Automating Production of Cross Media Content for Multi-channel Distribution

www.AXMEDIS.org

DE4.7.1
Analysis of Distribution towards Mobile

Version: 1.4
Date: 10/11/2005
Responsible: IRC (revised and approved by DSI)

Project Number: IST-2-511299 Project Title: AXMEDIS Deliverable Type: Report and Prototype Visible to User Groups: Yes Visible to Affiliated: Yes Visible to Public: Yes
Deliverable Number: DE4.7.1 Contractual Date of Delivery: M13 (end of September 2005) Actual Date of Delivery: 11/11/2005 Work-Package contributing to the Deliverable: (please state WP, subWP, See annex I) Task contributing to the Deliverable: WP4.7 Nature of the Deliverable: Report and Prototype Author(s): IRC, FUPF, DSI
Abstract: This report examines the state-of-the-art in standards and technologies that need to be integrated for media distribution to mobile clients. These include the state-of-the-art in device profiling standards, transcoding for mobile distribution, and, DRM support for mobiles and interoperability. The report describes the work performed in the respective period and an outline plan of the work to be done. It concludes with this plan elaborated and set out within the M13-M30 plan for the next period. **Keyword List:** media, content, distribution, profile, mobile, transcoding, ringtones, proxies, CC/PP, UAProf, W3C, Deli, adaptation algorithms, optimisation, DRM, interoperability. ……..

Table of Contents

AXMEDIS Copyright Notice

The following terms (including future possible amendments) set out the rights and obligations licensee will be requested to accept on entering into possession of any official AXMEDIS document either by downloading it from the web site or by any other means.

Any relevant AXMEDIS document includes this license. PLEASE READ THE FOLLOWING TERMS CAREFULLY AS THEY HAVE TO BE ACCEPTED PRIOR TO READING/USE OF THE DOCUMENT.

1. **DEFINITIONS**
   - i. "**Acceptance Date**" is the date on which these terms and conditions for entering into possession of the document have been accepted.
   - ii. "**Copyright**" stands for any content, document or portion of it that is covered by the copyright disclaimer in a Document.
   - iii. "**Licensor**" is AXMEDIS Consortium as a de-facto consortium of the EC project and any of its derivations in terms of companies and/or associations, see www.axmedis.org
   - iv. "**Document**" means the information contained in any electronic file, which has been published by the Licensor's as AXMEDIS official document and listed in the web site mentioned above or available by any other means.
   - v. "**Works**" means any works created by the licensee, which reproduce a Document or any of its part.
2. **LICENCE**
   1. The Licensor grants a non-exclusive royalty free licence to reproduce and use the Documents subject to present terms and conditions (the **Licence**) for the parts that are own and proprietary property the of AXMEDIS consortium or its members.
   2. In consideration of the Licensor granting the Licence, licensee agrees to adhere to the following terms and conditions.
3. **TERM AND TERMINATION**
   1. Granted Licence shall commence on Acceptance Date.
   2. Granted Licence will terminate automatically if licensee fails to comply with any of the terms and conditions of this Licence.
   3. Termination of this Licence does not affect either party's accrued rights and obligations as at the date of termination.
   4. Upon termination of this Licence for whatever reason, licensee shall cease to make any use of the accessed Copyright.
   5. All provisions of this Licence, which are necessary for the interpretation or enforcement of a party's rights or obligations, shall survive termination of this Licence and shall continue in full force and effect.
   6. Notwithstanding License termination, confidentiality clauses related to any content, document or part of it as stated in the document itself will remain in force for a period of 5 years after license issue date or the period stated in the document whichever is the longer.
4. **USE**
   1. Licensee shall not breach or denigrate the integrity of the Copyright Notice and in particular shall not:
      - i. remove this Copyright Notice on a Document or any of its reproduction in any form in which those may be achieved;
      - ii. change or remove the title of a Document;
      - iii. use all or any part of a Document as part of a specification or standard not emanating from the Licensor without the prior written consent of the Licensor; or
      - iv. do or permit others to do any act or omission in relation to a Document which is contrary to the rights and obligations as stated in the present license and agreed with the Licensor
5. **COPYRIGHT NOTICES**
   1. All Works shall bear a clear notice asserting the Licensor's Copyright. The notice shall use the wording employed by the Licensor in its own copyright notice unless the Licensor otherwise instructs licensees.
6. **WARRANTY**

1. The Licensor warrants the licensee that the present licence is issued on the basis of full Copyright ownership or re-licensing agreements granting the Licensor full licensing and enforcement power.
2. For the avoidance of doubt the licensee should be aware that although the Copyright in the documents is given under warranty this warranty does not extend to the content of any document which may contain references or specifications or technologies that are covered by patents (also of third parties) or that refer to other standards. AXMEDIS is not responsible and does not guarantee that the information contained in the document is fully proprietary of AXMEDIS consortium and/or partners.
3. Licensee hereby undertakes to the Licensor that he will, without prejudice to any other right of action which the Licensor may have, at all times keep the Licensor fully and effectively indemnified against all and any liability (which liability shall include, without limitation, all losses, costs, claims, expenses, demands, actions, damages, legal and other professional fees and expenses on a full indemnity basis) which the Licensor may suffer or incur as a result of, or by reason of, any breach or non-fulfilment of any of his obligations in respect of this Licence.

7. **INFRINGEMENT**

1. Licensee undertakes to notify promptly the Licensor of any threatened or actual infringement of the Copyright which comes to licensee notice and shall, at the Licensor's request and expense, do all such things as are reasonably necessary to defend and enforce the Licensor's rights in the Copyright.

8. **GOVERNING LAW AND JURISDICTION**

1. This Licence shall be subject to, and construed and interpreted in accordance with Italian law.
2. The parties irrevocably submit to the exclusive jurisdiction of the Italian Courts.

Please note that:

- You can become affiliated with AXMEDIS. This will give you the access to a huge amount of knowledge, information and source code related to the AXMEDIS Framework. If you are interested please contact P. Nesi at nesi@dsi.unifi.it. Once affiliated with AXMEDIS you will have the possibility of using the AXMEDIS specification and technology for your business.
- You can contribute to the improvement of AXMEDIS documents and specification by sending the contribution to P. Nesi at nesi@dsi.unifi.it
- You can attend AXMEDIS meetings that are open to public, for additional information see WWW.axmedis.org or contact P. Nesi at nesi@dsi.unifi.it

1. Executive Summary and Report Scope (mandatory)

The purpose of this deliverable is to report on the RTD work carried out to in order to characterise the challenges in automating the distribution of digital content to mobile clients. It also reports on the structure of the composite transcoding plug-in that has been researched and implemented. This is a composite plug-in in the sense that it performs both client's device profile resolution, and, the transcoding of ringtones.

An account of the latest evolution of the relevant profiling standards such as CC/PP and Deli is given at the outset. Similarly the state-of-the-art in DRM support for content distribution to mobile devices is described and the problems and the work performed to resolve them are outlined. The report ends with a description of the work to be done in the next phase of the project (M13-M30).

2. State-Of-The-Art Device Profiling Standards (IRC)

As the number and variety of devices connected to the Internet grows, there is a corresponding increase in the need to deliver content that is tailored to the capabilities of different devices. As part of a framework for content adaptation and contextualisation, a general purpose profile format is required that can describe the capabilities of a user agent and preferences of its user. CC/PP is designed to be such a format.

2.1 CC/PP Profile Structure

CC/PP stands for Composite Capabilities/Preferences Profiles, and is a way to specify what exactly a user agent (web browser) is capable of doing. This allows for sophisticated content negotiation techniques between web servers and clients, to produce optimised XML-based markup for display and use on a wide variety of web user agents.

A CC/PP profile is a description of device capabilities and user preferences that can be used to guide the adaptation of content presented to the particular device. Here profile does not refer to a subset of a particular specification, for example the CSS Mobile profile, but refers to the document(s) exchanged between devices that describe the capabilities of a device. As an example, a CC/PP profile of device "Sony Ericsson T39" in given in the Appendix.

The CC/PP proposal describes an interoperable encoding for capabilities and preferences of user agents, specifically web browsers. The proposal is also intended to support applications other than browsers, including email, calendars, etc. Support for peripherals like printers and fax machines will require other types of attributes such as type of printer, location, Postscript support, colour, etc. We believe an XML/RDF based approach would be suitable. However, metadata descriptions of devices like printers or fax machines may use a different scheme. The CC/PP proponents state that effort has been made to provide interoperability with other important proposals. However the recent trend for most mobile sets has been increasingly towards adopting UAProf as an alternative to CC/PP.

The basic data model for a CC/PP is a collection of tables. Though RDF makes modelling a wide range of data structures possible, it is unlikely that this flexibility will be used in the creation of complex data models for profiles. In the simplest form each table in the CC/PP is a collection of RDF statements with simple, atomic properties. These tables may be constructed from default settings, persistent local changes or temporary changes made by a user. One extension to the simple table of properties data model is the notion of a separate, subordinate collection of default properties. Default settings might be properties defined by the vendor. In the case of hardware the vendor often has a very good idea of the physical properties of any given model of the product. However, the current owner of the product may be able to add options, such as memory or persistent

store or additional I/O devices that add new properties or change the values of some original properties. These would be persistent local changes. An example of a temporary change would be turning the sound on or off.

2.2 W3C's CC/PP Working Group

The CC/PP Working Group was chartered in summer of 1999 and is an outgrowth of the W3C's Mobile Access Interest Group. The first meeting was held in Stockholm, Sweden, on the 23rd and 24th of September, and was hosted by Ericsson. Johan Hjelm of the W3C (and Ericsson) is the chair of the CC/PP Working Group.

