plus.

La haute disponibilité du NameNode pourra être assurée par l'ajout d'un Secondary NameNode qui se met en synchronisation avec le nœud maitre . Une table sur un système HDFS n'est que l'ensemble de plusieurs sous fichiers qui contiennent des données et qui sont reliés entre eux. Contrairement aux SGBD, le HDFS ne gère pas les index, ce qui fait de lui un système sans schéma avec un gain d'espace et de mémoire et avec une faible maintenance .

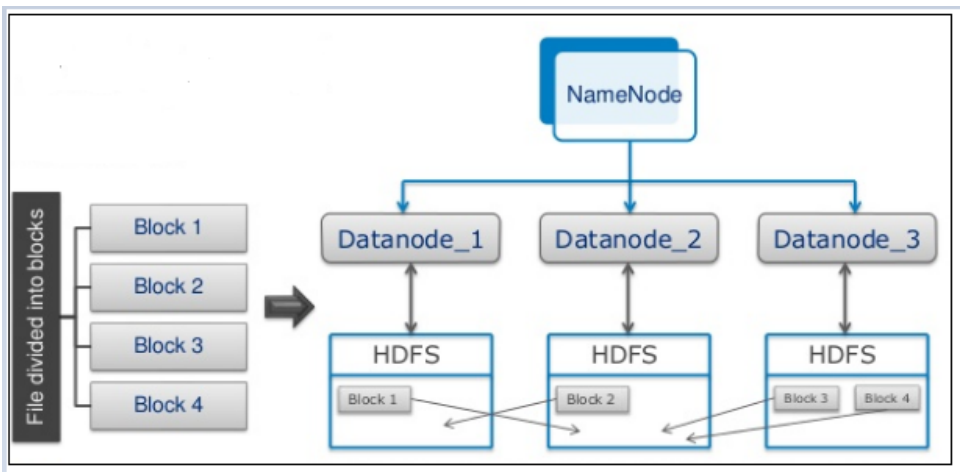


FIGURE 4.8 – Fonctionnement du Hadoop Distributed File System

- **Avantages**
  - Très efficace pour le traitement des données.
  - Système très fonctionnel pour traiter les grosses données (Big Data).
  - Chaque donnée est stockée à trois emplacements afin d'éviter la perte des données.
  - Déplacement des données afin d'éviter la congestion du réseau.[61]

- **Incovenients :**

  - Presence d'un Single Point of Failure. Bien qu'à l'heure actuelle une très grande partie des architectures Hadoop en service ait développé un Secondary Name Node, la structure originelle n'en contenait pas. Cela constituait un risque en cas de défaillance du Name Node responsable de l'archivage des données.
  - Absence de sécurité. Dans sa version initiale, Hadoop ne disposait pas de fonctionnalités de sécurité garantissant un déploiement sans risque.
  - Administration complexe d'Hadoop. Alors que la plupart des bases de données utilisent du SQL ou d'autres langages informatiques standardisés, Hadoop utilise son propre langage. Si une entreprise veut tirer profit de Hadoop, elle doit donc développer une expertise spécifique d'Hadoop en interne ou en faisant appel à un prestataire extérieur. Ce facteur est actuellement celui qui freine le plus son adoption. [62]

- **Alternatives :**

a- MapR : En mai 2011, MapR a annoncé une alternative au système HDFS. Ce système permet d'éviter le SPOF (Single Point Of Failure) représenté par le Name Node. Ce système n'est pas inconnu car il s'agit de HBase, dont elle propose une version propriétaire.

b- HBase (Apache) :

HBase est un sous-projet d'Hadoop, c'est un système de gestion de base de données non relationnel distribué, écrit en Java, disposant d'un stockage structuré pour les grandes tables. HBase est inspirée des publications de Google sur BigTable. Comme BigTable, c'est une base

de données orientée colonnes. HBase est souvent utilisé conjointement au système de fichiers HDFS, ce dernier facilitant la distribution des données de HBase sur plusieurs nœuds. Contrairement à HDFS, HBase permet de gérer les accès aléatoires read/write pour des applications de type temps réel.

c- Cassandra (Facebook) :

Cassandra est une base de données orientée colonnes développée sous l'impulsion de Facebook. Cassandra supporte l'exécution de jobs MapReduce qui peuvent y puiser les données en entrée et y stocker les résultats en retour (ou bien dans un système de fichiers).Cassandra, comparativement à Hbase, est meilleure pour les écritures alors que ce dernier est plus performant pour les lectures.[63]

2- MapReduce :

- **Définition :**

Initialement crée par Google pour son outil de recherche web. C'est un Framework qui permet la décomposition d'une requête importante en un ensemble de requêtes plus petites qui vont produire chacune un sous ensemble du résultat final : c'est la fonction Map.

- **Caracteristiques**

— Le modèle de programmation du MapReduce est simple mais très expressif. Bien qu'il ne possède que deux fonctions, map () et reduce (), elles peuvent être utilisées pour de nombreux types de traitement des données, les fouilles de données, les graphes... Il est indépendant du système de stockage et peut manipuler de nombreux types de variable.

— Le système découpe automatiquement les données en entrée en bloc de données de même taille. Puis, il planifie l'exécution des tâches sur les nœuds disponibles.

— Il fournit une tolérance aux fautes à grain fin grâce à laquelle il peut redémarrer les nœuds ayant rencontré une erreur ou affecter la tâche à un autre nœud.

— La parallélisation est invisible à l'utilisateur afin de lui permettre de se concentrer sur le traitement des données .

— Plusieurs implementation de ce framework dans différents langages (C++, Java, Python,

etc) et par nombreux organismes (Google, Yahoo, etc).[64]

• **Principe de Fonctionnement :**

— Accès aux données sur HDFS.

— Le système Hadoop se charge de splitter les données en blocs.

— Sur chaque bloc est appliqué la fonction map () que nous devons programmer.

— Le système réorganise les données d'une manière à regrouper les objets qui ont une clé unique.

— Pour chaque type de données possédant la même clé est appelé la fonction Reduce ().

— En sortie, on a une liste des éléments calculés. [59]

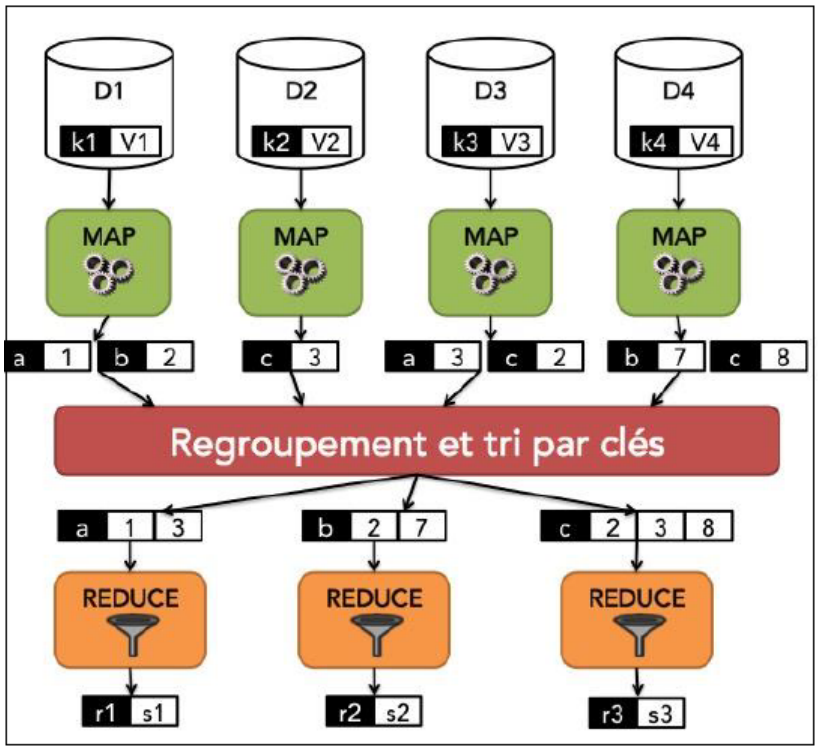


FIGURE 4.9 – Fonctionnement de MapReduce

• **Avantages :**

— Assurer le traitement parallèle en utilisant la distribution des données, le balancement de charges ainsi que la tolérance aux pannes.

— MapReduce est conçu pour fonctionner sur des machines à faible coût avec des disques durs simples, car il se base sur des opérations d'E/S sans que la mémoire soit une

priorité.

- **Incovenients :**

— la complexité de la programmation, même dans le cas d'une simple requête. Ainsi, en termes de performance, un PDBMS tel que Vertica par exemple, est plus rapide que MapReduce par un facteur de 3.2 à 7.4 pour un cluster de 100 nœuds [6] [12].

- **Alternative :**

YARN (HortonWorks) YARN (Yet-Another-Resource-Negotiator) est aussi appelé MapReduce 2.0, ce n'est pas une refonte mais une évolution du Framework. [63]

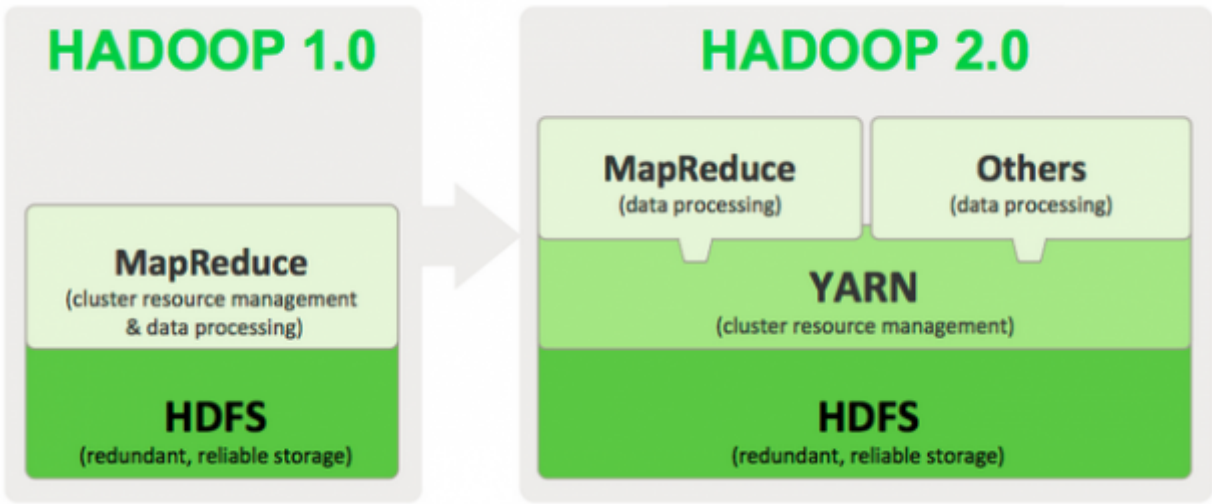


FIGURE 4.10 – D'Hadoop 1.X à Hadoop 2.X avec YARN. (Source : HortonWorks)

ii)- Les outils de Requêtage et de Scripting des données dans Hadoop

— **Hive (Facebook) :** Hive est à l'origine un projet Facebook qui permet de faire le lien entre le monde SQL et Hadoop. Il permet l'exécution de requêtes SQL sur un cluster Hadoop en vue d'analyser et d'agréger les données. Le langage SQL est nommé HiveQL. C'est un langage de visualisation uniquement, c'est pourquoi seules les instructions de type "Select" sont supportées pour la manipulation des données. Dans certains cas, les développeurs doivent faire le Mapping entre les structures de données et Hive.

— **Pig (Yahoo)** : Apportent un modèle de développement de plus haut niveau, et donc

beaucoup plus expressif et simple à appréhender, afin de démocratiser l'écriture de traitements MapReduce. Pig se rapproche plus d'un ETL où on part d'un ou plusieurs flux de données que l'on transforme étape par étape jusqu'à atteindre le résultat souhaité. Les différentes étapes de la transformation sont exprimées dans un langage procédural (Pig Latin).Pig est à l'origine un projet Yahoo qui permet le requêtage des données Hadoop à partir d'un langage de script. [63]

iii)- L'outil d'intégration SGBD-R (Relationnel) Sqoop (Cloudera)

Sqoop permet le transfert des données entre un cluster Hadoop et des bases de données relationnelles. C'est un produit développé par Cloudera, Il permet d'importer/exporter des données depuis/vers Hadoop et Hive. Pour la manipulation des données Sqoop utilise MapReduce et des drivers JDBC.

iiii)- Les outils de gestion et de supervision du cluster Hadoop

— **Apache ZooKeeper :** ZooKeeper est un service de coordination des services d'un cluster Hadoop, en particulier, le rôle de ZooKeeper est de fournir aux composants Hadoop les fonctionnalités de distribution, pour cela il centralise les éléments de configuration du cluster Hadoop, propose des services de clustérisassions et gère la synchronisation des différents éléments (événements).

ZooKeeper est un élément indispensable au bon fonctionnement de Hbase.[63]

— **Apache Ambari (HortonWorks) :** Ambari est un projet d'incubation Apache initié par HortonWorks et destiné à la supervision et à l'administration de clusters Hadoop. C'est un outil web qui propose un tableau de bord, cela permet de visualiser rapidement l'état d'un cluster. [63]

Ambari dispose d'un tableau de bord dont le rôle est de fournir une représentation :

— De l'état des services ;
— De la configuration du cluster et des services ;

— De l'exécution des jobs ;

— Des métriques de chaque machine et du cluster.

iiiiii)- Outil d'ordonnancement et de coordination : Apache Oozie (Yahoo)

Oozie est une solution de workflow (au sens scheduler d'exploitation) utilisée pour gérer et coordonner les tâches de traitement de données à destination de Hadoop. Oozie s'intègre parfaitement avec l'écosystème Hadoop puisqu'il supporte les types de jobs suivant :

— MapReduce (Java et Streaming) ;

— Pig ;

— Hive ;

— Sqoop.

4.1.6 Domaines d'application

Les domaines d'application d'Hadoop englobe tous les secteurs d'activités tel que :

a- Industrie de la Télécommunication

Les grandes entreprises de télécommunications ont un certain nombre de marchés verticaux tels que le marketing, produit, ventes, ressources humaines, technologies de l'information, recherche et développement etc, qui sont dans le besoin constant d'informations. Et à travers Hadoop ces entreprises de télécommunications peuvent :

— Analyser leurs enregistrements de données avec Hadoop : pour pouvoir améliorer continuellement la qualité des appels, la satisfaction du client et les marges d'entretien.

— Réduire le taux de désabonnement des clients : en implémentant Hadoop à la fin de l'année 2013. SFR est devenu le premier opérateur mobile français à s'engager dans l'expérience Hadoop pour réduire le nombre de désabonnement de ses clients.

— Recommander suivant le produit à acheter : avec Hadoop, l'entreprise de télécommunication identifie les besoins du client et lui recommande un produit adapté à ses attentes.

b- Industrie publicitaire

La chaine de supermarchés Target (USA) a mis au point un système d'analyse des achats de ses clients, basé sur Hadoop. Afin de pouvoir envoyer à chacun de ses clients de la publicité ciblée.

c- Domaine de la santé

La société "Treato" a mis au point un système basé sur Hadoop et Hbase qui est capable, en analysant les échanges sur des sites sociaux, de détecter une interaction médicamenteuse. Le système permet aussi d'identifier en temps quasi réel les préoccupations des patients, leur comportement, etc.

d- Dans le cadre des moteurs de recherche

Orange a amélioré les résultats de son moteur de recherche en fournissant aux utilisateurs des résultats plus adéquats à leurs requêtes et cela en mettant en place un système d'indexation des pages web et un système de type PageRank, basé sur la plate-forme d'Hadoop.

e- Industrie des villes intelligentes

Le réseau électrique intelligent utilise l'informatique comme moyen d'optimisation de la productivité et la consommation de l'énergie. En France, EDF prévoit d'installer des compteurs électriques communicants qui permettent de récolter des informations sur la consommation en énergie des clients, ces compteurs vont générer des volumes de données considérables. Dans ces conditions, EDF utilisra Hadoop comme solutions .

f- Domaine des réseaux sociaux

L'ensemble de l'infrastructure de traitement de données dans Facebook avant 2008 a été construit autour d'un entrepôt de données utilisant un SGBDR commercial. Le volume de données de Facebook ne cessait de s'accroitre de 15TB fixé en 2007 à un ensemble de données de 700TB en 2009. L'infrastructure employée était insuffisante. Ils avaient eu un besoin d'une plate-forme de donnée évolutif. Par conséquent ils se sont orientés vers Hadoop comme plate-forme de leur entrepôt de données Hive.[65]

4.1.7 Présentation de l'outil Hive

Hive est un projet open source dédié à la construction d'entrepôt de données sous la plateforme Hadoop. Hive utilise un langage de requête semblable à SQL appelé HiveQL, qui prend en charge certaines primitives de manipulation de données telles que la projection, jointure, le groupement, l'union et les sous-requêtes dans la clause FROM [66]. Le plan d'exécution d'une requête HiveQL se traduit par la création et l'exécution d'un ensemble de tâches map et de tâches reduce.

4.1.7.1 Les principales caractéristiques de Hive

Hive se distingue d'une base de données traditionnelle par les caractéristiques suivantes :

— Dans un SGBDR, le schéma d'une table est appliqué au moment de la charge, si les données en cours du chargement ne sont pas conformes au schéma, elles sont rejetées, cette conception est appelée schéma en écriture. Mais Hive ne vérifie pas les données lors du chargement mais lors de la récupération, cette conception est appelée schéma en lecture.L'opération de chargement des données dans Hive est juste une copie de fichier ou un déplacement.

— Il est basé sur la notation de Write Once Read Many. Il est donc possible d'écrire qu'une seule fois (Write Once) et de lire autant de fois souhaité (Read Many).

— Il n'autorise pas les opérations : insertions, suppressions et modification, car Hive est construit pour fonctionner sur des données HDFS utilisant MapReduce, où l'analyse et la mise à jour des données s'obtenaient en transformant les données en une nouvelle table.

— Il ne prend pas en charge les traitements de transaction en ligne OLTP (Online Transaction Processing) car Hadoop est un système orienté par lots.

4.1.7.2 Langage HiveQL (Hive Querry Language)

Le langage HiveQL supporte la définition de données (DDL) dans la création des tables avec des formats de sérialisation spécifiques, de partitionnement et de buckting. Et ne supporte

guère la mise à jour et la suppression de lignes des tables existantes.
Il soutient l'insertion multi-table, où les utilisateurs peuvent exécuter plusieurs requêtes sur les mêmes données d'entrée en utilisant une seule déclaration Hive, ce qui optimise l'exécution des requêtes. HiveQL est un langage extensible car il permet aux utilisateurs de créer leur propre fonction avec le langage Java, également de définir les fonctions d'agrégation et les fonctions de table génératrices .[67]

— **Types de données**

Les types de données de Hive sont de deux types soit primitifs ou complexes. Les types primitifs illustrés dans FIG 2.4 correspondent aux types de données du langage Java, Mise à part le type STRING qui est l'équivalent du type VARCHAR du langage SQL. Les types Complexes (illustrés au FIG 5.11)sont de type : STRUCT qui est un enregistrement encapsulant un ensemble de champs, ainsi que des deux types ARRAY et MAP semblables à leurs homonymes en langage Java .

Category	Type	Description	Literal examples
Primitive	TINYINT	1-byte (8-bit) signed integer, from -128 to 127	1
	SMALLINT	2-byte (16-bit) signed integer, from -32,768 to 32,767	1
	INT	4-byte (32-bit) signed integer, from -2,147,483,648 to 2,147,483,647	1
	BIGINT	8-byte (64-bit) signed integer, from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	1
	FLOAT	4-byte (32-bit) single-precision floating-point number	1.0
	DOUBLE	8-byte (64-bit) double-precision floating-point number	1.0
	BOOLEAN	true/false value	TRUE
	STRING	Character string	'a', "a"
	BINARY	Byte array	Not supported
	TIMESTAMP	Timestamp with nanosecond precision	1325502245000, '2012-01-02 03:04:05.123456789'

FIGURE 4.11 – Les types de données complexes de HiveQL .

4.1.7.3 Les bases de données sous Hive

Une base de données dans Hive se présente tel un catalogue ou un espace de tables. La création d'une base de données implique la création d'un répertoire à l'emplacement spécifié par la propriété "hive.metastore.warehouse.dir". Ce répertoire se compose d'un ensemble de sous répertoires associés à chaque table de la base. Dans le cas de la non spécification d'une base de données, Hive désigne la base de données par défaut. La suppression d'une base de données dans Hive s'effectue seulement après la suppression de toutes les tables ou en ajoutent le mot clé CASCADE à la syntaxe de suppression de la base

1. Les tables Hive

Une table Hive est constituée des données stockées et les métadonnées associées décrivant l'agencement des données de la table. Les données sont stockées généralement dans HDFS, comme elles peuvent résider dans un répertoire externe spécifié lors de la création des tables . Lors de la création d'une table, Hive ajoute deux propriétés à celle-ci. La propriété last_modified_by qui contient le dernier utilisateur qui a modifié la table. Et la propriété last_modified_time qui contient le moment de la la dernière modification [67]. Dans Hive, il y a deux types de tables :

- **Les tables internes :** aussi appelées managed qui sont des tables dont le cycle de vie de leurs données est contrôlé par Hive. L'entrepôt Hive stocke les données des tables internes dans un sous-répertoire du répertoire défini par hive.metastore.warehouse.dir, par défaut.
- **Les tables externes :** est une table dont les données sont stockées en dehors de l'entrepôt Hive. La création et la suppression des données sont contrôlées par l'utilisateur, en spécifiant l'emplacement des données pendant la création. Généralement les tables externes sont utilisées pour accéder à un ensemble de données initialement stocké dans HDFS par un autre. Comme elles peuvent être utilisées pour exporter les données pour un processus tiers .

