

# AN APPROACH TO THE STORAGE OF DICOM FILES FOR GRID-ENABLED MEDICAL IMAGING DATABASES

David Power, Eugenia Politou, Mark Slaymaker, Steve Harris, and Andrew Simpson

Oxford University Computing Laboratory

{David.Power, Eugenia.Politou, Mark.Slaymaker, Steve.Harris, Andrew.Simpson}@comlab.ox.ac.uk

**Abstract:** The Standard for Digital Imaging and Communications in Medicine (DICOM) specifies a non-proprietary digital imaging format, file structure and data interchange protocols for the transfer of biomedical images and non-image data related to such images - it is a specification of the components that are required in order to achieve inter-operability between biomedical imaging computer systems. We describe how a Grid-enabled medical imaging database (eDiaMoND) employs a relational approach to the storage of DICOM files. Although the work described has been carried out within the context of a specific mammography-related project, the underlying principles are applicable to other medical imaging systems dealing either with other modalities or with other diseases.

## Introduction

The main aim of the eDiaMoND project is to develop a prototype for a national database of digital mammograms to support the United Kingdom's breast imaging infrastructure. The project also intends to develop a number of applications to take advantage of that infrastructure. One of the secondary aims of the eDiaMoND project is to deliver generic solutions to the problems addressed during the development and deployment processes. This paper is presented in that spirit: we feel that the approach taken to database design for the eDiaMoND project will be of interest not only to other mammography-related projects, but also to a variety of Grid-based applications in which DICOM is utilised.

## DICOM: a brief overview

The Standard for Digital Imaging and Communications in Medicine (DICOM) is a widely used standard for the storage and transfer of medical images. What sets DICOM apart from other image formats is that DICOM files contain both the image data and also image-related data. Image-related data can include, for example, data pertaining to how and why the image was taken, annotations, and patient information. This ability to capture non-image data is taken to its extreme when DICOM Structured Report Documents are deployed. In such circumstances, no image is included in the file: it consists entirely of non-image data.

In the DICOM view of the entities related to medical images, a patient can be the subject of any number of studies. In the context of eDiaMoND, a study could represent the screening, then subsequent assessment of a patient; in the course of a study one or more series of images may be taken. For example, in breast screening a single series of images would be taken using an X-ray

machine. If further assessment were to be deemed necessary then a series of images may be taken using a different X-ray machine or possibly Ultrasound or Magnetic Resonance techniques. Typically, for each session (or for the use of a new piece of equipment), a new series is created. It is also possible to have a series of Structured Report (SR) Document files. SR Document files can refer to DICOM files (or simply the image part of such a file) belonging to other series without actually being part of that series; such referential power is available only to SR Document files.

Every DICOM file contains information pertaining to the patient, study, and series. It is common to store one image per DICOM file, so the patient, study and series data will be repeated in every file. This allows each image to be viewed in isolation without losing context, but involves storing the same data in many files and introduces the possibility of conflicting data in files that relate to the same patient.

Via the DICOM standard it is possible to exchange information relating to a wide variety of different image modalities. Within a single file, or as part of a single transfer, it is important that all parties involved understand what type of information is being exchanged. To facilitate this understanding, a reference is made to a Service Object Pair (SOP), which contains an Information Object Definition (IOD). It is the IOD that defines what type of object is being transferred.

Each Information Object Definition (IOD) contains a number of modules. Each module is related to an entity. Modules are not exclusive to IODs, and it may be the case that they are related to different entities in different IODs.

Each module has a usage, represented by one of three identifiers: if a module has usage M, it must be included;

if it has usage C, its inclusion will depend on an explicit condition; if it has usage U, it is optional.

Each module is made up of a number of attributes. Each attribute has a name, a tag and a type. The tag is a pair of 16 bit numbers used to identify the attribute. The first number signifies the Tag Group that the attribute belongs to; the second number identifies the Tag Element. Together, the pair represents the specific attribute. Even for mandatory modules it is not necessary to use all the attributes. The type of the attribute determines if it must be present and whether it must contain any data: type 1 attributes must be present and contain data; type 2 attributes must be present but may contain no data; type 3 attributes are optional. Type 1C and 2C attributes also exist: such attributes are present only if certain conditions are met. It is possible that attributes may be present in more than one module; in this case the attribute may have more than one type. In these cases the most restrictive type is used.

