Partie 2

Hadoop File System (HDFS)

Dr. H. EL BOUHISSI BRAHAMI

M2 IA

HDFS : Hadoop File System

- Système de fichiers distribué associé à Hadoop. C'est là qu'on stocke les données d'entrée, de sortie, etc.
- Les fichiers et dossiers sont organisés en arbre.
- Les fichiers sont stockés sur un grand nombre de machines de manière à rendre invisible la position exacte d'un fichier. L'accès est transparent, quelle que soient les machines qui contiennent les fichiers.
- Les fichiers sont copiés en plusieurs exemplaires pour la fiabilité et permettre des accès simultanés multiples.

HDFS - Caractéristiques

– Distribué, évolutif, tolérant aux pannes, à haut débit

– Accès aux données via MapReduce

– Fichiers divisés en blocs

– 3 répliques pour chaque élément de données par défaut

– Peut créer, supprimer, copier, mais pas mettre à jour

– Conçu pour lire en Streaming, pas pour accès aléatoire

– Data locality : traitement des données sur le support de stockage physique pour réduire la transmission des données

HDFS - notion de Blocks

les fichiers sont physiquement découpés en blocs d'octets de grande taille (par défaut 64 Mo) pour optimiser les temps de transfert et d'accès ;

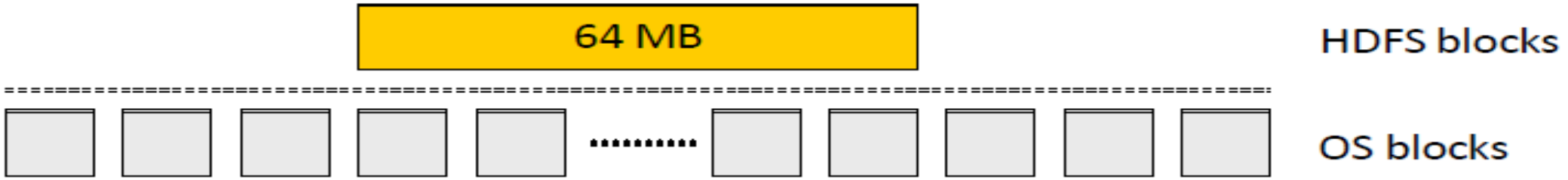
• ces blocs sont ensuite répartis sur plusieurs machines, permettant ainsi de traiter un même fichier en parallèle. Cela permet aussi de ne pas être limité par la capacité de stockage d'une seule machine pour au contraire tirer parti de tout l'espace disponible du cluster de machines ;

• enfin, pour garantir une tolérance aux pannes, les blocs de chaque fichier sont répliqués, de manière intelligente, sur plusieurs machines.

HDFS - notion de Blocks

HDFS est conçu pour supporter des fichiers très volumineux

- Chaque fichier est divisé en blocs
- Hadoop default: 64MB
- Les blocs résident sur différents DataNode physiques
- Un bloc HDFS est pris en charge par plusieurs blocs de système d'exploitation