2 . Format de stockage dans Hive

Il y a deux dimensions qui régissent le stockage de la table dans Hive, le format de la ligne et le format de fichier :

— **Le format de ligne :** est défini par SerDe(Sérialiseur/Désérialiseur). Un désérialiseur c'est dans le cas de l'interrogation d'une table et un sérialiseur, dans le cas de l'exécution d'une instruction INSERT .[67]

— **Le format de fichier :** détermine le format de conteneur pour les champs dans une rangée. Le format le plus simple est un simple fichier texte, mais il y a des formats binaires en colonnes ligne orientée .

4.1.7.4 Les limites de Hive

• Toutes les requêtes standard ANSI SQL ne sont pas prises en charge par le langage de requête Hive (HiveQL).

• Le langage Hive ne supporte pas l'insertion au niveau des lignes, la mise à jour et la suppression.

• Après la création de la table, le type de colonne ne peut être modifié. [69]

Conclusion

Nous avons présenté dans ce chapitre les différents composants d'Hadoop, principalement, HDFS et le modèle de programmation MapReduce. on a présenté également l'outil d'entreposage de données Hive ,en se basant sur les aspects théoriques pour pouvoir réaliser et implémenter la solution complète dans le chapitre qui suit.

Chapitre 5

Analyse et Conception

Introduction

La ville est de plus en plus instrumentée et l'information circulera partout et conditionnera la qualité de la vie et la qualité de la ville. L'objectif de cette instrumentation peut être l'optimisation de la consommation énergétique, le monitoring urbain (gestion de l'eau, de la circulation ou suivi de trajectoire , de la voirie,...), la maîtrise de risques naturels (inondations, incendies...) ou écologiques, technologiques ou urbains , l'amélioration de la qualité de vie.

Dans ce contexte, des masses de données sont ou vont être disponibles qu'elles soient sous formes de flux, structurées dans des bases de données ou moins structurées au sein de nombreux corpus. Ces données multisources et multiformats peuvent être issues de capteurs matériels (prises de mesures, de photos, de sons,...), d'enquêtes terrains (questionnaires, sondages, inventaires...), d'analyses statistiques mais aussi de réseaux sociaux ou plus communément d'annuaires ou d'archives. Une approche originale est de considérer tous ces systèmes comme des "capteurs", au sens captation d'information.

On retrouve cette approche adoptée dans diverses domaines :

- Domaine commercial
- Domaine médical
- Domaine environmental
- Domaine de surveillance

Actuellement, les applications émergentes dans le domaine géographique, nécessitent de plus en plus l'exploitation des informations géolocalisées provenant de capteurs, notamment pour la gestion de crises, la gestion de véhicules en temps réel, la gestion de risques urbains ou environnementaux, etc. Plus particulièrement, la surveillance de phénomènes environnementaux (i.e. inondations, avalanches, volcans...) est un processus complexe qui consiste à évaluer de façon continue les flux de données provenant de capteurs.
Le principal objectif est de **détecter un comportement inhabituel et de communiquer les informations aux experts afin d'éviter les dommages graves au sein des populations ou/et de l'environnement,et Comment structurer les données issues des capteurs ?**

Dans ce chapitre, nous introduisons dans la premiére parties les concepts de base de la ville intelligente ainsi les spécificités de systéme de surveillance environementale sur lesquels nous nous sommes appuyée pour l'analyse des données géolocalisées (spatio-temporelles) issues de capteurs. Ces spécificités concernent principalement l'architecture du système de surveillance environnemental, ainsi que la gestion de données issues de ce type de système, Dans la deuxiéme partie on a présenté les démarches suivie pour realiser la conception de notre systéme de surveillance envirenmental.

5.1 Concepts de base et définitions de la ville intelligente

Dans cette section nous allons donner quelques concepts se rapportant à la ville intelligente à savoir : **sa définition, ses composantes, ainsi que les différents éléments qui constitue la smart city.**

5.1.1 Définition de la ville Intelligente

Une "Smart city" se définit comme étant une large communauté (gouvernements, citoyens et entreprises) interconnectée en combinant des infrastructures de communications basée sur des standards et sur des services innovants afin de répondre à leurs besoins.[70]

Autrement dit l'expression "ville intelligente"désigne :

1. Une ville utilisant les technologies de l'information et de la communication (TIC) pour "améliorer" la qualité des services urbains ou encore réduire ses coûts.
2. Désigne un type de développement urbain apte à répondre à l'évolution ou l'émergence des besoins des institutions, des entreprises et des citoyens, tant sur le plan économique, social, qu'environnemental.

La notion de "ville intelligente", traduction de "smart city", s'est imposée dans les années 2000 avec l'émergence des nouvelles technologies. Lyon, Amsterdam, Le Caire, Dubaï, Yokohama, Edimbourg, Malte... comptent parmi les précurseurs de ce phénomène. Ces villes ont cherché à développer un service urbain apte à répondre aux attentes et besoins de tous les acteurs d'une même ville : citadins, entreprises, administrations, collectivités ou touristes.[71]

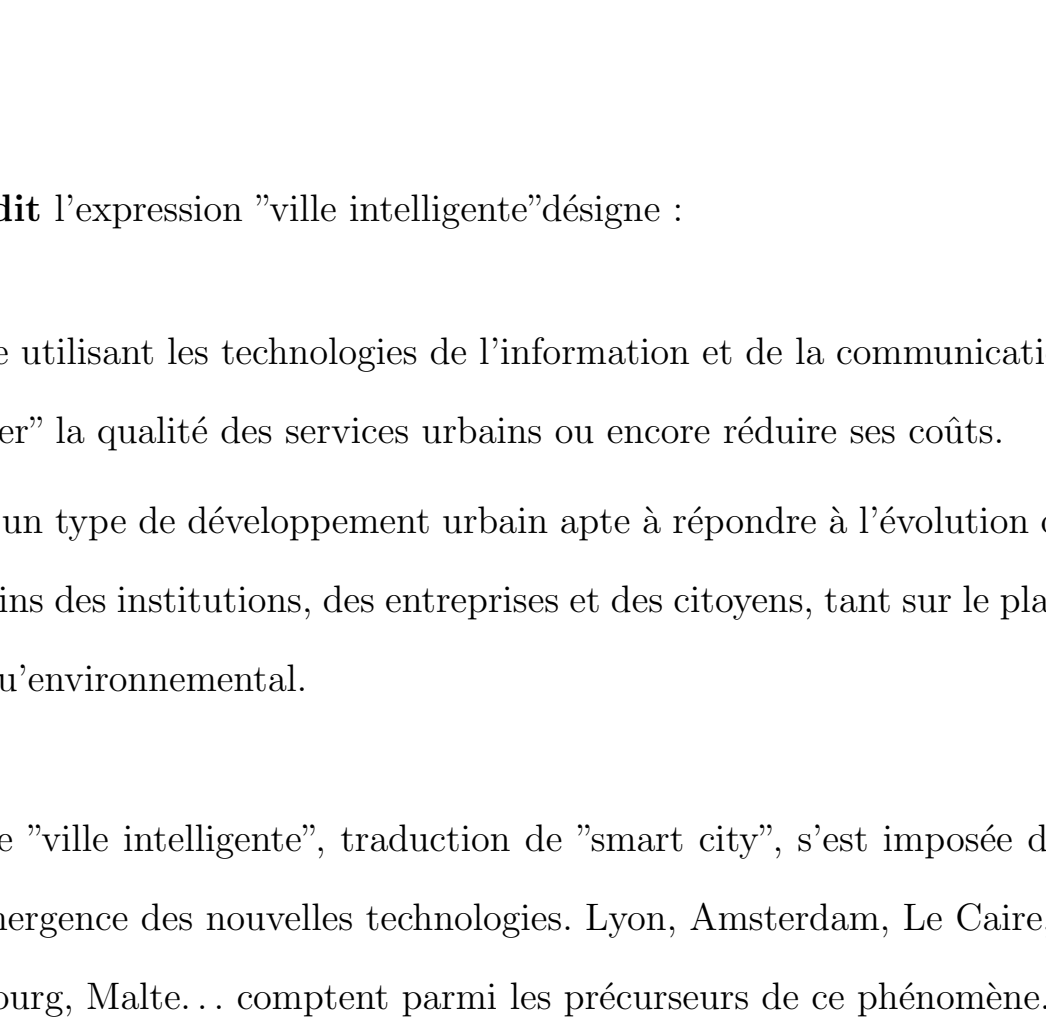


FIGURE 5.1 – Les outils permettant d'améliorer la fluidité de la ville [71]

5.1.2 Les six composantes de la ville intelligente [72]

Différents modèles de ville intelligente sont présentés dans la littérature. Le modèle holistique de Giffinger [1] est celui qui est le plus souvent utilisé pour démontrer les six composantes de la ville intelligente qui sont présentées en détail dans la présente section.

Le modèle de ville intelligente présenté ci-dessous, de Rudolf Giffinger, (Giffinger, s.d.) expert en recherche analytique du développement urbain et régional de l'université technologique de Vienne, présente les six leviers à considérer pour devenir une ville intelligente.

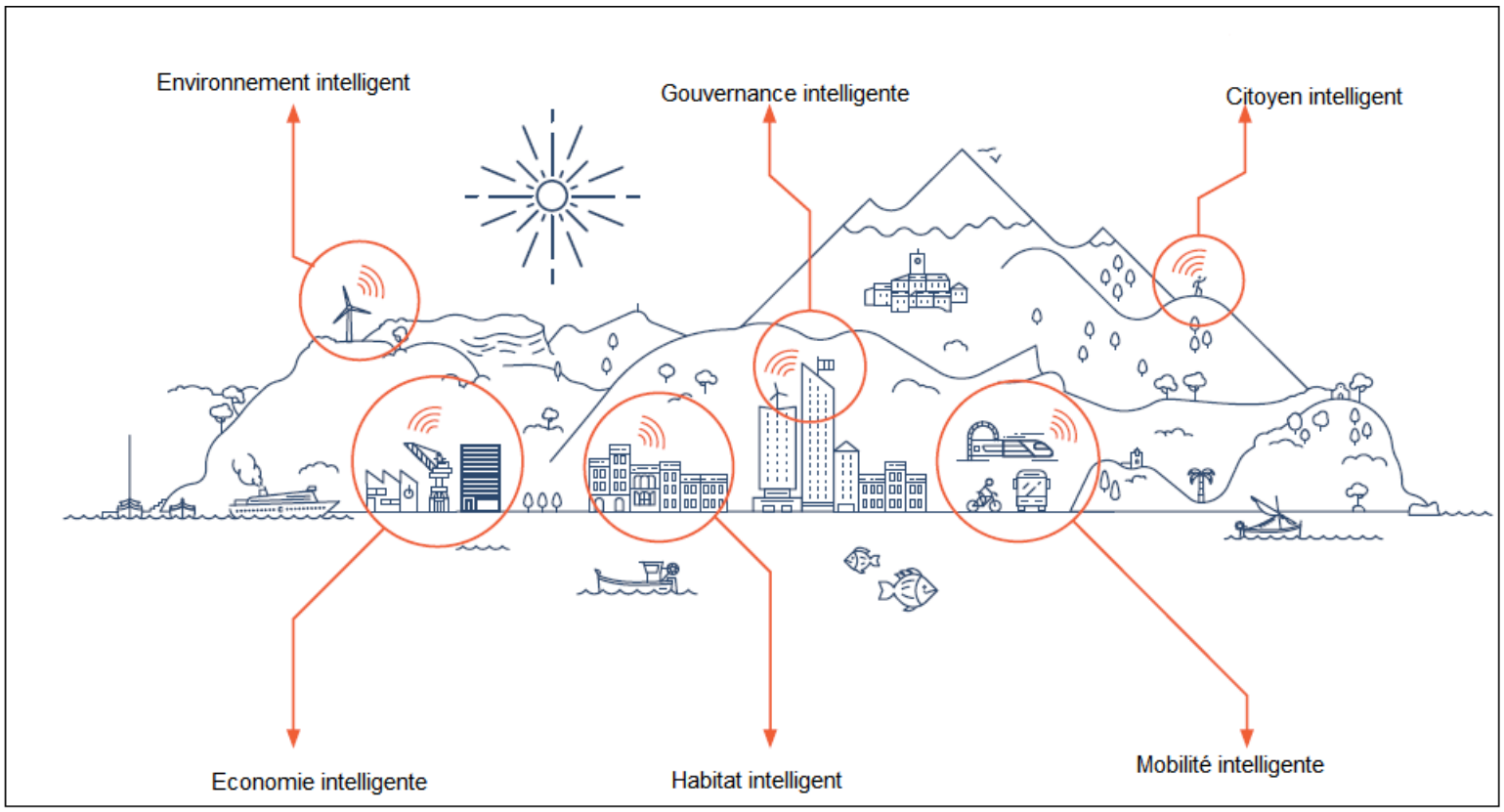


FIGURE 5.2 – Schéma des six leviers d'une ville intelligente (Inspiré de : Giffinger)

1. Gouvernance intelligente (Smart Governance)

La gouvernance à l'ère du numérique est collaborative, plus connectée et plus transparente grâce aux outils technologique, c'est-à-dire que les NTIC [2] servent de levier entre les décideurs, les acteurs publics ainsi que les citoyens.

Pensons notamment à des tableaux électroniques dans des lieux publics qui peuvent afficher de

1. Giffinger : Expert en recherche analytique sur le développement urbain et régional à l'université technologique de Vienne
2. Nouvelles Technologies de l'Information et des Communications

l'information à l'intention des citoyens ou encore à une diffusion web simultanée des rencontres du conseil pour permettre à un plus grand nombre de personnes d'y assister.
Cette gouvernance qui est dite intelligente est celle qui saura briser les silos au sein de l'administration et des services municipaux et qui permettra la collaboration étroite entre les différents acteurs et les citoyens. La ville devrait interagir avec les citoyens en direct, et ce, grâce à divers outils web dont des interfaces d'accès instantané .

Exemple

Projets d'e-gouvernance dans le domaine de la sécurité sociale et de la sante est la carte de sécurité sociale :

La Caisse nationale de la sécurité sociale des travailleurs salaries (CNAS) a réussi la dématérialisation de la carte de sécurité sociale. Le nouveau système permet de gérer cent millions de feuilles de soins électroniques par an grâce à des cartes à microprocesseur qui sécurisent les données de l'assuré et du professionnel de sante tout en garantissant une traçabilité des prescriptions.

2. Citoyen intelligent (Smart People)

Le citoyen est une importante partie prenante dans la ville intelligente. En effet, sa participation est requise, que ce soit dans la phase de consultation en amont ou pendant la phase de mise en œuvre, comme acteur pour la protection de l'environnement, en matière d'économie ou dans le volet social au sein de sa communauté. Ensuite, le citoyen intelligent est celui qui utilisera les nouveaux outils technologiques, notamment pour participer aux débats publics et à la vie de quartier.

Exemple

Les économies d'énergie ne peuvent pas être réalisées simplement avec des compteurs intelligents dans une maison. Afin de réduire la consommation d'énergie et d'économiser les factures, les consommateurs doivent non seulement surveiller leur consommation d'énergie, mais aussi s'efforcer de modifier le comportement quoti-

dien de toute la famille en matière d'utilisation de l'énergie.

3. Économie intelligente (Smart Economy)

Une économie intelligente, c'est un pilier économique dont on se sert comme vecteur pour l'innovation et la création d'emplois durables pour la ville.
Selon Giffinger, une économie intelligente est basée sur un esprit d'innovation et d'entrepreneriat, sur la productivité et la flexibilité du marché. Elle possède aussi une aptitude à se transformer et à enchâsser le marché international.
L'analyse d'une multitude de données en plus de l'accès à de nouvelles sources d'information permettra aux villes de créer de nouvelles opportunités, de la prospérité et de nouveaux emplois. Une des principales motivations de devenir intelligente est le pouvoir de devenir une ville attrayante sur la scène internationale, mais surtout un désir de développement économique.

4. Mobilité intelligente (Smart mobility)

L'accès aux données de transport en temps réel via des écrans électroniques dans les stations, dans les wagons de métro ou dans les autobus ou encore via les téléphones intelligents personnels permettrait aux usagers de connaître une foule d'informations,c'est-à-dire :

— l'état de la circulation sur le réseau routier,
— le temps d'attente aux arrêts et stations de transport en commun,
— les pannes et en somme une meilleure gestion des flux urbains.

Une mobilité intelligente qui serait possible grâce aux divers centres de gestion des données, aux capteurs d'informations et aux caméras. Ainsi, les utilisateurs des transports deviennent des producteurs de données, elle passe aussi par le développement et l'accès aux applications qui permettront aux usagers de vivre l'expérience d'une mobilité intelligente.

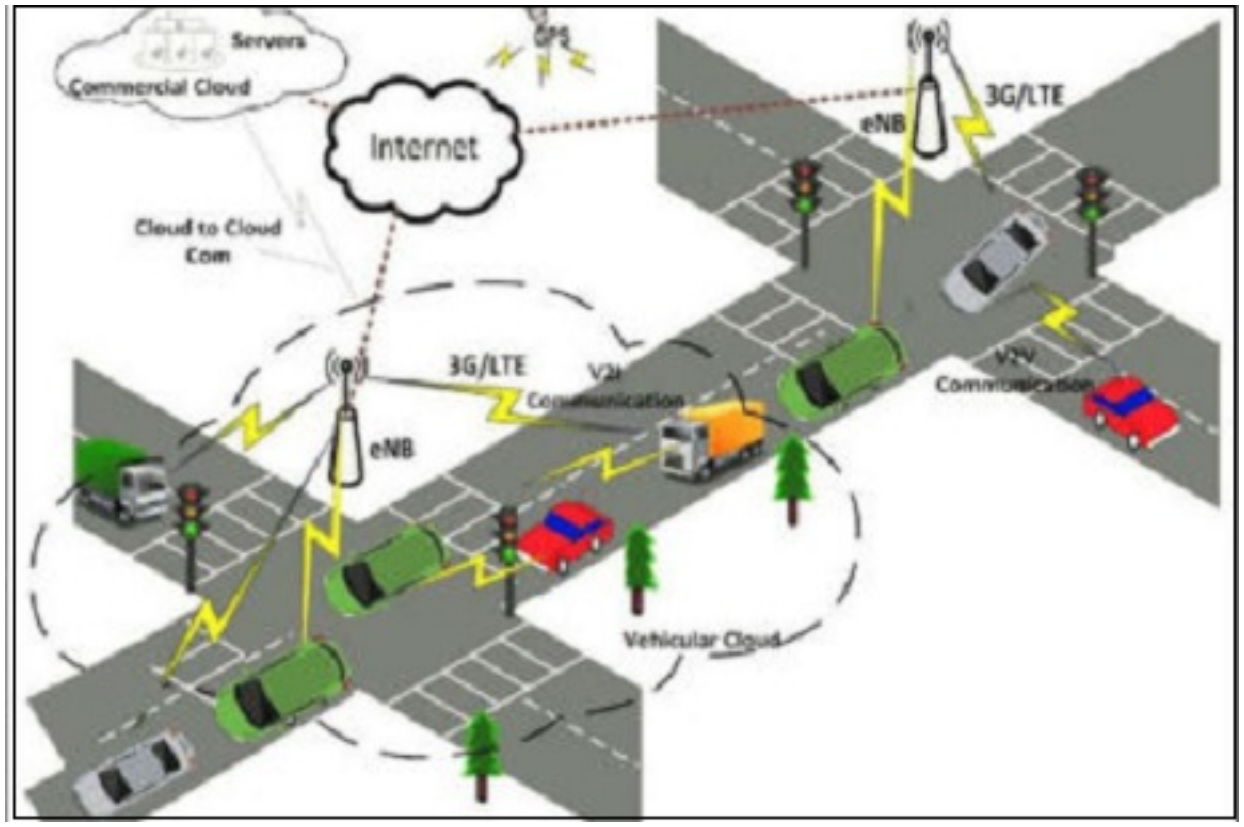


FIGURE 5.3 – Mobilité et Transport Intelligents

5. Environnement intelligent (Smart Environment)

La gestion de l'eau, la gestion des déchets et la gestion de l'énergie sont au cœur des préoccupations d'une ville en matière d'environnement.

Dans une ville intelligente, les divers outils technologiques permettent notamment une protection et une préservation de nos ressources et des milieux naturels,

Exemple

des capteurs pour mesurer le niveau de pollution de l'air. Il s'agit là de nouvelles technologies qui permettent de fournir une panoplie d'informations en temps réel. Des capteurs pour surveiller les variations du niveau de l'eau dans les rivières, les barrages et les réservoirs.

En matière d'énergie, les "smart grids "[3]*, une technologie informatique des réseaux de distribution d'électricité intelligents, peut optimiser la production et la distribution d'électricité tout en s'ajustant à la demande. Économiser de l'énergie via de nouvelles technologies c'est aussi ça un environnement intelligent.*

3. ***Les smart grids*** *sont des systèmes énergétiques à régénération automatique, à surveillance numérique, qui fournissent de l'électricité ou du gaz à partir de sources de production.*

— *Équiper les infrastructures de la ville et mettre en place des NTIC dans le domaine de l'environnement a pour objectifs la protection de l'environnement, une utilisation durable des ressources et la mise en valeur des milieux naturels.*

6. Habitat intelligent (Smart Living)

L'habitat intelligent peut être applicable à différentes échelles :

— **À l'échelle du milieu de vie,** il peut s'agir d'un milieu de vie sécuritaire, où foisonne la culture et qui offre des services de santé et d'éducation. De plus, il peut s'agir de développer des quartiers verts ou des éco-quartiers qui peuvent être par exemple élaborés dans le cadre de différents programmes, dont l'Agenda 21 [4].
— **À l'échelle de l'habitat,** il peut s'agir d'habitations écologiques, voire des habitations qui sont certifiées selon le Leadership in Energy and Environmental Design (LEED) [5]

Exemple : Les bâtiments intelligents

1. Utilisent différents systèmes pour assurer la sécurité et la sûreté des bâtiments, la maintenance des actifs et la santé globale des environs.

