

REPUBLIQUE DU SENEGAL
UNIVERSITE CHEIKH ANTA DIOP DE DAKAR



ECOLE SUPERIEURE POLYTECHNIQUE
DEPARTEMENT GENIE INFORMATIQUE

MEMOIRE DE FIN DE CYCLE

Pour l'obtention du :
DIPLOME D'INGENIEUR DE CONCEPTION (DIC) EN INFORMATIQUE

SUJET :

**Plateforme d'analyse distribuée sur l'écosystème
Hadoop : Analyse des risques de crédit bancaire**

Lieu de stage : 

Période stage : 02/2016 – 07/2017

Présenté et soutenu par
M. Thierno Adama BAH

Professeurs encadreur
Dr. Mamadou S. CAMARA

Maitres de stage
M. Ahmed Ben Sidy B. SEYE
M. Papa Ngali DIAO

Année universitaire : 2016 – 2017

REPUBLIQUE DU SENEGAL
UNIVERSITE CHEIKH ANTA DIOP DE DAKAR



ECOLE SUPERIEURE POLYTECHNIQUE
DEPARTEMENT GENIE INFORMATIQUE

MEMOIRE DE FIN DE CYCLE

Pour l'obtention du :
DIPLOME D'INGENIEUR DE CONCEPTION (DIC) EN INFORMATIQUE

SUJET :

**Plateforme d'analyse distribuée sur l'écosystème
Hadoop : Analyse des risques de crédit bancaire**

Lieu de stage : **Atos**

Période stage : **02/2016 – 07/2017**

Présenté et soutenu par
M. Thierno Adama BAH

Professeurs encadreur
Dr. Mamadou S. CAMARA

Maitres de stage
M. Ahmed Ben Sidy B. SEYE
M. Papa Ngaly DIAO

DEDICACES











Au nom d'Allah, le Tout Miséricordieux, le Très Miséricordieux.

Louange à Allah, Seigneur de l'univers, qui m'a donné force et courage dans l'accomplissement de ce projet. Que la paix et la bénédiction de Dieu soient sur le Saint Prophète Mouhamed (PSL).

Je dédie ce mémoire :

-  *A ma chère mère et complice **Aissatou BA** qui a toujours été présente, **Maman** je t'adore ;*
-  *A mon père **Boubacar BAH** que je ne saurais remercier assez pour l'éducation qu'il m'a donnée et les sacrifices consentis pour la bonne marche de mes études ;*
-  *A ma chère grand-mère **Fatoumata Kogui Diallo** arrachée à notre affection en 2012. Que Dieu vous accorde le paradis ;*
-  *A notre chère et bien aimée grande sœur **Feu Assiatou DIALLO**, arrachée à notre affection le 01 Juillet 2017. Que Dieu vous accorde le paradis ;*
-  *A mes frère et sœurs **Mamadou Sadio, Ibrahima, Aminatou, Amadou Diogo** ;*
-  *A mes **oncles, tantes, cousins, cousines, neveux, nieces** ;*
-  *A ma seconde Mère **Seynabou Coly** qui a beaucoup fait pour moi ;*
-  *A tous les membres du **Dahira Achiratoul Mouhammadiyya de Castors** ;*
-  *A tous les membres du **Dahira Tidiane ESP/ENSETP** ;*
-  *A la Team LG : **Abdou Khadre, Lamine Kane, Mar Diop, Mouhamed Niasse, Gassama** ;*
-  *A tous mes amis: **Ibrahima Ndoye, Boubacar Sissokho, Salamata Athie, Bernard Sané, Aliou Wélé, Baba Ly, Abdourahmane Diallo, EL-hadj Malick Coulibaly, Samba Amady Camara, Amadou Sy, Bienvenue, Aissatou Sene, Ahmed Diop ...***
-  *A tous **mes camarades** de la promotion DIC 2014/2017;*

J'adresse mes remerciements les plus chaleureux à:

-  ***Dr Alassane BAH**, Chef du département Génie Informatique, pour cet enseignement de qualité dont nous avons bénéficié,*
-  ***Dr Ibrahima FALL**, responsable pédagogique du DIC informatique,*
-  ***Dr Mamadou Samba CAMARA**, mon professeur encadreur, pour sa disponibilité, ses remarques et suggestions tout au long de la rédaction de ce mémoire,*
-  *L'ensemble du corps professoral de l'ESP,*
-  ***M. Momadou NDOYE**, Directeur des Opérations Atos GDC Sénégal,*
-  ***Mme Khadydiatou NDIAYE** Responsable RH Afrique de l'Ouest,*
-  *Mon référent **M. Tariq DJIBALENE** pour ses remarques et suggestions tout au long de la rédaction de ce mémoire,*
-  *Mes maitres de stage **Ahmed Ben Sidy Bouya SEYE** et **Papa Ngal DIAO**, pour les nombreux sacrifices qu'ils ont consentis à l'aboutissement du présent travail, qu'ils ont suivi patiemment, et surtout avec le professionnalisme qu'on leur connaît, je leur rends le témoignage de toute ma reconnaissance*
-  *Toute l'équipe du GDC SN du groupe ATOS,*
-  *Toutes les personnes qui de près ou de loin, ont contribué à la réalisation de ce document.*

L'Ecole Supérieure Polytechnique (ESP) est un établissement public de formation professionnelle qui forme aussi bien sur le plan théorique que pratique des techniciens supérieurs, des managers et des ingénieurs de conception dans différents domaines.

Dans l'optique de compléter leur formation d'ingénieur, les étudiants en dernière année du diplôme d'ingénieur de conception (DIC) en Informatique du département Génie informatique doivent effectuer un stage de cinq (5) mois au sein d'une entreprise afin de mettre en pratique les connaissances théoriques acquises au cours de la formation et d'être conscients de l'environnement socioprofessionnel.

C'est dans ce contexte que nous avons effectué un stage au sein du groupe ATOS sur la mise en place d'une plateforme d'analyse distribuée sur l'écosystème Hadoop et de l'appliquer à l'analyse au risque de crédit bancaire.

Ce document est un mémoire de fin de cycle pour l'obtention du diplôme d'ingénieur de conception en informatique. Un stage de fin d'étude d'une durée de cinq mois a été effectué au sein d'ATOS qui est l'un des dix plus grands acteurs des Entreprises de Services du Numérique (ESN) au niveau mondial.

Ce stage s'inscrit dans le cadre d'un projet de mise en place d'une plateforme Big Data au niveau des serveurs d'ATOS et d'y appliquer plusieurs cas d'étude. C'est ainsi qu'il nous a été demandé de faire une étude et mise en place d'une plateforme d'analyse distribuée et de l'appliquer au risques de crédit bancaire.

Dans le but de bien mener ce projet, nous allons d'abord choisir la méthodologie de gestion de projet à adopter, ensuite nous allons faire une étude sur le Big Data et l'écosystème Hadoop, nous allons également étudier l'apport du Big Data pour l'amélioration du modèle scoring pour terminer par la conception et la mise en œuvre de la plateforme de test.

This document is an end of cycle dissertation for obtaining the design engineer degree. A final internship study of a five-month period was performed within ATOS which is one of the ten largest actors in the Digital Services Companies at global level.

This internship is part of a project to set up a Big Data platform in ATOS datacenter and to apply several use cases. Therefore, we were asked to do the study and implementation Distributed analysis platform and apply it to the risks of bank credit.

For the purpose of good conduct this project, we will first choose the project management methodology to adopt, then we will do a study on the Big Data and the Hadoop ecosystem, we will also study the contribution of the Big Data for the improvement of the scoring model to finish with the design and implementation of the test platform.

Sigles et abréviations	viii
Table des figures.....	ix
INTRODUCTION.....	1
• Chapitre 1 : Présentation générale	3
I.1 Présentation de la structure d'accueil.....	3
I.1.1 Présentation du groupe Atos	3
I.1.2 GDC Sénégal.....	6
I.2 Présentation du sujet.....	8
I.2.1 Contexte	8
I.2.2 Problématique.....	9
I.2.3 Objectifs	10
• Chapitre 2 : Choix d'une méthode d'analyse et de conception.....	12
II.1 Définition des concepts	12
II.1.1 Projet	12
II.1.2 Cycle de vies d'un projet.....	12
II.1.3 Gestion d'un projet.....	12
II.1.4 Analyse.....	12
II.1.5 Conception	13
II.2 Principales méthodologies.....	13
II.2.1 Les méthodes classiques.....	13
II.2.2 Les méthodes agiles	17
II.3 Choix de la méthodologie	21
• Chapitre 3 : Big Data et l'écosystème Hadoop	22
III.1 Définitions du Big Data.....	22
III.1.1 Définition technologique du Big Data	22
III.1.2 Une révolution économique	22
III.2 Paysage Technologique du Big Data	23
III.2.1 Apache Hadoop	23
III.2.2 Apache MapReduce	26

III.2.3	Hadoop YARN	27
III.2.4	Hadoop Distributed File System (HDFS)	29
III.3	L'écosystème Hadoop	31
III.4	Les systèmes classiques de gestion de bases de données et ses limites	33
III.5	Le NoSQL	34
•	Chapitre 4 : Analyse et Conception de la solution	39
IV.1	Spécifications fonctionnelles.....	39
IV.1.1	Les fonctionnalités générales de notre plateforme.....	39
IV.2	Architecture de la plateforme	39
•	Chapitre 5 : Risque de crédit et Big Data	42
V.1	Le crédit scoring en général	42
V.2	Credit scoring et Big Data.....	43
V.3	Les méthodes statistiques usuelles et leurs limites.....	44
V.4	Le processus de Machine learning pour construire un modèle de crédit scoring Big data	47
V.5	Enjeux du Big Data pour l'analyse des risques bancaire	49
•	Chapitre 6 : Mise en œuvre de la plateforme.....	50
VI.1	Présentation des technologies utilisées	50
VI.1.1	Les distributions Hadoop	50
VI.1.2	Hortonworks Data Plateforme.....	53
VI.1.3	Apache Kafka.....	56
VI.1.4	Apache Spark	58
VI.2	Réalisation de la plateforme	62
VI.2.1	Déploiement du cluster via HD	62
VI.2.2	Collecte de données « Twitter ».....	63
	CONCLUSION.....	68
	Bibliographie.....	70
	Annexe	71

Nous présentons ici certains sigles et abréviations que nous utiliserons dans le document.

ACID	Atomicité Cohérence Isolation Durabilité
API	Application Programming Interface
CDR	Call Data Record
CRM	Customer Relationship Management
GDC	Global Delivery Center
HDFS	Hadoop Distributed File System
HDP	Hortonworks Data Platform
HQL	Hive Query Language
NoSQL	Not Only SQL
OLTP	OnLine Transaction Processing
RAD	Rapid Application Development
RDD	Resilient Distributed Dataset
RUP	Rational Unified Process
SGBDR	Système de Gestion de Base de Données
SQL	Structured Query Language
XP	eXtreme Programming
YARN	Yet Another Resource Negotiator

Table des figures

Figure 1: Répartition des employés d'Atos dans le monde	4
Figure 2: Historique du groupe d'Atos	4
Figure 3: Les activités d'Atos GDC Sénégal	7
Figure 4: Les secteurs d'activités du GDC Atos Sénégal.....	7
Figure 5: Organigramme GDC Sénégal	8
Figure 6: Modèle en cascade.....	14
Figure 7: Cycle en V	15
Figure 8: Cycle en spirale	16
Figure 9: Cycle de vie d'un projet XP.....	19
Figure 10: Fonctionnement de Scrum	20
Figure 11: Cluster Hadoop multinœud.....	25
Figure 12: Architecture de fonctionnement de MapReduce.....	27
Figure 13: Evolution de l'architecture MapReduce vers YARN	27
Figure 14: Architecture de fonctionnement de YARN.....	28
Figure 15: Architecture HDFS	31
Figure 16: Architecture de l'écosystème Hadoop	32
Figure 17: Architecture en couche	40
Figure 18: Architecture technique de la plateforme	41
Figure 19: Arbre de décision.....	45
Figure 20: Réseau de neurones.....	46
Figure 21: Etapes pour construire un outil de credit scoring.....	47
Figure 22: Architecture Hortonworks Data Plateforme	51
Figure 23: Architecture Cloudera.....	52
Figure 24: Architecture MapR	52
Figure 25: Fonctionnement d'un cluster Kafka.....	58
Figure 26: Style de traitement de Spark	59
Figure 27: Les composants de Spark.....	59
Figure 28: Fonctionnement de spark streaming (1).....	61
Figure 29: Fonctionnement spark streaming (2)	61
Figure 30: Page d'accueil d'Ambari	63
Figure 31: Démarrage du script de création de topic kafka et repertoires HDFS.....	64
Figure 32: Collecte de données twitter en temps réel.....	65
Figure 33: Streaming des données collectées avec Spark vers HDFS.....	66
Figure 34: Visualisation avec Ambari de répertoires HDFS créés pour chaque flux de tweet.....	66
Figure 35: Fichier crée pour un flux de tweet	67

Figure 36: Visualisation avec Ambari du flux de tweet stockées dans HDFS	67
Figure 37: Scripte permettant de créer un topic kafka, et des répertoires dans hdfs	71
Figure 38: Scripte pour démarrer le producer Twitter.....	71

La mise à disposition généralisée d'outils numériques de plus en plus performants et connectés, la démocratisation de l'accès instantané à l'information, l'influence du mouvement Open Data, et la multiplication des objets interconnectés, provoquent une explosion des données numériques qui se mesurent aujourd'hui en exaoctets. En 5 ans, la quantité de données générées et échangées à travers le monde a été multipliée par 7. Si plus de 90% des données existantes ont été créées ces deux dernières années, 88% d'entre elles ne sont pas exploitées. En plus de leur nombre, les données mises à disposition sont de plus en plus variées et provenant de différentes sources : les données de logs, les données mobiles, les vidéos, les réseaux sociaux etc. Ces données sont venues perturber les technologies traditionnelles de traitement de données.

En effet, soumises à de tels enjeux de volumétrie et d'hétérogénéité des données, les technologies traditionnelles n'ont pas tardé à montrer leurs limites et il a été nécessaire de réinventer un certain nombre d'outils qui pourront s'adapter à ces nouvelles contraintes qui sont le stockage, le traitement distribué, la collecte de données hétérogènes et multi-sources, l'analyse et l'exploitation de grands volumes de données pour créer de la valeur ajoutée et la restitution des données. D'ailleurs, tout l'enjeu pour les entreprises et les administrations consiste à ne pas passer à côté d'informations précieuses cachées dans la masse. C'est là qu'intervient le phénomène Big Data qui repose sur une analyse très fine des masses de données.

Afin de saisir l'impact de ce nouveau paradigme et répondre aux problématiques citées ci-dessus, il est important d'avoir des connaissances sur les nouveaux concepts, ainsi que sur les technologies clé, qui définissent le Big Data.

Par ailleurs depuis plusieurs années le risque bancaire est l'une des raisons de perte de profit pour les établissements financiers, l'environnement bancaire est devenu très vulnérable principalement à cause du non remboursement de crédit de la part des entreprises et des particuliers. Ce risque est souvent dû à un manque d'informations sur les demandeurs.

Conscient de l'utilité d'une bonne gestion du Big Data, le groupe Atos GDC Sénégal notre structure d'accueil souhaite mettre en place une plateforme Big Data. Il est nécessaire de faire une étude et une analyse approfondie sur l'écosystème Hadoop. Plusieurs cas d'usage

pourront être testés dont l'analyse des risques de crédit bancaire. Autrement dit utilisation du Big Data pour améliorer les algorithmes prédictifs de Machine Learning.

Ce présente mémoire s'articule autour de six chapitres :

- ❖ Le premier chapitre, « **Présentation générale** », fait une présentation de notre structure d'accueil le groupe **AtoS**, de même qu'une présentation de notre sujet.
- ❖ Dans le deuxième chapitre « **Méthodologie d'analyse et de conception** », nous allons présenter les différentes méthodologies d'analyse et de conception pour choisir celle qui sera mieux adaptée à notre projet.
- ❖ Dans le troisième chapitre « **Big Data et Ecosystème Hadoop** », nous allons présenter le concept du Big Data et les technologies de base utilisé à savoir l'écosystème Hadoop.
- ❖ Dans le quatrième chapitre « **Analyse et conception de la solution** », nous allons définir les spécifications fonctionnelles de la plateforme et proposer une architecture fonctionnelle.
- ❖ Dans le cinquième chapitre « **Crédit Scoring et Big Data** » nous allons définir le crédit scoring, état de l'art des différents modèles prédictifs utilisés dans le scoring credit ainsi l'apport du Big Data pour l'amélioration de ces modèles.
- ❖ Dans le sixième chapitre « **Mise en œuvre de la plateforme** », nous allons décrire le choix des outils et des technologies et présenter la plateforme.



Chapitre 1 : Présentation générale

Dans ce chapitre, nous allons présenter le Groupe AtoS qui est la structure au sein de laquelle nous avons effectué notre projet de fin d'étude. Ensuite, nous allons présenter le sujet à travers le contexte, la problématique et les objectifs du sujet.

I.1 Présentation de la structure d'accueil

I.1.1 Présentation du groupe Atos

a. AtoS en chiffre

Atos est l'un des dix plus grands acteurs des Entreprises de Services du Numérique (ESN) au niveau mondial. Avec un chiffre d'affaires annuel de 12 billions d'euros, actuellement la société emploie environ 100 000 personnes dans 72 pays, principalement en France, en Allemagne aux Pays-Bas, en Espagne, au Royaume-Uni, en Inde, au Maroc et depuis décembre 2014 au Sénégal. La figure suivante nous donne une idée de la répartition des employés d'AtoS à travers le monde.

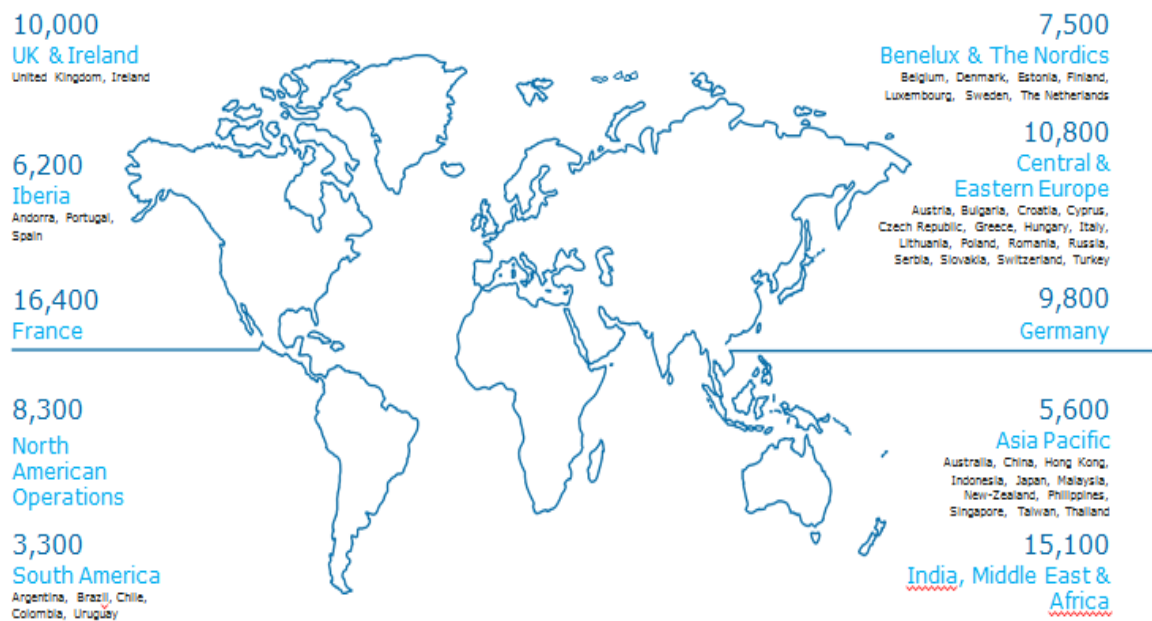


Figure 1: Répartition des employés d'Atos dans le monde

b. Historique

L'histoire du groupe AtoS a été rythmée par des acquisitions et des fusions dont la dernière en date est l'acquisition de la société française Bull comme illustrée par la figure ci-dessous :

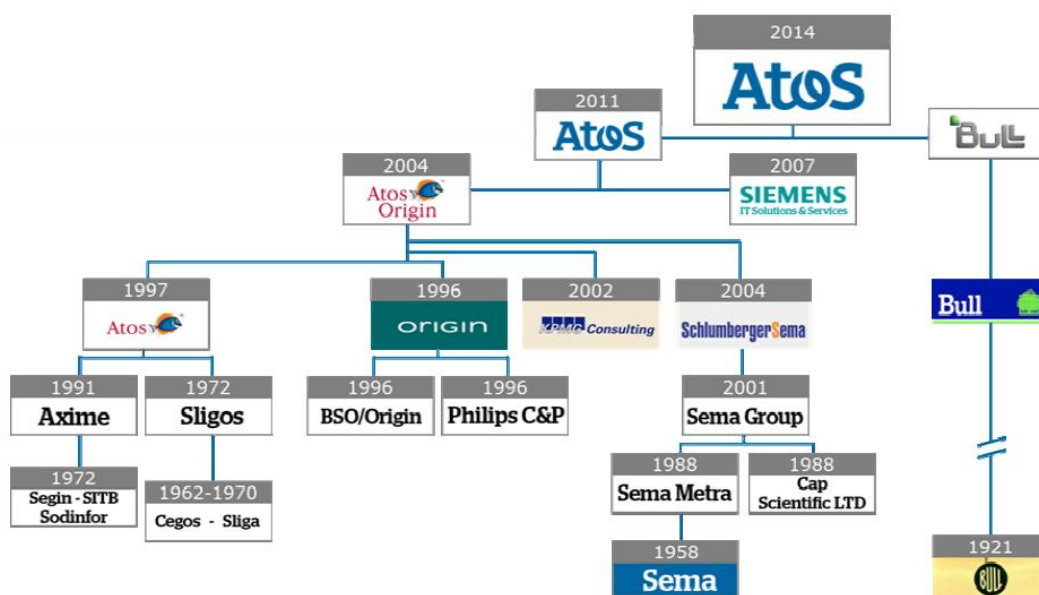


Figure 2: Historique du groupe d'Atos

c. Domaine d'activité

Coté sur le marché Euronext Paris, le Groupe exerce ses activités sous les marques:

- ❖ **Atos Consulting** : conseil en organisation et en processus métiers
- ❖ **Atos Worldgrid** : solutions intelligentes dans le domaine des énergies
- ❖ **Atos Worldline** : solutions de paiement, gestion de la relation client(CRM), e-services, BPO (infogérance).

