

ITKv4 DICOM Status

Mathieu Malaterre, Mark Roden
Alexandre Gouaillard, William Ryan, Daniel
Blezek

Summary

1. DICOM DataSet abstraction
2. DICOM Query/Retrieve (network protocol) abstraction
3. Streaming
4. RTSTRUCT
5. MOSAIC

itk::DICOMDataSet

- itk::DICOMDataSet should replace the itk::MetaDataDictionary implementation for storing DICOM Attributes.
- Will allow re-use of third party lib (eg. DCMTK or GDCM), it will hold a pointer to the internal implementation.
- Interface will implements Supp 118 for querying of attributes
- One itk::DICOMDataSet per file (3D multiframe is not a Series of 2D frames)

Supp 118 queries

```
(0008,1090) LO [RHAPSODE] # 8, 1 ManufacturersModelName
(0008,2111) ST [JPEG 2000 irreversible (lossy) 69:1] # 36, 1 DerivationDescription
...
(0008,9215) SQ (Sequence with undefined length #=1) # u/1, 1 DerivationCodeSequence
  (fffe,e000) na (Item with undefined length #=3) # u/1, 1 Item
    (0008,0100) SH [113040] # 6, 1 CodeValue
    (0008,0102) SH [DCM] # 4, 1 CodingSchemeDesignator
    (0008,0104) LO [Lossy Compression] # 18, 1 CodeMeaning
  (fffe,e00d) na (ItemDelimitationItem) # 0, 0 ItemDelimitationItem
(fffe,e0dd) na (SequenceDelimitationItem) # 0, 0 SequenceDelimitationItem
```

```
const itk::DICOMDataSet &ds = dicomio->GetDICOMDataSet();
string value;
string query = "/DicomNativeModel/DicomAttribute[@keyword='DerivationCodeSequence']"
  "/Item[@number=1]/DicomAttribute[@keyword='CodeMeaning']/Value[@number=1]";
ds.GetValueFromQuery( query, value );
std::cout << "Code Meaning=" << value << std::endl;
```

Filter DICOM DataSet(s)

```
ImageIOType::Pointer dicomio = ImageIOType::New();
ReaderType::Pointer reader = ReaderType::New();
reader->SetFileNames( filenames );
reader->SetImageIO( dicomio );
reader->Update();
FilterType::Pointer filter = FilterType::New();
filter->SetInput( reader->GetOutput() );
ImageIOType::Pointer dicomio2 = ImageIOType::New();
DICOMFilterType::Pointer anonymize = DICOMFilterType::New();
anonymize->SetInput( dicomio->GetDICOMDataSetArray() );
WriterType::Pointer writer = WriterType::New();
writer->SetInput( filter->GetOutput() );
dicomio2->SetDICOMDataSetArray(
        anonymize->GetDICOMDataSetArray() );
writer->SetImageIO( dicomio2 );
writer->SetFileNames( outputfilenames );
writer->Update();
```

Implementation Specific (GDCM)

```
itk::DICOMDataSet &dicomds = io->GetDICOMDataSet();  
itk::GDCMDataSet &gdcmds =  
    dynamic_cast<itk::DICOMDataSet&>(dicomds);  
gdcmd::DataSet &ds = gdcmds->GetGDCMDataSet();  
  
const gdcmd::PrivateTag  
    tstringdata(0x33,0x1f,"GEMS_GENIE_1");  
  
if( !ds.FindDataElement( tstringdata ) ) return 1;  
  
const gdcmd::DataElement& stringdata =  
    ds.GetDataElement( tstringdata );  
  
ProcessSDOHeader( stringdata );
```

DICOM Discussion

- http://www.itk.org/Wiki/ITK_Release_4/DICOM
- Represented:
 - CoSMo
 - Mayo
 - CTK/NAMIC
 - *INRIA*

DICOM Query/Retrieve

- **C-ECHO**
- **C-STORE**
- **C-FIND** (Patient & Study)
 - Patient/Study Only will *not* be available
- **C-GET**: will *not* be available
- **C-MOVE**
 - No external application required
 - C-STORE SCP association using different port