Companies and organisations currently participating in the CC/PP working group include:

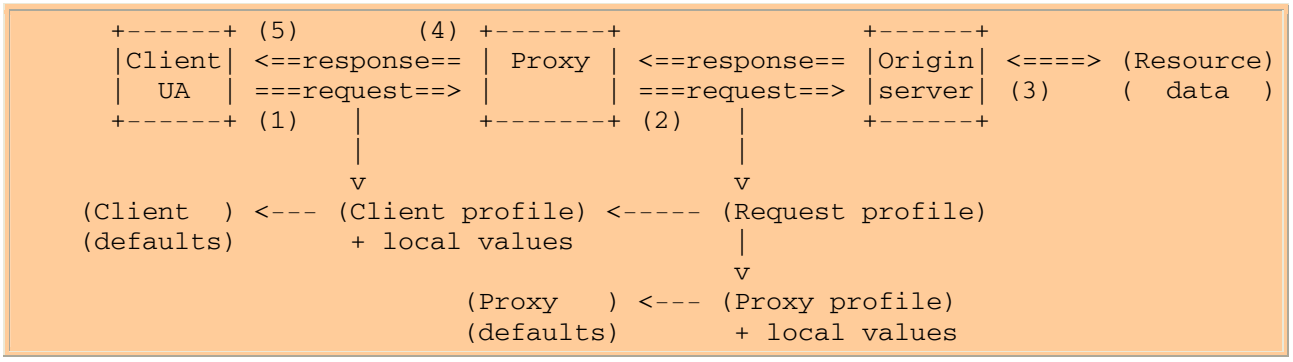
- Ericsson
- Fujitsu Laboratories
- HTML Writers Guild
- IBM
- IETF
- Interleaf
- Matsushita Electric Industrial Co.
- Nokia
- Nortel Networks
- SAP AG
- SBC Technology Resources
- Sun
- T-Mobile

For more information about CC/PP please refer to its homepage on:
http://www.w3.org/Mobile/CCPP/

Outline of request processing in HTTP

CC/PP is envisaged to be used with HTTP in the following fashion.

(This is not a protocol specification, just an indication of the kind of information flow envisaged. Defining a protocol to convey CC/PP information is a separate effort [CCPPEX]).



HTTP request processing(http://www.w3.org/TR/CCPP-struct-vocab/#Outline)

(a) The client sends an HTTP request, with an accompanying CC/PP client profile. The client profile may contain references to default profiles describing a range of common capabilities for the client concerned (e.g. a particular computer/operating system/browser combination, or a particular model of mobile device), and values that are variations from the default profile.

(b) The HTTP request may pass through a firewall/proxy that:

i) imposes constraints on the kinds of content that can be accessed, or,

ii) can adapt other forms of content to the capabilities of the requesting client.

(c) This proxy extends the CC/PP profile with a description of these constraints and adaptations, and sends this with the HTTP request onto the provider-server. The request may pass through several such proxies.

(d) The provider-server receives the request and interprets the CC/PP profile. It selects and/or generates content that matches the combined proxy and client capabilities described in the profile. This is sent to the last proxy in the request chain in an HTTP response.

(e) If required, the proxy applies any content adaptations, and any other functions it is designed to perform. The resulting response and content is passed back to the requesting client..

(f) The client receives the HTTP response and presents the content it contains.

NOTE: There is some overlap between CC/PP and the various HTTP accept-* headers. A protocol specification for using CC/PP with HTTP must indicate how HTTP 'accept-*' headers may be used, and how they interact with CC/PP profiles.

For CC/PP to work, several parties must participate. The good news is that everything's invisible to users unless they want to transmit preferences not found in the default profile, in which case some user configuration is required.

The parties involved and their respective roles are as follows:

Device vendors are responsible for authoring the reference profile describing the base capabilities of each type of device. In addition, vendors must provide browser support so that applications can attach a CC/PP profile to the HTTP request. Finally, vendors must ensure that all profiles are in valid CC/PP or UAProf syntax.

Content authors create text, images, sounds, and other media that can be read, viewed, or listened to.. The authors are responsible for creating multiple versions of the same content, suited to multiple devices. A widely used approach is to write content in XML and provide XSLT stylesheets that transform content for various delivery contexts.

Application developers write the Java code that retrieves different content based on the delivery context. Such applications can be written using JavaServer Pages (JSPs) or servlets.

2.3 UAProf

The UAProf specification is based on the CC/PP specification. Like CC/PP, a UAProf profile is a two level hierarchy composed of components and attributes. Unlike CC/PP, the UAProf specification also proposes a vocabulary - a specific set of components and attributes - to describe the next generation of WAP phones. The specification also describes two protocols for transmitting the profile from the client to the server. There is an open-source library called DELI developed at HP Labs that allows Java servlets to resolve HTTP requests containing delivery context information from CC/PP or UAProf capable devices and query the resolved profile Currently DELI only supports the W-HTTP protocol.

Profiles using the UAProf vocabulary consist of six components: HardwarePlatform, SoftwarePlatform, NetworkCharacteristics, BrowserUA, WapCharacteristics and PushCharacteristics. These components contain attributes. In DELI each attribute has a distinct name and has an associated collection type, attribute type and resolution rule.

In UAProf there are three collection types:

i. *Simple* contains a single value e.g. ColorCapable in HardwarePlatform.
ii. *Bag* contains multiple un-ordered values e.g. BluetoothProfile in the HardwarePlatform component.
iii. *Seq* contains multiple ordered values e.g. Ccpp-AcceptLanguage in the SoftwarePlatform component.

The profile is associated with the current network session or transaction. Each major component may have a collection of attributes or preferences. Examples of major components are the hardware platform upon which all the software is executing, the software platform upon which all the applications are hosted and each of the applications.

The UAProf vocabulary is described using the file uaprofspec.xml. This describes the attribute name, component, collectionType, attributeType and resolution rule of each component. The vocabulary description file has the following format:

```
<?xml version="1.0"?>
<vocabspec>
        <attribute>
                <name>CcppAccept</name>
                <component>SoftwarePlatform</component>
                <collectionType>Bag</collectionType>
                <attributeType>Literal</attributeType>
                <resolution>Append</resolution>
        </attribute>
</vocabspec>
```

The following is a simplified example of the sort of data expected to be encoded in these profiles.

Hardware Platform

Memory = 64mb

CPU = PPC

Screen = 640*400*8

BlueTooth = Yes

Software Platform

OS version = 1.0

HTML version = 4.0

Sound = ON

Images = Yes

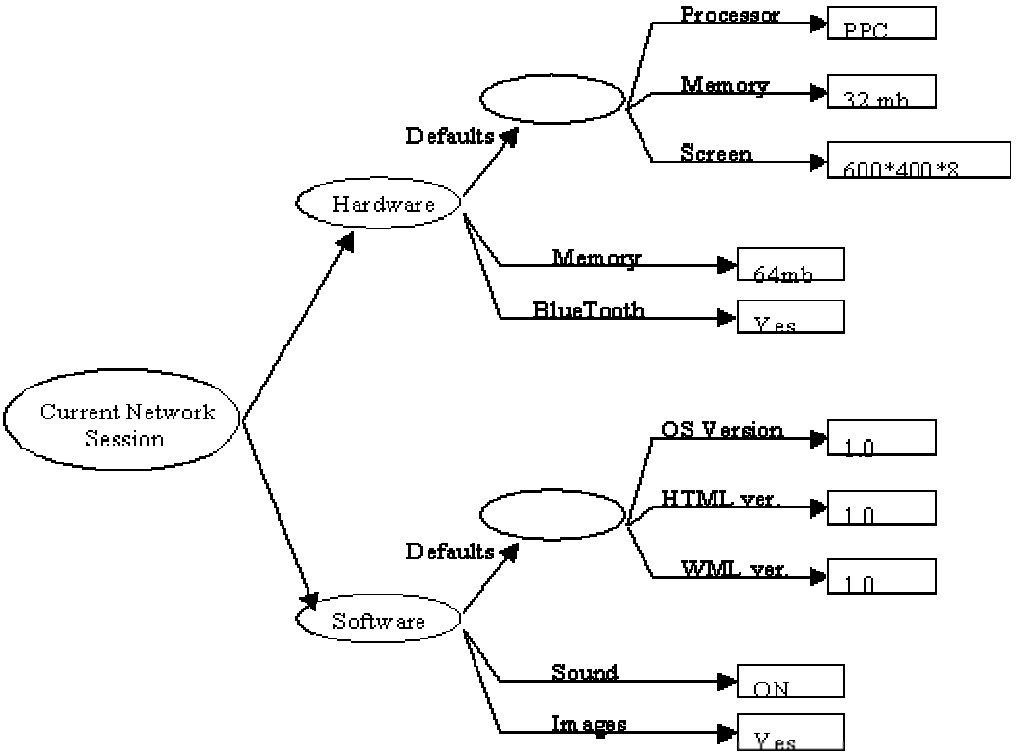
Email

Language = English

...

Some collections of properties and property values may be common to a particular component. For example: a specific model of a smart phone may come with a specific CPU, screen size and amount of memory by default. Gathering these "default" properties together as a distinct RDF resource makes it possible to independently retrieve and cache those properties. A collection of "default" properties is not mandatory, but it may improve network performance, especially the performance of relatively slow wireless networks.

Any RDF graph consists of nodes, arcs and leafs. Nodes are resources, arcs are properties and leafs are property values. An RDF graph based on the previous example that includes "Default" properties for each major component is relatively straightforward as depicted below:



The introduction of "Defaults" makes the graph of each major component more of a simple tree than a table. In this example the major components are associated with the current network session. In this case, the network session is serving as the root of a tree that includes the trees of each major component. RDF was originally intended to describe metadata associated with documents or other objects that can be named via a URL. The closest thing to a "document" associated with a CC/PP is the current network session.

From the point of view of any particular network transaction the only property or capability information that is important is whatever is "current". The network transaction does not care about the differences between defaults or persistent local changes; it only cares about the capabilities and preferences that apply to the current network transaction. Because this information may originate from multiple sources and because different parts of the capability profile may be differentially cached, the various components must be explicitly described in the network transaction.

The CC/PP is the encoding of profile information that needs to be shared between a client and a server, gateway or proxy. The persistent encoding of profile information and the encoding for the purposes of interoperability (communication) need not be the same. In this document we have considered the use of XML/RDF as the interoperability encoding. Persistent storage of profile information is left to the individual applications.

For the purpose of illustration consider below a more realistic example of inline encoding of a CC/PP for a hypothetical smart phone. This is an example of the type of information a phone might provide to a gateway/proxy/server. Note that we do not explicitly name the "current network session". Instead, the profiles of each major component is collected in a "Bag". This is probably not necessary since the document in question, the network session, is unlikely to contain additional RDF.

<?xml version="1.0"?>
<rdf:RDF

```
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:prf="http://www.w3.org/TR/WD-profile-vocabulary#">
 <rdf:Description about="HardwarePlatform">
   <prf:Defaults
    Vendor="Nokia"
    Model="2160"
    Type="PDA"
    ScreenSize="800x600x24"
    CPU="PPC"
    Keyboard="Yes"
    Memory="16mB"
    Bluetooth="YES"
    Speaker="Yes" />
  <prf:Modifications
    Memory="32mB" />
 </rdf:Description>
 <rdf:Description about="SoftwarePlatform">
  <prf:Defaults
  OS="EPOC1.0"
  HTMLVersion="4.0"
  JavaScriptVersion="4.0"
  WAPVersion="1.0"
  WMLScript="1.0" />
  <prf:Modifications
  Sound="Off"
  Images="Off" />
 </rdf:Description>
 <rdf:Description about="EpocEmail1.0">
   <prf:Defaults
  HTMLVersion="4.0" />
 </rdf:Description>
  <rdf:Description about="EpocCalendar1.0">
  <prf:Defaults
   HTMLVersion="4.0" />
 </rdf:Description>
  <rdf:Description about="UserPreferences">
 <prf:Defaults
       Language="English"/>
 </rdf:Description>
</rdf:RDF>
```

This sample profile is a collection of the capabilities and preferences associated with either a user or the hardware platform or a software component. Each collection of capabilities and preferences are organised within a description block. These description blocks may contain subordinate description blocks to describe default attributes or other collections of attributes.