Ses activités s'organisent autour de quatre grands métiers principaux :

- ❖ **Le conseil et les services technologiques :**

Atos accompagne ses clients en les aidant à gagner en rentabilité et en efficacité grâce aux technologies et à l'innovation.

- ❖ **L'intégration de système :**

L'intégration de système consiste à réunir au sein d'un même système d'information, des parties développées de façon séparée. Atos fournit des systèmes parfaitement intégrés.

- ❖ **L'infogérance :**

L'infogérance est la délégation à un prestataire extérieur de la gestion, l'exploitation, l'optimisation et la sécurisation du système d'information d'une société. Atos gère et transforme ainsi toutes les opérations informatiques de ses clients, y compris l'ensemble de leurs systèmes informatiques et de traitement de données.

- ❖ **Services transactionnels et de paiement :**

Worldline, filiale d'Atos, est le leader européen et un acteur mondial dans les services de paiement. Les activités de Worldline sont organisées autour de trois axes :

- Merchant Services & Terminals,
- Mobility & e-Transactional Services
- Financial Processing & Software Licensing

En 2013, les activités de Worldline, au sein du Groupe Atos, ont généré un chiffre d'affaires d'1,1 milliard d'euros.

d. Partenaire

Atos s'appuie sur un réseau de partenaires stratégiques afin de combiner leurs expertises et fournir à leurs clients des solutions innovantes. Les partenaires d'Atos sont les suivants :

- ❖ **EMC²** : Une technologie largement utilisée pour équiper leurs clients de services d'infrastructure robustes et fiables.
- ❖ **Microsoft** : La combinaison de solutions souples et innovantes pour anticiper et s'adapter aux évolutions de l'activité des clients.
- ❖ **Oracle** : Des solutions basées sur les systèmes d'entreprise Oracle et les processus business qu'ils supportent.
- ❖ **SAP** : Atos est partenaire de SAP Global Services & Hosting et membre de l'initiative de partenariat.
- ❖ **Siemens** : un fabricant de technologie
- ❖ **VMware** : Solutions cloud
- ❖ **HP** : solutions pour aider dans le développement de CA, la réduction des coûts et le gain d'efficacité.
- ❖ **IBM** : solutions couvrant Mainframes et serveurs Unix & Intel et leur association à des solutions logicielles et de stockage.

I.1.2 GDC Sénégal

Le Global Delivery Center du Sénégal (GDC SN) a été mis en place en décembre 2014 au Sénégal. C'est un centre de compétence technique capable de fournir des services informatiques à des entreprises d'Afrique Subsaharienne et plus globalement à tous les clients AtoS qui sont répartis dans les quatre coins du monde. La vocation de ce centre de compétence est avant tout l'export d'expertise informatique auprès des pays occidentaux encore connu sous le nom d'OffShoring. Ce phénomène désigne la délocalisation des services informatiques d'une entreprise vers un pays à bas salaire. Concrètement les entreprises des pays occidentaux font exécuter une partie de leurs travaux liée à des sujets technologiques ou des sujets de production par des ressources qui sont dans des pays où les coûts sont moindres. Ce centre s'activera dans l'intégration de solutions et dans le développement et la maintenance d'application entre autres, comme l'illustre la figure suivante :

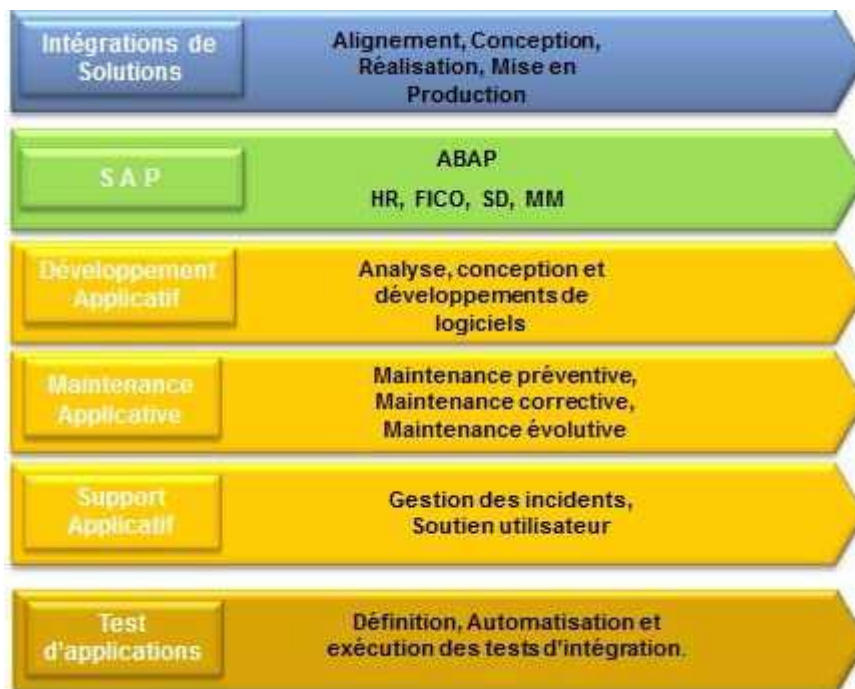


Figure 3: Les activités d'Atos GDC Sénégal

Les métiers cibles du GDC SN sont les télécoms, l'industrie, la banque, l'assurance, le secteur public et la distribution.

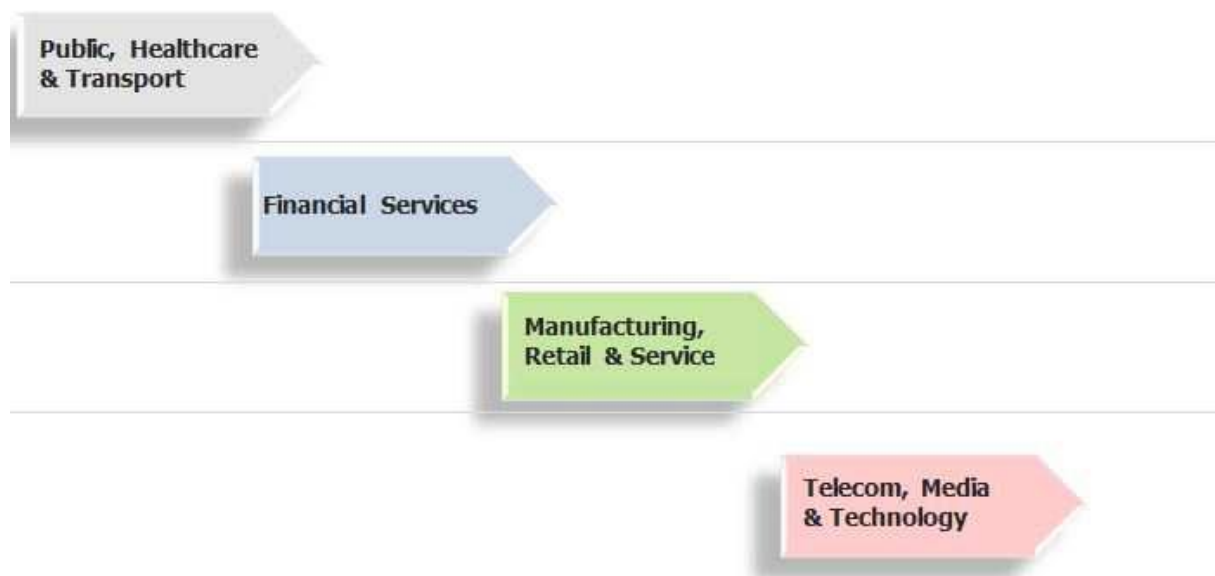


Figure 4: Les secteurs d'activités du GDC Atos Sénégal

Le GDC Sénégal compte à ce jour près de 250 collaborateurs.

Et comme toute structure, le GDC Sénégal a une hiérarchie et est organisé de la façon suivante :

GDC SI : Organisation

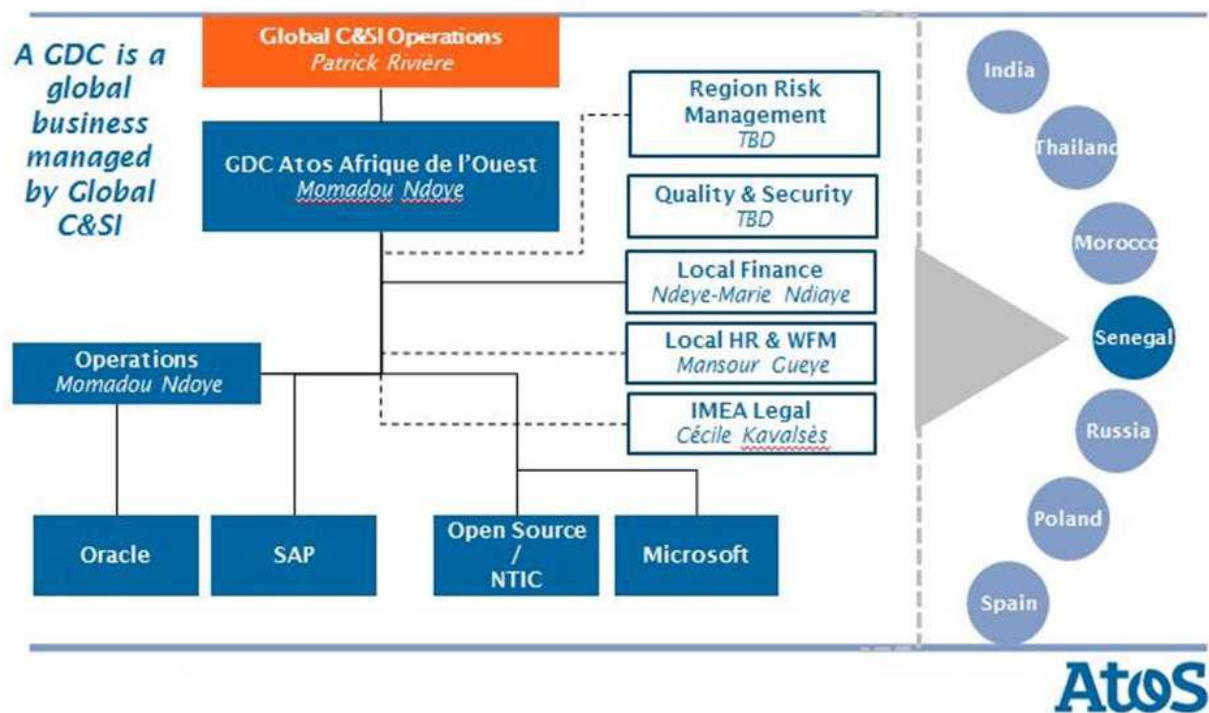


Figure 5: Organigramme GDC Sénégal

I.2 Présentation du sujet

I.2.1 Contexte

Le déluge de données produites par le monde digital (e-commerce, requêtes Internet, réseaux sociaux, capteurs, smartphones et de plus en plus d'objets connectés) conduit les entreprises - ou du moins certaines d'entre elles- à une utilisation radicalement différente des données. Poussée par les besoins de stockage et de traitement, l'évolution technologique ces dernières années est telle qu'elle ouvre des possibilités inenvisageables jusque-là : la gestion du « Big Data ».

Le Big Data contraint à de nouvelles manières de voir et analyser le monde. De nouveaux ordres de grandeur concernent la capture, le stockage, la recherche, le partage, l'analyse et la visualisation des données.

Plateforme d'analyse distribuée sur l'écosystème Hadoop : Analyse des risques de crédit bancaire

Pour caractériser le Big Data, on parle de la règle des 3V devenue par extension 5V qui fait référence à Volume, Variété, Vitesse, Véracité et la Valeur. S'agissant de la notion de Volume, on parle de Big Data au-delà de 100 To (téraoctets) de données structurées ou non structurées. La Variété nous rappelle que la diversité des données est fondamentale pour le Big Data. La vitesse de traitement est enfin cruciale et propre à la notion de Big Data. La véracité qui suppose à vérifier systématiquement la validité des données. Ces dimensions supposent que l'on soit capable de stocker et d'exploiter des volumes énormes de données, d'accéder à de multiples sources par un réseau performant, de traiter dans des temps toujours plus courts des volumes considérables de données.

Face à cette situation, le groupe AtoS leader international dans le secteur informatique, veut mettre en place une plateforme Big Data au niveau de ses puissants serveurs **bullion**¹ afin de faire travailler les données.

Par ailleurs avec toute cette masse de données on évoque de nouvelles possibilités d'exploration notamment l'analyse du risque de crédit qui est un problème pour beaucoup d'institution financière.

C'est dans ce cadre que le groupe Atos a proposé un projet de mise en place d'une plateforme d'analyse distribuée sur l'écosystème hadoop et puis d'analyser les risques de crédit.

I.2.2 Problématique

Le volume de données créées quotidiennement ces dernières années est impressionnant. On parle aujourd'hui couramment de péta octets (milliard de méga d'octets). Entre 2010 et 2012 le volume de données récoltées aurait été équivalent aux données générées depuis le début de l'humanité. Selon une étude du cabinet IDC, le volume de données va être multiplié par 50 entre 2005 et 2025. Cette explosion de données est liée au développement du digital facilitant la création, le stockage et la transmission des données. Celles-ci proviennent du développement de l'Internet, des pratiques de communication permanente des internautes et des mobinautes sous forme de production de contenus mais également de réactions, commentaires à ces contenus. 150 milliards de mails et 500 millions de tweets sont émis chaque jour, 24 péta octets traités chaque jour par google. En 2020, il y aura 5200 Go

¹ Serveurs x86, 2 à 16 processeurs Intel® Xeon® E7 v4 et jusqu'à 24To de mémoire

d'informations numériques pour chaque humain sur terre. C'est environ 500 tonnes de livres pour chaque humain à lire.

Face à cette croissance exponentielle du volume des données et la diversité de leurs formats, les organisations sont confrontées à des problèmes tels que la collecte, le stockage, l'analyse et l'exploitation de grands volumes de données pour créer de la valeur ajoutée. D'ailleurs, les bases de données traditionnelles ne suffisent plus pour gérer les données. Il faut imaginer de nouvelles architectures, de nouvelles technologies et des nouveaux algorithmes pour stocker, traiter et transporter ces masses considérables de données.

Donc, tout l'enjeu pour les entreprises et les administrations du XXIème siècle consiste à stocker à long terme ces volumes de données ainsi de les traiter et de les analyser pour ne pas passer à côté d'informations précieuses cachées dans la masse.

Par ailleurs depuis plusieurs années le risque bancaire est l'une des raisons de perte de profit pour les établissements financiers, l'environnement bancaire est devenu très vulnérable principalement à cause de non remboursement du crédit de la part des entreprises et les particuliers. Les méthodes de credit scoring (processus d'évaluation du risque de crédit) utilisées ont montré leurs limites et lésant des millions de personnes qui n'ont pas d'historiques de crédit suffisant mais qui peuvent être fiables ce qui représente un marché inexploité avec un potentiel gros bénéfice.

Il n'est donc pas surprenant à l'ère du Big Data, que les données tirées des recherches sur internet, des médias sociaux, des applications mobiles soient utilisées pour améliorer les risques de crédit.

Le groupe Atos, leader international de la transformation digital, veut étendre son expertise en service Big Data, en fournissant les bénéfices d'une plateforme d'analyse distribué sur l'écosystème Hadoop et l'application aux risques de crédit

I.2.3 Objectifs

L'objectif de ce projet est de réaliser une plateforme d'analyse distribué qui nous permettra de collecter et stocker les données.

Les macros taches que nous allons réaliser dans ce projet sont les suivantes :

- Etude et documentation sur l'écosystème Hadoop et proposer une architecture technique.

- Proposer une distribution Hadoop pour assurer le fonctionnement sans problème de toutes les versions de l'écosystème Hadoop utilisées.
- Mise en place de la plateforme de démonstration pour la collecte et le stockage
- Etude sur l'application aux risques de crédit

■ Chapitre 2 : Choix d'une méthode d'analyse et de conception

Dans ce chapitre, nous allons faire une étude approfondie des principales méthodologies d'analyse et de conception et choisir la méthodologie la mieux adaptée à notre projet

II.1 Définition des concepts

II.1.1 Projet

Il existe de nombreuses tentatives de normalisation de la notion de projet, donnant lieu à beaucoup de définitions relativement proches. Parmi celles-ci, nous avons celle proposée par la norme ISO 10006 selon laquelle: un projet est un processus unique, qui consiste en un ensemble d'activités coordonnées et maîtrisées comportant des dates de début et de fin, entrepris dans le but d'atteindre un objectif conforme à des exigences spécifiques telles que des contraintes de délais, de coûts et de ressources.

II.1.2 Cycle de vies d'un projet

On appelle « cycle de vie du projet » l'enchaînement dans le temps des étapes entre l'émergence du besoin et la livraison du produit.

II.1.3 Gestion d'un projet

La gestion de projet est la mise en œuvre de connaissances, de compétences, d'outils et de techniques appliqués au projet afin d'en respecter les exigences, vis-à-vis du client et de sa propre hiérarchie.

II.1.4 Analyse

Dans le cadre de projets informatiques, la phase d'analyse est une étape clé qui consiste à caractériser les besoins du futur produit, de lister les résultats attendus en terme de fonctionnalités, de performance, de maintenance, de robustesse, de sécurité etc. L'analyse

permet donc d'avoir une vision claire et précise du problème posé et du système à réaliser. Il est donc important, durant cette phase de ne pas perdre de vue les besoins des utilisateurs ainsi que les frontières du système à bâtir.

II.1.5 Conception

La conception, quant à elle, consiste à décrire de manière très claire, grâce à un langage de modélisation, le futur système, afin d'en faciliter la réalisation. Etant la phase qui suit l'étape de l'analyse, la conception doit répondre à la question du « Comment ? » en mettant l'accent sur une solution conceptuelle qui répond aux exigences plutôt que sur une implémentation. Ainsi, dans la phase de conception, nous allons procéder aux choix de l'architecture du système et des outils à utiliser. De ce fait, avec les modèles obtenus au sortir de l'étape de conception, seront assez détaillés pour passer à l'étape d'implémentation.

II.2 Principales méthodologies

II.2.1 Les méthodes classiques

Ces méthodes sont basées sur des activités séquentielles : recueil des besoins, définition du produit, développement puis test avant livraison au client. Elles sont utilisées pour gérer les projets depuis plusieurs années. Elles se basent sur le principe que tout doit être planifié au début et que tout doit être prévisible. Nous présentons ci-dessous quelques-unes des méthodes classiques.

a. Modèle en cascade

Modèle apparu en 1966, il est issu de la construction du bâtiment où la fondation doit être faite avant le reste de la maison. Les tâches s'exécutent d'une manière totalement successive. Dans son modèle de base, il se basait sur le principe du non-retour c'est-à-dire qu'il n'y a pas de remise en cause d'une étape précédente. Mais dans sa version actuelle, la fin de chaque phase a lieu à une date prédéterminée et s'achève par la production d'un certain nombre de documents ou logiciels. Ces livrables sont produits pour réaliser la Vérification/Validation qui permet d'approuver le passage à la phase suivante ou la correction de l'étape précédente. Il est très adapté aux petits projets simples avec peu d'incertitude et où les besoins sont clairement identifiés et stables.

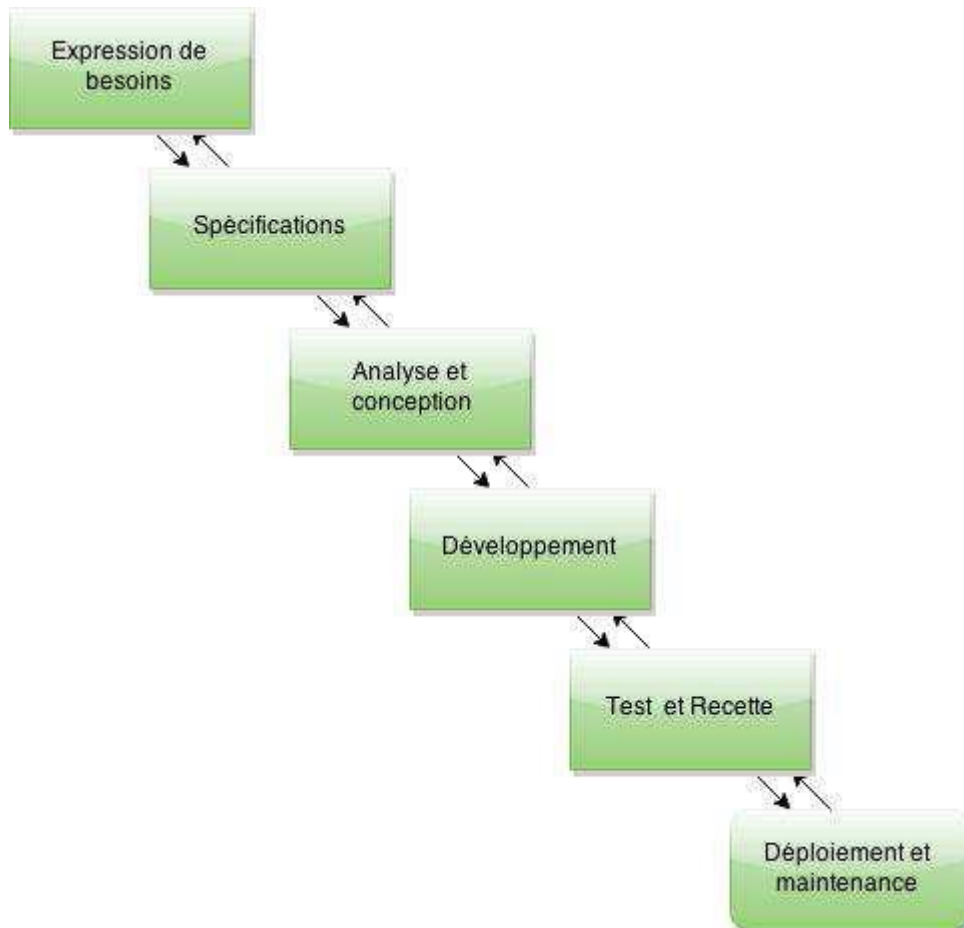


Figure 6: Modèle en cascade

b. Cycle en V

Issu du monde de l'industrie, le cycle en V est devenu un standard de l'industrie logicielle depuis les années 1980. C'est une méthode qui découle de l'amélioration du modèle en cascade où chaque phase du projet a une phase de test qui lui est associée. Cette association découle de l'idée selon laquelle chaque phase en amont du développement doit préparer une phase correspondante de vérification en aval de la production du logiciel. Cette méthode permet de limiter un retour aux étapes précédentes.