C-ECHO

- An implementation for Service Class User (SCU) for the Verification SOP Class. Sends a DICOM C-ECHO message to a Service Class Provided (SCP) and waits for an answer.
- Used to verify basic DICOM connectivity
- PING !

itk::DICOMSCU (C-ECHO)

```
itk::DICOMSCU scu;  
scu.SetPeer( "mi2b2.slicer.org" );  
scu.SetCalledAETitle( "MI2B2" );  
scu.SetPort( 11112 );  
try {  
    scu.SendEcho();  
} catch ( itk::DICOMException & e ) {  
    std::cerr << e.what() << std::endl;  
}
```

C-STORE

- This is used to send DataSet using the DICOM protocol (over TCP/IP).
- Equivalent to HTTP PUT



itk::DICOMSCU (C-STORE)

```
vector<string> filenames = ...;  
itk::DICOMSCU scu;  
scu.SetPeer( "mi2b2.slicer.org" );  
scu.SetCalledAETitle( "MI2B2" );  
scu.SetPort( 11112 );  
try {  
    scu.SendStore( filenames );  
} catch ( itk::DICOMException & e ) {  
    std::cerr << e.what() << std::endl;  
}
```

C-FIND

- This is used to query an SCP using the C-FIND message.
- In SQL

```
select PATIENT.PatientID,  
       STUDY.StudyDate,  
       STUDY.ModalitiesInStudy,  
       STUDY.StudyDescription  
  
from   PATIENT, STUDY  
  
where  PATIENT.PatientID = "99999"  
  
and    STUDY.Patient_fk = PATIENT.pk
```

itk::DICOMSCU (C-FIND)

```
itk::DICOMSCU scu;  
scu.Set...  
itk::SeriesQuery::Pointer query = SeriesQuery::New();  
query->SetPatientName( "X*" );  
vector<itk::DICOMDataSet> cfind_results;  
try {  
    cfind_results = scu.SendFind( query );  
} catch ( itk::DICOMException & e ) {  
    std::cerr << e.what() << std::endl;  
}
```

itk::DICOMSCU (C-FIND)

Open Questions

```
itk::SeriesQuery::Pointer query = SeriesQuery::New();  
query->SetPatientName( "*王^小東*" );
```

```
itk::SeriesQuery::Pointer query = SeriesQuery::New();  
query->SetPatientName( "*Ibáñez*" );
```

```
itk::SeriesQuery::Pointer query = SeriesQuery::New();  
query->SetPatientName( "*Jérôme*" );
```

```
itk::SeriesQuery::Pointer query = SeriesQuery::New();  
query->SetPatientName( "*Rüdiger*" );
```

C-GET

- This is essentially just a C-MOVE, except it does not require an extra step (another Association).
- This is not implemented in some private implementation.

C-MOVE

- This is used to retrieve DataSet using the DICOM protocol (over TCP/IP)
- Somewhat equivalent to HTTP GET

itk::DICOMSCU (C-MOVE)

```
itk::DICOMSCU scu;  
scu.SetPeer( "mi2b2.slicer.org" );  
scu.SetCalledAETitle( "MI2B2" );  
scu.SetPort( 11112 );  
scu.SetAETitle( "GDCMSCU" );  
scu.SetIncomingPort( 5678 );  
scu.SetOutputDirectory( "/tmp" );  
try {  
    scu.SendMove( cfind_results );  
} catch ( itk::DICOMException & e ) {  
    std::cerr << e.what() << std::endl;  
}
```

Server side configuration (dcm4chee)

The screenshot shows the 'AE List' page in a web browser. The browser's address bar displays 'http://localhost:8080/dcm4chee-web/ae.m'. The page features a navigation menu with options like 'Folder Trash', 'AE Management', 'Offline Storage', 'Worklist Console', 'MPPS Console', 'GP Worklist Console', 'GPPPS Console', 'User Admin', 'Audit Repository', and 'Logout'. Below the menu is a table with the following columns: 'AE Title', 'Hostname', 'Port', 'Cipher', 'Issuer', 'User ID', 'FS Group ID', and 'Description'. The table contains three entries: 'CDRECORD', 'DCM4CHEE', and 'GDCMSCU'. The 'GDCMSCU' entry is circled in red, and its 'Hostname' ('dicom.example.com') and 'Port' ('5678') are also circled in red.