In Part 6 of the DICOM standard, each tag is given a value representation (VR) and a value multiplicity (VM). The value representation describes the exact binary representation of the attribute within a DICOM file. The value multiplicity gives the number of such attributes.

Having a value multiplicity of more than one allows simple lists of attributes; for more complicated repeated patterns of attributes, DICOM defines sequences. A particular attribute may be present in many different sequences and many times in a particular sequence. It is also possible for sequences to contain other sequences. While this might tend to suggest that an object-oriented approach to the design of the database would be appropriate, the drawbacks associated with object-oriented databases, such as, for example, the fact that query optimisation compromises encapsulation means that an object-relational approach is more appropriate. The IBM DB2 database - which underpins eDiaMoND - offers such capabilities.

### Data requirements

In this section we describe some of the types of data that we are concerned with.

The IODs currently supported by the eDiaMoND system are those pertaining to Secondary Capture Images, Digital Mammography X-Ray Images, and Mammography CAD Structured Report Documents. It is fully intended that future versions of the database will support other modalities, such as Magnetic Resonance Images, Ultrasound Multi-frame Images, and Positron Emission Tomography Images. The eDiaMoND database is designed in such a way that the addition of further

modalities will not pose any implementation issues (at the database level, at least).

Standard sets of patient data forms are gathered at the same time as image acquisition. Screening forms are stored as DICOM Structured Report Documents, which contain standard BiRADS classifications.

There are, of course, other types of information to be captured by the database. Data pertaining to audit trail and access control capabilities are of particular relevance, given the nature of eDiaMoND. In particular, there is a clear need to have a sufficiently flexible access control mechanism to allow individual hospitals or departments to enforce their own local policies. These issues are, however, outside the scope of this paper.

All updates to the database take the form of DICOM files inserted into the database. The non-image components of these files are stored in the database, together with an identifier pertaining to the user that inserted the DICOM file, as well as additional information, such as, for example, the time that the file was inserted. The database also keeps a record of all queries on the data.

### Design

Users are not permitted to query the database directly, instead queries are sent in a pre-determined format as XML documents to the Query Service.

The Query Service, as illustrated in Figure 1, communicates with the database through an OGSA-DAI Grid Data Service. The Query Service can determine the access rights of an individual user by querying the database as a privileged user. Having determined the appropriate access permissions, the query is either rejected or translated into appropriate SQL. The Query Service also ensures that the access attempts are logged correctly by inserting the user details into the logs.

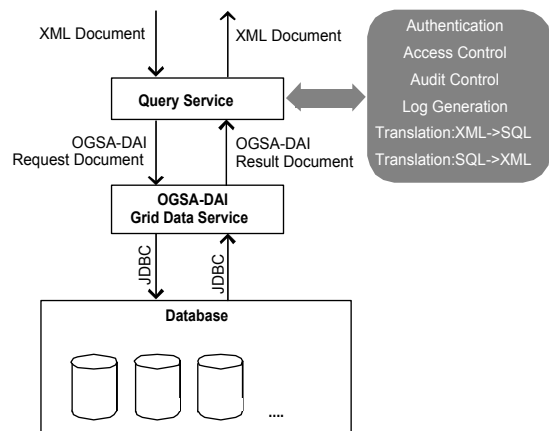


Figure 1. Grid Query Service Architecture

The JDBC connection is made to a single logical database - although, in reality - this will be either a federation of different physical databases or a group of replicated databases. The design supports the querying of both.

While it is desirable to be able to handle all types of DICOM files, it is not practicable to create a normalised database that could fully represent an arbitrary file. This problem can be easily understood if the system was required to deal with a new version of DICOM, which perhaps introduced new data fields or new IODs. In effect, we are designing for forwards-compatibility. In addition, employing an unnormalised structure - with the potential performance benefits that may result - is inappropriate for an application that requires guaranteed consistency of data; data integrity is essential for an application such as eDiaMoND. Despite the fact that it is undoubtedly the case that performance is important to us, delivering correct data in a longer time is infinitely more preferable than delivering incorrect data quickly.

