

# HDFS

Tomás Fernández Pena

Máster en Tecnologías de Análisis de Datos Masivos: Big Data

Universidade de Santiago de Compostela

Tecnologías de Computación para Datos Masivos

Material bajo licencia [Creative Commons Attribution-ShareAlike 4.0 International \(CC BY-SA 4.0\)](https://creativecommons.org/licenses/by-sa/4.0/)

[citius.usc.es](https://citius.usc.es)



Centro Singular de Investigación  
en Tecnologías Intelixentes

[Filesystems](#)

[Línea de comandos](#)

[Interfaz Java](#)

[Herramientas](#)

[Otras interfaces](#)

[Otros aspectos](#)

## Índice

- 1 Filesystems en Hadoop
- 2 Interfaz en línea de comandos
- 3 Interfaz Java
- 4 Herramientas para la gestión del HDFS
- 5 Otras interfaces a HDFS
- 6 Otros aspectos

# Índice

- 1 Filesystems en Hadoop
- 2 Interfaz en línea de comandos
- 3 Interfaz Java
- 4 Herramientas para la gestión del HDFS
- 5 Otras interfaces a HDFS
- 6 Otros aspectos

**ciTUS**

HDFS, TCDM

Filesystems

Línea de comandos

Interfaz Java

Herramientas

Otras interfaces

Otros aspectos

## Filesystems en Hadoop

Hadoop tiene una noción abstracta de los filesystems

- HDFS es un caso particular de filesystem

Algunos filesystems soportados:

FS	URI	Descripción
Local	<i>file</i>	Disco local
HDFS	<i>hdfs</i>	Sistema HDFS
HFTP	<i>hftp</i>	RO acceso a HDFS sobre HTTP
HSFTP	<i>hsftp</i>	RO acceso a HDFS sobre HTTPS
WebHDFS	<i>webhdfs</i>	RW acceso a HDFS sobre HTTP
S3 (nativo)	<i>s3n</i>	Acceso a S3 nativo
S3 (block)	<i>s3</i>	Acceso a S3 en bloques

Ejemplo:

- `hadoop fs -ls file:///home/pepe`

Para usar con HDFS se recomienda el comando `hdfs dfs`:

- `hdfs dfs -help`

**ciTUS**

HDFS, TCDM

# Interactuar con HDFS

Tres modos principales:

1. Usando línea de comandos: comando `hdfs dfs`
  - ▶ Permite cargar, descargar y acceder a los ficheros desde línea de comandos
  - ▶ Vale para todos los filesystems soportados
2. Usando el interfaz web
3. Programáticamente: API Java
4. Mediante otras interfaces: WebHDFS, HFTP, HDFS NFS Gateway

## Índice

1 Filesystems en Hadoop

2 Interfaz en línea de comandos

3 Interfaz Java

4 Herramientas para la gestión del HDFS

5 Otras interfaces a HDFS

6 Otros aspectos

## Interfaz en línea de comandos (I)

### Algunos comandos de manejo de ficheros

Comando	Significado
<code>hdfs dfs -ls &lt;path&gt;</code>	Lista ficheros
<code>hdfs dfs -ls -R &lt;path&gt;</code>	Lista recursivamente
<code>hdfs dfs -cp &lt;src&gt; &lt;dst&gt;</code>	Copia ficheros HDFS a HDFS
<code>hdfs dfs -mv &lt;src&gt; &lt;dst&gt;</code>	Mueve ficheros HDFS a HDFS
<code>hdfs dfs -rm &lt;path&gt;</code>	Borra ficheros en HDFS
<code>hdfs dfs -rm -r &lt;path&gt;</code>	Borra recursivamente
<code>hdfs dfs -cat &lt;path&gt;</code>	Muestra fichero en HDFS
<code>hdfs dfs -tail &lt;path&gt;</code>	Muestra el final del fichero
<code>hdfs dfs -stat &lt;path&gt;</code>	Muestra estadísticas del fichero
<code>hdfs dfs -mkdir &lt;path&gt;</code>	Crea directorio en HDFS
<code>hdfs dfs -chmod ...</code>	Cambia permisos de fichero
<code>hdfs dfs -chown ...</code>	Cambia propietario/grupo de fichero
<code>hdfs dfs -du &lt;path&gt;</code>	Espacio en bytes ocupado por ficheros
<code>hdfs dfs -du -s &lt;path&gt;</code>	Espacio ocupado acumulado
<code>hdfs dfs -count &lt;paths&gt;</code>	Cuenta nº dirs/ficheros/bytes