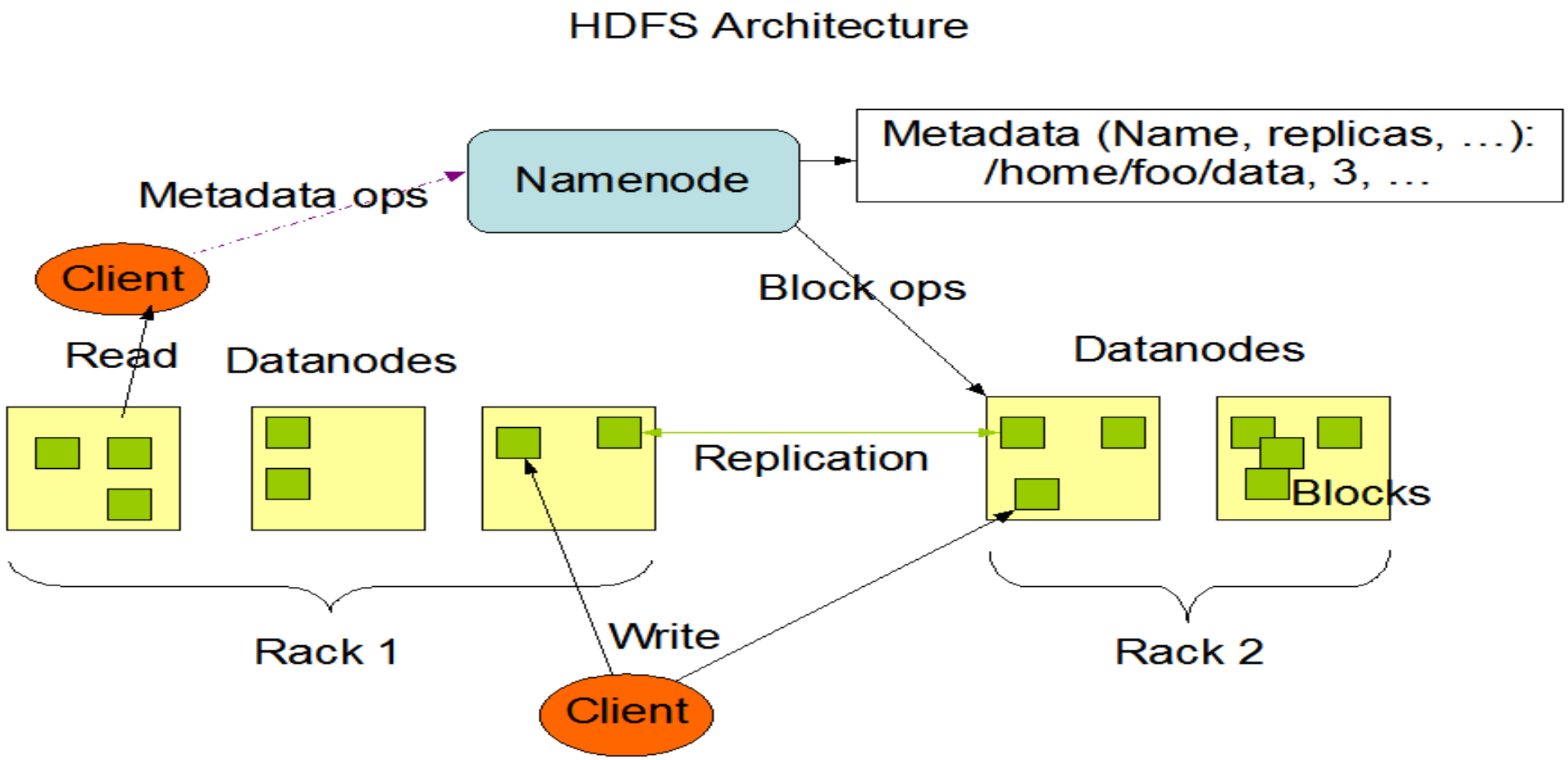
Si un fichier ou un morceau du fichier est plus petit que la taille du bloc, seul l'espace requis est utilisé. Dans: un fichier de 210 Mo est divisé comme

HDFS : Architecture (2)

L'architecture du HDFS est du type Maitre/Esclave , le maitre est appelé **NameNode** qui stocke les métadonnées, l'esclave est **DataNode** qui stocke les données actuelles, HDFS consiste en un seul NameNode et plusieurs DataNodes, en général, l'architecture du HDFS comporte :

- Un NameNode qui contient tous les noms et blocs des fichiers, comme un gros annuaire téléphonique.
- Une autre machine qui est le **secondary NameNode**, une sorte de NameNode de secours, qui enregistre des sauvegardes de l'annuaire à intervalles réguliers.
- Toutes les autres machines sont des DataNode. Elles stockent les blocs du contenu des fichiers.
- D'autres machines qui sont des clients. Ce sont des points d'accès au cluster pour s'y connecter et travailler.