2. Ces systèmes comprennent la mise en œuvre d'une surveillance à distance, de la biométrie, de caméras de surveillance IP et d'alarmes sans fil afin de réduire les accès non autorisés aux bâtiments et les risques de vol. Il inclut également l'utilisation du contrôle d'accès au périmètre pour bloquer l'accès aux zones restreintes du bien et détecter les personnes se trouvant dans des zones non autorisées.

4. **L'Agenda 21** est un programme international de mise en œuvre du développement durable pour le XXIe siècle, d'où son nom

5. LEED est une certification de bâtiment dépendant d'une organisation gouvernementale à but non lucratif. Le système de notation LEED aide les professionnels à améliorer la qualité de leurs bâtiments et leur impact sur l'environnement.

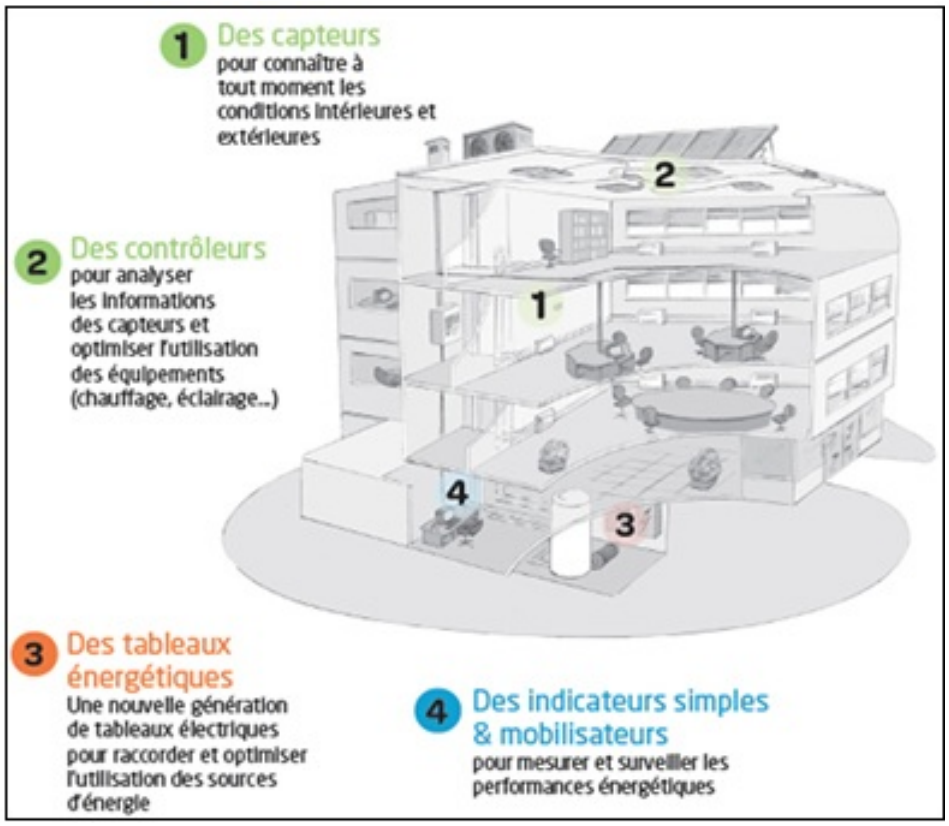


FIGURE 5.4 – Les bâtiments intelligents

5.2 Système de surveillance environnemental

On s'intéressera dans le cadre de notre travail à deux composantes qui sont : **Environnement intelligent et L'habitat intelligent** pour lesquels la récolte, le traitement et le stockage d'information sont pris en charge par un système de surveillance.
les systèmes de surveillance environnementaux, sont composés principalement par des réseaux de capteurs sans-fil (i.e. géocapteurs, réseaux de capteurs environnementaux), qui possèdent un ensemble de capteurs ou noeuds de capteurs, ainsi qu'un système de communication qui permet aux données de joindre un serveur central **(Figure 5.5)**.

On notera que dans les systèmes de surveillance environnementaux, les réseaux de capteurs sont composés par des capteurs fixes et agiles qui forment traditionnellement la majeure partie de ceux utilisés dans ce type de systèmes, alors que les capteurs mobiles sont souvent dédiés au suivi ou à la gestion de flottes. Dans ce type d'architecture, les noeuds capteurs collectent des données de façon automatique ; un réseau est utilisé pour transférer les données à la passerelle ou entre plusieurs passerelles, avant d'arriver au serveur central, appelé également centre de données.

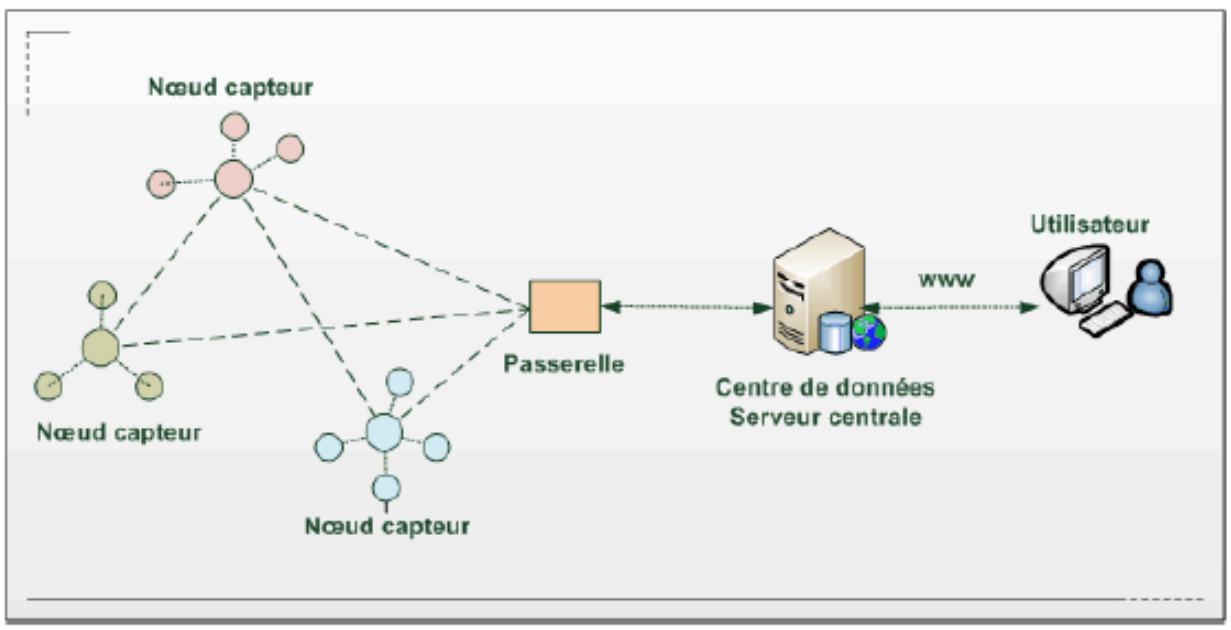


FIGURE 5.5 – Architecture technique d'un système de surveillance environnementale

Au niveau du serveur centrale , ces données sont traitées et ensuite exploitées soit par l'utilisateur, soit par un SIG (Système d'Information Géographique) ou par le web.
Dans la structure d'un système de surveillance environnemental, nous considérons trois couches principales dans lesquelles divers processus de gestion en temps réel sont effectués : **la couche Acquisition**, **la couche Traitement**, et **la couche Exploitation.(Figure 5.6).**

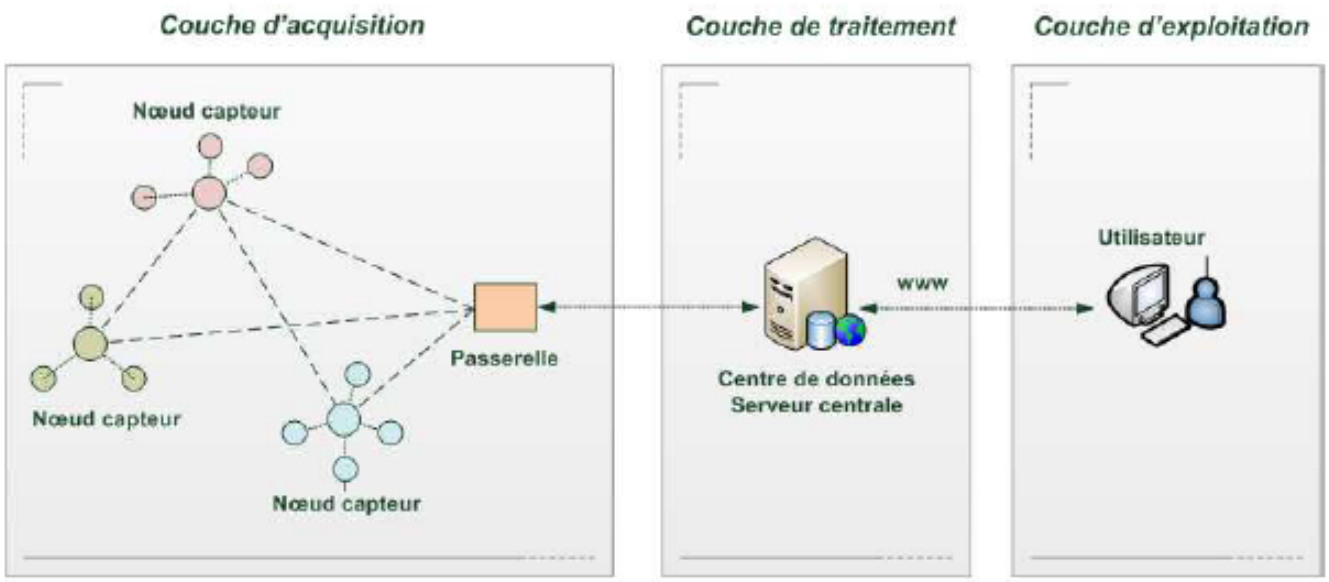


FIGURE 5.6 – Couches de traitement dans les systèmes de surveillance environnementaux

i- Couche Aquisition

Au niveau de **la couche d'acquisition**, les réseaux sont composés par des noeuds/**capteurs**

fixes ou agiles. Différentes types de données peuvent être fournis par ces noeuds/capteurs dépendant du type d'application.

Les données acquises, appelées aussi **données observation** correspondent à des éléments comme **l'humidité, la température, la pression,la monté du niveau d'eau** , etc., et peuvent être représentées sous différents formats, de façon numérique, analogique, spatiale, temporelle, spatio-temporelle, multimédia (image, vidéo), etc .

C'est au niveau de cette couche que les premières décisions concernant les éléments à mesurer et les données résultantes sont prises.

Au sein des noeuds capteurs, les données peuvent être stockées, agrégées ou filtrées afin de déterminer quelles données seront transmises, selon les conditions d'acquisition, et en considérant les capacités physiques et de calcul des noeuds/capteurs. Lors de leur acquisition et prétraitement, ces données sont transférées au centre de données, au sein du quel le processus de traitement se déroulera.

ii- Couche Traitement

Au niveau de **la couche Traitement,** le traitement des données collectées est réalisé. Leur traitement est effectué sur un serveur, dont les capacités d'énergie, de stockage et du calcul deviennent plus significatives. Même si le prétraitement est réalisé au sein des noeuds, ce processus a normalement pour but de filtrer et réduire la quantité des données à gérer au sein de la mémoire principale, grâce à l'agrégation des données, par exemple. Comme nous l'avons détaillé précédemment, afin d'assurer l'acquisition d'information le plus possible, et éviter la perte d'information à la volée, des approches externes pour la localisation des données, sont utilisées au niveau des systèmes de surveillance environnementale en évitant l'agrégation des données. Ces données seront par la suite mises à disposition aux utilisateurs.

iii- Couche Exploitation

Au niveau de **la couche Exploitation**, les informations sont traitées au centre des données par un SIG, combinées avec d'autres outils comme internet, qui permettent l'accès à distance aux données. Ceci représente un réel avantage pour les utilisateurs, qui peuvent maintenant avoir un accès plus simple à l'information. Ces informations peuvent être exploitées en temps

réel ou à posteriori selon les besoins de l'utilisateur et leur application. Dans des applications du type surveillance, le temps réel est le plus recommandé, permettant de suivre l'évolution d'un phénomène et réagir par rapport à son comportement. [73]

5.2.1 Spécificités des systèmes de surveillance environnementaux

Les systèmes de surveillance environnementaux ont fait preuve de notables développements technologiques, qui ont permis l'amélioration de la surveillance de l'environnement naturel. Ces systèmes peuvent être organisés de diverses façons, chaque application et système, présentant des particularités en fonction des besoins des organisations ou des utilisateurs.
De façon générale, nous pouvons signaler que le contexte d'application dans les systèmes de surveillance est assez dynamique et possède des caractéristiques liées au temps réel, qui le distinguent des applications traditionnelles.

Les données présentent aussi des caractéristiques dynamiques, voire l'évolution de la donnée au cours du temps. Par conséquent, ce type de données est notamment géré en temps-réel, et ceci n'est pas anodin car l'environnement change aussi au cours du temps.

Dans ce contexte, en adéquation avec les modèles existants de systèmes de surveillance environnementaux, nous considérons pour notre approche, les caractéristiques suivantes :

a- Spécificités liées au contexte d'acquisition des données capteurs

— Les noeuds capteurs contiennent de l'information géolocalisée (données spatiotemporelles), les mesures sont localisées par les différentes techniques exemple, le GPS ou la localisation distribuée de services comme la triangulation.
— Les noeuds capteur possèdent des propriétés fonctionnelles liées à la persistance, la gestion d'erreur, la gestion du cycle de vie, et leur déploiement physique. Ces capteurs utilisent des métadonnées pour leur identification et leur découverte.

b- Spécificités liées au traitement des données capteurs

— Les systèmes de surveillance sont de façon inhérente des systèmes temps réel, à savoir que

les actions de contrôle et surveillance doivent être exécutées dans les systèmes physiques dans des délais de temps impartis. Dans notre contexte, le temps réel ne veut pas dire ”rapide”, mais que les délais sont limités, prévisibles et gérables (soft real-time).

— Les mesures issues de capteurs sont valides (actuelles) pour un temps limité.

— Les mesures issues de capteurs sont enregistrées dans une base de données.

— Toutes les mesures collectées n'ont pas la même priorité pour être traitées selon différents critères, plus spécifiquement selon la période d'acquisition. Les mesures acquises en période de crise, sont plus importantes et doivent être prioritaires.

5.2.2 Données capteurs et temps réel

Dans les systèmes de surveillance environnementaux, l'accès aux données implique dans certains cas des propriétés temps réel. Le temps réel doit assurer qu'une tâche est réalisée selon les contraintes de temps spécifiées ; les requêtes doivent être achevées dans un temps limité.

Exemple

En cas d'inondation , les alertes doivent être disponibles aux informations dans un délai imparti. Les données capteurs sont par nature temps réel.

FIGURE 5.7 – Surveillance des inondations et alerte de crues par SMS

Dans les systèmes de surveillance du type environnemental, les contraintes de temps souples sont les plus adéquates. Ces contraintes sont les moins exigeantes et gênent moins l'exécution

du système, toutefois tout non-respect de ces contraintes est à éviter.
En comparaison à un système de surveillance nucléaire ,par exemple, dont les contraintes du temps sont beaucoup plus strictes et aucun délai n'est admissible, les systèmes de surveillance environnementaux ont une criticité plutôt épisodique et le respect des contraintes n'est pas toujours soutenu, car les phénomènes naturels sont souvent imprévisibles. Dans ce cas, les données repérées dans une situation de crise lors de la détection d'un événement, doivent être prioritaires et les contraintes de temps respectées. [73]

5.3 Contexte réel d'étude (La ville intelligente dans le contexte du phénomène d'inondation)

La problématique des risques naturels et plus particulièrement du risque d'inondation est un sujet d'actualité en France et dans le monde entier, donc La gestion de ce risque devient de plus en plus une nécessité qui doit impliquer tous les acteurs concernés (décideurs, techniciens et population) afin d'identifier les enjeux à protéger, les moyens disponibles et alternatives possibles pour atténuer les dégâts humains et matériels provoqués par ce phénomène. Le présent travail consiste à l'étude de protection de la ville Aix-les-Bains, département Savois, contre les inondations.

5.3.1 Définition du phénomène d'inondation

Le risque d'inondation peut être défini comme un événement dommageable, doté d'une certaine probabilité, lié à la conjonction de l'aléa inondation et de la vulnérabilité de la société. Il correspond à une contrainte hydrologique potentielle, car l'aléa est caractérisé par des variables en terme d'extension spatiale, de hauteur et de vitesse d'eau, dépendantes de sa fréquence ; potentielle aussi, par la définition même de la vulnérabilité de la société. Celle-ci est évaluée par la mesure des dommages matériels ou tangibles qu'elle pourrait subir en cas d'inondation mais aussi par l'appréciation des dommages non matériels ou intangibles (préjudice moral), susceptibles de peser lourdement sur le niveau de bien-être des individus. La notion de vulnérabilité traduit ainsi la mesure des dommages rapportée à l'intensité de l'aléa.

On peut alors dire qu‘une inondation est une submersion rapide ou lente d’une zone habitée ordinairement hors d‘eau. Ainsi, le risque inondation est la conséquence de deux composantes : l‘eau qui peut déborder de son lit habituel d‘écoulement et l‘homme qui s‘installe dans l‘espace alluviale. L‘importance de l‘inondation dépend de la hauteur d‘eau, la vitesse du courant et la durée de la crue. Ces paramètres sont conditionnés par la précipitation, l‘état du bassin versant et les caractéristiques du cours d‘eau (profondeur, largeur, etc.). Ces caractéristiques naturelles peuvent être aggravées par la présence d‘activités humaines. .

5.3.2 Présentation de quelques systèmes d’alerte existant à l’heure actuelle

Pour être averti en cas de crues, il existe divers système d’alerte et parmi ces derniers on cite :

1. **Le système telegrafia**

La solution fournie par la société Telegrafia intègre le système d’alerte et d’avertissement avec un ou plusieurs systèmes de suivi. La partie d’alerte garantit une alerte à terme de la population sur le territoire menacé dans plusieurs niveaux et cela sur la base des informations issues des capteurs de suivi. La partie d’avertissement du système informe en même temps les personnes responsables et appelle les équipes de crise. Pour identifier les informations sur le danger potentiel l’on utilise plusieurs démarches en fonction du fait s’il s’agit des inondations occasionnées par un temps pluvieux à long terme et par les niveaux élevés des rivières ou bien par une pluie torrentielle et par les inondations qui en suivent. Les simulations informatiques aident à définir la probabilité élevée de la menace des terrains en question.
Le système anti inondation comprend :

- Un centre de commande d’alerte et d’avertissement qui peut fonctionner dans un mode sans opérateur.
- Une infrastructure de communication
- Capteurs de suivi
- Sirènes électroniques Pavian
- D’autres systèmes autonomes intégrés (système de sonorisation Amadeo, système

d´alerte industriel, etc.) [74]

FIGURE 5.8 – Le système anti inondation Telegrafia

2. **Le système Ogoxe(système d'alerte sonore)**

Ce nouveau système se compose de capteurs installés sur les cours d'eau, d'un logiciel destiné aux collectivités locales qui permet de voir en temps réel les niveaux et d'un objet connecté destiné aux particuliers qui pourront être alertés. "Nous récupérons les données des capteurs et nous réalisons des prévisions. Sur leur boîtier de la taille d'un smartphone, les particuliers ont un indicateur de couleur des niveaux de danger en temps réel et un autre qui prédit le danger à venir. Si le maire de la commune décide de l'évacuation, cela se met à sonner. C'est un moyen d'être plus réactif", détaille l'inventeur.[75]

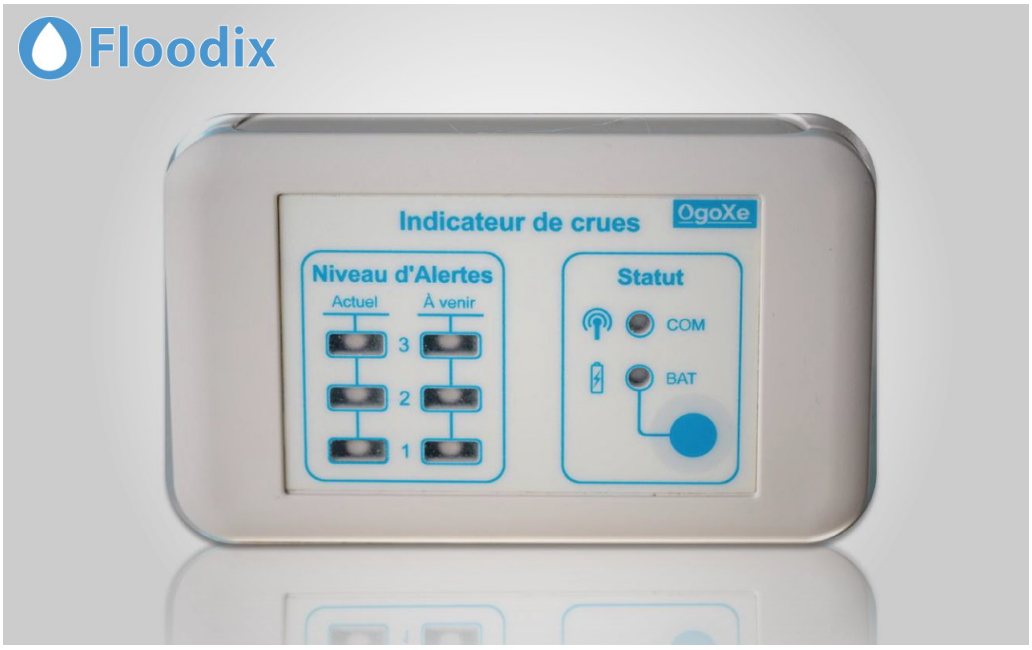


FIGURE 5.9 – Le système Ogoxe

Pour la mise en pratique de ce travail, on doit se servir des données véhiculées par une smart city et on prendra l'exemple de la ville de Sierroz, situées en France dans le département de la Savoie en région Auvergne-Rhône-Alpes.sur la commune d'Aix-les-Bains au cœur d'une zone urbanisée.