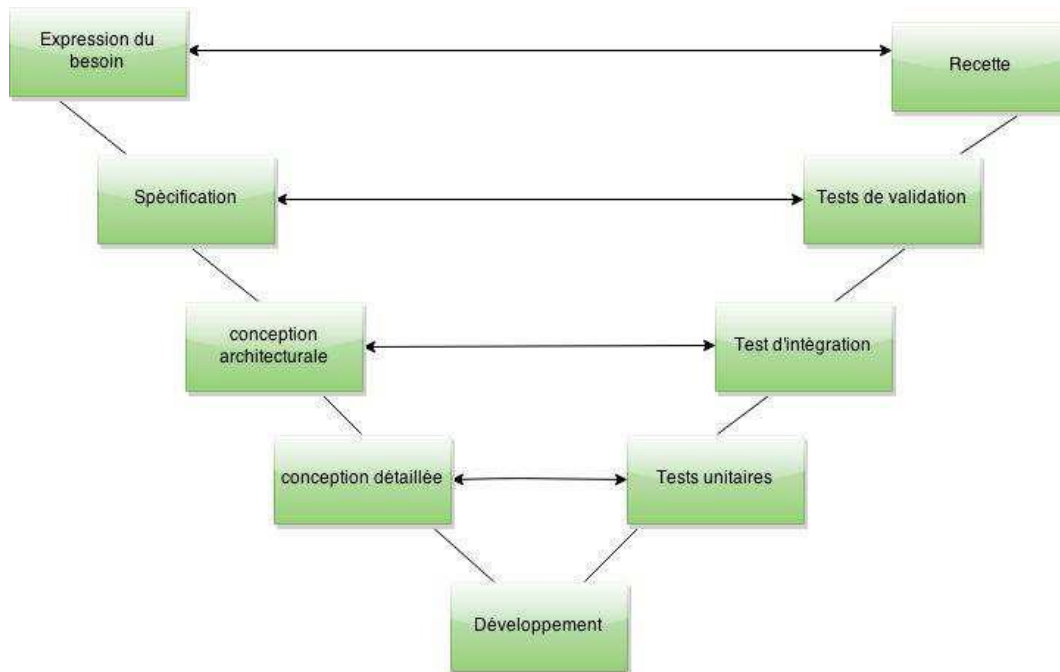


Figure 7: Cycle en V

c. Cycle en Spirale

C'est une méthode qui reprend les différentes étapes du cycle en V en proposant un produit de plus en plus complet et de plus en plus robuste grâce à des implémentations de versions successives. Toutefois, Cette méthode met plus l'accent sur la gestion des risques que le cycle en V. Ainsi, le début de chaque itération comprend une phase d'analyse des risques qui est rendue nécessaire par le fait que, lors d'un développement cyclique, il y'a plus de risques de défaire, au cours de l'itération, ce qui a été fait au cours de l'itération précédente.

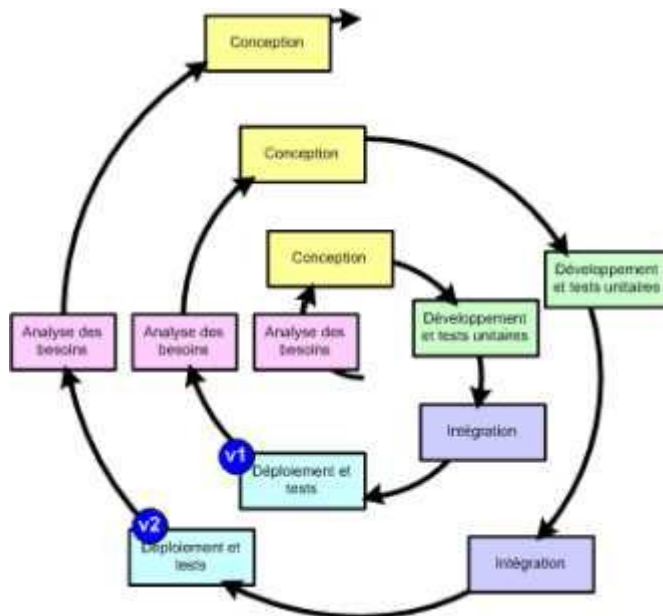


Figure 8: Cycle en spirale

d. Limites des méthodes classiques

Conçues pour piloter un projet par l'élaboration de plans, les méthodes classiques poussent les acteurs d'un projet à refuser systématiquement tout changement, que ce soit des changements dans le contenu ou le périmètre du projet, dans le processus de développement, au sein de l'équipe mais également à toute modification des plans initiaux, auxquels on doit rester conforme. Par conséquent ces méthodes, dites classiques, présentent les inconvénients suivants :

- ❖ **La rigidité de l'approche** On déplore que la nouveauté, la marge de manœuvre laissée, à juste titre, au client pour préciser ou faire évoluer ses attentes, la non-prévisibilité de tous les événements soient difficilement compatibles avec une approche prédictive comme celle du cycle en cascade
- ❖ **L'effet tunnel** qui est dû au fait que le client n'intervient que lors de l'expression des besoins et voit le résultat quelque temps après. Ce qui ne favorise pas la collaboration entre les utilisateurs et les informaticiens surtout si le résultat ne correspond pas aux attentes.
- ❖ **Une mauvaise communication** qui se manifeste par la prohibition par l'absence de jalons intermédiaires de la validation de ce que sera la version finale du projet. Ainsi

les surprises en fin de cycle et le refus de changement par les équipes de développement pénalisent la qualité des relations avec les utilisateurs.

- ❖ **La levée tardive des facteurs à risques** due au fait que les tests de performance et d'intégration sont reportés après le développement.
- ❖ **Une documentation pléthorique** qui permet de repousser la phase de codage qui est, toutefois, irréversible. Elle permet également de s'opposer au changement en brandissant un document contractuel validé précédemment.

II.2.2 Les méthodes agiles

Une méthode agile est une approche itérative et incrémentale qui est menée dans un esprit collaboratif, avec juste ce qu'il faut de formalisme. Elle génère un produit de haute qualité tout en prenant en compte l'évolution des besoins des clients.

L'approche itérative consiste à découper le projet en plusieurs étapes d'une durée de quelques semaines ; ce sont les itérations. Au cours d'une itération, une version minimale du produit attendu est développée puis soumise, dans sa version intermédiaire, au client pour validation. Les fonctionnalités sont ainsi intégrées au fur et à mesure du cycle de vie sur un mode incrémental, le système s'enrichissant progressivement pour atteindre les niveaux de satisfaction et de qualité requis.

Chaque itération est un mini-projet en soi qui comporte toutes les activités de développement, menées en parallèle: analyse, conception, codage et test, sans oublier les activités de gestion de projet. L'objectif est d'obtenir, au terme de chaque itération, un sous ensemble opérationnel du système cible et, au terme de la dernière itération, la version finale du produit. Nous allons présenter ci-dessous trois méthodologies agiles. La première, **Scrum** qui est la méthodologie la plus populaire. La méthodologie **eXtreme Programming (XP)** qui est souvent pratiquée conjointement à **Scrum**. Et enfin la méthodologie **RUP (Rational Unified Process)**. Il existe un grand nombre d'autres méthodologies agiles qui ne seront pas traitées ici car l'objectif n'est pas de présenter de façon exhaustive les méthodologies agiles mais plutôt de mettre en évidence celles qui sont les mieux adaptées ou les plus utilisées dans le développement logiciel. Toutefois, les méthodes suivantes méritent d'être citées : **Rapid Application Development (RAD)** qui est un précurseur des méthodologies agiles, **Dynamic Software Development Method (DSDM)**, **Feature-Driven Development (FDD)**, **Crystal** ou encore **Adaptive Software Development (ASD)**.

Plateforme d'analyse distribuée sur l'écosystème Hadoop : Analyse des risques de crédit bancaire

a. **eXtrem Programming(XP)**

XP ou eXtreme Programming est une méthode de développement agile, orientée projet informatique dont les ressources sont régulièrement actualisées. C'est une méthode de management de projet destinée à accélérer drastiquement la réalisation des projets de type flexible. Une règle fondamentale de la méthode est le fait que le client ou un représentant avisé participe au développement. Le but principal de XP est de réduire les coûts du changement. XP s'attache à rendre le projet plus flexible et ouvert au changement en introduisant des valeurs de base, des principes et des pratiques.

Les principes de cette méthode ne sont pas nouveaux : ils existent dans l'industrie du logiciel depuis des dizaines d'années et dans les méthodes de management depuis encore plus longtemps. L'originalité de la méthode est de les pousser à l'extrême :

- ❖ la revue de code sera faite en permanence (par un binôme),
- ❖ les tests seront faits systématiquement avant chaque mise en œuvre,
- ❖ la conception sera faite tout au long du projet (refactoring),
- ❖ la solution la plus simple sera toujours choisie,
- ❖ les métaphores seront définies,
- ❖ l'intégration des modifications se fera plusieurs fois par jour,
- ❖ des cycles de développement très rapides se feront pour s'adapter au changement.

L'Extreme Programming repose sur des cycles rapides de développement (des itérations de quelques semaines) dont les étapes sont les suivantes :

- une phase d'exploration détermine les scénarios qui seront fournis par le client pendant cette itération,
- l'équipe transforme les scénarios en tâches à réaliser et en tests fonctionnels,
- chaque développeur s'attribue des tâches et les réalise avec un binôme,
- lorsque tous les tests fonctionnels passent, le produit est livré

Le cycle se répète tant que le client peut fournir des scénarios à livrer. Généralement le cycle de la première livraison se caractérise par sa durée et le volume important de fonctionnalités embarquées. Après la première mise en production, les itérations peuvent devenir plus courtes

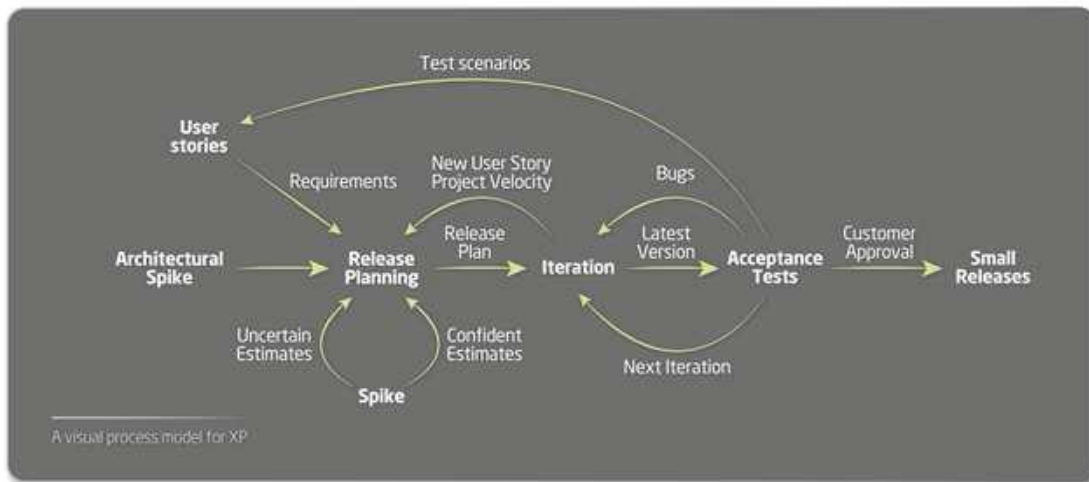


Figure 9: Cycle de vie d'un projet XP

b. Scrum

Scrum tient son origine du terme sportif de rugby signifiant : mêlée. La méthodologie à ses acteurs d'être soudés dans l'accomplissement d'un projet, dans l'atteinte d'un but. Elle utilise une procédure dite itérative (une itération est appelée sprint). Chaque itération ou sprint fournit une partie fonctionnelle du produit. C'est une méthodologie qui est principalement basée sur la gestion des ressources humaines et peut être adaptée à d'autres contextes où il existe un besoin de travailler en équipe. Scrum peut donc être vu comme le cadre de développement venant par exemple appuyer des pratiques XP.

Scrum suppose donc une intense collaboration entre les différentes personnes impliquées. Le directeur de produit (**product owner**) est le représentant du client, il définit les priorités pour la réalisation. Le gestionnaire de projet est nommé **ScrumMaster** : il a pour rôle de faciliter l'application de Scrum par l'équipe.

Le processus Scrum repose sur deux journaux ou "backlog" :

- ❖ **Product Backlog**: liste regroupant les exigences du client. Ce référentiel évolue au cours du développement pour prendre en charge au mieux les besoins du client.
- ❖ **Sprint Backlog**: recense les tâches du Sprint en cours.

Un projet utilisant Scrum a son cycle de vie composé de Sprints successifs. Un Sprint dure au plus quatre semaines. Pendant un Sprint, des réunions quotidiennes de moins de 15 minutes (appelées Scrum) permettent à toute l'équipe de faire le point sur le travail accompli par

chacun depuis la dernière réunion Scrum, les obstacles rencontrés, et le travail prévu d'ici la prochaine réunion.

Pendant un Sprint l'équipe développe un produit partiel. Elle déroule toutes les activités nécessaires pour cela : analyser, concevoir, développer, tester, documenter, intégrer. Chaque Sprint se termine par une revue de Sprint, pour que le directeur de produit évalue, au cours d'une démonstration, le produit partiel obtenu et modifie au besoin le backlog de produit.

En définitive, Scrum introduit des règles pour suivre un processus itératif empirique permettant d'obtenir un produit très proche des besoins qui évoluent et ainsi de maximiser la valeur pour les clients.



Figure 10: Fonctionnement de Scrum

c. Rational Unified Process (RUP)

Cette méthode qui peut être considérée comme la moins agile des méthodes présentées ici, est un mélange des pratiques issues des méthodes traditionnelles et des méthodes agiles. Le principe est de parcourir un cycle de vie (inspection, élaboration, construction, transition) durant une itération. Chaque phase du cycle de vie est très précisément détaillée. Son approche assez lourde et le coût d'investissement de cette méthode la réserve à des projets de grande ou moyenne taille.

II.3 Choix de la méthodologie

Un projet Big Data est souvent caractérisé par la finalité qui porte sur les données et la valeur que l'on peut en tirer. Il s'agit non pas d'un processus et de ses étapes mais d'un résultat et du contenu de celui-ci. Les données sont donc au cœur d'un projet Big Data et comme tout projet, il s'agit de répondre à un besoin défini dans des délais fixés et dans les limites d'un budget alloué.

L'organisation d'un projet Big Data dépend donc de l'objectif, du contexte et du périmètre à couvrir. Ainsi, dans le cadre de notre projet, il s'agit de mettre en place une plateforme d'analyse distribuée de données, pour analyser les risques de crédit bancaire et visualiser les résultats.

Cependant, dans ce type de projet, les méthodologies classiques sont déconseillées. Il faut au contraire construire la solution progressivement, en prévoyant quelques itérations comprenant des interactions avec les utilisateurs finaux. Un dialogue permanent avec les utilisateurs des données doit être établi car les données sont au cœur des attentes.

Les méthodes itératives et incrémentales sont donc adaptées à ce type de projet. Il s'agit de diviser le projet en incréments, c'est-à-dire en parties fonctionnelles cohérentes, chaque incrément pouvant être testé séparément et faisant l'objet de plusieurs itérations. L'objectif est d'impliquer les utilisateurs dans le développement, la fourniture des exigences et l'évaluation des itérations.

Par conséquent, il est clair que la méthodologie la mieux adaptée se doit d'être agile. Ainsi à partir des méthodologies étudiées précédemment, la méthodologie pour répondre aux besoins de notre projet résulte d'un mélange entre eXtreme Programming (XP) et Scrum. XP sera utilisé comme cœur de la méthodologie parce qu'elle permet de réduire le coût du changement et rend le projet plus flexible. Quant au choix de Scrum, il est dû à sa simplicité qui en fait une méthodologie facile d'utilisation et peu contraignante et de plus Scrum offre une excellente vision sur le management de projet. Par conséquent, en combinant les caractéristiques de XP à Scrum, nous obtenons la méthodologie la mieux adaptée à notre projet.

■ Chapitre 3 : Big Data et l'écosystème Hadoop

Dans ce chapitre nous allons présenter le concept Big Data en se focalisant sur sa définition et ses enjeux ensuite nous allons présenter le framework Hadoop et les bases de données NoSQL.

III.1 Définitions du Big Data

III.1.1 Définition technologique du Big Data

La définition initiale donnée par le cabinet **McKinsey and Company** en 2011 s'orientait d'abord vers la question technologique, avec la célèbre règle des 3V : un grand **Volume** de données, une importante **Variété** de ces mêmes données et une **Vitesse** de traitement s'apparentant parfois à du temps réel. Ces technologies étaient censées répondre à l'explosion des données dans le paysage numérique (**data deluge**). Puis, ces qualificatifs ont évolué, avec une vision davantage économique portée par le 4ème V de la définition, celui de **Valeur**, et une notion qualitative véhiculée par le 5e V, celui de **Véracité** des données (disposer de données fiables pour le traitement).

Ces cinq éléments ont servi pendant longtemps de boîte à outils pour comprendre les fondements du Big Data, à savoir l'apparition de technologies innovantes capables de traiter en un temps limité de grands volumes de données afin de valoriser l'information non exploitée de l'entreprise.

III.1.2 Une révolution économique

Certains considèrent que le Big Data s'apparente à une véritable révolution industrielle parce qu'on se trouve en présence d'un vrai bouleversement qui se généralise à tous les secteurs économiques : la donnée joue le rôle de matière première, les technologies jouent celles d'outils de production, et le résultat est un changement de paradigme dans la manière d'organiser les entreprises et de générer de la valeur dans tous les domaines de l'économie. Une comparaison osée et peut-être excessive, mais qui traduit bien l'impact global du Big Data et sa nature autant économique que technologique.

Plateforme d'analyse distribuée sur l'écosystème Hadoop : Analyse des risques de crédit bancaire

III.2 Paysage Technologique du Big Data

Les données, quel que soit leur structure, passent par plusieurs étapes avant que leur valeur ne soit perceptible. Par conséquent, l'univers technologique du Big Data s'appuie sur des outils bien identifiés pour pouvoir exploiter la valeur des données.

III.2.1 Apache Hadoop

Emblème par excellence du Big Data, Hadoop est un ensemble d'algorithmes (Framework open source écrit en Java) pour le stockage et le traitement distribué de très grands ensembles de données sur des clusters d'ordinateurs. Il a été créé par Doug Cutting et Michael Cafarella en 2005. Tous les modules de Hadoop sont conçus avec une hypothèse fondamentale que les défaillances matérielles sont monnaie courante et donc devraient être automatiquement gérées dans le logiciel par le Framework.

Le noyau d'Apache Hadoop se compose d'une partie de stockage (Hadoop Distributed File System (HDFS)) et une partie de traitement (MapReduce ou YARN).

Hadoop divise les fichiers en grands blocs (de 64 Mo ou 128 Mo par défaut) et distribue les blocs parmi les nœuds du cluster. Pour traiter les données, Hadoop/MapReduce transfère des codes (spécifiquement des fichiers JAR) aux nœuds qui ont les données nécessaires qu'ils traitent en parallèle. Cette approche prend l'avantage de l'emplacement des données pour permettre aux données d'être traitées plus rapidement et plus efficacement via le traitement distribué qu'en utilisant une architecture de superordinateur plus classique qui s'appuie sur un système de fichiers parallèle où le calcul et les données sont connectées via un réseau haut débit.

La base du Framework Apache Hadoop est constituée de :

- ❖ **Hadoop Common** qui contient les bibliothèques et les utilitaires nécessaires pour les autres modules d'Hadoop.
- ❖ **Hadoop Distributed File System (HDFS)** qui est un système de fichiers distribué qui stocke les données sur les machines, en fournissant une très haute bande passante globale sur le cluster.

- ❖ **Hadoop YARN** qui est une plate-forme de gestion des ressources chargée de la gestion des ressources de calcul dans des clusters et de les utiliser pour la planification des applications des utilisateurs.
- ❖ **Hadoop MapReduce** qui est un modèle de programmation pour le traitement de données à grande échelle.

Pour la planification efficace du travail, chaque système de fichiers compatible Hadoop devrait fournir un emplacement: le nom du rack (plus précisément, du commutateur de réseau), où se trouve un nœud travailleur. Les Applications Hadoop peuvent utiliser cette information pour exécuter les travaux sur le nœud où se trouvent les données, et, à défaut, sur le même commutateur, réduisant le trafic du réseau cœur.

Un petit cluster Hadoop comprend un seul nœud maître et plusieurs nœuds travailleurs. Le nœud maître a trois principaux rôles que sont :

- ❖ Le **JobTracker** qui permet au nœud maître de lancer des tâches distribuées, en coordonnant les esclaves. Il planifie les exécutions, gère l'état des machines esclaves et agrège les résultats des calculs.
- ❖ Le **NameNode** qui assure la répartition des données sur les machines esclaves et la gestion de l'espace de nom du cluster. La machine qui joue ce rôle contient des métadonnées qui lui permettent de savoir sur quelle machine chaque fichier est hébergé.
- ❖ Le **SecondaryNameNode** intervient pour la redondance du NameNode. Normalement, il doit être assuré par une autre machine physique autre que le NameNode car il permet en cas de panne de ce dernier, d'assurer la continuité de fonctionnement du cluster

Un nœud esclave ou nœud travailleur agit à la fois comme :

- ❖ un **DataNode** qui est une machine qui héberge une partie des données. Les DataNode sont généralement répliqués dans le cadre d'une architecture Hadoop dans l'optique d'assurer la haute disponibilité des données.

