Isolating FileSystem operations

https://github.com/matteobertozzi/hbase/tree/regionfs

Motivations

- DIR_NAME constants everywhere
- new Path() everywhere
- Each test create is own mocked fs layout
  mkdir(region), mkdir(new Path(region, family), …
- No way to change the fs layout without touching all the code (and breaking stuff).
  (see last slide for a sneak peek of the proposal for a future fs layout)

Goals

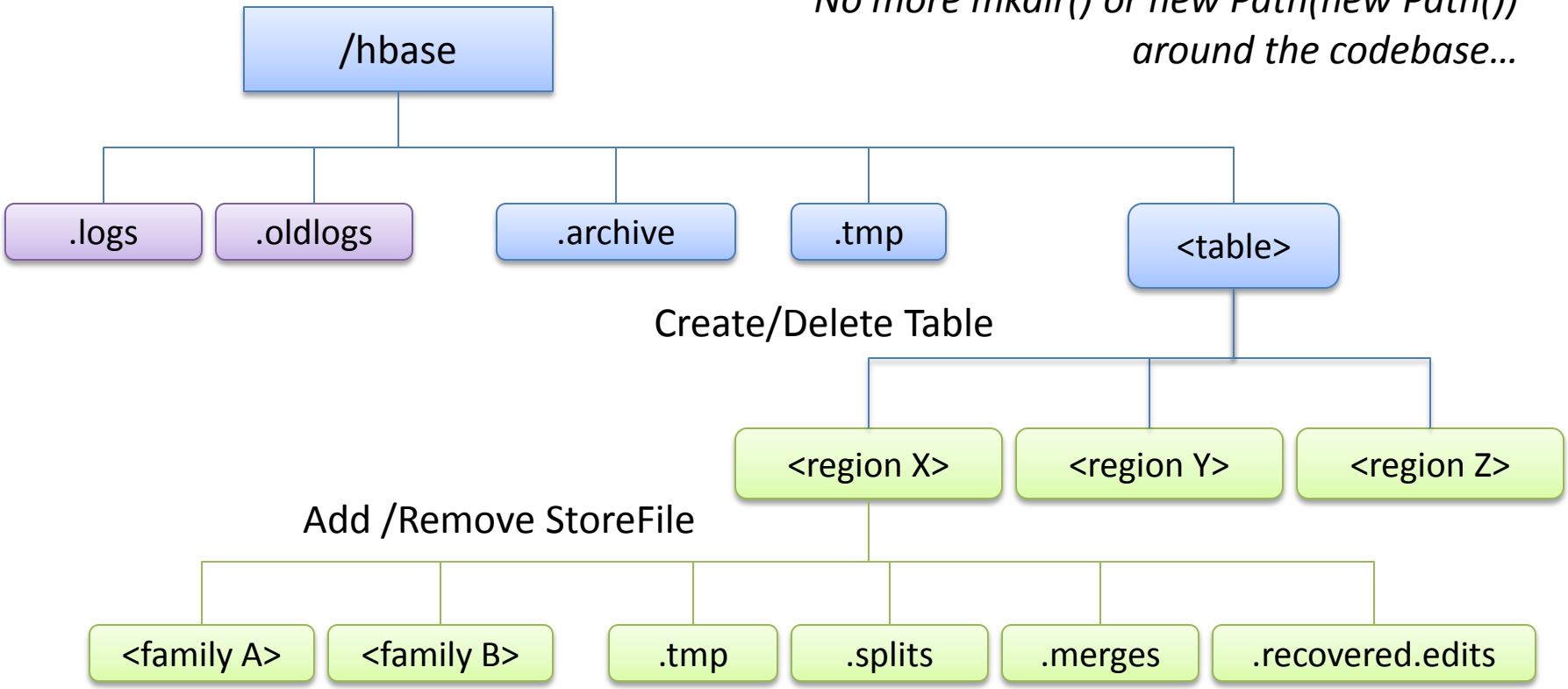
- Add the ability to create an on-disk region without using HRegion and initializing everything (e.g. CreateTable, tests, …)
- Removing all the fs.listStatus() calls
- Removing all the new Path() from the code
- Cleanup the tests to use the new classes instead of creating mock objects

Isolating FileSystem operations

- MasterFileSystem
  - Bootstrap/log splitting
  - Create/Delete Table (using .tmp dir and archiver)
  - Listing available tables
  - …
- RegionFileSystem
  - Create Hregion structure (.tmp, families, …)
  - Listing families/files (load store)
  - Creating StoreFiles in .tmp and commit to the family/dir
  - …

Step 1: Isolating FileSystem operations

No more mkdir() or new Path(new Path())
around the codebase…



The user should not know about
dir names or .tmp directory!