2.4 DELI Profile Resolution Software

As noted above, DELI is an open-source library developed at HP Labs that allows Java servlets to resolve HTTP requests containing delivery context information from CC/PP or UAProf capable devices and supporting queries on the resolved profile.

Different web-enabled devices have different input, output, hardware, software, network and browser capabilities. In order for a web server or web-based application to provide optimised content to different clients

it requires a description of the capabilities of the client known as the delivery context. Of the two recent standards that support device profiling for describing the delivery context, we have already described the Composite Capabilities / Preferences Profile (CC/PP) created by the W3C; the other is the User Agent Profile (UAProf); an increasingly popular standard created by the WAP Forum.

DELI provides support for legacy devices so that the proprietary delivery context descriptions currently used by applications can be replaced by standardised CC/PP descriptions.

DELI can read vocabularies described using RDF schemas. The WAP Forum has published two such schemas to describe the two versions of UAProf currently in use. However the RDF Schema in its previous form did not provide an easy way of describing attributeType or resolution rules. Therefore the UAProf schemas stores this information in the comments field for each attribute. Therefore DELI also parses the comments fields to create the vocabulary. This is not an ideal solution so hopefully a more robust way of representing this information will be used in later versions of UAProf.

2.4.1 DELI Structures and Profile Resolution Futures

Most mobile devices now use UAProf because this is capable of preserving the ordering of profile elements and resolution of how to treat the profiles as new profiles are added (device, personal and channel profiles).

As the personal use circumstances change (e.g. location etc) and these profile elements have to be merged within the profile structure in a consistent way (for example those which have to update a field should update it and those which are simply to be added (appended) to append it and those that are time/location invariant to remain as they were (for example the display size) etc …

Deli was produced to structure the three profile models as follows:

Device reference profiles e.g. Nokia 6255 phone profiles,

In-line personal profiles e.g. personal preferences/customisation of device

In-line channel profiles e.g. Network BW and other constraints affecting traffic management, filtering, buffering etc.

This was so as to preserve the ordering and in this way deliver what is needed to deliver Profile Resolution. So all Deli's main code (a small 60K of JavaServlet) does is it takes the profile elements and builds its own structure so that it could then use this structure consistently to merge the profiles in an order preserving manner and with consistent resolution i.e. update those elements to be updated in a correct manner i.e. either update or append or leave alone the fields respectively as it should for correct interpretation and usage of the profiles by the media transcoder and other server services provided to the client.

This Deli's task and its development was necessary before because all the profiling representation schemas have traditionally been RDF based and though RDF has a structure and a vocabulary, it is not structured in the sense of being order preserving and this structuring was what Deli had to do additionally at that time by building its own structures according to the above three categories..

Recently RDF Multiple Models concept has been introduced and it is now possible, whilst working from RDF, to distinguish the above three profiles as essentially belonging to three distinct models and thus having an easier Profile Resolution and order Preserving task to do by querying across the three models separately and for each using UAProf resolution as follows:

If a profile field is "locked" then leave alone or use the first element

If a profile field is "over-ride" then use the last element (updated one)

If a profile field is "append" then merge together in that sequence e.g. English Italian will mean that the sever will first serve the English and then the Italian version of whatever it is (e.g. as in aircraft or airport announcements or songs etc.

2.5 WURFL

Another relevant source is WURFL. This is a free, open source project that provides an alternative source of information to UAProf. It provides a comprehensive resource of device information, and contains device information for 6000 variants of devices. Because WURFL is open source, anyone can contribute device information and corrections, not just device manufacturers. WURFL provides its own XML format for device characteristics description.

The Documentation and Developers guide is available on:

http://www.hpl.hp.com/personal/marbut/DeliUserGuideWEB.htm.

3. State-Of-The-Art Tool and Algorithms for Mobile Transcoding (IRC)

3.1 Introduction to Mobile Transcoding

Mobile Transcoding refers to the adaptation of multimedia information to enhance usability and manage the variable delivery to cater for heterogeneous client devices and user requirements on-demand. Content adaptation refers to the modification of the parameters of a specific media type (for instance, an image can be encoded using various resolutions, colour levels and intensities). There are multiple facets to developing such a framework foremost of which is a generalised representation of content to allow access to individuals with varying disabilities.

Transcoding can be defined as the conversion of one coded signal to another. While this definition can be interpreted quite broadly, it should be noted that research on audio/video transcoding is usually focused on adaptation either to a reduced availability of transport channel bandwidth or to a smaller display or to both these constraints. In the earliest work on transcoding, the majority of interest focused on reducing the bit rate to meet an available channel capacity. Additionally, researchers investigated conversions between constant bit-rate (CBR) streams and variable bit-rate (VBR) streams to facilitate more efficient transport of audio/video. In recent times as mobile devices with limited display and processing power became prevalent, transcoding objectives included achieving spatial resolution reduction, as well as temporal resolution reduction

Some of these common mobile transcoding operations are illustrated in Figure 1. In all of these cases, it is always possible to use a cascaded pixel-domain approach that decodes the original signal, performs the appropriate intermediate processing (if any), and fully re-encodes the processed signal subject to any new constraints.

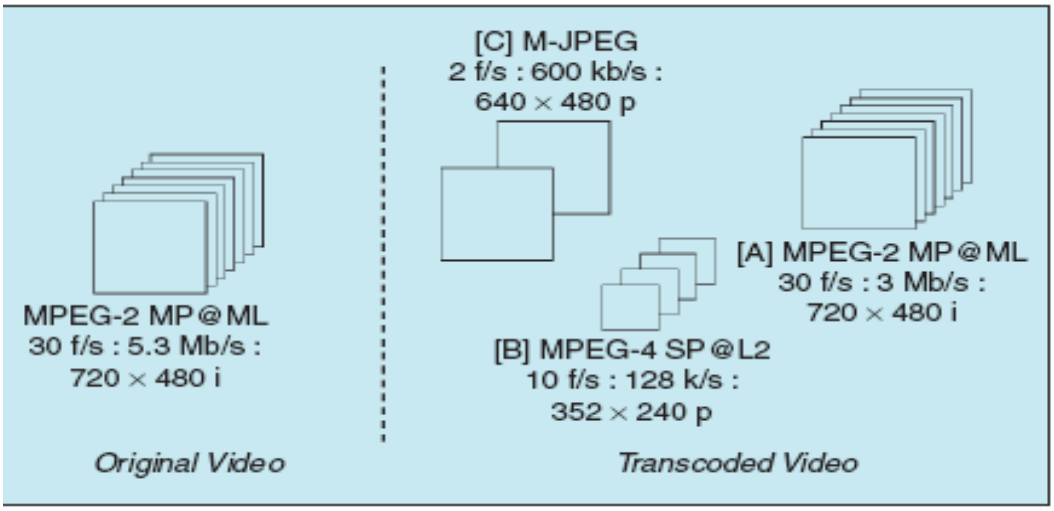


Fig. 1: Illustration of common video transcoding operations.

Original video is encoded in an MPEG-2 format (Main Profile at Main Level = MP@ML) at 5.3 Mb/s. The input resolution is 720 x 480 i (interlaced), and the temporal rate is 30 frames-per-second (f/s). As illustrated in Fig1 above, the transcoding pipeline is as follows:

1. Original video is transcoded to a reduced bit-rate of 3 Mb/s.
2. Original video is transcoded to an MPEG-4 format (Simple Profile at Level 2 = SP@L2) at 128 kb/s. The output resolution is 352 x 240 p (progressive) and the temporal rate is 10 f/s.
3. Original video is transcoded to a Motion-JPEG (M-JPEG) sequence of images at a temporal rate of 2 f/s, bit-rate of 600 kb/s, and output resolution of 640 x 480 p.

Transcoding techniques are used to serve the same underlying multimedia object at different quality levels to different users to accommodate users' operational constraints. The main effect of transcoding which is data reduction helps to reduce network end-to-end delay, loss, and delay jitter for the multimedia packets. Transcoding can be done in a number of ways including Spatial transcoding (to reduce the frame size), Temporal transcoding (to decrease the frame rate by dropping less significant frames), Colour transcoding (to reduce the data size by decreasing the colour depth). Design of the transcoding algorithm and the placement of the gateway are important issues in transcoder based adaptation schemes.

During transcoding, the transcoder must receive the information about the receiver's as well as the sender's device type and their basic specifications. The information about the device types are saved in xml files based on either CC/PP profile or UAProf protocol.

Key Goals of Transcoding

The key design goals of transcoding include two aspects:

1) To maintain the perceived quality during the transcoding process.
2) To keep the transcoding process complexity as low as possible.

The most straightforward solution is to simply decode the video signal, perform any post-processing if necessary, and re-encode the modified signal according to any new constraints. Although the quality is good, the complexity of this approach is relatively high. To avoid such a conversion, some researchers have proposed scalable coding schemes that can be easily scaled down according to the requirements of terminal or network. MPEG-4 Fine Granular Scalability is one of such scalable coding scheme. However, in the entertainment industry, many types of content exist with a fixed single-layer representation format. For example, the contents for Digital Television and DVD are encoded with an MPEG-2 format. In order for receivers, such as mobile terminals, to receive such signals, we must convert the bitstream from MPEG-2 to MPEG-4. Therefore, there are instances in which transcoding is certainly needed; the application scenarios are very clear. Such conversion must be done with low complexity and high quality.

During transcoding, the reference used for prediction is typically transcoded to a lower quality or spatial resolution; therefore, the error in this reference frame will accumulate. This accumulation of error is referred to as drift. Minimizing the amount of drift incurred during the transcoding process, while keeping complexity low, is a major goal of the transcoding design.

3.2 Adaptation Tools and Algorithms

The tasks which will be covered in transcoding algorithms are

1) Multiple job transcoding operation.
2) Transcoding of Ringtones, Sounds and Graphic content.
3) Transformation of Ringtones, Sounds and Graphic content.

We now discuss each of the above three aspects in turn.

