ORTHANC – A LIGHTWEIGHT, RESTFUL DICOM SERVER FOR HEALTHCARE AND MEDICAL RESEARCH

S. Jodogne, C. Bernard, M. Devillers, E. Lenaerts and P. Coucke

Department of Medical Physics, University Hospital of Liège, 4000 Liège, Belgium

ABSTRACT

In this paper, the Orthanc open-source software is introduced. Orthanc is a lightweight, yet powerful standalone DICOM store for healthcare and medical research. Multiple instances of Orthanc can easily be deployed in the hospital network or even in the same computer, which simplifies the interconnection between the DICOM modalities and the data management of medical images. Orthanc is unique with respect to the fact that it provides a modern RESTful API: Orthanc can be driven from any computer language to automate clinical processes. Finally, Orthanc comes bundled with an embedded Web interface that allows the end-users to browse and interact with the content of the DICOM store.

***Index Terms*—** DICOM Store, Scripting, REST.

1. INTRODUCTION

Combining the information arising from different imaging modalities is an essential step for the high-quality diagnosis and treatment of various diseases in the medical routine [1]. Because hospitals are equipped with a steadily growing number of medical imaging devices (including Computed Tomography – CT, Positron Emission Tomography – PET, Magnetic Resonance Imaging – MRI, and Cone Beam Computed Tomography – CBCT), the data management of clinical images and the administration of the computer network of an hospital imply great challenges. The number of exchanged medical images as well as their volume are indeed exploding.

DICOM (Digital Imaging and Communications in Medicine) has grown over years as the *de facto* standard for the encoding, the management and the exchange of medical images [2]. Each DICOM file can be thought of as an envelope that embeds the medical image together with the meta-data associated with this image. The meta-data is essentially a hierarchy of key/value pairs that is in spirit similar to the more recent XML file format. 3D images are frequently split as a sequence of 2D images, each of them being wrapped as a separate DICOM instance. Besides being a file format, DICOM is also a *network protocol* that specifies how a given modality can send an image to another one, and how it can browse the content of remote modalities. For this reason, DICOM can be seen as one of the earliest examples of Web services.

The DICOM protocol was initially developed with a point-to-point approach in mind. In practice, this means that any change to the DICOM topology of the hospital (such as the adding, the removal or the renaming of one DICOM modality) impacts *all* the DICOM modalities that are connected to the updated modality. This makes the computer network of the hospital tedious to manage, since the various DICOM modalities are typically supported by separate private companies: It is difficult to synchronize even a single intervention. For this reason, the DICOM topology is often implemented as a star-shaped network: The PACS (Picture Archiving and Communication System) of the hospital serves as a central hub that dispatches the images between the various DICOM modalities. This approach has the disadvantage of putting a high load on the PACS, and making it critical to the clinical routine: Any slackening or downtime has an immediate impact on the clinical flows in the entire hospital.

This problem could be solved by introducing several *DICOM bridges* for the clinical routine that are independent of the PACS in the hospital network. If these DICOM bridges could be driven by external scripting, this would further smooth the clinical processes by automating well-identified tasks. The data management of clinical studies as well as the researchers active in the field of medical imaging would also benefit from fine-grained, low-cost DICOM stores: The medical images associated with the different databases could indeed be stored in isolated, task-centric DICOM stores.

The discussed clinical needs motivate the development of a software that can be used as a DICOM store for healthcare and medical research. This software should be lightweight, easy to deploy and to administer, and scriptable. The commercial offer for such a platform is essentially nonexistent at present. In this paper, the Orthanc open-source software that meets these real-world clinical needs is introduced.

