DICOM Suported Sofware Configuration by XML Files

Bioing. Fabian M. Lucena G.1,2, Mgr. Andrés E. Valdez D. 1,2, Mgr. María E. Gomez2, Dr. Oscar H. Nasisi3

1Facultad de Ciencias Médicas, Universidad Católica de Cuyo,
2Gabinete de Tecnología Médica, Facultad de Ingeniería;
3Instituto de Automática, Facultad de Ingeniería, Universidad Nacional de San Juan

flucena@gateme.unsj.edu.ar, anvaldez@unsj.edu.ar, megomez@gateme.unsj.edu,ar, onasisi@unsj.edu.ar

Abstract. A method for the configuration of informatics systems that provide support to DICOM standards using XML files is proposed. The difference with other proposals is base on that this system does not code the information of a DICOM objects file, but codes the standard itself in an XML file. The development itself is the format for the XML files mentioned, in order that they can support what DICOM normalizes for multiple languages. In this way, the same configuration file (or files) can be use in different systems. Jointly the XML configuration file generated, we wrote also a set of CSS and XSL files. So the same file can be visualized in a standard browser, as a query system of DICOM standard, emerging use, that did not was a main objective but brings a great utility and versatility. We exposed also some uses examples of the configuration file mainly in relation with the load of DICOM information objects. Finally, at the conclusions we show the utility that the system has already provided when the edition of DICOM standard changes from 2006 to 2007.

1. Introduction
The informatics science improve permanently help to the development of imaging diagnosis [1], providing different formats for the processing, storage and transmission of images, depending on manufacturer’s criterion. In an effort pointing to the standardization, the American College of Radiology (ACR) and the National Electrical Manufacturers Association (NEMA) formed a committee together (ACR-NEMA) in 1983 publishing the first version of the standard in 1985. In 1992 the committee obtained the approbation from the Radiological Society of North America (RSNA) and by September of 1993 version 3 was published. Nowadays, current version of the standard is number 3.0, known as ACR-NEMA DICOM (Digital Imaging and Communication in Medicine) or just DICOM [2]. Each year a new edition of the standard is obtained, available for being downloaded via Internet from the site: http://medical.nema.org/dicom/, and though changes could seem to be minor, due to them, an imaging software with support for DICOM can become obsolete in a few time.

It appears in consequence the need to provide an auto-configuration system so the support that informatics systems give to DICOM standard would be versatile, autonomous, and transparent to the software itself. This can be reach easily by the use of a configuration file that could go jointly with the
software and can configure it whenever it is necessary. For being this an ideal solution, it moves software maintaining, in relation to its support for DICOM standard, to an alert and continuous maintenance of the configuration support file.

Present work proposes a methodology for providing a uniform format for the information of DICOM standard so any informatics system can interpret and employ it in its auto-configuration. In particular the configuration file was designed for the medical imaging processing system ProcIma [3], see figure 1. The chosen form for the configuration file corresponds to XML standard [4]. Previous work put together DICOM with XML too [5] [6], but always pointing to the way of coding the information contained in a DICOM objects file to XML format, and never with the objective of translating the standard towards XML, in order to be interpreted by software systems.

![Figure 1. Relationship between the different system blocks](image)

2. Design and Development

The design effort is oriented to the standardization of the text of the standard in order to be interpreted by a software system and auto-configure itself. Among the explored alternatives, that were: binary files, text files and free format data bases, we decided to give the XML format to the configuration file.

The eXtensible Markup Language (XML) is a meta-language that permits to structure an information document as plain text. The meaning of each data is stated by means of inserted marks on the same text. Although the text can be read, XML was not designed for being human readable, but to code information that will be processed by an adequate system. XML is a language that comes from SGML family (Standard Generalized Markup Language) as well as HTML (HiperText Markup Language) used for web sites. The main difference with it, is that HTML inserts visualization format in the information file, while XML just employs markers for information format.