3.2.1 Multiple Jobs Transcoding: tools for video adaptation

In most of video coding standards, the temporal redundancy is removed by predictive coding techniques, whereby the current field/frame is predicted with the previously coded field/frame. The predictive difference is then quantised and coded with variable run-length codes. In the case of video objects (but this also applies to audio and multiple objects), the FFMPEG and the FOBS library may be used.

3.2.1.1 Libraries for Video Transcoding

a. FFmpeg

FFmpeg is a complete solution to record, convert and stream audio and video. It includes libavcodec, the leading audio/video codec library. FFmpeg is developed under Linux, but it can be compiled under most operating systems, including Windows.

FFmpeg is a very fast video and audio converter. It can also grab from a live audio/video source. The command line interface is designed to be intuitive, in the sense that ffmpeg tries to figure out all the parameters, when possible. You have usually to give only the target bitrate you want. FFmpeg can also convert from any sample rate to any other, and resize video on the fly with a high quality polyphase filter.

A list of supported formats as well as windows platform specific information is given in the appendix. Moreover Developer guide is available on http://ffmpeg.sourceforge.net/ffmpeg-doc.html#SEC24

b. FOBS

FOB is an object oriented wrapper for the *ffmpeg* library. This allows one to build object oriented applications that work with MultiMedia files in multiplatform environments. FOBS are Ffmpeg OBjectS. It is a set of object oriented APIs to deal with media. It relies on the ffmpeg library, but provides developers with a much simpler programming interface.

FOB is currently available in C++. The Java API is being developed and the current version already includes a JMF plugin. It has been successfully tested in a range of platforms (Linux, MacOS X, Win32 (MinGW)). The documentation of the FOBS is available on http://fobs.sourceforge.net/features.html.

3.2.2 Multiple Jobs Transcoding: tools for ringtone adaptation

Almost everyone in the western industrialized world has heard a ringtone. The term today applies almost exclusively to those sometimes irritating but always informative sounds that mobile phones (cell phones) make.

Importance of Ringtones

Well the obvious reason for a ringtone is to alert the phone owner that they have an incoming call that should be attended to. But there are a number of reasons consumers use distinct ringtones.

a. Distinguish Callers- Advanced feature sets allow for different sounds to signify different callers.
b. Identification - When you are in a meeting and a phone rings you will know if it's yours without having to look.
c. Fun – a ringtone is viewed by some as imparting a personality on a cell phone with some character that can be appealing

Ringtones are likely to continue as a major component of mobile phone technology for quite some time. The actual ringtone may eventually be replaced by just standard music. It is feasible that any sound file could be used as a ringtone. There are several types of ring tone as follows:

Types of Ringtones

a. Note-based ringtones: These contain only information about *music events* (notes, instructions and instruments) comprising the ringtone. These can be *monophonic*, i.e. at any given point of time only one event (note or pause) is active; and *polyphonic*, when several number of notes can play at the same time. *Polyphony level* is the maximum number of events the phone can handle simultaneously. *Instant polyphony level* is the number of events active at a given time.

Examples of note-based ringtones are:

- *monophonic*: RTTTL, Nokia Smart Messaging, Sagem Mono, Motorola proprietary, IMelody, emelody

- *polyphonic*: MIDI, SP-MIDI, SMAF (without PCM samples), Sagem Poly, CMX (without PCM samples)

So the main difference is that monophonic just means that only one tone or sound can be played at a time and polyphonic means several notes can be played at once.

b. PCM ringtones (also known as *truetones*, *mastertones*, *voicetones):* These contain actual sound in the original audio waveform. PCM tones can contain much more information than note-based ringtones (*because certain sounds like human speech and sound effects generally can't be described as a note played by some instrument*), but this is at a cost since PCM files are larger than note-based files. The term "*polyphony*" is not applicable to PCM ringtones. PCM ringtones can be either *monophonic* or *stereophonic.*

Examples of PCM ringtones are as follows:

- MP3, AMR, QCP, AAC, SMAF (PCM-only), CMX (PCM-only), WAVE

c. Mixed ringtones. These ringtone types take the best of both worlds providing an ability to describe standard sounds like notes and to embed custom PCM samples into the music resulting in relatively smaller files and the good quality.

Examples of mixed ringtones are SMAF and CMX.

Ringtones Play Modes

The above three distinct types of ringtones get played in different ways, and the result is also different as described below:

a. Note-based ringtones: each event gets *synthesized* by the phone according either to the formula corresponding to each instrument (basically, imitating the sound of a particular instrument) or to the audio sample of the same event (stored in the phone's *sound bank* – a collection of all instruments and sounds). The sound gets synthesized in real time and immediately sent to the phone speaker.

b. PCM ringtones: the ringtone gets converted into the uncompressed form and then routed directly to the phone speaker.

c. Mixed ringtones: first the process described in 'a', takes place, and then the resulting audio stream gets mixed with the PCM samples (also in real time) and get played by the phone speaker.

The major difference amongst different ringtone play modes is the perceived quality of the resulting ringtone. One user may argue that polyphonic ringtones generally sound more clear than PCM ringtones, another user may disagree with this, Such quality judgments are however subjective but here are some facts:

a. The perceived quality of the playback of a monophonic, polyphonic or a mixed ringtone depends on the sound bank built into the phone and on the synthesizer (*formulas, algorithms, processing speed, etc.*). PCM ringtones on the other hand do not require the phone to make any guesswork – only to uncompress it and send to the speaker.

b. PCM ringtones can contain any kind of sound, and in a few cases if the phone supports polyphonic ringtones with the polyphony of say 16 and PCM ringtones, and if the source ringtone has the polyphony of say 40, sometimes it makes sense to synthesize the sound with a good sound bank on a computer and send the resulting PCM file to the phone supporting only 16 voices

Leading Ringtone Formats

As described earlier ringtones are available in different formats. Most important formats are MP3, WAVE, MIDI, IMelody, AMR..

In order to make available the media files (typically audio) to be used as a ringtone on various devices, the media should be converted from one file format to another using a transcoding algorithm. Even for the same file format different devices requires different characteristics (e.g. bit rate, sampling rate, etc) for the media. In order to cater for the leading mobile devices in the industry the following transformation should be considered:

MIDI to MP3
MP3 to MIDI
MP3 to IMelody
WAV to MIDI

Format transcoding is one of the main adaptation functions needed by the AXMEDIS Framework, as it implies bitrate reduction when transcoding among compressed formats. In the case of Ringtone, the Awave Studio v9.3, FFMPEG/FOBS, TiMidity++ and AmazingMIDI libraries may be used.

3.2.2.1 Libraries for Ringtone Transcoding

a. Awave Studio v 9.3 -Read (r) and/or Write (w) File Formats Repertoire

Awave Studio is a multi-purpose audio tool that reads a veritable host of audio carrying file formats from different platforms, synthesizers and trackers. It can be used in a variety of ways: as an audio file format converter, an audio editor, an audio and MIDI player, and, last but not least, as a wave-table synthesizer instrument editor and converter. If one does a point by point comparison between Awave Studio v 9.3 and other transcoding software one finds for Awave Studio v 9.3 that:

i. It is more versatile, e.g. it not only converts files, but it also has full editing features, it can do batch conversions, it can also play songs, etc.
ii. It supports many more file formats than any other audio software.
iii. It delivers relatively superior performance - converting more parameters and translating them more accurately.

A*wave Studio* can read almost any imaginable audio or instrument file format - about 250 of them in fact; it can also write some 125 of them. No other software even comes close to this. Complete supported format list is

given below. Moreover detail information about the software is available on http://www.fmjsoft.com/awavestudio.html.

Awave Studio Read (r) and/or Write (w) File Formats Repertoire

Ext.	*R/W*	*Format*
404	R+W	Muon DS404 Bank File.
404	R+W	Muon DS404 Patch File.
669	R	669 tracker module.
A3S	R	Yamaha A3000 sample files (no extension).
AIFC	R	Compressed Audio Interchange File Format.
AIFF	R+W	Audio Interchange File Format.
AIS	R	Velvet Studio Instruments.
AKAI	R	AKAI S-series floppy disk image file.
AKP	R+W	AKAI S5000/S6000 program file.
ALAW	R+W	European telephony format audio.
ALW	R+W	European telephony format audio.
AMS	R	Extreme's Tracker module format.
AMS	R	Velvet Studio Module.
APE	R+W	Monkey Audio losslessly compressed files.
APEX	R+W	AVM Sample Studio bank file.
ARL	R+W	Aureal sound bank format.
ASE	R	Velvet Studio Sample.
ASF	R =	Active Streaming Format (audio part only).
ATAK??	R+W	Soundscape Audio-Take files.
AU	R+W	Sun/NeXT/DEC Audio file.
AVI	R	MS Audio Video Interleave file (audio parts only).
AVR	R	Audio Visual Research sound file.
BWF	R	Broadcast Wave File.
C01	R	Typhoon wave files.
CDA	R	Audio-CD track.
CDR	R	Raw Audio-CD format data.
DCM	R	DCM module format.
DEWF	R	Macintosh SoundCap/SoundEdit recorded instrument format
DIG	R	Digilink format.
DIG	R	Sound Designer I audio file.
DLP	R	DirectMusic Producer DLS file.
DLS	R+W	DownLoadable Sounds level 1.
DLS	R+W	DownLoadable Sounds level 2.
DLS	R+W	DownLoadable Sounds level 2+.
DLS	R+W	DownLoadable Sounds level 2++.
DLS	R+W	Mobile DLS - 'minimal feature set'.
DLS	R+W	Mobile DLS - 'full feature set'.
DMF	R	Delusion/XTracker Digital Music Fileformat.
DR8	R+W	FXpansion DR-008 drumkit.
DSF	R	Delusion/XTracker Digital Sample Fileformat.
DSM	R	Digital Sound Module tracker format.
DSS	R @	Olympus DSS file.
DTM	R	DigiTrekker module.
DWD	R+W	DiamondWare Digitized file.
EDA	R	Ensoniq ASR disk image.
EDE	R	Ensoniq EPS disk image.
EDK	R	Ensoniq KT disk image.
EDQ	R	Ensoniq SQ1/SQ2/KS32 disk image.
EDS	R	Ensoniq SQ80 disk image.