## Interfaz en línea de comandos (II)

### Movimiento de ficheros del sistema local al HDFS:

Comando	Significado
<code>hdfs dfs -put &lt;local&gt; &lt;dst&gt;</code>	Copia de local a HDFS
<code>hdfs dfs -copyFromLocal ...</code>	Igual que -put
<code>hdfs dfs -moveFromLocal ...</code>	Mueve de local a HDFS
<code>hdfs dfs -get &lt;src&gt; &lt;loc&gt;</code>	Copia de HDFS a local
<code>hdfs dfs -copyToLocal ...</code>	Copia de HDFS a local
<code>hdfs dfs -getmerge ...</code>	Copia y concatena de HDFS a local
<code>hdfs dfs -text &lt;path&gt;</code>	Muestra el fichero en texto

# Interfaz en línea de comandos (III)

Otros comandos:

Comando	Significado
<code>hdfs dfs -setrep &lt;path&gt;</code>	Cambia el nivel de replicación
<code>hdfs dfs -test -[defsz] &lt;path&gt;</code>	Tests sobre el fichero
<code>hdfs dfs -touchz &lt;path&gt;</code>	Crea fichero vacío
<code>hdfs dfs -expunge</code>	Vacía la papelera
<code>hdfs dfs -usage [cmd]</code>	Ayuda uso de comandos

Más información: <http://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/FileSystemShell.html>

## Índice

- 1 Filesystems en Hadoop
- 2 Interfaz en línea de comandos
- 3 Interfaz Java
- 4 Herramientas para la gestión del HDFS
- 5 Otras interfaces a HDFS
- 6 Otros aspectos

# Interfaz Java

API que permite interactuar con los filesystems soportados por Hadoop

- Utiliza la clase abstracta

`org.apache.hadoop.fs.FileSystem`

Otras clases de interés en `org.apache.hadoop.fs` y `org.apache.hadoop.io`

- `Path`: representa a un fichero en un `FileSystem`
- `FileStatus`: información del fichero
- `FSDaataInputStream`: stream de entrada de datos para un fichero, con acceso aleatorio
- `FSDaataOutputStream`: stream de salida de datos para un fichero
- `IOUtils`: Funcionalidades para I/O

## Ejemplo: lectura de un fichero en HDFS

```
public class FileSystemCat {  
    public static void main(String[] args) throws Exception {  
        String uri = args[0];  
        // Configuracion por defecto  
        Configuration conf = new Configuration();  
        // Objeto para acceder al filesystem HDFS  
        FileSystem fs = FileSystem.get(URI.create(uri), conf);  
        // InputStream  
        FSDaataInputStream in = null;  
        try {  
            // Abre el FSDaataInputStream con el PATH indicado  
            in = fs.open(new Path(uri));  
            // Copia con un buffer de 4096 bytes  
            // No cierra los buffers al terminar (false)  
            IOUtils.copyBytes(in, System.out, 4096, false);  
        } finally {  
            IOUtils.closeStream(in);  
        }  
    }  
}
```

## Ejecución del código anterior

Compilación: comando `javac` (suponiendo package `test`)

```
$ javac -cp $(hadoop classpath) test/FileSystemCat.java
```

Definir correctamente la variable `HADOOP_CLASSPATH` y usar el comando `hdfs` para lanzar el fichero `class`

```
$ export HADOOP_CLASSPATH="."
```

```
$ hdfs test.FileSystemCat fichero_en_HDFS
```

