Dossier de cadrage

RCI Bank & Services

**Contributeurs :**

* Ryadh HADJ MOKHNECHE
* Asmha AMIN
* Hiba BEN HAMIDA
* Mathias TOULOUSE

Sommaire

[1 Compréhension du besoin 3](#_Toc942265)

[1.1 Contexte client : 3](#_Toc942266)

[1.2 Problématiquesclient : 3](#_Toc942267)

[1.3 Objectifs métier : 3](#_Toc942268)

[2 Benchmark 4](#_Toc942269)

[2.1 Benchmark des distributions 4](#_Toc942270)

[2.1.1 Comparatif (Aymen) 4](#_Toc942271)

[2.1.2 Détails du comparatif 4](#_Toc942272)

[2.2 Benchmark des Outils de traitement de la donnée 6](#_Toc942273)

[2.3 Benchmark des outils ETL/ELT 7](#_Toc942274)

[2.4 Benchmark des moteurs de recherches 7](#_Toc942275)

[2.5 Benchmark des bases NoSQL 8](#_Toc942276)

[3 Conclusions: 9](#_Toc942277)

[4 Architectures 11](#_Toc942278)

[4.1 Architecture fonctionnelle 11](#_Toc942279)

[4.2 Architecture technique 12](#_Toc942280)

[4.3 Architecture physique 13](#_Toc942281)

[4.3.1 Cluster MapR 13](#_Toc942282)

[4.3.2 Cluster Elasticsearch 16](#_Toc942283)

# Compréhension du besoin

## Contexte client :

* Filiale bancaire d’un groupe automobile français.
* Données structurées provenant des plusieurs systèmes applicatifs.
* Données client et contrat à traiter (volumétrie des données relativement faibles pour un usage big data).
* Plusieurs systèmes contenant les données client et contrat.

NB : Nous avons pour l’instant peu d’informations sur la qualité et la quantité des données à l’entrée (nombre de fichiers, volumétrie des données).

## Problématiques client :

* Multiplicité des formats et des structures des fiches clients.
* Pour un même client de la banque RCI, il est observé une multiplication des fiches clients, du fait de l’établissement de contrats dans les différents services de l’entreprise (ex: location, achat, assurance).
* Absence d’un référentiel client unique.

→ Campagne marketing peu efficace du fait de l’absence d’un système centralisé permettant de regrouper de manière pertinente l’ensemble des données clients. La présence d’une base de données, rassemblant de manière pertinente les données clients, permettrait d’effectuer une meilleure segmentation de la clientèle et donc permettrait de réaliser un meilleur positionnement marketing.

## Objectifs métier :

* Déduplication des fiches clients à partir des mesures de similitude des fiches clients.
* Election des “golden clients”, afin d’établir une base de données référentielle (Master Data Management) qui sera mise à jour quotidiennement, automatiquement du fait l'exécution journalière du script de génération des goldens clients et manuellement du fait de l’intervention d’un ou plusieurs “data stewards”.
* Offrir au service marketing la possibilité de retrouver les données des golden clients et les contrats associés au travers d’une table alimentée quotidiennement. Une plus faible résolution en temps est possible, cependant elle n’apporterait aucun avantage significatif au service marketing et nécessiterait une monopolisation fréquente des ressources en calcul.

# Benchmark

## Benchmark des distributions

### Comparatif (Aymen)

|  |  |  |  |
| --- | --- | --- | --- |
| **Distribution** | **MAPR** | **Cloudera** | **Horthonworks** |
| **Stockage** | 5 | 3 | 3 |
| **Haute disponibilité** | 5 | 3 | 3 |
| **Sécurité** | 4 | 4 | 5 |
| **Gouvernance** | 2 | 4 | 3 |
| **Supervision** | 3 | 4 | 3 |
| **Stabilité** | 5 | 4 | 4 |
| **Communauté** | 3 | 5 | 4 |

### Détails du comparatif

#### Stockage & Haute Disponibilité

##### HDFS & MAPR-FS similarities

* Compatibilités API HDFS
* Système de fichiers distribué
* Réplication des données
* Utilisation du Network File System Protocol (NFS)
* Reprise après sinistre