HDFS : Architecture



HDFS : Architecture (3)

Les datanodes contiennent des blocs. Les mêmes blocs sont dupliqués (*replication*) sur différents datanodes, en général 3 fois. Cela assure :

• fiabilité des données en cas de panne d'un datanode,

• accès parallèle par différents processus aux mêmes données.

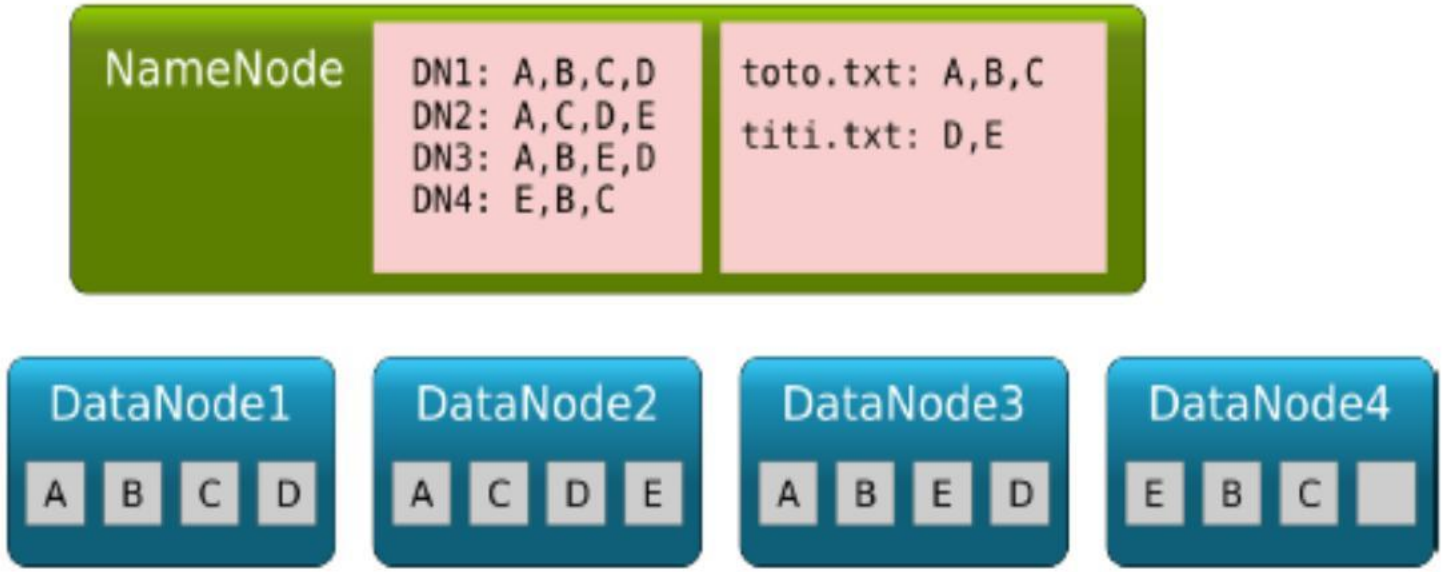
Le namenode sait à la fois :

• sur quels blocs sont contenus les fichiers,

• sur quels datanodes se trouvent les blocs voulus.

On appelle cela les *metadata*.

Inconvénient majeur : panne du namenode = mort de HDFS, c'est pour éviter ça qu'il y a le secondary namenode. Il archive les metadata, par exemple toutes les heures.



Les datanodes contiennent des blocs (A, B, C. . .), le namenode sait où sont les fichiers : quels blocs et sur quels datanodes.

NameNode (1) : Nœud principal

Stocke les méta-données, à savoir le nombre de blocs de données, de répliques et d'autres détails. Ces métadonnées sont disponibles en mémoire dans le maître pour une récupération plus rapide des données. NameNode gère les nœuds esclaves et leur attribue des tâches. Il devrait être déployé sur du matériel fiable, car il s'agit de la pièce maîtresse de HDFS.

Les tâches du NameNode:

- Gérer l'espace de noms du système de fichiers.
- Contrôler l'accès du client aux fichiers.
- Exécuter l'exécution du système de fichiers tel que nommer, fermer, ouvrir des fichiers / répertoires.
- Prendre en compte le facteur de réplication de tous les blocs.