Les digues de protection contre les inondations du Sierroz, entre le Pont Rouge et le pont de la voie SNCF, occupent un linéaire d'environ 400 m en rive droite comme en rive gauche. La hauteur des digues au-dessus du fond du lit du Sierroz est de l'ordre de 4 m sur le linéaire étudié. Leur hauteur au-dessus du terrain naturel côté val atteint au maximum 5,2 m en rive gauche et 3,5 m en rive droite. La hauteur moyenne pour les deux rives sur l'ensemble du linéaire est de 2,3m environ.

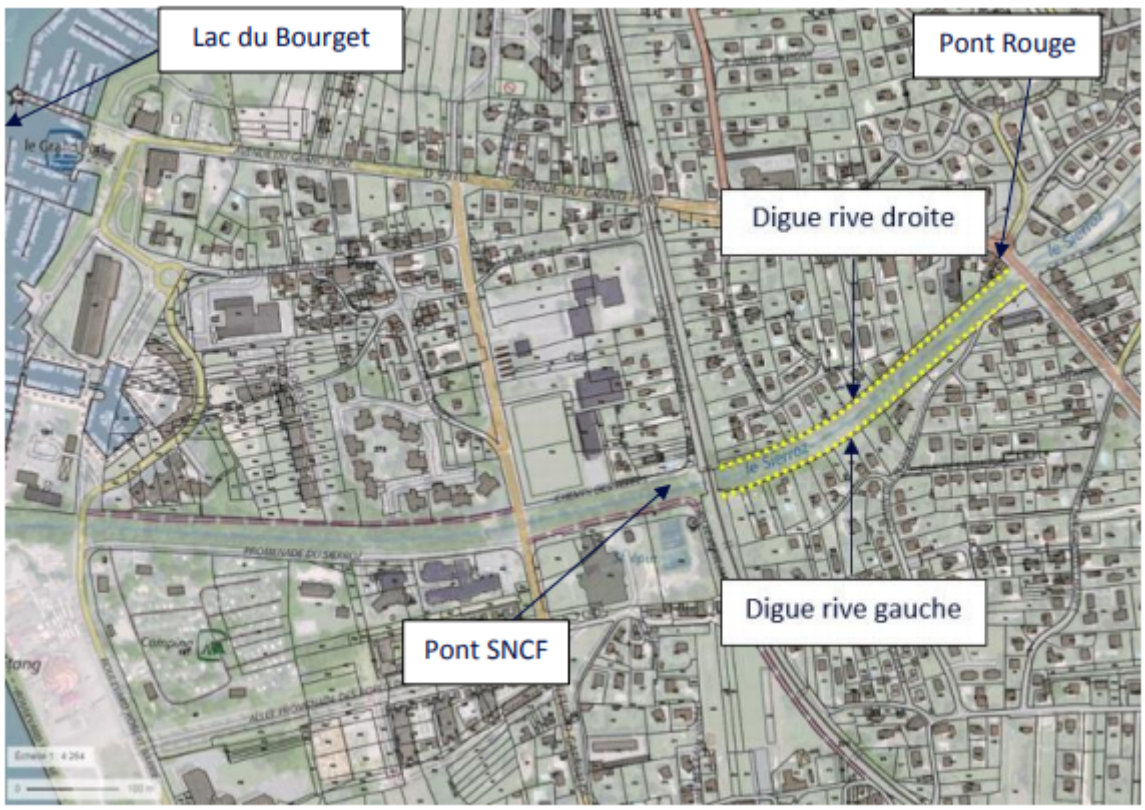


FIGURE 5.10 – Vue en plan de digues du Sierroz

Construites entre les années 1835 et 1875 (leur période de construction n'est pas connue précisément), ces digues ont pour fonction de protéger les quartiers pavillonnaires de Choudy, du Pont Rouge et des Painchins appartenant à la commune d'Aix-les-Bains.

Ces digues protègent environ 900 habitants des crues du Sierroz. Dans le cadre de la réalisation du Plan de Prévention des Risques d'Inondation (PPRI) du Bassin Aixois, approuvé en 2011 , les études hydrauliques et hydrologiques ont mis en évidence un fort risque de rupture par surverse de ces digues pour la crue centennale. En conséquence, des zones d'aléa fort, avec un risque potentiel de pertes humaines, ont été définies côté val des digues sur les deux rives.

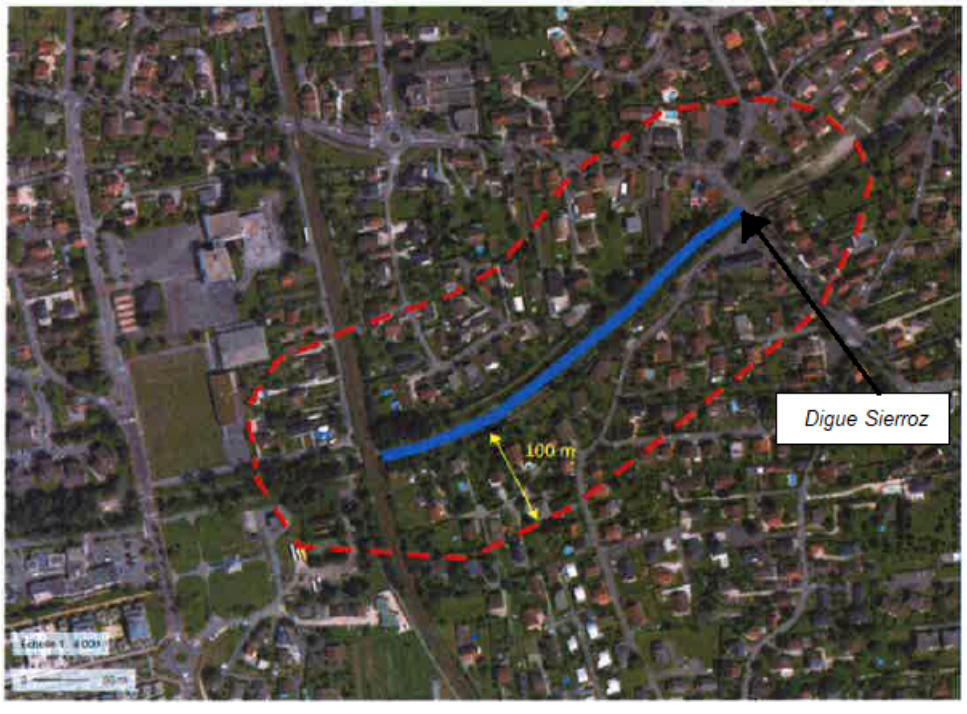


FIGURE 5.11 – Vue des digues de Sierroz

• La region Auvergne-Rhône-Alpes contient plusieurs communes **Aix-les-Bains, de Grésy-sur-Aix, d'Épersy, de Saint-Offenge-Dessous,de Saint-Offenge-Dessus du Montcel** ,Ces communes sont traversé par une rivière nommé **Sierroz,** cette dérnière sujette souvent à des inondations.Dans notre domaine d'étude on s'interèsse à la commune **Aix-les-Bains**.

• Réflexion sur les données utile pour ce systéme dans le contexte d'une inondation.

5.3.3 Présentation du système de surveillance à mettre en place

La ville intelligente telle que nous la présentons est constituée de différents éléments ,ces derniers véhiculent grace aux capteurs une grande masse d'information ,on retiendra essentiellement dans le tableau ci contre les éléments à prendre en considération dans le cadre d'une inondation.

dans le contexte de notre travail on s'intéresse particulièrement aux données produite par les sources suivantes :

Catégories	**Sous-catégories**	**Statistiques**
Routes	autoroute avenue boulevard	**175** rues, **154** chemins, **37** impasses, **26** boulevards, **21** places et **20** allées sont référencés sur Aix-les-Bains .
Habitations	collectif (Immeuble) individuel (Maison) autres hèbergements (hôtels,personnes âgées,étudiants...)	**29 799** habitants
Cours d'eau	Rivière	
Santé	Clinique Hôpital	4 cliniques 4 hopitaux
Securité/Secours	casernes de pompiers Centre Ambulancier gendarmerie police	**2** Casernes de pompiers **7** Ambulances **21** Gendarmeries **4** Commissariats

TABLE 5.1 – Les différents éléments composant une ville intelligente

5.4 Reflexion sur les scénarios en cas d'inondation :

Plusieurs scénarios peuvent être mis en place dans le cas d'une inondation, par exemple :

1. La fuite vers les zones de refuge tout en sachant où elles se trouvent sans l'intervention des agents de secours.
2. Détection des route les plus proches epour évacuer la population vers des unité de secours.
3. Diffuser des messages d'alerte et d'urgence par SMS

On peut imaginer plusieurs scénario, on s'interesse particuliérement à deux :

— **Scénario 1 : Surveillance de la montée du niveau d'eau**

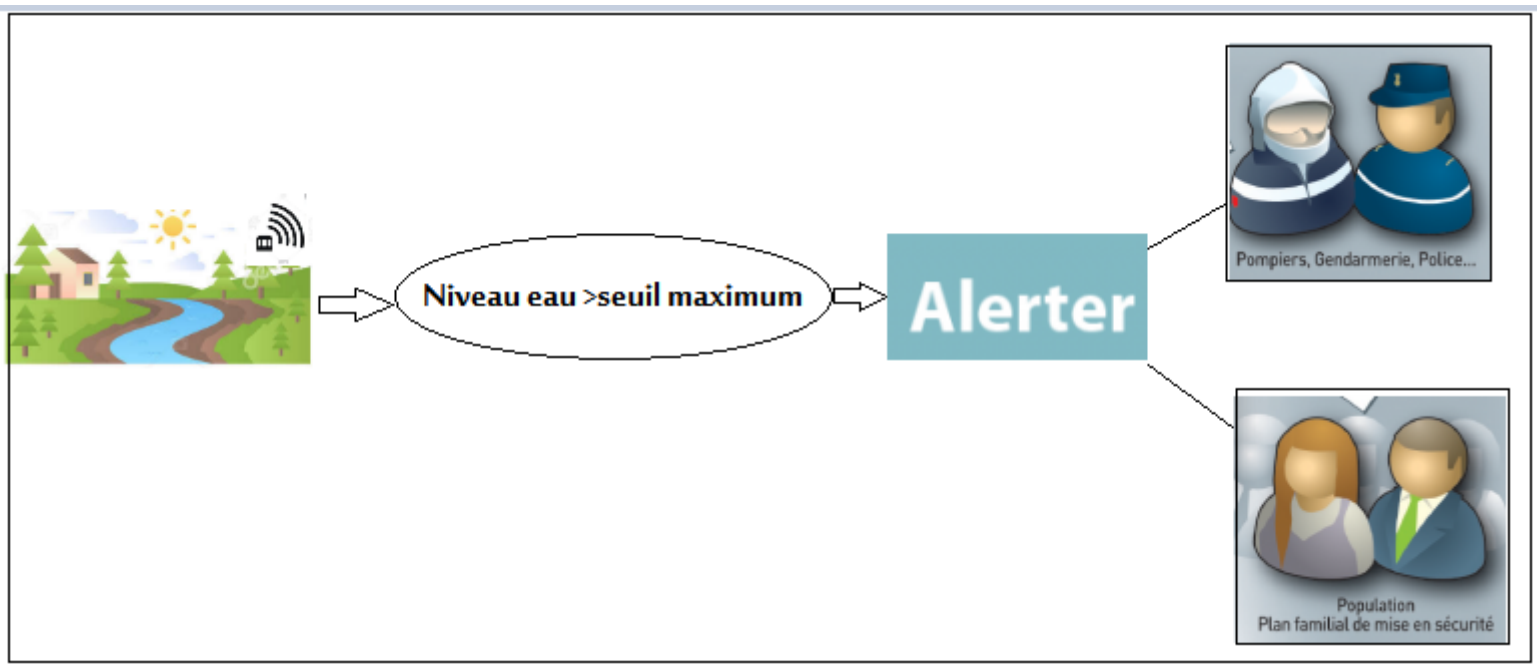


FIGURE 5.12 – Processus d'Alerte

Dès qu'une Alerte signalant que le niveau d'eau de la riviére a atteint une valeur maximale alrors alerter les secours et toute la populations.

— **Scénario 2 : Evacuation de la population vers unité de secours**

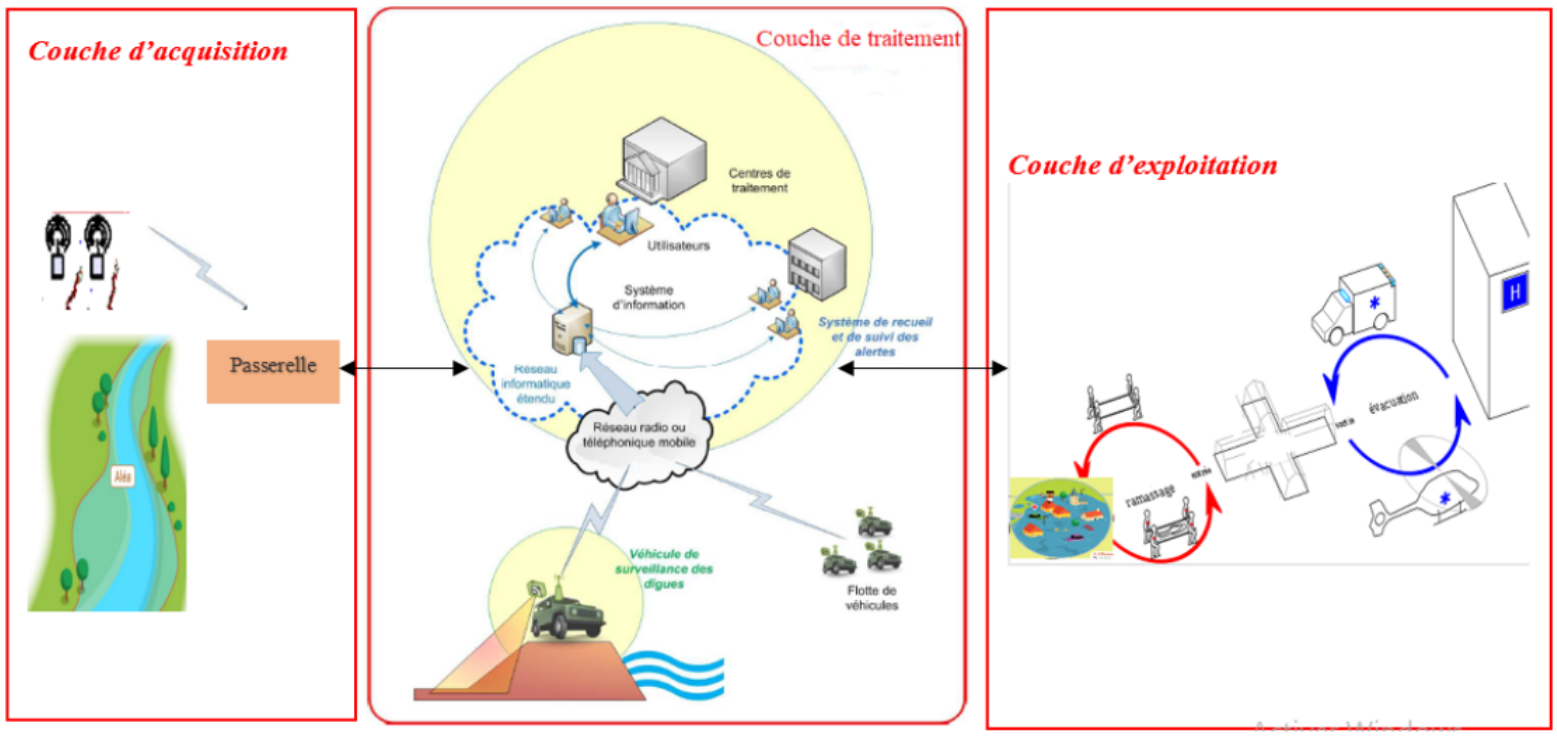


FIGURE 5.13 – Processus d'évacuation vers un hôpital

Dès qu'une Alerte a été donnée, les centres de secours (pompiers,centre ambulancier...) doivent procéder à l'évacuation des personnes qui se trouvent dans les zones inondée vers les hôpitaux les plus proche et cela à l'aide des capteurs placés sur les routes.

Les differents Capteurs mit en place sur les routes pour détécter l'itineraire le plus optimal pour le plan d'évacuation :

Capteur d'eau qui détécte l'etat de la route (inondée ,non inondée).

Capteurs analysant le degrés de congestion pour dire si la route est disponible.

Si une route ne posséde aucun signal d'alerte cela implique que la route est disponible.

5.4.1 Type des données relatives à la ville dans le cadre d'une innondation

Une donnée est un élément brut, qui n'a pas encore été interprétée, mis en contexte, Il existe deux type de données :données fixes et données changeantes.

- **Données fixes**

  Les données fixes (parfois appelées données permanentes ou données à contenu fixe)

sont des données qui ne peuvent pas être modifiées.

Exemple de données fixes : *Position géographique ou profondeur d'une riviére.*

- **Données changeantes**

Les données changeantes (parfois appelées données à contenu variable) sont des données qui peuvent être modifiées c'est-à-dire :le contenu de ces données change au fil du temps.

Exemple de données changeantes : *Niveau d'eau des riviéres ,Etat de la route (bloquée,non bloquée).*

Le tableau suivant illustre la récolte et la structuration de données de la ville :

Catégorie	**Données fixe**	**Données changeante**	**Description**
Capteur Riviere	Identificateur capteur riviére	Valeur Mesurée	Un capteur est un dispositif transformant une grandeur physique (température, pression, niveau d'eau, etc.) en un signal (souvent électrique) qui renseigne sur cette grandeur.
Rivière	adresse nom profondeur	niveau d'eau (monté,baissé)	. une rivière est un cours d'eau constitué d'un ensemble de capteurs dans le but de savoir le niveau d'eau (monté,baissé).
Route	Numéro d'identification Géolocalisation Nom de la route	etat de la route (inondée,non inondée)	chemin à suivre pour aller d'un lieu à un autre, constitué d'un ensemble de tronçons ,un capteur est placé sur chaque tronçon dans le but de savoir l'etat de la route
Capteur Route	Identificateur capteur route	Valeur Mesurée	Un capteur est un dispositif transformant une grandeur physique (température, pression, etat route etc.) en un signal (souvent électrique) qui renseigne sur cette grandeur.
Habitation	Identificateur Habitation Adresse Habitation Nom Habitation	etat d'habitation (inondée, non inondée)	action d'habiter, de séjourner d'une manière durable dans une maison, un immeuble Des capteurs sont mis en place pour détécter le niveau d'eau dans ces types d'habitations.
Sécurité et Secours	Adresse Nom Service Numéro de téléphone		assure la protection des personnes et des biens dans les campagnes comme dans les villes. Des capteurs sont mis en places dans chaque organisation pour détécter le niveau d'eau.
Santé	Adresse Nom Hôpital Numéro de téléphone		un hôpital et constitué d'un ensemble de d'étages,un capteur est placé sur chaque étage dans le but de détécter le niveau d'eau.

TABLE 5.2 – Récolte et structuration de données

5.5 Conception du systéme de surveillance envirenemental

5.5.1 Partie Données

1. Conception de la Base de Données pour Hadoop :

Nous proposons d'adopter une approche de modélisation des entrepôts de données basée sur les trois niveaux d'abstraction conceptuel, logique et physique.

Au niveau conceptuel, la modélisation multidimensionnelle repose sur les concepts de fait, de dimension et de hiérarchie. Cette modélisation consiste à décrire les données analysées au travers d'un schéma en étoile ou en constellation Le schéma conceptuel est ensuite traduit en un modèle logique lui-même traduit en modèle physique. L'approche la plus répandue est l'approche ROLAP consistant à implanter les bases de données multidimensionnelles en utilisant des SGBDR.

L'émergence des systèmes " Not-Only SQL " (NoSQL) permettent d'envisager de nouvelles approches pour implanter le schéma d'un entrepôt de données. Ces systèmes offrent l'avantage de gérer de grandes masses de données (mégadonnées ou ' big data ') et sont facilement extensibles. Contrairement aux systèmes relationnels, les systèmes NoSQL reposent sur des approches de dénormalisation des données, afin d'éviter les calculs de jointure. En contrepartie, une certaine redondance dans les données est tolérée

En général, plus le modèle de données se complexifie, plus l'écriture d'un job MapReduce qui les manipule devient fastidieuse. Si nous prenons le simple exemple du **Word count** que nous trouvons sur la documentation officielle de Hadoop, l'implémentation Java7 fait une centaine de lignes environ avec :

- 15 lignes pour le mapper,
- 12 lignes pour le reducer,
- 35 lignes pour le setup ainsi que les méthodes utilitaires pour le parsing des données en entrées,
- 30 lignes pour le main.

Afin de faciliter l'analyse de données stockées dans HDFS sans passer par la complexité de

MapReduce, certains frameworks comme Pig, Hive sont apparus. Ils proposent des langages de haut niveau pour lancer des requêtes sur Hadoop .

