DCM API SURVEY

D5.1

VPH NOE - WP3 Imaging Tools Group

Leader: CREATIS

Involved: INRIA, UPF

Version & Date	Authors	Modifications
0.1 05/02/09	Frederic Cervenansky, Denis Friboulet, Laurent Guigues	Document creation
0.2 05/28/09		Reviewed by Yves Martelli, Pedro Omedas
1.0 06/02/09		Final version

Table of Contents

INTRODUCTION :

They are many open-source DICOM images reader libraries: gdcm V1, gdcm V2, dcmtk,… In many developments, developers often need to develop their own advanced **DCM API**: a layer component who could be plugged on third parties applications, to perform and maintain their medical images, and specifically DICOM files. Some tools (software, library) seem to fill this gap. In an effort to show most useful tools to share with VPH toolkit community, a survey of DCM API was done with specific requirements established by WP3 Imaging Tools subgroup.

1. REQUIREMENTS

WP3 Imaging Tools subgroup members have given an exhaustive list of features for advanced DCM API with general requirements (global for all partners) and with more specific features (corresponding to expertise fields of WP3 Imaging Tools subgroup members). From these lists, a list of necessary criteria is established.

CREATIS:

The features can be grouped by actions field:

- ✓ Index medical images (DICOM, analyze, Ultrasounds, jpeg, tiff, MHD): scan user's source directory or files and add medical images (DICOM, jpeg, own format, …) to a local DB if, available, medical informations (name, modality, …).
- ✓ Structure (in a hierarchy)/create DB: allow to represent medical images in a specific format (root |_ patient |_series [_images), whished by user.
- ✓ Manage DB (add/remove/filter/edit tags / anonymize): allow to manipulate data in DB.
- ✓ Connection to different DB (local or network).
- ✓ Copy on local: allow to save data (medical images) on a specific support.
- ✓ Synchronize DB and medical images: synchronize DB with directory to detect atomically changes on directory (remove, add new images) and/or on files.
- ✓ Settings management: DICOM tags choice, synchronization event, DICOM – images reader library, …
- ✓ Abstract interface: user's interface shall be abstract and allow implementation on a specific language (WxWidget, MFC, Qt,…) with if possible one implementation already done.
- ✓ Command line application: basic actions can be invoked (print db content, add files/directories, remove…), batch possibility.
- ✓ Anonymyze: on DB and/or on original data.
- ✓ PACS connection: push/send data on PACS server, query actions on PACS.
- ✓ Lesson on a specific port: detect automatically CD/DV with medical images or USB port for example.
- ✓ Third party application connection: given as output medical images selected.
- ✓ Documentation: sources should be very well documented and additional informations (for architecture or installation) should be present too.

UPF:

The DCM API should define standards to:

- ✓ access/modify DICOM tags in a DICOM file
- ✓ load/write DICOM image slices (from a file containing one or multiples slices) into a structure that contains data and image related meta data (size, spacing, origin, modality, rescale slope and intercept, orientation)
- ✓ load/write DICOM meshes (*not clear what the status of the standard is*)
- ✓ load/write DICOM signals (for example ECG, *not clear neither what the status of the standard is*)
- ✓ create a 3D volume from the slices with possible reorientation
- ✓ create a 3D+t volume from the slices or volumes
- ✓ add specific DICOM readers/writers (for example equipment constructor specific)
- ✓ query/retrieve/save data from/to a PACS server
- ✓ query/retrieve/save from/to a local database
- ✓ Not provide PACS support but a generic PACS object that is connected to a low level PACS library (for example DCMTK) which is independent.
- ✓ Provide local database storage, with interfaces for querying it and retrieving data but does not impose anything about the database format.
- ✓ Documentation on architecture and classes interfaces.

INRIA:

The required features for the DCM API were the same than other members' features. Apart from two additional conditions: a DCM API shall have few library dependencies (from GUI library for example) to extend inter-portability and a reference database to allow users tests. More, the DCM API shall have passed or be structured to pass the IHE connectathon (during few days, professionals and free developers test that their application communicate between them and can exchange data).

CONCLUSION:

For all WP3 Images Toolkit members, the principal feature for DCM API shall be the indexation and management of DICOM images in a non-closed structure, with many actions to manipulate as well database as real data. DCM API shall provide at least an abstract interface to allow interoperability between software based on it. More, The DCM API shall have an important documentation to help its integration and propagation.

2. DCM API SURVEY :

METHODOLOGY:

The survey is based on criteria, established from the above requirements. These criteria are the following:

- ✓ **Language:** what language to develop tool.
- ✓ **Multi-platform:** Windows, Mac Os, Linux. More OS available, more flexibility for developers.
- ✓ **Data-format :** DICOM is the more accepted standard for medical images but it shall be necessary to have other formats (as input and output)
- ✓ **Anonymize:** this function allows to anonymize patients information.

- ✓ **PACS connection:** possibility to have “push-send”, query actions with a PACS server.
- ✓ **DB management:** indexate and manage medical images on a database with some actions: add, remove, sort, filter, edit, save as…
- ✓ **DB creation**: the tool shall allow to create its own format database with medical images tags specified by users.
- ✓ **Third API connection:** the tool shall allow to be connected (by library or by integrated code) on a third application.
- ✓ **DICOM Reader:** gdcm v1, gdcm v2, dcmtk, equipment constructor specific. Possibility to switch between them.
- ✓ **Documentation**: tool shall have an exhaustive document on its architecture and its functionality. Not only two lines of FAQ

For this survey, only non-commercial and open-source tools were studied.
Each tool is tracked to find its source and documentation. When it is possible, an installation is done to perform a brief test (creation, data manipulation, exportation).
In this table (cf. below), we do not show the many tools left without evolution since many years or dead-links to download.

