
CERN

External Tape Readers

Integration into CTA and OSM/Enstore cases

Jorge Camarero Vera / IT-SD-TAB

Introduction

- Tape Formats.
- Design and implementation of external tape format readers.
- Unit and functional testing. Case of OSM tape using CTA's CI system.

Motivation

- Enable migration of Tier1 and Tier2 systems to CTA
  - CTA must read tapes in external formats
  - Read-only capability, no writing has been implemented.
- Other migration tasks includes:
  - Migrating disk file metadata

Tape Formats - CTA

- CTA format. Similar to CASTOR tape format.
- Based in ISO/IEC 1001:2012.

Table 1.1: AUL label format

Tape Formats - Enstore / OSM

- The code for the OSM/Enstore-specific implementation was contributed by DESY/Fermilab.
- Enstore label is similar to CTA.

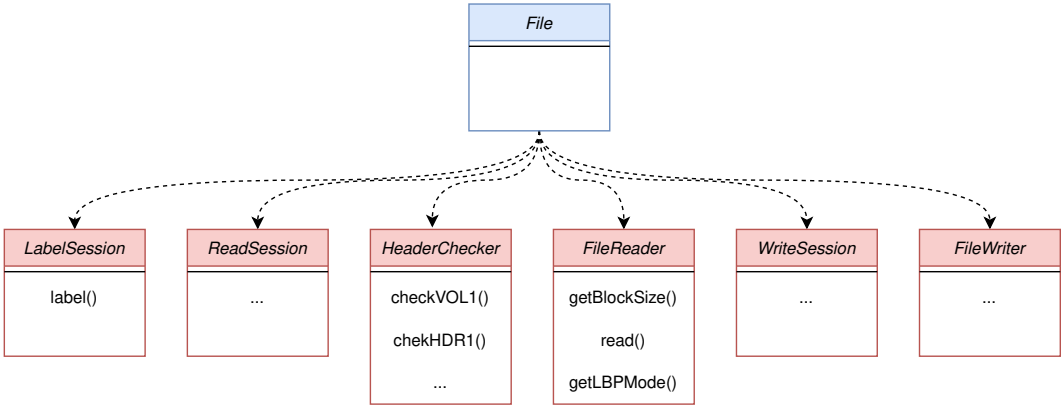
```
VOL1VR3025                                                    0
VOL1VR3026                          CTA                     023
```

- OSM label is different from CTA. Length of 64 kB.
- Use CPIO to read files.

New Tape Readers Creation

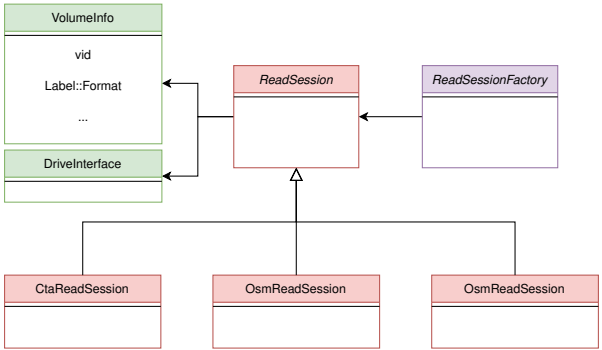
Refactoring and generalization of the previous code was needed to accommodate different tape file readers.

Refactoring of File.hpp/cpp

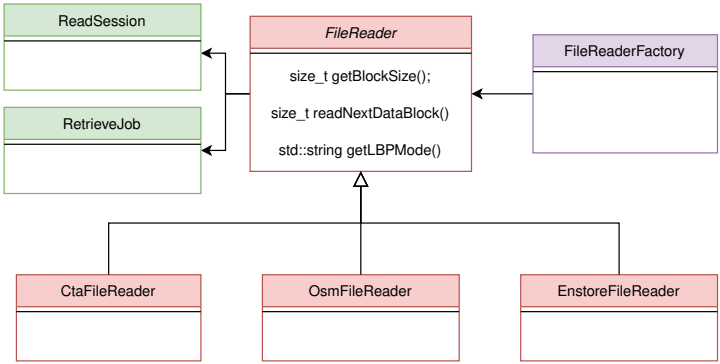


ReadSession

Class keeping track of a whole tape read session over a formatted tape.



