

# Rapport Projet Données Réparties Partie Hidoop

Axel GRUNIG Thomas GUILLAUD

Département Sciences du Numérique - ASR 2020-2021

# Contents

1	Par	tie Technique
	1.1	Test Ecriture
	1.2	Test Lecture
	1.3	Test Suppression
	1.4	Test Liste
	1.5	Correction : Validation des différentes fonctionnalités
	1.6	Performances
	1.7	Bugs: Presentation et proposition de correction
	1.8	Qualité du code : Points importants
<b>2</b>	Syn	athèse
	2.1	Correction
	2.2	Complétude
	2.3	Pertinence
	2.4	Cohérence
	2.5	Pistes d'améliorations
$\mathbf{L}$	ist	of Figures
	1	Test d'écriture des données
	2	Test de lecture des données
	3	Test de suppression des données
	4	Test d'affichage de la liste des données
	5	test performance suppression
	6	test performance lecture
	7	test performance ecriture
	8	Errair avac fichier métadonnées

## 1 Partie Technique

Tous les test ont été fait avec le fichier filesample.txt fourni.

## 1.1 Test Ecriture

Voici le résultat du test:

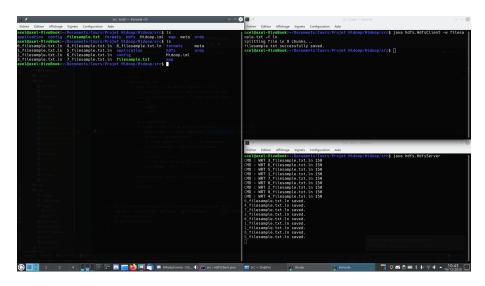


Figure 1: Test d'écriture des données

Pour ce test, nous avons commancé par choisir le fichier (filesample.txt) sans définir une taille de chunk. Après execution de la commande, le fichier original a bien été découpé et envoyé au serveur (ici fait localement) qui a écrit les différents chunk.

On en conclus donc que la fonction d'écriture fonctionne correctement.

## 1.2 Test Lecture

Voici le résultat du test:

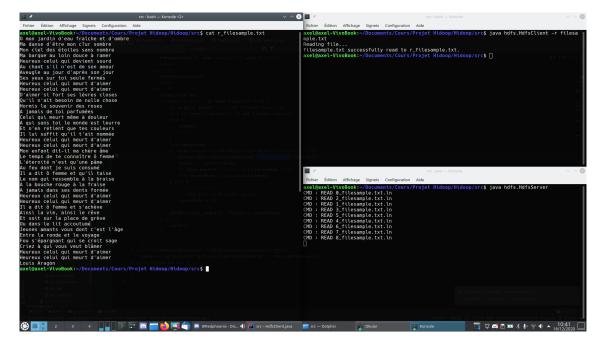


Figure 2: Test de lecture des données

Pour ce test, nous sommes parti du test précédent. en executant la commande de lecture du fichier (filesample.txt), un autre fichier "r\_filesample.txt" a été créé. En comparant ce fichier avec l'original, on remarque qu'ils sont identiques.

On en conclus donc que la fonction de lecture fonctionne correctement.

## 1.3 Test Suppression

Voici le résultat du test:

```
There distant Afficage Square Configuration And American American Afficage Square Configuration And American Ameri
```

Figure 3: Test de suppression des données

En partant du test précédent, nous avons executer la commande de suppression des chunk du fichier (filesample.txt). Tous les chunk ont bien été supprimées par le serveur et les métadonnées ont été modifiées (verification avec la fonction liste).

On en conclus donc que la fonction de suppression fonctionne correctement.

#### 1.4 Test Liste

Voici le résultat du test:

Figure 4: Test d'affichage de la liste des données

Ce test à été effectué plusieurs fois afin de suivre l'évolution des métadonnées. à chaque fois, cette option a fonctionné, indiquant la présence des fichiers copiées dans les serveurs.

Il faut cepandant notter qu'une erreur se produit sur le fichier de métadonnées si le serveur n'est pas lancé (plus d'information dans la partie Bugs).

#### 1.5 Correction : Validation des différentes fonctionnalités

Les différentes fonctionnalités implantées jusqu'à présent fonctionnennent correctement. Nous avons cependant détecté que dans certains cas, un chunk vide (dernier chunk produit) pouvait apparaître lors de l'utilisation de la fonction d'écriture (-w). Il ne s'agit pas vraiment d'une erreur puisque cela n'impacte pas beaucoup l'exécution et ne nuit en aucun à son bon fonctionnement. Il s'agit plus d'une optimisation possible. De plus nous n'avons pas réussi à déterminer la raison de son apparition.

#### 1.6 Performances

Nous avons mesuré les temps d'exécutions des différentes fonctionnalités présentes pour deux fichiers filesample.txt et filesample2.txt respectivement de taille 319,6 ko et 1,3 ko.