2. ORTHANC

According to the previous discussion, Orthanc is designed to fulfill the following objectives:

1. To serve as a bridge between DICOM modalities in order to improve the processes in the hospital by facilitating the interconnection between proprietary software;

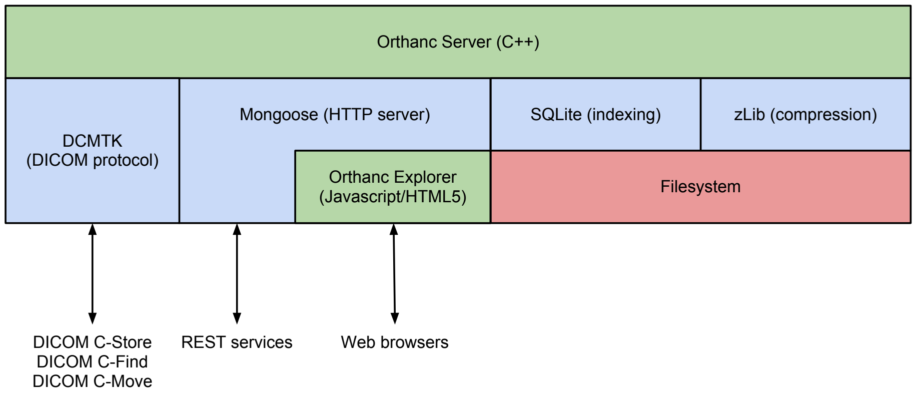


Fig. 1. The layered software architecture of Orthanc. The embedded third-party components are depicted in blue.

2. To facilitate the data management for clinical routine and medical research by providing an easy-to-install DICOM store;
3. To provide computer scientists with a simple, modern wrapping around the DICOM protocol so that DICOM network scripting is greatly facilitated;
4. To bring DICOM databases to the Computer Vision community so as to facilitate research about the automated analysis of medical images.

2.1. General Software Architecture

The aforementioned objectives put strong constraints on the software architecture of Orthanc. Most importantly, it must be easy to deploy multiple instances of Orthanc in the hospital network or even in the same computer. This constraint enables the data of multiple, but separate clinical studies to be stored on the same computer, hence reducing the hardware costs. It also enables the setup of bridges between DICOM modalities, hence breaking the typical star-shaped DICOM topology revolving around the PACS of the hospital.

In practice, this implies that Orthanc is designed to be *lightweight* and *standalone*. No complex database administration, nor special system configuration is required. There is no need to install any third-party component: Orthanc embeds its own database engine (SQLite) and stores the DICOM instances directly in a private arborescence of the filesystem. Once the binaries of Orthanc are downloaded, executing them immediately turns the computer into a DICOM store. This makes the deployment of Orthanc almost immediate: A single configuration file has just to be adapted to setup the DICOM AET (Application Entity Title) and the TCP port number.

It is important to notice that many computers in hospitals run under Microsoft Windows. To keep Orthanc as *fast* and *portable* as possible, Orthanc is written in C++ with a cross-platform approach. Thanks to this approach, Orthanc is currently compatible with Microsoft Windows and Linux.

Orthanc can thus run on any desktop computer. Because desktop computers do not necessarily have large amount of disk storage such as available in PACS servers, Orthanc can be configured to *compress* the incoming DICOM instances on-the-fly before writing them to the filesystem. This is useful, because the images are often stored as raw, uncompressed arrays in DICOM instances: The size of a typical DICOM instance can hopefully be divided by a factor 2 through lossless compression. This compression process is transparent to the user, as Orthanc automatically decompresses the file before sending it back to the external world. Furthermore, Orthanc implements *disk space recycling*: The oldest series of images can be automatically deleted when the available disk space drops below a threshold. A mechanism to protect important patients from recycling is also implemented.

Figure 1 outlines the architecture of Orthanc. The remaining components of this architecture will be discussed below.

2.2. DICOM Support

As far as the DICOM standard is considered, only few cross-platform, open-source software libraries support the DICOM network protocol. DMCTK (DICOM Toolkit) is one of such libraries. As written on the DCMTK homepage [3], "*DCMTK has been used [...] to provide central, vendor-independent image storage [...]. It is used by hospitals and companies all over the world for a wide variety of purposes ranging from being a tool for product testing to being a building block for research projects, prototypes and commercial products.*" Orthanc is built as a wrapper around the DCMTK library that hides the complexity of the DICOM protocol to its users.