NameNode (2)

Les fichiers présents dans les métadonnées NameNode sont les suivants:

FsImage : «fichier image», contient la totalité de l'espace de noms du système de fichiers et est stocké sous forme de fichier dans le système de fichiers local de NameNode.

EditLogs : Contient toutes les modifications récentes apportées au système de fichiers sur la plus récente FsImage. NameNode reçoit une demande de création / mise à jour / suppression du client. Après cela, cette demande est d'abord enregistrée dans le fichier de modifications.

DataNode : Esclave

DataNode stocke les données réelles dans HDFS. Il effectue des opérations de lecture et d'écriture selon la demande du client. DataNode peut se déployer sur du matériel standard.

Les tâche du DataNode :

- Bloquer la création, la suppression et la réplication de répliques conformément aux instructions de NameNode.
- Gérer le stockage de données du système.
- Envoyer des pulsations au NameNode pour signaler la santé de HDFS. Par défaut, cette fréquence est définie sur 3 secondes.

Secondary NameNode

Lorsque NameNode démarre, il lit d'abord l'état HDFS à partir d'un fichier image, FsImage. Ensuite, il applique les modifications du fichier journal des modifications. NameNode écrit ensuite le nouvel état HDFS dans FsImage.

Ensuite, il commence son fonctionnement normal avec un fichier d'édition vide. Au moment du démarrage, NameNode fusionne FsImage et édite les fichiers afin que le fichier journal d'édition puisse devenir très volumineux au fil du temps. L'un des effets secondaires d'un fichier d'édition plus volumineux est que le prochain redémarrage de NameNode prend plus longtemps.

NameNode secondaire résout ce problème. Secondary NameNode télécharge les fichiers FsImage et EditLogs à partir du NameNode. Et fusionne ensuite EditLogs avec le FsImage (FileSystem Image). Il garde la taille du journal des modifications dans une limite. Il stocke le FsImage modifié dans un stockage persistant. Et nous pouvons l'utiliser en cas d'échec de NameNode.

Secondary NameNode effectue un point de contrôle régulier dans HDFS.

HDFS - Réplication

La gestion du stockage est assurée par les daemons Hadoop. On a pas à se soucier d'où sont stockées les données ;

• Hadoop réplique lui-même les données : les fichiers sont disponibles à tout moment sur plusieurs DataNodes, et si une machine tombe en panne, on a toujours accès aux données grâce à la réplication ;

• Lorsqu'un des nœuds a un problème, les données seront perdues :

