Ministère de l'Enseignement Supérieur de Recherche Scientifique

Université KASDI Merbah Ouargla

Faculté des Nouvelles Technologies de l'Information et la Communication

Département de l'Informatique et Technologies de l'Information

Mémoire de Master Académique

Domaine : Mathématique et Informatique

Filière : Informatique

Spécialité : Informatique Fondamentale

Présenté par : ZEMMOURI Mohamed

Thème :

Plateforme BI basée MapReduce pour Big Data Management

Application au sein de l'ENSP

Soutenu publiquement le : 19/09/2019
Devant le jury :

M	KHALDI Amine	Président
M	BACHIR Mahjoub	Examinateur
Mlle	TOUMI Chahrazad	Examinateur & Encadreur

Année Universitaire : 2018/2019

Remerciements

En préambule à ce mémoire, je souhaite adresser mes remerciements les plus sincères aux personnes qui m'ont apporté leur aide et qui ont contribué à l'élaboration de ce travail.

Je commencerais par mes collègues Nadjib MESBAHI et Noureddine BOUKHECHEM.

Je remercie également mes responsables dans l'entreprise qui ont été compréhensif et indulgent.

Je tiens à remercier sincèrement mon encadreuse, Melle Chahrazad TOUMI, ainsi que tous mes enseignants de l'Université Kasdi Merbah Ouargla.

J'exprime ma gratitude à Abderrahmene BELHADI qui m'a beaucoup aidé dans ce travail.

Dédicaces

Je dédie ce modeste travail :

A mon père et ma mère,

A mes enfants Wadoud et Randa,

A tous mes frères et mes amis,

ملخص

تظل البيانات الضخمة والتقنيات المرادفة لها كإنترنت الأشياء والحوسبة السحابية أمرًا لا مفر منه، نظرًا للخدمات التي تقدمها. ومع ذلك، فإن هاته التقنيات الجديدة، وخاصة البيانات الضخمة، تجعل طرق معالجة وتحليل البيانات التقليدية غير مجدية، وتفرض أساليب مبتكرة تستجيب للتحديات الجديدة.

من هاته الأساليب نستلهم حلا يوفر تحليل البيانات من مصادر مختلفة في شكل لوحة قيادة. هذا الحل مقسم إلى أجزاء يمكن إعادة استخدامها، ومستقل عن أي نظام معلومات أو مصدر بيانات.

الكلمات الدالة: البيانات الضخمة، لوحة القيادة، تحليل البيانات، علوم البيانات، MVC.

Résumé

Le Big Data, ainsi que les autres technologies connexes notamment, IoT et le Cloud Computing, demeurent incontournables vu les services qu'ils offrent. Néanmoins, ces nouvelles technologies, particulièrement le Big Data, rendent les méthodes de traitement et d'analyse des données classiques obsolètes, et imposent de nouvelles pratiques qui répondent aux nouveaux défis posés.

En s'inspirant du Big Data, nous proposons un système qui fournit l'analyse de données de différentes sources sous forme d'un tableau de bord. Ce système est modulaire, réutilisable, et indépendant de tout système d'information ou source de données.

Mot clés : Big Data, Tableau de bord, Analyse de données, Science de données, MVC, Programmation Modulaire.

Abstract

Big data, as well as other adjacent technologies such as IoT and Cloud Computing, became unavoidable because of the services they offer. Nevertheless, these new technologies, especially Big Data, make conventional data processing and analysis methods outdated, and impose new practices that respond nowadays challenges.

Inspiring from Big Data, we suggest a system as a dashboard that provides data analysis from different sources. This system is modular, reusable, and independent from any information system or data source.

Keyword (s): Big Data, Dashboard, Data Analysis, Data Science, MVC, Modular Programming

Table des matières

Liste des tableaux

Liste des figures

Introduction

Depuis la démocratisation d'Internet et l'apparition des objets connectés, le volume des données générées quotidiennement explose. Les données générées par l'homme et par les machines connaissent un taux de croissance 10 fois supérieur à celui des données professionnelles [1]. Cet environnement favorise le développement du Big Data et d'auteurs technologies adjacents notamment l'IOT (Internet Of Things) et le Cloud Computing.

Le terme Big Data peut se définir comme tout ensemble de données caractérisé par un Volume massif, une grande Vélocité et une grande Variété qui nécessite une analyse avec des méthodes innovantes [2, 3]. Cette nouvelle technologie présente de nouveaux défis de stockage, manipulation et traitement des données de masse. Afin de répondre à ces défis, plusieurs plateformes sont développés autours de cette technologie notamment Apache Hadoop et Apache Spark, … etc.

Au sein de l'Entreprise National de Service aux Puits (ENSP) on dispose d'un volume de données important avec une certaine variété et hétérogénéité, ce qui représente une contrainte pour une analyse complète inter systèmes d'information.

Notre objectif est de fournir un système d'analyse sous forme de tableau de bord, tout en s'inspirant des méthodes et pratiques Big Data sans les utiliser directement. Ce système doit permettre aux utilisateurs d'effectuer une analyse étendue sur plusieurs sources de données, de caractère dissimilaire, et ce d'une manière transparente.

Afin d'atteindre notre objectif, nous proposons un système sous forme d'une couche décisionnelle au-dessus de nos sources de données, en by passant la logique de leurs systèmes d'information car il s'agit seulement de lecture de données. Nous avons opté pour une architecture modulaire, basée sur un découpage logique : sécurité, data et noyau du système.

Ce mémoire est structuré comme suit :

- **Le premier chapitre :** Consacré à introduire le concept du Big Data avec ses définitions, ses caractéristiques, et ses domaines d'utilisation.

- **Le deuxième chapitre :** Présentera les technologies et les solutions utilisées dans un traitement Big Data. Nous donnerons également dans ce chapitre une brève description des tableaux de bord.

- **Le troisième chapitre :** Exposera la problématique rencontrée et présentera notre proposition, son architecture et ses différents modules.

- **Le quatrième chapitre :** Dans ce dernier chapitre, nous entamerons l'implémentation de notre système. Nous citerons dans ce sillage les déférentes solutions utilisées et la plateforme de déploiement, avec une présentation de l'interface de travail réalisée. Nous finirons ce chapitre en partageant l'expertise que nous avons acquise durant ce projet.

Enfin, nous terminerons par une conclusion générale qui résume le résultat de notre travail et liste les perspectives futures de ce travail.

PREMIÈRE PARTIE :

Etat de l’art :

Big Data, fondements et outils

Chapitre 1

Big Data

1.1 Introduction

Dans ce premier chapitre nous passerons en revue les notions générales du Big Data, comme sa définition, sa genèse et ses caractéristiques.

Nous aborderons également plus en détail les facteurs dont l'apparition a favorisé le développement du Big Data. Les plus importants à notre sens sont IOT (Internet Of Things) et Cloud Computing.

Enfin, nous présenterons les domaines d'application ainsi que le processus de traitement du Big Data.

1.2 L'apparition du Big Data

Depuis l'apparition de l'informatique, jusqu'à l'omniprésence du web actuel dans la vie de tous les jours, les données ont été produites en quantités toujours croissantes. Textes, logs, photos, sons, vidéos en tout genre... Depuis la démocratisation d'Internet, ce sont des volumes impressionnants de données qui sont créés quotidiennement par les particuliers, les entreprises et maintenant aussi les objets connectés [4].

De nombreuses études prédisent une croissance exponentielle des données à l'horizon 2020 et au-delà. Cependant, elles s'accordent toutes sur le fait que la taille de l'univers numérique doublera au moins tous les deux ans, soit une multiplication de 50 entre 2010 et 2020. Les données générées par l'homme et par les machines connaissent un taux de croissance 10 fois supérieur à celui des données professionnelles traditionnelles, tandis que seuls les données captées ont un taux de croissance de l'ordre de 50x [1].

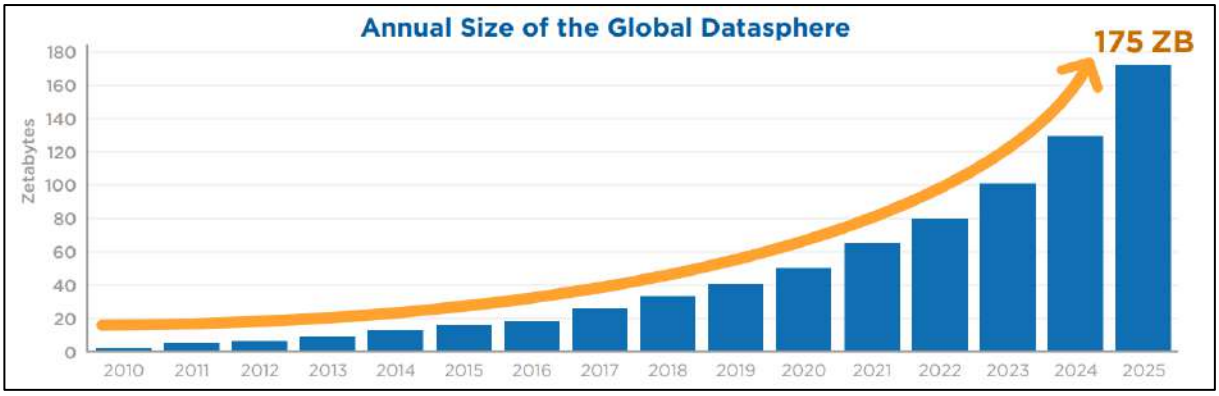

Figure 1-1 : Evolution annuelle des données universelles [5, 6]

Désormais, le terme « Big Data », littéralement traduit par « grosses données » ou « données massives » désigne cette explosion de données. On parle également de « data masse » en analogie avec la biomasse, écosystème complexe et de large échelle [4].

1.3 Définition

Bien que l'émergence du Big Data - Comme nous avons vu - soit due essentiellement à la croissance exponentielle des données avec la propagation de l'utilisation des outils informatique et l'apparition de nouvelles technologies - notamment *l'internet of things* ou les objets connectés- ce concept qui s'étant popularisé davantage dès 2012, mais l'expression « Big Data » date de 1997 selon *l'Association for Computing Machinery*.

Cependant en 2001, l'analyste du cabinet Meta Group (devenu *Gartner*)[1] Doug Laney décrivait le principe des « 3 V » (dans le contexte e-Commerce, sans citer le mot Big Data) [3], qui stipule que le Big Data est tout ensemble de données caractérisé par un Volume massif, grande Vélocité et / ou grande Variété qui nécessite une analyse Avec des méthodes innovantes et rentables, permettant d'avoir une meilleure vision afin de prendre des décisions opportunes et à temps. [3, 4]:

On peut trouver également des définitions distinctes (comme celles d'Oracle, Nist et IBM, Voir les déférentes définitions dans l'annexe A) qui ajoutent d'autres « V », notamment

[1] Gartner Inc. est une entreprise américaine de conseil et de recherche dans le domaine des techniques avancées dont le siège social est situé à Stamford dans le Connecticut, site Web : *https://www.gartner.com.*

la véracité qui évoque la nécessité de vérification, et la Valeur, tous ces caractéristiques seront évoquées ci-après.

1.4 Les Dimensions

Comme indique sa définition, le Big Data est caractérisé par le volume, la vélocité et la variété d'informations. Ces termes sont des caractéristiques souvent appelées des dimensions. Toutes les dimensions sont des cibles en perpétuel mouvement. Par conséquent, le Big Data peut croître dans n'importe quelle dimension, voire même toutes les dimensions [2].

En outre, et afin de mieux cerner les caractéristiques du Big Data, d'autres recherches ont étendu l'ensemble des dimensions selon le domaine d'application afin de mieux répondre aux attentes en matière de qualité, parmi ces recherches on cite celle d'IBM qui ajoute la Véracité.

Il existe diverses interprétations des dimensions Big Data (voir la figure ci-dessous), dans cette section on va cibler les caractéristiques les plus pertinentes selon les différentes études, autrement dit les 5V.

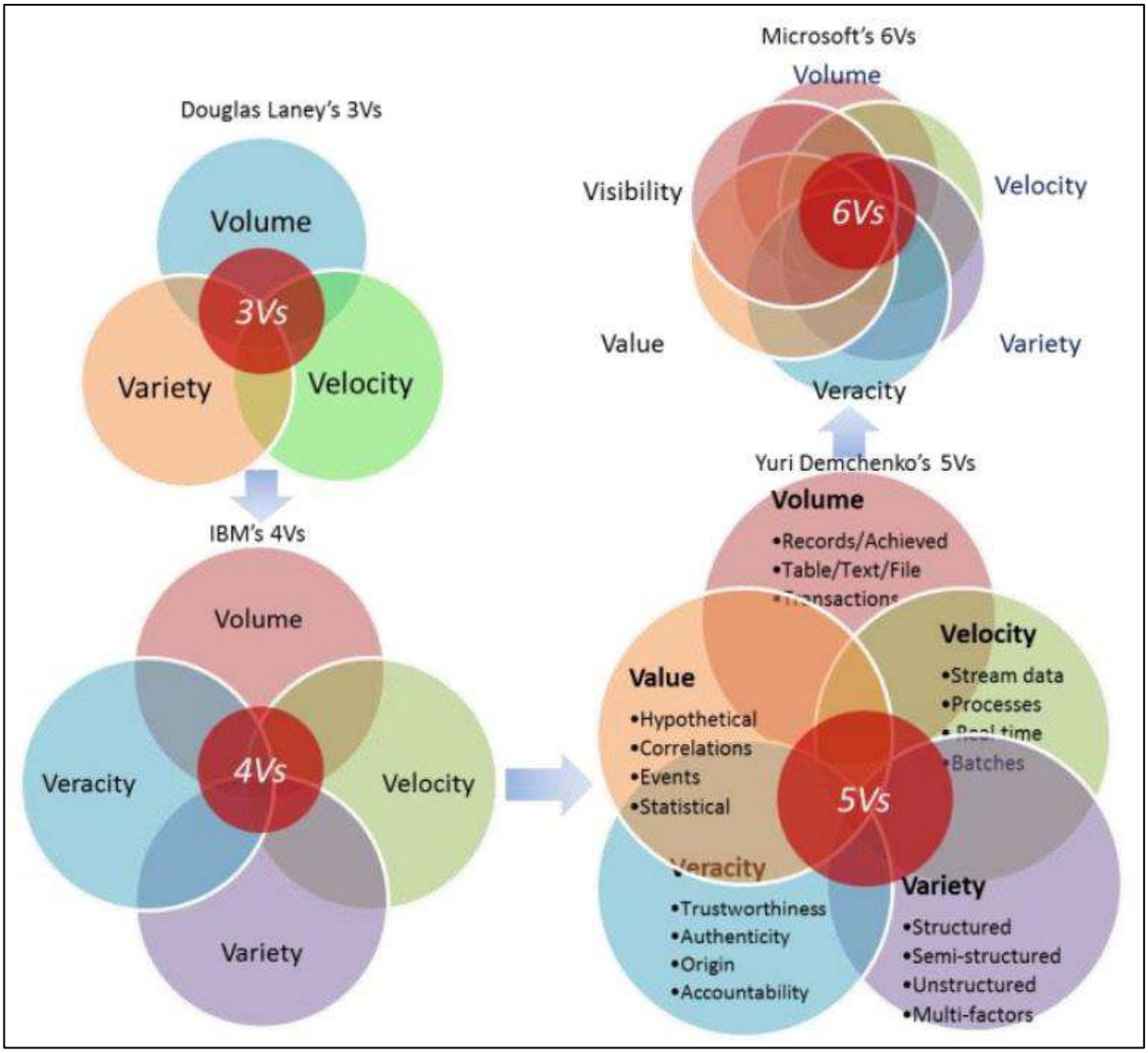

Figure 1-2 : Diverses interprétations des dimensions Big Data [7]

1.4.1 Volume

La dimension de volume représente le volume physique des données. Le Big Data est associé à un volume de données vertigineux, se situant actuellement entre quelques dizaines de téraoctets (1 To=2^{12} octets) et plusieurs pétaoctets (1 Po=2^{15} octets) en un seul jeu de données [3]. Le volume gère aussi la masse d'informations produite chaque seconde.

Les entreprises issues de tous les secteurs d'activité gérant des données massives, se voient assujetties à trouver des techniques nécessaires et moyens capables pour gérer les volumes de données vitales collectés [3].

Comme la croissance correspondante des données est plus rapide que la croissance de la capacité de stockage, et pour faire face à ce paradoxe les chercheurs ont fait appel à des solutions comme les systèmes de fichiers distribué et parallèle.

Parmi les technologies aujourd'hui couramment utilisées dans le traitement de volume de données se trouve le framework Hadoop et l'algorithme MapReduce, qu'on va voir en détail dans chapitre suivant.

1.4.2 Vélocité

La vélocité ou la vitesse d'échanges décrit la fréquence à laquelle les informations sont générées, capturées, stockées et partagées. La vélocité a connu une évolution similaire à celle du volume au sein des entreprises [3].

La vélocité couvre aussi le traitement des flux continus de données. Les entreprises doivent appréhender la vitesse non seulement en termes de création de données, mais aussi sur le plan de leur traitement, de leur analyse et de leur restitution à l'utilisateur en respectant les exigences des applications en temps réel [3].

Le paramètre vélocité est déterminant dans les situations où il s'agit de tenir compte les requêtes en temps réel (comme domaine militaire), ou pour tirer un avantage concurrentiel fondamental qui permet chaque jour de prendre une longueur d'avance dans la décision pour certains métiers comme celui de la finance [4].

1.4.3 Variété

La dimension de variété représente l'expansion de l'information résultant en de multiples types de données (textuelles, numériques, etc.), formats, structures (structurées, semi-structurées, non structurées), codage, syntaxe, sémantique, etc. Près de 85% des données d'une organisation ne sont pas structurées, mais elles doivent tout de même être intégrées à une analyse quantitative et à un processus décisionnel. Le texte, la vidéo, l'audio et d'autres données non structurées nécessitent une architecture et des technologies d'analyse différentes [2].

L'une des grandes ambitions du Big Data est de proposer le recoupement d'informations ou la mise en correspondance de données d'origines très diverses. Ce serait même l'une des

principales sources de valeur. Cependant, et à l'heure actuelle il n'existe aucune méthode universelle pour gérer la variété des données, le traitement et le recoupement de données de sources et de formats variés d'une situation à une autre [4].

Pour faire face à cette diversité de structures, certains systèmes NoSQL utilisent des schémas de données qui s'écartent du modèle relationnel classique. Les bases orientées graphes ou les bases orientées colonnes sont des exemples [4].

1.4.4 Véracité

Vu le volume, l'hétérogénéité et la vélocité des données dans le « Big Data », l'authenticité de ces dernières reste un obstacle. Les données ne sont pas forcément correctes et ils sont rapidement obsolètes. Ainsi, dans le cadre d'un sondage réalisé par IBM [8], 27% des entreprises interrogées avouent ne pas être certaines de l'exactitude des données qu'elles collectent.

Donc il est difficile de justifier l'authenticité et l'exactitude des contenus des différents volumes et variétés de données manipulées comme dans les conversations dans les réseaux sociaux avec les abréviations, le langage familier, les coquilles, les hashtags. D'où l'intérêt d'avoir des outils de vérification de ces données [3].

La Véracité fait référence à l'aptitude à juger la crédibilité et la fiabilité du nombre indéfini de données collectées.

1.4.5 Valeur

La dimension de valeur peut être considérée comme le fruit car toutes les technologies de stockage et d'analyse des Big Data n'ont de sens que si elles apportent de la valeur ajoutée. Parmi les défis les plus sensibles dans le Big Data figure la manière d'extraire des informations pertinentes à partir d'une masse de données pour d'éventuelles décisions décisives ?

La notion de Valeur correspond à l'intérêt qu'on peut tirer de l'utilisation de ces technologies. Selon les experts du domaine, les entreprises qui ne s'intéressent pas sérieusement au contenu de leurs volumes de données hébergées risquent d'être pénalisées et dépassées. Big Data désigne à la fois les grands volumes de données et la difficulté à extraire de cette masse

de données celles ayant suffisamment de valeur pour justifier leur analyse. Big Data offre un ensemble d'outils d'analyse de données qui peuvent servir à préserver un privilège concurrentiel [3].

1.5 Les facteurs d'émergence du Big Data

Bien que l'apparition du Big Data soit due essentiellement à l'éclatement du volume des données, il existe d'autres facteurs contribuant à cette émergence. Dans cette section on va voir les principaux facteurs qui ont préparé l'apparition du Big Data.

1.5.1 Croissance matérielle et baisse du prix

N'importe quelle solution informatique peut être vu comme des couches superposées, où chaque couche utilise les ressources de la couche inferieur et elle est limitée par ses contraintes, les solutions sont limitées par les contraintes des outils et des techniques utilisés qui elles même sont borné par les contraintes matériel. Donc n'importe quelle amélioration envisagée dans une solution donnée, revient à l'augmentation Hard ou bien l'optimisation soft.

De nombreux constructeurs ont été ravis de vendre de la puissance de calcul et des infrastructures toujours plus puissantes et de mois en mois chères. Toutefois la plupart des spécialistes sont restés prudents : certains d'entre eux ont rappelé tous les bienfaits de l'optimisation des traitements et de la rationalisation des systèmes d'informations, conscients qu'une limite des ressources allait prochainement imposer de revoir les besoins et les solutions architecturales proposées [4].

1.5.2 Evolution des systèmes de gestion des bases de données (SGBD)

Depuis l'apparition des précurseurs des bases de données vers les années 60 avec les systèmes IDS.I et IMS.I [9], les SGBD (*système de gestion des bases de données*) qui ont été créés afin de rationaliser, structurer et catégoriser les données, ont connu un vif succès [4], et ils n'ont pas cessé d'être améliorés pour répondre aux besoins fonctionnels en terme de complexité et de performance dans les limites des ressources matérielles.

La deuxième génération a été marquée par le modèle relationnel et son langage d'interrogation SQL (*Structured Query Language*), puis on est entré dans la troisième

génération qui est caractérisé par les extensions objet des systèmes relationnels et le langage SQL étendu (ou SQL3). Notons seulement que les SGBD réparties qui existent depuis les années 80 sont considérés comme des SGBD de la deuxième génération [9].

Quant à la quatrième génération, et avec l'arrivé du web, les SGBD ont élargi les types de donnés supportées vers le contenu multimédia d'où l'apparition de la notion *base de données multimédia*. Les SGBD de cette génération devront supporter et traiter des données plates, non structurées et même parfois mal introduites à l'aide du langage NoSQL (*Not only SQL*) qui sera détaillé dans le chapitre suivant.