The minimal structure of a XML system comprises a unique XML file that contains the information and usually has an “.xml” extension. It is possible to incorporate to this file another one that states the specific format for the information formatting. This file can be a Document Type Definition ”.dtd” or an “.xml” Scheme. These two files (the data one and the format one) provide the necessary information so a parser can interpret this information and get at the output, tables or relational data. Any current Internet browser can format the information for a basic visualization.

The XML file design permits Information Objects Definitions (IOD), according to current needs of ProcIma system from PICT 2003 Nº 430 project. Only four IODs have been defined, corresponding to: Computed Tomography, Magnetic Resonance, Nuclear Medicine and Positrons Tomography. The rest of IODs defined in the Standard can be incorporated without causing any kind of negative influence over the software system functioning.
The designed XML file has a main node called **DICOM**. This node has two attributes: **Version** and **Edition**, with these values at the present: **Version** = "3" and **Edition** = "2007". Inside this node other five are defined that constitute the following five parts of the document:

1. ValueRepresentations.
2. DataElements.
3. Macros.
4. InformationEntities.
5. InformationObjectDefinitions.

In order to avoid conflicts between node and attributes names four namespaces have been defined with the following URIs (Uniform Resource Identifier):

1. `http://gateme.unsj.edu.ar/procima/DICOM/ValueRepresentation`
   For child nodes of ValueRepresentations,
2. `http://gateme.unsj.edu.ar/procima/DICOM/DataElement`
   For child nodes of DataElements,
3. `http://gateme.unsj.edu.ar/procima/DICOM/InformationEntity`
   For child nodes of Macros and InformationEntities
4. `http://gateme.unsj.edu.ar/procima/DICOM/InformationObjectDefinition`
   For child nodes of InformationObjectDefinitions,

```xml
<DICOM Version="3" Edition="2007">
  <ValueRepresentations/>
  <DataElements/>
  <Macros/>
  <InformationEntities/>
  <InformationObjectDefinitions/>
</DICOM>
```

**Figure 2.** Main node, names spaces and the five sections. Sections have been folded for a better visualization.

First section **ValueRepresentations** contained all value representations that DICOM standard supports. Each value representation is a **ValueRepresentation** node which attributes **VR** and **Length** get their values from the short form of value and length representation according to the description of table 6.2-1 at part 5 of DICOM standard. Each of these nodes contained at least one **Name** node and can contain one or more **Definition**, **Example** or **Note** nodes. These last three contain also one or more **Text** nodes. **Name** and **Text** nodes must contain the **Language** attribute, and optionally the **Country** attribute to indicate the language and the place where this name and text are applied. The values that can get **Language** and **Country** attributes are the ones defined in ISO 639-1 [7] and ISO 3166-1 alfa-2 [8] standards. It can be also added a **Repertoire** node to the **ValueRepresentation** node to specify the corresponding repertoire. This is independent from the language.
Figure 3. ValueRepresentation Section: two ValueRepresentation entries are shown, for a better view Text nodes have been folded.

The DataElements section contains the complete data dictionary according to the definition at part 6 in DICOM standard. For each data element a DataElement entry is specified with Tag, VR and VM attributes, which values are the ones defined in the both columns of the declared list at the standard. Each DataElement node contains Name, Description, Example or Note nodes with the explained restrictions in the previous paragraph. EnumeratedValues or DefinedTerms node can also be added, but just one of the two. DataElement and ValueRepresentation nodes are interrelated by means of the value equality of their VR attributes.

Figure 4. Part of DataElements section: three DataElements entries are shown, for a better view Text nodes have been folded.

InformationEntities section contains entities entries in their Entity nodes with a Name attribute which vale is the entity name according to the figure in part 3 of DICOM standard. Each entity contains the said set of nodes, Name, Description, Example or Note (with their mentioned restrictions)
and Module nodes with the Name attribute according to part 3 of DICOM standard. Module nodes also contain the said nodes set Name, Description, Example or Note. Finally, all DICOM attributes that contain that module are numbered by means of Attribute node entries which Tag attribute, relates it with the corresponding DataElement node. The Type attribute maintains the kind of attribute. Sequence and Condition attributes are also accepted. Attribute nodes, for specialization, can contain (not a must) Name, Description, Example, North, EnumeratedValues and DefinedTerms nodes. Module nodes also accept Include nodes that include macros, these nodes at the same time need as a mandatory attribute Name that specifies the name of the macro and can contain BaselineContextID, DefinedContextID and Sequence attributes for specialization.