2. La Modélisation

Il existe plusieurs types de modèles qui sont mis en œuvre dans les DataWarehouse et qui utilisent ces notions de table de faits et de dimensions :

a- Schéma en étoile : est un schéma relationnel dans lequel une table centrale contenant les faits anaylisés référence des tables de dimensions, chaque dimension étant décrite par une seule table dont les attributs représentent les diverses granularité .[76]

b- Schéma en flocon : est un schéma relationnel dans lequel une table centrale contenant les faits analysés référence des dimensions du premier niveau, chaque dimension étant décrite par une succession de tables représentant les diverses granularités, chaînées par contraintes référentielles.[76]

c- Schéma en constellation : Ce modèle est un ensemble de schémas en étoiles et/ou en flocon dans lesquels les tables de faits se partagent certaines tables de dimensions. C'est de cette accumulation que découle un modèle en constellation.[76]

5.5.2 Rappel sur les concepts des Bases de Données Multidimentionnelle(BDMD)

- **Definition**

Méthode de conception logique qui vise à présenter les données sous une forme standardisée, intuitive et qui permet des accès hautement performants

Permet de considérer un sujet analysé comme point dans un espace à plusieurs dimensions :

Les données sont organisées de manière à mettre en évidence :

- Le Sujet -> Le Faits
- Les perspectives de l'analyse ->La table des dimensions

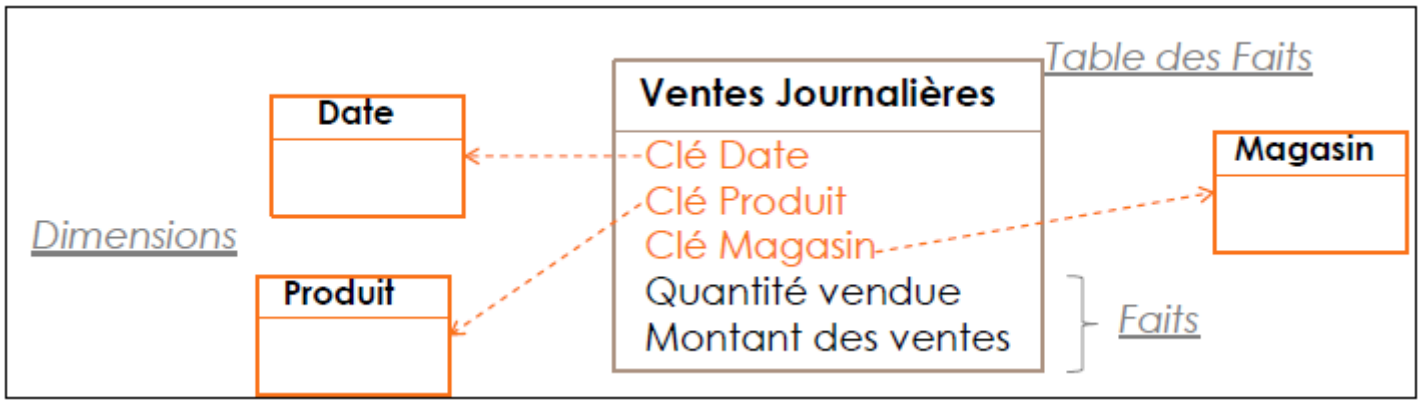


FIGURE 5.14 – Exemple d'une table de Fait

- **Table de Fait**

Une table de fait est une table qui contient les données observables (les faits) que l'on possède sur un sujet et que l'on veut étudier, selon divers axes d'analyse (les dimensions). les ' faits '

Les tables de faits représentent des associations dont l'existence d'une occurrence dépend de l'existence des occurrences correspondantes dans les tables dimensionnelles, c'est-à-dire la table de fait contient l'ensemble des mesures correspondant aux informations de l'activité à analyser. Mais rappelons que certaines tables de faits peuvent contenir aucun attribut et représentent que des liaisons entre tables dimensionnelles.

Exemple :

Fait : *Montant des ventes, chaque jour pour chaque produit dans chaque magasin*

- **Table de dimension**

Une dimension est une table qui contient les axes d'analyse (les dimensions) selon lesquels on veut étudier des données observables (les faits) qui, soumises à une analyse multidimensionnelle, donnent aux utilisateurs des renseignements nécessaires à la prise de décision.

5.5.3 Niveaux d'abstraction

Concevoir une Base de données sous hadoop avec l'outil Hive nécessite une phase de modélisation des données multidimensionnelles. Plusieurs approches ont été proposées selon deux niveaux d'abstraction :

- **Conceptuel :** Ce niveau d'abstraction consiste à représenter l'espace de données multidimensionnelles .

- **Logique :** Ce niveau d'abstraction désigne une technique de modélisation (relationnel, objet, NoSQL, etc).

● **Niveau conceptuel**

Différents concepts sont définis pour représenter les données multidimensionnelles. Les sujets d'analyse (appelés faits), regroupent un ensemble d'indicateurs . Les valeurs de ces indicateurs sont observables selon des axes d'analyse (appelés dimensions). Ces dimensions sont constituées de différents niveaux de détail, eux-mêmes organisés en hiérarchies ; par exemple, nous pourrions analyser le fait ventes au travers d'une mesure montant, ces montants pouvant être observés en fonction d'une dimension temps constituée de trois niveaux de détails (jour, mois, année) organisés au sein d'une hiérarchie définissant le jour comme un niveau de détail inférieur au mois, lui-même inférieur à l'année. Ces différents concepts permettent de concevoir des schémas multidimensionnels, appelés constellation. Les dimensions peuvent ainsi être partagées entre les faits. Un cas particulier consiste à ramener la constellation à un seul fait, on parle alors de schéma en étoile (star schema) .

● **Niveau logique**

Plusieurs modèles logiques ont été proposés pour convertir les schémas en constellation,dans notre cas nous avons utilisé l'approche R-OLAP.

● L'approche R-OLAP consiste à utiliser le modèle relationnel pour représenter un schéma en constellation Elle est de loin l'approche la plus utilisée. Ce modèle traduit chaque fait dans une table appelée table de fait. Chaque dimension est traduite dans une table appelée table de dimension. Dans la table de fait on retrouve les attributs les clés étrangères permettant la relation avec chacune des tables de dimensions. La table de dimension est constituée des paramètres et de la clé primaire .

Pour cela, nous avons conçu un modèle de données multidimensionnelles qui répond aux exigences du système environemental ,constitué de table de fait et de dimension.

Les tables de dimension :

nous avons proposé les tables de dimensions suivantes :

• **La table de dimension Riviére :** elle contient la clé primaire IDRiviere et les attributs AdresseRiviere,Nomriviere et Profondeur.

• **La table de dimension CapteurRiviére :**elle contient la clé primaire IDcapteurRiviere et l'attribut ValeurMesure.

• **La table de dimension Route :** elle contient la clé primaire IDRoute et les attributs Géolocalisation,NomRoute ,typeRoute et EtatRoute

• **La table de dimension Capteur Route :** elle contient la clé primaire IDCapteurRoute et l'attribut ValeurMesurée.

• **La table de dimension Service Urgence :** elle contient la clé primaire IDService et les attributs NomService,AdresseService et NumTel .

• **La table de dimension Habitation :** elle contient la clé primaire IDHabit et les attributs AdresseHabit,NomHabit

• **La table de dimension Unité Secours** : elle contient la clé primaire IDHopital et les attributs AdresseHopital,NomHopital et NumTel.

• **La table de dimension Structure Gouvernemental** :elle contient la clé primaire IDStructure et les attributs NomStructure,AdresseStructure et NumTel.

• **La table de dimension Zone Géographique** : elle contient la clé primaire IDZoneGéo et l'attribut NomZoneGeo.

• **La table de dimension Zone critique** : elle contient la clé primaire IDZoneCrit et les attributs NomZoneCrit,AdresseZoneCrit et NumTel.

Les tables de Fait

• **La table de fait Surveillance :**Représente la gestion du niveau d'eau de la riviére dans une zone géographique, elle contient la clé primaire IDSurveillance, les clés étrangères :IDRoute, IDRiviere,IDCapteurRiviere, IDCapteurRoute,CodeService,IDHopital,IDHabit,IDStructure,IDZoneCrit.

• **La table de fait Evacuation :** Expose l'ensemble d'informations sur la disponibilité d'une route pour évacuer vers une unité de secours. Elle contient la clé primaire IDEvacuation, et les clés étrangères CodeService,IDStructure,IDHopital,IDZoneCrit,IDHabitation,IDRoute,IDZoneCrit.

Dans notre modélisation nous avons utilisé le schéma en constellation, où les tables de faits partagent quelques tables de dimensions, tel illustré dans la **FIG 5.15**

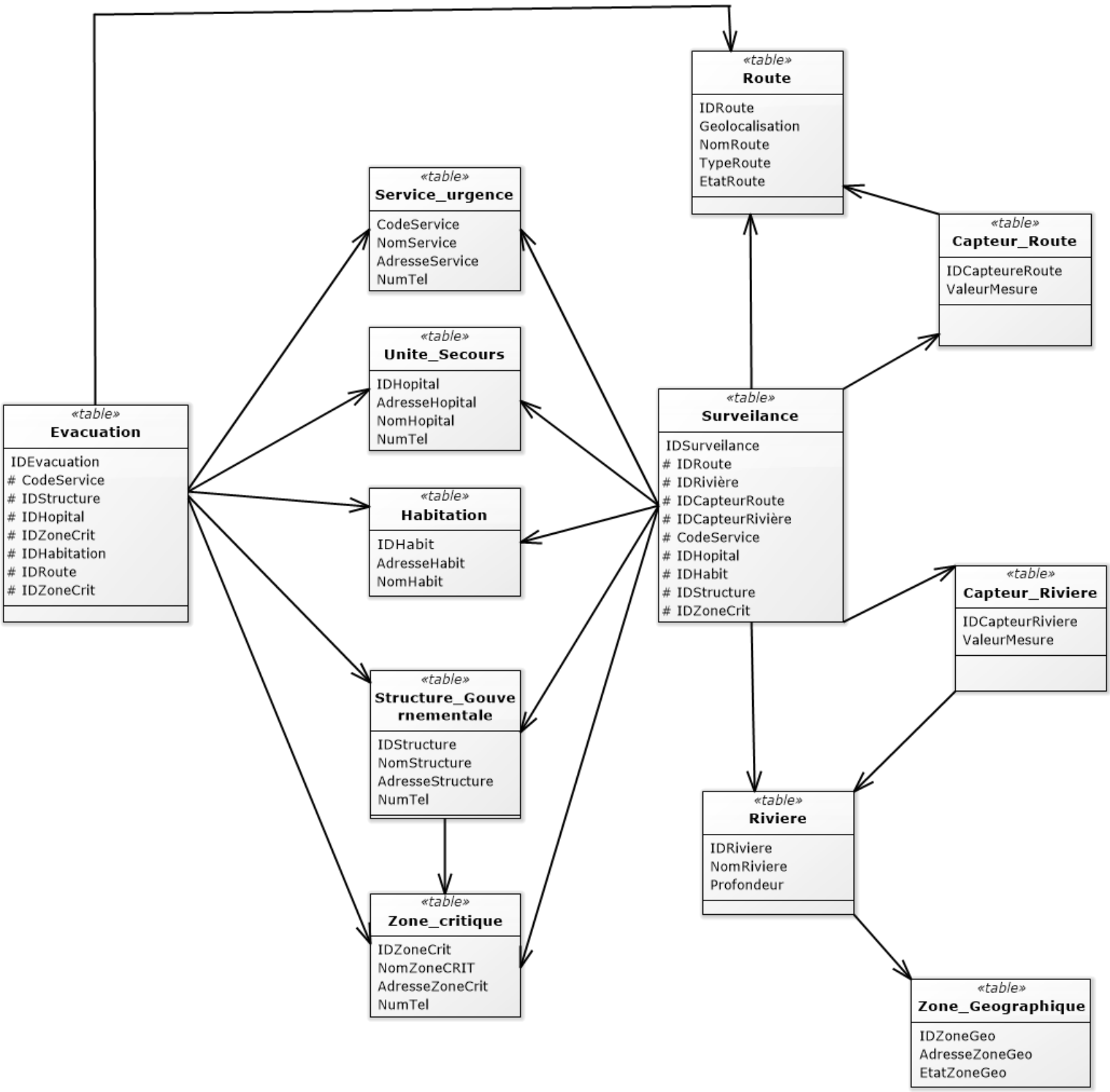


FIGURE 5.15 – Schèma de Constellation

5.5.4 Partie Traitement

On rapelle que dans le cadre de ce travail, la Base de Données NoSQL mise en place est déterminée a fournir des informations en temps réel dans le cadre d'action d'urgence **Avant** et **Après inondation,** pour ce faire comme on l'a déja cité dans la **(sous section 5.4.3),** plusieurs scénarios peuvent être imaginés.
On s'est intéressé essentiellement à deux scénarios :

- **Scénario1 :** Surveillance de la montee du niveau d'eau.
- **Scénario2 :** Evacuation de la population vers des unités de secours

Ceci nous a conduit a implémenter deux modules :

- **Module 1 :** destiné à la surveillance du niveau d'eau

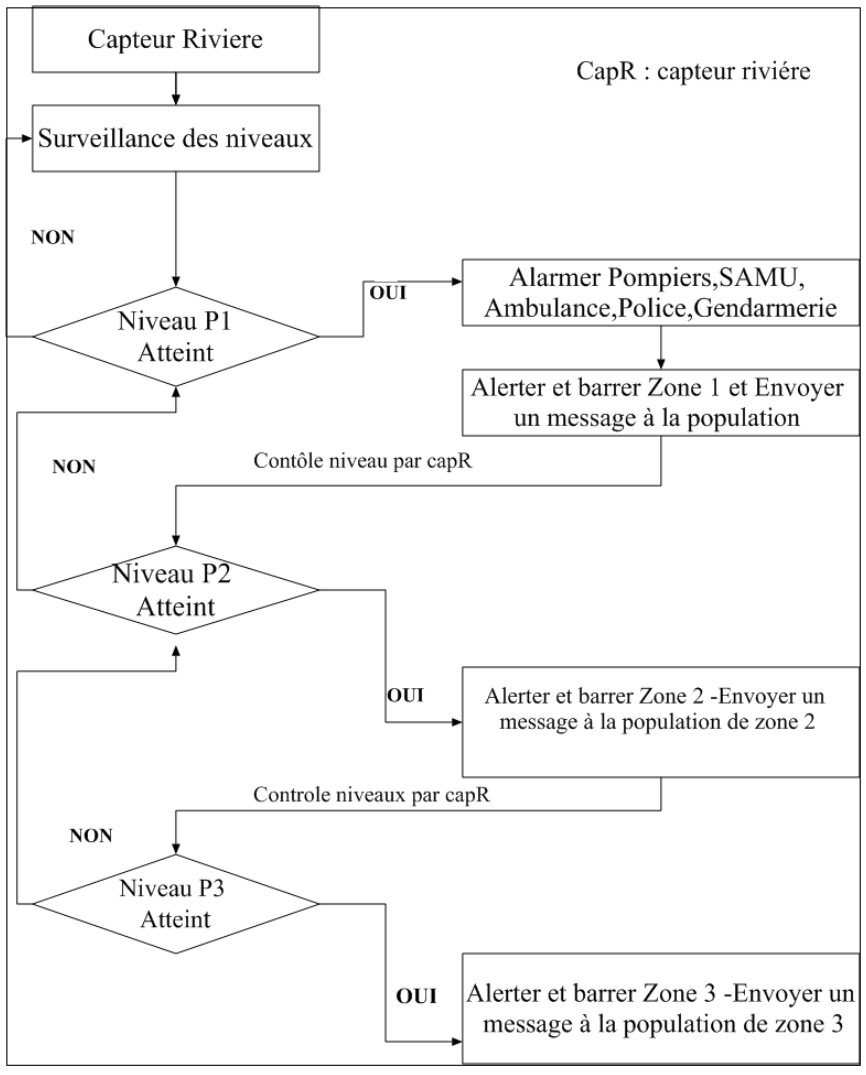


FIGURE 5.16 – Organigramme de surveillence de la monté du niveau d'eau

- **Module** 2 :destiné à l'évacuation de la population vers des unités de secours.

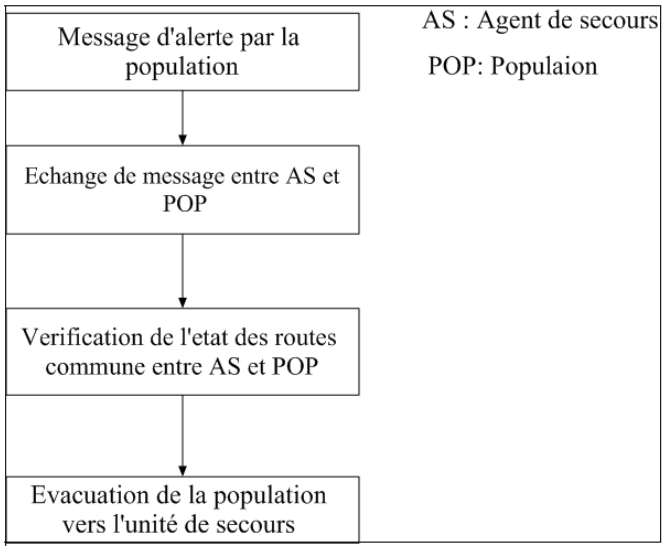


FIGURE 5.17 – Organigramme d'évacuation de la population vers des unités de secours

pour les besoins de l'application et faute de moyen matériel et impossible du bénificier du cloud, nous avons décidé de créer un simulateur pour la récolte des données des capteurs, et un organigrame pour expliquer son fonctionnement.

Nous présentons le schéma final de notre système de surveillance par couches :

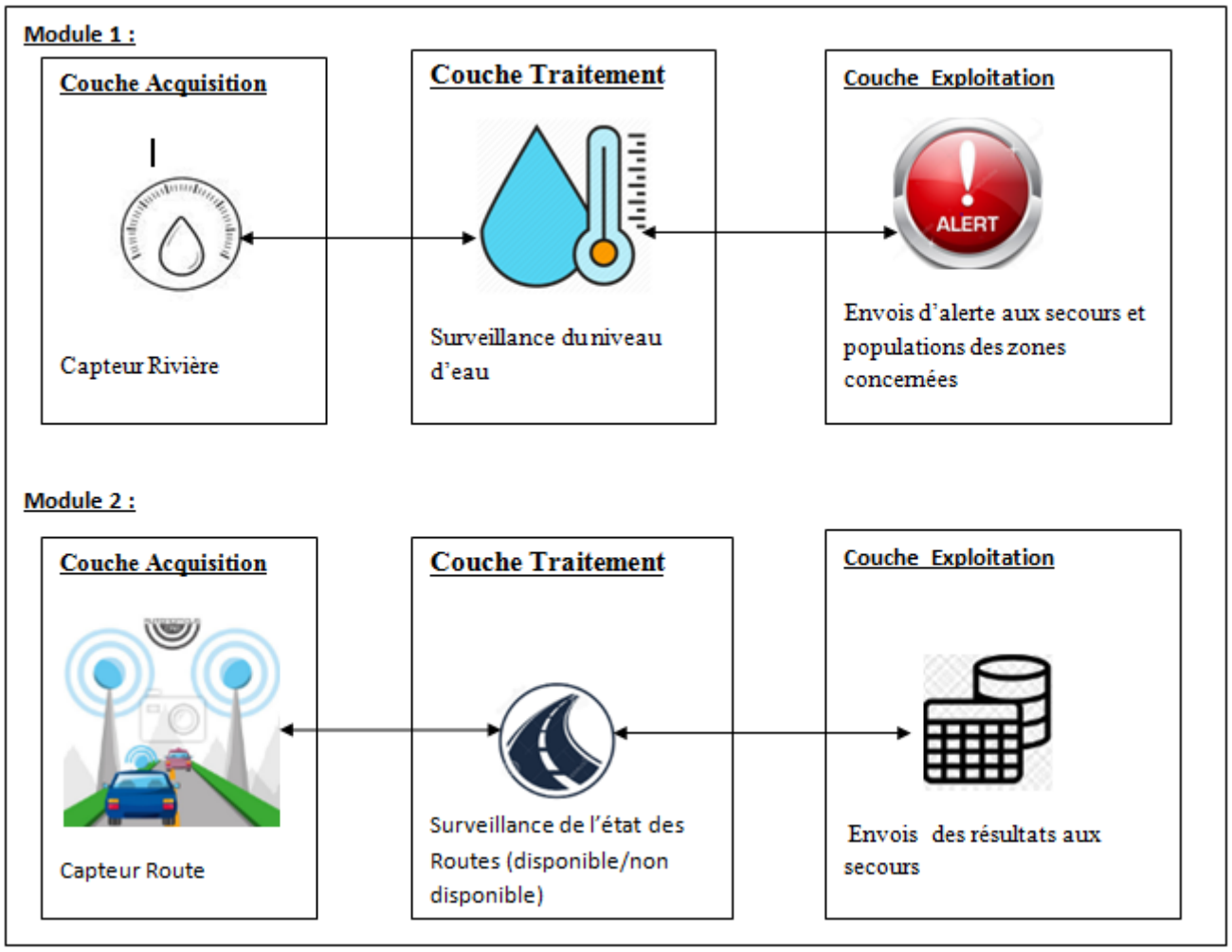


FIGURE 5.18 – Système de surveillance par couches

Conclusion

Concernant les systèmes de surveillance, nous avons décrit leurs différentes caractéristiques notamment dans le domaine environnemental. Nous avons également définit des couches de traitement dans ce type de systèmes qui sont : les couches Acquisition, Traitement et Exploitation. Ces couches nous permettent de faire un suivi des données et de découper les processus qui les ”impactent” à travers ce type de systèmes. Enfin, nous avons décrit les aspects que nous avons pris en considération dans notre étude (le cas d'inondation).

Au cours de ce chapitre, nous avons fait la conception de la Base de Données pour une répartition optimale du systéme envirenmental afin de subvenir aux exigences de ce système. A traves la démarche suivie dans la conception , nous avons pu donner une vision globale de l'ensemble d'informations composant le système de surveillance, ainsi qu'un ensemble de statistiques qui va permettre une prise de décision stratégique.

Dans le chapitre qui suit,nous allons présentés les outils implémentés pour la structuration des données dans le système de surveillance du phénomene environnemental (inondation).

Chapitre 6

Choix des outils technologiques

Introduction

Dans ce chapitre, nous allons aborder la partie pratique de notre projet qui consiste à la mise en œuvre complète de l'environnement Hadoop. Dans une distribution Hadoop on va retrouver les éléments suivants (ou leur équivalence) HDFS, MapReduce, ZooKeeper, HBase, **Hive**, HCatalog, Oozie, Pig, Sqoop, ...

les trois distributions majeures de Hadoop sont Cloudera, HortonWorks et MapR, toutes les trois se basent sur Apache Hadoop.