1.5.3 Les systèmes distribués

La notion de distribution est basée sur deux principes simples qui peuvent être observé, et appliqué au quotidien :

- Diviser -le problème complexe- pour mieux régner -en terme du temps et ressources-
- Résultat ($\sum$ Effort $_i$) $\geq \sum$ Résultat (Effort $_i$)

L'informatique distribuée moderne a sans doute commencé avec les efforts de construction des cluster d'ordinateurs et les débuts des ad hoc dans les années 90 [10]. Bien que plus tard on a eu des accroissements consécutifs des capacités matérielles accompagnés d'une diminution considérable du coût, les supercalculateurs classiques restent très chers et face à des problèmes de plus en plus complexes, elles sont limitées par leurs architectures, d'où l'intérêt du développement des systèmes distribués.

Il y a lieu de noter qu'on a deux grands axes d'application des systèmes informatiques distribués. Le premier est les SGBD Distribués. Le second est le système de fichiers distribué, actuellement dominé par Hadoop.

1.5.4 Cloud computing

Un des facteurs importants dans l'émergence du Big Data est le cloud computing qui a grandement facilité l'accès aux infrastructures. Basé sur des ressources ajustables, par durée

identifiée et à un coût plus adapté, le cloud computing a ouvert de nombreuses portes aux projets innovants en abaissant considérablement le coût du ticket d'entrée sur ces solutions [4].

1.5.4.1 Définition

Selon NIST[2], Le cloud computing est un modèle permettant un accès réseau omniprésent, pratique et sur demande à un pool partagé de ressources informatiques configurables (réseaux, serveurs, stockage, applications et services) pouvant être rapidement mis en service et libéré avec un effort de gestion minimal. Interaction fournisseur de services [11, 12].

1.5.4.2 Modes de déploiement du cloud

Il existe trois modes de déploiement des services cloud [11]:

- **Cloud public :** Un cloud public est détenu et exploité par un fournisseur de services cloud, dans ce type de cloud tout le matériel, tous les logiciels et toute l'infrastructure sont la propriété du fournisseur du cloud.

- **Cloud privé :** Le cloud privé est un cloud dans lequel les services et l'infrastructure se trouvent sur un réseau privé.

- **Cloud hybride :** Le cloud hybride regroupe des clouds publics et privés, liés par une technologie leur permettant de partager des données et des applications. En permettant que les données et applications se déplacent entre des clouds privé et public.

1.5.4.3 Types de services cloud computing :

La plupart des services de cloud computing peuvent être classés en trois grandes catégories : IaaS, PaaS et SaaS. On les appelle parfois pile de cloud computing, car elles s'empilent les unes sur les autres.

[2] Le National Institute of Standards and Technology, ou NIST, est une agence du département du Commerce des États-Unis. Son but est de promouvoir l'économie en développant des technologies, la métrologie et des standards de concert avec l'industrie, site Web : *https://www.nist.gov*.

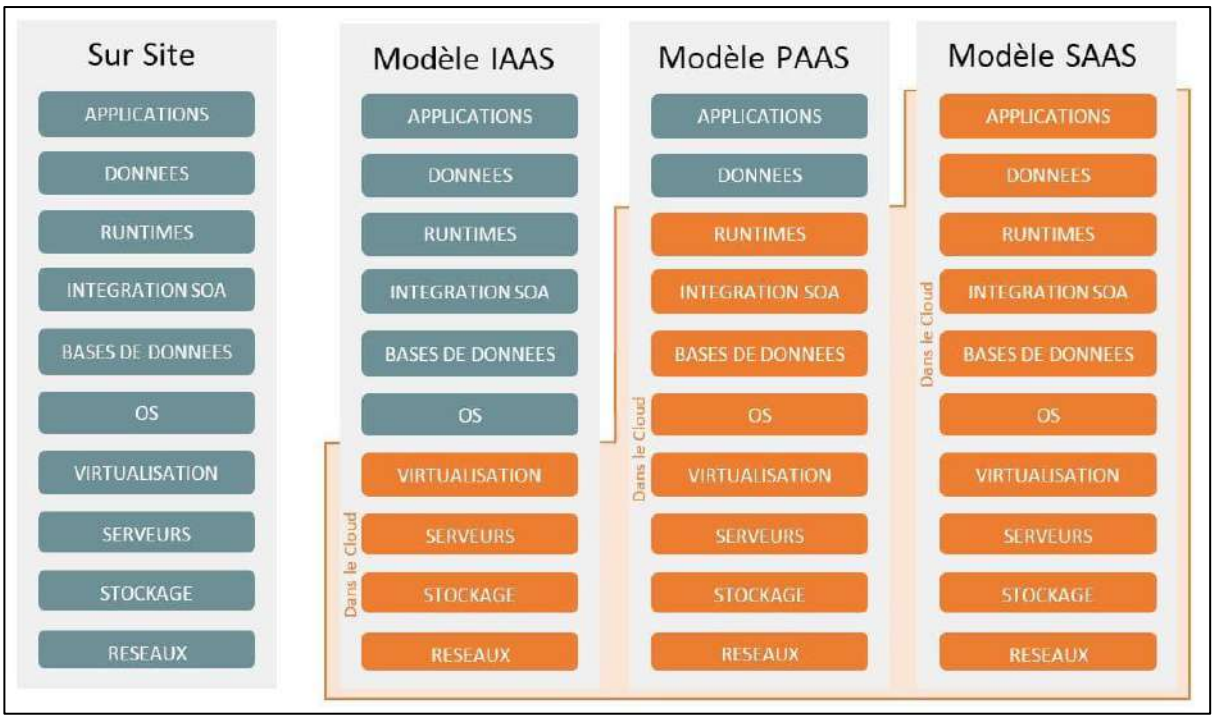

Figure 1-3 : Les types de service cloud (IaaS, PaaS et SaaS) [13-15]

- **IaaS (*Infrastructure as a service*) :** La catégorie la plus basique des services cloud. Avec l'IaaS, vous louez une infrastructure informatique (serveurs, machines virtuelles, stockage, réseaux, systèmes d'exploitation) auprès d'un fournisseur de services cloud, avec un paiement en fonction de l'utilisation.

- **PaaS (*Platform as a service*) :** Le Paas inclut les services de l'Iaas mais va encore plus loin, le prestataire fournit également l'ensemble des applications middleware : système d'exploitation, base de données, serveur web… En d'autres termes, le client loue l'exploitation des serveurs et les outils intégrés.

- **SaaS (*Software as a service*) :** Le Saas est le service le plus connu du grand public. Le fournisseur s'occupe de l'installation, de la configuration, du fonctionnement et de la maintenance de l'interface. Le client paye en général un abonnement mensuel et peut directement utiliser la plateforme que le fournisseur met à sa disposition.

1.5.5 Internet of Things (IOT)

Parmi les sources potentielles des données dont le stockage et le traitement entrent dans le cadre du Big Data on trouve l'*internet of things* (IoT). Le terme «internet of things» (ou internet des objets) été utilisé pour la première fois en 1999 par Kevin Ashton (le cofondateur de l'auto-ID center du MIT), ce terme a été employé pour désigner le monde des objets, appareils et capteurs interconnectés via internet [16].

Selon l'Union internationale des télécommunications (IUT), l'Internet of Things (IoT) 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 » [17].

Malgré le fait que tout le monde est d'accord pour dire que IoT est l'internet du futur, il n'existe pas de définition standard, unifiée et partagée de l'internet des objets, car chacun décrit la notion à partir de sa vision (technique [18] ou conceptuelle [19]). Parmi les définitions qui regroupent les deux visions on trouve celle qui stipule : « un réseau de réseaux qui permet, via des systèmes d'identification électronique normalisés et unifiés, et des dispositifs mobiles sans fil, d'identifier directement et sans ambiguïté des entités numériques et des objets physiques et ainsi de pouvoir récupérer, stocker, transférer et traiter, sans discontinuité entre les mondes physiques et virtuels, les données s'y rattachant » [20].

1.5.5.1 Domaines d'application d'IoT

En 2019, 8,3 milliards (estimé) d'objets sont connectés dans le monde, soit une augmentation de 18% par rapport à 2018 (on compte 7 milliards d'objets connectés en 2018) [21]. Bientôt chaque appareil que nous possédons et presque chaque objet qui existe seront connectés à l'internet, ce qui signifier que les IoT couvrira tous les aspects de notre vie.

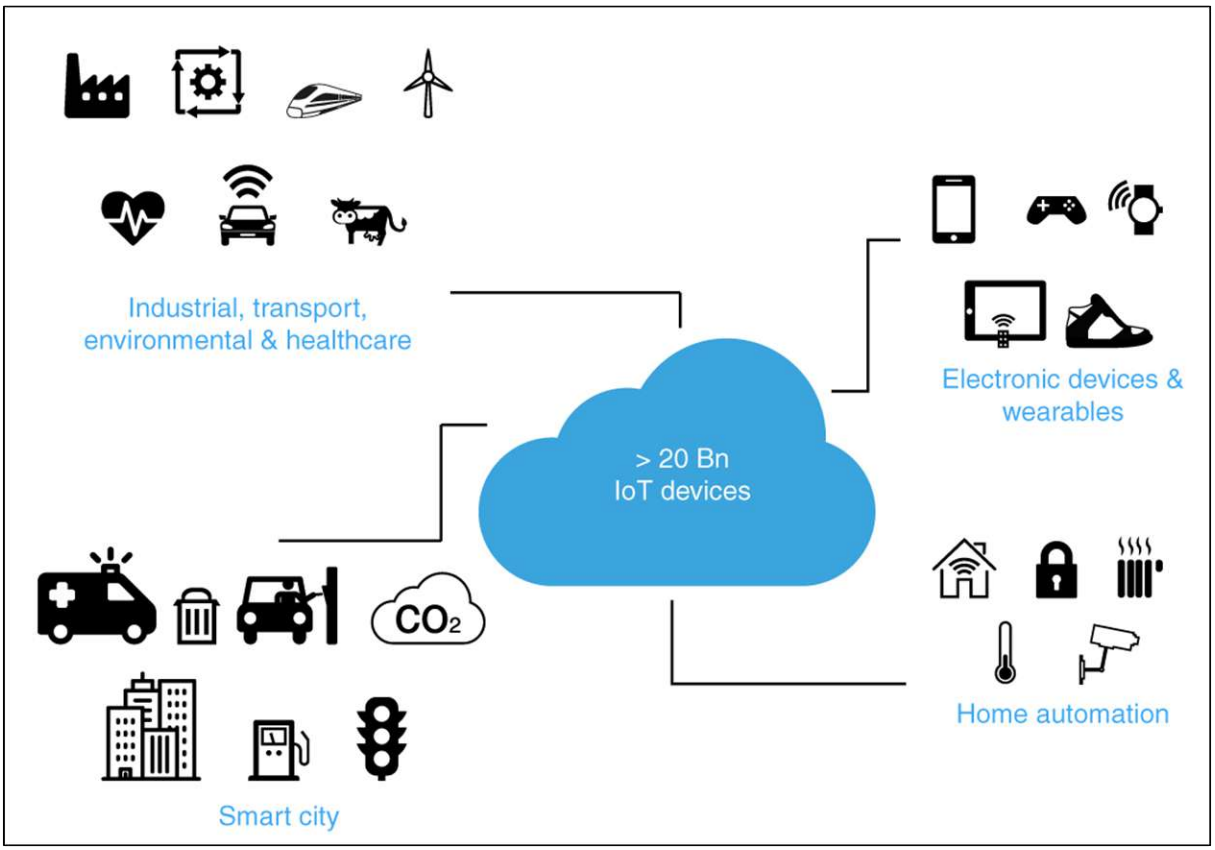

Figure 1-4 : Domaines d'application d'internet of Things [22]

On va citer quelques exemples d'application de l'IoT [23],

- **Les maisons intelligentes :** l'IoT recouvre tous les appareils électroménagers communicants, les capteurs (thermostat, détecteurs de fumée, de présence...), les compteurs intelligents et systèmes de sécurité connectés des appareils de type box domotique.

- **Médical, e-santé et fitness:** Le phénomène IoT est également très visible dans le domaine de la santé et du bien-être avec le développement des montres connectées, des bracelets connectés et d'autres capteurs surveillant des constantes vitales. Selon diverses projections (cf. Cisco et le cabinet Gartner), le nombre d'objets connectés devrait largement augmenter au fil des ans.

- **Transports intelligents:** l'IoT peut sauver des vies, réduire le trafic et minimiser l'impact des véhicules sur l'environnement, non seulement en connectant des voitures

qui se conduisent toutes seules, mais aussi en rendant le transport intelligent ainsi que les systèmes de logistique.

- **The smart city** : Grâce à des capteurs, le concept smart cities offre à ses habitants une qualité de vie maximale avec une optimisation de la consommation des ressources grâce à une combinaison intelligente des infrastructures (énergie, transport, communication) aux différents niveaux hiérarchiques (ville, quartier, bâtiment).

1.6 Les domaines d'application du Big Data

Les domaines d'application du Big Data sont vastes, tout secteur qui génère et traite un volume important de donnés peut-être une cible du Big Data. Il peut s'agir des logs d'un serveur Web, de données de flux de clics Internet, de contenu des réseaux sociaux, de texte provenant des mails, de journaux d'appels téléphoniques et de données des objets connectés.

Des organisations de différents domaines investissent dans des applications Big Data pour examiner de grands ensembles de données afin de découvrir toutes les tendances du marché, les préférences des clients et d'autres informations commerciales utiles. Parmi ces domaines on trouve :

- **Le secteur commercial :** les données Big Data sont traitées afin d'extraire une valeur offrant des opportunités d'innovation et de compétitivité. Cette valeur est réalisée en améliorant les processus de prise de décision, en étudiant de manière précise la satisfaction des clients et la performance des produits ou encore en personnalisant plus que jamais les produits et services [24]. Le secteur commercial s'incruste également dans l'Internet à travers le e-commerce et les sites de vente en ligne tels que eBay et Amazon qui doivent gérer des millions de transactions et pister les clics des utilisateurs afin de leur offrir les meilleurs produits.

- **Web behaviour :** les géants du Web tels que les réseaux sociaux font face à un lot énorme de données qu'ils doivent stocker, organiser et transférer. Ces données peuvent également servir à l'analyse afin de recueillir les préférences et les tendances utilisateur.

- **Le secteur de la santé :** En mappant les données sur les soins de santé avec des ensembles de données géographiques, il est possible de prédire une maladie susceptible de dégénérer dans des zones spécifiques. Sur la base de prévisions, il est plus facile d'élaborer des stratégies de diagnostic et de planifier le stockage des sérums et des vaccins.

- **Les médias et le divertissement :** Le Big Data fournit des points d'information exploitables sur des millions de spectateurs. Aujourd'hui, les environnements de publication adaptent les publicités et le contenu pour attirer les consommateurs.

- **Le secteur gouvernemental :** étant donné que le gouvernement agit dans tous les domaines, il joue donc un rôle important dans l'innovation des applications Big Data dans tous les domaines. Parmi les principaux domaines gouvernementaux on trouve :

    - Cybersécurité et renseignement
    - Prévision et prévention du crime
    - Prévision météo
    - Conformité fiscale
    - Optimisation du trafic
    - Recherche scientifique

1.7 Le Big Data Management et le processus décisionnel

Comme n'importe quel travail d'analyse de données, les applications Big Data suivent un processus de traitement sur les données passant par différentes phases nommé Big Data Management [25-27], Inespéré de *KDD process* (Knowledge Discovery in Databases) [28]. Cependant ce processus, n'est pas linéaire, on peut avoir besoin de revenir à des étapes

précédentes pour corriger ou ajouter des données. Un autre modèle adapté au Big Data connu sous l'acronyme KUBD (Knowledge Unveiling in Big Data) est en train d'émerger [29].

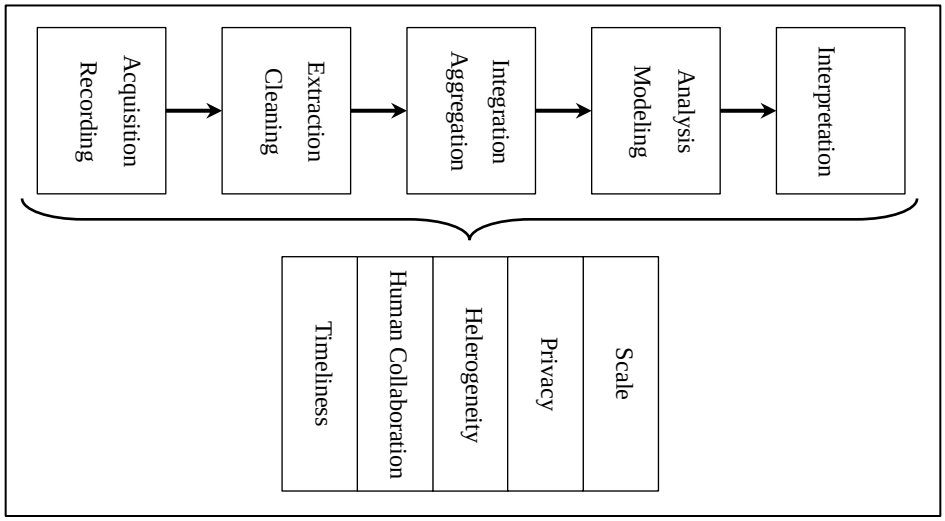

Figure 1-5 :Les phases principales du processus de traitement Big Data [25, 26]

On va décrire succinctement chacune de ces étapes.

1.7.1 Acquisition/Enregistrement

Correspond à la procédure d'acquisition des données Big et éliminer certaines données inutiles grâce à des filtrages et des compressions. Ce stage devra également assurer la génération des métadonnées sur la structure et la provenance des données, mais également sur les détails de l'opération de capture. Les métadonnées auront une importance capitale pour la suite des phases, plus particulièrement, l'analyse des données [25-27].

1.7.2 Prétraitement (pre-processing)

Souvent, les données capturées se trouvent dans un format inadapté à l'analyse. Cette phase s'occupe de corriger leur structure et d'extraire l'information significative, mais également d'éliminer les données potentiellement erronées. En effet, le critère de véracité du Big Data stipule que les données sont parfois non fiables et doivent être épurées avant l'analyse [25-27].

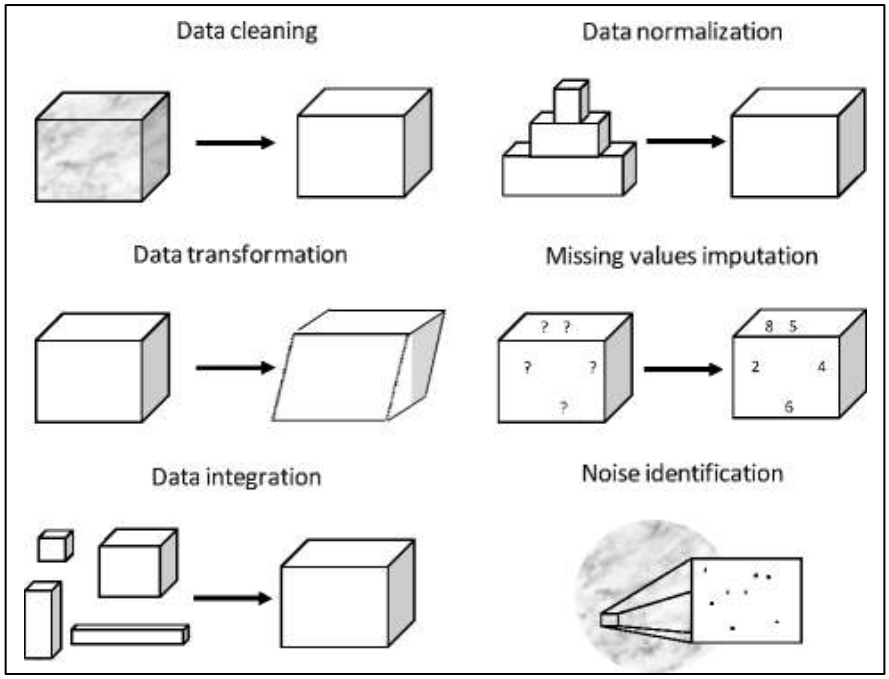

Figure 1-6 : Les différentes tâches de la phase prétraitement [30]

1.7.3 Traitement (processing)

Les analyses à grande échelle font appel à des ensembles de données différents en structure et en taille. Un défi important correspond à trouver la représentation la plus adéquate pour les stocker et à intégrer ces ensembles entre eux de façon à conduire une analyse globale [25-27].

1.7.4 Analyse / Modélisation

Il s'agit de l'analyse des données afin de déceler des modèles intrinsèques, d'extraire des relations et des connaissances, mais aussi de corriger les erreurs et d'éliminer les ambiguïtés [25-27].

1.7.5 Interprétation

Les décideurs devront valider les résultats en retraçant les opérations effectuées. Des outils doivent être mis en place afin de faciliter ce processus. Ils doivent offrir des visualisations

interactives des données, permettre de retracer leur provenance et d'appliquer des modifications dessus puis voir l'impact sur les résultats en temps réel [25-27].

1.8 Conclusion

On vient de voir dans ce chapitre les aspects théoriques du Big Data, comme la définition, les caractéristiques, et les principaux facteurs d'émergence notamment l'*Internet of Things* et le cloud.

Ainsi on a vu les phases d'un traitement Big Data, ce travail qui nécessite souvent l'intervention humaine. Afin de concrétiser le dite travail, des systèmes conçus interviennent dans les différentes phases de ce processus. Dans le chapitre suivant on va présenter quelques-uns de ces systèmes et ces outils.

Chapitre 2

Technologies Big Data et outils d'analyses

2.1 Introduction

Après avoir exposé des notions générales sur le Big Data dans le chapitre 1, nous décrirons dans ce chapitre les différents aspects techniques et les technologies nécessaires pour un traitement Big Data : (1) Comment les données sont stockées ?; (2) Comment les données sont manipulées ?; (3) Comment le traitement et l'analyse seront faits?

Ce sont justement les questions auxquelles n'importe quelle solution Big Data doit répondre.

Nous allons également présenter les solutions Big Data les plus répondues, notamment Apache Hadoop et Apache Spark, leurs principaux composants, les points de convergence et leurs dissimilitudes.

En plus, nous allons voir les différents outils d'analyse de données qui permettent de concevoir les tableaux de bord.

2.2 Les Technologies

Dans cette section on développe les notions de stockage des données volumineux via les systèmes de fichiers distribués, le nouveau modèle de gestion de données dit *NoSQL* qui permet la manipulation d'un tel type de données, ainsi que le paradigme *MapReduce* de traitement des données distribuées.