Figure 5. Part of the InformationEntities section: two Entity entries are shown, the first one has been folded for a better visualization.

Macros section contains Macro entries with a unique Name attribute. Their content define in the same way a Module node and serve for getting expanded at Include nodes when the Name attribute of both nodes are the same.

InformationObjectDefinitions section contains the entries of the information objects at Object nodes with the Name attribute according to part 3 of the DICOM standard. Each of these nodes contains the set of nodes Name, Description, Example or Note. It also contains Entity entries with their corresponding Name attribute that relates them with the corresponding entity of the InformationEntities section. Each Entity node has a Module node entry to complete the link to the corresponding module of the entity by means of the Name attribute. Module nodes also have the Usage attribute with a value given by the usage of this module for this object in the DICOM standard.

Figure 6. Part of the InformationObjectDefinitions section: two Object entries, with two Entity entries and their respective modules are shown.

The five mentioned sections maintain relations between them. The InformationObjectDefinitions section employs Entities entities and Modules modules defined in InformationEntities section. In turn modules’ attributes are data elements defined in DataElements section and each data element has a
value representation defined in `ValueRepresentations` section. `Macros` section contain Macros entries that expand into `Include` nodes, inside each `Module` or `Macro` node, when `Name` attribute are the same.

Figure 7 shows an example of the existing relationships between four of the five sections (Macro section was omitted for a better view). It shows how for the definition objects CT, Patient module’s attributes are searched, contained at Patient entity. In turn for the attribute (0010,0010) –Patient’s Name–, corresponding data is searched in `DataElements` section, to find finally its value representation in `ValueRepresentation` section.

**Figure 7.** Existing relationship between different sections.

3. Experimentation

Preliminary essays were done before employing C++ code. The essay probes the well-formed and validity characteristics of the file, for this purpose an XSL document was generated. XSL is a programming language created specifically to format the information contained in XML documents, and generate, from them, a human readable form. The output can be a text document, PDF or a XHTML. We chose the last one because it is possible to observe it in a current web browser as Internet Explorer or Mozilla Firefox.

**Figure 8.** Screen snapshot during a query to DICOM standard with Firefox2.0.0.3 browser, under Windows.
Jointly with the XSL file, we use a sheet of CSS style that defines the visualization style (letter and color kinds). XSL is a real programming language, and due to that deserves a description that lies out of the scope of this article. Initial experimentation and integrity test of the configuration XML file concluded in a query system in hypertext for the DICOM standard. The query is done like a normal web site, visualized in the configured language and place. Figure 8 shows a part of this visualization.

The real power of this system relies in the XPath syntax [9]. With some guidelines it is possible to write an XPath that returns exactly what one is looking for. XPath syntax is similar to the Unix route (or the URL of Internet) with some differences. For example if you want to find the value representation data for the DICOM tag (0010,0010), as it is shown in figure 7, an XPath as it follows is written:

```
/DICOM/DE:DataElements/DE:DataElement[@Tag='(0010,0010)']/@VR
```

Returns PN value, with a new XPath the data corresponding to the value representation is queried.

```
/DICOM/VR:ValueRepresentations/VR:ValueRepresentation[@VR='PN']
```

With this known data now is possible to load the element. To know in which entity or module must be put this element the following XPath can be used:

```
/DICOM/IE:InformationEntities/IE:Entity/IE:Module/IE:Attribute[@Tag='(0010,0010)']/../@Name
```

It will present a list with the names of all the modules that contain the attribute (0010,0010) according to DICOM standard. To recover the name of the module in Spanish the following XPath should be used:

```
/DICOM/IE:InformationEntities/IE:Entity/IE:Module/
IE:Attribute[@Tag='(0010,0010)']/../IE:Name[@Languaje='ES']
```