También es posible obtener el fichero `jar` y ejecutarlo con `hadoop jar` (no es una aplicación YARN)

## FSDataInputStream

Interfaces implementadas por `FSDataInputStream`:

1. **Seekable**: permite movernos a una posición en el fichero (método `seek`)
2. **PositionedReadable**: permite copiar a un buffer partes de un fichero

# Escritura de ficheros

Dos métodos de `FileSystem` para abrir los ficheros para escritura:

1. `create`: crea un fichero para escritura (crea los directorios padre, si es preciso)
2. `append`: abre un fichero para añadir datos

Ambos métodos devuelven un `FSDataOutputStream`

- `FSDataOutputStream` no permite seek (solo escritura al final del fichero)
- El método `hflush()` garantiza coherencia, los datos son visibles para nuevos lectores
- El método `hsync()` garantiza que los datos se mandan a disco (pero pueden estar en la caché del disco)

## Otras operaciones con ficheros y directorios

Crear un directorio:

- Método `mkdirs` de `FileSystem`

Información sobre ficheros y directorios:

- Métodos `getFileStatus` y `listStatus` de `FileSystem`
- Clase `FileStatus`

Patrones de nombres de ficheros (*globbing*)

- Método `globStatus` de `FileSystem`
- Interfaz `PathFilter`, para filtrar con expresiones regulares

Borrar ficheros o directorios, de forma recursiva o no:

- Método `delete` de `FileSystem`



# Otras herramientas para mover datos

Es posible mover datos a/desde HDFS usando otras herramientas

- **distcp** Transferir datos en paralelo entre dos filesystems Hadoop

- ▷ Ejemplo

```
hadoop distcp hdfs://nnode1/foo hdfs://nnode2/bar
```

- ▷ Aplicación MapReduce map-only

- ▷ Puede usar otros filesystems (HFTP, WebHDFS, etc.)

- ▷ Interesante para mover cantidades masivas de datos

- ▷ Más opciones: `hadoop distcp`

- **Apache Flume** servicio para recoger, agregar y mover grandes cantidades de datos de log a HDFS

- **Apache Sqoop** transferencia masivas de datos entre bases de datos estructuradas y HDFS

## Índice

1 Filesystems en Hadoop

2 Interfaz en línea de comandos

3 Interfaz Java

4 Herramientas para la gestión del HDFS

5 Otras interfaces a HDFS

6 Otros aspectos

# Herramientas para la gestión del HDFS

Hadoop proporciona un conjunto de herramientas para chequear y optimizar el HDFS

- `hdfs dfsadmin`: obtiene información del estado del HDFS
- `hdfs fsck`: chequeo del filesystem
- `hdfs balancer`: herramienta de rebalanceo de bloques entre datanodes

```
hdfs dfsadmin
```

Algunas opciones (usar `hdfs dfsadmin` **comando**)

Comando	Significado
<code>-help</code>	Ayuda
<code>-report</code>	Muestra estadísticas del filesystem
<code>-setQuota</code>	Fija una cuota en el número de nombres en un directorio (nº de ficheros/directorios)
<code>-clrQuota</code>	Borra la cuota de nombres
<code>-setSpaceQuota</code>	Fija una cuota en el espacio ocupado en un directorio
<code>-clrSpaceQuota</code>	Borra la cuota de espacio
<code>-refreshNodes</code>	Actualiza los nodos que se pueden conectar
<code>-safemode</code>	fija o chequea el <i>safe mode</i>
<code>-saveNameSpace</code>	en <i>safe mode</i> , salva el filesystem en memoria a un nuevo fichero <code>fsimage</code> y resetea el fichero <code>edits</code>

# hdfs fsck

## Chequea la salud de los ficheros en HDFS

### ■ Chequea los bloques:

- ▷ *Over-replicated*: con replicas de más
- ▷ *Under-replicated*: con replicas de menos
- ▷ *Misreplicated*: replicas mal colocadas
- ▷ Corruptos
- ▷ *Missing replicas*: sin réplicas