2.2.1 Systèmes de fichiers distribués

En vue de faciliter le traitement des données, plusieurs solutions de stockage distribuées ont été développées tout en tenant compte du paradigme de traitement (en cluster de nœuds) utilisé. Nous citerons ci-après les solutions les plus populaires :

2.2.1.1 Google File System (GFS)

Ce système a été développé par Google fin 2003 [31] afin d'offrir un environnement de stockage redondant qui fonctionne sur un cluster constitué d'un grand nombre de machines [32]. Le GFS est orchestré par un nœud maître et contient un grand nombre de serveurs de stockage. Chaque fichier est divisé en plusieurs blocs de fichiers, dits *Chunks*. Les *Chunks* sont stockés et remplacés plusieurs fois tout au long du réseau (au minimum trois fois). Le nœud maître stocke les métadonnées associées. Ces informations sont tenues à jour grâce aux messages de mise-à-jour de statut de chaque serveur de stockage, dits *HeartBeat* [33, 34].

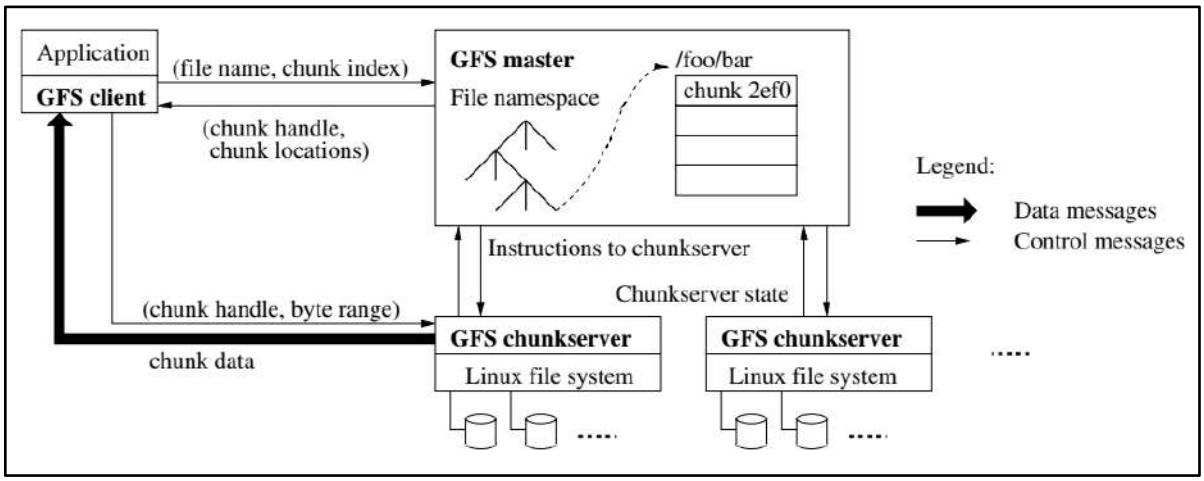

Figure 2-1 : Architecture GFS [31]

2.2.1.2 Hadoop Distributed File System (HDFS)

C'est l'homologue de GFS, ce système repose sur une « architecture HDFS Master/Slave » où chaque cluster comporte un *NameNode* qui joue le rôle de serveur principal et plusieurs autres nœuds qui stockent les données appelés *DataNodes*. Ainsi, les clients peuvent accéder aux bonnes données au bon moment. Le *NameNode* se charge également d'ouvrir, fermer et renommer les fichiers ou les dossiers [35, 36].

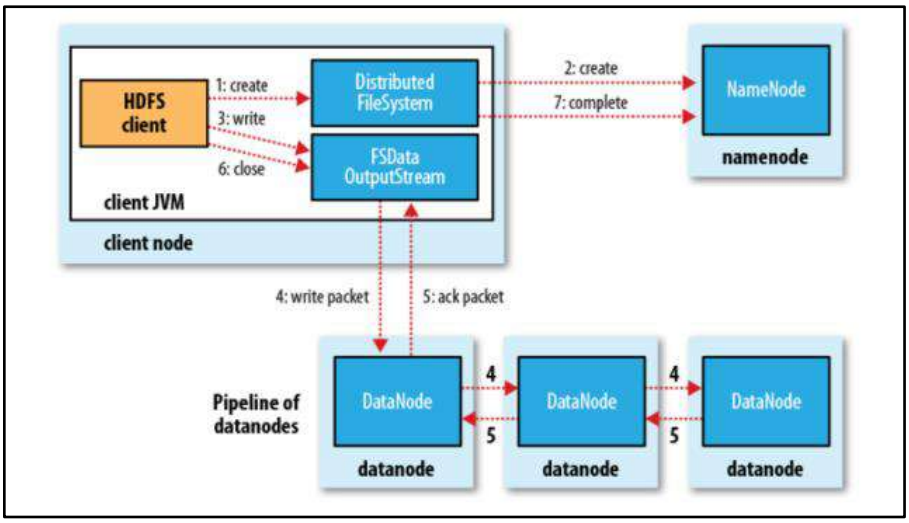

Figure 2-2 : Architecture d'Apache HDFS [37]

2.2.1.3 BlobSeerted File System (BSFS)

Est un système de fichiers distribués optimisé pour les opérations concurrentes. Son architecture lui permet de gérer la réplication et la fragmentation de manière transparente. Sa gestion du *versioning* lui permet de garder plusieurs versions de la même donnée en même temps, une version finale consolidé sera reconstruite à la demande. Le système offre un haut débit, mais requiert une configuration méticuleuse qui n'est pas toujours facile à maitriser [35, 36];

Néanmoins, il existe un tableau de comparaison dans l'index B décrit quelques points de différences entre ces systèmes.

2.2.2 Les bases de données NoSQL

Depuis l'apparition des systèmes de gestion des données, le modèle relationnel règne sur la scène. Cette domination du relationnel provient de son pragmatisme et de ses capacités à répondre aux défis du moment [32], en permettant une meilleure performance de traitement avec les ressources disponibles à une époque où les capacités matérielles étaient limitées.

Néanmoins, avec l'évolution du Web, les besoins ont changé, les types de données manipulées ont évolué, de même que les ressources matérielles. Face à ce changement (ce qu'on appelle souvent *Web scale*) le modèle relationnel montre des points de faiblesse.

Afin de faire face à ces défis, un nouveau modèle de gestion de données, dit NoSQL (*Not Only SQL*), a était conçu. Celui-ci garantit les propriétés de scalabilité en s'affranchissant de la règle ACID (voir sa définition dans l'Annexe A) qui a longtemps gouverné les systèmes conventionnels [35, 38, 39].

Depuis qu'il a été inventé en 2009 lors d'un événement sur les bases de données distribuées [32], le NoSQL est vu - par ses défenseurs- comme une évolution bienvenue de l'antique modèle relationnel, tandis que ses détracteurs le considèrent plutôt comme une régression.

2.2.2.1 Types de moteurs NoSQL

On distingue dans ce cas quatre modèles selon le schéma ou la structure des données qu'ils manipulent (Voire la figure ci-dessous).

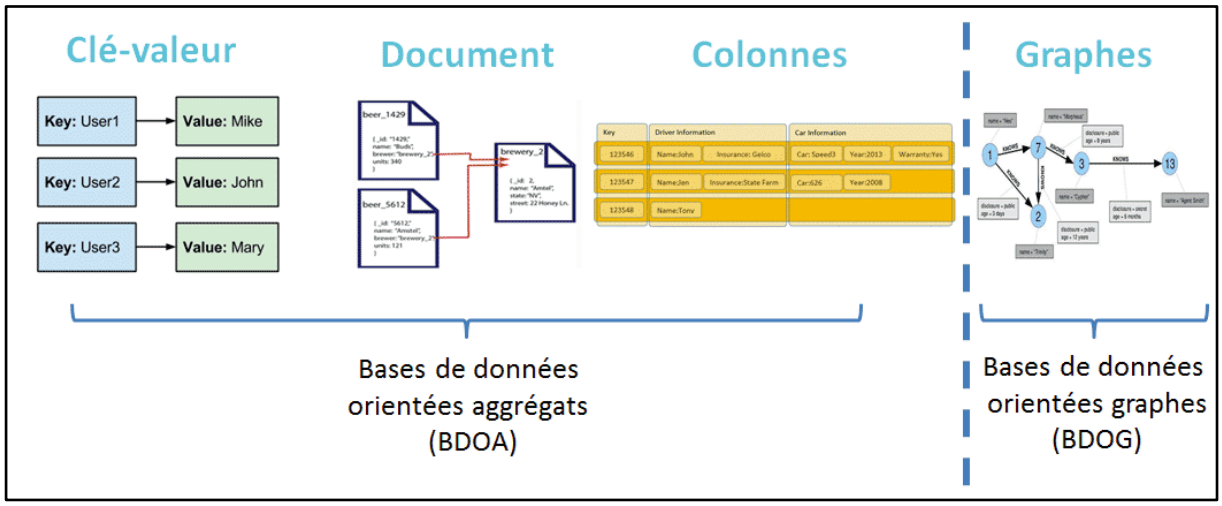

Figure 2-3 : Les types de bases de données NoSQL [40]

- **Paires clé-valeur (ECV) :** Il s'agit d'une représentation simple de la donnée basée sur une collection de paires « clé : Valeur » appelé ECV (entrepôt clé-valeur) dont la clé forme un identifiant unique. La plupart des solutions clé-valeur se basent sur le papier d'Amazon Dynamo [41], parmi ces solution, on trouve : Amazon DynamoDB, *Riak* [42, 43] et *Voldemort* [38], on peut également trouver d'autres solution qui se basent sur *Memcached* conçu par Brad Fizpatrick [44] comme *Redis* [45-47],

- **Orientés documents (BDOD) :** Ces bases de données stockent des données semi-structurées : le contenu est formaté JSON ou XML, les objets dits « documents » stockent les données sous forme d'attributs où chaque attribut peut être un autre document offrant ainsi une structure récursive. *MongoDB* et *CouchDB* sont des exemples de systèmes qui implémentent ce modèle [38].

- **Orientés colonnes (BDOC) :** Ces bases de données se rapprochent des bases de données relationnelles, à ceci près qu'elles permettent de remplir un nombre de colonnes variable. Les solutions les plus populaires dans cette catégorie sont *Cassandra* offerte par Facebook, BigTable de Google, et l'open source *Apache HBase* [38].

- **Orientés graphe (BDOG) :** Ces bases de données, basées sur la théorie des graphes, sont gérées par nœuds, relations et propriétés. Elles gèrent des données spatiales, sociales ou financières (dépôts/retraits). Parmi les BDOG les plus utilisées aujourd'hui citons : *Neo4J*, *Infinite Graph*, et *Titan* [32].

2.2.3 Le paradigme MapReduce

MapReduce est un modèle de programmation (*Design Pattern*) parallèle inventé par Google en 2004 [48, 49]. Il est principalement utilisé pour la manipulation et le traitement des données volumineuses (stockés sous le format ECV ou les BDOD) au sein d'un cluster de nœuds.

MapReduce se base sur deux fonctions *map()* et *reduce()*.

- **Map :** Dans cette étape le nœud (dit *mappers*) analyse un problème, le décompose en sous-problèmes (le découpage se fait sur les données et/ou tâches et peut être récursive a plusieurs niveaux). Les sous-problèmes sont ensuite affectés par la fonction Map aux nœuds fils.
  Interface : *map(clé1, valeur1)* → *list(clé2, valeur2)*

- **Reduce :** Dans une seconde phase, les nœuds fils retournent leurs résultats au nœud parent (*reducer*). Qui calcule un résultat intermédiaire puis il remonte l'information à son tour. À la fin du processus la fonction Reduce du *root* retourne le résultat final.
  Interface : *reduce(clé, list(valeur))* → *result*

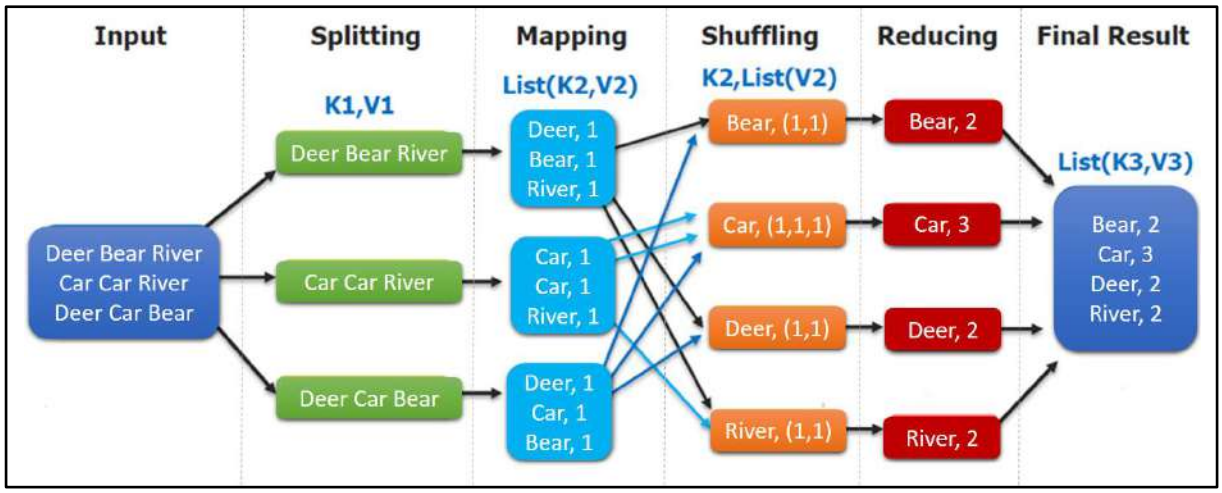

Figure 2-4 : Exemple d'un programme MapReduce - processus comptage des mots [50]

Néanmoins, il existe d'autres étapes indispensables comme :

- **Splitting :** il s'agit de la décomposition *dataset* en sous-ensembles pour permettre la distribution de son traitement.

- **Shuffling :** c'est un processus intermédiaire entre l'opération *map* et l'opération *reduce*, le but de cette opération est de **réorganiser et de restructure** les résultats de la fonction *map* pour faciliter la tâche à la fonction *reduce*, Notons seulement que le *shuffling* peut commencer même avant la fin de la phase de la carte, afin de gagner du temps.

Depuis son invention, le model MapReduce a fait ses preuves dans le domaine de programmation distribué et il a été largement implémenté par plusieurs Frameworks : *Apache Hadoop* (le plus populaire), *Apache CouchDB* [51], *Disco Project*, *Infinispan* [52], *Riak* ...etc.

Ce modèle est largement utilisé dans le domaine du Big Data. Parmi les exemples d'utilisation de MapReduce on cite :

- Analyse statistique.
- Gestion des jointures pour les Big tables.
- Calcul matriciel parallèle.

Certaines études définissent des facteurs de performance pour les algorithmes du MapReduce [53]

- Mode d'entrée/sortie : la façon dont le lecteur récupère les informations;

- Analyse des données : ensemble des opérations et des traitements appliqués sur les données;
- Indexation des données : la présence ou l'absence d'indexation influence grandement sur la performance.
- Stratégie de tri : tri des données traitées et des résultats intermédiaires.

2.3 Les solutions Big Data

Parmi les système d'analyse des données de type Big Data, figure la solution Google dit Google Cloud, la solution Azure de Microsoft, et le Framework open source Apache Hadoop, souvent appelé écosystème, et qui est le plus populaire.

Nous décrivons ci-après la solution Hadoop ainsi que la solution Apache Spark.

2.3.1 Le Framework Hadoop

Apache Hadoop (*High-availability distributed object-oriented platform*) est un Framework open source crée en 2009, qui permet le stockage et le traitement distribué de grands ensembles de données. Son architecture *scalable* lui permet de passer d'un serveur à des milliers de machines d'une manière transparente [54].

Pour répondre aux différents défis du Big Data, allant du stockage jusqu'à l'analyse de données volumineuse, Hadoop se base sur une structure modulaire (voir la figure ci-dessous) extensible avec la possibilité de substituer les modules par défaut (le cas de MapR FS qui remplace HDFS).

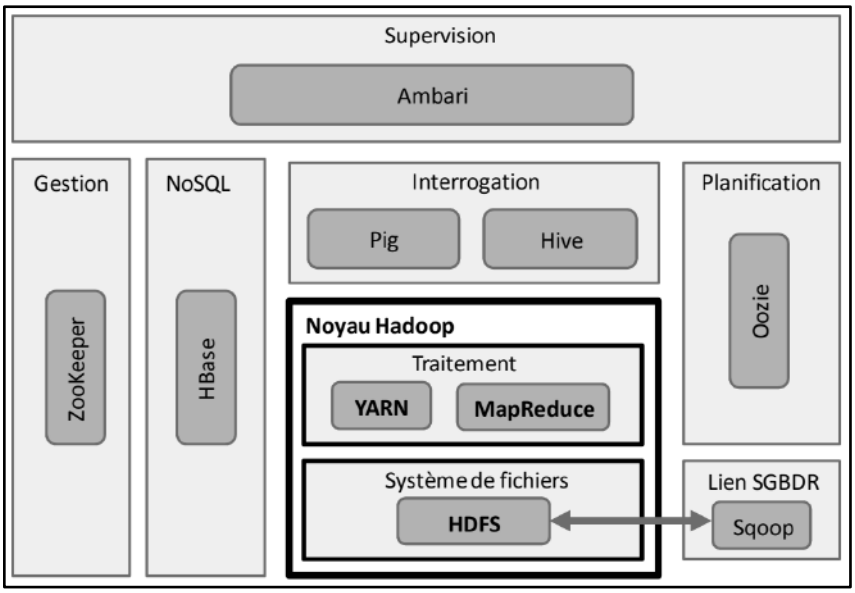

Figure 2-5 : Composants de l'écosystème Apache Hadoop [4]

On va voir dans cette section ces principaux composants ainsi que les distributions les plus populaires.

2.3.1.1 Les Composants du Hadoop

Voici une liste non-exhaustive des principaux composants du Framework Hadoop :

1. **Hadoop Distributed File System (HDFS) :** Déjà vus dans la section précédente, le HDFS gère la partie stockage en prenant en charge les fonctions de réplication et de tolérance aux pannes sur un cluster. Notons seulement qu'Hadoop propose une API qui permet d'intégrer d'autres systèmes comme le système S3 d'Amazon [4].

2. **YARN :** Est l'implémentation *MapReduce* d'Hadoop. L'architecture de YARN repose sur deux principaux processus: *ResourceManager* (le gestionnaire de ressources partagées) et *ApplicationMaster* (réclame les ressources auprès du *ResourceManager*) [4].

3. **HBase :** Est une base NoSQL orientées colonnes, inspirée du *BigTable* de Google, HBase permet de gérer de très grandes tables avec des milliards d'enregistrements et plusieurs millions de colonnes.

4. **ZooKeeper :** Est un service de coordination distribué open source déployée sur un cluster Hadoop pour faciliter l'administration des infrastructures, gérer les informations de

configuration et garantir la synchronisation, pour une large variété d'applications distribuées [55].

5. **Pig :** Apache Pig est une plate-forme d'analyse de très grands ensembles de données. La propriété saillante des programmes Pig est que leur structure permet une parallélisation importante. Pig utilise un compilateur dit *Pig Latin* pour produire des séquences de programmes MapReduce [56].

6. **Hive :** Comme Pig, Hive est utilisé pour l'analyse d'ensembles de données très volumineux, cependant il utilise un langage plus proche du SQL dit *HiveQL*, d'où sa facilité. Il est utilisé surtout en mode batch [4].

7. **Sqoop :** Apache Sqoop est un outil qui permet de transférer de grands volumes de données entre Hadoop et un SGBDR (comme MySQL, Oracle).

8. **Oozie :** C'est un outil de planification de tâches Hadoop, qui permet de regrouper plusieurs types d'opérations : Pig, Hive, MapReduce ou Sqoop.

9. **Flume :** Est un service distribué, fiable et disponible pour la collecte, l'agrégation et le transfert efficaces de grandes quantités de données de log [57].

2.3.1.2 Les Principales Distributions Hadoop

Une distribution est un ensemble cohérent de différentes briques qui constituent l'écosystème Hadoop packagées par un fournisseur. Parmi les principales distributions existent :

1. **Cloudera :** Au noyau Hadoop, Cloudera ajoute ses propres interfaces graphiques d'administration, des assistants de déploiement et des solutions d'intégration à l'aide d'une collection d'extensions (Ambari, Hue et Impala). Cloudera propose par ailleurs une version open source de sa plateforme [4].

2. **Hortonworks :** Fondée en 2011 par Yahoo dans le but de faciliter l'adoption de la plateforme Hadoop d'Apache. Ce projet n'utilise que des composants open source, et il contribue même dans le noyau d'Hadoop [4].

3. **MapR :** Fondée en 2009, MapR est un partenaire technologique de Google. En enrichissant le noyau Hadoop et remplaçant quelques composants de base avec des solutions propriétaires (comme MapR FS au lieu du HDFS), MapR propose trois distributions : M3 (gratuite), M5, M7(la plus complète) [4].

4. **Amazon Elastic MapReduce (EMR) :** Proposée par Amazon en 2009, EMR est une distribution cloud d'Hadoop ce qui offre une grande élasticité. Parmi ses inconvénients majeurs figure le temps de latence élevé pour les entrées/sorties [4].

Chaque distribution Hadoop a ses propres avantages et inconvénients. Le choix d'une distribution Hadoop se base justement sur la valeur supplémentaire offerte par chaque distribution en pesant les risques et le coût.

2.3.2 Le Framework Spark

Bien que MapReduce a grandement simplifié l'analyse Big Data, mais face à des applications exécutant des opérations itératives sur un même échantillon de données (comme Machine Learning par exemple). Les algorithmes MapReduce passent ainsi la majorité de leur temps d'exécution sur des opérations de lecture et écriture [4].

Pour répondre à cette problématique, des chercheurs de Berkeley au début des années 2010 ont introduit un nouveau paradigme appelé RDD (*Resilient Distributed Datasets*). RDD se base sur le traitement des données en mémoire ce qui réduit les Entrées / Sorties disque. Ce concept de RDD, est à l'origine de la naissance de Spark, et lui permet de se révéler, selon les traitements, jusqu'à 100 fois plus rapide que MapReduce [4]. Donc on peut considérer Spark comme une adaptation du système Hadoop et ses composants au modèle RDD.

2.3.2.1 Les Composants de Spark

Comme le system Hadoop, Spark se base sur une structure en briques (voir figure).

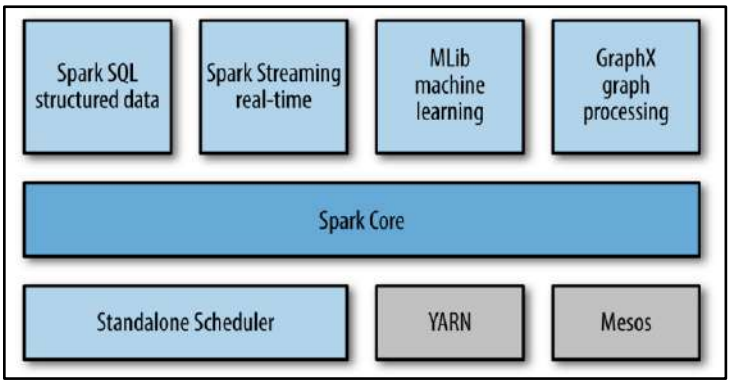

Figure 2-6 : Architecture d'Apache Spark [58]

Nous décrivons ci-dessous les composants de base de ce système.