EDV	R	Ensoniq VFX-SD disk image.
EFA	R	Ensoniq ASR file.
EFE	R+W	Ensoniq EPS file - should work with all EPS-family!
EFK	R	Ensoniq KT file.
EFQ	R	Ensoniq SQ1/SQ2/KS32 file.
EFS	R	Ensoniq SQ80 file.
EFV	R	Ensoniq VFX-SD file.
EMB	R	Everest embedded bank file.
EMD	R	ABT Extended MoDule.
EMY	R+W	EMelody Ericsson mobile phone ring-tone format.
ESPS	R	ESPS audio files.
EUI	R	Ensoniq EPS family compacted disk image.
F2R	R	Farandoyle linear module format.
F3R	R	Farandoyle blocked linear module format.
F32	R	Raw 32bit IEEE floating point values.
F64	R	Raw 64bit IEEE floating point values.
FAR	R	Farandoyle tracker module.
FFF	R+W	GUS PnP bank file format.
FLAC	R+W	Free Lossless Audio Codec files.
FNK	R	FunkTracker module format.
FSM	R	Farandoyle Sample format.
FZB	R+W	Casio FZ-1 Bank dump.
FZF	R+W	Casio FZ-1 Full dump.
FZV	R+W	Casio FZ-1 Voice dump.
G721	R+W	Raw CCITT G.721 4bit ADPCM format data.
G723	R+W	Raw CCITT G.723 3 or 5bit ADPCM format data.
G726	R+W	Raw CCITT G.726 2, 3, 4 or 5bit ADPCM format data.
GDM	R	Bells, Whistles, and Sound Boards module format.
GI!	R	GigaStudio/GigaSampler split files.
GIG	R+W	GigaStudio/GigaSampler files (normal).
GIG	R+W	GigaStudio/GigaSampler files (compressed).
GKH	R	Ensoniq EPS family disk image file.
GSM	R+W	Raw GSM 6.10 audio stream.
GSM	R+W	Raw 'byte aligned' GSM 6.10 audio stream.
GSM	R+W	US Robotics voice modems GSM w.o. header / VoiceGuide / RapidComm.
GSM	R+W	US Robotics voice modems GSM w. header / QuickLink.
HCOM	R	Sound Tools HCOM format.
IFF	R+W	Interchange file format.
IMY	R+W	iMelody mobile phone ring-tone format.
INI	R	MWave DSP synth's mwsynth.ini GM-setup.
INI	R+W	Gravis UltraSound bank setup.
INRS	R	INRS-Telecommunications audio.
INS	W	Cakewalk Instrument definition file.
INS	R	Ensoniq EPS family instrument.
INS	R+W	Sample Cell / II Mac instrument.
INS	R+W	Sample Cell / II PC instrument.
IST	R	Digitrakker Instrument File.
IT	R	Impulse Tracker module.
ITI	R+W	Impulse Tracker instrument.
ITS	R+W	Impulse Tracker sample.
K25	R	Kurzweil K2500 file (Note: KRZ is W also).
K26	R	Kurzweil K2600 file (Note: KRZ is W also).
KFT	R	Kort T-series image file.
KIT	R+W	Native Instruments Battery v1 drum kit file.
KMP	R+W	Korg Triton KeyMaP file.
KMP	R+W	Korg Trinity KeyMaP file.

KR1	R	Kurzweil K2000/K2500/K2600 split file.
KRZ	R+W	Kurzweil K2000 file (Note: Also works with K2500/K2600).
KSC	R+W	Korg Triton SCript file.
KSC	R+W	Korg Trinity SCript file.
KSF	R+W	Korg Triton Sample File.
KSF	R+W	Korg Trinity Sample File.
MAP	R+W	Native Instruments Reaktor format.
MAT	R+W	Matlab variables binary file.
MAUD	R	MAUD sample format.
MDL	R	Digitrakker Module.
MED	R	OctaMED tracker module.
MID	R+W	Standard Midi File.
MLD	R	MFi/MFi2 - Melody Format for i-Mode a.k.a. i-Melody.
MLS	R	Miles Sound Tools 'compressed DLS'.
MMF	R	SMAF songs - Synthetic Music Mobile Application Format.
MOD	R	Amiga and PC tracker module.
MOV	R	Apple QuickTime audio.
MP1	R	MPEG audio stream, layer I.
MP2	R+W	% MPEG audio stream, layer II CBR.
MP3	R+W	* MPEG audio stream, layer III CBR.
MP3	R+W	* MPEG audio stream, layer III ABR.
MP3	R+W	* MPEG audio stream, layer III VBR.
MPA	R	MPEG audio stream, layer I, II or III.
MPEG	R	MPEG 1 system stream.
MPG	R	MPEG 1 system stream.
MSS	R+W	Miles Sound System DLS 1 + XMI file.
MT2	R	MadTracker 2 Module.
MTI	R+W	MadTracker 2 Instrument.
MTM	R	MultiTracker Module.
MUS	R	Doom/Heretic music file.
MUS	R+W	Musifile MPEG Layer II Audio stream.
MUS10	R	Mus10 audio.
MWS	W	MWave DSP synth's instrument extract.
NIST	R+W	NIST Sphere audio.
NVF	R+W	Creative Nomand Voice file.
O01	R	Typhoon vOice file.
OGG	R+W	! Vorbis Ogg stream.
OKT	R	Oktalyzer tracker module.
OUT	R	Roland S-5xx series floppy disk image (S-50, S-51, S-330, W-30, S-500, S-550).
OUT	R	Roland S-7xx series floppy disk image (S-70, S-700, S-750, S-760, S-770, S-772).
P	R+W	AKAI S1000/S1100/S01 program.
P	R+W	AKAI S3000/S3200/S2000/S2800 program.
PAC	R	SB Studio II package.
PAF	R+W	Ensoniq PARIS audio file.
PAT	R+W	Advanced Gravis Ultrasound / Forte tech. Patch.
PBF	R+W	Turtle Beach Pinnacle Bank File.
PCG	R+W	Korg Triton bank file.
PCG	R+W	Korg Trinity bank file.
PCM	R	OKI MSM6376 synth chip PCM format.
PGM	R+W	AKAI MPC-1000 drum set file + .WAV files.
PGM	R+W	AKAI MPC-2000XL drum set file + .WAV files.
PGM	R+W	AKAI MPC-2000 drum set file + .SND files.
PGM	R+W	AKAI MPC-3000 drum set file + .SND files.
PLM	R	DisorderTracker2 module.
PLS	R	DisorderTracker2 sample.
PPF	R	Turtle Beach Pinnacle Program File.

PRG	R+W	WAVmaker program.
PSB	R	Pinnacle Sound Bank.
PSION	R	PSION a-law audio.
PSM	R	Protracker Studio Module Format.
PTM	R	Poly Tracker module.
RA	W #	RealAudio audio file.
RAW	R	Raw signed PCM data.
RIF	R	Rapidcom Voice/Quicklink Rockwell ADPCM files.
RMI	R+W	RIFF-MIDI file (with or without embedded DLS).
Rockwell	R+W	Rockwell 2,3,4 bit ADPCM format.
Rockwell-2	R+W	Rockwell 2 bit ADPCM format.
Rockwell-3	R+W	Rockwell 3 bit ADPCM format.
Rockwell-4	R+W	Rockwell 4 bit ADPCM format.
S	R	AKAI S900/S950 sample.
S	R+W	AKAI S1000/S1100 sample.
S	R+W	AKAI S3000/S3200/S2000/S2800/S01 sample.
S3I	R+W	Scream tracker v3 instrument.
S3M	R	Scream tracker v3 module.
S3P	R+w	AKAI MESA II/PC S-series program.
SAM	R	Signed 8bit Sample data.
SB	R+W	Raw Signed Byte (8bit) data.
SBK	R+W	Emu SoundFont v1.x file.
SD	R+W	Sound Designer I audio.
SC2	R	Sample Cell / II PC instrument (Also W as .INS).
SD2	R+W	Sound Designer II flattened file.
SD2	R+W	Sound Designer II data fork.
SDK	R	Roland S-5xx series floppy disk image (S-50, S-51, S-330, W-30, S-500, S-550).
SDK	R	Roland S-7xx series floppy disk image (S-70, S-700, S-750, S-760, S-770, S-772).
SDS	R+W	Raw Midi Sample Dump Standard file.
SDW	R	Raw Signed DWord (32bit) data.
SDX	R	Midi Sample Dump Standard files as compacted by SDX.
SEQ	R+W	Sony Playstation MIDI Sequence.
SF	R+W	MTU/IRCAM SoundFile format.
SF2	R+W	Emu SoundFont v2.1 file.
SFARK	R	Melody Machine Compressed SoundFont.
SFD	R	SoundStage Sound File Data.
SFI	R	SoundStage Sound File Info.
SFR	R	Sonic Foundry Sample Resource.
SMD	R	SmdEd / J-Phone mobile songs.
SMP	R	Avalon sample file.
SMP	R	Ad Lib Gold Sample.
SMP	R+W	Samplevision format.
SND	R	Raw unsigned PCM data (Also W as .UB).
SND	R+W	AKAI MPC-60/2000/2000XL/3000 series samples.
SNDR	R	Sounder sound file.
SNDT	R	Sndtool sound file.
SOU	R	SB Studio II sound.
SPD	R	Speach Data file.
SPL	R	Digitrakker Sample.
SPPACK	R+W	SPPack sound sample.
STM	R	Scream tracker v2 module.
STS	R+W	Creamware STS-series sampler program.
SVX	R	Interchange file format, 8SVX/16SV.
SW	R+W	Raw Signed Word (16bit) data.
SXT	R+W	Reason 2.x NN-XT format.
SYW	R+W	Yamaha SY-series wave files (really named W??).