### ■ Ejemplos:

- ▷ Chequea recursivamente todo el HDFS  
`hdfs fsck /`
- ▷ Informa del número de bloques de un fichero y su localización

```
hdfs fsck /user/pepe/foo -files -blocks -racks
```

## Índice

1 Filesystems en Hadoop

2 Interfaz en línea de comandos

3 Interfaz Java

4 Herramientas para la gestión del HDFS

5 Otras interfaces a HDFS

6 Otros aspectos

# Otras interfaces a HDFS

## Otros modos de acceder a HDFS

- **WebHDFS**: proporciona una API REST para acceder a HDFS mediante HTTP
  - ▷ Necesita acceso a los nodos del cluster (los datos se transmiten directamente desde los nodos)
  - ▷ Activada mediante la propiedad `dfs.webhdfs.enabled` del fichero `hdfs-site.xml` del Namenode (por defecto, `true`)
- **HttpFS**: servidor que proporciona un gateway REST HTTP que soporta las operaciones de HDFS (lectura/escritura)
  - ▷ El servidor de HttpFS actúa como gateway: permite acceder a los datos en HDFS detrás de un firewall
- **HDFS NFS Gateway**: soporta NFSv3 y permite que HDFS sea montado como parte del sistema de ficheros local del cliente
  - ▷ Permite usar HDFS como un sistema de ficheros UNIX local
  - ▷ Permite copiar ficheros de HDFS al sistema de ficheros local y viceversa

## Índice

- 1 Filesystems en Hadoop
- 2 Interfaz en línea de comandos
- 3 Interfaz Java
- 4 Herramientas para la gestión del HDFS
- 5 Otras interfaces a HDFS
- 6 Otros aspectos**

# Namenode principal

## Estructura de directorios

```
${dfs.namenode.name.dir}/  
├── current  
│   ├── VERSION  
│   ├── edits_000000000000000000000001-0000000000000000000019  
│   ├── edits_inprogress_0000000000000000000020  
│   ├── fsimage_000000000000000000000000  
│   ├── fsimage_000000000000000000000000.md5  
│   ├── fsimage_000000000000000000000019  
│   ├── fsimage_000000000000000000000019.md5  
│   └── seen_txid  
└── in_use.lock
```

## Ficheros en el namenode

### Ficheros en `dfs.namenode.name.dir`

- **VERSION** información sobre la versión de HDFS
- Ficheros `edits_startID-endID`: logs de transacciones ya finalizadas
- Fichero `edits_inprogress_startID`: logs de transacciones actuales
- Ficheros `fsimage`: información de los metadatos del filesystem
  - ▷ Contiene información de directorios y ficheros, incluyendo los bloques (inodos) que los forman
  - ▷ La localización de los bloques en los Datanodes se guarda en memoria

# Inicio del Namenode

Cuando se inicia el Namenode:

1. Carga el último `fsimage` en memoria y aplica las modificaciones indicadas en `edits`
2. Con esta imagen reconstruida, crea un nuevo `fsimage` y un `edits` vacío
3. Espera a que los Datanodes le envíen información de los bloques que tienen
  - ▶ Esta información se guarda en memoria

## Modo seguro

Durante la inicialización, el sistema está en modo seguro (*safe mode*)

- Solo permite acceso de lectura
- El modo seguro termina 30 segundos después de que el 99.9 % de los bloques alcancen un nivel mínimo de replicación
- Propiedades ajustables:

Propiedad	Por defecto
<code>dfs.namenode.replication.min</code>	1
<code>dfs.namenode.safemode.threshold-pct</code>	0.999
<code>dfs.namenode.safemode.extension</code>	30 s