1. **Apache Spark Core :** C'est le noyau sur lequel toutes les autres fonctionnalités se basent. Il permet le traitement des données en mémoire.

2. **Spark SQL :** Développé à la base de Hive, Spark SQL est le composant qui permet à Spark non seulement d'interroger une variété de sources de données (notamment les tables Hive, Parquet et JSON), mais il permet aussi de faire des analyses complexes en mixant le SQL avec les langages évoluées (Python, Java et Scala).

3. **Spark Streaming :** C'est le composant qui prend en charge le traitement des flux de données à chaud. Ces flux peuvent être les logs de serveur ou les messages d'une file d'attente d'un service Web. Via un API, Spark Streaming permet aux programmeurs de manipuler des données stockées en mémoire, sur disque ou arrivant en temps réel d'une manière transparente.

4. **MLlib (Machine Learning Library) :** MLlib est la machine learning de Spark qui travaille en mémoire distribuée. MLlib fournit plusieurs types d'algorithmes d'apprentissage automatique, notamment la classification, la régression... etc.

5. **GraphX :** C'est un Framework de traitement distribué des graphes sur Spark. GraphX qui fournit divers opérateurs permettant de manipuler des graphiques (comme *subgraph* et *mapVertices*) et une bibliothèque d'algorithmes de graphes les plus connus (par exemple *PageRank*).

6. **Gestionnaires de cluster (Cluster Managers) :** Afin de garantir une meilleure scalabilité et flexibilité, Spark dispose de son propre gestionnaire de cluster *Standalone Scheduler*, comme il peut s'exécuter sur divers gestionnaires de cluster, notamment Hadoop YARN, Apache Mesos.

Notons seulement que cette liste est extensible. Dans les versions plus récentes on peut trouver également d'autres composants, notamment Spark R et BlinkDB.

2.4 Les tableaux de bord et l'analyse des données

L'analyse des données est le processus de traitement d'un nombre important de données en se basant sur des méthodes statistiques afin d'extraire des informations utiles. Le succès de cette discipline dans les dernières années est dû à la facilité d'analyse apportée par les outils BI permettant de représenter des données complexes sous forme de synthèses et de graphes en mettant en évidence des relations difficilement saisies par l'analyse directe des données.

2.4.1 Définition d'un tableau de bord

Un tableau de bord de gestion est une façon de sélectionner, d'agencer et de présenter les indicateurs essentiels et pertinents, de façon sommaire et ciblée, en général sous forme de « coup d'œil » accompagné de reportage ventilé ou synoptique, fournissant à la fois une vision globale et la possibilité de forer dans les niveaux de détail [59].

Le tableau de bord mise principalement sur la qualité de l'information et non sur la quantité. Il met en évidence les résultats significatifs, les exceptions, les écarts et les tendances ; il fournit à son utilisateur un modèle cohérent en regroupant les indicateurs de façon à frapper son imagination – ce schéma intégré permet d'enrichir d'autant l'analyse et l'interprétation de l'information ; il représente les indicateurs sous une forme compréhensible, évocatrice et attrayante, pour en faciliter la visualisation [59].

2.4.2 Les indicateurs de performances d'un tableau de bord

Les KPI sont utilisés pour déterminer les facteurs pris en compte pour mesurer l'efficacité globale d'un dispositif commercial ou marketing ou celle d'une campagne ou action particulière. Ils peuvent donc être utilisés de manière ponctuelle pour une campagne ou de façon

permanente pour mesurer les résultats d'un dispositif (site e-commerce, magasin, community management, centre de relation client). Pour assurer le pilotage d'une activité, les KPI peuvent être regroupés dans un tableau de bord [60].

2.5 Les solutions BI

Les outils Business Intelligence (BI) comprennent les stratégies et technologies utilisées par les entreprises pour l'analyse de données d'informations commerciales [61]. Avant de voir quelques exemples, il faut noter qu'il y a trois catégories d'outils : Tableaux de bord, outils de conception, et outils de développement.

2.5.1 Tableaux de bord

Il s'agit des solutions clé en main destinées aux utilisateurs finaux, ces solutions sont spécialisées dans des domaines bien définis (Marketing, gestion de production, suivi des log …) et ils sont fortement liées à leurs sources de données.

2.5.2 Solutions de conception des tableaux de bord

Ce sont les différentes solutions qui permettent de concevoir des tableaux de bord et de les personnaliser. Nous en citerons quelque unes :

2.5.2.1 Power Pivot

C'est un composant Excel, Power Pivot est une technologie de modélisation de données qui permet de créer des modèles de données, d'établir des relations et de générer des calculs. Power Pivot permet de travailler avec de grands ensembles de données, de développer des relations étendues et de générer des calculs complexes (ou simples) au sein d'un environnement Excel [62, 63].

Power Pivot est l'un des trois outils d'analyse des données disponibles dans Excel :

- Power Pivot
- Power Query
- Power View

2.5.2.2 Power BI

Power BI est un service d'analyse commerciale de Microsoft. Son objectif est de fournir des visualisations interactives et des fonctionnalités de business intelligence avec une interface assez simple pour que les utilisateurs finaux puissent créer leurs propres rapports et tableaux de bord [64]. Il existe deux versions : Desktop gratuit, et la version server payante.

2.5.2.3 Klipfolio

C'est un service en ligne - payant - de conception de tableaux de bord qui permet de créer plusieurs tableaux de bord en temps réel rapidement et facilement. Cet outil prend en charge plus de 100 applications cloud, dont Google Analytics, HubSpot, Facebook et SalesForce [65].

2.5.3 Outils de développements BI

Comme les outils de conception, les outils de développement permettent de concevoir des tableaux de bord, mais ils sont destinés aux informaticiens car ils requièrent un travail de développement et de codage afin d'obtenir des solutions personnalisées.

2.5.3.1 Oracle BI

Cette technologie offre aux utilisateurs à peu près toutes les fonctionnalités de BI, telles que les tableaux de bord, l'intelligence proactive, les alertes, ad hoc, etc. Oracle est également idéal pour les entreprises qui ont besoin d'analyser de gros volumes de données (provenant de sources Oracle et non Oracle) car il s'agit d'une solution très robuste [66].

2.5.3.2 SAP BI

SAP Business Intelligence propose plusieurs solutions d'analyse avancées, notamment l'analyse prédictive de la BI, l'apprentissage automatique, la planification et l'analyse en temps réel. La plateforme de Business Intelligence en particulier propose des applications de reporting et d'analyse, de visualisation et d'analyse de données, d'intégration bureautique et d'analyse mobile. SAP est un logiciel robuste destiné à plusieurs domaines (informatique, utilisation finale et gestion) et offre une multitude de fonctionnalités sur une seule et même plate-forme [67].

2.5.3.3 SQL Server Analysis Services (SSAS)

SSAS est utilisé par les organisations pour analyser et donner un sens aux informations éventuellement dispersées dans plusieurs bases de données, ou dans des tables ou des fichiers disparates. Microsoft a inclus dans SQL Server un certain nombre de services liés à la veille stratégique et à l'entreposage de données. Ces services incluent Integration Services, Reporting Services et Analysis Services. Analysis Services inclut un groupe de fonctionnalités OLAP et d'exploration de données et se décline en deux versions: multidimensionnelle et tabulaire [68].

2.6 Conclusion

Dans ce chapitre nous venons de décrire les aspects techniques d'un traitement Big Data, et la manière avec laquelle les différentes solutions ont répondu à ces aspects via des composants, dont chaque composant traite un aspect.

Par ailleurs, nous avons passé en revue les deux principaux Frameworks open source Hadoop et Spark. Ce dernier est considéré comme une reproduction de Hadoop avec une certaine adaptation, car Spark implémente le modèle RDD (*Resilient Distributed Datasets*) tandis qu'Hadoop se base sur le modèle MapReduce. Ce qui a permis à Spark de gagner davantage de rapidité par rapport à Hadoop. Cependant, la rapidité n'est pas un facteur décisif pour toutes les applications. De plus, Spark ne dispose pas de son propre système de fichiers distribués.

Notons seulement qu'il existe d'autres Frameworks, comme Apache Storm qui est lui-même une copie d'Hadoop mais qui est spécialisé dans le traitement en temps réel. Il apparait donc clairement qu'Hadoop, avec ses services, ses diverses distributions et plusieurs Frameworks clones, représente un précurseur des systèmes Big Data, certains utilise même la notion « jungle de l'éléphant » [4].

Nous avons également vu dans ce chapitre quelques solutions d'analyse de données et de conception de tableaux de bord, le choix entre ces solutions dépend d'une étude qui se base sur le besoin client, la nécessité de personnalisation des résultats, la variété des sources de données, …etc.

DEUXIEME PARTIE :

Contribution :

Conception & implémentation

Chapitre 3

Conception

3.1 Introduction

Après avoir vu la notion du Big Data et ses déférentes technologies et solution, nous posons dans le présent chapitre notre problématique engendrée par les limites et les contraintes des systèmes d'information actuels.

Ensuite nous donnons une vue globale de notre proposition qui sera détaillée plus en avant dans ce chapitre en :

- Décrivant le système proposé.
- Donnant l'architecture du système via des schémas et des diagrammes.
- Décortiquant ses déférents modules et composants.

L'étape de conception traitée par le présent chapitre est décisive, car elle est considérée comme 'Road Map' de l'implémentation et la réalisation du projet.

3.2 Problématique

Bien que l'Entreprise Nationale de Services aux Puits « ENSP » soit une moyenne entreprise (3200 employés), elle possède une importante infrastructure informatique étalée sur plusieurs sites, avec une collection de solutions pour gérer ses déférentes activités. Malheureusement, ces solutions sont développées avec des technologies distinctes et utilisent des bases de données hétérogènes (SQL SERVER, Oracle, InterBase, Hyper File, Listes SharePoint, Fichier plat structurés (Excel, CSV, DBF) et des documents Word ou PDF) avec

un volume important dépassant les 100 Go tous types confondus, et avec une certaine vélocité notamment pour les solutions métier des chantiers.

Rien que pour le volet Ressources Humaines, l'ENSP exploite parallèlement un progiciel de gestion « ResHum » sous Oracle et une autre solution pour le traitement de la paie sous Hyper File, sans compter l'historique (ultérieur a 2002) qui est sous format DBF. Ce caractère hétéroclite constitue d'un côté une contrainte pour une analyse globale, et d'un autre côté un environnement opportun pour l'utilisation des méthodes d'analyse Big Data.

Donc notre problématique est de faire une analyse inter systèmes d'information, avec la contrainte que ces derniers utilisent des sources de données hétérogènes.

3.3 Proposition

Notre objectif est d'offrir une solution d'analyse pour l'ensemble des données, sous forme d'un service au-dessus de nos solutions et systèmes d'information qui existent. Comme il s'agit d'un système décisionnel dont on n'a pas besoin de modifier la donnée, nous pouvons interroger directement les sources de données sans passer par la logique de leurs systèmes d'information.

Ce service se compose de trois étapes qui seront implémentés en modules par la suite :

- **Préparation des données :** phase d'importation et de prétraitement des données. Cette étape de prétraitement, même si elle n'est pas demandée pour toutes les sources de données, elle demeure primordiale pour les sources non fiables telles que les données des capteurs ou des fichiers saisis.

- **Récupération et Normalisation :** le but de cette phase est de récupérer les données de différentes sources et les fournir en format exploitable par plusieurs outils ou composants de traitement de données.

- **Traitement :** c'est l'étape d'analyse et de traitement des données.

La figure ci-dessous illustre notre proposition.

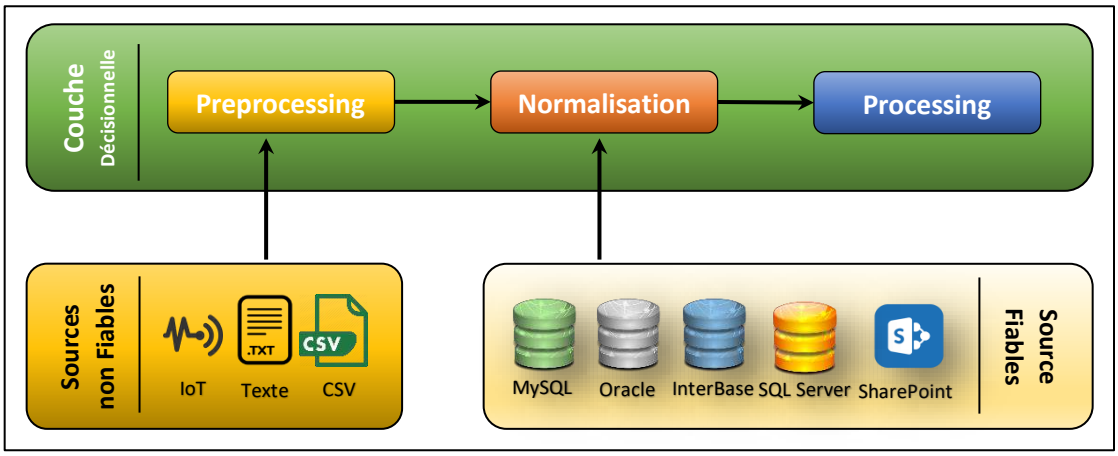

Figure 3-1 : le fonctionnement et étapes de la solution proposée

3.4 Description et Architecture du système

Notre vision consiste à introduire une couche décisionnelle qui englobe un ensemble de tableaux de bord de déférentes disciplines[3] de l'entreprise. Cette couche doit être indépendante de tout système d'information (voir la figure ci-dessous).

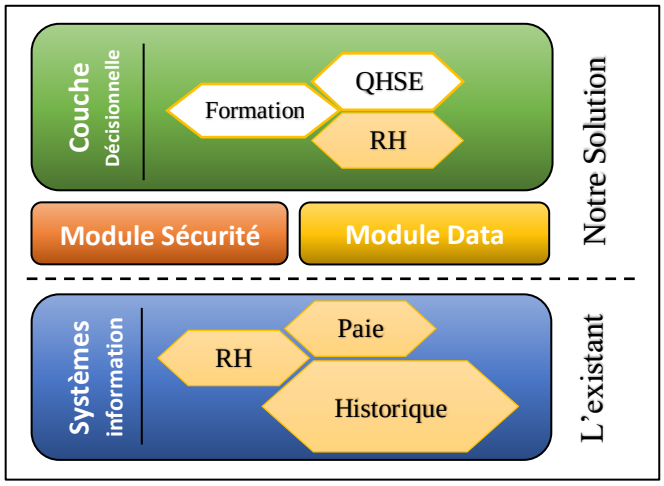

Figure 3-2 : Architecture modulaire de notre système

[3] A titre de validation, nous n'implémentons que le tableau de bord RH.

La conception de l'application est basée sur une architecture modulaire comportant :

- **Un Module sécurité :** gère la partie sécurité d'accès et les profils utilisateurs.
- **Un Module Data :** s'occupe du prétraitement des données, de leur récupération des déférentes sources et de leur consolidation et normalisation.
- **Un noyau du système :** contient les fonctionnalités propres au tableau de bord, telles que l'affichage des données, les synthèses et les graphes.

3.5 Diagrammes

Nous présentons dans cette section quelques diagrammes UML[4] importants, notamment le diagramme de séquence et le diagramme de cas d'utilisation.

3.5.1 Diagramme de Séquence

Nous allons détailler le diagramme de séquence d'un exemple exhaustif qui illustre tous les cas de figure possibles comportant une authentification à deux reprises, une demande de page accordée ou non avec personnalisation d'affichage.

[4] Unified Modeling Language (UML) est un langage de modélisation graphique conçu pour définir une méthode normalisée pour visualiser la conception d'un système. Il est couramment utilisé en développement logiciel et en conception orientée objet. Ces diagrammes sont conçus à l'aide de « visual paradigm » un outil de conception online https://online.visual-paradigm.com

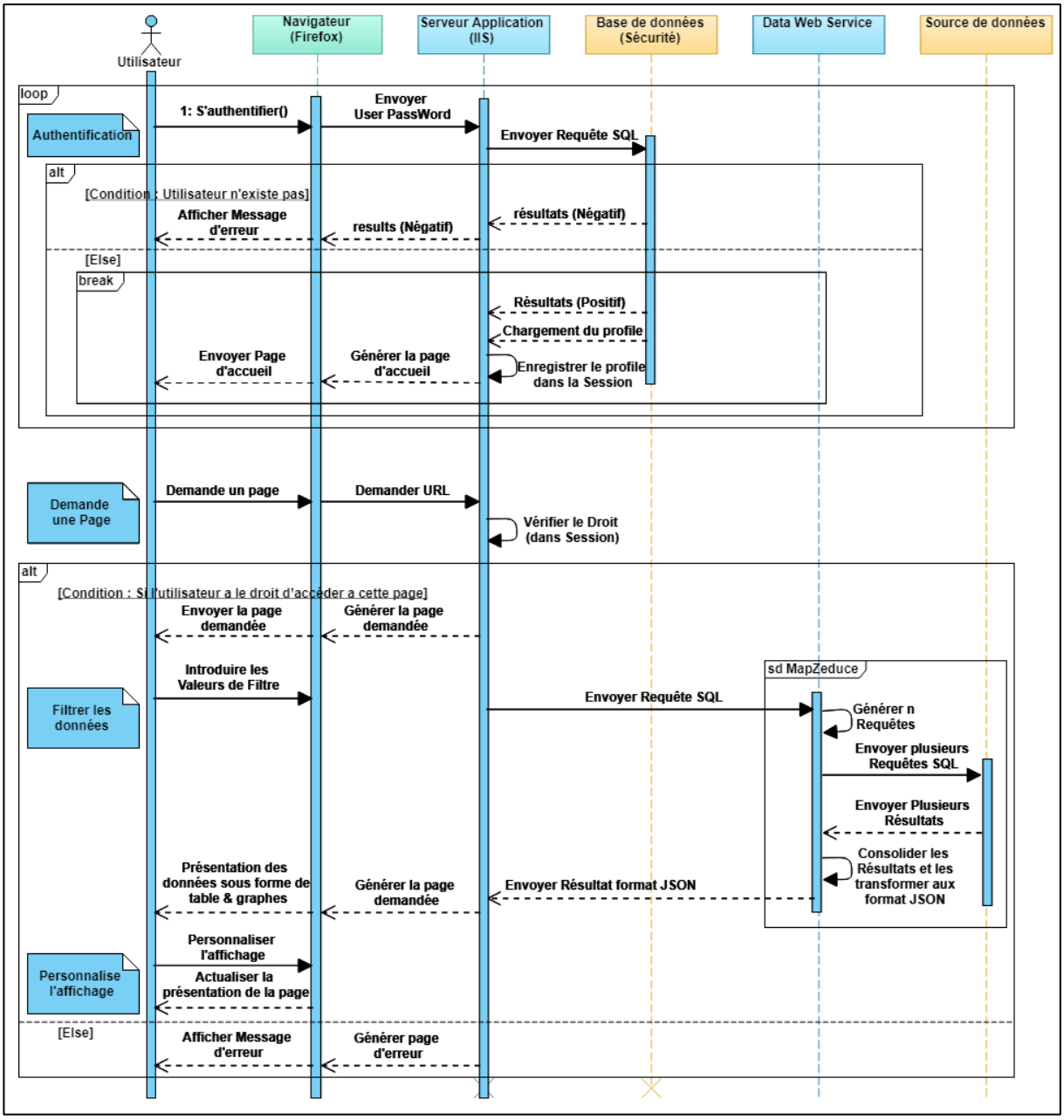

Figure 3-3 : Diagramme de Séquence

3.5.2 Diagramme de Cas d'utilisation

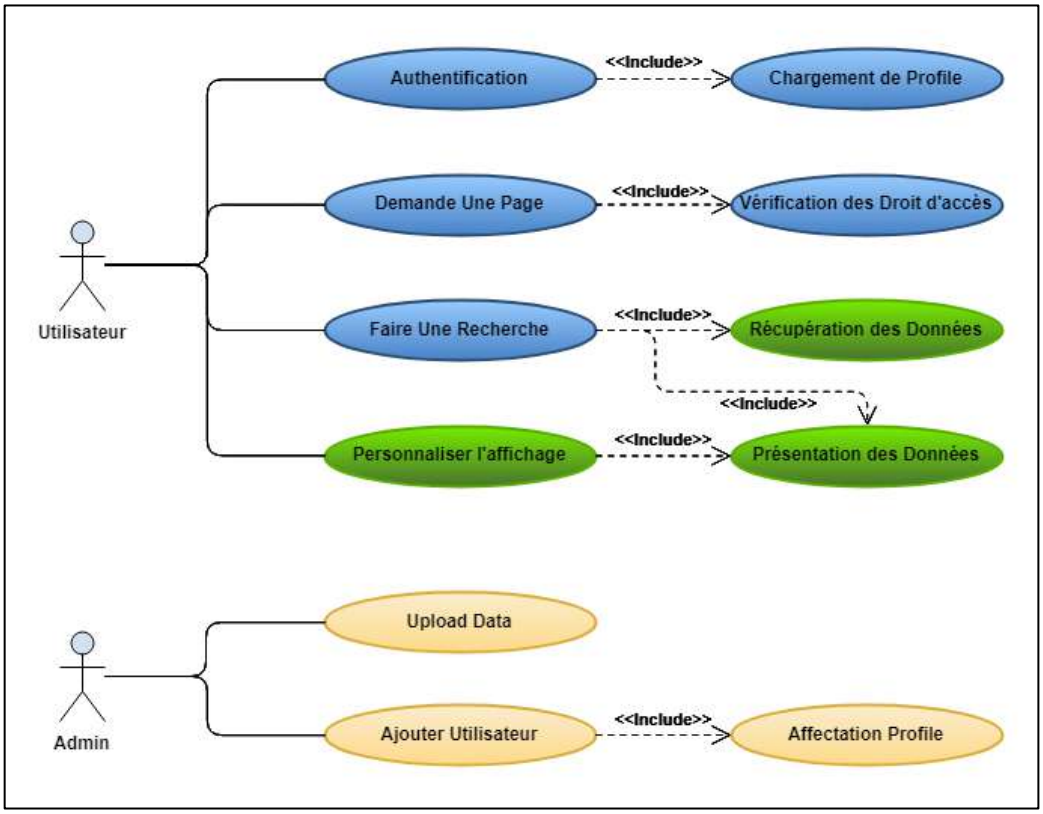

Figure 3-4 : Diagramme de cas d'utilisation

3.5.2.1 Identification des acteurs

Dans le cas de notre solution, nous distinguons deux acteurs : l'administrateur et l'utilisateur standard.