— MapR : noyau Hadoop mais repackagé et enrichi de solutions propriétaires.
— Cloudera : fidèle en grande partie sauf pour les outils d'administration.
— HortonWorks : fidèle à la distribution Apache et donc 100% open source.

Il existe d'autres distributions, voire des offres Cloud, mais qui n'offrent pas l'ensemble des fonctionnalités d'une plate forme Hadoop ou ne sont pas open source (non gratuites) comme Intel Distribution for Hadoop.
Dans ce projet, nous allons utiliser la distribution de Cloudera CDH5 (Cloudera Distribution of Hadoop, version 5) avec l'outil Hive qui est l'un des outils de l'écosystème d'Hadoop qui permet d'implémenter une nouvelle approche entreposage de données.

6.1 L'environnement d'implémentation

Dans cette section, on va décrire les outils et systèmes composant notre environnement d'implémentation :

1. **VM VirtualBox :**

VMware est un logiciel qui permet la création d'une ou plusieurs machines virtuelles simulant plusieurs systèmes d'exploitation x86 au sein d'un même système d'exploitation. Celles-ci, pouvant être reliées au réseau local avec des adresses IP différentes, tout en étant sur la même machine physique qui existe réellement. Il est possible de faire fonctionner plusieurs machines virtuelles en même temps, Workstation
maximise les ressources de votre ordinateur de façons à permettre l'exécution des applications plus gourmandes dans un environnement virtuel. Il permet aussi de construire des machines virtuelles complexes pour l'exécution des applications Big Data sous la forme d'un cluster Apache Hadoop

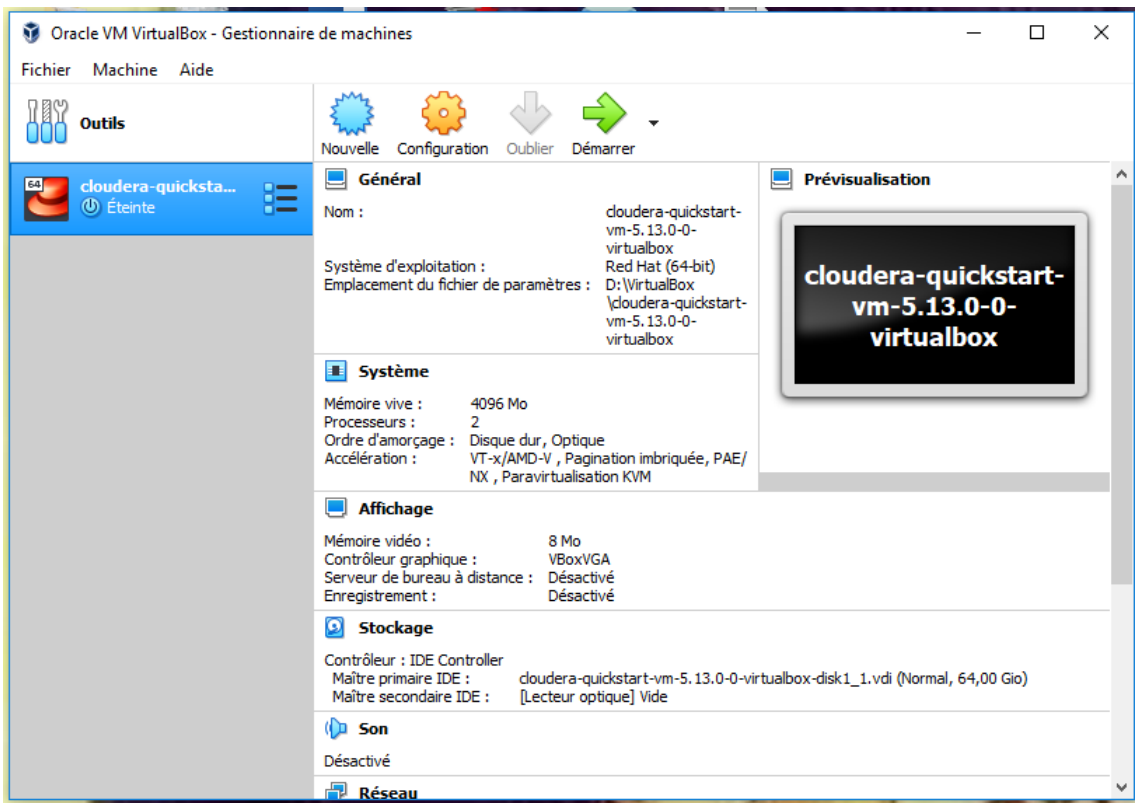


FIGURE 6.1 – Oracle VM VirtualBox 6.0

2 . Linux CentOS-6.7

CentOS (Community entreprise Operating System) est une distribution GNU/Linux principalement destinées aux serveurs. Tous ses paquets, à l'exception du logo, sont des paquets compilés à partir des sources de la distribution RHEL (Red Hat Entreprise Linux), éditée par la société Hat. On peut télécharger CentOS sous la forme DVD OU CD.

FIGURE 6.2 – Système linux CentOS 6.7

3. Cloudera CDH

CDH est la distribution open source la plus complète et la plus populaire dans le monde. Cloudera est une société de logiciels américaine cofondée en 2008 par le mathématicien Jeff Hammerbach, un ancien de Facebook. Les autres cofondateurs sont Christophe Bisciglia, ex-employé de Google, Amr Awadallah, ex-employé de Yahoo, Mike Olson, PDG de Cloudera. En mars 2009, le fonds AccelPartners a investi 5 millions de dollars dans Cloudera, suivie de Diane Greene, la cofondatrice de VMware, de MartenMickos, l'ex-PDG de MySQL, et de GideonYu, le responsable des finances de Facebook. La firme Cloudera se consacre au développement de logiciels fondés sur Apache Hadoop, permettant l'exploitation de Big Data, à savoir des bases de données accumulant plusieurs pétaoctets. CDH contient les principales composantes d'Apache Hadoop pour le stockage évolutif et des calculs distribués.

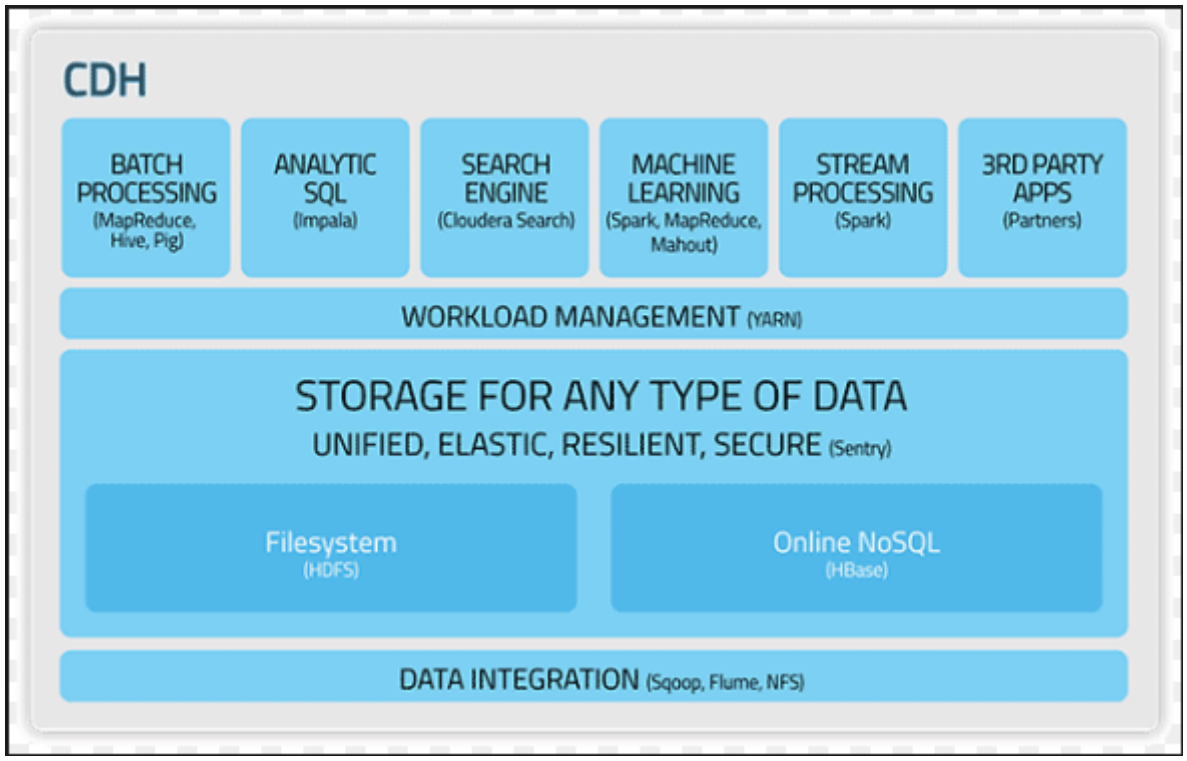


FIGURE 6.3 – Les composants de la distribution Hadoop de Cloudera

4 . Installation de Hadoop (Distribution CDH5)

— Installation de Cloudera-Quick-Starts :qui est une suite complète d'Apache Hadoop préconfiguré pour une machine virtuelle [42].

— Décompression du fichier téléchargé Cloudera-quickstart-vm-5.13.0-0-virtualbox en le sauvegardant dans un répertoire.

— En lançant VirtualBox, une nouvelle fenêtre, nommée Oracle VM VirtualBox - Gestionnaire de machines, s'affiche. Puis en cliquant sur le bouton Nouvelle en haut et à gauche de la fenêtre, nous avons saisi un nom à notre nouvelle machine dans le champ Nom (CDH5 dans notre exemple), et nous avons choisi Linux dans la première liste déroulante et Ubuntu (64 bit) ,dans la deuxième (en fait c'est la distribution CentOS 64 bits 6.7 de Linux qui sera installée). (voir Figure 6.4)

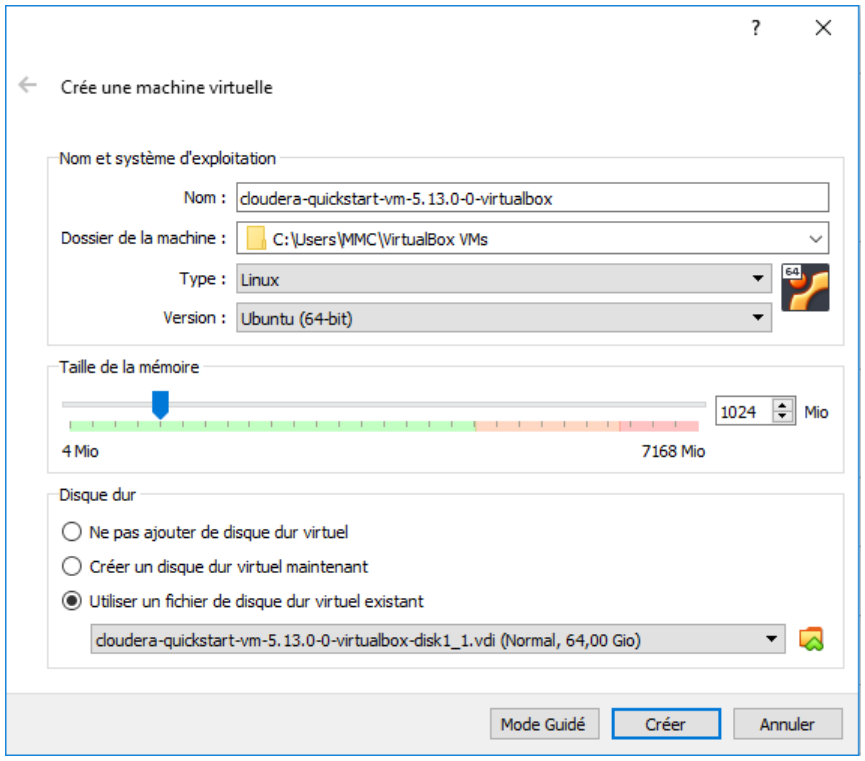


FIGURE 6.4 – le nom et le système d'exploitation de la machine crée

En cliquant ensuite sur le bouton Créer. La fenêtre principale de VirtualBox s'affichait à nouveau. Une machine virtuelle, intitulée CDH5, apparaît à gauche de la fenêtre dans la liste des machines disponibles. En sélectionnant la machine virtuelle CDH5 puis en cliquant sur la flèche verte intitulée Démarrer, et après 3 à 4 minutes environ, le bureau de CentOS , affiche "Cloudera Ask Bigger Questions". (voir Figure 6.5)

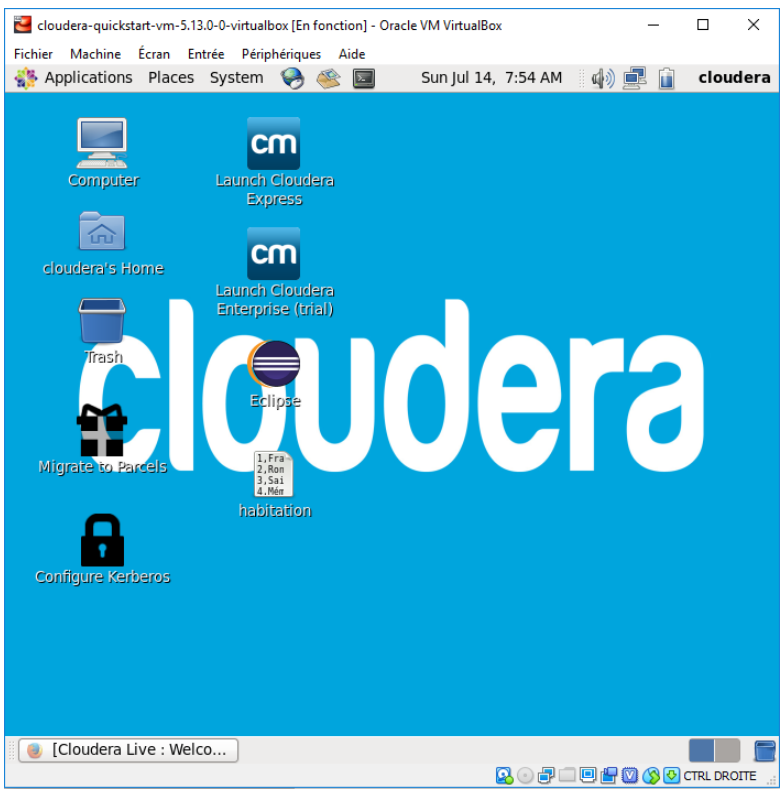


FIGURE 6.5 – le bureau de CentOS "Cloudera "

Nous allons maintenant vérifier que Hadoop est bien installé sur notre station de travail, et

que bigdata n'est pas une commande reconnue (Figure 6.6)

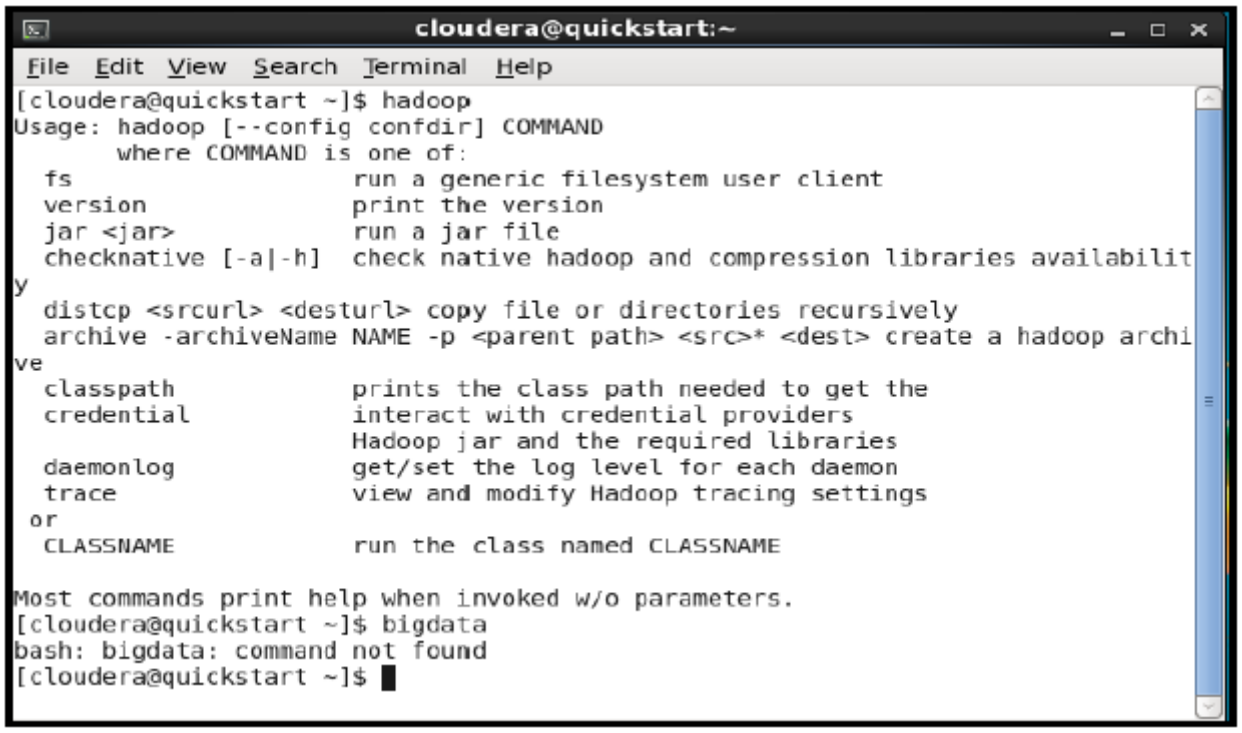


FIGURE 6.6 – Vérification de l'installation de Hadoop

Nous venons d'installer Hadoop sur notre station de travail en mode local.

6.2 Outils de réalisation du système de surveillance

— **Modèle-vue-contrôleur ou MVC**

est un motif d'architecture logicielle destiné aux interfaces graphiques lancé en 1978 et très populaire pour les applications web. Le motif est composé de trois types de modules ayant trois responsabilités différentes : les modèles, les vues et les contrôleurs.

- Un modèle (Model) contient les données à afficher.
- Une vue (View) contient la présentation de l'interface graphique.
- Un contrôleur (Controller) contient la logique concernant les actions effectuées par l'utilisateur.

— **Eclipse**

est un environnement de développement (IDE) historiquement destiné au langage Java, même si grâce à un système de plugins il peut également être utilisé avec d'autres langages de programmation, dont le C/C++ et le PHP.

Eclipse nécessite une machine virtuelle Java (JRE) pour fonctionner. Mais pour compiler du code Java, un kit de développement (JDK) est indispensable. JRE et JDK sont disponibles sur le site.

— **Java**

Java est une technique informatique développée initialement par Sun Microsystems puis acquise par Oracle suite au rachat de l'entreprise. Défini à l'origine comme un langage de programmation, Java a évolué pour devenir un ensemble cohérent d'éléments techniques et non techniques.Il est utilisé dans une grande variété de plates-formes depuis les systèmes embarqués et les téléphones mobiles, les ordinateurs individuels, les serveurs, les applications d'entreprise, les superordinateurs, etc.

— **jQuery**

jQuery est une bibliothèque JavaScript libre et multiplateforme créée pour faciliter l'écriture de scripts côté client dans le code HTML des pages web. La première version est lancée en janvier 2006 par John Resig.

Le but de la bibliothèque étant le parcours et la modification du DOM (y compris le support des sélecteurs CSS 1 à 3 et un support basique de XPath), elle contient de nombreuses fonctionnalitées ; notamment des animations, la manipulation des feuilles de style en cascade (accessibilité des classes et attributs), la gestion des évènements, etc. L'utilisation d'Ajax est facilitée et de nombreux plugins sont présents.

— **Hive**

Apache Hive est une infrastructure d'entrepôt de données intégrée sur Hadoop permettant l'analyse, le requétage via un langage proche syntaxiquement de SQL ainsi que la synthèse de données. Bien que initialement développée par Facebook, Apache Hive est maintenant utilisée et développée par d'autres sociétés comme Netflix. Amazon maintient un fork d'Apache Hive qui inclut Amazon Elastic MapReduce

dans Amazon Web Services.

Conclusion

Tout au long de ce chapitre, nous avons abordé notre environnement de travail. Par la suite, nous avons expliqué notre architecture d'application afin de présenter finalement les différentes principales parties d'implémentation de notre travail réalisé.

Chapitre 7

Implementation de la Base de Donnée et des scénarios

Introduction

Dans ce chapitre nous allons d'abord rapporter la démarche suivie dans la réalisation de notre projet ensuite nous avons détaillé le côté implémentation de notre base de données,les configurations effectuées ainsi le chargement des données dans les tables de dimension et faits, pour terminer avec une simulation des scenarios implementés dans le chapitre précedent.

7.1 Implémentation du modele multidimensionnelle sous Hive

Pour garantir une répartition optimale des moyens d'un système environemental, Il est nécessaire de mettre en oeuvre une modélisation de données qui fournit une vision globale de l'ensemble des informations concernant ce système : le flux de riviére, type de capteurs, la disponibilité des services d'urgences et de secours, etc. Pour cela, nous avons conçu un modèle de données multidimensionnelles qui répond aux exigences du système environemental d'une région . Il se compose de :

a) Les Dimensions

Les tables de dimension contiennent des descriptions textuelles sur les sujets composant le système de l'environement. Pour pouvoir rependre aux besoins du secteur environemental, nous avons proposé les tables de dimensions suivantes :

La table de dimension Riviére : elle contient la clé primaire IDRiviere et les attributs AdresseRiviere,Nomriviere et Profondeur.

FIGURE 7.1 – Fig. 4.1 Table de dimension Riviére.

La table de dimension CapteurRiviére :elle contient la clé primaire IDcapteurRiviere et l'attribut ValeurMesure.