- ❖ un **TaskTracker** qui permet à un esclave d'exécuter une tâche MapReduce sur les données qu'elle héberge. Le TaskTracker est piloté par le JobTracker d'une machine maître qui lui envoie la tâche à exécuter.

Il est possible d'avoir des nœuds travailleurs pour des données uniquement et des nœuds de travail orientés calcul uniquement. Hadoop nécessite Java Runtime Environment (JRE) 1.6 ou une version supérieure. Les scripts de démarrage et d'arrêt standards exigent que Secure Shell (SSH) soit mis en place entre les nœuds dans le cluster.

Dans un cluster plus grand, HDFS est géré par un serveur NameNode dédié à l'hébergement de l'index du système de fichiers, et un NameNode secondaire qui peut générer des captures instantanées des structures de mémoire du NameNode, empêchant ainsi la corruption du système de fichiers et la réduction de la perte des données. De même, un serveur autonome JobTracker peut gérer la planification des travaux.

Dans des clusters où le moteur Hadoop MapReduce est déployé sur un autre système de fichiers, le NameNode, NameNode secondaire, et l'architecture de DataNode de HDFS sont remplacés par les équivalents du système de fichiers spécifiques.

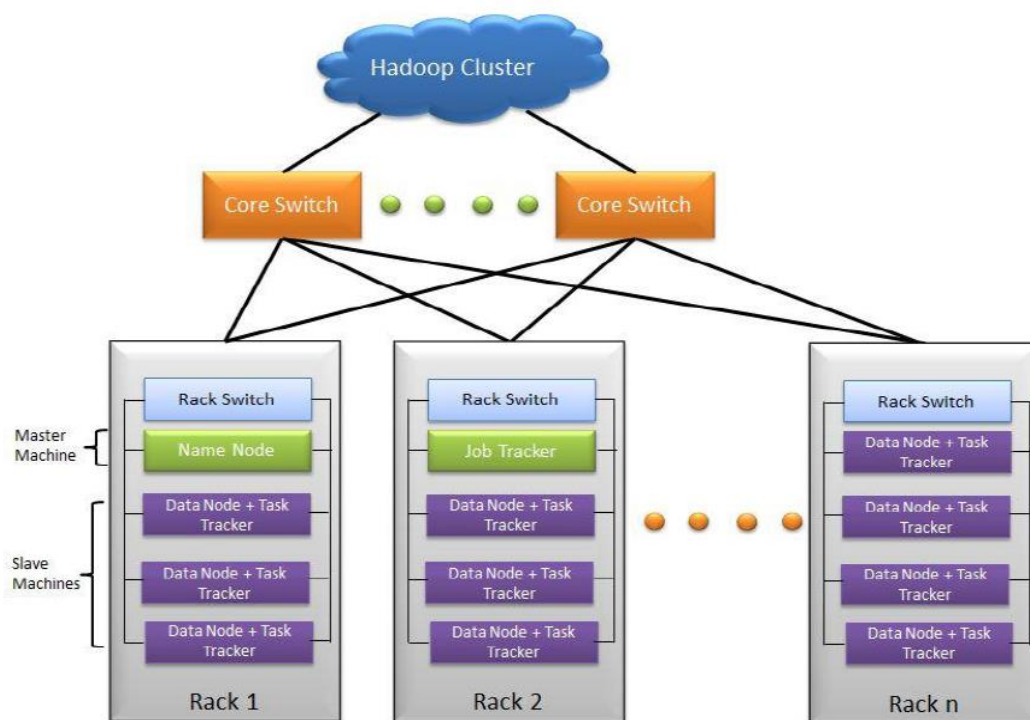


Figure 11: Cluster Hadoop multinœud

III.2.2 Apache MapReduce

C'est un Framework qui permet d'écrire facilement des applications traitant en parallèle de grandes quantités de données sur de grands clusters de milliers de machines, de façon fiable et insensible aux pannes.

Un job MapReduce divise généralement l'ensemble des données d'entrée en morceaux indépendants qui sont traités par les tâches Map d'une manière complètement parallèle. Le Framework trie les résultats des tâches Map qui sont ensuite des entrées pour les tâches Reduce. Typiquement, l'entrée et la sortie de la tâche sont stockées dans un système de fichiers. Le Framework s'occupe de la planification des tâches, de leur suivi et ré-exécute les tâches qui ont échoué.

Généralement, les nœuds de calcul et les nœuds de stockage sont les mêmes. Le Framework MapReduce et le système de fichiers Hadoop s'exécutent sur le même ensemble de nœuds. Cette configuration permet au Framework de planifier efficacement les tâches sur les nœuds où les données sont déjà présentes, aboutissant à une très haute bande passante à travers le cluster.

Le client d'un job Hadoop soumet le job et une configuration au JobTracker qui assure la distribution du logiciel ou de la configuration aux esclaves, la planification des tâches et leur suivi, en fournissant l'état et les informations de diagnostic au job client. Bien que le Framework Hadoop soit implémenté en Java, les applications MapReduce n'ont pas besoin d'être toujours écrites en Java.

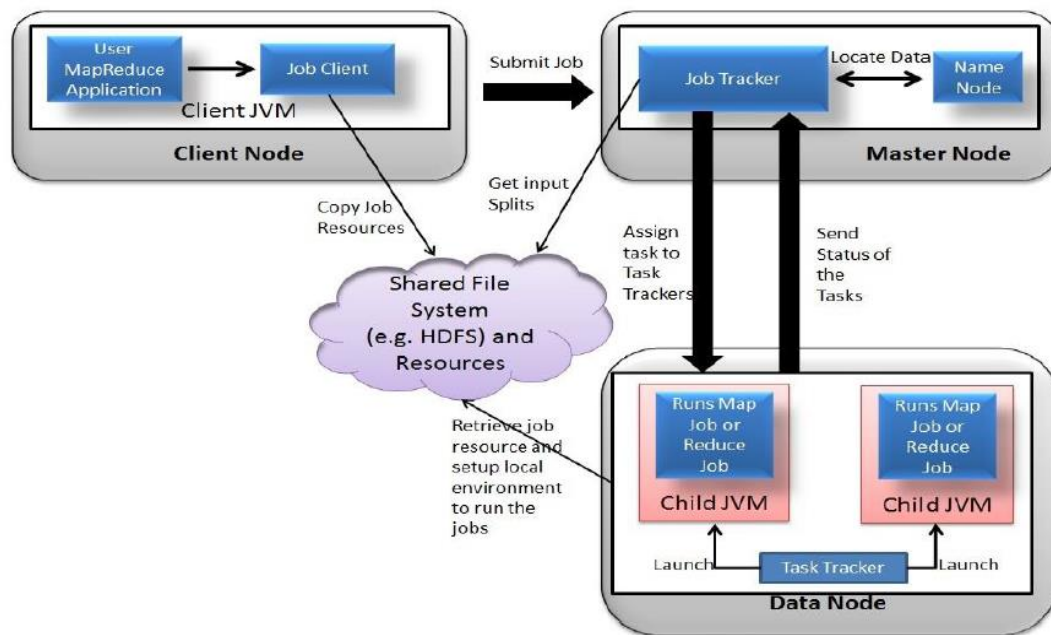


Figure 12: Architecture de fonctionnement de MapReduce

III.2.3 Hadoop YARN

MapReduce a subi une transformation dans Hadoop-0.23 et nous avons maintenant, ce que nous appelons, MapReduce 2.0 (MRv2) ou YARN.

L'idée fondamentale de MRv2 est de diviser les deux grandes fonctionnalités du JobTracker, la gestion des ressources et la planification ou le suivi de job, en daemons distincts. L'idée est d'avoir un ResourceManager (RM) global et une ApplicationMaster par application. Une application est soit un seul job dans le sens classique des jobs MapReduce ou un DAG de jobs.

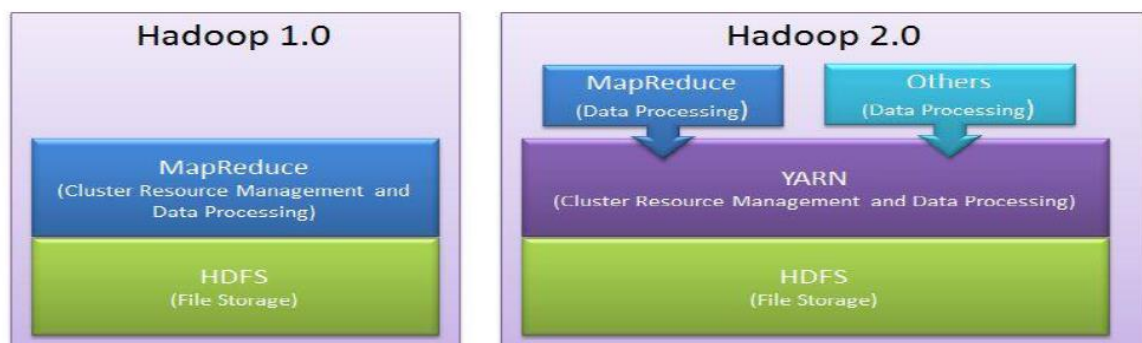


Figure 13: Evolution de l'architecture MapReduce vers YARN

Le **ResourceManager** et le **NodeManager** (NM) forment le Framework de calcul de données. Le ResourceManager est l'autorité ultime qui gère les ressources entre toutes les applications dans le système.

L'ApplicationMaster est, en effet, une bibliothèque spécifique d'un Framework et est chargé de la négociation des ressources du ResourceManager et de travailler avec le NodeManager pour exécuter et suivre les tâches.

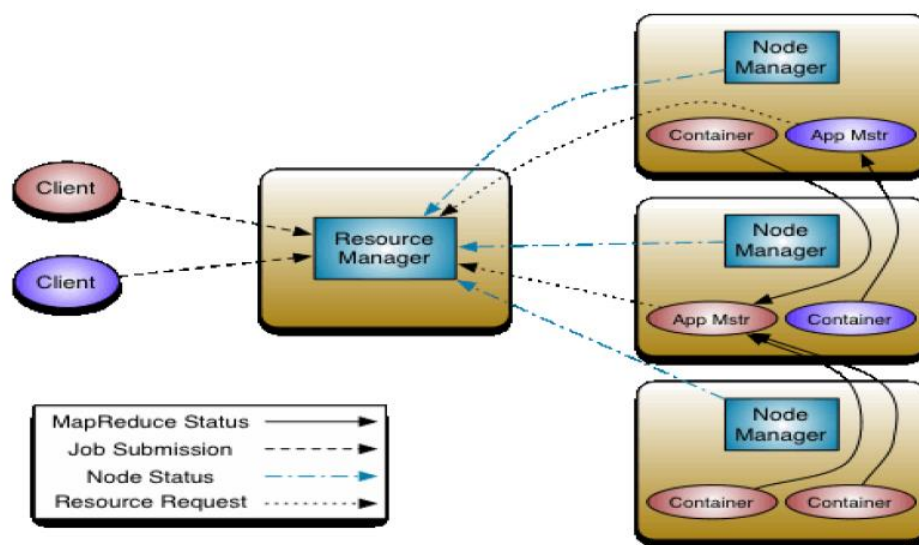


Figure 14: Architecture de fonctionnement de YARN

Le ResourceManager a deux composants essentiels : le **planificateur** et l'**ApplicationsManager**.

Le **planificateur** est responsable de l'allocation des ressources aux différentes applications en cours soumises à des contraintes familières de capacités, de files d'attente, etc. Le **planificateur** est un pur ordonnanceur en ce sens qu'il n'effectue pas de contrôle ou de suivi de l'état de l'application. En outre, il n'offre aucune garantie sur le redémarrage des tâches qui ont échoué en raison de l'échec de l'application ou du matériel. Le planificateur améliore sa fonction de planification basée sur les besoins en ressources des applications. Il le fait en se basant sur la notion abstraite d'un conteneur de ressources qui intègre des éléments tels que la mémoire, le CPU, le disque, le réseau, etc.

Le planificateur a un plug-in enfichable, qui est responsable du partitionnement des ressources du cluster parmi les différentes files d'attente, applications, etc. Les ordonnanceurs courants de MapReduce tels que le CapacityScheduler et le FairScheduler seraient des exemples de plug-in.

Le CapacityScheduler supporte les files d'attente hiérarchiques pour permettre un partage plus prévisible des ressources du cluster. L'**ApplicationsManager** est responsable de l'acceptation des soumissions de job, en négociant le premier conteneur pour l'exécution de l'**ApplicationMaster** d'une application spécifique et fournit le service pour redémarrer le conteneur **ApplicationMaster** en cas d'échec.

Le NodeManager est l'agent de Framework qui est responsable pour les conteneurs, en suivant leur utilisation des ressources (CPU, mémoire, disque, réseau) et en les rapportant au ResourceManager.

L'ApplicationMaster a la responsabilité de négocier des conteneurs de ressources appropriés à partir de l'ordonnanceur, de surveiller leur état et de suivre leur évolution.

III.2.4 Hadoop Distributed File System (HDFS)

Le système de fichiers distribué de Hadoop (HDFS) est un système de fichiers distribué, évolutif, portable, écrit en Java pour le Framework Hadoop.

Un cluster Hadoop a nominalelement un seul NameNode plus un cluster de DataNode, bien que des options de redondance soient disponibles pour le NameNode en raison de sa criticité. Chaque DataNode sert des blocs de données sur le réseau en utilisant un protocole de bloc spécifique à HDFS. Le système de fichiers utilise des sockets TCP / IP pour la communication. Les clients utilisent des appels de procédure distante (RPC) pour communiquer entre eux.

HDFS stocke des fichiers volumineux entre plusieurs machines. Il réalise la fiabilité en répliquant les données entre plusieurs hôtes, et donc théoriquement ne nécessite pas de stockage RAID sur les hôtes (mais pour augmenter les performances d'E / S des configurations RAID sont toujours utiles). Avec la valeur de réplication par défaut, 3, les données sont stockées sur trois nœuds: deux sur le même châssis, et une autre sur un châssis différent. Des Nœuds de données peuvent communiquer entre eux pour rééquilibrer des données, pour déplacer des copies autour, et pour garder la réplication des données de haut. HDFS n'est pas totalement compatible POSIX, car les exigences relatives à un système de

fichiers POSIX diffèrent des objectifs cibles d'une application Hadoop. Le compromis de ne pas avoir un système de fichiers totalement compatible POSIX est une performance accrue pour un débit de données et un soutien pour les opérations non-POSIX tels que Append.

HDFS a ajouté les capacités de haute disponibilité, annoncé pour la version 2.0 en mai 2012, en laissant le serveur de métadonnées principal (NameNode) basculer manuellement vers une sauvegarde. Le projet a également commencé à développer le fail over automatique.

Le système de fichiers HDFS comprend un NameNode dit secondaire, un nom trompeur que certains pourraient mal interpréter comme un NameNode de sauvegarde lorsque le NameNode primaire se met hors ligne. En fait, le NameNode secondaire se connecte régulièrement sur le NameNode primaire et construit des captures instantanées des informations du répertoire du NameNode primaire, que le système enregistre ensuite dans des répertoires locaux ou distants. Ces images de points de reprise peuvent être utilisées pour redémarrer un NameNode principal défaillant sans avoir à refaire l'ensemble des actions du journal de système de fichiers, puis d'éditer le journal pour créer une structure de répertoire mis à jour. Puisque le NameNode est le seul point pour le stockage et la gestion des métadonnées, il peut devenir un goulot d'étranglement pour supporter un grand nombre de fichiers, notamment un grand nombre de petits fichiers. Une Fédération HDFS, un nouvel ajout, vise à s'attaquer à ce problème dans une certaine mesure en permettant à plusieurs espaces de noms desservies par des NameNode distincts.

Un avantage d'utiliser HDFS est la prise de connaissance des données entre le JobTracker et le TaskTracker. Les JobTracker planifient aux TaskTracker les jobs Map ou Reduce avec la prise de connaissance de l'emplacement des données. Cela réduit la quantité de trafic qui passe sur le réseau et empêche le transfert de données inutiles. Lorsque Hadoop est utilisé avec d'autres systèmes de fichiers, cet avantage n'est pas toujours disponible. Cela peut avoir un impact significatif sur les temps d'achèvement des jobs, qui a été démontré lors de l'exécution des tâches à données intensives.

Un principe important de HDFS est que les fichiers sont de type « write-once » ; ceci est lié au fait que lors des opérations analytiques, la lecture des données est beaucoup plus utilisée que l'écriture.

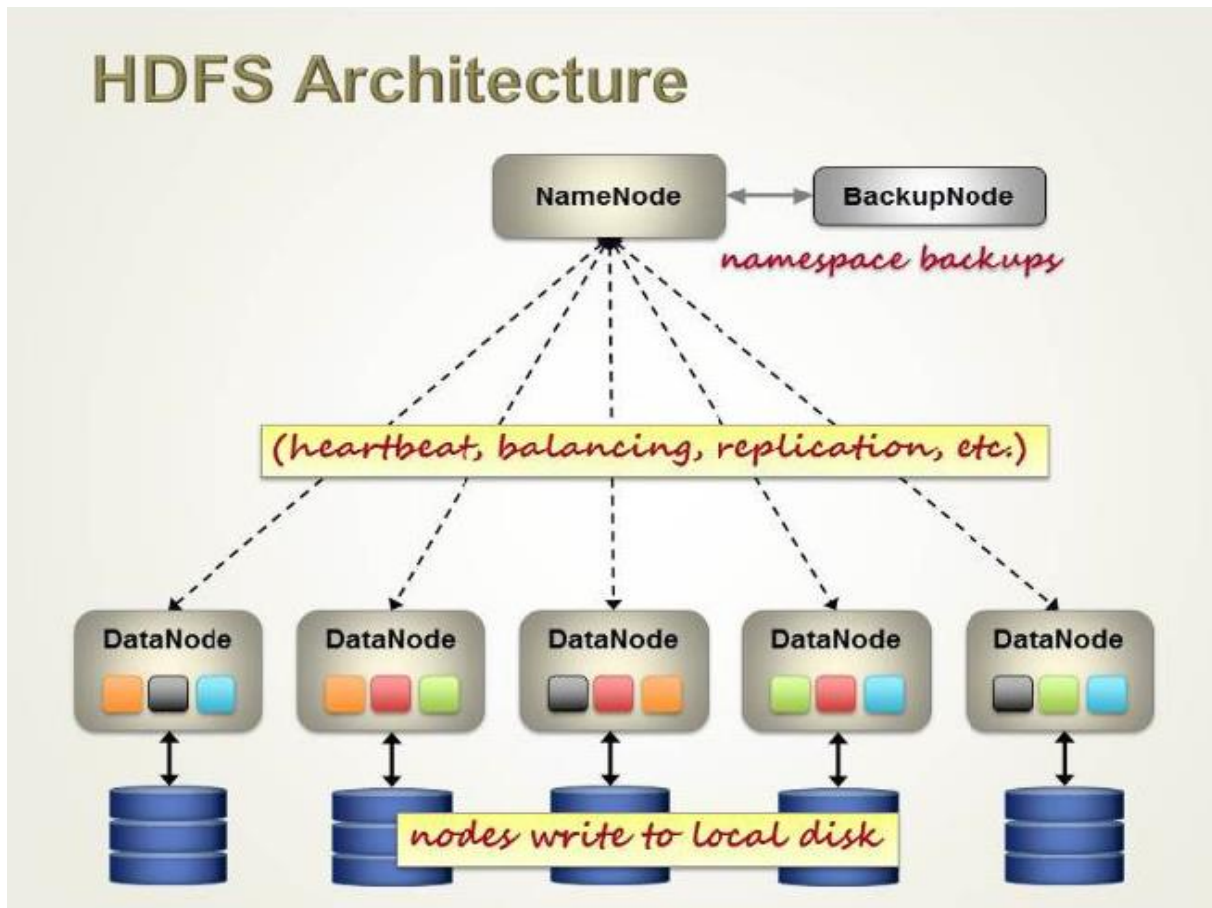


Figure 15: Architecture HDFS

III.3 L'écosystème Hadoop

Au-delà de ces outils de base que nous venons de présenter ci-dessus, le Big Data se doit de proposer des services en lien avec les besoins directs de l'entreprise. Il est donc possible de rajouter des briques fonctionnelles aux spécifications initiales de la technologie Hadoop. C'est tout l'intérêt de l'écosystème Hadoop qui s'est développé en parallèle de ces outils.

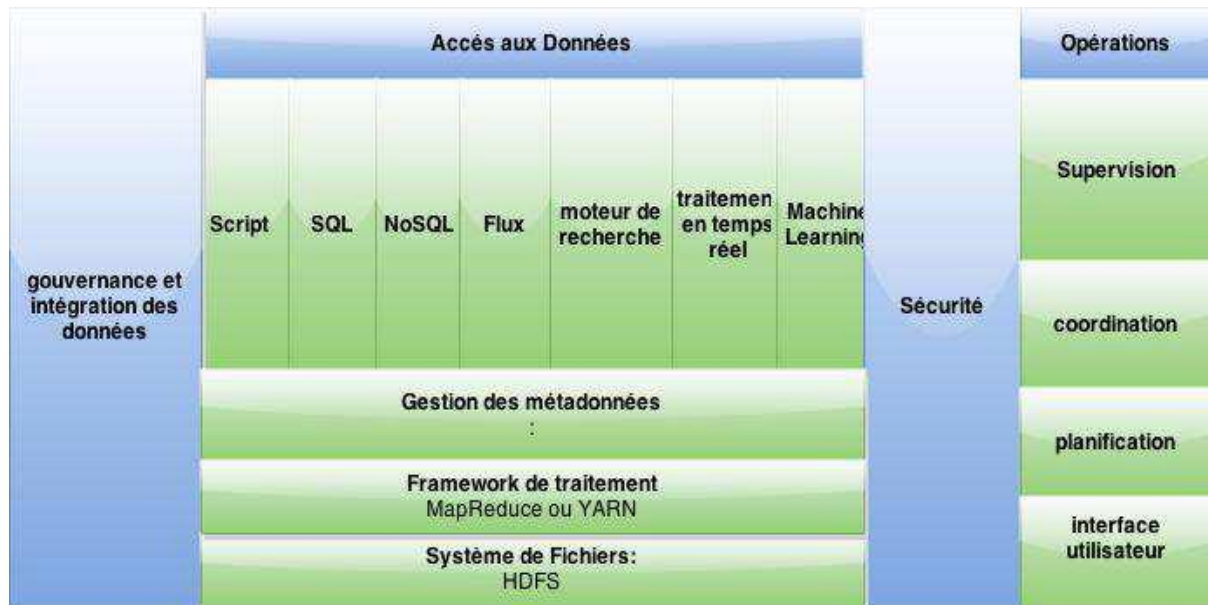


Figure 16: Architecture de l'écosystème Hadoop

❖ La gestion des données

C'est la couche qui englobe toutes les technologies qui permettent de stocker et de traiter de grandes quantités de données dans une couche de stockage qui varie linéairement.