Voici les résultats que nous avons obtenus respectivements pour la fonction de suppression, de lecture et d'écriture :

```
axel@axel-VivoBook:~/Documents/Cours/Projet Hidoop/Hidoop/src$ java hdfs.HdfsClient -d filesa mple.txt
Deleting file...
filesample.txt successfully deleted.
Durée exécution: 35996
axel@axel-VivoBook:~/Documents/Cours/Projet Hidoop/Hidoop/src$ java hdfs.HdfsClient -d filesa mple2.txt
Deleting file...
filesample2.txt successfully deleted.
Durée exécution: 96n partant du test précédent, nous avons executer la commande de suppression des chunk du fichier axel@axel-VivoBook:~/Documents/Cours/Projet Hidoop/Hidoop/src$ métadomées ont été modifiées (verification avec la fonction liste).

On en conclus donc que la fonction de suppression fonctionne correctements.
```

Figure 5: test performance suppression

Figure 6: test performance lecture

```
axel@axel-VivoBook:~/Documents/Cours/Projet Hidoop/Hidoop/src$ javac hdfs/*.java
axel@axel-VivoBook:~/Documents/Cours/Projet Hidoop/Hidoop/src$ java hdfs.HdfsClient -w filesa
mple.txt -f ln
Splitting file in 2183 chunks...
filesample.txt successfully saved.
Durée exécution: 32944
axel@axel-VivoBook:~/Documents/Cours/Projet Hidoop/Hidoop/src$ java hdfs.HdfsClient -w filesa
mple2.txt -f ln
Splitting file in 9 chunks...
filesample2.txt successfully saved.
Durée exécution: 246
axel@axel-VivoBook:~/Documents/Cours/Projet Hidoop/Hidoop/src$
```

Figure 7: test performance ecriture

Les durées observées sont en milisecondes.

On observe bien le résultat attendu à savoir que le fichier le plus lourd prend plus de temps à être traité. On observe également que la durée est relativement courte.

## 1.7 Bugs: Presentation et proposition de correction

Lors de l'execution des différents test, nous avons remarqué que lorsque le serveur distant n'est pas mis en place, et que nous effectuons des commandes au travers du client, le fichier de métadonnées est quand même modifié.

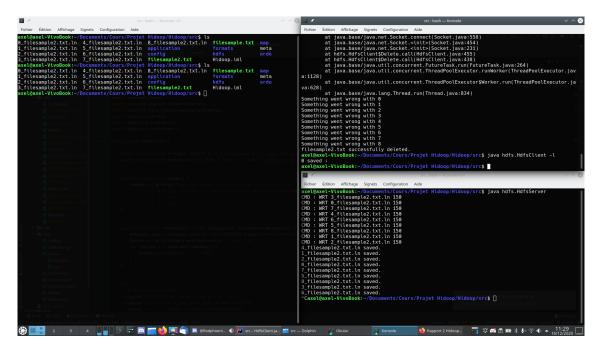


Figure 8: Erreur avec fichier métadonnées

Nous nous somme replacées après une écrture de fichier (ici filesample2.txt). Nous avons ensuite arreté le serveur et fait une suppression avec le client: aucun fichier n'a été supprimé (ce qui est normal), mais les métadonnées ont été mis à jour (la liste des fichiers stockées dans ler serveurs est vide).

La même type d'erreur se produit lors de l'écriture du fichier: rien n'est écrit dans les serveurs, mais les métadonnées sont mis à jour.

Une correction possible est de bloquer l'écriture des métadonnées si il y a un problème avec le serveur en placant cette écriture dans une boucle de gestion d'exception.

## 1.8 Qualité du code : Points importants

Le codes est clair, le nom des variables est bien choisi. De plus, les différents serveurs ont été créées avec l'idée qu'il peut y avoir plusieurs clients qui font des demandes en même temps.

## 2 Synthèse

### 2.1 Correction

Le produit fonctionne correctement, les résultats sont ceux attendus. Aucune erreur majeur n'est détectée lors de l'execution des différentes fonctionnalitées implantées.

## 2.2 Complétude

Toutes les fonctionnalitées attendues ont été mis en place. Comme indiqué sur le premier rapport, d'autres fonctionnalitées tel que la fiabilité du service avec l'échange de message de confirmation, et d'erreur, seront implantées ulterieurement.

#### 2.3 Pertinence

Le travail présenté répond au sujet. Les fonctionnalités présentes (ainsi que celles prévues dans le futur) correspondent à celles demandées. On peut en effet réaliser les actions prévues initialement à l'aide du code fourni. Le rapport apporte des explications claires et détaillées sur les commandes qui doivent être utilisées pour réaliser ces actions. On y trouve également leur fonctionnement, le détail des paramètres ainsi qu'une vue générale des classes présentes afin de comprendre leur fonctionnement global.

#### 2.4 Cohérence

Le code fourni est clair. Il comporte assez de commentaires pour comprendre le fonctionnement de celuici. On observe également qu'il est bien structuré en plusieurs classes dont les objectifs sont bien définis et différents. A l'intérieur de celles-ci les méthodes sont organisées (comme dit plus haut les commentaires sont bien présent et aide à la compréhension). Chacune réalise bien un but différent à chaque fois, ce qui permet d'avoir toutes les fonctionnalités disponibles sans doublons.

#### 2.5 Pistes d'améliorations

1. Remplacer le paramètre taille d'un chunk dans la commande "write" (-w) par un paramètre nombre de chunk.

Raison : Il est plus facile pour un utilisateur de se représenter un nombre qui correspond au découpage du fichier qu'un nombre représentant la taille d'un chunk dont on ne connait pas l'échelle. On ne sait pas vraiment à quoi correspond l'entier que l'on entre en tant que paramètre. On n'a aucune idée de quel nombre donne un certain résultat avant de l'avoir testé.