TXT	W	Ascii Text parameter description.
TXT	R+W	Ascii Text formatted audio data.
TXT	R+W	RTTTL/NokRing mobile phone ring-tone format.
TXT	R+W	Steinberg LM-4 banks.
TXW	R+W	Yamaha TX16W wave files (really named .W??).
UAX	R	Unreal Tournament Audio package.
UB	R+W	Raw Unsigned Byte (8bit) data.
UDW	R	Raw Unsigned DWord (32bit) data.
ULAW	R+W	US telephony format (CCITT G.711) audio.
ULW	R+W	US telephony format (CCITT G.711) audio.
ULT	R	UltraTracker modules.
UMX	R	Unreal Tournament Music package.
UNI	R	MikMod 'UniMod' format.
UW	R+W	Raw Unsigned Word (16bit) data.
UWF	R	UltraTracker Wave File.
V8	R	Covox 8bit audio (Also W as .UB).
VAB	R+W	Sony PlayStation/PS2 bank file.
VAG	R+W	Sony PlayStation/PS2 wave file.
VAP	R+W	Annotated speech file.
VM1	R	Panasonic voice file
VOC	R+W	Creative labs 'newer style' sound file.
VOC	R+W	Creative labs 'older style' sound file.
VOX	R+W	Dialogic 4-bit ADPCM.
VOX	R	Talking Technology Incorporated file.
VOX-6K	R+W	Dialogic 4-bit ADPCM, 6000 Hz.
VOX-8K	R+W	Dialogic 4-bit ADPCM, 8000 Hz.
VSB	R+W	Virtual Sampler Bank file.
W??	R+W	Yamaha TX16W waveform (Also see TXW).
W??	R+W	Yamaha SY-series waveform (Also see SYW).
W2A+W3A	R	Yamaha Motif 'all' file.
W2V+W3V	R+W	Yamaha Motif 'voices' file.
W2W+W3W	R+W	Yamaha Motif 'waveforms' file.
W7A+W8A	R	Yamaha Motif ES 'all' file.
W7V+W8V	R+W	Yamaha Motif ES 'voices' file.
W7W+W8W	R+W	Yamaha Motif ES 'waveforms' file.
W64	R+W	Sonic Foundry Wave-64 format.
WAV	R+W	Microsoft Wave file.
WA!	R+W	GigaSampler/GigaStudio compressed wave file.
WFB	R+W	Turtle Beach WaveFront Bank (Maui/Rio/Monterey).
WFD	R+W	Turtle Beach WaveFront Drum set (Maui/Rio/Monterey).
WFP	R+W	Turtle Beach WaveFront Program (Maui/Rio/Monterey).
WMA	R+W	= Windows Media Audio 4.
WMA	R+W	= Windows Media Audio 9 CBR.
WMA	R+W	= Windows Media Audio 9 VBR.
WMA	R+W	= Windows Media Audio 9 Voice.
WMA	R+W	= Windows Media Audio 9 Lossless.
WMV	R	= Windows Media Video (audio part only).
WUSIKSND	R+W	Wusikstation multisampled sounds.
XI	R+W	Fast Tracker 2 instrument.
XM	R	Fast Tracker 2 extended module.
XMI	R+W	Miles Sound System Extended Midi file.
YADPCM	R+W	Raw Yamaha 4-bit ADPCM format data.

* R Unrecognized formats prompts user to supply basic info.

@ DSS decoding requires Olympus DSS Player v3.x.
% MP2 encoding requires a free 3rd party DLL (TooLame.dll).

** MP3 encoding requires a free 3rd party DLL (Lame_enc.dll).*
! OGG encoding and decoding requires free external DLLs.
 # RA encoding requires free external DLLs from RealNetworks.
= WMA encoding and decoding requires the free Windows Media 9 run-time.

b. FFmpeg

FFmpeg is a complete solution to record, convert and stream audio and video. It includes libavcodec, the leading audio/video codec library. FFmpeg is developed under Linux, but it can be compiled under most operating systems, including Windows.

FFmpeg is a very fast video and audio converter. It can also grab from a live audio/video source. The command line interface is designed to be intuitive, in the sense that ffmpeg tries to figure out all the parameters, when possible. You have usually to give only the target bitrate you want. FFmpeg can also convert from any sample rate to any other, and resize video on the fly with a high quality polyphase filter.

A list of supported formats as well as windows platform specific information is given in the appendix. Moreover Developer guide is available on http://ffmpeg.sourceforge.net/ffmpeg-doc.html#SEC24. For more details in those libraries, see Part C of Axmedis Framework and Tool specification document

c. FOBS

FOB is an object oriented wrapper for the *ffmpeg* library. This allows one to build object oriented applications that work with MultiMedia files in multiplatform environments. FOBS are Ffmpeg OBjectS. It is a set of object oriented APIs to deal with media. It relies on the ffmpeg library, but provides developers with a much simpler programming interface. For more details in those libraries, see Part C of Axmedis Framework and Tool specification document.

FOBS is currently available in C++. The Java API is being developed and the current version already includes a JMF plugin. It has been successfully tested in a range of platforms (Linux, MacOS X, Win32 (MinGW)). The documentation of the FOBS is available on http://fobs.sourceforge.net/features.html.

d. AmazingMIDI

AmazingMIDI automatically transcribes music, converting WAV files into MIDI files. It can recognize single-instrument polyphonic music. It is a powerful tool to help one to transcribe music, to practice musical instruments and to make MIDI files, and so on.

AmazingMIDI creates an Output File (.mid) from an Input File (.wav) that contains musical data, and a Tone File (.wav) that consists of monotone data. AmazingMIDI analyzes the Input File, assuming that every sound in the file is played with the same tone colour as the one in the Tone File. As a result, even if the music contains several different instruments, AmazingMIDI writes down all detected notes as single-instrument music.

Supported File Formats

Input/Tone File (wav): PCM, 16-bit, 22.050kHz/44.100kHz, mono/stereo

Output File (mid): Standard MIDI File

Further detail information about the software is available on: http://www.pluto.dti.ne.jp/~araki/amazingmidi/.

e. TiMidity++

TiMidity++ is an open source MIDI to WAVE converter and player. It uses Gravis Ultrasound-compatible patch files and/or SoundFont Banks to generate digital audio data from general MIDI files. The audio data can be played through any sound device or stored on disk. On a fast machine, music can be played in real time. TiMidity++ is written in C and runs under Linux, FreeBSD, HP-UX, SunOS, MacOSX, and Win32, and porting to other systems with gcc should be easy.

Further detailed information and the software to download are available on:
http://timidity.s11.xrea.com/index.en.html#links.

3.2.2.2 Ringtones Processing Problems

There are some problems arising from mis-matches of either available open-source transcoding.

1. There is no open source software available for direct transcoding of some of the file formats into one another. Thus a two-pass solution has to be adopted as the transcoding regime in certain cases. For example for transcoding from MIDI to MP3 the conversion can be done via a two-steps process as follows:

   a. MIDI to WAV and then
   b. WAV to MP3.

2. With AmazingMIDI: The WAV file is a recording of any sound (including speech) and the MIDI file is a sequence of notes (similar to musical score). Therefore it is not possible to reproduce original WAV sound on a MIDI file EXACTLY.

3. The other problems that are associated with the Ringtones are transformations within the same file format, that are dependent on the target device capabilities, e.g. Bit-rate reduction, channel mixing, etc. These problems are those associated with general audio processing and hence are covered in the relevant section of audio transformations.

3.2.3 Tools for Image Adaptation

Image adaptation tools provide functions which can be used for scaling, resolution improvements, text drawing, and image decomposition etc. of an image file. In the case of Image Objects, the following libraries can be used for implementing the functions described above.

3.2.3.1 Libraries for Image Transcoding

a. ImageMagick Library

ImageMagick, version 6.2.3, is a free software suite designed to create, edit, and compose bitmap images. It can read, convert and write images in a large variety of formats. Images can be cropped, colors can be changed, various effects can be applied; images can be rotated and combined, and text, lines, polygons, ellipses and Bezier curves can be added to images and stretched and rotated.

Here are just a few examples of what ImageMagick can do:

i. Convert an image from one format to another (e.g. PNG to JPEG)
ii. Resize, rotate, sharpen, colour reduce, or add special effects to an image
iii. Create a montage of image thumbnails
iv. Create a transparent image suitable for use on the Web
v. Turn a group of images into a GIF animation sequence
vi. Create a composite image by combining several images
vii. Draw shapes or text on an image
viii. Decorate an image with a border or frame
ix. Describe the format and characteristics of an image

A list of supported formats is given below. The documentation of the ImageMagick is available on http://www.imagemagick.org/.

Tag	Mode	Description	Notes
ART	R	PFS: 1st Publisher	Format originally used on the Macintosh (MacPaint?) and later used for PFS: 1st Publisher clip art.
AVIhttp://www.jmcgowan.com/avi.html	R	Microsoft Audio/Visual Interleaved	
AVS	RW	AVS X image	
BMPhttp://msdn.microsoft.com/library/sdkdoc/gdi/bitmaps_9c6r.htm	RW	Microsoft Windows bitmap	
CGM	R	Computer Graphics Metafile	Requires ralcgm to render CGM files.
CIN	RW	Kodak Cineon Image Format	Cineon Image Format is a subset of SMTPE DPX.
CMYK	RW	Raw cyan, magenta, yellow, and black samples	Set -size and -depth to specify the image width, height, and depth.
CMYKA	RW	Raw cyan, magenta, yellow, black, and alpha samples	Set -size and -depth to specify the image width, height, and depth.
CUR	R	Microsoft Cursor Icon	
CUT	R	DR Halo	
DCM	R	Digital Imaging and Communications in Medicine (DICOM) image	Used by the medical community for images like X-rays.
DCX	RW	ZSoft IBM PC multi-page Paintbrush image	
DIB	RW	Microsoft Windows Device Independent Bitmap	DIB is a BMP file without the BMP header. Used to support embedded images in compound formats like WMF.
DPX	RW	Digital Moving Picture	

		Exchange	
EMF	R	Microsoft Enhanced Metafile (32-bit)	Only available under Microsoft Windows.
EPDF	RW	Encapsulated Portable Document Format	
EPI	RW	Adobe Encapsulated PostScript Interchange format	Requires Ghostscript to read.
EPS	RW	Adobe Encapsulated PostScript	Requires Ghostscript to read.
EPS2	W	Adobe Level II Encapsulated PostScript	Requires Ghostscript to read.
EPS3	W	Adobe Level III Encapsulated PostScript	Requires Ghostscript to read.
EPSF	RW	Adobe Encapsulated PostScript	Requires Ghostscript to read.
EPSI	RW	Adobe Encapsulated PostScript Interchange format	Requires Ghostscript to read.
EPT	RW	Adobe Encapsulated PostScript Interchange format with TIFF preview	Requires Ghostscript to read.
FAX	RW	Group 3 TIFF	See TIFF format. Note that FAX machines use non-square pixels which are 1.5 times wider than they are tall but computer displays use square pixels so FAX images may appear to be narrow unless they are explicitly resized using a resize specification of "150x100%".
FIG	R	FIG graphics format	Requires TransFig.
FITS	RW	Flexible Image Transport System	
FPX	RW	FlashPix Format	Requires FlashPix SDK.
GIF	RW	CompuServe Graphics Interchange Format	8-bit RGB PseudoColor with up to 256 palette entires. Specify the format "GIF87" to write the older version 87a of the format.
GPLT	R	Gnuplot plot files	Requires gnuplot3.5.tar.Z or later.
GRAY	RW	Raw gray samples	Use -size and -depth to specify the image width, height, and depth.
HPGL	R	HP-GL plotter language	Requires hp2xx-3.2.0.tar.gz
HTML	RW	Hypertext Markup Language with a client-side image map	Also known as "HTM". Requires html2ps to read.
ICO	R	Microsoft icon	Also known as "ICON".
JBIG	RW	Joint Bi-level Image experts Group file interchange format	Also known as "BIE" and "JBG". Requires jbigkit-1.0.tar.gz.
JNG	RW	Multiple-image Network Graphics	JPEG in a PNG-style wrapper with transparency. Requires libjpeg and libpng-1.0.2 or later, libpng-1.2.5 or later recommended.
JP2	RW	JPEG-2000 JP2 File Format Syntax	Requires jasper-1.600.0.zip
JPC	RW	JPEG-2000 Code Stream Syntax	Requires jasper-1.600.0.zip
JPEG	RW	Joint Photographic Experts Group JFIF format	Requires jpegsrc.v6b.tar.gz
MAN	R	Unix reference manual pages	Requires that GNU groff and Ghostcript are installed.
MAT	R	MATLAB image format	
MIFF	RW	Magick image file format	Open ImageMagick's own image format (with ASCII header) which ensures that no image attributes understood by ImageMagick are lost.
MONO	RW	Bi-level bitmap in least-significant-byte first order	