## Checkpoint node (aka Namenode secundario)

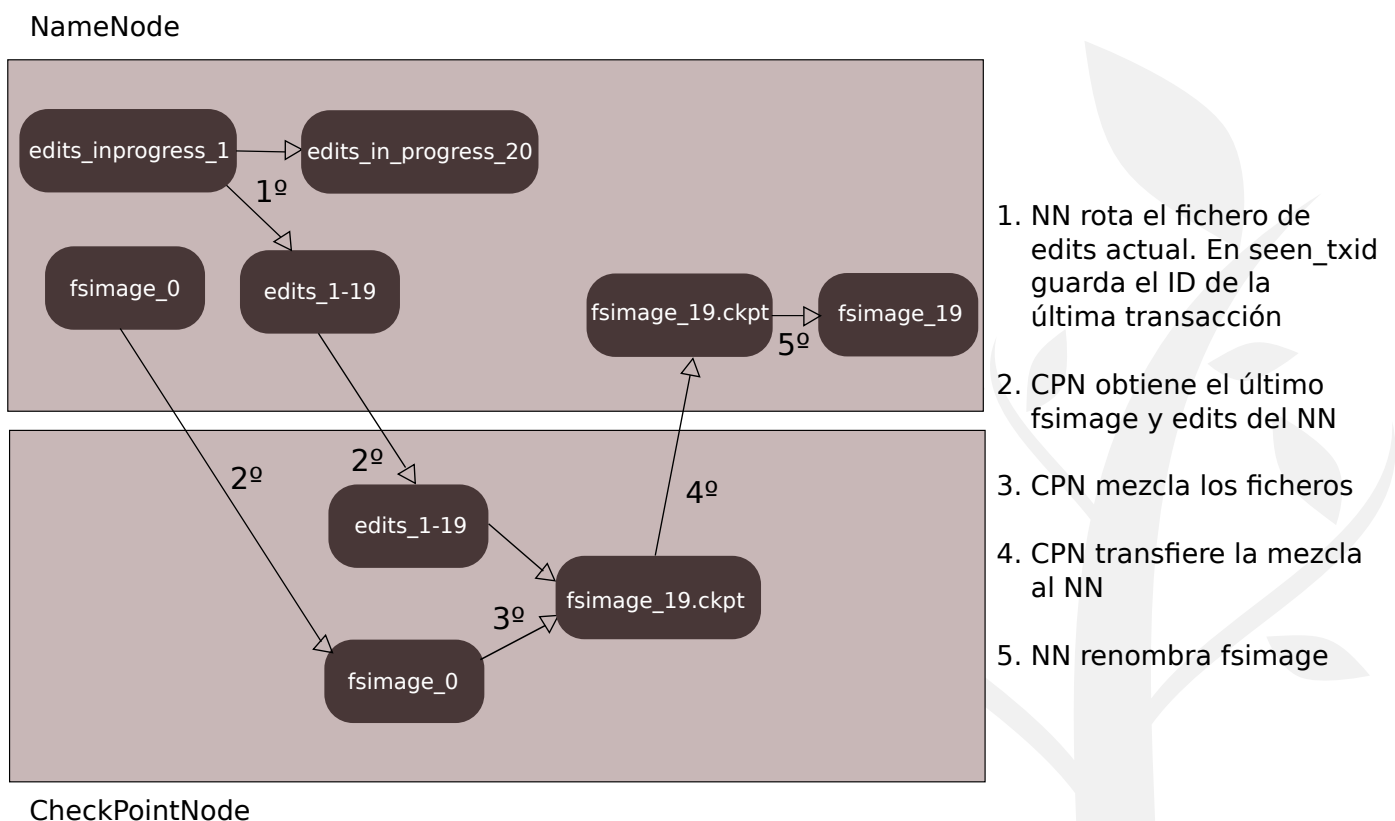
En un sistema ocupado, el fichero del `edits` puede crecer demasiado

- El **Checkpoint Node** se ocupa de mezclar `edits` y `fsimage` para inicializarlo
  - ▷ Proceso costoso en recursos
  - ▷ El CPN tiene requisitos de memoria similares a los del NN
- Checkpoint realizado cada hora  
(`dfs.namenode.checkpoint.period`) o cada 1 M transacciones (`dfs.namenode.checkpoint.txns`)
- Se puede cambiar por un **Backup Node**
  - ▷ Replica completa de la memoria del NN (necesita la misma cantidad de memoria)
  - ▷ Realiza los checkpoints

