Réponses

1-Hdfs:

- Ajout d'un NameNode : ceci nous permet de savoir exactement les serveur connectées et stocker les informations sur le type des fichiers.
- Correction du problème de la commande "READ" qui supposait que le fichier était de type KV. En effet, grâce au NameNode on arrive à obtenir le type du fichier qui est stocké dans un objet java de type "Map".
- Pour éviter l'erreur **Exception in thread "main" java.lang.OutOfMemoryError:Java heap space** , on envoie maintenant les KV une par une et non pas tout le fichier à la fois.
- Changement de la manière avec laquelle on obtient la configuration: précédemment, on devait ajouter notre machine manuellement dans Project.java, maintenant nous n'avons besoin que d'un fichier de configuration en XML.
- Lancement des commandes en parallèle.
- Le programme fonctionne sur des machines différentes.
- Une amélioration non ajoutée : stocker dans le NameNode(attribut files) les serveurs qui contiennent le fichier (si on lance par exemple un nouveau serveur et que le fichier est déjà stocké, lorsqu'on veut le lire on cherche aussi dans le nouveau serveur).
- On doit aussi enlever le serveur du NameNode lorsqu'il se ferme (avec la méthode removeDeamon() du NameNode).

2-Hidoop:

- De même que dans Hdfs, les workers sont lancés avec des fichiers de configuration écrits avec XML.
- Le runMap est devenu maintenant un lancement de thread

- Job prend lui aussi un fichier de configuration pour connaître le NameNode.
- Job peut encore être amélioré: Au lieu de lancer un thread et l'attendre (launchWork.start() puis launchWork.join()) on devrait mettre en thread la boucle qui fait le "runMap" et lancer autant de thread que de Worker.