RESULTS:

The first result of this survey is that no tool can satisfy all criteria. **Osirix**, **Conquest**, **Tudor DICOM** or **dcm4che** respond to many fields but not all. The second information is that most tools are self-contained software. For developers, it will be not easy or possible to integrate some new functionality in their own application. Thirdly, many tools are DICOM compliant (in its version 3) but they do not allow new specificities as reverse-anonymyze. **gdcm v2** provides such functionality but no tool is based on this DICOM reader library. Moreover, for a given tool the DICOM reader (when information was available) is unique and thus cannot be switched.
Osirix would be the perfect candidate for our DCM API due to its popularity and its great robustness but the main limitation is its development only based on Macintosh OS.
In the same way, **Tudor DICOM** and **dcm4che** seem to be a good candidate but they are developed using Java. It would limit its propagation on medical images community. Most developments are indeed based on C++ language (as VTK-ITK library for example).
For **Conquest** based on C++, many features are missing: not a cross-platform (only Windows and Linux), no documentation and do not allow to modify database structure.
CTN provides a strong documentation (to install or to run) but last update was in 2003.
DICOM Java ImageIO SPI, DICOM Viewer, JDicom and **Loni Inspector** are developed using Java.
Exhalaion-B, fpImage, MEDWX, IRAD and **Sobox Image Viewer** do not allow to create a specific database and anonymize data.
More, from available documents, no tool provides an efficient dissociation between implementation and GUI, no specific layer for a DCM API is available.

TOOL (Software, library)	Language	Multi platform	DATA format	Anonymize	PACS connection	DB management	DB creation	DICOM reader	Documentation	Third API connection
ConQuest	C++		DICOM	x	x	x		?		x
CTN	C	x	DICOM	x	x	x		?	xx	x
dcm4che	Java	x	DICOM		x			dcmtk	x	x
DicmDirExplorer	C++		DICOM					?		
DICOM Java ImageIO SPI	Java	x	DICOM					dcmtk		x
DICOM Viewer	Java	x	DICOM					?		
Exhalation-B			DICOM					?		
fpimage			DICOM jpeg nema raw	x		x		?	x	x
JDicom	Java			x	x			?	x	x
LONI Inspector	Java	x	INTERFILE DICOM ANALYZE JPEG LONI	x				?		
MEDWX	Python	x	DICOM		x	x		?		x
Osirix	Objective C, C++		Analyze DICOM Jpeg	x	x	x		dcmtk	x	

IRad	Objective C		DICOM		x	x		?		
Sobox Image Viewer		x			x	x		?		
Tudor DICOM Tools	Java	x	DICOM	x	x	x		?		x

Table 1: advanced DCM API Survey

Conclusion:

Based on results of this survey, no tools fulfil the requirements as an advanced DCM API. To fill the technology gap, the next action is then to find the principal components needed to develop such a DCM API with WP3 criteria.

3. Search for components :

For an advanced DCM API, the fundamental needed components are: a database (or a structure to record data) and associated library, a DICOM reader, an interface library, a PACS connection library.

Database:
The main open-source databases are:

- ✓ SQL database with popular open-source library SQLite.
- ✓ Mysql
- ✓ Oracle from version 10g has a free version. But this version is very restricted and can't handle large data amount.

The advantage of SQLite is to implement a self-contained, serverless, zero-configuration, transactional SQL database engine. SQLite is the most widely deployed SQL database engine in the world for embedded applications.

Medical Images Reader:
A strong feature for a DCM API is to allow the switch between different DICOM readers or with users' readers (developed for a specific scanner, for example). The main DICOM readers are: gdcm v1, gdcm v2, dcmtk.
Gdcm v1 has a strong background but new features from DICOM v3 miss as reversible-anonymization implemented in gdcm v2.
For DICOM readers, each library seems to have their advantages.

Interface library:
It could be obvious than an advanced DCM API provides a GUI other than an abstract one: the implementation of DCM API GUI should be left to developers. On the other hand providing an interface (who could be removed or extended) would allow promoting DCM API not only to developers but also to other end-users for managing their data.
Open-source GUI library shall be cross-platform application:

- ✓ FLTK (Fast Light ToolKit) uses a more lightweight design and restricts itself to GUI functionality but does not allow the advanced C++ features.
- ✓ Qt
- ✓ Swing is a widget toolkit for Java only.
- ✓ WxWidget (previously WxWindows) is best described as a native mode toolkit as it provides a thin abstraction to a platform's native widgets, as opposed to emulating the display of widgets using graphic primitives. Calling a native widget on the target platform results in a more native looking interface

PACS connection:
Dcmtk provides this option.

CONCLUSION:

With no tool corresponding to an advanced (in sense of WP3 Imaging Tools subgroup criteria) DCM API, WP3 Imaging Tools subgroup proposes to develop a DCM API with the proposed components to fit all features. DCM API will have a pure abstraction layer to allow interoperability and also provide a normalised interface. The next action for CREATIS will be to provide technological requirements documentation with a global overview of API architecture to other WP3 Imaging Tools members for their agreements. Secondly, WP3 Image Toolkit group shall develop and test this DCM API based on these specifications.