For these reasons we divide the eDiaMoND database logically into two parts, as illustrated in Figure 2.

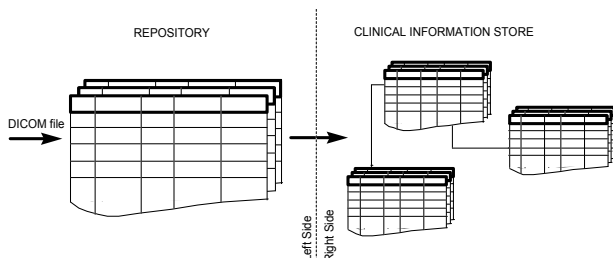


Figure 2. Database Architecture

In one logical part of the database - which we term *the repository* - the data is stored in a relatively unstructured fashion. In the other part - which we term *the clinical information store* - data from specific IODs can be stored in a high-order relational database (at least 3NF). In this way, when a DICOM file is inserted, it is parsed, with all non-image data being stored in the repository. Then, automatically, via automated constraint-enforcing procedures (triggers), the necessary data is inserted into the clinical information store. It is noted that in the repository, all the tags of the DICOM file are stored, including optional and private tags. In contrast, the clinical information store holds *only* the data that is currently useful for eDiaMoND applications. If any tags were to be deemed relevant in future, it would be trivial to create additional relevant triggers.

In this architecture there are some rules that have to be obeyed. First, the repository allows only INSERTs, but not UPDATEs or DELETEs; this ensures that there is no potential for data loss. Second, the clinical information

store allows only INSERTs and UPDATEs as a result of INSERTs to the repository; this ensures that the data in the clinical information store is consistent with that in the repository.

One of the most critical issues when developing a medical database is the provision of appropriate mechanisms for allowing updates and tracking changes. This importance is derived from the legal and ethical requirements to record all updates of patient and screening data. In the eDiaMoND database, the deletion of previously captured DICOM files is forbidden. The principal reasons for this are the necessity for keeping a history of previous data and cooperating with the existing health and legal regulations.

When an update of information is needed, e.g., a change of name or a change of address, this will take place as a two-phase operation. The first phase involves the insertion of a new DICOM file that contains the updated - or corrected - data. The second phase - which is triggered by the successful completion of the first phase - involves a copy and an update.

This second phase of the update mechanism is described by the following example, which is depicted in Figure 3. We consider a patient - Helen Jones - who has a unique ID: 12. Associated with this record is a related record - referenced by a foreign key called *Audit Index* - in another table, called *Audit Table*, which records the "who", the "when", and the "where" corresponding to the creation of a record in the *Patient Table*. If the patient were to get married, with a consequent change of surname, we would, of course, wish to update the patient's name, with all other pertinent information remaining the same. To achieve this, the patient record in *Patient Table* has to be duplicated, resulting in a new record, with a primary key of 13. The information contained in the fields of the old record, having a primary key of 12, will remain the same, except the one that has to change, which in our case is the *Name* field, and also the field called *Audit Index*. The latter has to reference a new record in *Audit Table*, which will contain all the necessary information pertaining to the conditions under which the alteration has taken place. This new record is the one having primary key 1043 in the *Audit Table*. In the above example we have used one *Audit Table* for a given table in the database. However, one *Audit Table* can be used to track the changes of many tables. The concept remains the same: the only modification is the addition of another field in the *Audit Table* containing a reference to the table that has been changed.

BEFORE UPDATE					AFTER UPDATE				
<b>Patient Table</b>					<b>Patient Table</b>				
PK	Name	Parent Index	Audit Index		PK	Name	Parent Index	Audit Index	
12	Jones Helen	NULL	879		12	Lloyd Helen	NULL	1043	
					13	Jones Helen	12	879	
<b>Audit Table</b>					<b>Audit Table</b>				
PK1	Res.Person	Date/Time	Reason	Place	PK1	Res.Person	Date/Time	Reason	Place
879	Dr. Miller	2002-27- 12:02:39445	Insertion	John Radcliffe Hospital	879	Dr. Miller	2002-27- 12:02:39445	Insertion	John Radcliffe Hospital
					1043	Dr. Baker	2002-12-10 12:02:39346	Marriage	Churchill Hospital

Figure 3. An Update

The schema for the clinical information store has been designed in a way that allows an arbitrary set of DICOM files to be stored in the database, while ensuring that the relationships between Patients, Studies, Series and Equipment is maintained.