Dans la **figure 11**, cette couche est représentée par le système de fichier HDFS et le framework de traitement des données qui peut être MapReduce ou YARN.

❖ La gouvernance et l'intégration des données

C'est la couche où se situent les différentes technologies qui permettent de charger les données rapidement et facilement puis de gérer celles-ci conformément aux politiques.

❖ L'accès aux données

C'est la couche qui permet d'interagir avec les données par une multitude de modes : par batch, en temps réel, etc. Elle est constituée des technologies qui interagissent avec les données via un script, via le langage SQL ou le langage NoSQL et aussi les technologies qui permettent de traiter les données par flux, en temps réel ou encore les moteurs de recherche et les outils de machine learning.

❖ Sécurité

Cette couche regroupe les différentes technologies qui permettent de répondre aux exigences d'authentification, d'autorisation, de comptabilité et de protection des données.

❖ Opérations

C'est la couche destinée aux outils qui permettent de déployer, gérer, surveiller et exploiter les clusters Hadoop de façon adaptée. C'est ainsi qu'elle est découpée en 4 couches qui sont :

- ✓ **Supervision** : déploiement, gestion et contrôle des clusters Hadoop
- ✓ **Coordination** : coordination des tâches au niveau du cluster Hadoop
- ✓ **Planification** : planification de flux de travail
- ✓ **Interface utilisateur** : interface Web pour l'interaction avec Hadoop et pour l'analyse de données.

III.4 Les systèmes classiques de gestion de bases de données et ses limites

Les systèmes classiques de gestion de bases de données ont les spécificités suivantes :

- Basé sur le modèle relationnel
- Un langage de requêtes standard : SQL
- Données stockées sur disque
- Relations (tables) stockées ligne par ligne
- Système centralisé, avec possibilités limitées de distribution

a. Forces des systèmes de gestion de bases de données relationnels classiques

- Indépendance entre :
 - Modèle de données et structures de stockages.
 - Requêtes déclaratives et exécution
- Requêtes complexes
- Optimisation très fine des requêtes, index permettant un accès rapide aux données
- Logiciels mûrs, stables, efficaces, riches en fonctionnalités et en interfaces
- Contraintes d'intégrité permettant d'assurer des invariants sur les données
- Gestion efficace de grands volumes de données (gigaoctet, voire téraoctet)
- Transactions (ensembles d'opérations élémentaires) garantissant la gestion de la concurrence, l'isolation entre utilisateurs, la reprise sur panne.

b. Propriété ACID des systèmes de gestion de bases de données relationnels classiques

Les transactions des SGBD relationnels classiques respectent les propriétés ACID :

- Atomicité : L'ensemble des opérations d'une transaction est soit exécuté en bloc, soit annulé en bloc
- Cohérence : Les transactions respectent les contraintes d'intégrité de la base
- Isolation : Deux exécutions concurrentes de transactions résultent en un état équivalent à l'exécution sérielle des transactions
- Durabilité : Une fois une transaction confirmée, les données correspondantes restent durablement dans la base, même en cas de panne

c. Limites des systèmes de gestion de bases de données relationnels classiques

- Incapable de gérer de très grands volumes de données (de l'ordre du péta-octet)
- Impossible de gérer des débits extrêmes (plus que quelques milliers de requêtes par seconde)
- Le modèle relationnel est parfois peu adapté au stockage et à l'interrogation de certains types de données (données hiérarchiques, faiblement structurées, semi-structurées)
- Les propriétés ACID entraînent de sérieux surcoûts en latence, accès disques, temps CPU (verrous, journalisation, etc.)
- Performances limitées par les accès disque

III.5 Le NoSQL

NoSQL désigne une famille de systèmes de gestion de base de données (SGBD) qui s'écarte du paradigme classique des bases relationnelles. L'explicitation du terme la plus populaire de l'acronyme est Not only SQL (« pas seulement SQL » en anglais) même si cette interprétation peut être discutée. Développé à l'origine pour gérer du big data, l'utilisation de base de données NoSQL a explosée depuis quelques années.

a. Pourquoi le NoSQL

Le système de gestion de bases de données relationnel (SGBDR) a été la norme de facto pour la gestion des données depuis sa première apparition d'IBM au milieu des années 1980. Le SGBDR a vraiment explosé dans les années 1990 avec Oracle, Sybase, Microsoft SQL Server et d'autres bases de données similaires apparaissant dans les centres de données de presque toutes les entreprises.

Avec la première vague d'applications Web, les systèmes de gestion de bases de données relationnelles open source (SGBDR) tels que MySQL et Postgres ont émergé et sont devenus Plateforme d'analyse distribuée sur l'écosystème Hadoop : Analyse des risques de crédit bancaire

une norme dans de nombreuses entreprises qui souhaitaient des alternatives aux bases de données propriétaires coûteuses vendues par des fournisseurs tels que Oracle.

Cependant, il ne fallait pas longtemps avant que les choses ne changent, les besoins en applications et les data centers des géants du web comme Amazon, Facebook et Google ont commencé à dépasser le SGBDR pour certains types d'applications. La nécessité de disposer de modèles de données plus souples qui prennent en charge les méthodologies de développement agiles et les exigences pour consommer de nombreuses quantités de données rapides provenant de millions d'utilisateurs d'applications en nuage dans le monde entier - tout en conservant des performances et des temps de disponibilité extrêmes - nécessitaient l'introduction de nouvelles Plate-forme de gestion de données : Le **NoSQL**

Aujourd'hui, avec toutes les entreprises utilisant des applications modernes sur le cloud, les problèmes de données rencontrés à l'origine par les géants Internet sont devenus des problèmes communs pour chaque entreprise.

b. Caractéristiques des SGBD NoSQL

Les caractéristiques principales des SGBD NoSQL sont de permettre la manipulation de volumes de données importants et de permettre une scalabilité horizontale. Ces systèmes ne respectent en général pas les standards des SGBD relationnels, mais il ne s'agit pas à proprement parler d'une propriété recherchée mais plus d'une concession permettant des traitements plus rapides pour certains types d'applications.

- SGBD avec d'autres compromis que ceux faits par les systèmes classiques
- Écosystème très varié
- Fonctionnalités recherchées : modèle de données différent, passage à l'échelle, performances extrêmes
- Fonctionnalités abandonnées : ACID, (parfois) requêtes complexes

Ces systèmes fonctionnent avec un modèle de données différent

c. Les types de bases de données NoSQL

Il existe différents types de bases de données NoSQL, la différence principale étant caractérisée par leur modèle de données sous-jacent et leur méthode de stockage des données. Les principales catégories de bases de données NoSQL sont:

i. Les systèmes orientés colonnes

- Au lieu de stocker les données ligne par ligne, les stocker colonne par colonne

- Organisation plus riche que dans les systèmes clef-valeur (plusieurs colonnes par objet stocké)
- Rendent plus efficace l'agrégation ou le parcours des valeurs d'une même colonne
- Distribution transparente, passage à l'échelle grâce à des arbres de recherche distribués ou des tables de hachages distribués



ii. Systèmes orientés document

- Requêtes toujours très simples
 - **get** récupère le document (JSON, XML, YAML. . .) associé à une clef
 - **put** ajoute un nouveau document associé à une clef
- Des index additionnels permettant de récupérer les documents contenant tel mot-clef, ayant telle propriété, etc.
- Documents organisés en collections, gestion de méta-données (versions, dates), etc.
- Accent mis sur la simplicité de l'interface, la facilité de manipulation dans un langage de programmation



iii. Systèmes clé-valeur

- Requêtes très simples :
 - **get** récupère la valeur associée à une clé
 - **put** ajoute un nouveau couple clé/valeur
- Accent mis sur le passage à l'échelle transparent, une faible latence, un débit très élevé
- Exemple d'implémentation : table de hachage distribuée



d. Les avantages du NoSQL sur les SGBDR classiques

Bien qu'il existe des centaines de bases de données différentes NoSQL offertes aujourd'hui, chacune avec ses propres fonctionnalités et avantages particuliers, il faut noter qu'une base de données NoSQL diffère généralement d'un SGBDR traditionnel de la manière suivante :

- **Modèle de données** : tandis qu'un SGBDR gère principalement les données structurées dans un modèle de données rigide, une base de données NoSQL fournit généralement un modèle de données plus flexible et plus fluide et peut être plus apte à servir les méthodologies de développement agiles utilisées pour les applications cloud modernes. Notez qu'une idée fausse concernant les modèles de données NoSQL est qu'ils ne gèrent pas les données structurées, ce qui est faux. Enfin, comme indiqué ci-dessus, certains moteurs NoSQL prennent en charge plusieurs modèles de données contre un seul backend.
- **Architecture** : alors qu'un RDBMS est normalement structuré d'une manière centralisée, à grande échelle et à maître-esclave, les systèmes NoSQL tels que Cassandra opèrent de manière distribuée, échelonnée et «masterless» (c'est-à-dire qu'il n'y a pas de nœud «maître»). Cependant, certaines bases de données NoSQL (par exemple, MongoDB, HBase) sont conçues en Master Slave.
- **Modèle de distribution de données** : en raison de leurs architectures maître-esclave, un SGBDR distribue des données aux machines esclaves qui peuvent servir de copies en lecture seule des données et / ou du basculement pour la machine primaire. En revanche, une base de données NoSQL comme Cassandra distribue uniformément les données à tous les nœuds constituant un cluster de base de données et permet à la fois de lire et d'écrire sur toutes les machines. En outre, le modèle de réplication d'un SGBDR (y compris le master-to-master) n'est pas conçu pour une réplication et une synchronisation à grande échelle, multi-géographique, et une synchronisation de données entre différentes zones locales et zones de disponibilité cloud, alors que la réplication de Cassandra a été construite à partir de la base pour gérer de telles choses.

- **Modèle de disponibilité** : un SGBDR utilise généralement une conception de basculement où un master échoue sur une machine esclave, alors qu'un système NoSQL comme Cassandra est sans master et fournit une redondance de données et de fonctions sur chaque nœud afin qu'il offre une disponibilité continue sans temps d'arrêt.
- **Modèle de scalabilité et de performance** - un SGBDR est généralement scalable de manière verticale en ajoutant une CPU supplémentaire, une RAM, etc., à une machine centralisée, tandis qu'une base de données NoSQL est scalable horizontalement en ajoutant des nœuds supplémentaires qui offrent une échelle et une performance accrues de manière linéaire.

■ Chapitre 4 : Analyse et Conception de la solution

Dans ce chapitre, nous allons aborder la phase d'analyse et de spécifications fonctionnelles. Ainsi nous allons proposer une architecture de la plateforme.

IV.1 Spécifications fonctionnelles

Dans cette partie nous allons lister les spécifications fonctionnelles de notre plateforme. Ce qui permettra d'avoir une vue globale des différentes fonctions que devrait avoir notre plateforme.

IV.1.1 Les fonctionnalités générales de notre plateforme

Fonctionnalités de la plateforme :

- ❖ **Collecte de données de différentes sources**
- ❖ **Stockage des données collectées**

Nous allons détailler chacune des fonctionnalités dans la suite.

a. Collecte de données

Cette fonctionnalité consiste à collecter des données de différentes sources accessibles, qui peuvent être entre autres :

- ✓ Les données sur internet et des réseaux sociaux, capteur etc...
- ✓ Les données des transactions financière (wari, joni joni ...)
- ✓ Importation de données d'une base existante

Dans notre cas nous allons nous concentrer sur les données de twitter qui sont disponible avec l'API de twitter, ainsi les données collectées seront stockés dans des topics Kafka.

b. Stockage des données collectées

Cette fonctionnalité consiste à récupérer les données collectées dans les topics Kafka grâce à Spark Streaming mais aussi les données d'une base de données existante avec Sqoop et de les stockées dans les systèmes de fichiers de Hadoop HDFS.

IV.2 Architecture de la plateforme

Notre plateforme sera composée de plusieurs technologies qui forment l'écosystème Hadoop et qui constitue le cœur de la plateforme. Ces différentes technologies sont regroupées sous Plateforme d'analyse distribuée sur l'écosystème Hadoop : Analyse des risques de crédit bancaire

forme de distribution Hadoop pour faciliter le déploiement et éviter certains problèmes liés aux différentes versions et release des différentes technologies. D'autre part nous avons la couche applicative qui est composé des applications clientes et les outils de visualisation.



Figure 17: Architecture en couche

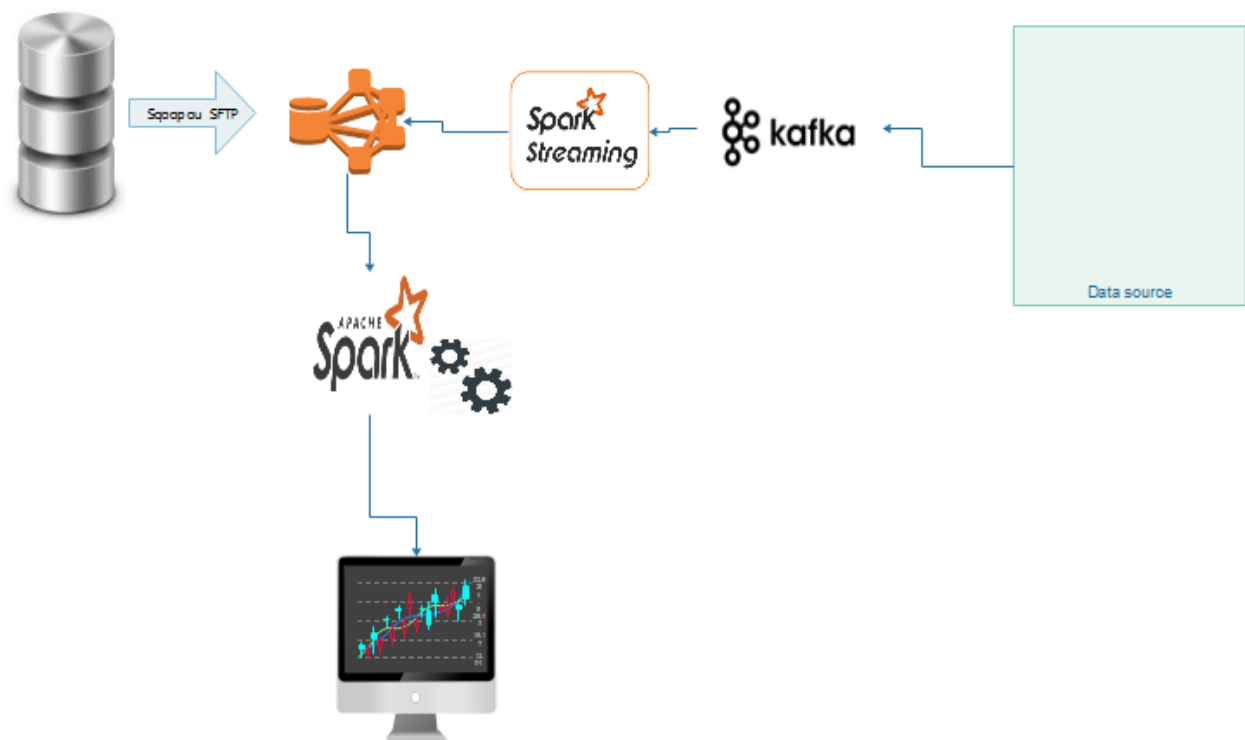


Figure 18: Architecture technique de la plateforme

La figure 17 représente l'architecture en couche de la plateforme, elle décrit les différentes parties telles que la partie Intégration, Accès aux données, Gestion des données, Applicative etc...

La figure 18 représente l'architecture technique de la plateforme, les technologies utilisées ainsi que les relations entre elles.

Les principaux aspects de l'architecture sont :

- ✓ Les sources de données qui sont identifiées
- ✓ Intégration des données au repos et les données en Streaming dans HDFS
- ✓ Appliquer les traitements aux données stockées

■ Chapitre 5 : Risque de crédit et Big Data

Le risque de crédit est le risque qu'un emprunteur ne rembourse pas tout ou une partie de son crédit aux échéances prévues par le contrat signé entre lui et l'organisme prêteur (généralement une banque). Nous allons voir dans ce chapitre comment le risque crédit est analysé avec la méthode scoring.

V.1 Le crédit scoring en général

Le crédit scoring a donc pour but de prédire la probabilité de défaillance d'un client et en conséquence de réduire le risque d'avoir des engagements non honorés. Le Credit Scoring peut être globalement défini comme un ensemble de modèles de décision et les techniques sous-jacentes qui aident les prêteurs dans la décision d'octroi des crédits de consommation¹.

Le Credit Scoring est le processus d'assignation d'une note (ou score) à un emprunteur potentiel pour estimer la performance future de son prêt (Flaman, 1997).

Le Credit Scoring utilise des mesures quantitatives de performance et les caractéristiques des prêts précédents pour prédire performance des prêts futurs avec des caractéristiques similaires. Le Credit Scoring n'approuve, ni ne rejette une demande de prêt, il peut plutôt prédire la probabilité d'occurrence de mauvaise performance (défaut) telle que définie par le prêteur (Caire et Kossmann, 2003).

Comme toute méthode de prévision, le credit scoring s'appuie sur l'historique des résultats de remboursement, des caractéristiques des clients et du type du crédit pour construire une fonction de score qui sera utilisée pour la discrimination (bon payeur / mauvais payeur) des nouveaux demandeurs de crédits.

Le crédit scoring utilise des modèles qui aboutissent à un score d'octroi de crédit à partir d'un seuil fixé par l'institution financière, selon sa tolérance aux risques.

¹ Thomas, Edelman and Crook (2002).

Il y a différentes étapes pour déterminer un score de crédit.

a. La préparation des données

Le processus de crédit, commence d'abord par une phase de collecte d'informations auprès du client, et auprès de sources externes, afin de former le dossier de crédit. Ces informations portent sur la forme (la "qualité" de l'emprunteur, l'équilibre du montage financier, le respect de la réglementation, etc.), le fond (la capacité de remboursement des emprunteurs, les éléments d'appréciation du risque, etc.) et les garanties (cautions, hypothèques, etc.).

Des exemples de facteurs utilisés dans un système traditionnel de crédit scoring comprennent entre autres :

- Nom, Prénom, Age, salaire etc...
- Historique du paiement des obligations passées
- Durée de l'historique de crédit
- Montants dus
- Types de crédit déjà détenu
- Les garanties etc...

b. Quantification et la détermination des variables

Traditionnellement le banquier remplit le dossier du client. Avant la phase de modélisation, les données collectées sont analysées pour déterminer les variables les plus pertinentes et discriminantes des clients.

c. Phase de modélisation

Une fois la phase de préparation des données achevée, commence la phase de modélisation qui permet la construction de la fonction de scores. Différentes techniques de scoring pourront être appliquées pour la construction de la fonction de score.

V.2 Credit scoring et Big Data

Le volume et la variété de l'information disponible en environnement Big Data vont permettre d'enrichir considérablement la connaissance du client, de pouvoir créer des profils de risque en fonction de ses goûts, de son réseau social. Nous pourrions étudier une source inépuisable de données, en temps quasi réel. L'individu ne sera plus vu de manière isolée,

mais en tant qu'élément d'un groupe formant un réseau. Les sources de données peuvent être entre autres :

- **Les systèmes transactionnels à grande échelle** : Ce sont des systèmes tels que OLTP (Transaction en ligne : opérations bancaires, achats de biens, billets, réservations), ERP (Planning des ressources d'entreprise) et CRM (Management des relations clients). Des tableaux de bord sont ainsi construits à partir de données extraites de ces systèmes.
- **Les réseaux sociaux** : Facebook, Twitter, LinkedIn, WeChat etc. ces réseaux sociaux capturent des informations sur près de deux milliards de personnes au sujet de leurs amis, de leurs comportement ce qui représente une immense piste de données.
- **Les objets connectés**

Toutes les sources de données citées ci-dessus offrent un potentiel énorme pour construire de meilleurs modèles de notation de crédit. Ainsi les types de données traditionnelles ou les données structurées telles que le nom du client, la date de naissance du client, etc. sont de plus en plus complétées par des données non structurées telles que les images, les empreintes digitales, les tweets, les courriels, les pages Facebook, les données des capteurs, les données GPS, etc. La vitesse de mise à jour des données, parfois en temps réel, est une excellente aide pour la maîtrise du risque et la prise de décision. Mais un travail important devra être fait en amont sur la méthodologie à employer pour traiter des informations, dont le format est différent du format standard (textes et chiffres), comme les images, les vidéos ou encore les mails. En effet, les données issues de la toile seront en grande partie non structurées, car elles auront potentiellement un nombre de valeurs toutes différentes et difficiles à catégoriser. Pour leur exploitation, elles devront être structurées. Avec l'augmentation considérable du volume de données, les méthodes statistiques usuelles ne pourront plus être appliquées.

