Accessing DICOM 2D/3D-Image and Waveform Data on Mobile Devices

Kroll M, Melzer K, Lipinski HG
michael.kroll@fh-dortmund.de
Germany

Abstract: Displaying medical images and waveforms which are stored in the DICOM file format is no longer a task that can be performed on desktop pc's only. This article describes a feasibility study where we have developed a set of applications for viewing and even analysing DICOM image and waveform objects on mobile devices such as mobile phones or PDAs. In order to overcome the shortcomings, especially the computing power, of those devices we use a client/server solution that has been completely implemented using the Java Programming Language on the server and the c lient devices as well.

Keywords: DICOM, Waveforms, Java 2 Micro Edition, Mobile Information Device Profile

1 Purpose

Accessing medical images as well as waveforms everywhere and every time on mobile hardware is very important for both, routine diagnostic and research. Therefore, we developed a set of applications for mobile devices during a feasibility study. In order to maintain platform independency the application suite has been developed using the Java Programming language especially the Java 2 Micro Edition (J2ME) [Sun02b] on top of the Connected Limited Device Configuration (CLDC) [Sun02a] and the Personal Java (Pjava) [Sun02d] specification to support a wide range of currently available devices. The second part that needs to be examined during the study was to investigate if a Java 2 Standard Edition [Sun02c] library used for decoding DICOM objects is also suitable for the J2ME platform by adding small configuration specific changes or if it is necessary to implement a completely new DICOM decode for this resource constrained platform.

2 Methods and Materials

In order to develop our mobile medical application suite we selected the Java programming language, especially the Java 2 Micro Edition for currently available mobile phones and low end PDAs such as the Palm Connected Organizer. For more powerful devices such as the Compaq IPaq or the Nokia Communicator 9210 smart phone we used Personal Java. On top of the J2ME technology we used the Connected Limited Device Configuration (CLDC) and the Mobile Information Device Profile (MIDP).

For transferring DICOM objects to the mobile platforms two approaches were implemented. The first uses the HTTP protocol which is the only protocol needed to be implemented on a MIDP supporting platform, while and the second approach uses DICOM objects which are stored in files directly. These files are accessed from removable media cards such as Compact-Flash-, Multi-Media-, Smart-Media- or Secure-Digital Cards directly. Internal databases such as the Palms build in databases are not supported. The images that are transferred to an MIDP device are stored in the Record Management System of the device in a proprietary format in order to speed up image visualisation.

For both, the image and waveform data that should be displayed on the device the DICOM objects are decoded on the client devices directly. The image data is decoded once during transferring them into the mobile devices memory and the stored in a proprietary format, whereas the waveforms are decoded from the removable media directly whenever they are displayed on the device. The image data can be viewed in a special form of stereoscopic visualisation technique. Since 3D displays are not available for mobile devices at present we used the chromatek [Chr02] technique to visualize 3D scenes of for instance an CT-image block that has been generated by a previously developed medical interactive stereo-3D visualisation tool called DicomGL-real3D [Mel01].

3 Results

We have chosen the Java Programming language because of the growing range of java-enabled hardware such as mobile phones, personal-digital-assistants, tablet-pc's and even embedded systems. On top of the MID-Profile the image viewer application called MIDPDicom was developed enabling the user to display images that are stored on a web server. MIDPDicom does not use a DICOM library that has been encapsulated in a set of classes in order to use them in other projects as well, but implemented directly into the application code to minimize the code size and decoding time of DICOM images to a minimum. The decode has been implemented without the use of floating point data types since CLDC does not support float or double data types. Using the MIDPDicom application DICOM images such as the CT-slice shown in figure 3 can be displayed on mobile devices.

The MIDPDicom MIDlet suite consists of multiple MIDlets, one for transferring a DICOM image object over HTTP, one for viewing a previously transferred images that is stored in the devices' memory and one for displaying the DICOM tags of the image. Transferring the image is achieved using the MIDPDicomFileView MIDlet as illustrated in figure 3.

Once the image is transferred to the device it can be selected an viewed offline using the MIDPDicomSavedFileView MIDLet. The screenshot illustrated in figure 3 shows the CT-slice on a Palm organizer and a mobile device emulation in the J2ME Wireless Toolkit.