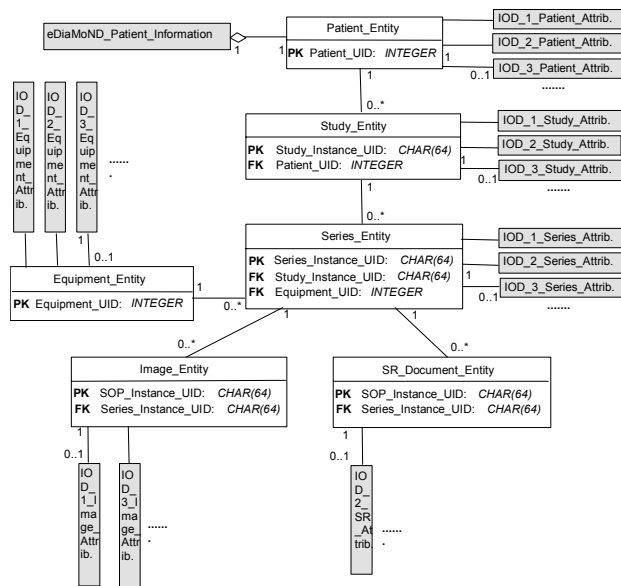


Figure 4. DICOM Data Schema

The schema of Figure 4 is logically divided into two parts: there is the common spine of DICOM entities and additional tables relating to specific IODs. In the diagram, three IODs are represented: IOD\_1 and IOD\_3 represent two different types of DICOM *image* IODs, while IOD\_2 represents a *Structured Report* IOD. While all three IODs have attributes related to Patient, Study, Series and Equipment, only the Image IODs have attributes related to the Image entity and only the Structured Report IOD has attributes related to the Document entity.

As mentioned previously, the modules of an IOD are all related to specific entities. Those modules contain attributes that may be present in more than one module, however the structure of DICOM is such that the same attribute will not be present in more than one entity. For this reason it makes more sense to group the attributes of an IOD by entity and not by module. Although not shown in the diagram, the tables relating to each IOD are linked to the entity tables by a shared primary key, for example, the *IOD\_Image\_Attrib* table has a primary key of *SOP\_Instance\_UID*, which has a foreign key constraint to the *SOP\_Instance\_UID* primary key of the *Image\_Entity* table.

The *Patient Entity* table is also linked to the *eDiaMoND\_Patient\_Information* table; this contains the additional patient data that was described previously, which is essential for epidemiology and integration with existing patient record systems.

The entity spine contains a bare minimum of data allowing efficient querying when using Unique Identifiers (UIDs). The *Patient\_UID* and *Equipment\_UID* are not part of the DICOM standard. The *Patient\_UID* is represented as the *Patient\_ID* attribute in the DICOM files, as an additional requirement on the users of the eDiaMoND system it is essential that all *Patient\_ID*s are unique. The *Equipment\_UID* is stored in the Device Serial Number attribute, and is also unique.

### Discussion

We have described an approach to the storage of DICOM files that has been taken in the development of a Grid-enabled medical image database. As DICOM is a widely recognised standard for the transfer and storage of medical images, the applicability of the approach is by no means restricted to the area of mammography. Furthermore, even though the eDiaMoND database is based on IBM's DB2 and Content Manager technologies, there is nothing vendor-specific about the *principles* that have been employed.

The database has been designed with the storage of any class of DICOM file in mind: although we have restricted ourselves only to IODs of relevance to eDiaMoND, the database has been designed so that it can easily be adapted to handle other types of file or other IODs. This design decision was taken with the view of future-proofing the system: new epidemiological studies, new equipment and new reporting forms all have the potential to introduce new types of data into the database. The logical divide of the database into the clinical information store and repository means that one part can easily be restructured without there being the potential for major disruption to the system as a whole.