HMasterFileSystem
HRegionFileSystem

```
StoreFile storeFile = regionFs.createNewFile(family);
storeFile.createWrite().append(…)
regionFs.commit(storeFile);
```

Step by Step

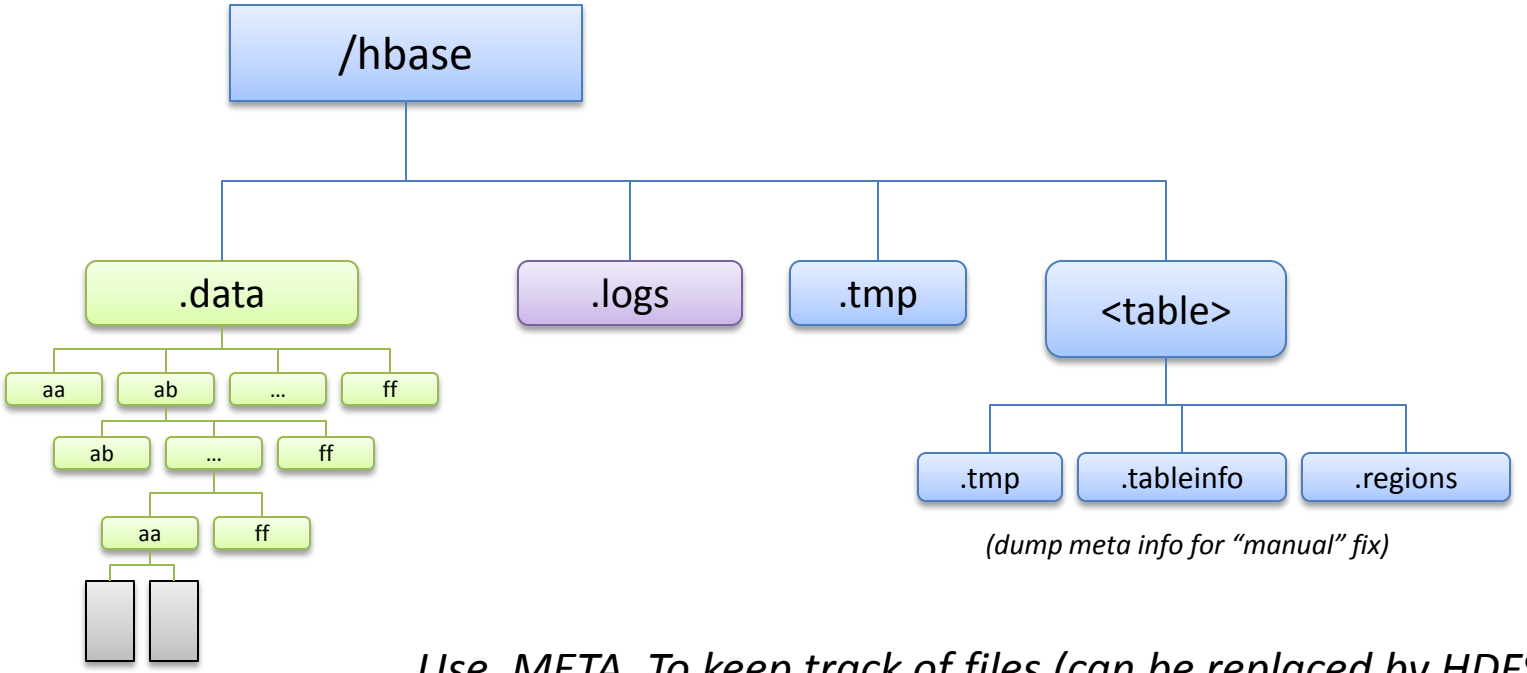
- Add the new HRegionFileSystem class and use it in HRegion
  - Move all the dir name constants inside (SPLIT_DIR, TMP_DIR, …)
  - Move all the region on-disk creation (.regioninfo, directories, …)
    Useful for small tests, that will not need anymore to create by hand the dir layout
- Extend the HRegionFileSystem use to HStore & co
  - Move to RegionFs the fs.listStatus() and isReference()/isLink() operations.
  - Move the file tmpCreation+commit logic to regionFs
- Make the Archiver transparent using MasterFs & RegionFs
- Cleaning up Tests to use HRegionFileSystem and avoid mkdirs/listStatus & co
- …

Future

reduce failures by
removing lots of code & tricks

Why?

- Avoid half states on table creation/deletion/split/merge/…
  - We need Transactions between META & FS
- HFile/Log Archiver: Avoid to move files around with the risk of huge folders.
- HFileLink: Avoid the hack “if file is not here, try there”
- File sharing between different table without links “Clone Table”
- Simplify snapshot/restore reference code.
  - Scan & copy table/snapshot “meta” row instead of multiple fs.listStatus(), fs.createNew()
- Future
  - File Deduplication, bulk import file that already exists & copy table, avoid the N copy of the file and use the one already present (file sharing)
  - Switch to HDFS block instead of files (better compaction, less I/O by moving pointers instead of copying all the stale data)
- *…and more*



Use .META. To keep track of files (can be replaced by HDFS blocks)
Allows to share files between different tables
Avoid to move files around and create links/references
Snapshot/Splits/Merge are scan/put transaction.

/hbase/.data/d4/1d/8c/d41d8cd98f00b204e9800998ecf8427e

file name prefix
(no need to store that)
256 items per level

L0 256, L1 64K, L2 16M, L3 4G files