V.3 Les méthodes statistiques usuelles et leurs limites

a. La régression Logistique

La régression logistique est une méthode statistique qui s'utilise lorsque l'on cherche à expliquer la survenue d'un événement (par ex. le défaut de paiement) à l'aide de variables explicatives (pouvant être continues ou discrètes) susceptibles d'influencer la survenue de cet événement (par ex. l'âge). La régression logistique est très sensible au problème de multi-

colinéarité partielle, qui se manifeste lorsqu'une variable explicative est fortement corrélée à une ou plusieurs variables explicatives (ou à l'une de leur combinaison).

b. L'arbre de décision

L'arbre de décision est une méthode de segmentation représentée graphiquement comme un arbre, composé de nœuds de décision (les ronds foncés) reliés par des branches et de nœuds terminaux (ronds clairs) qui donnent la décision finale. C'est une méthode statistique peu robuste en présence d'un grand nombre de variables. En effet, plus il y a de variables en entrée, plus le nombre de test augmente. Un mauvais choix de variable au début peut conduire à un arbre avec un faible pouvoir discriminant.

De plus, si le partitionnement est fait sur des variables qualitatives alors le temps de traitement augmente avec le nombre de modalités des variables. Et si le partitionnement est établi sur des variables quantitatives, alors il choisit le seuil de découpage de la variable et risque de construire des classes trop petites.

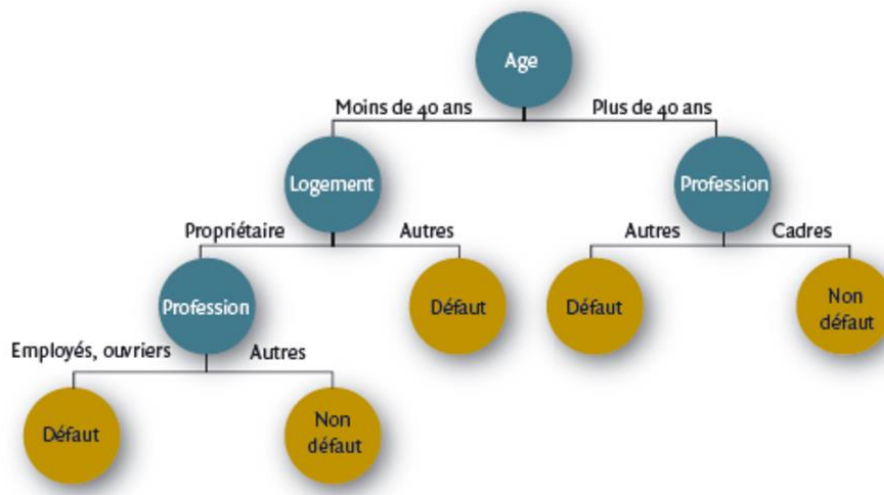


Figure 19: Arbre de décision

c. Le réseau de neurones

Le réseau de neurones est une méthode statistique d'apprentissage supervisé inspirée du fonctionnement du cerveau humain. Il est composé de plusieurs neurones. À chaque entrée de neurone, correspondent n variables notées x_i ($i \in [1 ; n]$) à laquelle on affecte un poids (w_{ij}), pour le j ème neurone. Une combinaison linéaire entre les variables d'entrée et les poids est effectuée, ce qu'on appelle somme pondérée. Cette somme pondérée est soumise à une fonction d'activation (linéaire par exemple) déterminée en fonction du seuil. En sortie de la

fonction d'activation, un paramètre de sortie (o_j) est transmis à un autre neurone comme paramètre d'entrée.

Les réseaux de neurones sont composés de trois couches :

- la première couche, dite d'entrée, reçoit les données sources ; sa taille est directement impactée par le nombre de variables en entrée (les variables explicatives) ;
- la seconde, cachée, n'a pas de contact avec « l'extérieur » ;
- la couche de sortie donne le résultat final, ici le défaut ou non.

Cette méthode a de nombreux inconvénients, en plus de sa non-gérance des valeurs manquantes, comme l'initialisation des poids du réseau ou encore le choix du nombre de neurones dans la couche cachée qui demande de tester plusieurs tailles possibles afin d'obtenir de bons résultats. Il existe de surcroît un risque de sur-apprentissage (ou d'apprentissage « par cœur ») lorsque le modèle comporte trop de variables explicatives, car il expliquera les résidus au lieu du comportement global.

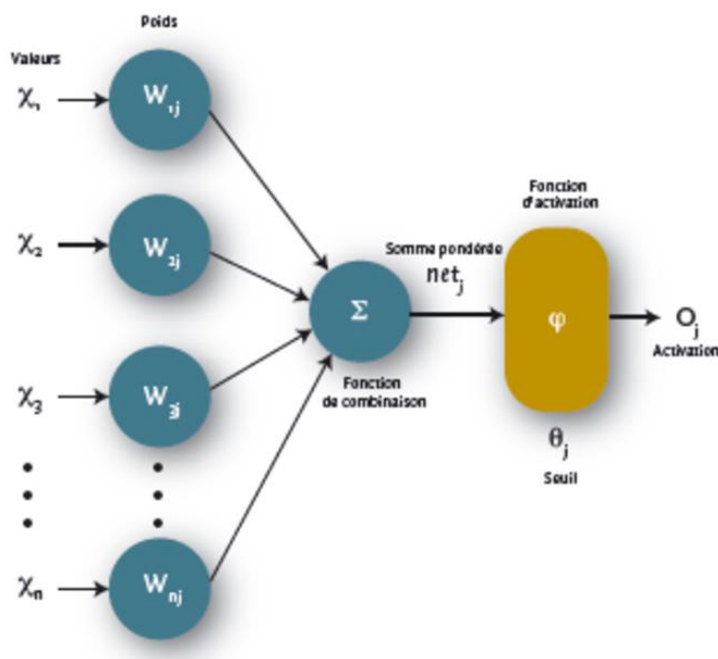


Figure 20: Réseau de neurones

Ainsi les limites de ces modèles vont exiger d'autres méthodes de scoring qui prennent en compte les nouveaux formats de données.

V.4 Le processus de Machine learning pour construire un modèle de crédit scoring Big data

Il n'y a pas de méthodologie unique pour concevoir un outil de crédit scoring Big Data, mais il y a trois étapes majeures à suivre :

- définir le problème à résoudre (la définition de la solvabilité) et spécifier une variable cible représentant le résultat qu'on souhaite prédire
- recueillir des données et la transformer en une forme utilisable
- développer et affiner le modèle grâce à l'exposition aux données d'étude et à la sélection des fonctionnalités

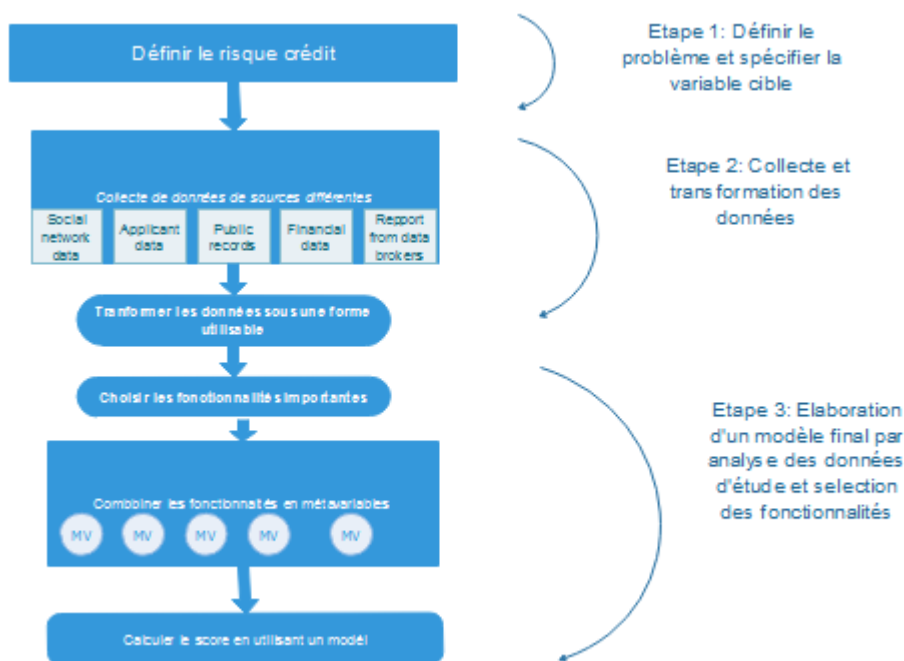


Figure 21: Etapes pour construire un outil de credit scoring

a. Définition du problème à résoudre

Avant d'utiliser des techniques de machine learning supervisées pour résoudre un problème ou faire des prédictions, nous devons d'abord définir le problème et déterminer précisément ce qu'il veut prédire. Cette étape peut sembler évidente, mais dans le cas d'un problème non structuré, comme la prédiction de la solvabilité, où il n'y a pas de réponse correcte, l'articulation d'une définition appropriée et quantifiable est essentielle. En effet, nous devons traduire le problème en une question qui peut s'exprimer dans des termes plus formels que les

ordinateurs peuvent analyser. Une façon d'y parvenir est de sélectionner une variable cible qui peut être définie en fonction d'exemples de résultats passés souvent décrits comme des attributs de classe. Par exemple on peut segmenter les emprunteurs en fonction de leur niveau de revenu et de leurs antécédents de remboursement des cartes de crédit. Les personnes à faible revenu, ou celles qui ne payaient pas régulièrement les soldes des cartes de crédit.

b. La collecte de données et la transformation

Une fois la variable cible et les classes établies, on rassemble ensuite les informations associées aux personnes pour lesquelles les différents résultats sont déjà connus. Cette information constituera les «données d'étude» qui seront utilisées tout au long du processus d'apprentissage automatique pour développer un modèle final. La vision dominante est que plus l'ensemble de données disponible pour l'analyse est grand, plus précis et prédictif est le modèle final. Les données peuvent être collectées à partir de plusieurs sources qui peuvent être classées en quatre grandes catégories :

- **Les données de l'emprunteur** : Contient les informations fournies directement par le demandeur pendant le processus de demande
- **Données propriétaires** : se réfère à l'information obtenue à partir de magasins de données appartenant à des particuliers ou appartenant à des entreprises publiques. Cette deuxième catégorie est peut-être la plus large et peut englober tout, de l'historique d'achat en ligne et hors ligne d'une personne aux données médicales et de santé etc...
- **Données publiques** : contient des informations obtenues à partir de recherches automatisées d'internet et des techniques telles que l'exploration et le raclage web.
- **données du réseau social** : consiste en une activité sociale, y compris une information agrégée des publications des médias sociaux de l'emprunteur et des informations sur les profils sociaux pour tous les membres du réseau social de l'emprunteur.

c. Définition et affinement du modèle

Une fois les données recueillies et transformées, le processus de machine learning peut commencer. Mais toutes les entrées dans les données d'étude ne sont pas pertinentes et de nombreuses entrées sont vraisemblablement rejetées lorsque le système apprend ce qui est

pertinent pour la variable cible et ce qui ne l'est pas. Le processus de machine learning implique généralement une routine d'optimisation qui tente d'identifier les variables d'entrée les plus importantes et les poids appropriés qui devraient être attribués à chacun.

V.5 Enjeux du Big Data pour l'analyse des risques bancaire

La disponibilité de réseaux sociaux et de données CDR (qui capture l'utilisation du téléphone) peut être bénéfique dans différents contextes. Tout d'abord, il peut être utile de marquer des clients qui n'ont pas d'expérience d'emprunt (par exemple, parce que c'est leur premier prêt ou qu'ils ont récemment déménagé dans un nouveau pays) et seraient automatiquement perçus comme risqués selon les modèles traditionnels de notation de crédit qui reposent sur des informations historiques. En utilisant ces sources de données alternatives, une meilleure évaluation du risque de crédit peut être effectuée, ce qui peut ensuite être traduit en un taux d'intérêt plus favorable. Cela inciterait évidemment le client à divulguer son réseau social, son CDR ou d'autres données pertinentes à la banque.

Dans les pays en développement où les banques manquent souvent d'information sur l'historique des crédits, d'autres sources de données devraient être utilisées pour optimiser l'accès au crédit. Compte tenu de l'utilisation répandue des réseaux sociaux et / ou des téléphones mobiles (même dans les pays en développement), les données recueillies pourraient constituer un complément intéressant pour le credit scoring.

■ Chapitre 6 : Mise en œuvre de la plateforme

Dans ce chapitre nous allons expliquer les différentes étapes qui constituent la réalisation, l'architecture et les choix des technologies

VI.1 Présentation des technologies utilisées

VI.1.1 Les distributions Hadoop

Toutes ces technologies de l'écosystème doivent être intégrées manuellement dans Hadoop. Mais pour cela, il faut faire attention aux différentes versions et releases. Malheureusement, tous les releases ne fonctionnent pas parfaitement bien ensemble. Donc, nous devons comparer les notes de livraison et nous débrouiller par nous-même. Déjà que Hadoop lui-même propose tellement de versions, de releases et de features différentes, le déploiement de l'écosystème Big Data, basé sur Hadoop, devient donc un problème. Pour résoudre ces problèmes, il y'a eu les distributions. Une distribution contient différents projets de l'écosystème Hadoop. Ceci assure que toutes les versions utilisées fonctionnent ensemble sans problème. Il y a des releases réguliers avec des versions mises à jour de différents projets. En plus du package, les fournisseurs de distribution offrent des outils graphiques pour le déploiement, l'administration et le monitoring des clusters Hadoop. De cette façon, il est beaucoup plus facile d'installer, de gérer et de surveiller les clusters.

En plus d'Apache Hadoop, il y a plus ou moins trois distributions Hadoop qui en ce moment se distinguent, à savoir **Hortonworks**, **Cloudera** et **MapR**.

a. Hortonworks Data Plateforme

Conçu, développé et construit à partir du libre, la distribution Hortonworks fournit une plateforme de données pour les entreprises prête à l'emploi qui leurs permet d'adopter une architecture de données moderne.

C'est le seul fournisseur qui utilise 100% du projet open source Apache Hadoop sans ses propres modifications. Hortonworks est le premier vendeur à utiliser les fonctionnalités Apache HCatalog pour des services de métadonnées. Par ailleurs, leur initiative Stinger optimise massivement le projet Hive. Hortonworks offre un très bon bac à sable, facile à utiliser pour commencer. Hortonworks a développé et validé des améliorations au niveau du cœur qui rend Apache Hadoop exécutable nativement sur les plateformes Microsoft Windows incluant Windows Server et Windows Azure.

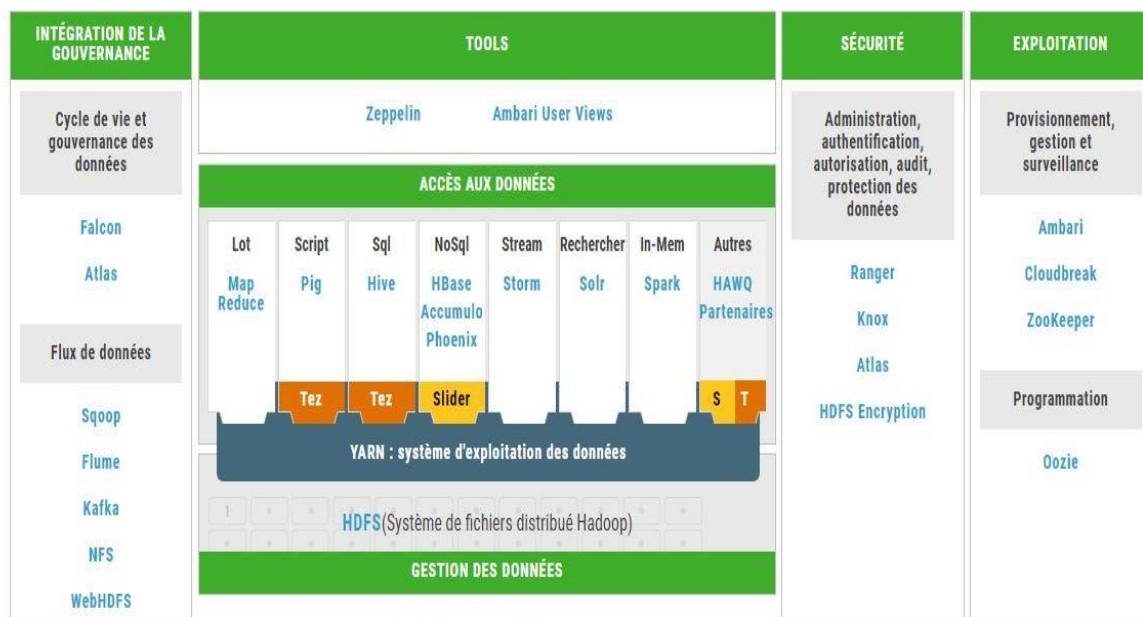


Figure 22: Architecture Hortonworks Data Plateforme

b. Cloudera

C'est la distribution de loin la plus installée avec le plus grand nombre de déploiements référencés. Un outillage puissant pour le déploiement, la gestion et le suivi est disponible. Impala est développée par Cloudera pour offrir des traitements temps réel de Big data.

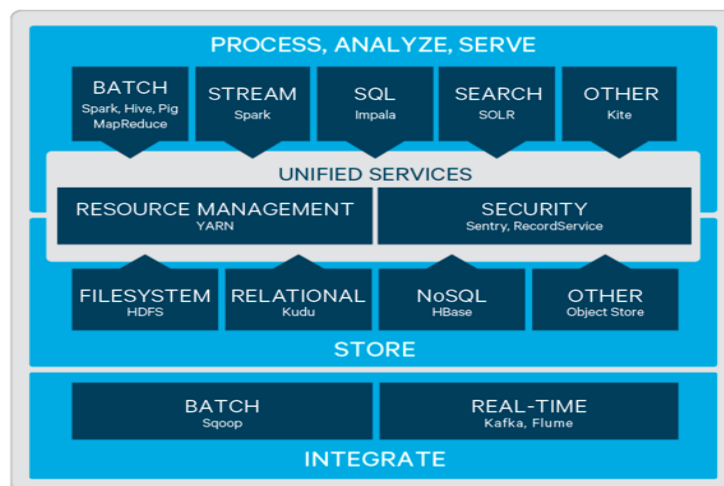


Figure 23: Architecture Cloudera

c. MapR

C'est une distribution qui utilise quelques concepts différents de ses concurrents, en particulier du support pour un système de fichier Unix natif au lieu de HDFS (avec des composants non open source) pour une meilleure performance et une facilité d'utilisation. Les commandes natives Unix peuvent être utilisées à la place des commandes Hadoop. De plus, MapR se différencie de ses concurrents avec des fonctionnalités de haute disponibilité comme les snapshots, la réplication ou encore le basculement avec état.

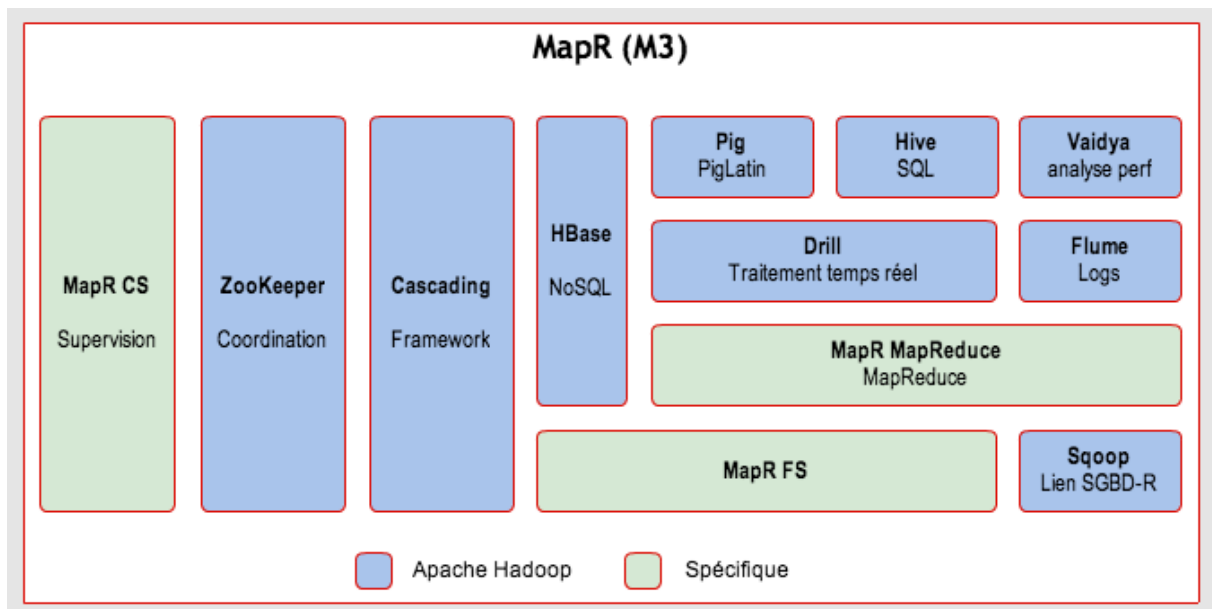


Figure 24: Architecture MapR

d. Tableau comparatif des trois principales distributions

	Hortonworks	Cloudera	MapR
Intégration des données	Par Batch	Par batch	Batch et flux
Architecture des métadonnées	Centralisée	Centralisée	Distribuée
Application NoSQL	Application batch	Application batch	Application batch et temps réel
Réplication	Données	Données	Données + métadonnées
Outil de management	Ambari	Cloudera Manager	Système de contrôle MapR
Sécurité : Liste d'accès	Oui	Oui	Oui
Licence	100% Open Source	Open Source / commerciale	Open Source / Commerciale

Pour notre projet, nous avons choisi d'utiliser la distribution Hortonworks Data Platform vu que c'est la seule distribution qui est à 100% open source et elle offre la majorité des fonctionnalités nécessaires à la mise en place d'une plateforme Big Data.