En caso de fallo total del Namenode, se puede **recuperar el último checkpoint**

- Iniciar el demonio del Namenode usando `hdfs namenode -importCheckpoint`

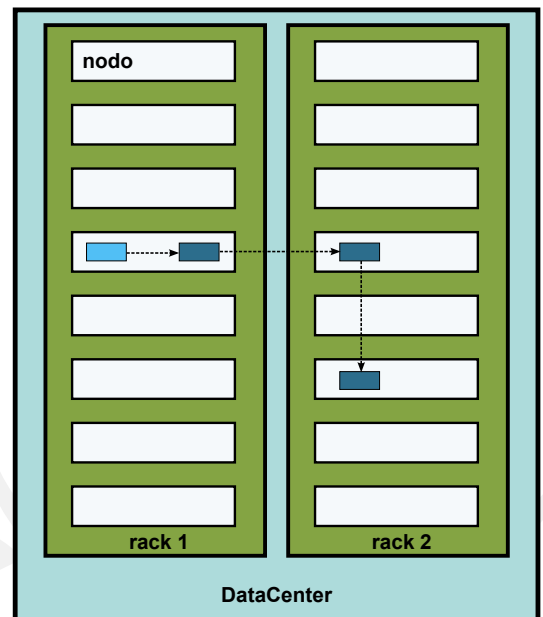
## Checkpoint node



# Localización de las réplicas

Política por defecto:

- 1ª réplica: en el nodo del cliente o en un nodo al azar
- 2ª réplica: en un **rack** diferente de la primera (elegido al azar)
- 3ª réplica: en el mismo rack que la 2ª, pero en otro nodo
- Otras réplicas: al azar (se intenta evitar colocar demasiadas réplicas en el mismo rack)



- Más información [hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html#Data\\_Replication](http://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html#Data_Replication)

## Problemas con el Namenode

El Namenode es un *single point of failure* (SPOF)

- Si falla es imposible acceder a los datos
- Posibilidad de recuperación a partir de los checkpoints
- Conveniente guardar varias réplicas de los datos del namenode (RAID, indicar en `dfs.namenode.name.dir` directorios en diferentes máquinas, etc)

Mejoras en la versión 2.0

- HDFS High-Availability
- HDFS Federation



## HDFS High-Availability

Un par de Namenodes en configuración activo-standby

- si falla el Namenode activo, el otro ocupa su lugar

Consideraciones

- Los Namenodes deben usar un almacenamiento compartido de alta disponibilidad
- Los Datanodes deben enviar informes de bloques a los dos Namenodes (el block mapping va en memoria, no en disco)
- Los Clientes deben manejar el fallo del Namenode de forma transparente

Más información: [hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HDFSHighAvailabilityWithNFS.html](http://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HDFSHighAvailabilityWithNFS.html),  
[hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HDFSHighAvailabilityWithQJM.html](http://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HDFSHighAvailabilityWithQJM.html)

## HDFS Federation

El Namenode mantiene, en memoria, referencias a cada fichero y bloque en el filesystem

- problemas de escalabilidad

HDF Federation, introducida en la versión 2.0

- Permite usar varios Namenodes
- Cada uno gestiona una porción del espacio de nombres del filesystem
- Los Namenodes no se coordinan entre sí
- Cada Datanodes se registra con todos los Namenodes

Más información: [hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/Federation.html](http://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/Federation.html)

# Hadoop v3

## Novedades en HDFS v3

- Uso de **códigos de borrado** (*erasure coding*) para reducir el overhead de la replicación
  - ▷ Reduce el overhead a no más del 50 %
  - ▷ Ejemplo: ficheros de 6 bloques:
    - replicación x3: 18 bloques
    - EC: 9 bloques (6 datos + 3 paridad)
  - ▷ Implica un mayor coste de procesamiento
- Soporte de múltiples NameNodes en stand-by
- Soporte de balanceo de datos intra-nodo