- **L'administrateur :** gère l'aspect sécurité (gestion des utilisateurs, définitions et affectation des profils) et le chargement des données.
- **L'utilisateur standard :** peut utiliser le tableau de bord pour visualiser les données, faire des synthèses et personnaliser les graphes, selon son profile accordé par l'administrateur.

3.6 Modules de la solution

Dans la conception de notre solution, nous avons opté pour une programmation modulaire. Cette dernière reprend l'idée de développer une application en la décomposant en

modules pour pouvoir les développer indépendamment. Ce style de programmation permet l'organisation du code source en unités de travail logiques, facilite la maintenance et l'amélioration progressive, ainsi que la réutilisabilité et le partage du code.

3.6.1 Le module Sécurité

Afin d'alléger la base de production, et pour mieux maitriser le côté sécurité et la gestion des profils, le volet sécurité est délégué à une application tierce avec sa propre base de données appelée « applications_settings ». Le diagramme de classe de ce composant se présente comme suit :

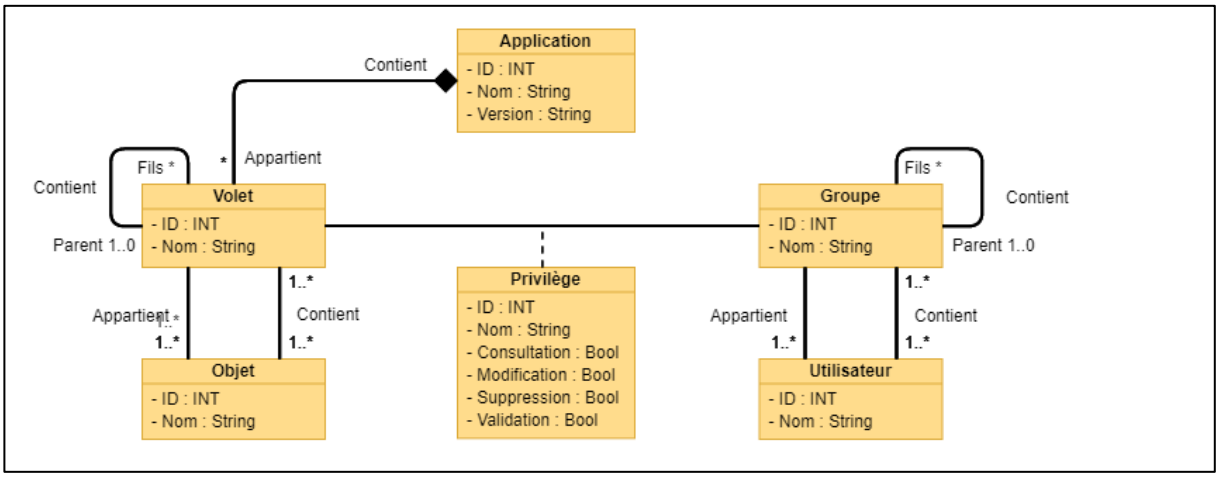

Figure 3-5 : Digramme de classe du composant sécurité

L'externalisation de l'aspect sécurité par un composant dédié permet la gestion de celui-ci pour d'autres solutions d'une manière unifiée (voir la figure ci-dessous).

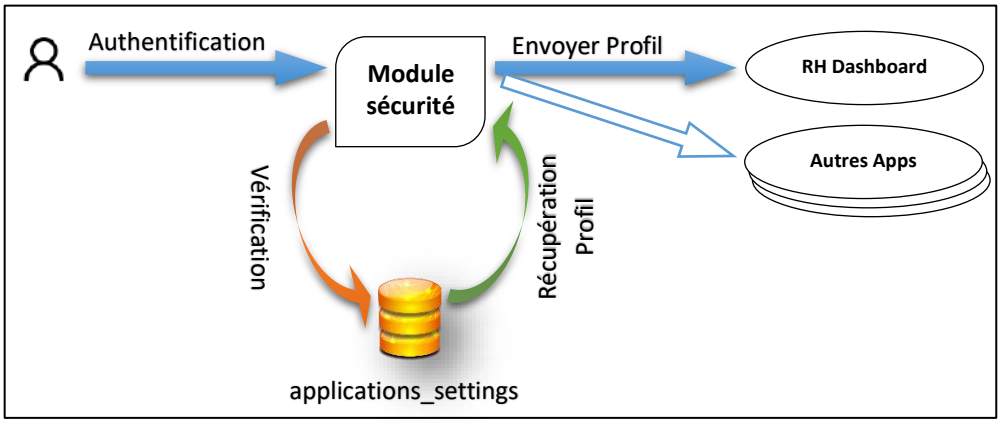

Figure 3-6 : Fonctionnement du module Sécurité

3.6.2 Le module Data

Dans l'idée de la décomposition de la solution en déléguant les services (tâches) a des composants dédiés, un module spécial pour la récupération des données et implémenté.

Le rôle principal de ce composant est de :

- Démocratiser l'accès aux données.
- Unifier la façon d'accéder aux déférentes sources.
- Optimiser et perfectionner la récupération des données.
- Unifier le format des données récupérées.

Pour répondre à ces exigences nous avons opté pour une solution basée sur les Services Web[5] qui garantit la démocratisation et l'unification d'accès, avec une architecture en couche (voir la figure ci-dessous). La Couche modèle du composant s'occupe de la récupération des données avec la possibilité de passer les requêtes aux sources d'une manière standard, ou en utilisant la méthode MarZeduce (qui sera détaillée ci-dessous) afin d'optimiser le temps d'exécution, tandis que la couche métier prend en charge la transformation des données résultat au format JSON[6] afin de répondre à la quatrième et dernière exigence définit ci-dessus.

[5] Un service web est un protocole d'interface basé sur le HTTP permettant la communication et l'échange de données entre applications et systèmes hétérogènes

[6] JSON : JavaScript Object Notation est un format de données textuelles, tous comme XML, il permet de représenter de l'information structurée, JSON est largement utilisé dans les solutions web.

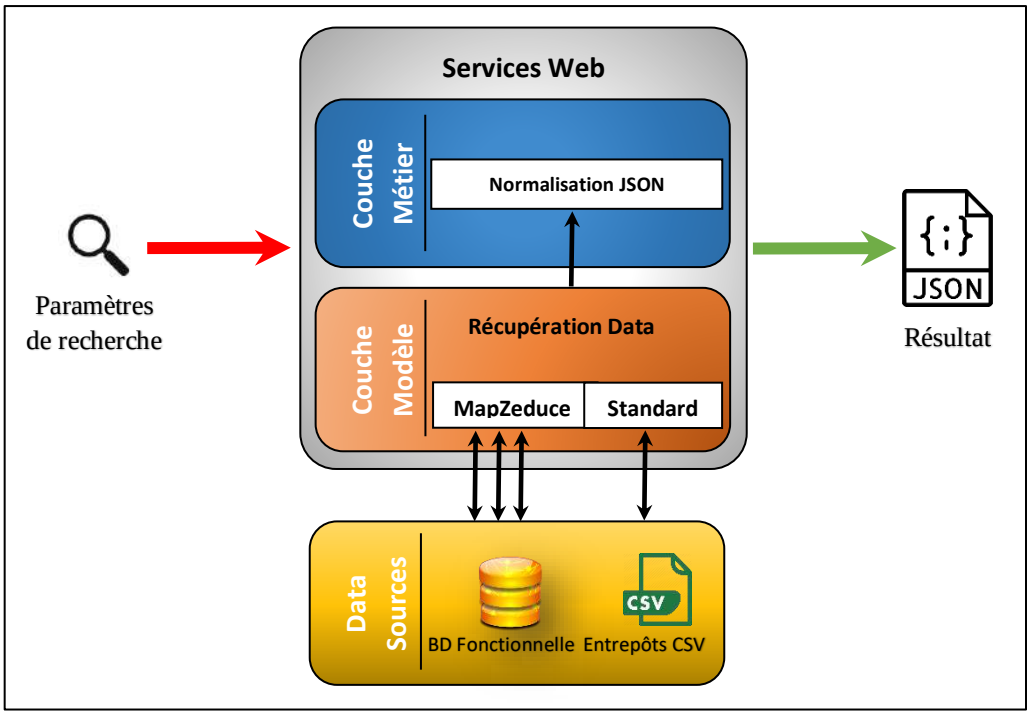

Figure 3-7 : Architecture du Data Web Service

3.6.2.1 MapZeduce

Afin d'accélérer la recherche et la récupération des données, toute en respectant les standards Big Data, nous avons implémenté notre algorithme *MapReduce*, nommé MapZeduce. Le principe de MapZeduce est de fragmenter les requêtes en sous requêtes qui peuvent être exécutées en parallèle d'une manière asynchrone. La décomposition est faite selon le Mois pour trois raisons :

1. L'information du Mois est concluante, on peut savoir s'il s'agit de l'archive ou de la base en cours d'exploitation.

2. La décomposition sera équilibrée, car on a pratiquement la même quantité de données entre les mois.

3. La recherche selon la colonne 'Mois' est rapide, grâce à l'indexation dans SQL server, ou même pour les fichiers CSV.

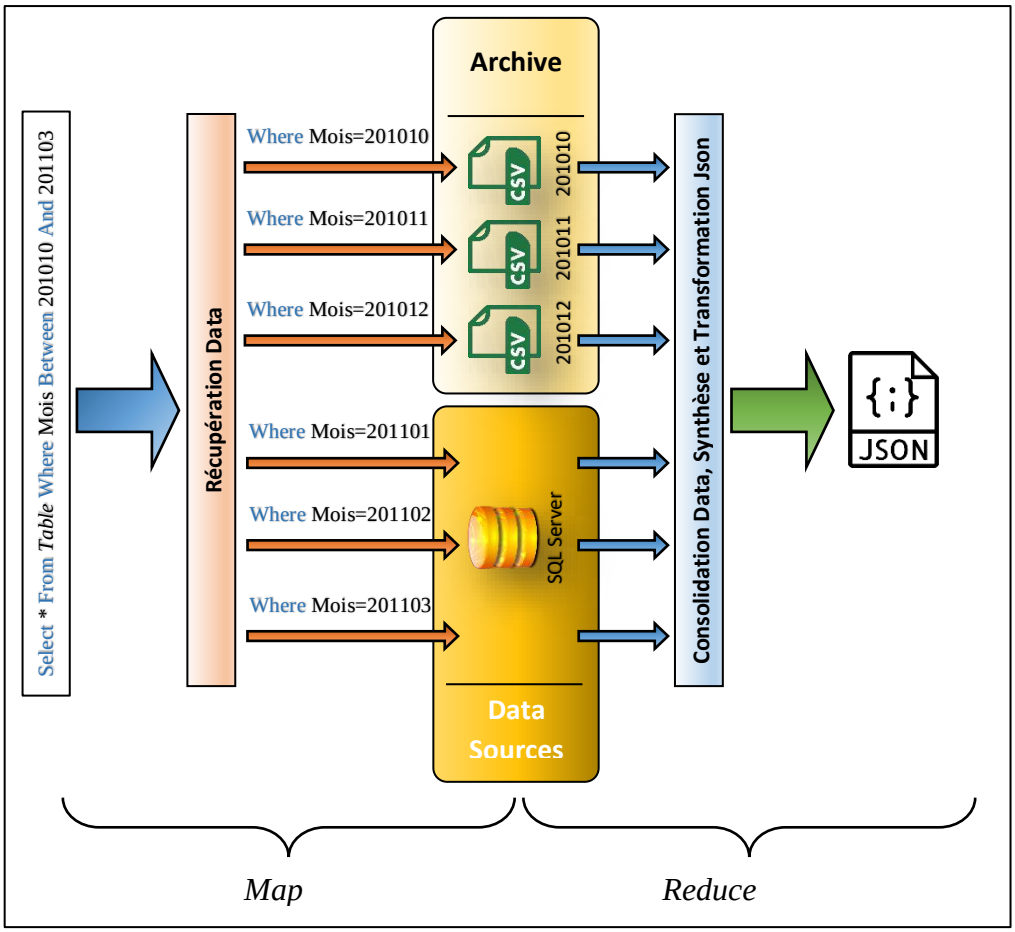

Figure 3-8 : Exemple d'exécution d'une requête avec *MapZeduce*

3.6.2.2 Normalisation JSON

Pour ce qui concerne le format de récupération des données nous avons utilisé le format JSON car il est très répondu dans les solutions Web et Mobile, dans notre cas les données JSON sont utilisées (voir la figure ci-dessous) à la fois par :

- **Les tables HTML :** pour afficher les données brutes.
- **Les tables récapitulatives :** pour afficher les données synthèses.
- **Les graphes**.

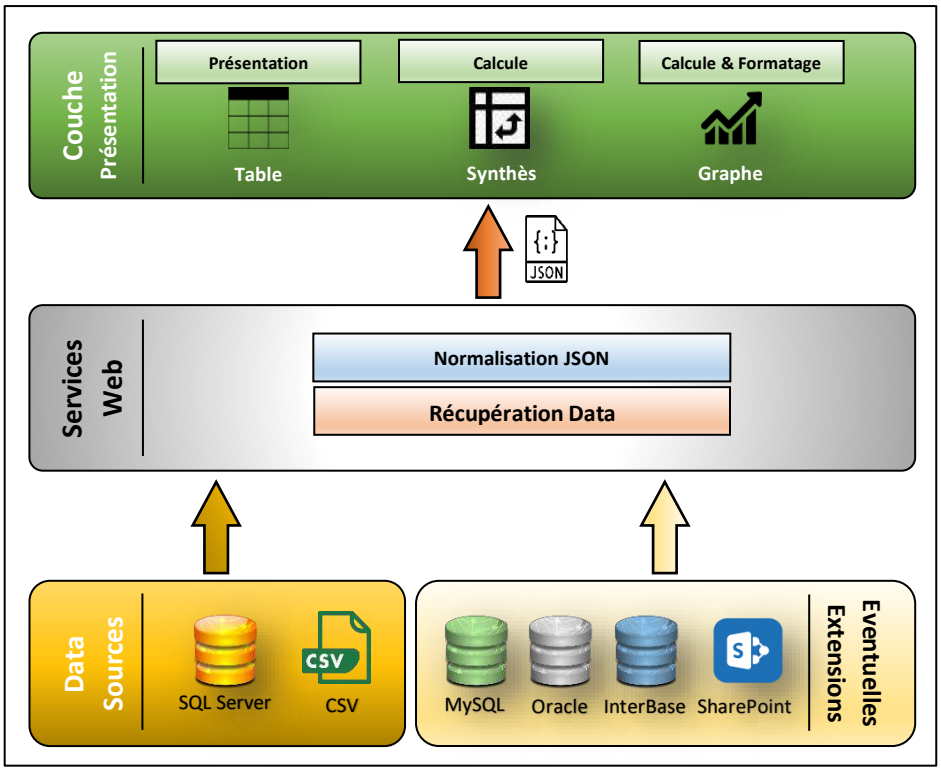

Figure 3-9 : Interfaçage et fonctionnement du Data Web Service

3.7 Conclusion

A l'issus de cette étape, nous avons posé notre problématique consistant à effectuer une analyse étalée sur plusieurs systèmes hétérogènes. La réponse que nous proposons est une solution modulaire, où chaque ensemble de tâches, logiquement liées, est assuré par un composant spécifique.

Les principaux modules Sécurité et Data, sont détaillés davantage en citant leurs rôles et objectifs. De plus, nous avons schématisé leur architecture, fonctionnement et l'interface avec les autres composants.

Dans notre conception, nous avons suivi le modèle 'MVC' qui consiste à découper le système en couches : Modèle, Métier et présentation. Afin de clarifier notre travail nous avons introduit les principaux diagrammes d'UML, notamment le Digramme de classe, de séquence et de cas d'utilisation.

Enfin, et après avoir exprimé les objectifs attendus du futur système à concevoir, son architecture ainsi que ses différents modules, nous pouvons entamer l'implémentation et la réalisation de notre solution.

Chapitre 4

Implémentation

4.1 Introduction

A ce stade, la problématique posée a été analysée et l'architecture conceptuelle globale du système ainsi que ses modules (sécurité et Data) définis. Nous pouvons dès lors entreprendre la dernière partie qui est l'implémentation. L'objectif de la phase d'implémentation est de fournir un produit final, exploitable qui répond au besoin et à la problématique exprimée.

Dans ce chapitre nous allons présenter brièvement l'entreprise objet de notre étude, comme nous illustrerons l'architecture physique requise par notre système, l'environnement de développement, les technologies et les bibliothèques que nous avons utilisées.

Ensuite, nous dévoilerons, à l'aide de captures d'écrans, la structure générale de l'interface utilisateur, le menu et l'arborescence de la solution, ainsi que les déférents filtres développés.

4.2 Présentation ENSP

L'Entreprise Nationale de Services aux Puits « ENSP Group » est un groupe intégré, présent sur la chaîne para pétrolière qui dispose d'un portefeuille d'activités diversifié et emploie plus de 4 000 agents versés dans ses métiers de base. L'Entreprise mère « ENSP » est une société par actions au capital de 8 000 000 000 DA, filiale à 100 % Sonatrach.

Le groupe ENSP a pour mission de répondre en priorité aux besoins sur le long terme de Sonatrach et de ses associés étrangers. Sa finalité est de créer de la valeur par l'exercice performant de ses métiers en système concurrentiel. Pour atteindre cet objectif, les ressources humaines constituent le levier principal de nos succès et le partenariat une option stratégique.

Le groupe capitalise une expérience de plus de 40 ans dans les services para pétroliers sur les différents champs pétroliers et gaziers algériens. Aujourd'hui, ses équipes sont implantées dans les champs de : Hassi Messaoud, Hassi R'mel, In Amenas et Ourhoud.

L'ENSP est certifiée ISO 9001 version 2015 pour la qualité, ISO 14001 version 2015 pour l'environnement et ISO 45001 version 2018 pour la santé et sécurité au travail

4.3 Plateforme et Outils de la solution

A l'instar de beaucoup d'autres applications, notre solution nécessite une plateforme d'exécution, ainsi son développement se base sur plusieurs solutions et outils divers, la figure ci-dessous liste l'ensemble de ces outils en les regroupant selon la couche d'appartenance.

Figure 4-1 : Plateforme et outils de la solution

4.3.1 Sources de données

A titre de validation de notre proposition, deux sources de données de types différents ont été utilisées :

1. **Une base de données SQL Server 2014 :** qui contient les données du système de gestion de la paie actuel « WinStar ». Notons seulement qu'une phase de préparation de données est nécessaire car les données sont à l'origine sous Windev HyperFile[7]. Cette phase de préparation sera détaillée dans la section Expérimentation.

2. **Des fichiers CSV :** afin d'étaler l'analyse sur une longue durée, l'historique généré par les systèmes de gestion de la paie antérieur est exporté sous le format CSV afin d'être utilisé.

Néanmoins, et vu l'architecture en couches, l'extension éventuelle des sources de données (notamment Oracle ou SharePoint) sera transparente et requerra un minimum de modification dans le code source.

4.3.2 Couche modèle

L'accès aux différentes sources de données se fait de deux manières :

1. **Accès natif :** en utilisant le composant data du Framework .Net (ADO.Net), l'accès aux différents types de source de données est désormais possible.

2. **Via Service Web :** pour gagner davantage de flexibilité, nous avons introduit une couche intermédiaire entre le code Behind et les sources de données. Cette couche est implémentée par les Services Web. Ce composant est détaillé dans la section Data Web Service.

4.3.3 Couche métier

Cette couche est interprétée par des ASP.Net avec un code Behind C#.

1. **Microsoft Visual Studio 2017 :** est l'IDE (Integrated Development Environment) publié par Microsoft regroupant un ensemble d'outils qui facilitent le développent

[7] WinDev est un atelier de génie logiciel édité par la société française PC SOFT. Il propose son propre langage : le WLangage, et son propre SGBD nommé HyperFile, La première version de l'AGL est sortie en 1993.

d'applications.VS.Net[8] possède une large gamme de type de projets (de la console au Mobile et au Cloud). Pour ce qui concerne les applications ASP.Net il couvre la totalité du processus de développement.

4.3.4 Couche présentation

1. **jQuery :** il s'agit d'une bibliothèque JavaScript open Source côté client, simplifiant ainsi le développement d'applications Web 2.0 (y compris Ajax). Actuellement, jQuery est utilisé par plus de la moitié des principaux sites Web.

2. **Bootstrap :** c'est une bibliothèque de développement web frontale, open source. Bootstrap se compose du HTML, CSS et JavaScript (JS) pour faciliter le développement de sites et d'applications responsives (réactives et orienté mobile).

3. **Underscore.js :** est une bibliothèque JavaScript qui fournit plus de 100 fonctions utilitaires (*map*, *reduce*, *filter*, etc.), ainsi que des gadgets plus spécialisés (liaison de fonction, création de modèles *javascript*, création d'index rapides, etc.). Parmi ses caractéristiques se trouve l'exécution asynchrone des boucles.

4. **Highcharts :** est une bibliothèque JavaScript pour la création de graphes, basée sur SVG. Highcharts prend en charge la gestion des évènements, ce qui rend ses graphes interactifs coté client et marque sa spécificité par rapport au composant Chart Server-Slide.

    - Notre travail consiste à implémenter les fonctions de préparation des données (à l'aide de Underscore) et les formater pour être exploitable par Highcharts, ainsi que la création d'une bibliothèque de modèles de graphes pour faciliter leur utilisation.

5. **PivotTable.js :** est une bibliothèque JavaScript de génération des tableaux croisés dynamiques.

    - Notre participation se résume à fournir un assistant graphique de paramétrage et de personnalisation de la génération des tableaux croisées dynamiques.

[8] Supporté même par le système mac OS

4.3.5 Plateforme de déploiement

Comme notre solution est orienté Microsoft, la plateforme se compose de :

1. **Microsoft SQL Server :** est un système de gestion de base de données (SGBD) développé par Microsoft multi plateformes[9], SQL Server offre une meilleure intégration avec ses outils de développement[10].

2. **IIS 7.5 :** Internet Information Services est le serveur Web par défaut des systèmes Windows Servers, conçu essentiellement pour héberger les applications Active Server Page (ASP)[11].

3. **Navigateur :** comme il s'agit d'une application Web, théoriquement n'importe quel navigateur peut être utilisé, mais pour des raisons de compatibilité avec certaines bibliothèques JavaScript utilisées nous recommandons Mozilla Firefox[12].

4.4 Interface de l'application

L'interface graphique désigne la manière dont est présenté un logiciel aux utilisateurs. S'agissant d'une application Web dans notre cas, l'interface fait référence à la structure des pages et le positionnement des différents éléments : menus, Body, footer, et les autres fonctionnalités. Une interface graphique bien conçue est ergonomique et intuitive afin que l'utilisateur la comprenne tout de suite.