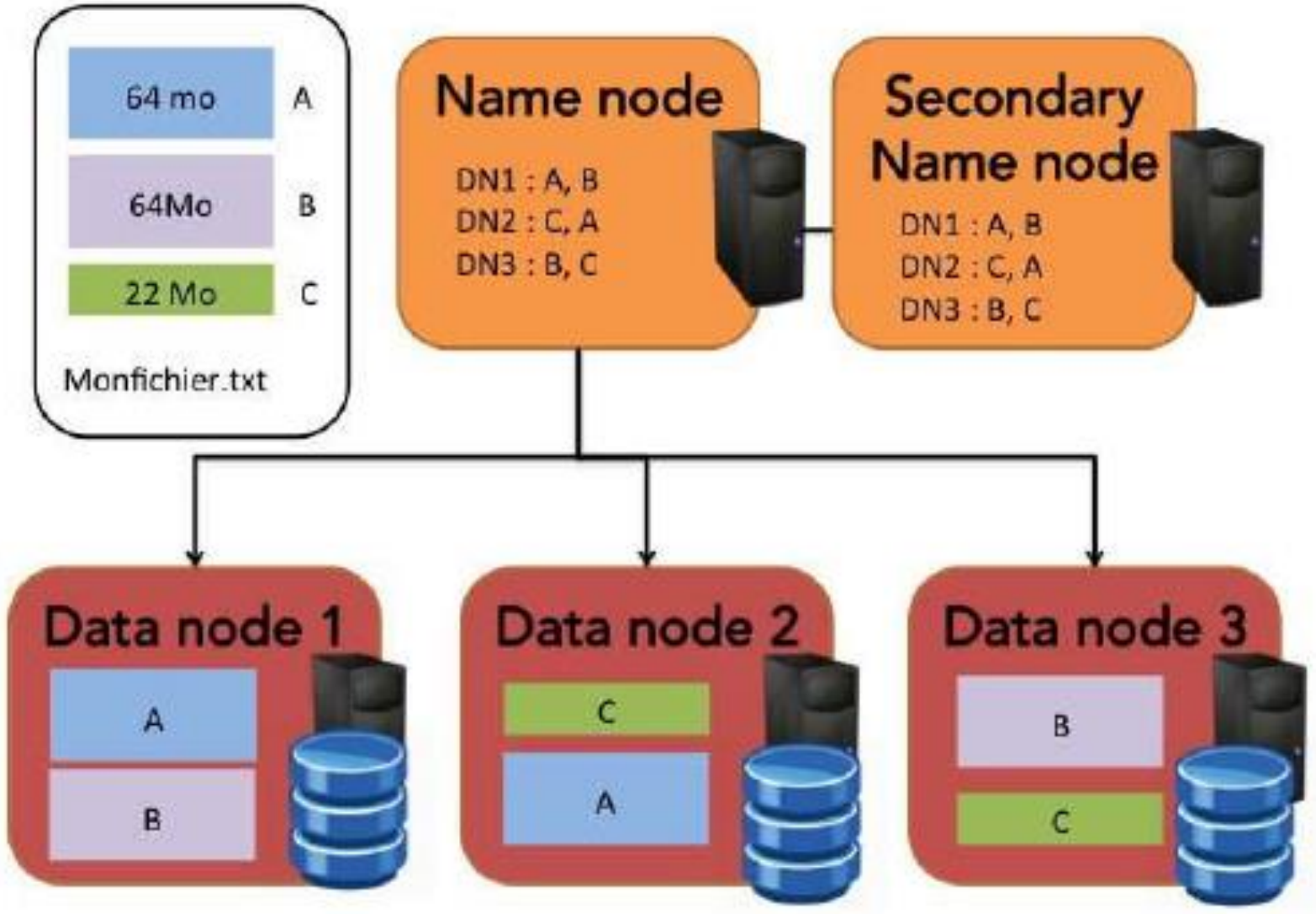
– Hadoop réplique chaque bloc 3 fois ;

– Il choisit 3 nœuds au hasard, et place une copie du bloc dans chacun d'eux ;

– Si le nœud est en panne, le NameNode le détecte, et s'occupe de répliquer encore les blocs qui y étaient hébergés pour avoir toujours 3 copies stockées.

