

Rapport Projet Données Réparties

Partie Hadoop

Axel GRUNIG
Thomas GUILLAUD

Département Sciences du Numérique - ASR
2020-2021

Contents

1	Partie Technique	3
1.1	Test Ecriture	3
1.2	Test Lecture	3
1.3	Test Suppression	4
1.4	Test Liste	4
1.5	Correction : Validation des différentes fonctionnalités	4
1.6	Performances	5
1.7	Bugs : Presentation et proposition de correction	6
1.8	Qualité du code : Points importants	6
2	Synthèse	7
2.1	Correction	7
2.2	Complétude	7
2.3	Pertinence	7
2.4	Cohérence	7
2.5	Pistes d'améliorations	7

List of Figures

1	Test d'écriture des données	3
2	Test de lecture des données	3
3	Test de suppression des données	4
4	Test d'affichage de la liste des données	4
5	test performance suppression	5
6	test performance lecture	5
7	test performance ecriture	5
8	Erreur avec fichier métadonnées	6

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:

```

axel@axel-VivoBook:~/Documents/Cours/Projet Hadoop/Hadoop/src$ ls
0_filesample.txt.in  4_filesample.txt.in  8_filesample.txt.in  formats            meta
1_filesample.txt.in  5_filesample.txt.in  application          hdfs              ordo
2_filesample.txt.in  6_filesample.txt.in  config              Hadoop.lml        r_filesample.txt
3_filesample.txt.in  7_filesample.txt.in  filesample.txt       map
axel@axel-VivoBook:~/Documents/Cours/Projet Hadoop/Hadoop/src$ ls
application  filesample.txt  hdfs      map  ordo
config       formats         Hadoop.lml meta  r_filesample.txt
axel@axel-VivoBook:~/Documents/Cours/Projet Hadoop/Hadoop/src$

axel@axel-VivoBook:~/Documents/Cours/Projet Hadoop/Hadoop/src$ java hdfs.HdfsClient -d filesample.txt
Deleting file...
filesample.txt successfully deleted.
axel@axel-VivoBook:~/Documents/Cours/Projet Hadoop/Hadoop/src$

axel@axel-VivoBook:~/Documents/Cours/Projet Hadoop/Hadoop/src$ java hdfs.HdfsServer
CMD : DEL 4_filesample.txt.in
CMD : DEL 2_filesample.txt.in
CMD : DEL 5_filesample.txt.in
CMD : DEL 8_filesample.txt.in
CMD : DEL 3_filesample.txt.in
CMD : DEL 6_filesample.txt.in
CMD : DEL 1_filesample.txt.in
CMD : DEL 2_filesample.txt.in
CMD : DEL 7_filesample.txt.in
3_filesample.txt.in deleted.
5_filesample.txt.in deleted.
4_filesample.txt.in deleted.
7_filesample.txt.in deleted.
0_filesample.txt.in deleted.
1_filesample.txt.in deleted.
6_filesample.txt.in deleted.
2_filesample.txt.in deleted.
0_filesample.txt.in deleted.
1 saved :
- filesample.txt
axel@axel-VivoBook:~/Documents/Cours/Projet Hadoop/Hadoop/src$

```

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:

```

axel@axel-VivoBook:~/Documents/Cours/Projet Hadoop/Hadoop/src$ java hdfs.HdfsClient -l
1 saved :
- filesample.txt
axel@axel-VivoBook:~/Documents/Cours/Projet Hadoop/Hadoop/src$

```

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 cependant noter 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 fonctionnent correctement. Nous avons cependant détecté que dans certains cas, un chunk vide (dernier chunk produit) pouvait apparaitre 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 respectivement pour la fonction de suppression, de lecture et d'écriture :

```
axel@axel-VivoBook:~/Documents/Cours/Projet Hadoop/Hadoop/src$ java hdfs.HdfsClient -d filesample.txt
Deleting file...
filesample.txt successfully deleted.
Durée exécution: 35996
axel@axel-VivoBook:~/Documents/Cours/Projet Hadoop/Hadoop/src$ java hdfs.HdfsClient -d filesample2.txt
Deleting file...
filesample2.txt successfully deleted.
Durée exécution: 96
axel@axel-VivoBook:~/Documents/Cours/Projet Hadoop/Hadoop/src$
```

Figure 5: test performance suppression

```
axel@axel-VivoBook:~/Documents/Cours/Projet Hadoop/Hadoop/src$ java hdfs.HdfsClient -r filesample.txt
Reading file...
filesample.txt successfully read to r_filesample.txt.
Durée exécution: 2076
axel@axel-VivoBook:~/Documents/Cours/Projet Hadoop/Hadoop/src$ java hdfs.HdfsClient -r filesample2.txt
Reading file...
filesample2.txt successfully read to r_filesample2.txt.
Durée exécution: 200
axel@axel-VivoBook:~/Documents/Cours/Projet Hadoop/Hadoop/src$
```

Figure 6: test performance lecture

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

Figure 7: test performance ecriture

Les durées observées sont en millisecondes.

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'exécution 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 sommes replacées après une écriture de fichier (ici filesample2.txt). Nous avons ensuite arrêté le serveur et fait une suppression avec le client: aucun fichier n'a été supprimé (ce qui est normal), mais les métadonnées ont été mises à jour (la liste des fichiers stockés dans les 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 mises à jour.

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

1.8 Qualité du code : Points importants

Le code est clair, le nom des variables est bien choisi. De plus, les différents serveurs ont été créés 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'exécution des différentes fonctionnalités implantées.

2.2 Complétude

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

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 celui-ci. 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 connaît 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.