Dans cette section nous présentons l'interface de notre système.

4.4.1 Structure des pages

En utilisant la notion de *MasterPage* dans les projets ASP.Net, la structuration de vos pages deviendra plus facile, et elle nous donne l'impression de travailler dans la même page bien que le contenu change partiellement.

---

[9] Microsoft SQL Server fonctionne sous linux à partir de la version 2016.
[10] La versions SQL Server 2016 (et ultérieur) dotée de fonctions de traitement *Json* intégrées, ce qui permet de combiner le *NoSQL* et le relationnel dans la même requête.
[11] IIS Support les applications PHP à partir de la version 6.
[12] Mozilla Firefox est un navigateur web libre, développé par la communauté open source et piloter par la Mozilla Foundation

Dans notre solution toutes les pages ont la même structure (Titre, Filtre, Zone D'affichage), dont la zone d'affichage peut contenir des Données, Synthèse ou bien des graphes sous forme des onglets (voir la figure ci-dessous).

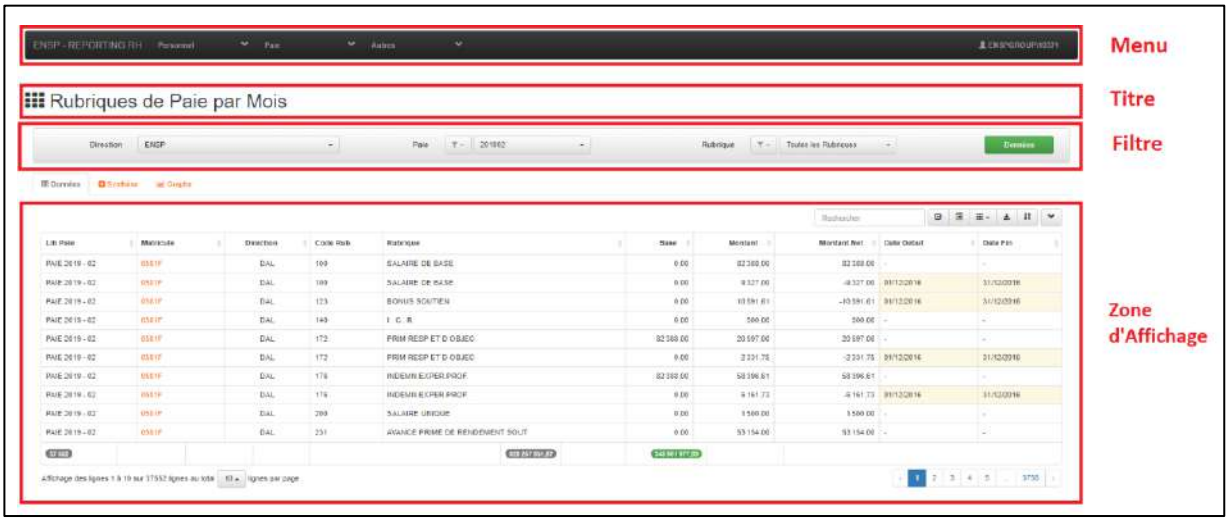

Figure 4-2 : Screenshot - Structure des pages de l'application

4.4.2 Zone d'affichage

C'est la zone de travail. Elle contient 3 modes :

- Mode données brutes
- Mode Synthèse
- Mode Graphe

Le basculement entre ces 3 modes se fait à l'aide d'un contrôle Tab (voir la figure ci-dessous) qui est situé au sommet de ladite zone d'affichage.

Figure 4-3 : Screenshot - Tab de Modes d'affichage

4.4.2.1 Mode Données

Ce mode permet d'afficher les données dans une table HTML enrichi[13], avec une barre d'outils donnant à l'utilisateur la possibilité de :

- Masquer / Afficher des colonnes
- Faire un tri avancé
- Faire une recherche rapide
- Exporter les données format XLS
- Masquer / Afficher la pagination

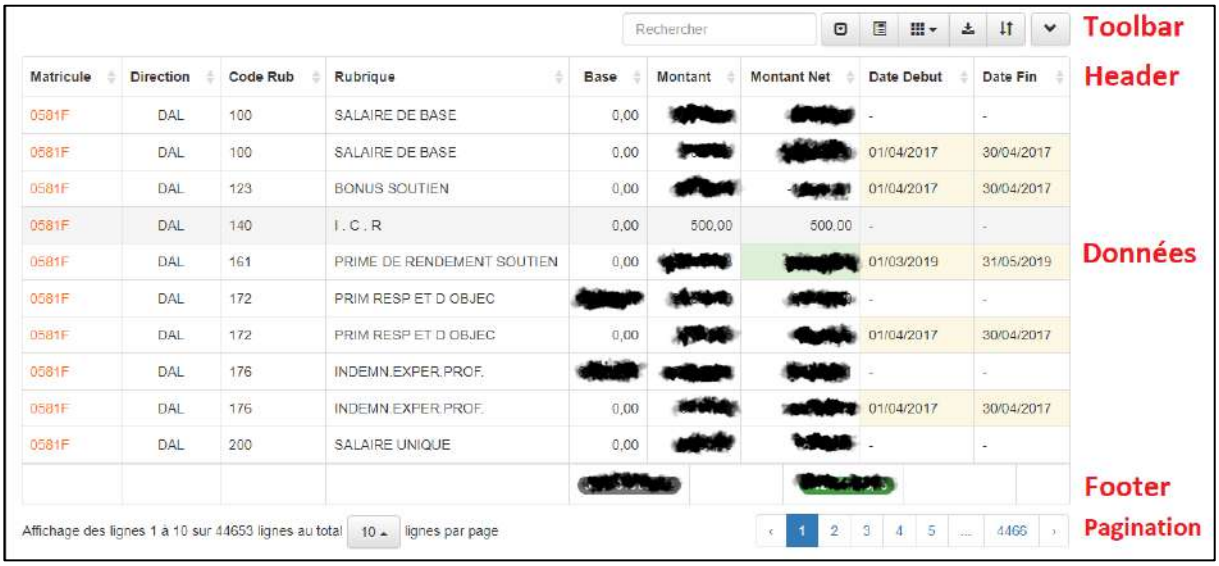

Figure 4-4 : Screenshot - Tableau d'affichage des données

4.4.2.2 Mode Synthèse

Dans ce mode nous avons la synthèse des données sous forme d'un tableau croisé avec la possibilité de l'exporter vers Excel.

[13] Nous avons utilisé le contrôle Bootstrap Table, https://bootstrap-table.com

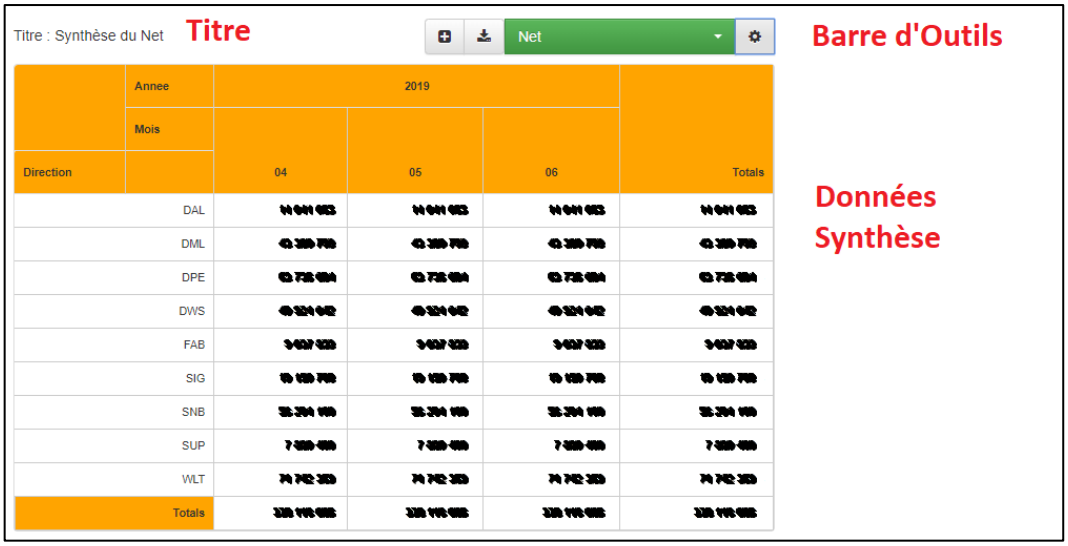

Figure 4-5 : Screenshot - tableau de synthèse des données

Pour chaque page nous fournissons un ensemble de modèles de synthèse prédéfinis avec la possibilité de les personnaliser (voir la figure Modèles de synthèse).

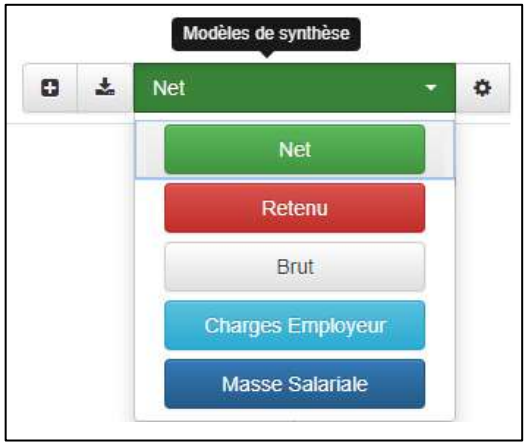

Figure 4-6 : Screenshot - Modèles de synthèse

La personnalisation de la synthèse permet à l'utilisateur de spécifier les lignes du tableau croisé, les colonnes, ainsi que la fonction d'agrégat à travers une interface graphique intuitive que nous avons-nous même développé (voir la figure ci-dessous).

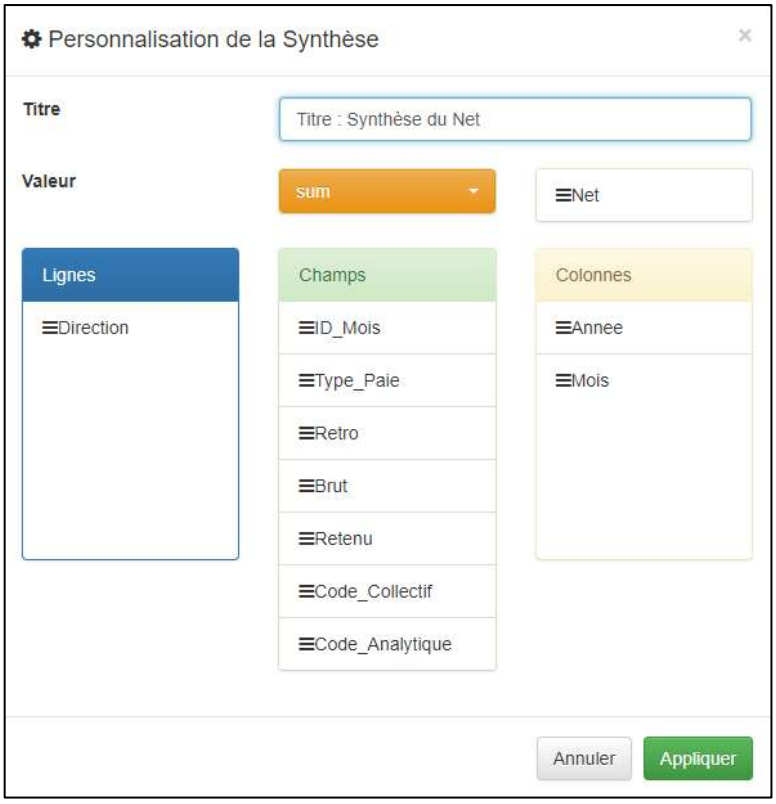

Figure 4-7 : Screenshot - Assistant de Personnalisation de la synthèse

4.4.2.3 Mode Graphes

Dans le mode graphes nous présentons la synthèse des données sous forme de graphe afin de faciliter leur lecture avec la possibilité d'exporter ces graphes en format Image.

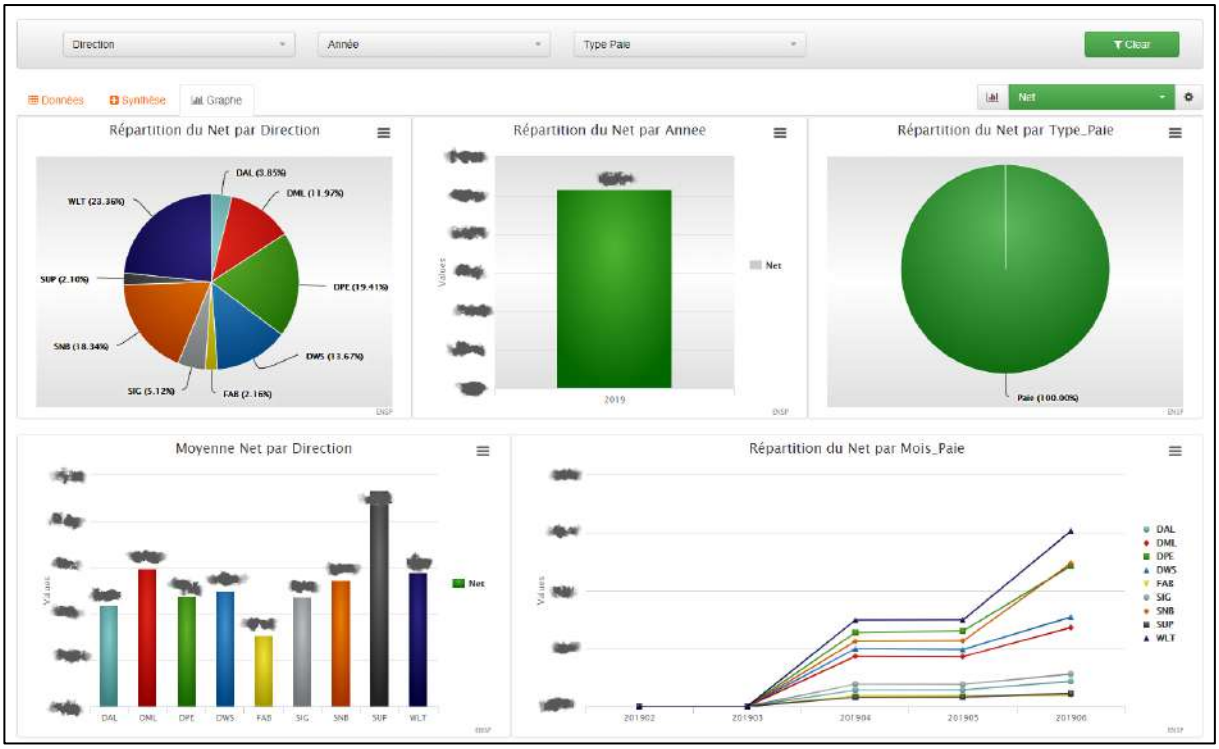

Figure 4-8 : Screenshot - exemple de Mode Graphe

Pour chaque page nous fournissons un ensemble de modèles de graphes préalablement définis et assez exhaustif (voir la figure Modèles de graphes).

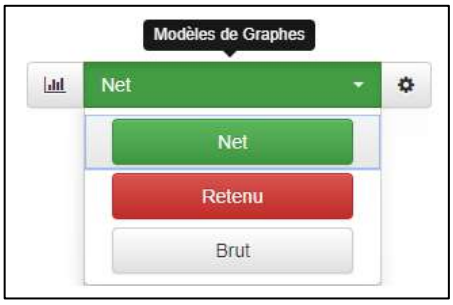

Figure 4-9 : Screenshot - Modèles de Graphes

Afin de permettre aux utilisateurs une meilleure expérience d'analyse, nous fournissons des filtres de données. Une fois l'utilisateur sélectionne une partie des données, tous les graphes seront redessinés avec la nouvelle sélection de données. Le filtrage et le réaffichage se font en mode déconnecté sans faire appels aux serveurs.

Figure 4-10 : Screenshot - Barre des filtres des données de Graphes (offline)

Il existe deux façons de filtrer :

- A partir de la barre de filtrage.
- Directement depuis les graphes de type Filtre. Afin de distinguer les graphes de type filtre, nous les avons présenté dans un arrière-plan de couleur grise (voir la figure ci-dessous).

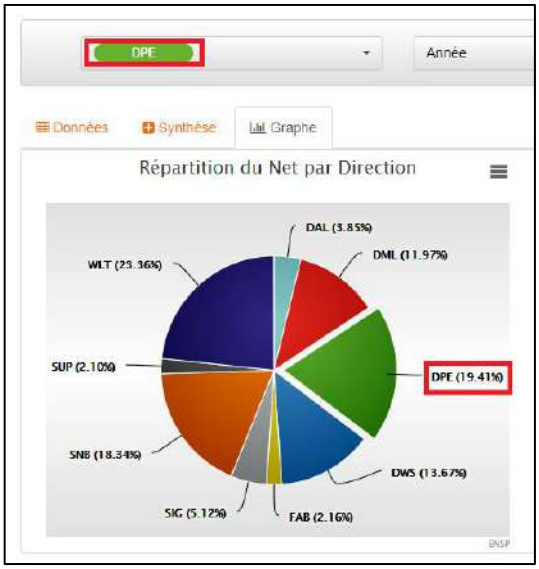

Figure 4-11 - Screenshot - Les Graphes type Filtre

4.4.3 Menu

Nous avons opté pour un menu horizontal afin de gagner davantage d'espace pour le contenu. Ce menu contient trois volets :

- Volet personnel,
- Volet paie,

- Volet divers pour regrouper le reste des pages (formation, prêt …) qui par nature ne rentrent pas dans les deux premiers volets.

4.4.3.1 Volet Personnel

Ce volet regroupe les pages qui traitent les données du personnel, les changements de situation dans une période, ainsi qu'un récapitulatif mensuel.

Figure 4-12 : Menu - Volet Personnel

4.4.3.2 Volet paie

Le volet paie contient les pages qui manipulent les données de la paie, les différentes rubriques de la paie auxquelles s'ajoute le calcul de la masse salariale.

Figure 4-13 : Menu - Volet Paie

4.4.3.3 Volet divers

Additivement aux données du personnel et de la paie, nous avons d'autres pages à l'instar du suivi des remboursements des prêts, le suivi des apprentis et les cotisations. Toutes ces pages sont regroupées dans un volet nommé **Divers**.

Figure 4-14 : Menu - Volet Divers

4.4.4 Filtres

Chaque page dispose d'une barre des filtres pour spécifier les paramètres de chargements des données. Généralement chaque page contient trois types de filtre (contrôles) plus le bouton de chargement des données (voir la figure ci-dessous). Afin d'accélérer la récupération des données, la recherche se fait sur des colonnes indexées.

Figure 4-15 : Barre des filtres

Nous avons développé pour chaque filtre son propre contrôle Web. Cette approche permet d'alléger les pages, de réutiliser à volonté les contrôles et facilite leur maintenance. Parmi ces contrôles filtres nous citons :

4.4.4.1 Filtre par direction

Ce contrôle permet de sélectionner les données d'une ou plusieurs directions, ou bien toute l'ENSP.

Figure 4-16 : Filtre - Direction

4.4.4.2 Filtre par Employer

Ce contrôle permet de sélectionner un ou plusieurs employés, soit parmi l'effectif actif ou les départ (retraite, démission…).

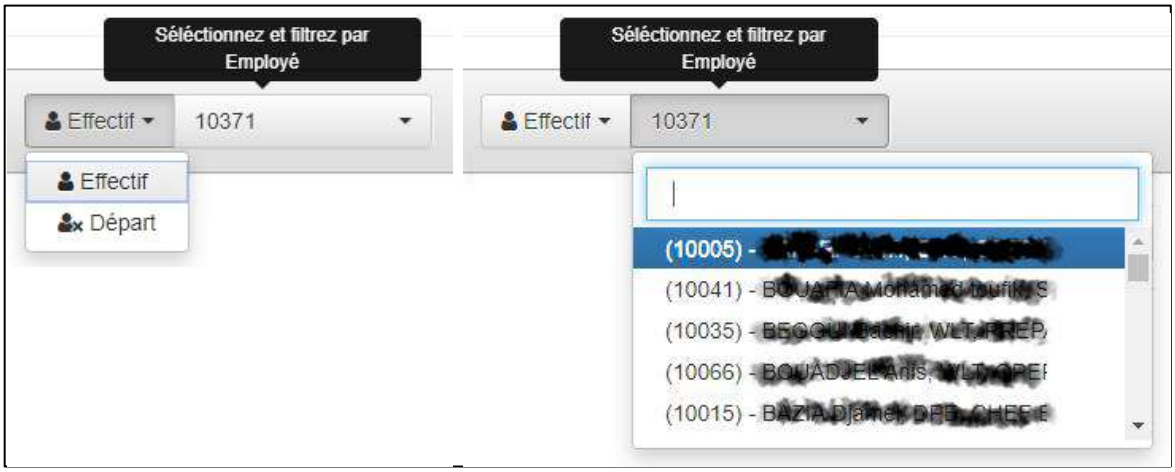

Figure 4-17 : Filtre - Employer

4.4.4.3 Filtre par Mois / Période

Pour spécifier un mois ou un intervalle de temps, un contrôle a été créé avec la possibilité de switcher entre (Mois <> Période)

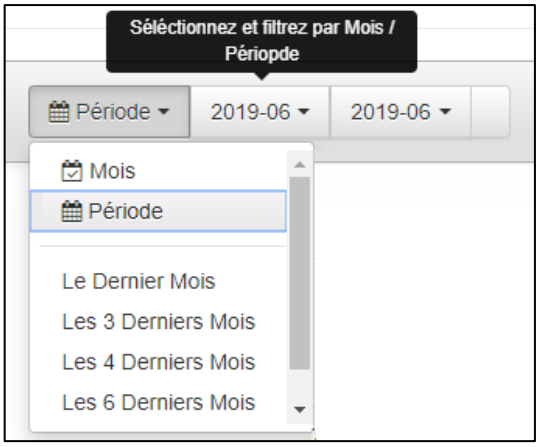

Figure 4-18 : Filtre Mois / Période

4.4.4.4 Filtre par type paie

Pour spécifier la nature de paie (normal, rappel ou Prime)

Figure 4-19 : Filtre - Type de paie

4.4.4.5 Filtre par paie

Le contrôle ‘Filtre par paie’ nous donne la possibilité de sélectionner une ou plusieurs paies selon leur type.

Figure 4-20 : Filtre - Paie

4.4.4.6 Filtre par rubrique

Le Filtre des rubriques sert à récupérer les données concernant une ou plusieurs rubriques de paie selon leur nature ou sens.