FIGURE 7.2 – Table de dimension Capteur Riviere.

La table de dimension Route : elle contient la clé primaire IDRoute et les attributs Géolocalisation,NomRoute ,typeRoute et EtatRoute .

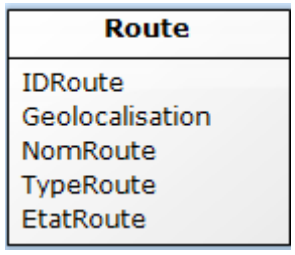


FIGURE 7.3 – Table de dimension Route

La table de dimension Capteur Route : elle contient la clé primaire IDCapteurRoute et l'attribut ValeurMesurée.

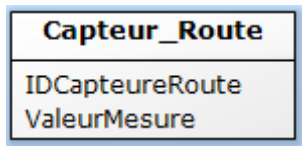


FIGURE 7.4 – Table de dimension Capteur Route.

La table de dimension Service Urgence : elle contient la clé primaire IDService et les attributs NomService,AdresseService et NumTel .

La table de dimension Habitation : elle contient la clé primaire IDHabit et les attributs AdresseHabit,NomHabit.

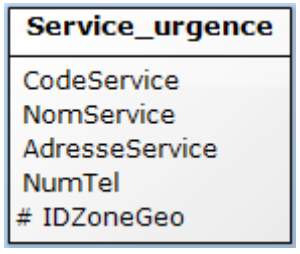


FIGURE 7.5 – Table de dimension Service Urgence

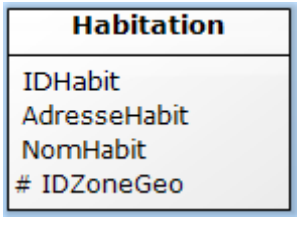


FIGURE 7.6 – Table de dimension Habitation

La table de dimension Unité Secours : elle contient la clé primaire IDHopital et les attributs AdresseHopital,NomHopital et NumTel.

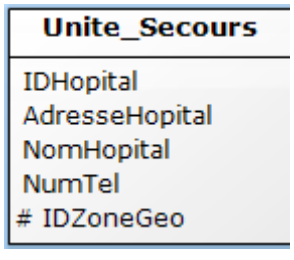


FIGURE 7.7 – Table de dimension Unité secours

La table de dimension Service Gouvernemental :elle contient la clé primaire IDStructure et les attributs NomStructure,AdresseStructure et NumTel.

FIGURE 7.8 – Table de dimension Unité secours

La table de dimension Zone critique : elle contient la clé primaire IDZoneCrit et les attributs NomZoneCrit,AdresseZoneCrit et NumTel.

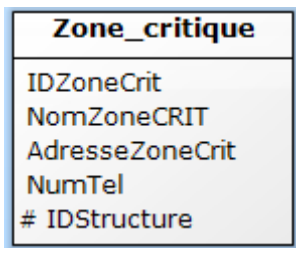


FIGURE 7.9 – Table de dimension Zone critique.

La table de dimension Zone Géographique : elle contient la clé primaire IDZoneGéo et l'attribut NomZoneGeo.

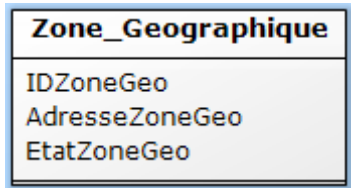


FIGURE 7.10 – Table de dimension Zone géographique.

b) Les tables de faits

Une table de fait est une table qui contient les données observables (les faits) que l'on possède sur un sujet et que l'on veut étudier, selon divers axes d'analyse (les dimensions). Notre conception contient deux tables de faits qui sont :

— **La table de fait surveillance :**Représente la gestion du niveau d'eau de la riviére dans une zone géographique, elle contient la clé primaire IDSurveillance, les clés étrangères :IDRoute, IDRiviere,IDCapteurRiviere, IDCapteurRoute,CodeService,IDHopital,IDHabit,IDS

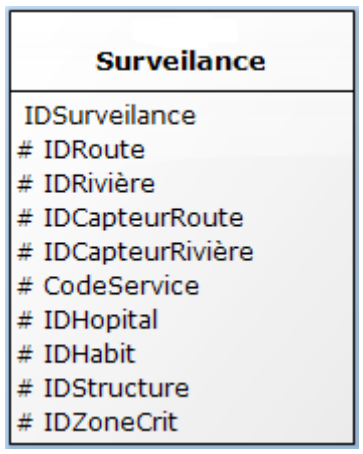


FIGURE 7.11 – Table de Fait Surveillance

— **La table de fait Evacuation :** Expose l'ensemble d'informations sur la disponibilité d'une route pour évacuer vers une unité de secours. Elle contient la clé primaire IDEvacuation, et les clés étrangères CodeService,IDStructure,IDHopital,IDZoneCrit,IDHabitation,IDRout

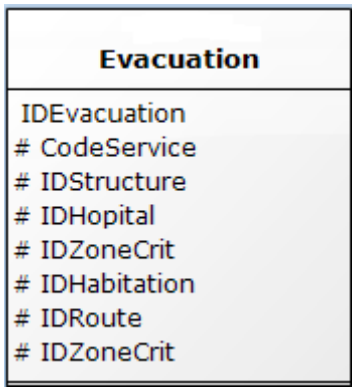
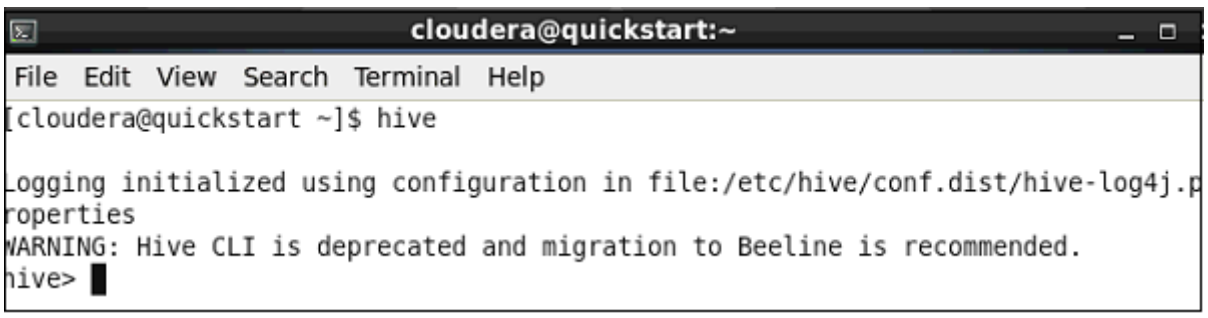


FIGURE 7.12 – Table de Fait Evacuation

7.1.1 Procédés de création de la Base de Données sous Hive

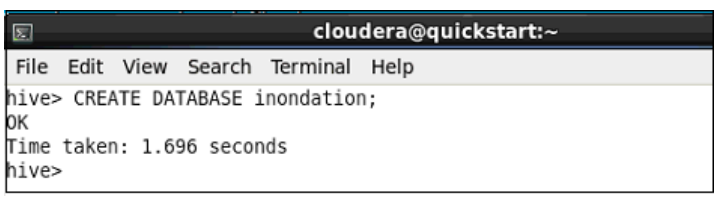
Nous avons utilisé les requêtes HiveQL pour créer les tables et les peupler avec les données du secteur envirenemental, pour cela nous avons suivi les étapes suivantes :

a- Lancement de Hive :



Création de la base de données :

Pour la création de la base de données sous Hive, nous avons utilisé les instructions suivantes : Lorsque la base de données " Inondation " est créée, Hive va créer le répertoire : /user/hive/warehouse/Inondation.db.Cet emplacement par défaut peut être remplacé par un nouveau répertoire en utilisant : **LOCATION** <nouveau répértoire>.



Ici, IF NOT EXISTS est une clause facultative, qui avertit qu'une base de données avec le même nom existe déjà. COMMENT ajoute un commentaire descriptif à la base de données, qui sera présenté par la commande : DESCRIBE DATABASE. Ce dernier affiche le commentaire

ajouté et également l'emplacement du répertoire de la base de données. À tout moment, nous pouvons voir les bases de données qui existent déjà, avec la clause :

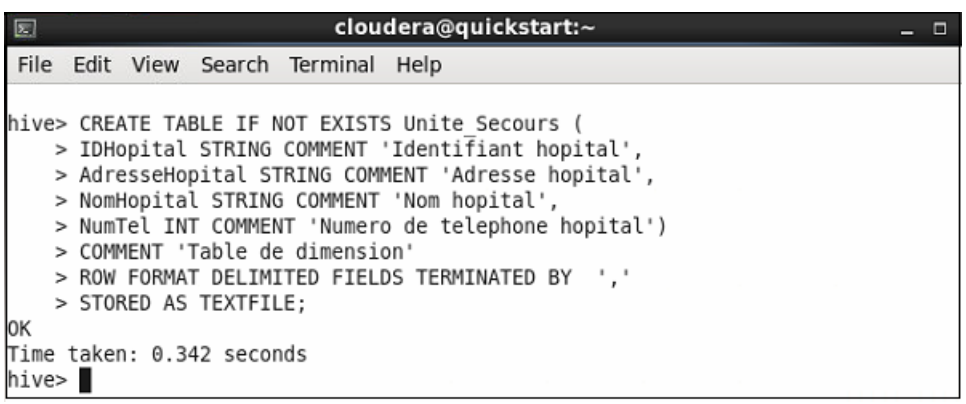
Malheureusement, il n'y a pas de commande pour nous montrer quelle base de données est la base de données en cours d'utilisation. Alors, il est toujours prudent de répéter, la clause :

Pour désigner la base " Inondation ", comme base de données de travail.

c- Création des tables de dimensions et faits

— **Les tables de dimensions**

Pour la création de la dimension 'Riviére' pour notre Base de données, nous avons utilisé les instructions suivantes :



— **Les tables de faits**

Pour la création de la table des fait 'surveillance' pour notre Base de données, nous avons

utilisé les instructions suivantes :

```
hive> CREATE TABLE IF NOT EXISTS Surveillance (
    > IDSurveillance STRING COMMENT 'Clé primaire',
    > IDRiviere STRING COMMENT 'la Clé etrangère riviere',
    > IDRoute STRING COMMENT 'la Clé etrangère route',
    > IDCapteurRoute STRING COMMENT 'la Clé etrangère capteur_route',
    > IDCapteurRiviere STRING COMMENT 'la Clé etrangère capteur_riviere',
    > CodeService STRING COMMENT 'la Clé etrangère service_urgence',
    > IDStructure STRING COMMENT 'la Clé etrangère structure_gouvernementale',
    > IDHopital STRING COMMENT 'la Clé etrangère unite_secours',
    > IDZoneCrit STRING COMMENT 'la Clé etrangère zone_critique',
    > IDHabit STRING COMMENT 'la Clé etrangère Habitation',
    > IDZoneGeo STRING COMMENT 'la Clé etrangère zone geographique',

    > COMMENT 'Table de fait'
    > ROW FORMAT DELIMITED FIELDS TERMINATED BY  ','
    > STORED AS TEXTFILE;
OK
Time taken: 0.456 seconds
```

c) **Peupler les tables de dimensions et faits**

Au cours de notre projet nous avons rencontré des difficultés en ce qui concerne la récolte de données environementale dans les riviéres de la region Aix-les-bains en france, pour remédier à cela on s'est orienté vers des données de l'environement aléatoires issues du web

— Pour peupler la table dimension 'Riviére', en charge les données à partir d'un fichier texte 'Riviere', qui contient les données de la riviére. Les données peuvent être aussi chargées à partir d'HDFS. Nous avons utilisé les instructions suivantes, pour peupler la table dimension 'riviere' :

Etape 1 : Ecriture sous hdfs.

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
[cloudera@quickstart ~]$ hadoop fs -put '/home/cloudera/Desktop/dataHadoop/Road'
 /user/cloudera
[cloudera@quickstart ~]$ hadoop fs -ls  /user/cloudera
Found 7 items
-rw-r--r--   1 cloudera cloudera        287 2019-07-17 02:32 /user/cloudera/CapteurRoute
-rw-r--r--   1 cloudera cloudera        301 2019-07-17 02:35 /user/cloudera/Critique
-rw-r--r--   1 cloudera cloudera        634 2019-07-17 02:11 /user/cloudera/Habitation
-rw-r--r--   1 cloudera cloudera        886 2019-07-17 02:48 /user/cloudera/Road
-rw-r--r--   1 cloudera cloudera        639 2019-07-17 02:07 /user/cloudera/habitation
-rw-r--r--   1 cloudera cloudera        219 2019-07-17 02:45 /user/cloudera/riviere
-rw-r--r--   1 cloudera cloudera        291 2019-07-17 02:26 /user/cloudera/sensor
```

Etape2 : chargement des données sous Hadoop avec l'outil Hive

```
hive> LOAD DATA INPATH '/user/cloudera/Road' overwrite into table route;
Loading data to table inondation.route
chgrp: changing ownership of 'hdfs://quickstart.cloudera:8020/user/hive/warehous
e/inondation.db/route/Road': User does not belong to supergroup
Table inondation.route stats: [numFiles=1, numRows=0, totalSize=886, rawDataSize
=0]
OK
```

Etape3 : verification du chargement des données