As depicted on Figure 2, Orthanc acts as a Service Class Provider (SCP) for the Echo and the Store DICOM commands. This means that remote DICOM modalities can check whether they are properly configured to connect to Orthanc (by sending an Echo command) and store DICOM instances into Orthanc (by wrapping them together with a Store command).[1] Orthanc also acts as a DICOM Service Class User (SCU) for the Store, Find and Move DICOM

[1]Future versions of Orthanc will also act as C-Find and C-Move SCP.

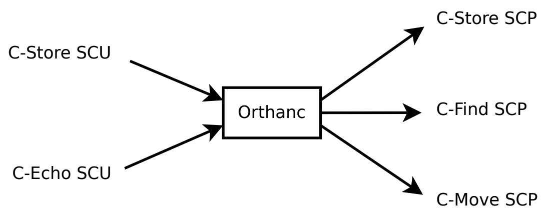


Fig. 2. DICOM commands supported by Orthanc.

commands. This means that Orthanc can send its DICOM instances to other modalities (C-Store), search the content of remote modalities (C-Find), and ask some remote modality to send some instances to another modality (C-Move). For the purpose of medical traceability, Orthanc logs in its database each time it sends a DICOM instance to remote modalities.

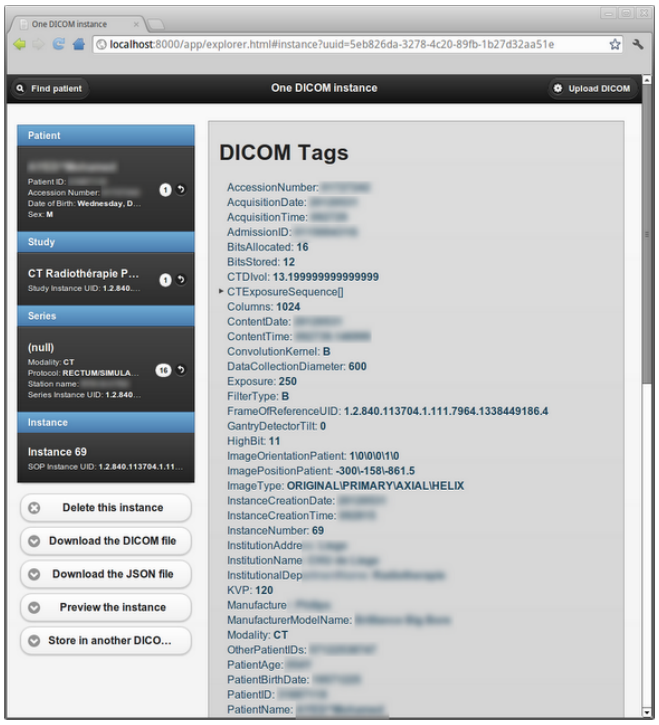
Importantly, Orthanc is able to convert DICOM files to modern file formats that are well-known to computer scientists on-the-fly. The 2D images embedded in the DICOM instances can be downloaded as standard PNG files. The meta-data associated with the instances can be downloaded as JSON files. Orthanc does the required conversions transparently to the user. We argue that this feature will bring medical images closer to the file formats the Computer Vision community commonly deals with, hence stimulating the research on the automated analysis of medical images.

2.3. Orthanc Explorer

To fulfill the objective of making the data management of DICOM instances easier, it is necessary for the user to browse and interact with the content of Orthanc. To this end, Orthanc proposes a GUI (Graphical User Interface) that is known as *Orthanc Explorer*. This GUI does not require the installation of any application on the client computer: Orthanc Explorer is a *Web Application* that runs inside Orthanc and that can be opened from anywhere in the hospital by a Web browser such as Mozilla Firefox, Google Chrome or Apple Safari. Security is provided by password protection and SSL encryption.

Orthanc Explorer presents the content of its hosting Orthanc instance using the Patient/Study/Series/Instance DICOM model: The list of patients is first displayed, then one can successively open the studies of the patient, the series of the studies and finally the instances of the series. Figure 3 shows how an instance is displayed inside Orthanc Explorer.