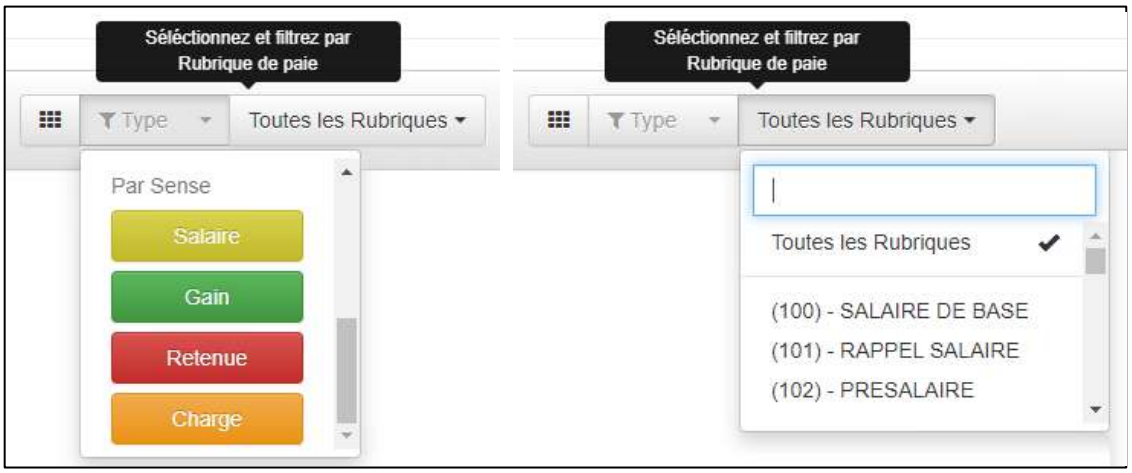

Figure 4-21 : Filtre - Rubrique de paie

4.5 Interface Data Web Service

Le module Data, comme tout service Web permet son exécution depuis n'importe quel browser. La figure ci-dessous liste les opérations (méthodes) que notre service fournit[14].

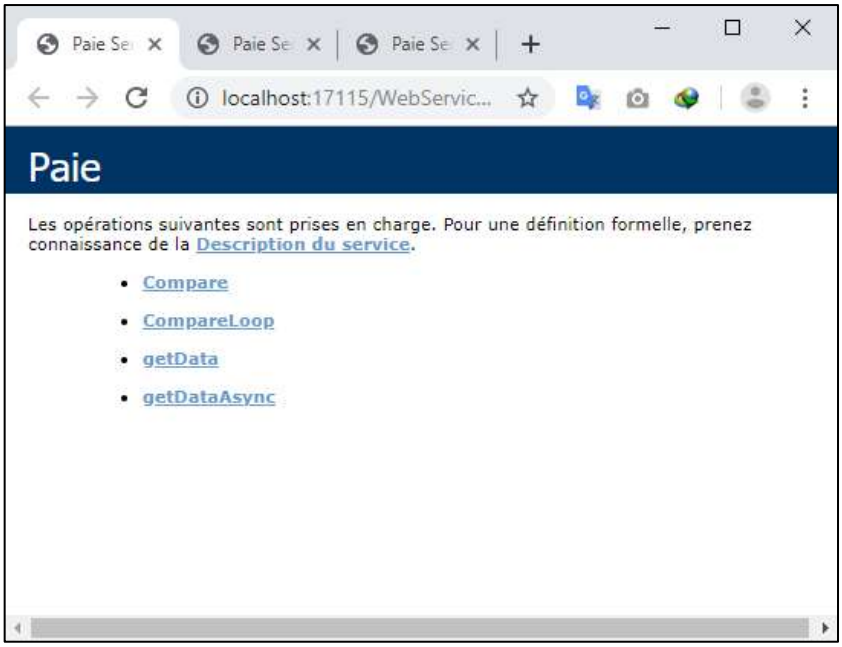

Figure 4-22 : Data Web Service - Description du Service Paie

Notre service permet la récupération des données de deux manières différentes :

- Standard : via la méthode getData.
- MapZeduce : avec la méthode getDataAsync.

[14] Nous avons créé les deux méthodes Compare, CompareLoop pour étudier l'impact d'utilisation de *mapZeduce* sur le temps d'exécution.

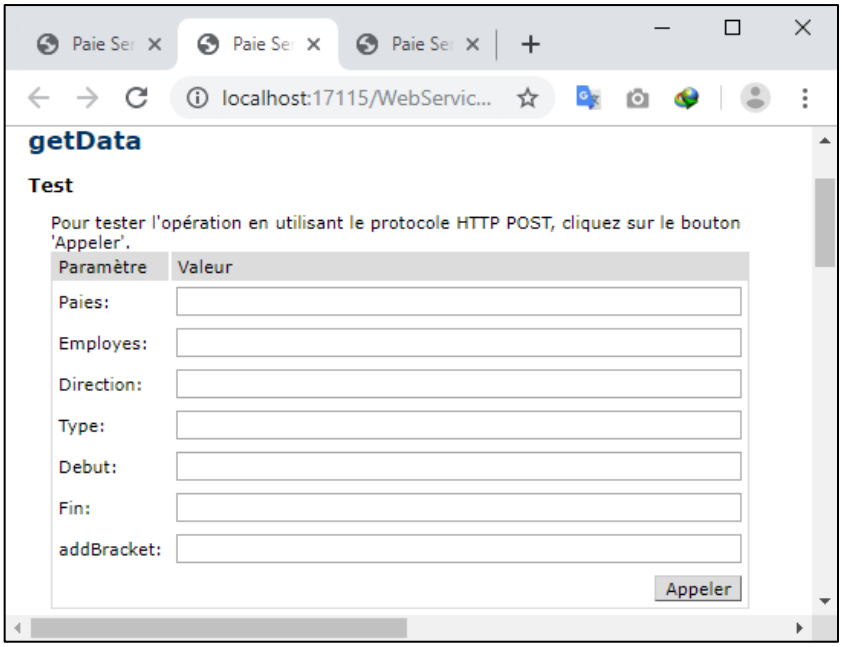

Figure 4-23 : Data Web Service - Interface de test getData (Execution Standard)

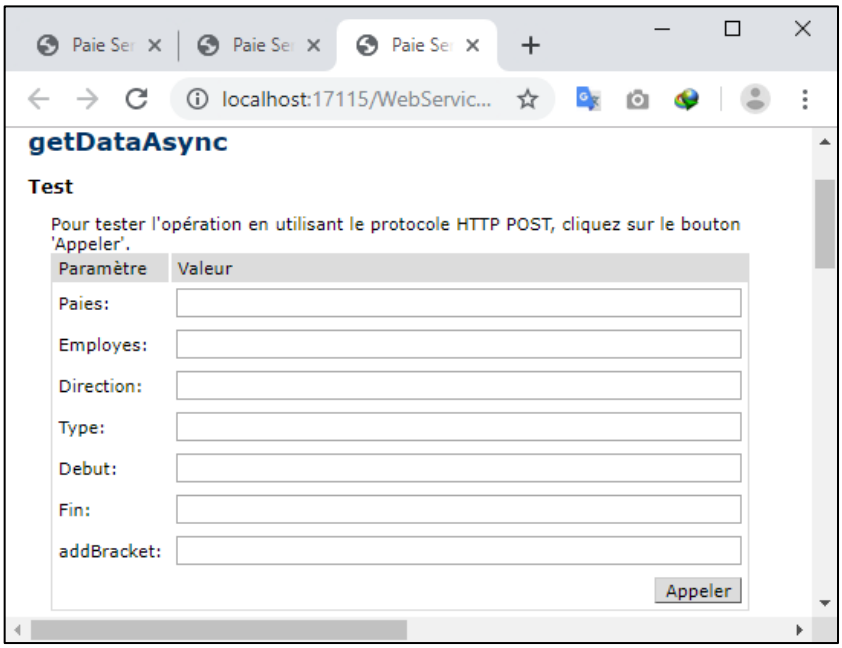

Figure 4-24 : Data Web Service - Interface de test getDataAsync (Execution MapZeduce)

4.6 Expérimentation

Durant la réalisation de notre travail, nous avons fait face à des défis d'ordre conceptuel (comme les Requêtes récursives) et technique. Nous avons dû notamment réaliser le prétraitement des données (comme Data préparation).

Afin de partager notre expérience et les enseignements tirés lors de la réalisation de notre travail, nous citerons ces points de manière détaillée dans cette section.

4.6.1 Data Préparation

Le logiciel de gestion de la paie WinStar utilisé au sein de l'entreprise objet de notre étude a été développé avec WinDev 5.5 (année 1998) et se base sur le SGBD HyperFile. Cette version d'HyperFile est non seulement mono poste mais elle ne supporte pas les requêtes SQL. De plus, par lacune conceptuelle, WinStar stocke les données de chaque mois dans une table séparée. Il est donc évident qu'il s'agit d'une collection de fichiers plats plutôt qu'une base de données.

Au vu de toutes ces limites et l'évolution des besoins en matière de traitement des données, nous étions obligés de passer vers un SGBD fiable. Etant donné que les solutions de l'entreprise sont orientées Microsoft, le choix le plus adapté a été SQL Server.

Le passage de HyperFile 5.5 vers SQL Server 2014 ne peut pas se faire d'une manière directe, nous devons passer par des étapes intermédiaires. La figure ci-dessous illustre ces étapes.

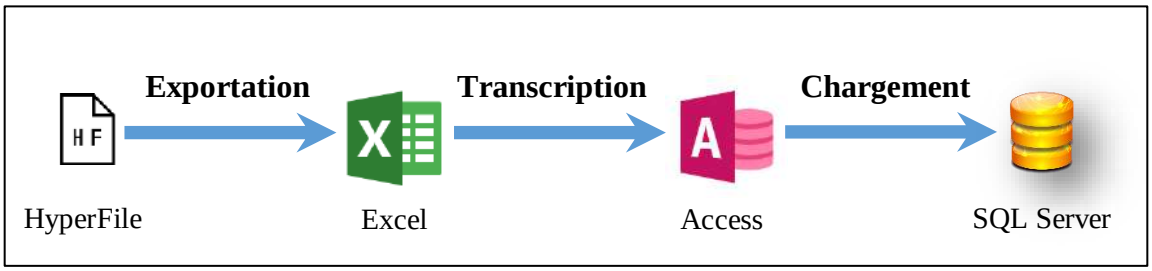

Figure 4-25 : Les phases de Data Préparation

Les étapes de 'Data préparation' sont :

- **Exportation :** exporter les données HyperFile vers un fichier Excel.
- **Transcription :** copier le contenu du Fichier Excel vers Access. Nous sommes passés par MS Access pour deux raisons :
  - Vérification des contraintes de format et de langueur des données (Excel ne le permet pas).
  - Faciliter le chargement des données vers SQL Server en utilisant les tables liées.
- **Chargement :** le chargement des données de MS Access vers SQL Server.

4.6.2 Le Module Sécurité - Aspects techniques

- **Authentification :** le module de sécurité supporte une authentification mixte
  - SQL standard : utilisateur / Mot de passe,
  - Windows : Via Active Directory.
- **Relations Récursives et requêtes CTE :** afin d'assurer une arborescence avec un nombre de niveaux illimité, requise dans l'implémentation de la hiérarchie des groupes et des volets (voir la figue de schéma de la base ci-dessous), nous avons utilisé les relations récursives et les requêtes CTE (Common Table Expression). Cependant, comme l'exécution de ce type de requêtes est lourde en termes de temps d'exécution nous limitons son utilisation à une seule fois après le login.

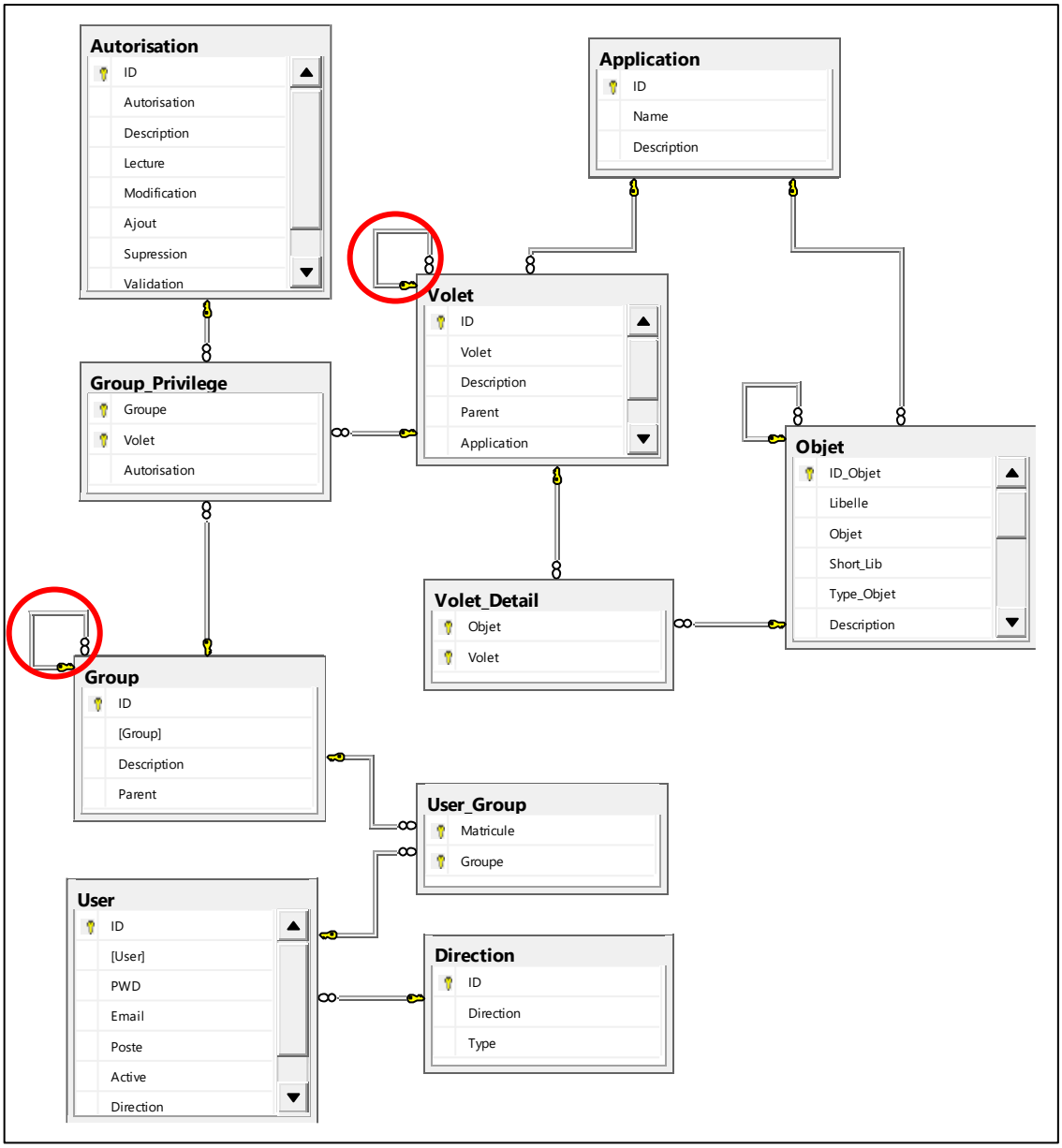

Figure 4-26 : Schéma de la base de données "applications_settings"

4.6.3 Data Web Service - L'impact de MapZeduce

Afin de savoir l'impact d'utilisation de MapZeduce sur le temps d'exécution, nous avons exécuté une série de requetés (soit 102 requêtes par déférentes paramètres) avec les deux méthodes standard et MapZeduce, ci-dessous un tableau contient un extrait de résultats collectés.

Tableau 4-1 : Extrait de comparatif entre Exécution standard et MapZeduce

Mois	NB Mois	NB Employés	Data Length (Chars)	Standard (Ms)	MapZeduce (Ms)	Gain (Ms)	Gain (%)
201101-201101	1	All	958 973	235	67	168	251%
201101-201106	6	All	5 792 677	519	251	268	107%
201101-201111	11	All	10 843 967	1 158	455	703	155%
201101-201204	16	All	16 541 104	1 118	828	290	35%
201101-201209	21	All	22 359 416	2 865	996	1 869	188%
201101-201302	26	All	28 331 109	3 212	1 187	2 025	171%
201101-201307	31	All	34 283 976	3 276	1 517	1 759	116%
201101-201312	36	All	40 354 601	4 388	2 052	2 336	114%
201101-201405	41	All	46 674 047	4 877	2 395	2 482	104%
201101-201508	56	All	65 897 580	7 097	3 196	3 901	122%
201101-201601	61	All	72 628 870	8 726	3 184	5 542	174%
201101-201606	66	All	79 398 031	12 133	6 803	5 330	78%
201101-201611	71	All	86 191 761	11 111	5 181	5 930	114%
201101-201704	76	All	93 107 528	12 097	4 870	7 227	148%
201101-201709	81	All	99 927 751	12 149	5 830	6 319	108%
201101-201802	86	All	106 753 547	13 370	6 441	6 929	108%
201101-201807	91	All	113 704 485	16 226	8 511	7 715	91%
201101-201812	96	All	120 790 652	14 940	6 277	8 663	138%
201101-201905	101	All	128 025 194	15 959	5 679	10 280	181%
201101-201101	1	1	440	414	5	409	8180%
201101-201111	11	1	4 835	59	101	-42	-42%
201101-201209	21	1	9 237	32	116	-84	-72%
201101-201307	31	1	13 650	22	150	-128	-85%
201101-201405	41	1	18 063	51	200	-149	-75%
201101-201503	51	1	22 478	46	264	-218	-83%
201101-201601	61	1	26 891	82	392	-310	-79%
201101-201611	71	1	31 301	67	400	-333	-83%
201101-201709	81	1	35 712	47	470	-423	-90%
201101-201807	91	1	40 125	59	535	-476	-89%
201101-201905	101	1	44 539	64	607	-543	-89%

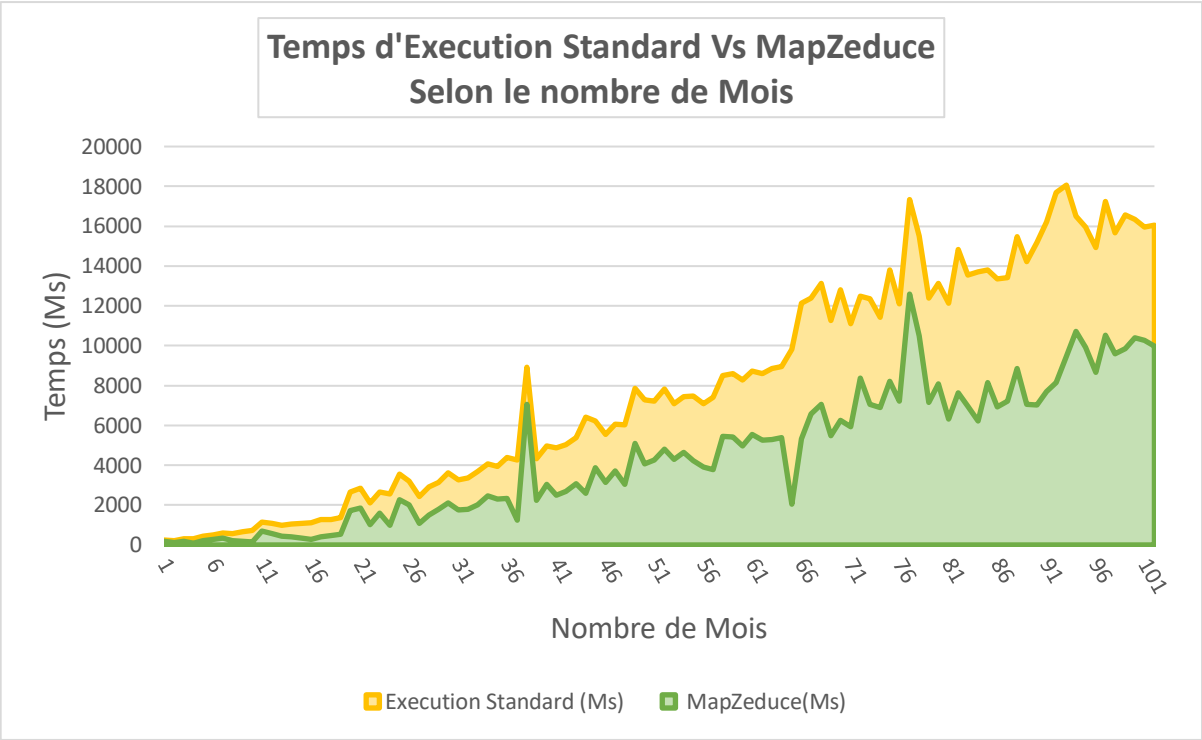

Figure 4-27 : Graphe comparatif de Temps d'exécution Standard Vs MapZeduce

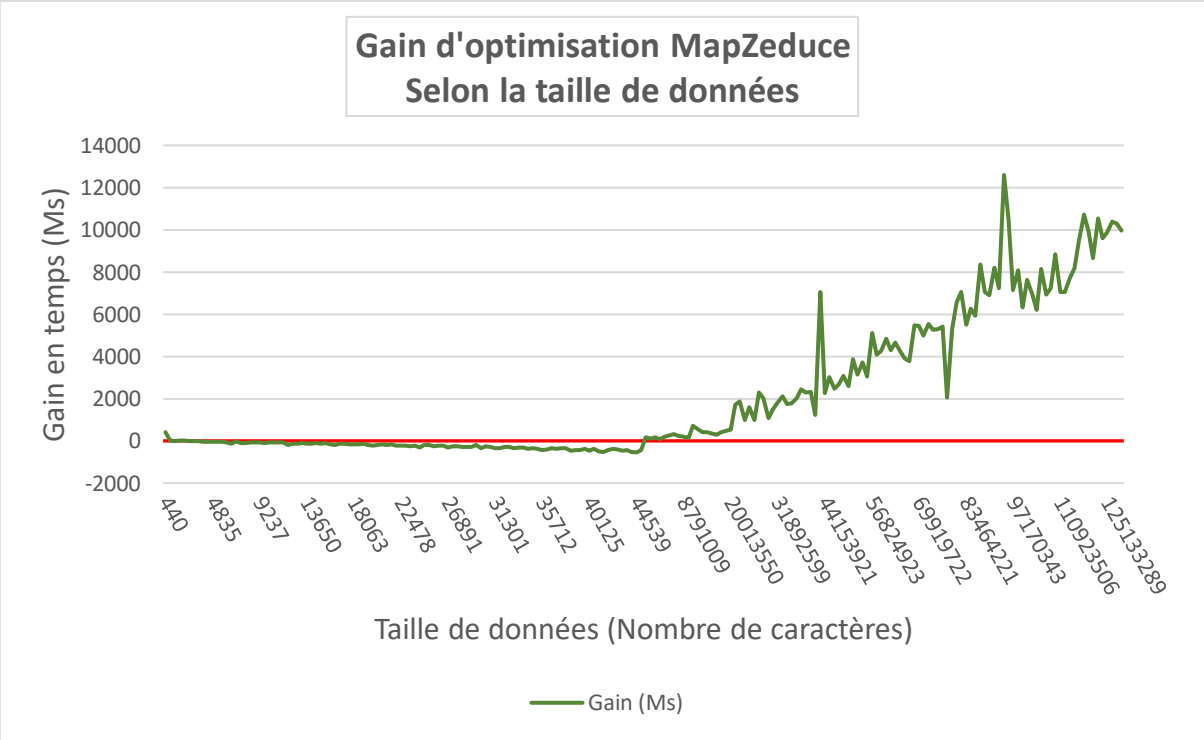

Figure 4-28 : Gain d'optimisation MapZeduce par rapport à la quantité de données

Selon les résultats obtenus nous pouvons retenir les remarques suivantes :

- MapZeduce est globalement plus ***rapide*** (voir la figure Graphe comparatif).
- MapZeduce est rentable au-delà d'un certain ***seuil*** (voir la figure Gain d'optimisation).
- Le gain d'exécution de MapZeduce est ***proportionnel*** à la quantité de données récupérées (voir la figure Gain d'optimisation).