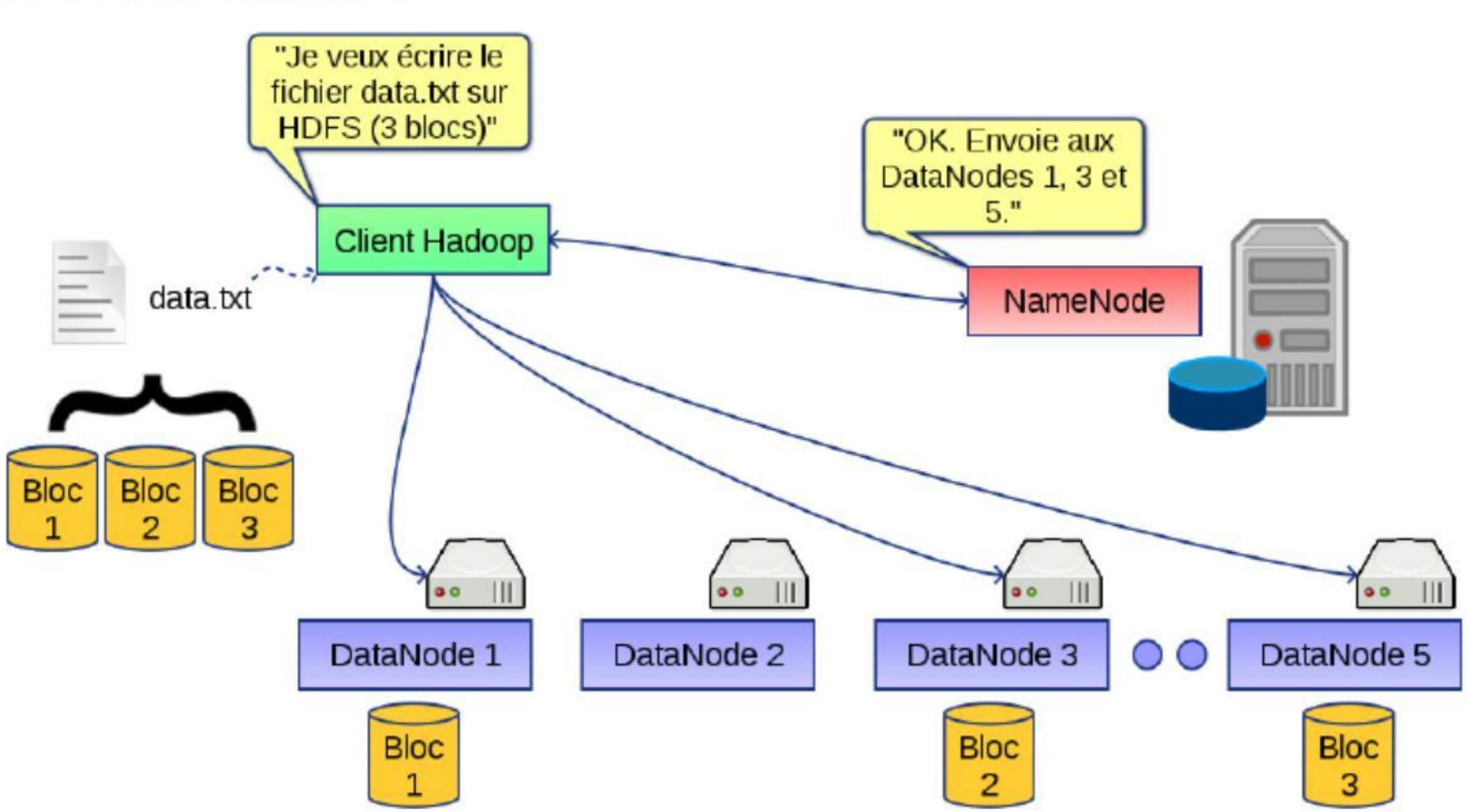
• Si le NameNode a un problème = mort de HDFS? c'est pour éviter ça qu'il y a le secondary namenode. Il archive les metadata, par exemple toutes les heures.