FileReader


ReadSession
RetrieveJob
FileReader
size_t getBlockSize();
size_t readNextDataBlock()
std::string getLBPMode()
FileReaderFactory
CtaFileReader
OsmFileReader
EnstoreFileReader

Unit Testing - ReadSession

```
const std::string testString("Hello␣World!");
// Read it back and compare
const auto readSession = castor::tape::tapeFile::
   ReadSessionFactory::create(m_drive, m_volInfo, true);
ASSERT_NE(readSession, nullptr);
ASSERT_EQ(readSession->getCurrentFilePart(), castor::tape::
   tapeFile::PartOfFile::Header);
ASSERT_EQ(readSession->getCurrentFseq(), static_cast<uint32_t
   >(1));
ASSERT_EQ(readSession->isCorrupted(), false);
ASSERT_EQ(readSession->m_vid.compare(m_label), 0);
ASSERT_EQ(readSession->m_useLbp, true);
```

Unit Testing - FileReader

```
m_fileToRecall.positioningMethod = cta::PositioningMethod::
    ByBlock;
const auto reader = castor::tape::tapeFile::FileReaderFactory::
    create(readSession, m_fileToRecall);
size_t blockSize = reader->getBlockSize();
ASSERT_EQ(blockSize, m_block_size);
char *data = new char[blockSize+1];
size_t bytes_read = reader->readNextDataBlock(data, blockSize);
data[bytes_read] = '\0';
ASSERT_EQ(bytes_read, static_cast<size_t>(testString.size()));
ASSERT_EQ(testString.compare(data), 0);
delete[] data;
```

System Test - OSM Tape

A system test for OSM tape format was created to run on the CI kubernetes environment.

Mounting Image of the Tape

To simulate an OSM tape in the library we mount an image of a tape in MHVTL.

Mounting Image of the Tape

```
# Load a virtual tape into the tape drive and rewind it.
mtx -f /dev/smc load 1 0
mt -f $device rewind
# The header files have 4 more bytes in the git file
truncate -s -4 /osm-mhvtl/L08033/L1
touch /osm-tape.img
# Copy the headers and files
dd if=/osm-mhvtl/L08033/L1 of=/osm-tape.img bs=32768
dd if=/osm-mhvtl/L08033/L2 of=/osm-tape.img bs=32768 seek=1
dd if=/osm-tape.img of=\$device bs=32768 count=2
dd if=/osm-mhvtl/L08033/file1 of=\$device bs=262144 count=202
dd if=/osm-mhvtl/L08033/file2 of=\$device bs=262144 count=202

mt -f \$device rewind
```

Reading an OSM Tape - Mocking Catalogue

```
class OSMTapeCatalogue : public cta::catalogue::
    DummyTapeCatalogue {
public:
  using LabelFormat = cta::common::dataStructures::Label::
      Format;
  OSMTapeCatalogue() = default;

  LabelFormat getTapeLabelFormat(const std::string& vid) const
      override {
    return LabelFormat::OSM;
  }
};
```

Reading an OSM Tape - Creation MHVTL Drive

```
// Create drive object and open tape device
m_dev.product = "MHVTL";
m_dev.nst_dev = m_nstDev;
m_drive.reset(castor::tape::tapeserver::drive::createDrive(
    m_dev, m_sWrapper));

// Checks if the drive ready to use the tape installed loaded
    into it.
m_drive->waitUntilReady(5);
m_drive->rewind();
```

Reading an OSM Tape - Setting Volume Info

```
castor::tape::tapeserver::daemon::VolumeInfo m_volInfo;
// Volume ID of the Tape
m_volInfo.vid = "L08033";
m_volInfo.nbFiles = 1;
// Get Tape format from Catalogue
m_volInfo.labelFormat = static_cast<OSMTapeCatalogue*>(
  m_catalogue->Tape().get())->getTapeLabelFormat(
    m_volInfo.vid);
// Setting type to Retrieve, we want to read
m_volInfo.mountType = cta::common::dataStructures::MountType::
   Retrieve;
```

Reading an OSM Tape - ReadSeasson creation

And creation of a Job to retrieve a file from the tape

```
// Create Read Session OSM
auto readSession = castor::tape::tapeFile::ReadSessionFactory::
    create(*m_drive, m_volInfo, false);
BasicRetrieveJob fileToRecall;
// Fill the RetrieveJob with the parameters
fileToRecall.selectedCopyNb = 0;
...

fileToRecall.positioningMethod = cta::PositioningMethod::
    ByBlock;
```

Reading an OSM Tape - FileReader creation and Reading

```
// Create Read File OSM
auto reader = castor::tape::tapeFile::FileReaderFactory::create
   (readSession, fileToRecall);
size_t bs = reader->getBlockSize();
char *data = new char[bs+1];
size_t j = 0;
while (j < 100) {
    reader->readNextDataBlock(data, bs);
    j++;
}
```

Summary

- Tape formats
- Adapt CTA default File Reader to accept different kind of Tape Formats.
- Creation of Unit Tests
- System test to check full integration


CERN