##### HDFS vs MAPR-FS

|  |  |
| --- | --- |
| **HDFS** | **MAPR-FS** |
| Metadatas centralisées (namenode) | Metadatas distribuées |
| Gestion difficile d’un grand nombre de petits fichiers | Gestion facilitée d’un grand nombre de petits fichiers |
| Modes séparés Lecture et Ecriture | Mode Simultané Lecture et Ecriture |
| Interactions du système HDFS avec le disque via une JVM | Interactions avec le disque directement en code natif (C/C++) |
| Les “name nodes” contiennent les métadonnées et les informations de bloc pour tous les fichiers | Base de données de localisation des conteneurs du cluster (CLDB) |
| Taille des blocs = 120 MB | Le système de fichiers MapR utilise des unités de tailles multiples pour organiser son contenu allant de petits blocs de 8 KB à très grand conteneurs de 30 GB. Les unités de taille intermédiaire appelées chunks sont par défaut à peu près comparables aux blocs HDFS. |

#### Sécurité

##### Chiffrement des flux de données : SSL

SSL (Secure Socket Layer) est un protocole cryptographique qui assure une communication sécurisée sur un réseau non sécurisé. SSL est un protocole obsolète; sa version moderne s'appelle TLS (Transport Layer Security).

##### Authentification

Kerberos est un protocole d'authentification de réseau informatique qui fonctionne sur la base de tickets pour permettre aux nœuds communiquant sur un réseau de prouver leur identité aux autres de manière sécurisée.

##### Habilitation (gestion des droits)

Sentry offre aux utilisateurs authentifiés la possibilité d'appliquer d'une manière très fine un contrôle d'accès aux données/métadonnées et/ou des privilèges d'accès aux données/métadonnées basées sur les rôles. (Cloudera, Map-R)

Ranger offre une solution pour l'administration de la sécurité. Ranger peut gérer grâce à des plug-ins la sécurité des différents composants de l’architecture. Ranger permet de gérer finement les droits, permettant aux utilisateurs d'effectuer une action ou une opération spécifique sur un composant Hadoop. (Horton)

Access Control List (ACL) est liste de contrôle d'accès associé à une liste d'utilisateurs ou de groupes. Chaque utilisateur ou groupe de la liste est associé à un ensemble défini de permissions qui limitent les actions que l'utilisateur ou le groupe peut effectuer sur un objet sécurisé.

##### Chiffrement, Authentification et Habilitation (MapR Built-in Security solution)

La plate-forme de données convergentes MapR fournit des services d'authentification, d'autorisation et de cryptage pour protéger les données. MapR tire parti des modules d'authentification pour prendre en charge les principaux protocoles d'authentification prêts à l'emploi. Pour l'autorisation, MapR fournit des listes de contrôle d'accès (ACL) pour les files d'attente de travaux, les volumes et le cluster dans son ensemble. Parce que MapR supporte les permissions POSIX sur les fichiers et répertoires, le système de fichiers MapR vérifie les permissions sur chaque accès aux fichiers. Les clusters MapR intègrent également une sécurité au niveau du fil (WLS) pour crypter la transmission des données pour le trafic au sein du cluster, ainsi que le trafic entre le cluster et les machines clientes.

## Benchmark des Outils de traitement de la donnée

|  |  |  |  |
| --- | --- | --- | --- |
| **Technologie** | **Map-Reduce** | **Spark** | **Flink** |
| **Moteur de traitement de données** | Batch | Batch | Stream |
| **Speed** | Plus faible que Spark | 100x plus rapide que Map-Reduce (mise en mémoire dans la RAM) | Plus que rapide que Spark grâce au moteur streaming |
| **Optimisation** | à optimiser manuellement | à optimiser manuellement | optimisé automatiquement |
| **Planification** | Nécessite un outil externe de planification | Outil interne de planification | Outil interne de planification ou YARN |
| **Gestion du cache** | Ne pas enregistrer des données en cache | Enregistre des données en cache ce qui améliore ses performances | Enregistre des données en cache ce qui améliore ses performances |
| **Ingestion des données** | Batch | Batch et Micro-Batch | Temps réel et Batch |