Technically, this means that Orthanc is not only a DICOM server, but also a HTTP server that listens on another TCP port than the DICOM server. To this end, Orthanc embeds the open-source Mongoose HTTP server. Orthanc Explorer is a lightweight combination of HTML5 and Javascript code that is developed within the jQuery Mobile framework and that is served through Mongoose. Note that to preview a DICOM file inside the Web browser, Orthanc Explorer exploits the fact that the Orthanc Server can dynamically convert DI-



localhost:8000/app/explorer.html#instance?uuid=5eb826da-3278-4c20-89fb-1b27d32aa51e

One DICOM instance

[redacted]
Patient ID: [redacted]
Accession Number: [redacted]
Date of Birth: Wednesday, D...
Sex: M

CT Radiothérapie P...
Study Instance UID: 1.2.840....

(null)
Modality: CT
Protocol: RECTUM/SIMULA...
Station name: [redacted]
Series Instance UID: 1.2.840...

Instance 69
SOP Instance UID: 1.2.840.113704.1.11...

DICOM Tags

AccessionNumber: [redacted]
AcquisitionDate: [redacted]
AcquisitionTime: [redacted]
AdmissionID: [redacted]
BitsAllocated: 16
BitsStored: 12
CTDIvol: 13.199999999999999
▸ CTExposureSequence[]
Columns: 1024
ContentDate: [redacted]
ContentTime: [redacted]
ConvolutionKernel: B
DataCollectionDiameter: 600
Exposure: 250
FilterType: B
FrameOfReferenceUID: 1.2.840.113704.1.111.7964.1338449186.4
GantryDetectorTilt: 0
HighBit: 11
ImageOrientationPatient: 1\0\0\0\1\0
ImagePositionPatient: -300\-158\-861.5
ImageType: ORIGINAL\PRIMARY\AXIAL\HELIX
InstanceCreationDate: [redacted]
InstanceCreationTime: [redacted]
InstanceNumber: 69
InstitutionAddre[redacted]
InstitutionName [redacted]
InstitutionalDep[redacted]
KVP: 120
Manufacture [redacted]
ManufacturerModelName: [redacted]
Modality: CT
OtherPatientIDs: [redacted]
PatientAge: [redacted]
PatientBirthDate: [redacted]
PatientID: [redacted]
PatientName: [redacted]

Fig. 3. Displaying some DICOM instance. The right column displays the hierarchy of the DICOM tags. The left column displays the Patient/Study/Series/Instance organization. It is possible to delete the instance, download the raw DICOM file, download a JSON file containing the DICOM meta-data, preview a PNG image, or send the instance to another modality.

COM instances into PNG files[2].

2.4. RESTful Application Programming Interface

Orthanc Explorer provides Orthanc users with an interface to interact with the content of the DICOM store. Similarly, Orthanc provides computer scientists and network administrators with an *Application Programming Interface* (API) to automate the interactions with the stored DICOM instances.

The Orthanc API can be used to browse the instances using the Patient/Study/Series/Instance DICOM model, and to download them as DICOM, JSON or PNG files. Once an instance has been located, it is possible to instruct Orthanc to send it to another modality: Orthanc automatically makes the corresponding C-Store DICOM command, which frees the programmer from learning the API of some low-level DICOM library. It is as well possible to ask Orthanc to make a DICOM query to search the content of a remote DICOM modality. As a consequence, Orthanc has the further advantage of cutting down the learning curve that is required for

[2]PNG images can indeed be natively displayed by any state-of-the-art Web browser, which contrasts with DICOM images.

computer scientists to master the DICOM protocol.

Technically speaking, Orthanc provides a modern *RESTful API* (REpresentational State Transfer) [4]. This API is a well-documented set of HTTP paths against which it is possible to make HTTP requests with any client tool, such as the standard `curl` command-line tool. Similarly to Orthanc Explorer, the RESTful API is served through the embedded Mongoose HTTP server. It is thus really easy for shell scripts or computer languages (such as Python, Java or C#) to externally drive Orthanc, hence automating DICOM processes (such as searching the instances, moving data around modalities, deleting series,...). In other words, Orthanc was designed with the *Service Oriented Architecture* (SOA) approach in mind. A detailed discussion of the Orthanc API is out of the scope of this paper. The interested readers are kindly invited to refer to the Orthanc homepage that contains practical examples of Orthanc scripting[3].