VI.1.2 Hortonworks Data Plateforme

HDP est la seule véritable distribution d'Apache Hadoop Open Source sécurisée et optimisée pour l'entreprise fondée sur une architecture centralisée (YARN). HDP répond à la totalité des besoins pour les données au repos, optimise les applications client en temps réel et fournit des analyses efficaces pour accélérer la prise de décision et l'innovation. La distribution est constituée par les différentes technologies suivantes :

Plateforme d'analyse distribuée sur l'écosystème Hadoop : Analyse des risques de crédit bancaire

❖ Gouvernance et intégration des données :

- **Falcon :** Falcon est un framework de gestion des données qui simplifie la gestion du cycle de vie des données et traite les pipelines sur Hadoop. Il permet aux utilisateurs d'orchestrer les déplacements de données, le traitement des pipelines, la reprise après sinistre et les flux de rétention des données
- **Flume :** Flume permet d'agréger et de déplacer efficacement de gros volumes de données de journal à partir de sources multiples vers Hadoop.
- **Sqoop :** Apache Sqoop transfère efficacement des données groupées entre Apache Hadoop et des banques de données structurées telles que des bases de données relationnelles. Sqoop peut également être utilisé pour extraire des données de Hadoop et l'exporter vers des banques de données structurées externes. Sqoop fonctionne avec des bases de données relationnelles telles que Teradata, Netezza, Oracle, MySQL, Postgres et HSQLDB.
- **Kafka :** Apache Kafka est un système de messagerie de publication et d'abonnement rapide, évolutif, durable et tolérant aux pannes. Kafka est souvent utilisé à la place de courtiers de messages traditionnels comme JMS et AMQP en raison de son débit, de sa fiabilité et de sa réplication plus élevés.

❖ Accès aux données :

- **Pig :** Pig est une plate-forme de script pour le traitement et l'analyse de grands ensembles de données. Pig interagissent avec les données stockées dans le cluster Hadoop. Apache Pig permet d'écrire des transformations MapReduce complexes en utilisant un langage de script simple appelé Pig Latin.
- **Hive :** Construit sur le framework MapReduce, Hive est un entrepôt de données qui permet de synthétiser les données et d'effectuer des requêtes ad-hoc de façon simple, via une interface similaire au SQL pour les gros volumes de données stockés dans HDFS.
- **HBase :** Apache HBase est une base de données NoSQL ouverte fournissant un accès en lecture / écriture en temps réel à ces grands ensembles de données. HBase évolue linéairement pour gérer d'énormes ensembles de données avec des milliards de lignes et des millions de colonnes, et combine facilement des sources de données qui utilisent une grande variété de structures et de schémas

différents. HBase est intégré nativement avec Hadoop et fonctionne parfaitement avec d'autres moteurs d'accès aux données via YARN.

- **Storm** : Un système de traitement des données en temps réel
Apache Storm ajoute des fonctionnalités fiables de traitement de données en temps réel à Hadoop. Storm avec YARN est puissant pour les scénarios nécessitant des analyses en temps réel, la machine learning et le suivi continu des opérations..
- **Solr** : Apache Solr est la plate-forme open source pour les recherches de données stockées dans HDFS dans Hadoop. Solr gère les fonctionnalités de recherche et de navigation sur bon nombre des plus grands sites Internet du monde, permettant une recherche puissante de texte intégral et une indexation quasi temps réel
- **Spark** : Apache Spark est un moteur de traitement de données rapide et en mémoire avec des API de développement élégantes et expressives pour permettre le streaming, la machine learning ou les charges de travail SQL qui nécessitent un accès itératif rapide aux jeux de données

❖ **Gestion des données** :

- **HDFS** : Le système de fichiers distribué Hadoop (HDFS), écrit en Java, permet de stocker des données de façon fiable et évolutive, sur des grappes importantes de serveurs standards
- **YARN** : C'est un framework de nouvelle génération qui élargit les fonctionnalités de MapReduce dans le traitement des données Hadoop grâce à la prise en charge de flux de travail associés à des modèles de programmation autres que MapReduce.

❖ **Sécurité** :

- **Knox** : La passerelle Knox fournit un point d'authentification et d'accès unique pour les services Hadoop dans un cluster. Le projet vise à simplifier la sécurité d'Hadoop pour les utilisateurs qui accèdent aux données du cluster et exécutent des tâches, ainsi que pour les opérateurs qui contrôlent l'accès au cluster.
- **Ranger** : Apache Ranger offre une approche globale de sécurité pour un

cluster Hadoop. Il fournit une plate-forme centralisée pour définir, administrer et gérer des stratégies de sécurité de manière uniforme dans les composants Hadoop. À l'aide de la console Apache Ranger, les administrateurs de sécurité peuvent facilement gérer les stratégies d'accès aux fichiers, aux dossiers, aux bases de données, aux tables ou à la colonne. Ces politiques peuvent être définies pour les utilisateurs ou les groupes individuels, puis appliquées de manière constante sur la pile HDP.

❖ **Exploitation:**

- **Ambari** : Une plate-forme de gestion entièrement open source pour l'approvisionnement, la gestion, la surveillance et la sécurisation des clusters Apache Hadoop. Apache Ambari retire les conjectures de l'exploitation de Hadoop.
Apache Ambari, dans le cadre de la plate-forme de données Hortonworks, permet de planifier, installer et configurer de manière sécurisée HDP, ce qui facilite la maintenance et la gestion continue des grappes, peu importe la taille du cluster.
- **Oozie** : Apache Oozie est une application Web Java utilisée pour planifier les travaux d'Apache Hadoop. Oozie combine plusieurs travaux séquentiellement dans une unité de travail logique. Il est intégré à la pile Hadoop, avec YARN en tant que centre d'architecture et prend en charge les travaux Hadoop pour Apache MapReduce, Apache Pig, Apache Hive et Apache Sqoop.
- **Zookeeper** : Apache ZooKeeper fournit des services opérationnels pour un cluster Hadoop. ZooKeeper fournit un service de configuration distribuée, un service de synchronisation et un registre de dénomination pour les systèmes distribués. Les applications distribuées utilisent Zookeeper pour stocker et diffuser des mises à jour sur les informations de configuration importantes

VI.1.3 Apache Kafka

Apache Kafka est un système de messagerie rapide, évolutive, durable et tolérant aux pannes publish-subscribe. Kafka est souvent utilisé à la place de courtiers de messages traditionnels comme JMS et AMQP en raison de son plus grand débit, la fiabilité et la réplication.

Apache Kafka prend en charge un large éventail de cas d'utilisation en tant que système de messagerie à des fins générales pour les scénarios où un débit élevé, une livraison fiable, et l'évolutivité horizontale sont importantes. Ces cas d'utilisation incluent :

- ❖ **Traitement des flux de données**
- ❖ **Tracking de site web**
- ❖ **La collecte et le suivi**
- ❖ **Agrégation des journaux**

Certaines caractéristiques font de Kafka une option attrayante pour les cas d'utilisations citées, telle que :

- ❖ **Scalabilité** : Système distribué qui évolue facilement sans temps d'arrêt
- ❖ **Durabilité** : Persiste les messages sur le disque et fournit la réplication intra-cluster
- ❖ **Fiabilité** : Réplique des données, prend en charge plusieurs abonnés et équilibre automatiquement les consommateurs en cas d'échec
- ❖ **Performance** : Haut débit pour la publication et l'abonnement, avec des structures de disque qui fournissent des performances constantes, même avec de nombreux téraoctets de messages stockés

La conception du système de Kafka peut être considérée comme celui d'un journal distribué commit, où les données entrantes sont écrites séquentiellement sur le disque. Il existe quatre principaux composants impliqués dans le déplacement des données dans et hors de Kafka:

- **Topics**
- **Producers**
- **Consumers**
- **Brokers**

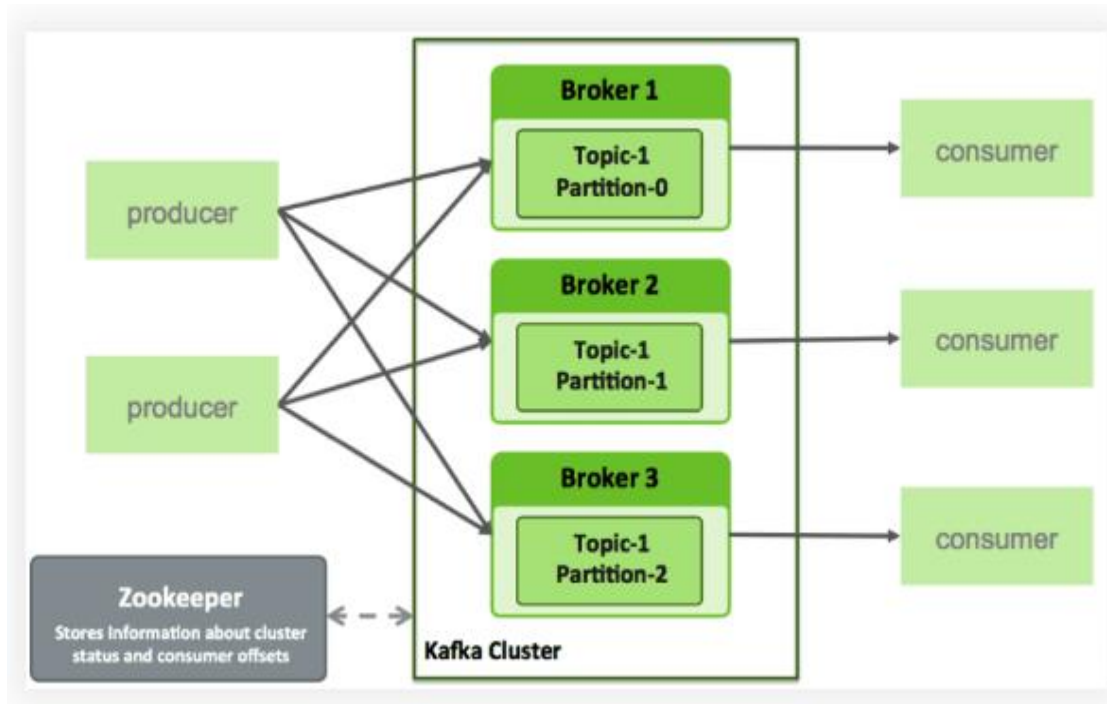


Figure 25: Fonctionnement d'un cluster Kafka

VI.1.4 Apache Spark

Apache Spark est une plate-forme cluster computing conçue pour être rapide et polyvalente.

Sur le plan de la vitesse, Spark étend le modèle MapReduce populaire pour supporter efficacement plus de types de calculs, y compris les requêtes interactives et le traitement des flux. L'une des principales fonctionnalités de Spark est d'offrir une vitesse d'exécution pour les calculs en mémoire, mais le système est également plus efficace que MapReduce pour les applications complexes exécutées sur disque.

Sur le plan généraliste, Spark est conçu pour couvrir une large gamme de charges de travail qui nécessitaient auparavant des systèmes distribués séparés, y compris des applications par lots, des algorithmes itératifs, des requêtes interactives et des flux. En soutenant ces charges de travail dans le même moteur, Spark facilite la combinaison de différents types de traitement, ce qui est souvent nécessaire dans les pipelines d'analyse de données de production.

Spark est unique en ce sens qu'il permet quatre styles d'analyse et de traitement de données différents. Spark peut être utilisée dans :

- ✓ **Batch** : ce mode est utilisé pour manipuler de gros ensembles de données, effectuant généralement de gros travaux map-reduce.

Plateforme d'analyse distribuée sur l'écosystème Hadoop : Analyse des risques de crédit bancaire

- ✓ **Streaming** : ce mode est utilisé pour traiter les informations entrantes à temps réel
- ✓ **Itérative** : ce mode est utilisé pour les algorithmes de machine learning tels que la descente en gradient où les données sont accessibles de façon répétitive afin d'atteindre la convergence.
- ✓ **Interactif** : ce mode est utilisé pour l'exploration de données en mémoire et en raison du temps de réponse très rapide de Spark.

La figure suivante illustre les quatre styles de traitement :

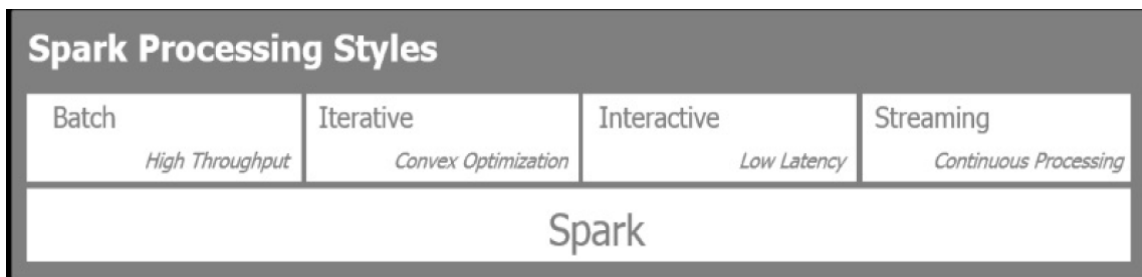


Figure 26: Style de traitement de Spark

Spark fonctionne en trois modes:

- ✓ un mode autonome sur une seule machine
- ✓ deux modes distribués sur un cluster de machines: **Yarn**(gestionnaire de ressources distribuées Hadoop), **Mesos**(gestionnaire de cluster).

Nous présentons chacun des composants de Spark illustré sur la **figure 27**.

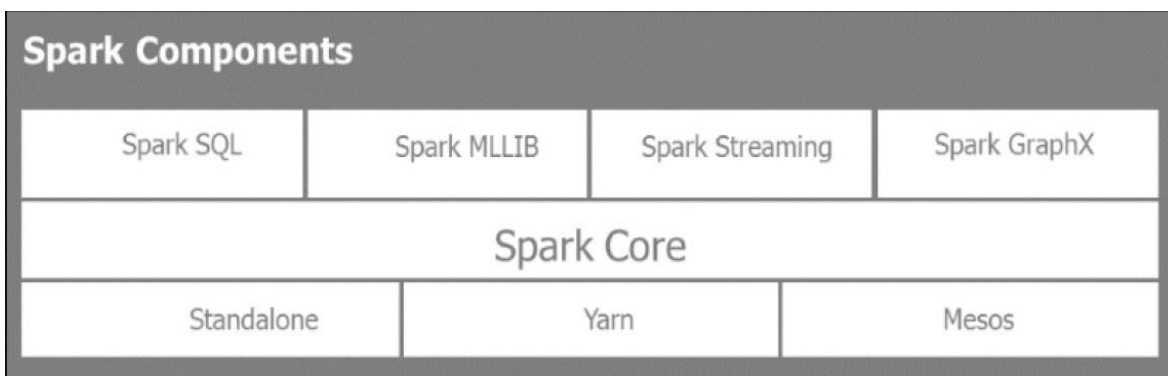


Figure 27: Les composants de Spark

- ❖ **Spark Core** : Spark Core contient la fonctionnalité de base de Spark, y compris les composants pour la planification des tâches, la gestion de la mémoire, la récupération

des défauts, l'interaction avec les systèmes de stockage et plus encore. Spark Core abrite également l'API qui définit des jeux de données distribués résilients (RDD), qui sont l'abstraction de la programmation principale de Spark. Les RDD représentent une collection d'éléments répartis sur de nombreux nœuds de calcul qui peuvent être manipulés en parallèle. Spark Core fournit de nombreuses API pour la construction et la manipulation de ces collections.

- ❖ **Spark SQL :** Spark SQL est le paquet de Spark pour travailler avec des données structurées. Il permet d'interroger les données via SQL ainsi que la variante Apache Hive de SQL - appelée Hive Query Language (HQL) - et elle prend en charge de nombreuses sources de données. Au-delà de la fourniture d'une interface SQL à Spark, Spark SQL permet aux développeurs d'interférer les requêtes SQL RDD dans Python, Java et Scala, toutes dans une seule application, combinant ainsi SQL avec des analyses complexes
- ❖ **Spark Machine Learning :** Spark est livré avec une bibliothèque contenant une fonctionnalité commune de Machine Learning (ML), appelée MLlib. MLlib fournit plusieurs types d'algorithmes de machine learning, y compris la classification, la régression, le regroupement et le filtrage collaboratif, ainsi que les fonctionnalités de support telles que l'évaluation du modèle et l'importation des données. Il fournit également des primitives ML de bas niveau, y compris un algorithme générique d'optimisation de descente en gradient.
- ❖ **GraphX :** GraphX est une bibliothèque pour manipuler des graphiques (par exemple, un graphe d'un ami d'un réseau social) et des calculs graphiques parallèles. Comme Spark Streaming et Spark SQL, GraphX étend l'API Spark RDD, ce qui nous permet de créer un graphique dirigé avec des propriétés arbitraires attachées à chaque sommet et bord.
- ❖ **Spark Streaming :** Spark Streaming est un composant Spark qui permet de traiter des flux de données en direct. Des exemples de flux de données comprennent les fichiers journaux générés par les serveurs Web de production ou les files d'attente contenant des mises à jour d'état publiées par les utilisateurs d'un service Web

i. Fonctionnement de Spark Streaming

Spark Streaming est une extension du noyau API spark qui permet l'évolutivité à haute débit,

la tolérance aux pannes pour le traitement de flux de données en streaming. Les données peuvent être ingérées à partir de nombreuses sources comme KAFKA, Flume ou même des sockets TCP et peuvent être traitées en utilisant des algorithmes complexes exprimés avec des fonctions de haut niveau comme MAP, REDUCE, JOIN, WINDOW. Enfin, les données traitées peuvent être poussées vers des systèmes de fichiers (HDFS par exemple), des bases de données ou des tableaux de bord en direct. On peut également appliquer des machines learning de spark et des graphes processing sur les flux de données.



Figure 28: Fonctionnement de spark streaming (1)

En interne, cela fonctionne comme suit :

- Spark Streaming reçoit des flux de données d'entrée en temps réel
- Spark divise les données en lots
- Les lots de données sont traités par le moteur de Spark
- Enfin, on génère les flux final des résultats en batches



Figure 29: Fonctionnement spark streaming (2)

Spark Streaming fournit un haut niveau d'abstraction appelé Discretized Stream ou DStream, qui représente un flux continu de données. Les DStreams peuvent être créés soit à partir des flux de données d'entrée provenant de sources telles que KAFKA, Flume ou en appliquant des opérations de haut niveau sur les autres DStreams. En interne, un DStream est représenté comme une séquence de RDD.

VI.2 Réalisation de la plateforme

VI.2.1 Déploiement du cluster via HD

Le déploiement du cluster peut se faire soit manuellement en utilisant les packages, soit automatiquement en utilisant Apache Ambari. Nous avons opté pour une installation automatique pour éviter les éventuelles erreurs d'installation et aussi pour gagner en temps.

La figure suivante représente la page d'accueil de l'interface web du gestionnaire de cluster Ambari.

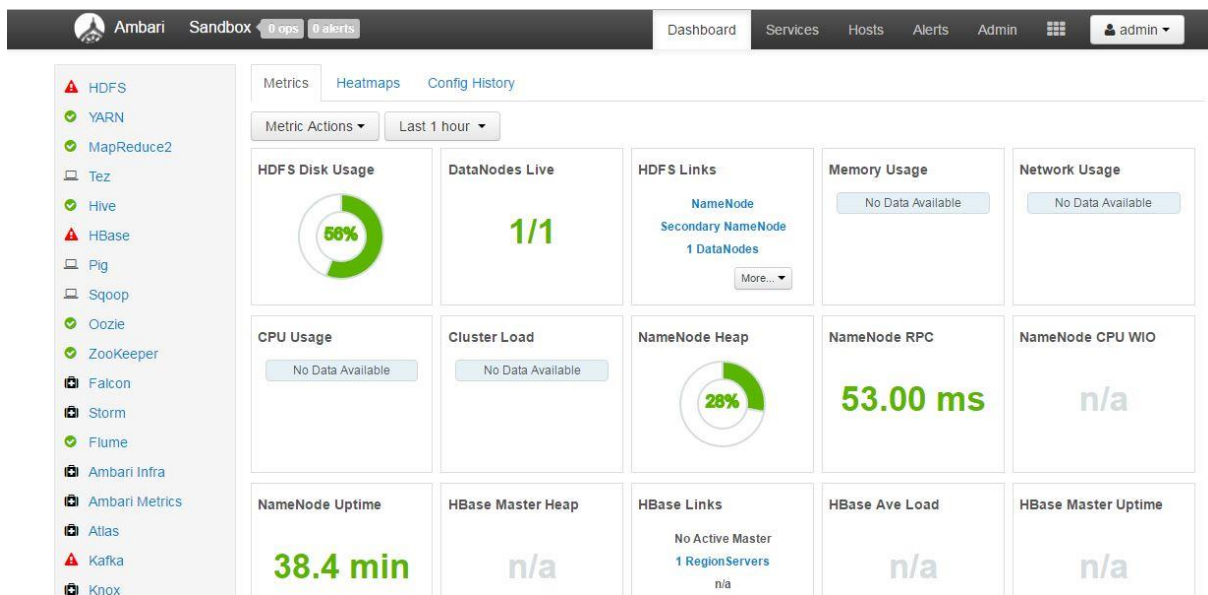


Figure 30: Page d'accueil d'Ambari

VI.2.2 Collecte de données « Twitter »

a. Création de topic Kafka

Une fois le cluster déployé via HDP nous avons besoin de créer un topic kafka qui peut contenir un nombre de partition défini, pour notre cas, nous créons un topic nommé « twitterStream » avec une seule partition. Augmenter le nombre de partitions d'un topic permet d'augmenter le niveau de parallélisme de l'émission et de la consommation des messages du topic.