## Benchmark des outils ETL/ELT

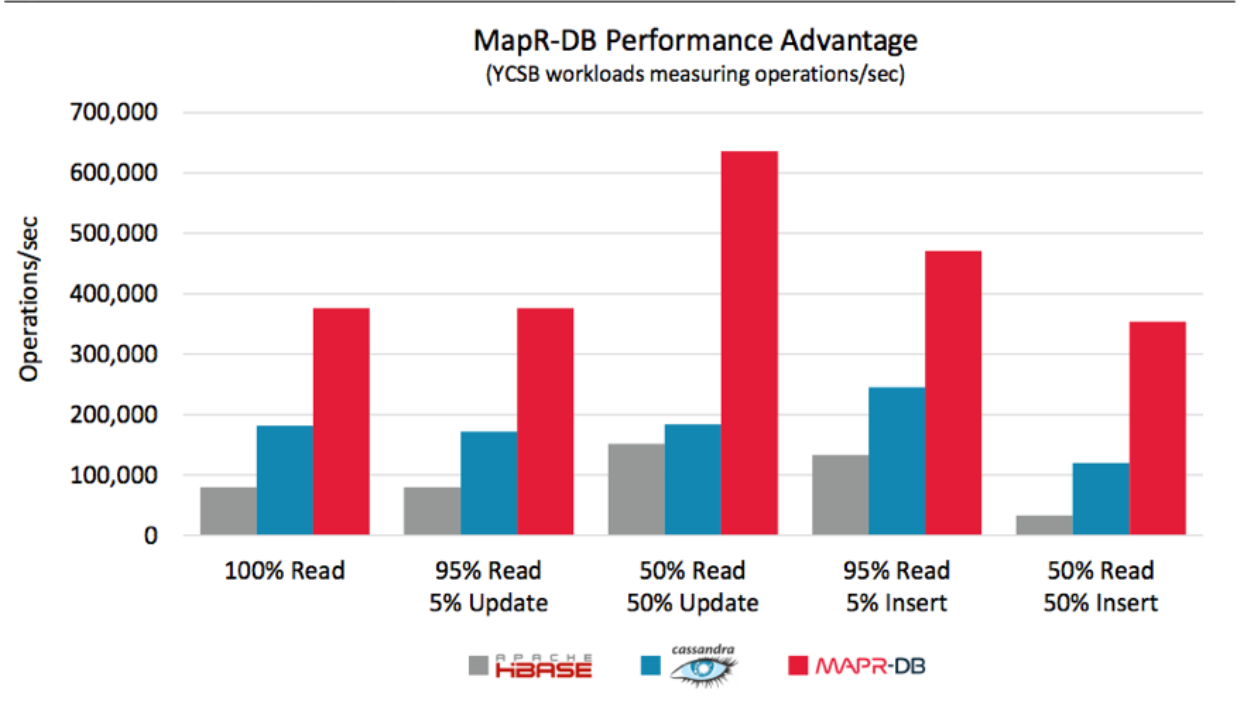
|  |  |  |  |
| --- | --- | --- | --- |
| **Technologie** | **Talend** | **Nifi** | **Flume** |
| **Ingestion des données** | Temps réel + batch | Temps réel | Temps réel |
| **Communauté** | +++ | + | ++ |
| **Distribution des jobs** | Oui | Non | Oui |
| **Filtrage des données** | Oui | Oui | Oui |
| **Concaténation de données** | Oui | Oui | Oui |

## Benchmark des moteurs de recherches

|  |  |  |
| --- | --- | --- |
| **Technologie** | **Solr** | **Elasticsearch** |
| **Performance recherche** | Données statiques | Données dynamiques |
| **Requête DSL** | JSON, XML, URL | JSON |
| **Moteur d’analyse** | Analyse des données | Analyse, calcul statistique et agrégation des données |
| **Parallélisation des traitements** | Non | Oui |
| **Partitionné** | Non | Oui |

## Benchmark des bases NoSQL

|  |  |  |  |
| --- | --- | --- | --- |
| **Solution** | **Famille** | **Avantage** | **Inconvénient** |
| Cassandra | Orientée colonne | * Forte compression de données * Recherche verticales optimisées * Indexation * Plus rapide en lecture | Mise à jour peut être coûteuse |
| HBase | Orientée colonne | * Forte compression de données * Recherche verticales optimisées * Indexation * Plus rapide en écriture * Données plus volumineuse | Mise à jour peut être coûteuse |
| MapR DB | Orientée document et larges colonne | * La base de données n'a aucune couche à traverser lors de l'exécution d'opérations sur des données, car MapR-DB s'exécute à l'intérieur du processus MFS et ce processus lit et écrit directement sur les disques |  |



# Conclusions:

Afin d’atteindre nos objectifs nous avons décidé de proposer à notre client, la Banque RCI, de faire le choix de la distribution **MapR** à laquelle sera associée l’outil **Talend** afin de gérer le data flow, **Spark** nous permettra de réaliser le traitement des données. Les données issues de ce traitement seront stockées dans une base de données NoSQL **MapR-DB (document)** à laquelle sera associée le moteur de recherche **Elasticsearch**. Les motivations de nos choix sont détaillées ci-dessous :

**MapR-FS** comme système stockage pour les raisons suivantes:

* La présence d’un système stockage de données hautement performant offert par MapR-FS.
* L’architecture du cluster MapR qui garantit sa stabilité et une haute disponibilité de la donnée.
* La scalabilité horizontale des clusters MapR qui permettra dans le futur un accroissement de la taille du cluster sans modification architectural.