3. CASE STUDIES

The CHU (*Centre Hospitalier Universitaire*) of Liège is one of the major hospitals in Belgium. It is composed of 45 medical departments that are spread over 5 sites. Its PACS system is connected to about 300 DICOM modalities, 180 of them producing medical images. The CHU of Liège is thus clearly subject to the scale problems that were outlined in the Introduction. Orthanc is currently used at the CHU of Liège to improve two real-world clinical processes.

Firstly, our Nuclear Medicine (NM) department daily sends PET-Scans together with tumor contouring to the TPS (Treatment Planning System) of our Radiotherapy (RT) department through the DICOM protocol. Unfortunately, it is necessary to delete each night the files that were sent to the TPS to prevent the system from freezing. As a consequence, the RT medical physicists frequently have to phone the NM engineers to ask them to re-send the files that have been deleted, which leads to a waste of time for both departments. For this reason, Orthanc has been deployed as a buffer between the two departments: The NM department sends its files to Orthanc, where the RT physicists can immediately find and restore them back if needed through Orthanc Explorer, from any computer and independently of the PACS.

Secondly, another instance of Orthanc is configured to gather the medical images that are produced during the treatments in the RT department: Orthanc centralizes the so-called *in-room images* (CBCT and radiograph) that are produced by the RT treatment machines for positioning the patients together with the DRR (Digitally Reconstructed Radiograph) images that are produced by the TPS system as a reference for this positioning. This opens the path to the automated assessment of the quality of the patient positioning and to the clinical research about adaptive radiotherapy.

[3]`https://code.google.com/p/orthanc/`

4. CONCLUSION

This paper introduces the Orthanc open-source software. Orthanc is a lightweight, standalone, scriptable DICOM store. It is designed to smooth clinical processes, to ease the data management of medical images, and to bring the DICOM standard closer to the Computer Vision community thanks to its support of well-known file formats (JSON, PNG) and network protocols (RESTful API). Orthanc abstracts the complexity of the DICOM format and of the DICOM protocol: As a consequence, its target audience consists of the network administrators of an hospital, as well as of the computer scientists that develop software for the automated analysis of medical images. Contrarily to software such as OMERO or Bisque for the data management and the remote visualization of biological images, Orthanc is primarily conceived as a central, robust building block for the automation of the various medical imaging tasks that are specific to each hospital.

We have shown how Orthanc has been used at the pilot site of the CHU of Liège (Belgium) to solve real-world problems inside the clinical flows by *decoupling* the DICOM modalities, which improves the interoperability between proprietary software. Orthanc can be freely downloaded and used under the terms of the GPLv3 license[3]. Orthanc is an integral part of the Debian Med project[4], which makes the installation of Orthanc on Debian-based servers immediate.

Future work on Orthanc will focus on the anonymization of the stored instances, on the development of a FUSE (Filesystem in Userspace) Linux module to present the content of Orthanc as a standard filesystem, and on the implementation of a Web-based DICOM viewer that meets the clinical requirements of radiotherapy. Besides these software developments, Orthanc is also planned to be integrated in more clinical flows at the CHU of Liège thanks to its scripting capabilities, such as in a platform for exporting clinical studies to external companies, or as in a platform for the automated analysis of cardiac MRI.

5. REFERENCES

[1] S.R. Cherry, "Multimodality in vivo imaging systems: Twice the power or double the trouble?," *Annu Rev Biomed Eng*, vol. 8, pp. 35–62, 2006.

[2] National Electrical Manufacturers Association, "Digital imaging and communications in medicine (DICOM)," `http://medical.nema.org/standard.html`, 2011.

[3] OFFIS e.V., R&D Division Health, "DCMTK," `http://dicom.offis.de/dcmtk.php.en`, 1994–2011.

[4] R.T. Fielding, *Architectural Styles and the Design of Network-based Software Architectures*, Ph.D. thesis, University of California, Irvine, 2000.

[4]`http://www.debian.org/devel/debian-med/`