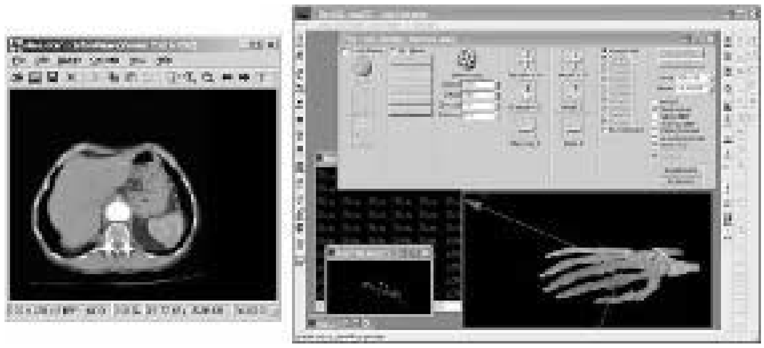

Figure 1: CT-slice shown in an image viewing application / medical 3d-simulation

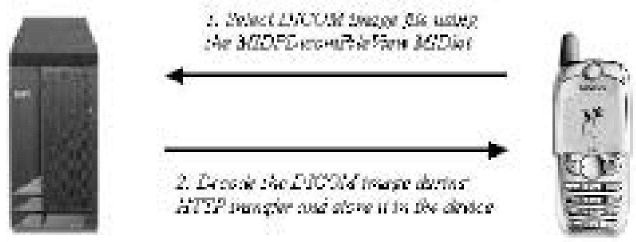

Figure 2: Transfer of a DICOM image to a MIDP enabled mobile phone

Figure 3: Working with 2d/3d images on different MIDP device platforms

Figure 4: The Waveform viewer running on different Java enabled mobile devices

The time that is needed to transfer an image over a GSM wireless network depends on the size of the image and the available transfer rates. We have tested images between 10-512kB using GPRS and HSCSD networks as well where the transfer of an image may last 1-30 min (speed is not optimal at this time because of control-method-overhead in the code for research purposes and there is only uncompressed-image support).

The second application created during the described feasibility study was an application that is capable of viewing DICOM waveform objects. Here we used a stream based DICOM toolkit, which has been initially developed for the Java 2 Standard Edition for use in Desktop PC applications. The main problem porting this toolkit to mobile devices was the reduced set of classes that is available on CLDC or Pjava platforms. Since the toolkit has been developed using Java 1.1 most of basic functionality of the toolkit could be used without adoption, but low level streams that are responsible for decoding number based values needed to be completely rewritten. The DICOM dictionary was implemented based on classes which are dynamically loaded, if information of a specific group is needed on the Pjava platform whereas the dictionary is stored in a RecordStore in the Jbed implementation. As a result from poring the toolkit to Pjava and the Jbed CLDC [Sun02a] platform the application runs on PalmOS powered organizers, the Compaq IPaq organizer and the Nokia Communicator 9210 as shown in figure 3.

All three implementations can load DICOM waveform objects from removable media cards directly. In view of the fact that all known memory cards support FAT file systems, file sharing between the Desktop PC and the above listed mobile devices is possible without need to convert the files into a proprietary file format that is optimised for the use on mobile devices.

Figure 5: Scenario of the MobileMed Internet portal

4 Summary

The discussed applications show that it is currently possible to view DICOM image and waveform objects on mobile devices. The goals of the mobile implementation is not to replace medical desktop workstations, but to enlarge their functionality. The implementation includes methods for intelligent interaction and reduction of drawbacks of limited devices. Because of using distributed standards and the standalone implementation the application can easy be integrated in medical systems.

5 Outlook

Both, the image and waveform viewer are standalone implementations at this time. Currently we are developing the so called MobileMed portal in order to offer server based image and waveform analysis that can be used with image and waveform data that is available on mobile devices. Figure 5 gives an overview of such a client-/server based image and waveform analysis scenario. The portal is capable of storing a set of DICOM objects in a password protected file repository and provides some standard analysis algorithms that can be used on mobile platforms using plug-ins. The previously discussed applications will be extended in order to communicate with such a file repository to use the files stored on the server for viewing them on the mobile device. Additionally the set of available algorithms can be extended by the user and used on the mobile device as well. A possible plug-in would be to create a cromatek stereoscopic image based on a set of CT-image slices, which can be displayed on a mobile device. Using the client server approach it will be possible to use complex analysis algorithms on mobile devices although the computing power of the mobile devices is not sufficient to perform those tasks itself.

6 Acknowledgements

Parts of the project presented in this paper has been supported by the Ministry of Education, Science, and Research, Duesseldorf (North Rhine Westfalia) Germany, (TRAFO 514-80081801/Lipinski).

Bibliography

[Chr02] Chromatek Inc. stereoscopic visualisation. http://www.chromatek.com/, 2002.

[Esm02] Esmertec Inc. Jbed Microedition CLDC. http://www.esmertec.com/, 2002.

[Mel01] Kay Melzer. DicomGL-real3D, OpenGl and Chromatek in medical applications. http://www.kaymelzer.de.tf/, 2000-2001.

[NEM01] NEMA Standards Publication PS 3.x. *Digital Imaging and Communications in Medicine (DICOM).* National Electrical Manufacturers Association, 2101 L Street, N. W., Washington, D. C. 20037, 1992-2001.

[Sun02a] Sun Microsystems Inc. CLDC and the K Virtual Machine (KVM). http://java.sun.com/products/cldc/, 2002.

[Sun02b] Sun Microsystems Inc. Java 2 Platform, Micro Edition (J2ME Platform). http://java.sun.com/j2me/, 2002.

[Sun02c] Sun Microsystems Inc. Java 2 Platform, Standard Edition (J2SE Platform). http://java.sun.com/j2se/, 2002.

[Sun02d] Sun Microsystems Inc. Personal Java (Pjava). http://java.sun.com/products/personaljava/, 2002.