Figure 8 shows a C++ code fragment used by an XML syntax analyzer. An object named DICOM is shown, in which was loaded the configuration file. Over it an XPath is executed with the aim of recovering the name of a specific data element, it is searched through information entities section, and if it isn’t found it is searched through the data element section.

```cpp
pXMLElement Element = DICOM->XPath( sString( "[/DICOM" ) 
+ "/IE:InformationEntities/IE:Entity[@Name='" 
+ Entity + "]/IE:Module/IE:Attribute[@Tag='" 
+ Tag + "]/IE:Name" ));

if(!Element->Size)
    Element = DICOM->XPath( sString( "/DICOM" ) 
+ "/DE:DataElements/DE:DataElement[@Tag='" 
+ Tag + "]/DE:Name" ));
```

**Figure 9.** C++ code fragment employed in the configuration file in order to find the name of a DICOM data element.

## 4. Discussion and Conclusions

The fact of provide a uniform format to the DICOM standard, and that also is compatible with multiple languages, was awarded in two opportunities. Firstly when the edition of the standard change from 2006 to 2007, only the configuration file was upgraded, and the whole system get actualized by the few hours that take to upgrade the XML file. Secondly, the query method of the DICOM standard by means of a browser showed to be more functional and versatile that using PDF or DOC files. Moreover it exist the possibility to consult it translated or in it original language.

The translation of the whole text of the DICOM standard to XML format results in a configuration file extremely huge. Furthermore, if it is in multiple languages, total size is multiplied by the quantity
of supported languages. In consequence the search or data query within the file can result too slowly and thorny due to the computational cost involved.

The solution of obtaining an adequate configuration file for all the cases is not efficient, the best solution is still an indexed binary file of specific format for each software system. But the generation of this specific configuration file is possible to be done from the DICOM standard in XML, using in some stage of the generation a syntax analyzer with the corresponding XSL file.

Finally with the depicted standardization for the XML file that has the DICOM standard, the proposed objective was reached because the application of the PICT 2003 Nº 430 project, that employs as medical imaging processing system their proprietary libraries ProcIma, can configure itself providing support to the DICOM standard from an XML file.

**Acknowledgments**

Present work was funded by the FONCIT Project, PICT 2003 Nº 430 “Fusion and Analysis of Multimodal Images Applied to Diagnosis and Oncological Radiotherapy”, San Juan Node.

**References**

[1] A. Valdèz, “Registración de Imágenes de PET y CT para Planificación de Radioterapia”, thesis of graduation of Magister in Biomedical Engineering, 2005.

[2] National Electrical Manufacturers Association, “DICOM, Digital Imaging and Communications in Medicine”, SUITE 1752 North 17th Street Roslyn, VA 22209 Ph: (703) 841-3285, http://medical.nema.org/, 2007.

[3] F. Lucena, A. Valdez, M. Gómez, J. Graffigna, O. Nasisi, “Diseño de una Librería para Procesamiento de Imágenes Médicas” at SIS 2006 – Argentinean Symposium in Informatics and Health.

[4] W3C, World Wide Web Consortium, “Extensible Markup Language (XML)” http://www.w3.org/XML/.

[5] Benjamin Jung, Ph.D, “DICOM-X - seamless integration of medical images into the HER”, en 18th IEEE Symposium on Computer-Based Medical Systems (CBMS’05).

[6] Anne Jumppanen, Sami Jaatinen, Teemu Toivanen, “DICOM XML” 27/05/2005 / Seminar in Medical Informatics.

[7] ISO: Internacional Organization for Standardization, “ISO 639-1:2002”, 1, ch. de la Voie-Creuse, Case postale 56, CH-1211 Geneva 20, Switzerland, http://www.iso.org/.

[8] ISO: International Organization for Standardization, “ISO 3166-1:2006”, 1, ch. de la Voie-Creuse, Case postale 56, CH-1211 Geneva 20, Switzerland, http://www.iso.org/.

[9] W3C, World Wide Web Consortium, “Extensible Markup Language (XML)” http://www.w3.org/XML/, 16/11/1999.