MNG	RW	Multiple-image Network Graphics	A PNG-like Image Format Supporting Multiple Images, Animation and Transparent JPEG. Requires libpng-1.0.2 or later, libpng-1.2.5 or later recommended.
MPEG	RW	Motion Picture Experts Group file interchange format (version 1)	Requires mpeg2vidcodec v12.tar.gz.
M2V	RW	Motion Picture Experts Group file interchange format (version 2)	Requires mpeg2vidcodec v12.tar.gz.
MPC	RW	Magick Persistent Cache image file format	The native "in-memory" ImageMagick uncompressed file format. This file format is identical to that used by Open ImageMagick to represent images in memory and is read in "zero time" via memory mapping. The MPC format is not portable and is not suitable as an archive format. It is suitable as an intermediate format for high-performance image processing. The MPC format requires two files to support one image. When writing the MPC format, a file with extension ".mpc" is used to store information about the image, while a file with extension ".cache" stores the image pixels. The storage space required by a MPC image (or an image in memory) may be calculated by the equation (5*QuantumDepth*Rows*Columns)/8.
MSL	RW	Magick Scripting Language	MSL is the XML-based scripting language supported by the conjure utility.
MTV	RW	MTV Raytracing image format	
MVG	RW	Magick Vector Graphics.	The native ImageMagick vector metafile format. A text file containing vector drawing commands accepted by convert's -draw option.
OTB	RW	On-the-air Bitmap	
P7	RW	Xv's Visual Schnauzer thumbnail format	
PALM	RW	Palm pixmap	
PBM	RW	Portable bitmap format (black and white)	
PCD	RW	Photo CD	The maximum resolution written is 768x512 pixels since larger images require huffman compression (which is not supported).
PCDS	RW	Photo CD	Decode with the sRGB color tables.
PCL	W	HP Page Control Language	For output to HP laser printers.
PCX	RW	ZSoft IBM PC Paintbrush file	
PDB	RW	Palm Database ImageViewer Format	
PDF	RW	Portable Document Format	Requires Ghostscript to read.
PFA	R	Postscript Type 1 font (ASCII)	Opening as file returns a preview image.
PFB	R	Postscript Type 1 font (binary)	Opening as file returns a preview image.
PGM	RW	Portable graymap format (gray scale)	
PICON	RW	Personal Icon	
PICT	RW	Apple Macintosh QuickDraw/PICT file	
PIX	R	Alias/Wavefront RLE image format	
PNG	RW	Portable Network Graphics	Requires libpng-1.0.2 or later, libpng-1.2.5 or later recommended.
PNM	RW	Portable anymap	PNM is a family of formats supporting portable bitmaps (PBM) , graymaps (PGM), and pixmaps (PPM). There is no file format associated with pnm itself. If PNM is used as the output format

			specifier, then ImageMagick automatically selects the most appropriate format to represent the image. The default is to write the binary version of the formats. Use +compress to write the ASCII version of the formats.
PPM	RW	Portable pixmap format (color)	
PS	RW	Adobe PostScript file	Requires Ghostscript to read.
PS2	RW	Adobe Level II PostScript file	Requires Ghostscript to read.
PS3	RW	Adobe Level III PostScript file	Requires Ghostscript to read.
PSD	RW	Adobe Photoshop bitmap file	
PTIF	RW	Pyramid encoded TIFF	Multi-resolution TIFF containing successively smaller versions of the image down to the size of an icon. The desired sub-image size may be specified when reading via the -size option.
PWP	R	Seattle File Works multi-image file	
RAD	R	Radiance image file	Requires that *ra_ppm* from the Radiance software package be installed.
RGB	RW	Raw red, green, and blue samples	Use -size and -depth to specify the image width, height, and depth.
RGBA	RW	Raw red, green, blue, and alpha samples	Use -size and -depth to specify the image width, height, and depth.
RLA	R	Alias/Wavefront image file	
RLE	R	Utah Run length encoded image file	
SCT	R	Scitex Continuous Tone Picture	
SFW	R	Seattle File Works image	
SGI	RW	Irix RGB image	
SHTML	W	Hypertext Markup Language client-side image map	Used to write HTML clickable image maps based on a the output of montage or a format which supports tiled images such as MIFF.
SUN	RW	SUN Rasterfile	
SVG	RW	Scalable Vector Graphics	Requires libxml2 and freetype-2. Note that SVG is a very complex specification so support is still not complete.
TGA	RW	Truevision Targa image	Also known as formats "ICB", "VDA", and "VST".
TIFF	RW	Tagged Image File Format	Also known as "TIF". Requires tiff-v3.6.1.tar.gz or later. Note that since Unisys claims a patent on the LZW algorithm (expiring in the US as of June 2003) used by LZW-compressed TIFF files, ImageMagick binary distributions do not include support for the LZW algorithm so LZW TIFF files can not be written. Although a patch is available for libtiff to enable building with LZW support. Users should consult the Unisy LZW web page before applying it.
TIM	R	PSX TIM file	
TTF	R	TrueType font file	Requires freetype 2. Opening as file returns a preview image.
TXT	RW	Raw text file	
UIL	W	X-Motif UIL table	
UYVY	RW	Interleaved YUV raw image	Use -size command line option to specify width and height.
VICAR	RW	VICAR rasterfile format	
VIFF	RW	Khoros Visualization Image File Format	
WBMP	RW	Wireless bitmap	Support for uncompressed monochrome only.
WMF	R	Windows Metafile	Requires libwmf. By default, renders WMF files using the dimensions specified by the metafile header. Use the -density option to adjust the output resolution, and thereby adjust the ouput size. The default output resolution is 72DPI so "-density

			144" results in an image twice as large as the default. Use **-background color** to specify the WMF background color (default white) or **-texture filename** to specify a background texture image.
WPG	R	Word Perfect Graphics File	
XBM	RW	X Windows system bitmap, black and white only	Used by the X Windows System to store monochrome icons.
XCF	R	GIMP image	
XPM	RW	X Windows system pixmap	Also known as "PM". Used by the X Windows System to store color icons.
XWD	RW	X Windows system window dump	Used by the X Windows System to save/display screen dumps.
YCbCr	RW	Raw Y, Cb, and Cr samples	Use -size and -depth to specify the image width, height, and depth.
YCbCrA	RW	Raw Y, Cb, Cr, and alpha samples	Use -size and -depth to specify the image width, height, and depth.
YUV	RW	CCIR 601 4:1:1	

3.2.4 Tools for Audio Adaptation

3.2.4.1 Libraries for Audio Transcoding

The most important libraries for the audio transcoding are FFMPEG, FOBS, SoundTouch and Libsndfile.

a. FFmpeg

FFmpeg is a very fast video and audio converter. It can also grab from a live audio/video source. The command line interface is designed to be intuitive, in the sense that ffmpeg tries to figure out all the parameters, when possible. Thus in using FFmpeg one usually has to specify only the target bit-rate required. FFmpeg can also convert from any sample rate to any other, and re-size video on the fly with a high quality polyphase filter. The list of audio codecs supported by FFMPEG through the libavcodec library is given in the Appendix.

b. Libsndfile

Libsndfile is a C library for reading and writing files containing sampled sound (such as MS Windows WAV and the Apple/SGI AIFF format) through one standard library interface. It is released in source code format under the GNU LGP License.

libsndfile has the following main features :

i. Ability to read and write a large number of file formats.
ii. A simple, elegant and easy to use Applications Programming Interface.
iii. Usable on Unix, Win32, MacOS and others.
iv. On the fly format conversion, including endian-ness swapping, type conversion and bit-width scaling.
v. Ability to open files in read/write mode.
vi. The ability to write the file header without closing the file (only on files open for write or read/write).
vii. Ability to query the library about all supported formats and retrieve text strings describing each format.

viii. The list of audio codecs supported by Libsndfile library is given in the Appendix.

c. SoundTouch

SoundTouch is an open-source audio processing library for changing the Tempo, Pitch and Playback Rates of audio streams or files:

i. Tempo (time-stretch)**:** Changes the sound to play at faster or slower speed than original, without affecting the sound pitch.
ii. Pitch (key)**:** Changes the sound pitch or key, without affecting the sound tempo or speed.
iii. Playback Rate**:** Changes both the sound tempo and pitch, as if an LP disc was played at wrong RPM rate.

The software as well as source code of the SoundTouch is available on:
http://sky.prohosting.com/oparviai/soundtouch/.

3.3 Leading Mobile Devices: Broad Profile of the Top Ten Devices

We have investigated the functionalities of the latest leading wireless devices such as mobile phones, smart phones and PDA. Thus we have compiled information on the main characteristics and specifications of the devices. The specifications of the leading devices are given in the excel sheet provided with this document. On the basis of this research we are able to set out the general characteristics of the top 10 leading devices as follows:

Characteristics	Specification
Screen Type	Transflective TFT
Color Screen	Yes
Colors	65K
Screen size	2.5” – 4.0”
Screen resolution	208 x 320 – 480 x 640
Memory Size	21 MB- 256 MB
ROM Size	32 MB- 256 MB
CPU Speed	266 MHz – 624MHz
Operating System version	Microsoft Windows Mobile 2000 - 2003
MP3 Player	Yes (09/10)
Voice Recording	YES (07/10)
Interface	USB (10/10)
Infrared	YES (10/10)
Bluetooth	Yes (08/10)
WLAN	No (06/10)
Battery Type	Lithium Ion / Lithium Polymer
Battery Time	8 hrs – 400 hrs
Weight	145 grams – 210 grams
Built-in Camera	Yes (05/10)
Width	58 mm – 113 mm
Height	58 mm – 130 mm
Thickness	15 mm – 26mm

4. JS_TRANSCODING & JS_CLIENT_PROFILE(IRC)

JS_TRANSCODING & JS_CLIENT_PROFILE are JAVAScript written as C++ dynamic link libraries for the content processing engines inside the scheduler. These executables are responsible for supporting the online transcoding jobs and determination of the device profile for the clients.