Afin de réaliser les calculs de similitude qui nous permettront de réaliser l’élection des Golden clients et d’établir la base de données référentielle, nous utiliserons **Spark** qui nous permettra de traiter rapidement par batch l’ensemble des données clients déjà existantes.L’action qui consiste à charger un grand nombre de données et à effectuer un grand nombre de calculs sur ces données dont les résultats finaux dépendent de l’ensemble de ces données chargés est pleinement dans la philosophie de Spark mais nous aurions pu tout aussi bien réaliser tout cela avec Flink. Cependant la possibilité à partir de Talend de générer du code Java pour l’exécuter dans Spark ce qui n’est pas possible avec Flink est l’élément clé de notre prise de décision dans le choix de Spark pour effectuer notre traitement. Talend associé à Spark nous permettra aussi de réaliser rapidement le nettoyage et la restructuration des données brutes.L’ajout de nouvelles données client grâce à un Batch journalier nous permettra de réaliser une mise à jour automatisée du fichier Golden Client, le traitement de ces données issues du batch journalier sera aussi réalisé avec Spark.

Afin de réaliser l’intégration du Data flow, nous utiliserons **Talend**. Nous avons choisi Talend pour plusieurs raisons :

* La possibilité de gérer un flux de données entrant en mode batch.
* La possibilité de générer du code java pour Spark grâce à Talend, nous pourrons ainsi depuis Talend exécuter notre traitement Spark sur le cluster Map-R.
* La diversité de ses connecteurs, connecteurs qui nous permettront de récupérer les données brutes et une fois le traitement terminé d’écrire dans une base MapR-DB.

La base de données NoSQL **MapR-DB (document)** qui associée à MapR-FS offre en lecture et en écriture des performances supérieures aux autres bases de données NoSQL. De plus l’association de MapR-DB à MapR-FS permet d’en augmenter les performances. La base de données MapR-DB permettra d’assurer la gestion complémentaire des mises à jour manuelles des données clients réalisées par les data stewards, à la mise à jour automatique quotidienne réalisée à l’issue de notre traitement.

MapR-DB sera associée au moteur de recherche **Elasticsearch** qui permet grâce à la parallélisation des traitements et au partitionnement des données de traiter rapidement un grand volume de données, ainsi cela permettra au service marketing via une interface (non déterminée par nos soins) d’accéder aux données des Golden clients et d’en réaliser l’analyse.

Les points que nous venons de développer ont été les éléments clés de notre décision. La distribution MapR nous offre des solutions intégrées pour la supervision, la gouvernance des données et la sécurité que nous avons choisi de conserver.

Depuis le passage à la version MapR 6.0, le nouveau système de contrôle MapR (**MCS**) permet via une interface simple, intuitive de gérer les volumes, les tables et flux. Le MCS offre en temps réel une vue sur les métriques du cluster, les alarmes et les logs.

La plateforme **MapR Converged Data Platform** offre plusieurs fonctionnalités destinées à la gouvernance des données et à la mise en conformité avec la GDPR. Cette plateforme fournit des capacités d’audit, un suivi d’accès aux données, et intègre des capacités permettant de contrôler le mouvement des données.

Concernant la sécurité c’est à dire la gestion du chiffrement des données, l’authentification et la gestion des habilitations, il est possible d’utiliser la solution **Built-in offerte par MapR**. Il est cependant possible d’implémenter au sein de notre cluster d’autres solutions comme **ACL,Sentry** et **Kerberos,** c’est cette proposition que nous avons choisi de fait à notre client.

# Architectures

## Architecture fonctionnelle

|  |  |
| --- | --- |
| **Fonction** | **Description** |
| **Transfert des données** | A partir des trois sources de données, transférer les données sans aucune transformation (EL) |
| **Stockage et historisation** | Stockage et conservation des données dans le datalake et partitionnement selon les dates d’import |
| **Nettoyage et restructuration** | Mise en qualité et structuration des données |
| **Déduplication des clients** | Déduplication des données clients afin de retrouver le client golden |
| **Stockage des données dans une base NoSQL** | Exposition des données dans la data view dans une base NoSQL |
| **Mise en place d’un moteur de recherche** | Indexation des données avec un moteur de recherche |
| **Visualisation des données Golden** | Mise en place d’une IHM afin de visualiser les données golden et les liens entre le client golden et ses clients sources |