HDFS - Réplication

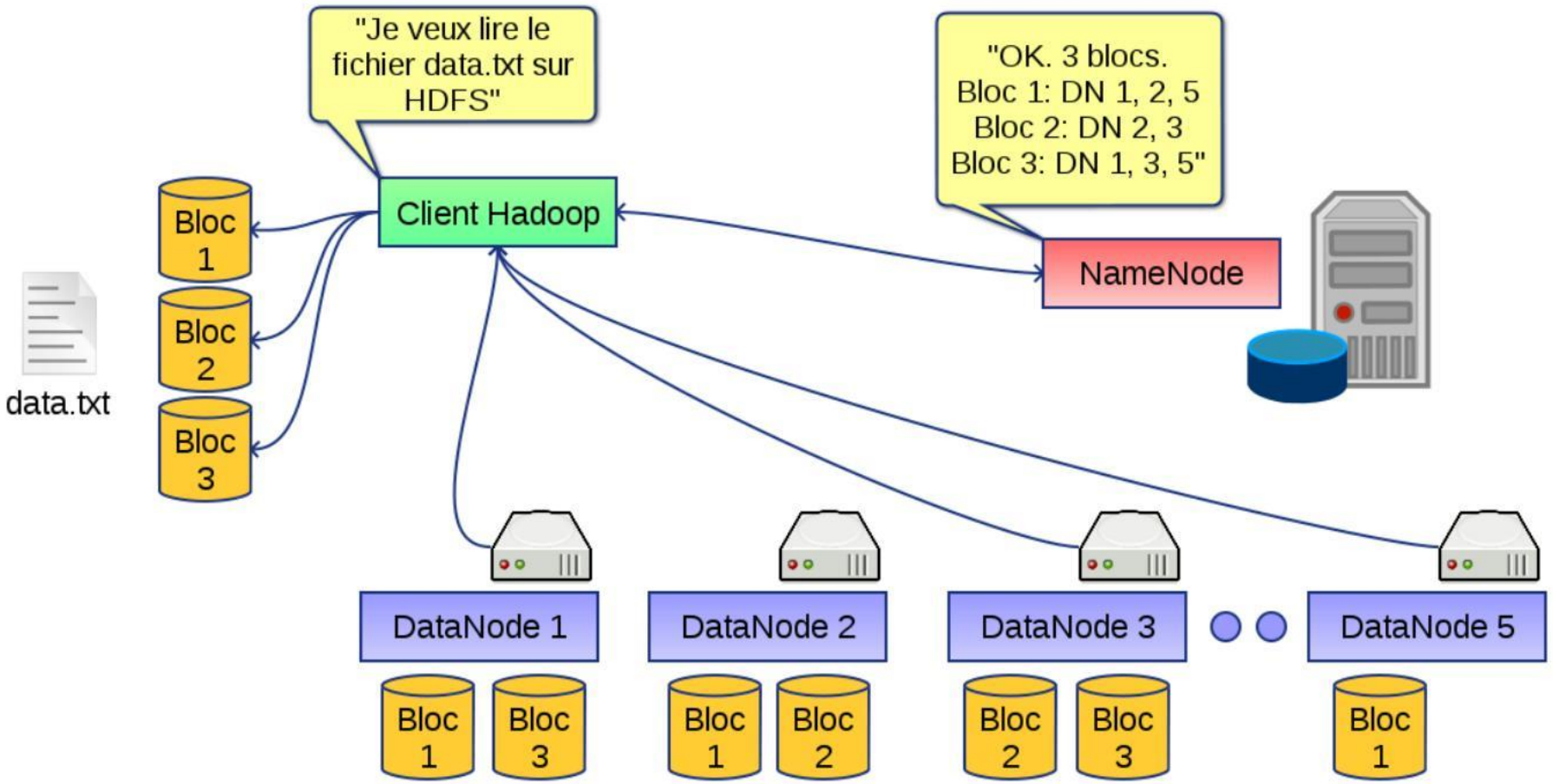


HDFS : Écriture d'un fichier

- Le client indique au NameNode qu'il souhaite écrire un bloc.
- Celui-ci lui indique le DataNode à contacter.
- Le client envoie le bloc au Datanode.
- Les DataNodes répliquent le bloc entre eux.
- Le cycle se répète pour le bloc suivant.



HDFS : Lecture d'un fichier



- Le client indique au NameNode qu'il souhaite lire un fichier.
- Celui-ci lui indique sa taille et les différents DataNode contenant les N blocs.
- Le client récupère chacun des blocs à un des DataNodes.
- Si un DataNode est indisponible, le client le demande à un autre.

Conclusion

HDFS avec l'aide de NameNode et DataNode :

- Stocke de manière fiable des fichiers très volumineux sur des ordinateurs d'un grand cluster.
- Stocke chaque fichier sous forme de séquence de blocs. La réplication de bloc offre une tolérance aux pannes. Il offre une haute disponibilité, car les données sont hautement disponibles en dépit d'une panne matérielle.
- Lorsqu'une machine ou un matériel plante, nous pouvons accéder aux données d'un autre chemin.

Fin

Dr. H. EL BOUHISSI BRAHAMI