

Extension of an open source DICOM toolkit to support SCP-ECG Waveforms

D A Clunie
Pixelmed Publishing
Bangor, PA, USA

Keywords: SCP-ECG, OpenECG, DICOM, Java, open source software, ECG compression, standards

1. Introduction

The PixelMed DICOM toolkit is a set of classes intended for creating, reading, exchanging, displaying and validating DICOM images. It is implemented in 100% pure Java and hence is completely platform-independent. It is supplied as free and open source code and hence may be used or modified by anyone without restriction.

The software is primarily oriented towards display and encoding of images, which are the primary focus of DICOM. The recent addition to the toolkit of features for waveforms in the form of Magnetic Resonance Spectroscopy provided the core elements for 1D waveform display. This led naturally to the consideration of support in the toolkit for DICOM waveforms.

2. DICOM Waveforms

The DICOM cardiovascular working group added the ability to encode 12-lead ECGs, Holter™ data and hemodynamic waveforms to the DICOM standard in 2000, primarily for use in conjunction with images within a cardiovascular imaging environment, such as a cardiac catheterization laboratory. This approach reused the existing DICOM mechanism of tag-value pair encoding of attributes, together with an array of 16-bit waveform data samples. No waveform-specific compression mechanism is available in DICOM, though the entire dataset can be compressed with the same algorithm as used in zip. The approach of reusing an existing standard allows for simple and swift implementation, particularly if one already has access to existing toolkits and software such as the PixelMed DICOM toolkit. There are many available, both freely and commercially. Though there are relatively few commercial implementations of DICOM waveforms, some sample data is available.

It seemed logical in the course of the PixelMed toolkit waveform extension effort to add support for other standard waveform formats, particularly since there is no clear industry consensus as to whether DICOM or any other format of ECG encoding should be preferred. Indeed, there seems to be very poor support by vendors of ECG equipment of any standards at all.

3. SCP-ECG

SCP-ECG was chosen as a candidate for another format. SCP-ECG is potentially a more plausible format for widespread adoption by ECG vendors, since it was developed exclusively for the purpose of exchanging 12-lead ECGs. Despite this, there seems to be little evidence that this standard has been widely implemented in commercial products either, at least so far. Be that as it may, the standard is well documented, and is supported by a community group in the form of the OpenECG group. In addition, reference data sets and sample software are available for testing and experimentation. Specifically useful for this project was the availability of a web-based SCP to DICOM ECG conversion service. The announcement of the OpenECG programming competition provided additional incentive, as well as specific timelines to be met.

The SCP-ECG standard defines attributes of the data in different sections with different encoding forms. In section 1, the basic demographic information describing the patient and the acquisition are encoded as tag-value pairs. Most of the other sections, on the other hand, make use of fixed binary fields and variable length repeating segments. A mechanism is also present for the encoding of tagged annotations, though in the absence of any sample data for this section, it was excluded from the scope of the current project. However, this mixed approach of tagged and fixed length encoding was extremely tedious to implement. It required considerable hand typing of code, which was extremely error-prone. As a consequence, extreme attention to detail, and considerable testing and experimentation were necessary. A unified tag-value pair approach to encoding throughout the SCP-ECG standard would have been considerably easier and faster to implement.

A notable feature of the SCP-ECG standard is the emphasis on compression of the waveform data. Redundancy reduction by entropy coding alone is supported, and is implemented with Huffman coding. This is conventional and was very straightforward to implement. The numeric examples in the standard provided a

useful guide. However, since only modest reduction in size is achieved with entropy coding alone, the SCP-ECG standard also supports additional, irreversible (lossy) compression by the subtraction of reference beats and decimation (reduced sampling rate) outside protected zones. Implementing this was by far the most difficult part of the entire project. Indeed, the implementation is still not quite right, judging by the results of quantitative evaluation of reference samples, though the visual results are not obviously incorrect. Furthermore, the additional steps of filtering and interpolation defined as options in the standard have not yet been implemented, both due to their complexity as well as uncertainty that the preceding steps are as yet numerically correct.

4. Features Implemented

At the present time, the PixelMed toolkit is capable of reading and displaying either DICOM or SCP-ECG format 12-lead ECGs, and can validate the SCP-ECG format for correct structure whilst reading. If necessary, 12 leads can be reconstructed from the 8 usually encoded in the SCP-ECG format. The display example allows the user to scroll and scale the time and voltage axes. The header of the DICOM or SCP-ECG file can be displayed as a tree of attributes with names and values, including the tag-value pairs of SCP-ECG Section 1.

5. Implementation Choices

The choice of Java as a programming language and platform resulted in code that is inherently multiplatform, and has been shown to run on Mac OS X, Windows, Solaris and Linux, without any additional development effort.

Parsing and decompression take trivial time, largely due to the high performance of modern computing hardware. The user interface supported by the demonstration ECG-viewing application in the toolkit is sluggish when scrolling or scaling, largely as a consequence of the naïve approach taken of displaying path of thousands of line segments between samples, with no effort at re-sampling to the actual resolution of the display device.

6. Future Directions

It would be highly desirable to improve the quantitative results for reference beat subtraction, reversal of decimation, filtering and interpolation. Indeed, until this is done, the toolkit has limited usefulness for the high compression datasets.

The simple display application is limited to displaying the information contained in a single user-selected file, and it would be desirable to implement a patient and exam browser built from set of files, a folder or DICOMDIR file in the case of DICOM waveforms. The essential elements for support of this are already present in the toolkit, just not yet incorporated in the application.

The ability to annotate displayed waveforms with measurements supplied in files is important, and was only deferred due to lack of examples. Similarly, it would be desirable to support the creation and saving of annotations and measurements from within the viewing application.

Conversion between different formats is relatively straightforward, at least if compression is not required, and may be include in future versions. Furthermore, the support of additional formats, such as the HL7 V3 XML format that is now designated for FDA Definitive QT Study, and the Japanese Medical Format Encoding Rules (MFER) format, will also be considered.

7. Conclusions

Implementation of SCP-ECG was considerably more complex than adding DICOM waveform support. The existence of a toolkit that already supported the fundamental mechanisms of DICOM encoding was one obvious reason. Indeed this emphasizes the importance of writing standards based on existing widely implemented mechanisms, rather than creating everything down to the lowest level of bits and bytes de novo. That is, do not reinvent the wheel in terms of fundamental information encoding.

However, the primary reason for the implementation complexity of SCP-ECG was the high compression mode. Given this complexity, the potential for quantitative loss of information, and the existence of modern high bandwidth communication and storage mechanisms, as well as relatively effective general purpose compression schemes like zip and bzip2, is the additional complexity worth the effort, or is it instead a barrier to widespread adoption?