Pour se faire nous avons réalisé un scripte appelé « setup-demo.sh » permettant d'exécuter les commandes de création de topic, d'installation de l'outil de gestion des dépendances (Maven), de création de répertoire et fichier dans HDFS

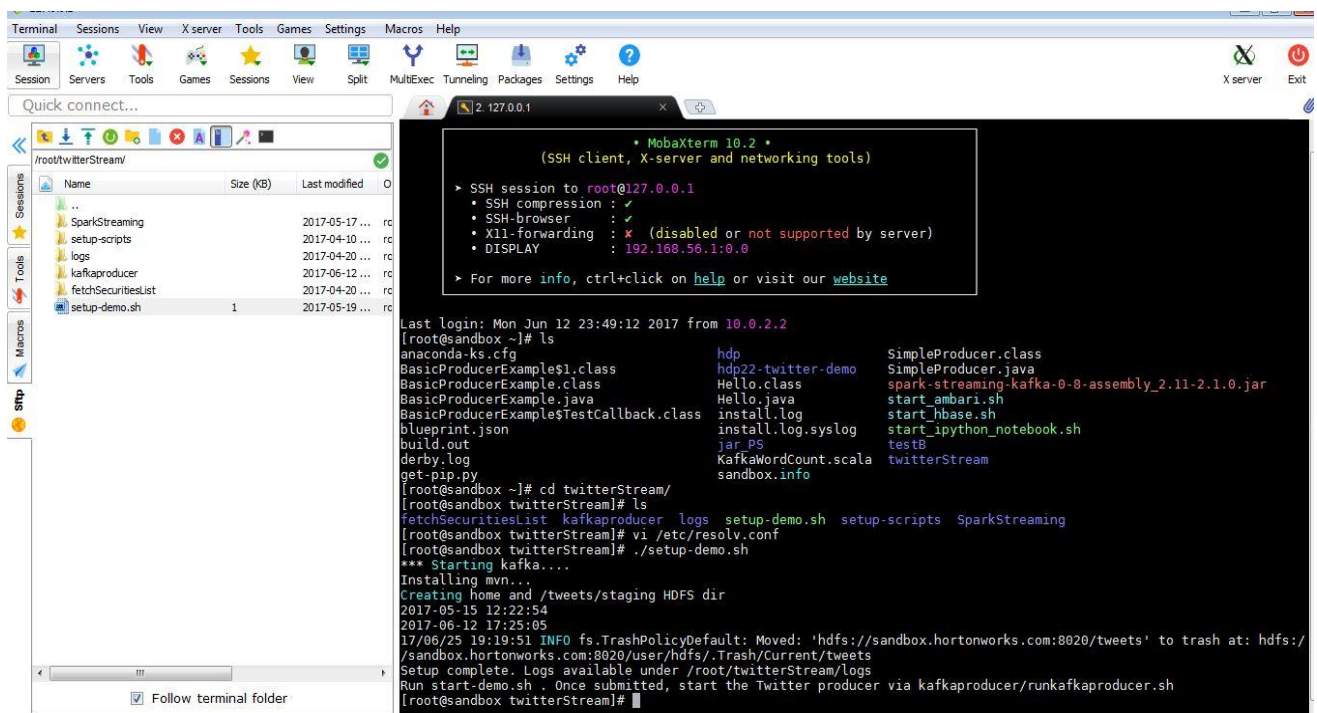


Figure 31: Démarrage du script de création de topic kafka et repertoires HDFS

b. Collecte de « tweets »

Après la création de topic kafka, de fichier hdfs etc. nous devons créer un « producer kafka » pour envoyer des messages à kafka. En pratique, le producer kafka va lire le flux Twitter en utilisant l'API Twitter Stream.

L'API « Twitter streaming » est accessible dans toutes les langues de programmation. "twitter4j" est une bibliothèque open source Java, qui fournit un module basé pour accéder facilement à l'API « Twitter streaming ». Le "twitter4j" fournit un cadre d'écoute sur la base pour accéder aux tweets. Pour accéder à l'API « Twitter streaming », nous avons besoin de nous connecter à un compte développeur Twitter et obtenir les détails d'authentifications suivantes.

- CustomerKey
- CustomerSecret
- AccessToken
- AccessTookenSecret

Une fois que le compte développeur est créé, il faut télécharger le fichier jar « twitter4J » et le placer dans le « path » Java. Ainsi, on peut commencer à collecter les informations twitter en exécutant le scripte « runkafkaproducer.sh » et ainsi les traiter avec « Spark Streaming »

```
Starting ntpd: [ OK ]
Mode is nondebug
Reading stock symbols to send to Twitter from /root/twitterStream/fetchSecuritiesList/thier.csv
Adding to hashtag filters:
Read 1 hashtags to send to Twitter
log4j:WARN No appenders could be found for logger (kafka.utils.VerifiableProperties).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
StatusJSONImpl{createdAt=Mon Jun 26 01:25:29 UTC 2017, id=87914858006760448, text='Keske jmenhui', source='<a href="
http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>', isTruncated=false, inReplyToStatusId=-1,
inReplyToUserId=-1, isFavorited=false, isRetweeted=false, favoriteCount=0, inReplyToScreenName='null', geoLocation=nu
ll, place=PlaceJSONImpl{name='Paris', streetAddress='null', countryCode='FR', id='09f6a7707f18e0b1', country='France'
, placeType='city', url='https://api.twitter.com/1.1/geo/id/09f6a7707f18e0b1.json', fullName='Paris, France', boundin
gBoxType='Polygon', boundingBoxCoordinates=[[Ltwitter4j.GeoLocation;@1b50e6d1], geometryType='null', geometryCoordina
tes=null, containedWithin=null], retweetCount=0, isPossiblySensitive=false, lang='ht', contributorsIDs=[], retweetedS
tatus=null, userMentionEntities=[], urlEntities=[], hashtagEntities=[], mediaEntities=[], symbolEntities=[], currentU
serRetweetId=-1, user=UserJSONImpl{id=966388406, name='Ah oue', screenName='UnDzzzzz', location='Paris, France', desc
ription='Italien Cambodgien Serbe Kabyle. Si t'es noirte me follow pas. [][][], isContributorsEnabled=false, profileIm
ageUrl='http://pbs.twimg.com/profile_images/777978964123979776/TmzLr84w_normal.jpg', profileImageUrlHttps='https://pb
s.twimg.com/profile_images/777978964123979776/TmzLr84w_normal.jpg', isDefaultProfileImage=false, url='https://curious
cat.me/undzzzzz', isProtected=false, followersCount=4209, status=null, profileBackgroundColor='C0DEED', profileTextCo
lor='333333', profileLinkColor='1DA1F2', profileSidebarFillColor='DDEEF6', profileSidebarBorderColor='C0DEED', profil
eUseBackgroundImage=true, isDefaultProfile=true, showAllInlineMedia=false, friendsCount=1679, createdAt=Fri Nov 23 16
:12:39 UTC 2012, favouritesCount=299, utcOffset=7200, timeZone='Amsterdam', profileBackgroundImageUrl='http://abs.twi
mg.com/images/themes/theme1/bg.png', profileBackgroundImageUrlHttps='https://abs.twimg.com/images/themes/theme1/bg.pn
g', profileBackgroundTiled=false, lang='fr', statusesCount=133363, isGeoEnabled=true, isVerified=false, translator=fa
lse, listedCount=21, isFollowRequestSent=false}}StatusJSONImpl{createdAt=Mon Jun 26 01:25:29 UTC 2017, id=87914857979
8964864, text='Graças a Deus ninguém entende nada [][] https://t.co/a2AyNnMm6S', source='<a href="http://twitter.com/d
ownload/android" rel="nofollow">Twitter for Android</a>', isTruncated=false, inReplyToStatusId=-1, inReplyToUserId=-1
, isFavorited=false, isRetweeted=false, favoriteCount=0, inReplyToScreenName='null', geoLocation=null, place=PlaceJSO
NImpl{name='Paris', streetAddress='null', countryCode='FR', id='09f6a7707f18e0b1', country='França', placeType='city'
, url='https://api.twitter.com/1.1/geo/id/09f6a7707f18e0b1.json', fullName='Paris, France', boundingBoxType='Polygon'
, boundingBoxCoordinates=[[Ltwitter4j.GeoLocation;@796eab2], geometryType='null', geometryCoordinates=null, contained
Within=null], retweetCount=0, isPossiblySensitive=false, lang='pt', contributorsIDs=[], retweetedStatus=null, userMen
tionEntities=[], urlEntities=[], hashtagEntities=[], mediaEntities=[MediaEntityJSONImpl{id=879148557210726400, url=ht
tps://t.co/a2AyNnMm6S, mediaURL=http://pbs.twimg.com/media/DDNc9HFXsAAUFJ7.jpg, mediaURLHttps=https://pbs.twimg.com/m
edia/DDNc9HFXsAAUFJ7.jpg, expandedURL=https://twitter.com/Dapaazsilva/status/879148579708964864/photo/1, displayURL='
pic.twitter.com/a2AyNnMm6S', sizes={0=Size{width=150, height=150, resize=101}, 1=Size{width=510, height=680, resize=1
```

Figure 32: Collecte de données twitter en temps réel

c. Streaming des données vers le système de fichier de Hadoop «HDFS »

Les données sont envoyées par les « producer » vers notre topic kafka auquel s'abonne notre application consommatrice (Spark Streaming). Spark Streaming va souscrire au topic « twitterStream » et récupérer sous forme de flux les informations relatives à un tweet (nom utilisateur, pays, date de tweet, texte, géolocalisation etc.).

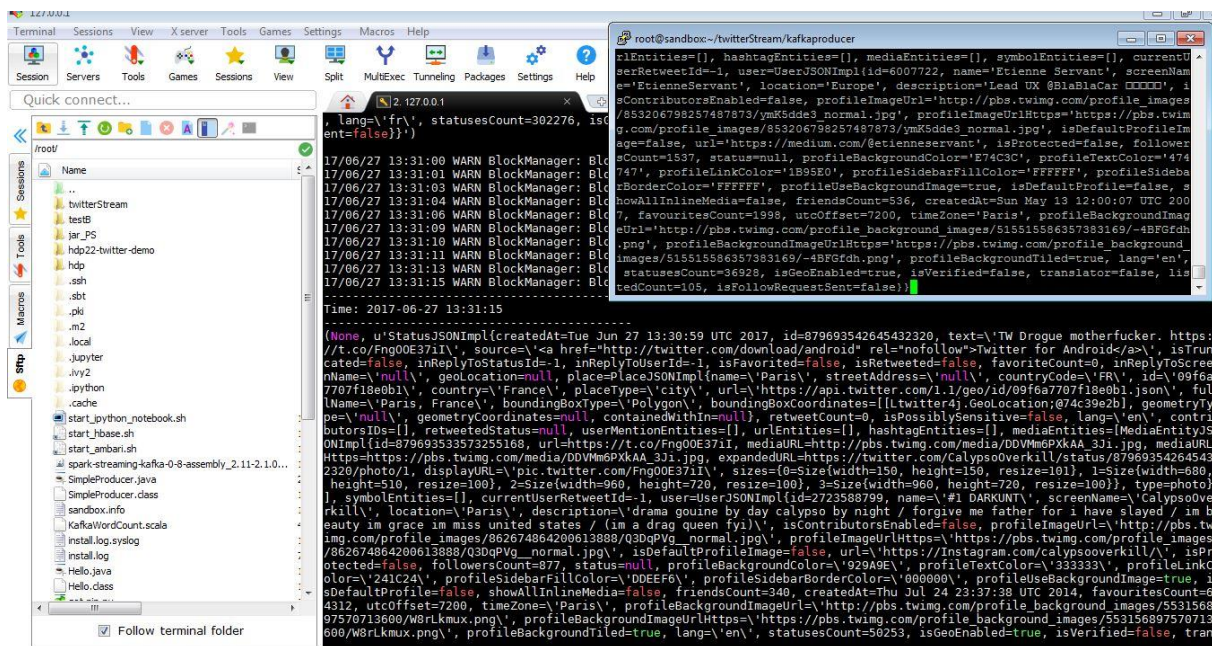


Figure 33: Streaming des données collectées avec Spark vers HDFS

Les données récupérées sont stockées dans le hdfs. Chaque flux de données (par période de 60 secondes) est stockées dans un fichier appelé « tweet ».

/ > creditRisk > TweetStream					
Total: 12 files or folders					
Search in current directory...					
Name >	Size >	Last Modified >	Owner >	Group >	Permission
tweet.txt-1498588125000	--	2017-06-27 18:29	root	hdfs	drwxr-xr-x
tweet.txt-1498588035000	--	2017-06-27 18:27	root	hdfs	drwxr-xr-x
tweet.txt-1498588065000	--	2017-06-27 18:27	root	hdfs	drwxr-xr-x
tweet.txt-1498588080000	--	2017-06-27 18:29	root	hdfs	drwxr-xr-x
tweet.txt-1498588095000	--	2017-06-27 18:29	root	hdfs	drwxr-xr-x
tweet.txt-1498588110000	--	2017-06-27 18:29	root	hdfs	drwxr-xr-x
tweet.txt-1498588050000	--	2017-06-27 18:27	root	hdfs	drwxr-xr-x
tweet.txt-1498588140000	--	2017-06-27 18:29	root	hdfs	drwxr-xr-x
tweet.txt-1498588155000	--	2017-06-27 18:30	root	hdfs	drwxr-xr-x
tweet.txt-1498588170000	--	2017-06-27 18:30	root	hdfs	drwxr-xr-x

Figure 34: Visualisation avec Ambari de répertoires HDFS créés pour chaque flux de tweet

Plateforme d'analyse distribuée sur l'écosystème Hadoop : Analyse des risques de crédit bancaire.

Name	Size	Last Modified	Owner	Group	Permission
._SUCCESS	0.1 kB	2017-06-27 18:27	root	hdfs	-rw-r--r--
part-00000	88.6 kB	2017-06-27 18:27	root	hdfs	-rw-r--r--

Figure 35: Fichier crée pour un flux de tweet

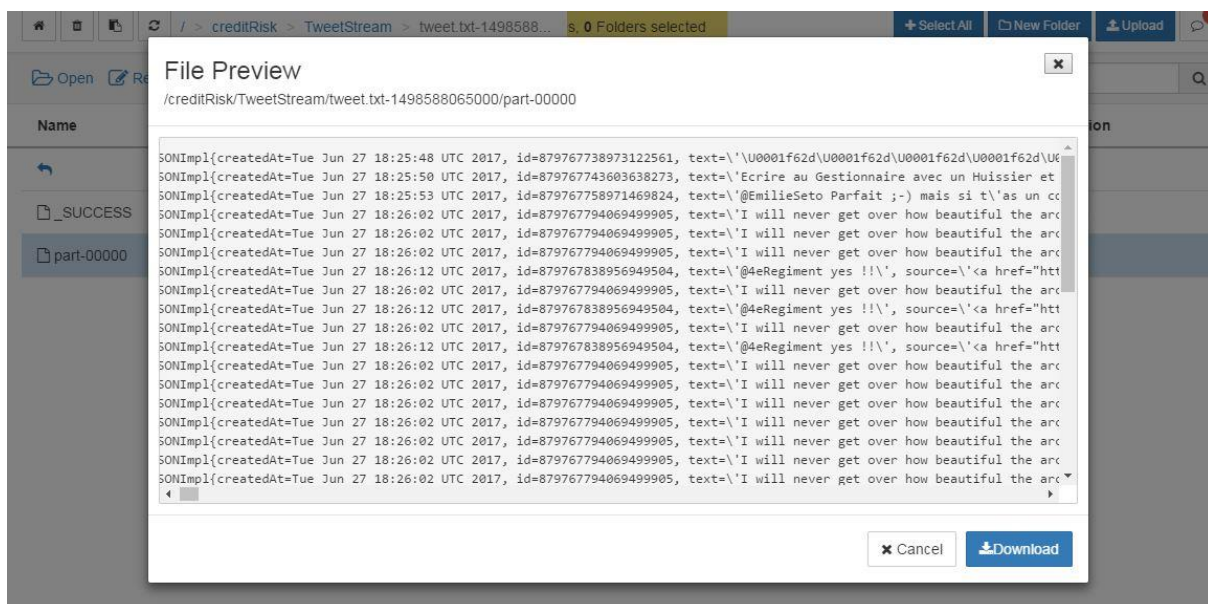


Figure 36: Visualisation avec Ambari du flux de tweet stockées dans HDFS

Dans ce chapitre nous avons présenté les technologies utilisées. Nous avons fait une étude comparative de quelques distributions Hadoop et nous avons choisi la plus adaptée pour notre projet. Nous avons également mis en place la plateforme et réalisé un exemple de collecte et stockage des données de twitter.

CONCLUSION

La croissance exponentielle du volume des données générées par l'humanité et la variété des formats de ces données ont provoqué des problèmes au niveau des entreprises dans le stockage des données mais également dans leur traitement pour créer de la valeur. C'est ainsi que de nouvelles architectures, de nouveaux algorithmes et de nouvelles technologies ont vu le jour pour permettre de stocker, traiter et transporter ces masses considérables de données. Ces technologies innovantes capables de traiter en un temps limité de grands volumes de données afin de valoriser les informations non exploitées de l'entreprise constituent l'écosystème du Big Data.

Dans le cadre de notre stage nous étions amenés à faire une étude et mise en place d'une plateforme d'analyse distribuée sur l'écosystème Hadoop pour ensuite analyser les risques de crédit bancaire.

Dans la réalisation de ce projet, nous avons effectué une étude détaillée du concept Big Data à travers sa définition son architecture et les technologies sur lesquelles se base son écosystème notamment le framework Hadoop. Ensuite, sur la base de cette étude nous avons proposé une architecture technique de la plateforme. Nous avons également fait l'étude du Big Data sur la notation de crédit ou credit scoring pour montrer comment le Big Data pourrait aider à améliorer la manière d'obtenir le score de crédit. Enfin, nous avons mis en place la plateforme et réalisé une collecte et stockage de données en temps réel avec l'API Twitter.

En guise de perspective, nous comptons mettre en place la plateforme dans les datacenters d'Atos (**Bullion**) et ainsi avec des données bancaires qui seront disponibles dans un avenir proche réaliser la partie machine learning avec Spark.

Durant ce stage j'ai été confronté à des difficultés d'ordre matériel ; les ressources matérielles à disposition n'étaient pas adaptées pour un environnement de test. Il était plus judicieux d'utiliser des machines physiques avec des caractéristiques optimales. D'autres difficultés sont celles rencontrées lors de l'intégration d'outils avec la release adaptée et aussi lors des configurations. J'ai été confronté également à des difficultés d'accès à l'information auprès des banques sur les risques de crédit et les méthodes utilisées.

Au début du stage je n'avais aucune connaissance des technologies du Big Data notamment Hadoop et tout l'écosystème.

Plateforme d'analyse distribuée sur l'écosystème Hadoop : Analyse des risques de crédit bancaire.

A l'issu du stage j'ai acquis une compréhension sur les technologies Big Data et j'ai découvert des solutions qui existent sur le marché en réponse aux problématiques du Big Data. Ce stage m'a également permis de comprendre le fonctionnement de l'environnement bancaire notamment l'étude des risques.

Bibliographie

⇒ Bibliographie

- [1] **Learning Apache Kafka Second Edition**, Nishant Garg
- [2] **Hadoop in practice**, Alex Holmes
- [3] **Spark for Python Developers**, Amit Nandi
- [4] **Apache Kafka Cookbook**, Saurabh Minni
- [5] **Big Data Analytics with Spark A Practitioner's Guide to Using Spark for Large Scale Data Analysis**, Mohammed Guller
- [6] **Credit Scoring with Social Network Data**, Yanhao Wei, Pinar Yildirim, Christophe Van den Bulte, Chrysanthos Dellarocas
- [7] **CREDIT SCORING IN THE ERA OF BIG DATA**, Mikella Hurley
Georgetown University Law Center, Julius Adebayo S.M. Computer Science & Technology and Policy, MIT 2016

⇒ Wébographie

- [7] Spark
 - URL : <http://spark.apache.org/>
 - Date de consultation: Mars 2015
- [8] Hortonworks Data Plateforme
 - URL: hortonworks.com
 - Date de consultation: Mars 2015
- [9] Big Data et Credit scoring
 - URL : https://www.lesechos.fr/08/03/2016/LesEchos/22145-040-ECH_comment-le-big-data-va-bousculer-le-credit.htm
https://www.sciencesetavenir.fr/high-tech/data/credit-2-0-quand-le-big-data-intervient-dans-l-accord-d-un-pret_103736
 - Date de consultation: Avril 2017

Annexe

```
set -e
|
echo '*** Démarrage kafka....'
startWait KAFKA

sleep 3

KAFKA_HOME=/usr/hdp/current/kafka-broker
TOPICS=`$KAFKA_HOME/bin/kafka-topics.sh --zookeeper sandbox.hortonworks.com:2181 --list | wc -l`
if [ $TOPICS == 0 ]
then
    echo "Pas de topics Kafka...création de topics..."
    $KAFKA_HOME/bin/kafka-topics.sh --create --zookeeper sandbox.hortonworks.com:2181 --replication-factor 1 --partitions 1 --topic twitterStream
fi

if [ ! -d '/root/twitterStream/logs' ]
then
    mkdir /root/twitterStream/logs
fi

find /root/twitterStream -iname '*.sh' | xargs chmod +x
echo "Installation mvn..."
/root/twitterStream/setup-scripts/install_mvn.sh > /root/twitterStream/logs/install_mvn.log

echo "Creation de repertoire dans HDFS"
if ! hadoop fs -stat /user/root
then
    sudo -u hdfs hadoop fs -mkdir /user/root
    sudo -u hdfs hadoop fs -chown root /user/root
fi

if hadoop fs -stat /tweets
then
    sudo -u hdfs hadoop fs -rm -R /creditRisk/TweetStream
fi
sudo -u hdfs hadoop fs -mkdir /creditRisk/TweetStream
sudo -u hdfs hadoop fs -chmod 777 /creditRisk/TweetStream

echo "Setup complete. Logs disponible sur /root/twitterStream/logs"
echo "Démarrer le producer Twitter en executant le scripte kafkaproducer/runkafkaproducer.sh"
```

Figure 37: Script permettant de créer un topic kafka, et des répertoires dans hdfs

```
MODE=$1
KAFKA_HOME=/usr/hdp/current/kafka-broker/
yum install -y ntp
if [ "$MODE" == "" ]
then
    MODE="nondebug"
fi

TOPICS=`$KAFKA_HOME/bin/kafka-topics.sh --zookeeper sandbox.hortonworks.com:2181 --list | wc -l`
if [ $TOPICS == 0 ]
then
    echo "creation de topic kafka..."
    $KAFKA_HOME/bin/kafka-topics.sh --create --zookeeper sandbox.hortonworks.com:2181 --replication-factor 1 --partitions 1 --topic twitterStream
fi

echo "Compilation jar..."
cd /root/twitterStream/kafkaproducer/
rm -f producertest.jar
rm -rf classes
mkdir classes
export CLASSPATH=$KAFKA_HOME/libs/\\*.\\*
javac -d classes *.java
jar -cvf producertest.jar -C classes/ .

echo "Fixing system time..."
service ntpd stop
ntpdate pool.ntp.org
service ntpd start

java producer.TestProducer $MODE
```

Figure 38: Script pour démarrer le producer Twitter