AE Title	Hostname	Port	Cipher	Issuer	User ID	FS Group ID	Description	
CDRECORD	localhost	10104					Media Creation Server (part of dcm4chee)	
DCM4CHEE	localhost	11112		DCM4CHEE			This dcm4chee archive instance	
GDCMSCU	dicom.example.com	5678			admin			

Streaming

- Read/Write:
 - RAW
 - JPEG 2000
- User demand:
 - JPEG
 - JPEG-LS
 - RLE

JPEG 2000 Streaming

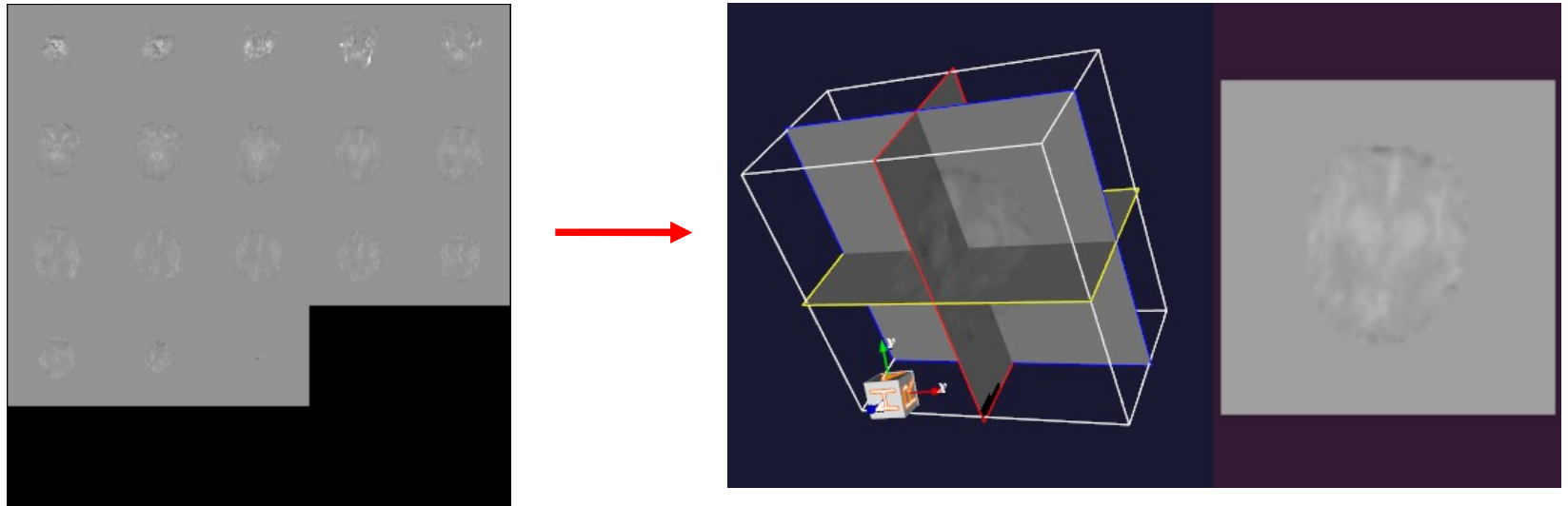
- Reading:
 - Support in OpenJPEG v2, using `opj_set_decode_area` API
- Writing:
 - Support in OpenJPEG v2 using `opj_write_tile`
 - Open Question: when user extent do not match tile requirement

JPEG/JPEG-LS/RLE

- No direct support for streaming.
 - Need to decompress the whole image, to access the last scanline
 - 16 bits based standard, worse case scenario is 4Gb dataset.

itk::MOSAICImageIO

- Subclass of itk::GDCMImageIO
- Identical to itk::GDCMImageIO, except image is *untiled*.



itk::RTSTRUCT

- 2D contours will be loaded as `itk::QuadEdgeMesh` (subclass of `itk::Mesh`)
- Provide a `itk::RTSTRUCTProperties` to attach properties to each individual `itk::QuadEdgeMesh` (eg. Generation Algorithm Name)

References

- <http://gdcm.git.sf.net>
- <http://github.com/malaterre/ITK/tree/itkdicomnetwork>