4.6.4 Décentralisation de tâches

La décentralisation des tâches veut dire externaliser le traitement et le déléguer aux niveaux inférieurs (vers les clients). C'est dans le même sens que la programmation modulaire mais avec une différence cruciale concernant niveau d'exécution du traitement. En effet, dans la programmation modulaire les tâche sont migrés d'un composant à un autre mais ils restent dans la même couche (la couche métier), par contre avec la décentralisation, le traitement est expatrié vers la couche présentation.

Le fait de glisser le traitement vers les clients offre énormément d'avantages :

- Libérer les ressources partagées des serveurs, ce qui permet d'éviter un éventuel inter blocage.
- Paralléliser les tâches et incorporer les ressources individuelles dans le processus de traitement.
- Minimiser les requêtes vers les serveurs, en évitant 67% d'entre eux.
- Possibilité de modifier et de personnaliser l'affichage des données, des synthèses et des graphes au niveau client sans solliciter les serveurs, ce qui rend nos pages interactives.

Tableau 4-2 : Comparaison entre la méthode standard et décentralisée

Mode	Tâche \ Méthode / Couche	Standard			Décentralisation		
		SGBD	Serveur Web	Client	SGBD	Serveur Web	Client
Données	Récupération des données	✓			✓		
	Présentation des données		✓				✓
	Visualisation			✓			✓
Synthèse	Calcule des données	✓					✓
	Présentation synthèse		✓				✓
	Visualisation			✓			✓
Graphes	Calcule des données	✓					✓
	Génération Graphes		✓				✓
	Visualisation			✓			✓
Résultat	**Nombre de sollicitation**	**3**	**3**		**1**	**1**	

Le seul inconvénient de la décentralisation qui apparait est la surcharge des pages au niveau client, mais qui reste, d'après notre expérience, supportable et justifiée.

4.7 Conclusion

Dans ce chapitre, nous avons présenté les plateformes utilisées en les découpant par la logique des couches, nous distinguons deux types de plateformes :

- **Plateforme de développement :** qui se compose de l'ensemble des outils et solutions auxquels nous nous sommes référés dans la phase de développement.
- **Plateforme de production :** il s'agit de l'environnement de déploiement.

Ensuite, et après avoir présenté l'interface de notre système avec des Screenshorts des principales pages et composants (filtres) utilisés, nous avons mis en évidence les contraintes conceptuelles et techniques rencontrées durant le projet. Parmi les points que nous avons cité dans le présent chapitre la phase « prétraitement des données », notre implémentation de MapReduce nommé MapZeduce, enfin l'aspect de la décomposition et de la décentralisation des traitements.

Conclusion générale

Au sein des entreprise, les tableaux de bord constituent un élément clé pour les « Top Manager » dans la définition des stratégies. Et à nos jours, dans l'air du Big Data, cet élément demeure indispensable vu la quantité des données a analysée.

L'objectif de ce présent travail, comme il a été déjà défini dans la problématique, est d'offrir un système d'analyse de données capable de manipuler des données à caractère similaire au Big Data dans notre entreprise.

Dans l'état de l'art développé dans les deux premiers chapitres, nous avons vu les notions théoriques du Big Data ainsi que leurs technologies, outils et solutions. Ce qui nous a justement permit de s'inspirer de quelques méthodes pour pouvoir les utiliser dans notre travail.

Dans la deuxième partie -celle de la conception et implémentation- et après avoir posé la problématique rencontrée, nous avons commencé à définir les traits généraux de notre solution, son architecture et sa conception. Cette conception se résume comme suit :

- **Externaliser les services :** d'ailleurs le projet tout entier peut être considéré comme une externalisation du service décisionnel.
- **Décomposer en module :** afin de perfectionner le travail, et pouvoir le réutiliser.
- **Décentralisation et délégation des tâches :** afin d'alléger les ressources partagées, et impliquer les ressources individuelles.

La phase implémentation, exposée dans le chapitre 4, montre quant à elle comment nous avons concrétisé la conception théorique par une solution opérationnelle en utilisant un ensemble d'outils. Nous avons aussi décrit l'environnement de déploiement de notre solution, et nous avons fait ressortir les expériences et les enseignements tirées durant tous notre travail, que nous considérons comme les fruits à récolter. Parmi ces fruits nous citons :

- L'implémentation de la solution MapZeduce que nous avons développée et la comparaison de ses performances avec l'exécution standard des requêtes nous indique qu'elle présente un avantage certain en termes de temps de réponse. Cet avantage apparait au-delà d'un certain seuil de quantité de données traitées et est proportionnel à celui-ci.

- Le choix de décentraliser le traitement des données au niveau des clients permet de libérer les ressources communes (Serveurs) et évite l'inter blocage. Ce choix offre également la possibilité de personnaliser la présentation des résultats au niveau des clients.

Enfin, nous pouvons dire que notre objectif a été en grande partie atteint. Mais comme tout travail humain, il reste perfectible. Il nous semble que des perspectives d'améliorations sont possibles notamment l'automatisation de la phase Data préparation.

Annexe A
Définitions

A.1 Divers définition du terme Big Data

- **Gartner:** Big data is high-volume, high-velocity and/or high-variety information assets that demand cost-effective, innovative forms of information processing that enable enhanced insight, decision making, and process automation [69, 70].

- **McKinsey Global Institute:** Big data refers to datasets whose size is beyond the ability of typical database software tools to capture, store, manage, and analyze [24, 71].

- **Oracle:** Big Data is the derivation of value from traditional relational database driven business decision making, augmented with new sources of unstructured data [71, 72].

- **Microsoft:** Big data is the term increasingly used to describe the process of applying serious computing power - the latest in machine learning and artificial intelligence - to seriously massive and of- ten highly complex sets of information [71, 73].

- **Intel:** According to Intel, Big data opportunities emerge in organizations generating a median of 300 terabytes of data a week. Intel asserts that the most common data type involved in analytics is business transactions stored in relational databases (consistent with Oracle's definition), followed by documents, email, sensor data, blogs and social media [71].

- **National Institute of Standards and Technology (NIST):** Big data is data which exceed the capacity or capability of current or conventional methods and systems [71, 74-76].

- **The Method for an Integrated Knowledge Environment (MIKE2.0) Project:** The high degree of permutations and interactions within a dataset is what defines Big Data. Viewed this way, Big data is not a function of the size of a data set but its complexity [71].

A.2 Définition des critères ACID de la transaction

- **Atomique :** Une transaction représente une unité de travail qui est validée intégralement ou totalement annulée. C'est tout ou rien [77].

- **Cohérente :** La transaction doit maintenir le système en cohérence par rapport à ses règles fonctionnelles. Durant l'exécution de la transaction, le système peut être temporairement incohérent, mais lorsque la transaction se termine, il doit être cohérent, soit dans un nouvel état si la transaction est validée, soit dans l'état cohérent antérieur si la transaction est annulée [77].

- **Isolée :** Comme la transaction met temporairement les données qu'elle manipule dans un état incohérent, elle isole ces données des autres transactions de façon à ce qu'elle ne puisse pas lire des données en cours de modification [77].

- **Durable :** Lorsque la transaction est validée, le nouvel état est durablement inscrit dans le système [77].

A.3 Le Modèle MVC

Le modèle MVC est un modèle de conception, considérer comme standard par de nombreux développeurs, largement dans les applications Web. Son architecture se base sur trois couches principales [78]:

- **Model :** Les classes Modèles sont utilisées pour implémenter la logique de manipulation des données. Ces classes sont utilisées pour récupérer, insérer ou mettre à jour les données dans la base de données associée à notre application.

- **Vues / Présentation :** Les vues servent à préparer l'interface de notre application. Les utilisateurs interagissent avec notre application via cette interface.

- **Contrôleur :** Les classes de contrôleur sont utilisées pour répondre aux demandes de l'utilisateur. Les classes de contrôleur effectuent les actions demandées par les utilisateurs. Ces classes fonctionnent avec les classes de modèle et sélectionnent la vue appropriée à afficher pour l'utilisateur en fonction de ses demandes.

Annexe B
Comparaisons

B.1 GFS Vs HDFS

Tableau B-1 : Comparaison entre GFS et HDFS

HDFS	**GFS**
Platform Convergé	Linux
Développé en JAVA	Développé en C, C++
Free, Open Source Framework	Commercial, propriété de Google
Une structure basé sur (*NameNode*, *DataNode*)	Basé sur (*MastreNode*, *Chunkk* Server)
Taille de block 128 MB par défaut	Taille de block 64 MB par défaut

B.2 HBase Vs Google BigTable

Tableau B-2 : Comparaison entre BigTable et HBase [79]

BigTable	**HBase**
Au démarrage d'un nouveau Mastre la liste des serveurs est lue et mis à jour à partir de *Chubby*.	*Zookeeper* n'a pas cette fonction, les serveurs de régions transmettent les métadonnées aux maîtres eux-mêmes.
Exécution de script client à l'aide de *Sawzall*.	Cette fonctionnalité n'existe pas.
Obtenir plusieurs métadonnées en plus de nécessaire. Le client utilisera son cache de métadonnées dans les prochaines demandes de lecture/écriture.	Cette fonctionnalité n'existe pas.
Locality Groups.	Cette fonctionnalité n'existe pas.
Scanner le cache	Cette fonctionnalité n'existe pas.
Utilise CRC checksums pour vérifier l'intégrité des données ont été transférées.	Cette fonctionnalité n'existe pas.
Cette fonctionnalité n'existe pas.	plusieurs serveurs maîtres de secours pour une récupération plus rapide en cas de panne.
Stocke les horodatages en microsecondes.	Stocke les horodatages en millisecondes.
Repose sur GFS (Google File System).	Fonctionne (HDFS) et peut également s'exécuter sur d'autres systèmes de fichiers.
Peut stocker des fichiers en mémoire	Cette fonctionnalité n'existe pas
Implémente un cache clé/valeur, a deux journaux de validation et est capable de choisir lequel utiliser.	Pour des raisons de performances, la journalisation est optionnelle.
Peut mapper en mémoire des fichiers de stockage entiers et les utiliser pour effectuer des recherches sans rechercher un seul disque.	A une option en mémoire par famille de colonnes et utilise son cache LRU pour conserver les blocs en vue d'une utilisation ultérieure.

Bibliographie

1. ffoulkes, P., *The Intelligent Use of Big Data on an Industrial Scale*. 2017.
2. Kalický, A., *High Performance Analytics*. 2013, Charles University in Prague.
3. Maatallah, H., *Vers un nouveau modèle de stockage et d 'accès aux données dans les Big Data et les Cloud Computing*. 2018, Université de Tlemcen-Abou Bekr Belkaid.
4. Lemberger, P., et al., *Big Data et Machine Learning: Les concepts et les outils de la data science*. 2016: Dunod.
5. Reinsel, D., J. Gantz, and J. Rydning, *The digitization of the world: from edge to core.* Framingham: International Data Corporation, 2018.
6. Reinsel, D., J. Gantz, and J. Rydning, *Data age 2025: The evolution of data to life-critical.* Don't Focus on Big Data, 2017.
7. Wu, C., R. Buyya, and K. Ramamohanarao, *Big data analytics= machine learning+ cloud computing.* arXiv preprint arXiv:1601.03115, 2016.
8. Bastien, L. *Les quatre V du Big Data expliqués par IBM*. Data Analytics 2016; Available from: https://www.lebigdata.fr/infographie-quatre-v-big-data-expliques-ibm.
9. Gardarin, G., *Bases de données*. 2003: Editions Eyrolles.
10. Dean, J., *Big Data, Data Mining, and Machine Learning: Value Creation for Business Leaders and Practitioners*. 2014: Wiley.
11. Mell, P. and T. Grance, *The NIST definition of cloud computing*. 2011.
12. Liu, F., et al., *NIST cloud computing reference architecture*, in *NIST special publication*. 2011. p. 1-28.
13. ENSI-MARIA. *IaaS, PaaS, SaaS – What do they mean?* 2017 01/08/2017 [cited 2019 01/09/2019]; Available from: http://cloudonmove.com/iaas-paas-saas-what-do-they-mean/.
14. Fedoseenko, V. *What is XaaS? IaaS vs SaaS vs PaaS: what's the difference. Examples*. 2018 30/08/2018 [cited 2019 01/09/2019]; Available from: https://www.ispsystem.com/news/xaas.
15. Stephen Watts, M.R. *SaaS vs PaaS vs IaaS: What's The Difference and How To Choose*. 2019 15/06/2019 [cited 2019 01/09/2019]; Available from: https://www.bmc.com/blogs/saas-vs-paas-vs-iaas-whats-the-difference-and-how-to-choose/.

16. Ashton, K., *That 'internet of things' thing.* RFID journal, 2009. **22**(7): p. 97-114.

17. Union, I.T., *Overview of the Internet of things*, in *GLOBAL INFORMATION INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS AND NEXT-GENERATION NETWORKS*. 2012.

18. Le Pallec, S., *La convergence des identifiants numériques.* études, 2005. **3**: p. 4.

19. Bassi, A. and G. Horn, *Internet of Things in 2020: A Roadmap for the Future*. European Commission: Information Society and Media, 2008. **22**: p. 97-114.

20. Benghozi, P.-J., S. Bureau, and F. Massit-Folea, *L'Internet des objets. Quels enjeux pour les Européens?* 2008.

21. Lueth, K.L., *State of the IoT 2018: Number of IoT devices now at 7B–Market accelerating.* IoT Analytics, 2018.

22. astellia. *Harness the business potential of IoT*. 01/02/2018 01/09/2019]; Available from: https://www.astellia.com/solutions/technologies/harness-the-business-potential-of-iot/.

23. Gubbi, J., et al., *Internet of Things (IoT): A vision, architectural elements, and future directions.* Future generation computer systems, 2013. **29**(7): p. 1645-1660.

24. Manyika, J., et al., *Big data: The next frontier for innovation, competition, and productivity.* 2011.

25. Labrinidis, A. and H.V. Jagadish, *Challenges and opportunities with big data.* Proceedings of the VLDB Endowment, 2012. **5**(12): p. 1-15.

26. Agrawal, D., et al., *Challenges and Opportunities with Big Data. A community white paper developed by leading researchers across the United States.* Accessed on September, 2012. **21**: p. 2017.

27. Jagadish, H., et al., *Big data and its technical challenges.* Communications of the ACM, 2014. **57**(7): p. 86-94.

28. Costagliola, G., et al., *Monitoring online tests through data visualization.* IEEE Transactions on Knowledge and Data Engineering, 2009. **21**(6): p. 773-784.

29. Lounes, N., et al. *From KDD to KUBD: Big Data Characteristics Within the KDD Process Steps*. in *World Conference on Information Systems and Technologies*. 2018. Springer.

30. García, S., et al., *Big data preprocessing: methods and prospects.* Big Data Analytics, 2016. **1**(1): p. 9.

31. Ghemawat, S., H. Gobioff, and S.-T. Leung, *The Google file system.* 2003.

32. Bruchez, R., *Les bases de données NoSQL et le Big Data: Comprendre et mettre en oeuvre*. 2015: Eyrolles.

33. Hashem, H. and D. Ranc, *An Integrative Modeling of BigData Processing.* IJCSA, 2015. **12**(1): p. 1-15.

34. Hashem, H., *Modélisation intégratrice du traitement BigData*. 2016, Université Paris-Saclay.

35. Mehdi Acheli, S.K. *Big Data : définition, applications et outils*. 2019; Available from: https://mehdiacheli.developpez.com/tutoriels/bigdata/introduction-definitions-applications-outils/.

36. Tudoran, R., *High-performance big data management across cloud data centers*. 2014, ENS Rennes.

37. White, T., *Hadoop: The definitive guide*. 2012: " O'Reilly Media, Inc.".

38. Sakr, S., *Big Data 2.0 Processing Systems: A Survey*. 2016: Springer International Publishing.

39. Zarate Santovena, A., *Big data: evolution, components, challenges and opportunities*. 2013, Massachusetts Institute of Technology.

40. *Qu'est-ce qu'une base NoSQL ? Les cas Datastax et MongoDB | Digora*. 2019 [cited 2019 28/06/2019]; Available from: https://www.digora.com/fr/blog/definition-base-nosql-datastax-mongodb.

41. DeCandia, G., et al. *Dynamo: amazon's highly available key-value store*. in *ACM SIGOPS operating systems review*. 2007. ACM.

42. Gaspar, D. and I. Coric, *Bridging Relational and NoSQL Databases*. 2017: IGI Global.

43. *riak*. 26/02/2019 10/09/2019]; Available from: https://riak.com/riak-kv/.

44. Fitzpatrick, B. *Distributed Caching with Memcached*. 2004 01/08/2004; Available from: https://www.linuxjournal.com/article/7451.

45. Sanfilippo, S. *redis*. 2009; Available from: https://redis.io/.

46. Da Silva, M.D. and H.L. Tavares, *Redis Essentials*. 2015: Packt Publishing.

47. Macedo, T. and F. Oliveira, *Redis Cookbook: Practical Techniques for Fast Data Manipulation*. 2011: O'Reilly Media.

48. SANJAY GHEMAWAT, J.D., *MapReduce: Simplified Data Processing on Large Clusters.* OSDI, 2004, 2004.

49. Dean, J. and S. Ghemawat, *MapReduce: simplified data processing on large clusters.* Communications of the ACM, 2008. **51**(1): p. 107-113.

50. V, S., V. Govindasamy, and A. Gopu, *Job Scheduling in Big Data-A Survey*. Vol. 08. 2018.

51. Apache. *CouchDB*. 2019 12/03/2019; Available from: http://couchdb.apache.org/.

52. RedHat. *Infinispan*. 2019 23/08/2019; Available from: https://infinispan.org/.

53. Jiang, D., et al., *The performance of mapreduce: An in-depth study*. Proceedings of the VLDB Endowment, 2010. **3**(1-2): p. 472-483.

54. *Apache Hadoop*. 2018 13/11/2018; Available from: https://hadoop.apache.org/.

55. *ZooKeeper: Because Coordinating Distributed Systems is a Zoo*. 2019 20/05/2019; Available from: https://zookeeper.apache.org/doc/r3.5.5/zookeeperOver.html.

56. *Apache Pig*. 2019 18/06/2018; Available from: https://pig.apache.org/.

57. *Apache Flume*. 2019 08/01/2019; Available from: https://flume.apache.org/.

58. Karau, H., et al., *Learning spark: lightning-fast big data analysis*. 2015: " O'Reilly Media, Inc.".

59. Voyer, P., *Tableaux de bord de gestion et indicateurs de performance: 2e édition*. 2011: Presses de l'Université du Québec.

60. Bathelot, B. *Key Performance Indicator*. 2019 09/01/2019 01/09/2019]; Available from: https://www.definitions-marketing.com/definition/kpi/.

61. Dedić, N. and C. Stanier. *Measuring the Success of Changes to Existing Business Intelligence Solutions to Improve Business Intelligence Reporting*. 2016. Cham: Springer International Publishing.

62. Collie, R. and A. Singh, *Power Pivot and Power BI: The Excel User's Guide to DAX, Power Query, Power BI & Power Pivot in Excel 2010-2016*. 2015: Holy Macro! Books.

63. Microsoft. *Power Pivot - Overview and Learning*. 2019 01/09/2019 [cited 2019 01/09/2019]; Available from: https://support.office.com/en-us/article/power-pivot-overview-and-learning-f9001958-7901-4caa-ad80-028a6d2432ed.

64. Allington, M., *Super Charge Power BI: Power BI Is Better When You Learn to Write DAX*. 2018: Tickling Keys, Incorporated.

65. *Klipfolio*. 2019 01/09/2019]; Available from: https://www.klipfolio.com/.

66. Ward, A., C. Screen, and H. Khan, *Oracle Business Intelligence Enterprise Edition 12c*. 2017: Packt Publishing.

67. Niefert, W., *Business Intelligence with SAP BI Edge*. 2015: CreateSpace Independent Publishing Platform.

68. Harinath, S., et al., *Professional Microsoft SQL Server 2012 Analysis Services with MDX and DAX*. 2012: Wiley.

69. Laney, D., *3D Data Management: Controlling Data Volume, Velocity, and Variety*. 2001.

70. Beyer, M.A. and D. Laney, *The importance of 'big data': a definition.* Stamford, CT: Gartner, 2012: p. 2014-2018.

71. Ward, J.S. and A. Barker, *Undefined by data: a survey of big data definitions.* arXiv preprint arXiv:1309.5821, 2013.

72. Dijcks, J.-P., *Oracle: Big data for the enterprise.* Oracle white paper, 2012: p. 16.

73. Redmond, W. *The Big Bang: How the Big Data Explosion Is Changing the World*. 2012; Available from: https://news.microsoft.com/2013/02/11/the-big-bang-how-the-big-data-explosion-is-changing-the-world/.

74. Chen, M., S. Mao, and Y. Liu, *Big data: A survey.* Mobile networks and applications, 2014. **19**(2): p. 171-209.

75. Zakir, J., T. Seymour, and K. Berg, *BIG DATA ANALYTICS.* Issues in Information Systems, 2015. **16**(2).

76. Grady, N. and W. Chang, *NIST Big Data Interoperability Framework: Volume 1, Definitions(2015)*. 2015, NIST.

77. Bruchez, R., *Les bases de données NoSQL et le BigData: Comprendre et mettre en oeuvre*. 2015: Eyrolles.

78. Gupta, P. and M.C. Govil, *MVC Design Pattern for the multi framework distributed applications using XML, spring and struts framework.* International Journal on Computer Science and Engineering, 2010. **2**(04): p. 1047-1051.

79. Rakhmatulin, A. and S. Saquib, *BigTable vs. HBase*, in *Internet Scale Distributed Systems*. 2015: TU Munich.