```
hive> SELECT * FROM route;
OK
---------      -------------- --------------------   ---------       --------
---
IDRoute         Geolocalisation NomRoute               TypeRoute       EtatRout
e
---------      -------------- --------------------   ---------       --------
---
01routehop      Aix les Bains   Avenue Aix les Bains    Avenue          disponib
le
02routehop      Aix les Bains   Chemin des Berthets     Chemin          indispon
ible
03routehop      Aix les Bains   Boulvard Berthollet     Boulvard        disponib
le
04routehop      Aix les Bains   Avenue Grand Port       Avenue          indispon
ible
05routehop      Aix les Bains   Chemin de saint-Pol     chemin          indispon
ible
06routehab      Aix les Bains   Avenue de petit Port    Avenue          disponib
le
07routehab      Aix les Bains   Boulvard lepic port     Avenue          indispon
ible
08routehab      Aix les-Bains   chemin observatoire     chemin          disponib
le
09routehab      Aix les-Bains   chemin observatoire     chemin          indispon
ible
```

7.1.2 Présentation de l'interface

- **Scénario1 :** Surveillance de la montee du niveau d'eau.

1. (niveau d'eau < seuil maximum de la zone 1) : aucune zone n'est alertée.

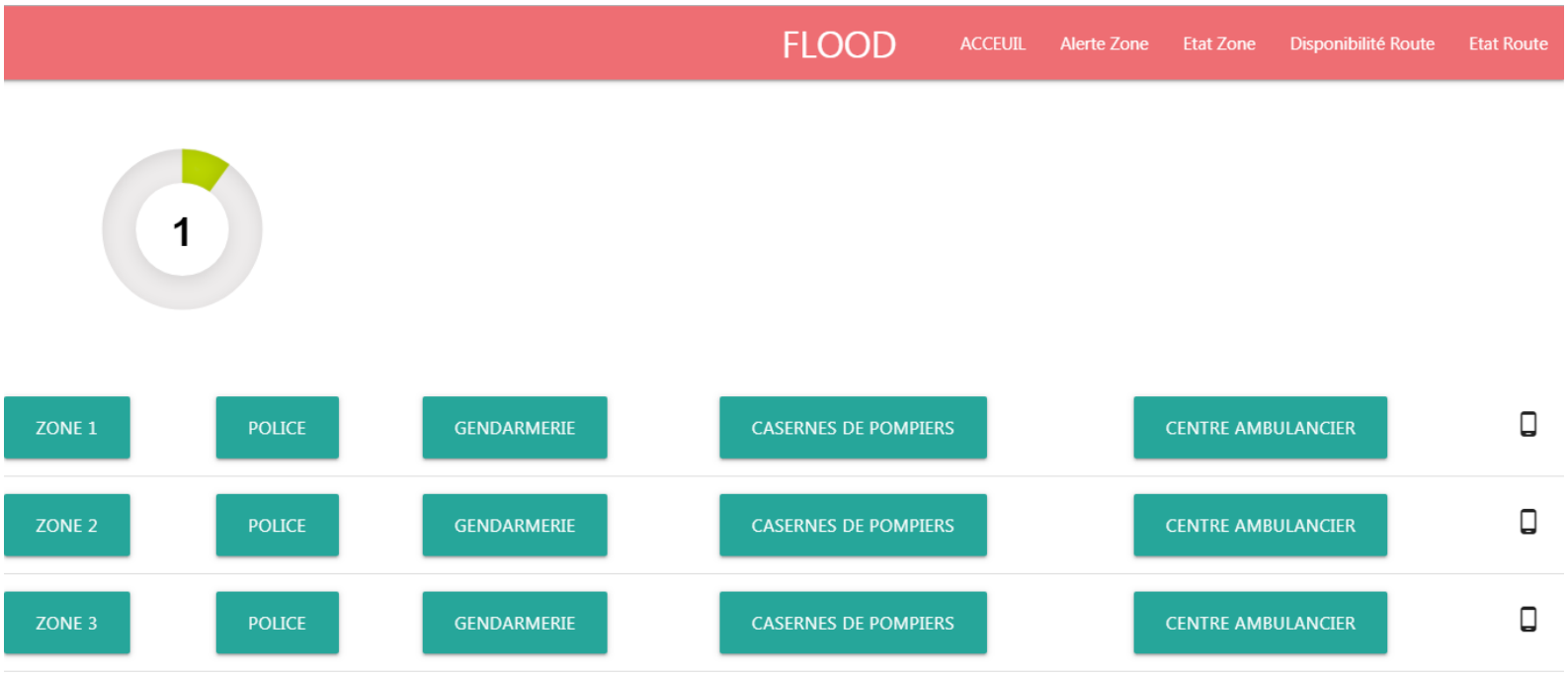
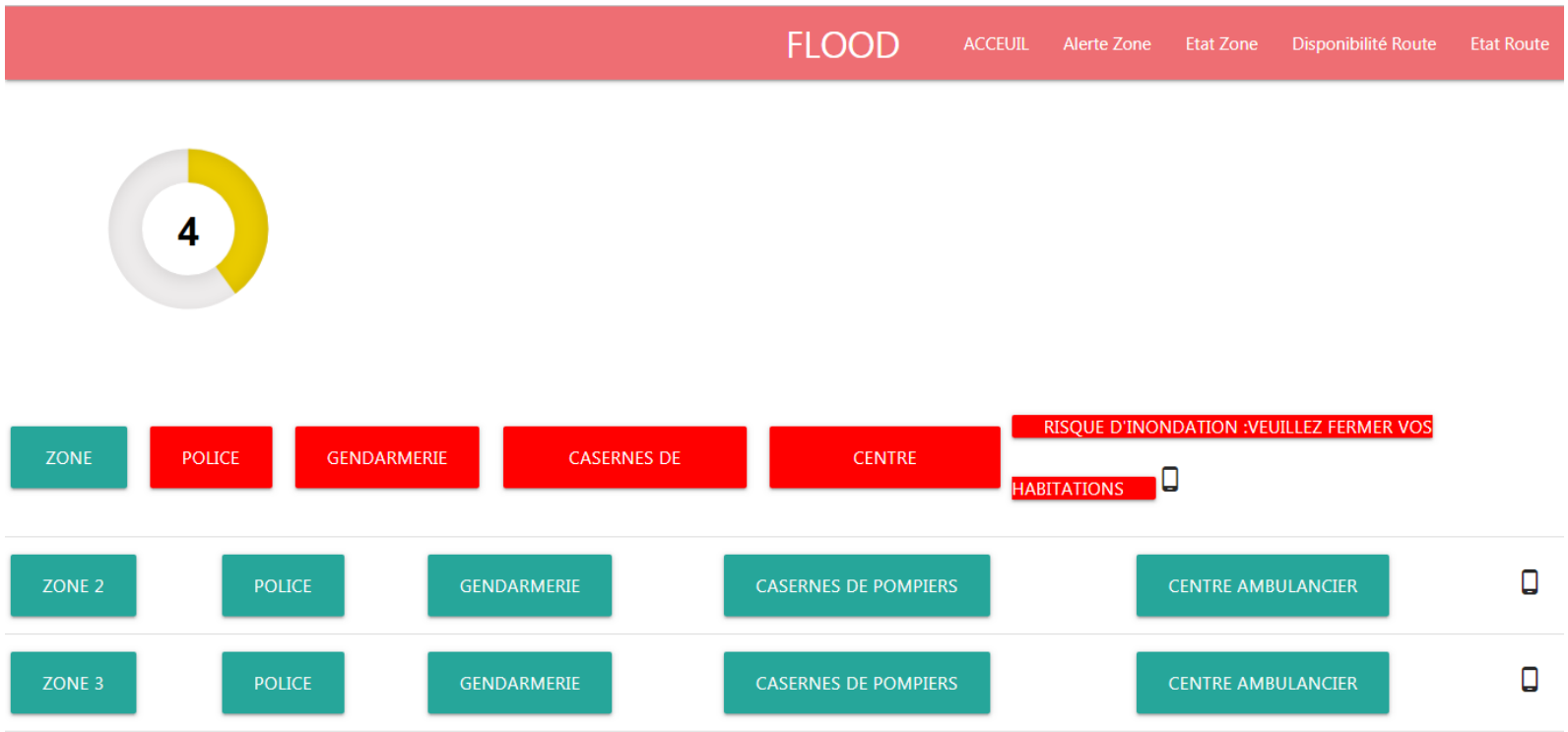
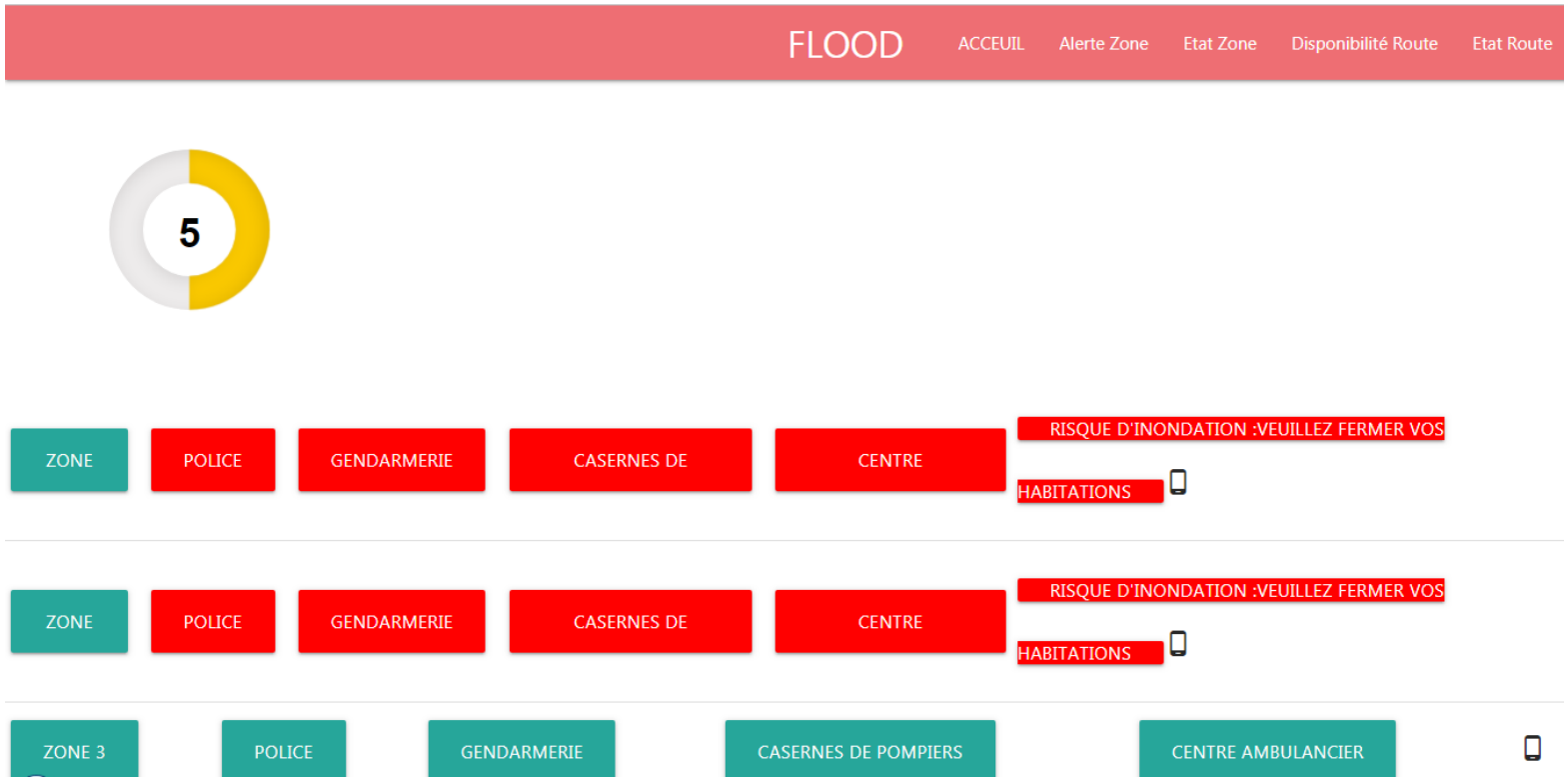


FIGURE 7.13 – Interface scénario 1

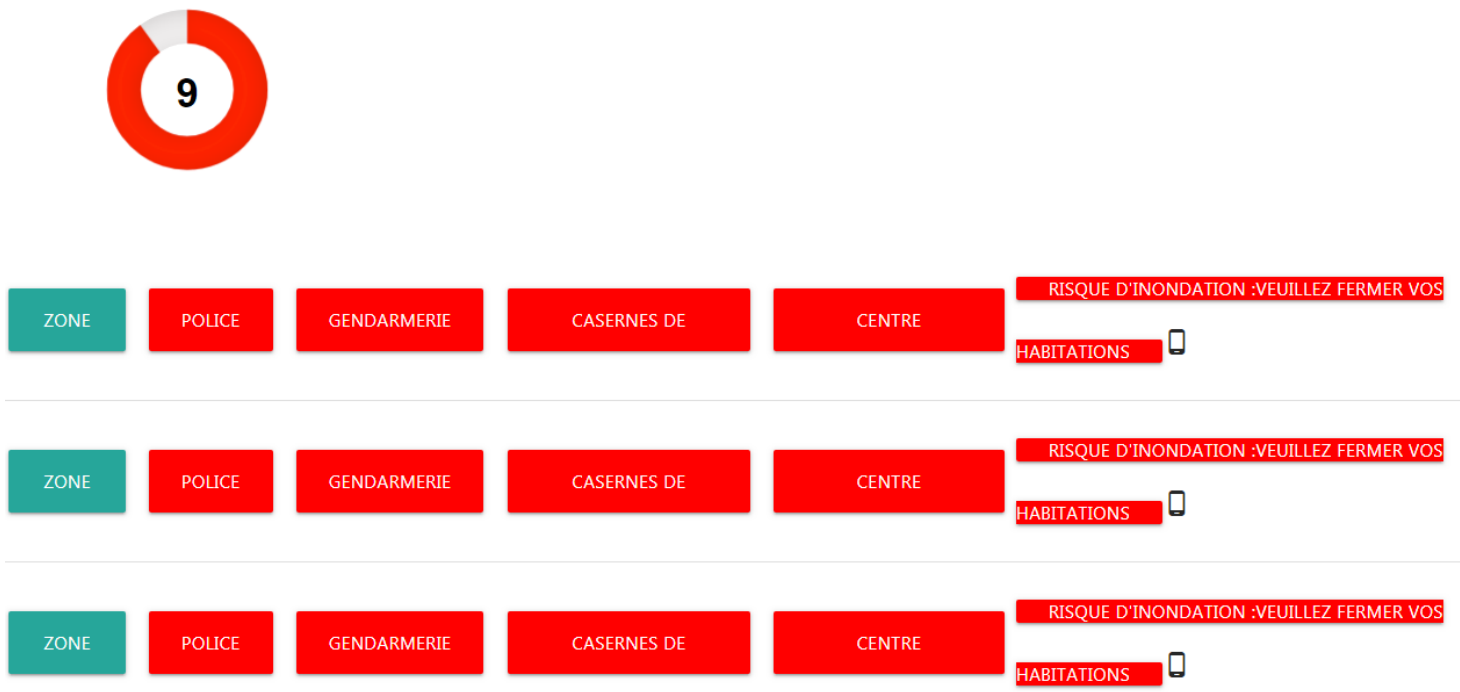
2. (niveau d'eau > seuil maximum de la zone 1) : alerte zone1 inondé.



3. (niveau d'eau > seuil maximum de la zone 2) : alerte zone1,zone 2 inondées.



4. (niveau d'eau > seuil maximum de la zone 3) : Alerte zone1,zone 2 ,zone3 inondées.



- **Scénario2 :** Evacuation de la population vers des unités de secours

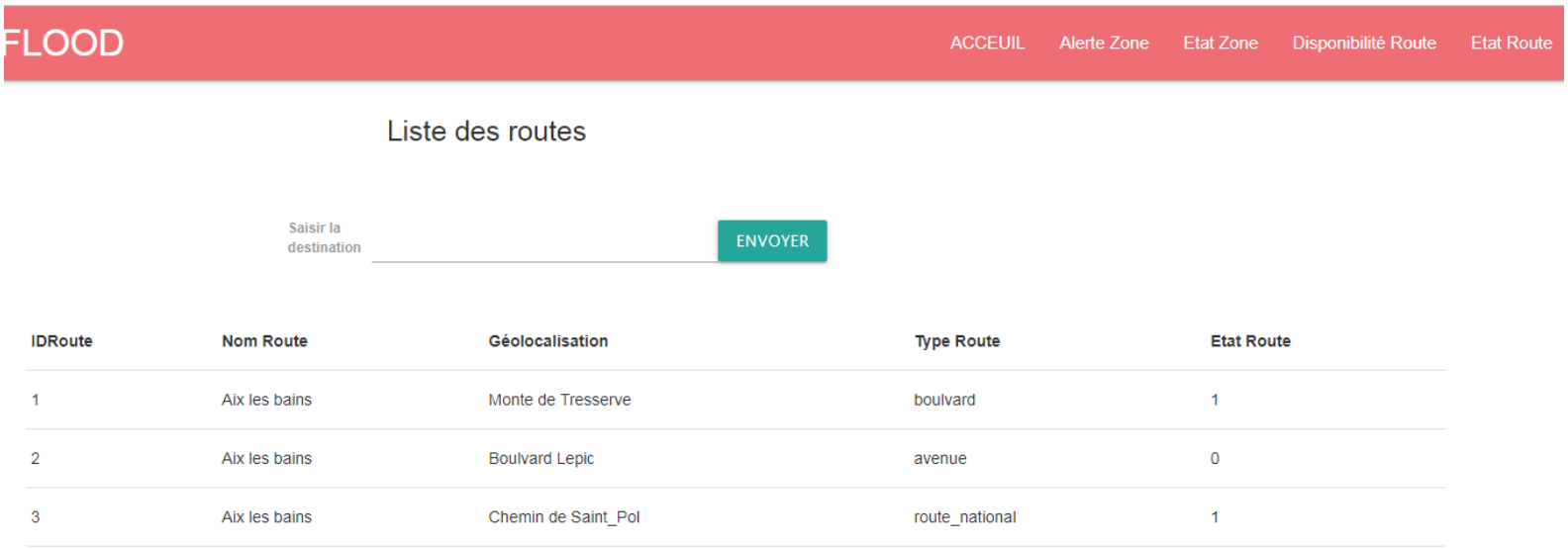


FIGURE 7.14 – Interface scenario2

Conclusion

Au cours de ce chapitre, nous avons implémenté une base de données sous hadoop avec l'outil Hive afin de subvenir aux exigences du système du surveillance. A traves la démarche suivie dans la conception et l'implémentation, nous avons pu donner une vision globale de l'ensemble d'informations composant le système de de surveillance envirenmental en implementant les differents scénarios mis en place pour la ville inteligente.

Conclusion

Pour présenter notre conclusion nous devons rappler les objectifs qui nous a été fixé.

1. Faire une étude Bibliographique approfondie des domaines de Big Data, de l'IoT et des Bases de Données NoSQL.

Ce qui nous a permis suite a des nombreuses lectures d'approfondir nos connaissances dans ces domaines, et acquerir un minimum de savoir faire dans l'étude, l'analyse, la syntése et des connaissances aquises.

2. Etudier , analyser et critiquer l'outil Hadoop.

Inspiré d'article de recherche (Annexe2) proposé par nos Encadreurs, nous avons pu analyser l'outil Hadoop et enformé une critique , ceci nous a permis de dégager la critique donnée en (Annexe 1).

3. Réflichir et imaginer la ville intelligente dans le cadre d'une inondation.

Nous ne pouvons pas tout imaginer , cela est de l'ordre de l'impossible mais il nous fallait dans un cadre pédagogique délimiter une fenêtre qui sort à notre porté .

Nous avons pu ainsi réflichir et imaginer parmi plusieurs scénarios et plusieurs données possibles :

- ***Les données qui nous semble utiles à e système***
- ***Deux scénarios à implémenter.***

4. Implémenter une base de donnée NoSQL avec l'outil Hive qui nous a été proposé à savoir Hadoop.

Pour ce faire nous avons été obligé de nous adapter à l'outil Hive-Hadoop qui implémente la base de donnée NoSQL en multidimentionnelle, nous avons donc dû revoir et reaprendre à utiliser les base de données multidimentionnelle et donc finir par implémenter notre base de donnée et l'exploiter à travers les scénarios implémentés.

Comme perspective

- Il serait vraiment interéssant de pouvoir faire une réelle mise à grande échelle de notre base de donnée ce que ne nous pouvons pas faire , faute de moyen.
- Faire une comparaison concernant le temps de réponse entre une requete SQL et une requete NoSQL.
- Pouvoir travailler sur des requêtes qui poseraient le problème sémantique perdue en passant du SQL au NoQSL.

Webographie

Big Data

[3] https ://ressources.blogdumoderateur.com/2018/06/minute-internet-2018.jpg

[10] https ://www.lebigdata.fr/infographie-quatre-v-big-data-expliques-ibm

[12] http ://www.zdnet.com/article/ten-examples-of-iot-and-big-data-working-well-together/

[15] http ://www.objetconnecte.com/4-organisations-combinaison-bigdata-iot-2306/

[18] https ://www.journaldunet.com/solutions/dsi/1194284-base-nosql-laquelle-choisir-pour-quels-besoins/

IOT

[20] https ://www.supinfo.com/articles/single/4800-presentation-internet-of-everything

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

[22] https ://www.supinfo.com/articles/single/4800-presentation-internet-of-everything

[23] https ://www.supinfo.com/articles/single/4799-piliers-ioe-objets

[24] https ://www.supinfo.com/articles/single/4799-piliers-ioe-donnees

[25] https ://tel.archives-ouvertes.fr/tel-00619303

[26] https ://www.supinfo.com/articles/single/4804-piliers-ioe-personnes

[27] https ://www.supinfo.com/articles/single/5154-piliers-ioe

[28] https ://eskimon.fr/tuto-arduino-501-généralités-sur-les-capteurs

[29] https ://www.globalsign.fr/fr/blog/qu-est-ce-qu-une-passerelle-iot/

[31] https ://blog.lesjeudis.com/10-applications-de-l-internet-des-objets-qui-revolutionnent-la-societe

[32] https ://www.inpi.fr/fr/innovation-la-galerie/talents/citizen-sciences-champion-francais-du-textile-connecte

[33] https ://villedurable.org/2014/09/10/les-smart-grids-quest-ce-que-cest/

[34] http ://www.smartgrids-cre.fr/index.php ?p=smartcities-caracteristiques

[35] http ://www.smartgrids-cre.fr/index.php ?p=smartcities-masdar

[36] https ://www.boursorama.com/actualite-economique/

[37] https ://itsocial.fr/innovation/objets-connectes/top-5-vulnerabilites-objets-connectes/

[38] https ://safenet.gemalto.fr/data-protection/securing-internet-of-things-iot/

[39] https ://safenet.gemalto.fr/data-protection/securing-internet-of-things-iot/

[40] https ://www.orange-business.com/fr/magazine/4-reflexes-a-adopter-pour-securiser-l-iot

NoSQL

[45] https ://docplayer.fr/647126-Le-nosql-cassandra.html (Xavier MALETRAS, Le NoSQL-Cassandra, Thèse Professionnelle, université Paris 13, 27/05/2012)

[46] http ://circe.univ-fcomte.fr/Marie-France-Lasalle/bda/COURSHTM/cours/chap01/lec02/pag
(Marie-France Lasalle, Cours Du Relationnel a l'objet : Limites du Relationnel)

[47] https ://www.lemagit.fr/conseil/Quand-envisager-NoSQL (Mark Whitehorn, Quand faut-il envisager d'utiliser une base de données NoSQL (plutôt qu'une base relationnelle) ? , University of Dundee.)

[49] https ://studylibfr.com/doc/2551740/l-avenir-du-nosql (Meyer Léonard, L'AVENIR DU NoSQL Quel avenir pour le NoSQL ?)

HADOOP

[53] https ://fr.wikipedia.org/wiki/Hadoop, consulté le 10/05/2019

[54] https ://mbaron.developpez.com/tutoriels/bigdata/hadoop/introduction-hdfs-map-reduce/

[55] http ://www.devx.com/opensource/exploring-the-hadoop-distributed-file-systemhdfs.html, consulté le 10/05/2019.

[59] https ://openclassrooms.com/fr/courses/4297166-realisez-des-calculs-distribues-sur-des-donnees-massives/4308656-familiarisez-vous-avec-hadoop

[60] https ://blog.ippon.fr/2013/05/14/big-data-la-jungle-des-differentes-distributions-open-source-hadoop/

[61] https ://www.supinfo.com/articles/single/8093-hdfshttps ://www.france-science.org/Hadoop-une-technologie-en-plein.html

[62] https ://www.france-science.org/Hadoop-une-technologie-en-plein.html

[63] www.virtu-desk.fr/blog/le-cloud/la-jungle-des-differentes-distributions-open-source-hadoop.html

[64] https ://fr.wikipedia.org/wiki/MapReduce

[68] https ://cwiki.apache.org/confluence/display/Hive/LanguageManual+DDL

[69] http ://bigdatawarehouse.blogspot.com/

Analyse et concption

[71] https ://www.afcdud.com/fr/smart-city/393-concept-smart-city.html

[74] https ://www.telegrafia.eu/fr/solutions/alerte-massive-aux-populations/systeme-dalerte-inondation/

[75] https ://www.20minutes.fr/toulouse/2060531-20170502-etre-averti-cas-crues-inventent-objet-connecte-taille-smartphone

Bibliographie

[1] Amrane Abdesalam, Big Data Concepts et Cas d'utilisation, Rapport, CERIST centre de recherche sur l'information scientifique et technique, 2015.

[2] Luc Bretones et Al, Big data l'accélération d'innovation, Livre blanc, l'institut G9+, 122 pages..

[4] Corp Agency, guide du Big data l'annuaire de référence à destination des utilisateurs, guide, Paris, 180 pages.

[5] Gilles Babinet et Al, Big data et objets connectés faire de la France un champion de la révolution numérique, article, Institut Montaigne, Avril 2015, 228 pages.

[6] A Zaslavsky et Al, Sensing as a Service and Big Data, article, Research School of Computer Science. Australie, 8 pages

[7] Émilie Baro, Vers une définition des Big data en santé basée sur la littérature, thèse de doctorat, Université Lille 2 droit et santé Faculté de médecine Henri Warembourg, 11 mai 2015 , 69 pages.

[8] Jean-Louis Monino, Big data open data et valorisation des données, article, Réseau de Recherche sur l'Innovation. France, 2015, 17 pages.

[9] Arthur Haimovici, EBG - L'Encyclopédie des Big Data 2016, guide,2016, France 202 pages

[11] Orange, acteur de l'Internet des objets et du Big Data, article, Novembre 2015

[13] MNN Mother Nature Network, Big Data = Big Wins for the Environment, article, UPS United Parcel Service of America, 2013, 1 page.

[14] UPS 2014 Corporate Sustainability Report, Committed to More, 29 Juin 2015, guide, UPS United Parcel Service of America, 142 pages.

[16] Jean-Louis Ermine et Al, une chaîne de valeur de la connaissance, article, Management international

[17] Bernard ESPINASSE, Introduction aux systèmes NoSQL (Not Only SQL), Ecole Polytechnique Universitaire de Marseille, 20p.

[19] D.Christin, A.Reinhardt, P.Mogre et R.Steinmetz. Wireless sensor networks and the internet of things : Selected challenges. Proceedings of the 8th GI/ITG KuVS Fachgesprach Drahtlose sensornetze ,

[30] Alicia Asín ET David Gascón, 50 Sensor Applications for a Smarter World, article, Libelium. Espagne, May 2015, 41 pages.

[41] Codd, EF (1970). A relational model of data for large shared data banks, Communications of the ACM, Volume 13,p. 377-387.

[42] Matteo Di Maglie, Adoption d'une solution NoSQL dans l'entreprise, Travail de Bachelor réalisé en vue de l'obtention du Bachelor HES, Carouge, 12 septembre 2012 Haute École de Gestion de Genève (HEG-GE).

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

[44] Kouedi Emmanuel, Approche de migration d'une base de données relationnelle vers une base de données NoSQL orientée colonne, Mémoire présenté en vue de l'obtention du diplôme de MASTER II RECHERCHE, Option : S.I & G.L ; Université de YAOUNDE I. Mai 2012.

[48] Rudi Bruchez, Les bases de données NOSQL et le BIGDATA comprendre et mettre en œuvre ,2iéme édition : ÉDITIONS EYROLLES ,

[50] Introduction au Big Data - Opportunités, stockage et analyse des mégadonnées par Bernard ESPINASSE, Patrice BELLOT 10 février 2017

[51] Adriano Girolamo PIAZZA, NoSQL Etat de l'art et benchmark ;Travail de Bachelor réalisé en vue de l'obtention du Bachelor HES ; Genève, 9 octobre 2013 Haute École de Gestion de Genève (HEG-GE).

[52] Tom White, The Definitive Guide , Edition Morgan Kaufmann,

[56] Jonathan Lejeune, Hadoop : une plate-forme d'exécution de programme Map-reduce, École des Mines de Nantes

[57] Charley CLAIRMONT, Introduction à HDFS Hadoop Distributed File System, HUG France SL2013, 20p,Mai 2013.

[58] Benjamin Renaut,Introduction à Hadoop & MapReduce ,université nice Sophia Antipolis

[65] Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Ning Zhang, Suresh Antony, Hao Liu et Raghotham Murthy, Article Hive-A Petabyte Scale Data Warehouse Using Hadoop, 2010.

[66] Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Ning Zhang, Suresh Antony, Hao Liu et Raghotham Murthy, Article Hive-A Petabyte Scale Data Warehouse Using Hadoop, 2010.

[67] Edward Capriolo, Dean Wampler, and Jason Rutherglen, Programmin Hive, first edition d'Oreilly,octobre 2012.

[70] Les Villes Intelligentes à Travers le Monde,Université Laval

[72] La ville inteligente comme vecteur pour le développement durable,Essai présenté au Centre universitaire de formation en environnement et en développement durable en vue de l'obtention du grade de maître en environnement ,Par Joëlle Simard

[73] Thèse Qualité des données capteurs pour les systèmes de surveillance de phénomènes environnementaux,Claudia Catalina GUTIERREZ RODRIGUEZ

[76] Sandro Bimonte, Thèse Intégration de l'information géographique dans les entrepôts de données et l'analyse en ligne : de la modélisation à la visualisation, Institut National des Sciences Appliquées de Lyon.