## Architecture technique

|  |  |
| --- | --- |
| **Fonction** | **Technologie associée** |
| Transfert des données | Talend (comme outil EL) |
| Stockage et historisation | MapR-FS |
| Nettoyage et restructuration | Talend (générateur de code), Spark (traitement) |
| Déduplication des clients | Talend/Spark |
| Stockage des données dans une base NoSQL | MapR-DB (stockage), Talend/Spark (écriture des données) |
| Mise en place d’un moteur de recherche | Elasticsearch (indexation des données) |
| Visualisation des données Golden | API JAVA (lecture des données depuis MapR-DB et Elasticsearch) et IHM web (angular, PHP, etc) |

## Architecture physique

### Cluster MapR

#### Dimensionnement du cluster

Les considérations prises en compte pour l'architecture du matériel du cluster sont les besoins anticipés en stockage de données et en bande passante réseau, y compris les données intermédiaires générées lors de l'exécution du travail.

Le type de charge de travail (workload) est important :

* Nous devons déterminer si l'utilisation prévue du cluster sera gourmande en ressources processeur, E/S ou mémoire.
* Nous devons réfléchir à la manière dont les données seront chargées dans et hors du cluster et à la quantité de données susceptibles d'être transmises sur le réseau.

Exemple d’Architecture standard :

L'architecture ci-après fournit les spécifications d'un nœud de traitement/stockage standard.

Nœud de traitement/stockage standard

* 2 unités de Châssis
* Carte mère simple, double prise (ou sockets ou réceptacle ou slot)
* (2 x 4-cœurs + 32 GB RAM) ou (2 x 6-cœurs + 48 GB RAM)
* 12 x 2 TB de disques 7200 tr/min (RPM)
* 2 ou 4 interfaces réseau [carte réseau intégrée (on-board NIC) + carte réseau supplémentaire (additionnel NIC)]
* OS sur une seule partition sur un lecteur disque (le reste du lecteur est utilisé pour le stockage)

Pour agrandir le cluster, il suffira juste d’ajouter simplement plus de nœuds, en ajoutant les instances de service supplémentaires en fonction des besoins. MapR rééquilibre automatiquement le cluster.

Pour notre projet, nous choisirons la configuration suivante :

Nœud de traitement/stockage standard

* Carte mère simple, double prise (ou sockets ou réceptacle ou slot)
* (1 x 8-cœurs + 64 GB RAM)
* 1 TB de disque SSD (550Mo/s de lecture/écriture séquentielle)
* 2 interfaces réseau [carte réseau intégrée (on-board NIC) + carte réseau supplémentaire (additional NIC)]
* OS sur une seule partition sur le lecteur disque (le reste du lecteur est utilisé pour le stockage)

Pour avoir déterminé le dimensionnement de notre cluster, nous nous sommes basés sur des estimations approximatives :

* 3 Ko par contrat
* 50 Millions de contrats
* Données d’entrée : 150 Go
* Taille disque requise : 333 Go (au début de la période)
* Taille disque requise : 560 Go (au bout d’une année) [progression de 5% mensuel]
* Taille disque chosie : 1 To

Les données sont considérées comme étant non compressées

#### Planification du cluster

La manière dont nous affectons les services aux différents nœuds dépend de la taille de notre cluster et du niveau de licence MapR.

* + Pour un cluster à un seul nœud, aucune décision n'est impliquée. Tous les services que nous utilisons sont exécutés sur le même nœud.
  + Sur les clusters de taille moyenne, ce qui est pour notre cas, les exigences de performance des services CLDB et ZooKeeper proposent que ces derniers soient affectés à des nœuds distincts pour optimiser les performances.
  + Sur les grands clusters, de bonnes performances du cluster nécessitent que ces services s'exécutent sur des nœuds distincts.

Nous pouvons concevoir notre clusterdans l’un des modes suivants :

* + MapReduce Classique : Tous les nœuds du cluster exécutent MapReduce v1.
  + YARN : Tous les nœuds du cluster exécutent YARN (MapReduce v2 et d'autres applications pouvant s'exécuter sur YARN).
  + Mode mixte : Les nœuds du cluster peuvent exécuter YARN ou MapReduce v1.

Pour notre cluster à 5 nœuds, nous utiliserons le mode YARN.

### Cluster Elasticsearch

Equilibrage de la charge avec des répliques : ici 3 partitions avec 2 réplicas.