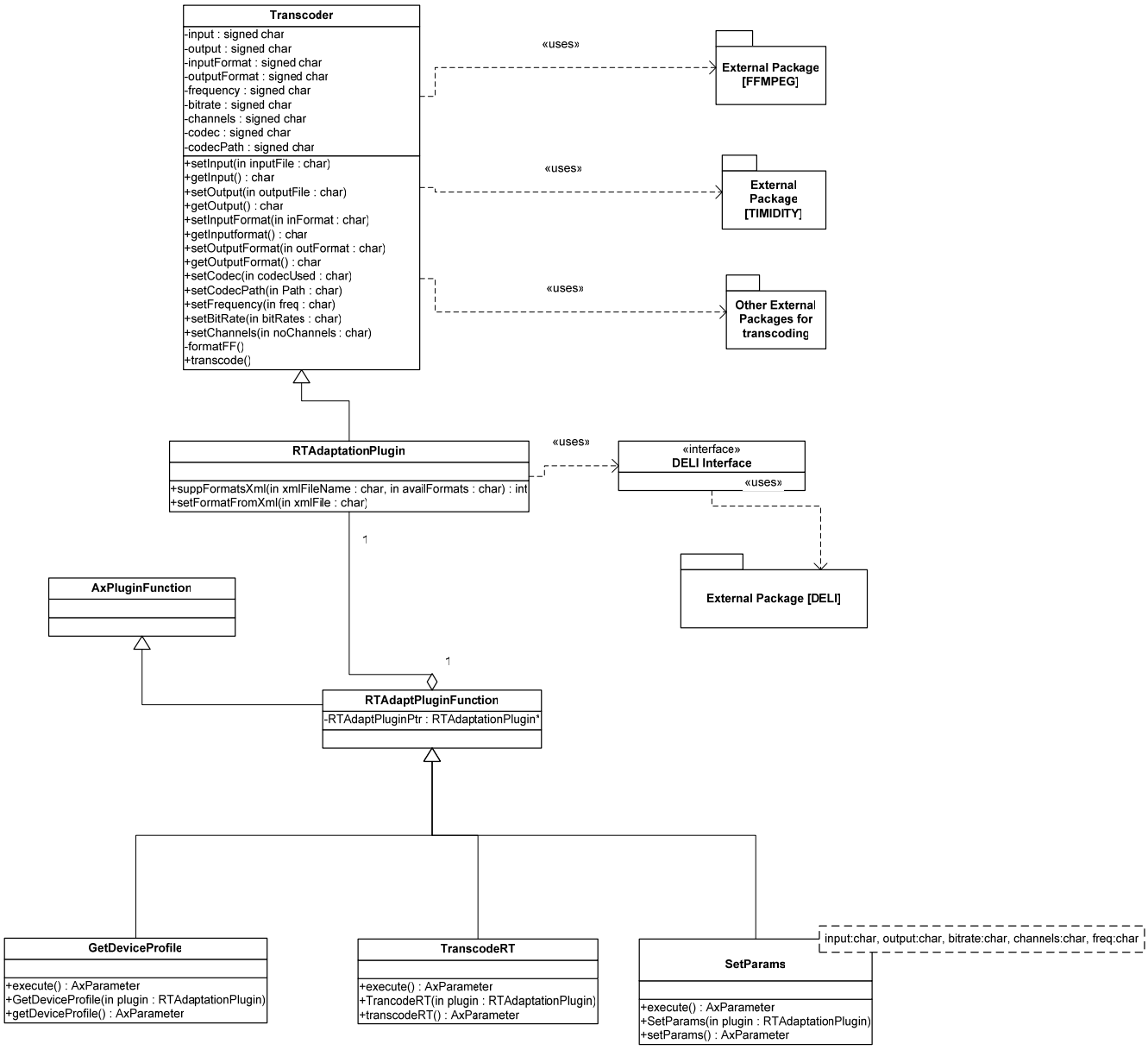
4.1 Work Done

Essentially, as already outlined elsewhere in the document (deli reference), the stages involved in media adaptation for on-demand delivery to mobile devices include the representation of the relevant profile types using appropriate classes which can be interrogated dynamically using a profiling standard such as CC/PP or UAProf and thereafter invoking a suitable transcoding algorithm to adapt the delivery of the media to the client device.

Our work in this area beyond the examination of the state-of–the-art has entailed the analysis of the available device profiling packages such as DELI as well as the transcoding algorithm libraries in order to distinguish the key profile types (e.g. inline, append, override) as well as the common class types and methods exposed by such algorithms in order to conclude the optimal design of appropriate transcoding algorithms to serve the axmedis platform having regard to the most sufficient and inclusive set of class types and methods to specify suitable transcoding algorithms.

The following UML diagram models the actual ringtone transcoding plug-in as developed. This is capable of resolving client's device profile and producing the transcoded media accordingly.

It must be noted that this plug-in is a composite plug-in in the senses that it performs both client's device profile resolution and transcoding of ringtones.



4.2 Work to be Done

During this period of the project, the ringtone transcoding plug-in has been developed based on the above UML architecture. Thisis a composite plug-in that is able to convert media types for the ringtones and also change the properties of the media e.g. bit rate, sampling frequency, channel reduction, etc.

This Plug-in also extracts the required parameters from the device profile and feeds them to the transcoder. Thus this Plug-in also delivers the client's device profile resolution capabilities.

In the next period of the project, this Plug-in will be decomposed into two dedicated Plug-ins; one to perform client profile resolution and the other to carry out the required transcoding job separately. These plug-ins will then be integrated with the AXCP engine.

Additionally, separate Plug-ins will be developed that models user preferences and context description for the mobile transcoding.

This work will be restructured according to:

- develop an AXMEDIS Plugin solution for the identified algorithm to process ringtones as described in the DE4.7.1, only for the ringtones, while the other plug-ins are developed by other partners. The Plug in has to work without parameters setting and has to provide the functionalities as a libraries of transcoding functionalities So that they would be used for a large number of processing.
- develop C++ module for loading CCPP device capabilities and make this module usable in AXCP Java Script by wrapping it with a JS class,
- develop C++ module for loading XML Context description coming from the device and make this module usable in AXCP Java Script by wrapping it with a JS class,
- develop C++ module for loading XML User Preferences coming from a server that allow users to set them and make this module usable in AXCP Java Script by wrapping it with a JS class.
- identification and/or develop a module to be used also from the inside of the AXCP Java Scrip to process Device Capabilities, Context Description and User Profile in order to take decision on the transcoding to be performed.

Thus the Device Capabilities, Context Description and User Profile would be used as general information for implementing different strategies of adaptation on different kind of digital resources and complex AXMEDIS objects. DSI, if needed, would provide support for the implementation of the module to be used also from the inside of the AXCP Java Scrip to process Device Capabilities, Context Description and User Profile in order to take decision on the transcoding to be performed.

5. DRM support for mobiles and interoperability (FUPF)

DRM support for mobiles and interoperability issues, may involve the adaptation of licenses and associated rights in order that mobiles are able to consume AXMEDIS objects. This adaptation usually means that a new license has to be created. Therefore, this newly created license has to respect the terms and conditions fixed in the original license or licenses for the adapted AXMEDIS objects or contents within these objects.

In order to support DRM for mobiles in AXMEDIS we have to take into account mainly two factors:

- AXMEDIS is following MPEG-21 standard, including MPEG-21 REL, the rights expression language (REL) defined in this standard to describe licenses and, in general, rights expressions
- Open Mobile Alliance is a leading organisation in the mobile sector. They are describing their own rights expression language, OMA REL, based on another rights expression language, ODRL

These two factors and the need of interoperability between RELs define the work to do inside AXMEDIS regarding DRM support and interoperability.

The following sections describe how DRM support for mobiles and interoperability can be done from the point of view of the rights expression languages that are being used in the mobile sector and in the AXMEDIS. Nevertheless, research cannot stop at this point, as other possibilities should be taken into account when describing DRM support.

Currently, we do not have a prototype that demonstrates the work described, but isolated modules that provide a brief part of the functionality. Nevertheless, the implementation of this prototype was not planned for this period and it is expected that the corresponding prototype will be ready for the planned date.

5.1 DRM Support and interoperability: State of the art

This section describes the different rights expression languages considered from the point of view of the DRM Support for mobiles and interoperability.

5.1.1 MPEG-21 Rights Expression Language (REL)

The different parties involved in the online distribution and consumption of multimedia resources need to exchange information about the rights, terms, and conditions associated with each resource at each step in the multimedia resource lifecycle. For example in distribution and super distribution business models, the information related to the rights and the terms and conditions under which the rights may be exercised needs to be communicated to each participant in the distribution chain.
In an end-to-end system, other considerations such as authenticity and integrity of Rights Expressions become important. For example, any content provider or distributor who issues rights to use or distribute resources must be identified and authorized. In addition, a Rights Expression may be accessed by different participants which requires mechanisms and semantics for validating the authenticity and integrity of the Rights Expression. A common Rights Expression Language that can be shared among all participants in this digital workflow is required.

Part 5 of the MPEG-21 standard specifies the syntax and semantics of a Rights Expression Language. MPEG chose XrML [1] as the basis for the development of the MPEG-21 Rights expression language.
MPEG-21 Rights Expression Language (REL) [2] specifies the syntax and semantics of the language for issuing rights for Users to act on Digital Items, their Components, Fragments, and Containers.
The most important concept in REL is the license that conceptually is a container of grants, each one of which conveys to a principal the sanction to exercise a right against a resource. A license if formed by the following elements:

- Title: this element provides a descriptive phrase about the License that is intended for human consumption in user interfaces. Automated processors must not interpret semantically the contents of such title elements.
- Inventory: this element is used for defining variables within a License. In the Inventory element of a license can be defined LicensePart elements that in turn can have licensePartId attributes that can be referenced from elsewhere in the license.
  Therefore, REL provides a syntactic mechanism for reducing redundancy and verbosity in Licenses that can be used throughout a License.
- Grant or GrantGroup: The Grants and GrantGroups contained in a license are the means by which authorization policies are conveyed in the REL architecture. Each Grant or GrantGroup that is an immediate child of a license exists independently within that license, no collective semantic (having to do with their particular ordering or otherwise) is intrinsically associated with the presence of two or more of them within a certain license.
- Other information: Using the wildcard construct from XML Schema, a License provides an extensibility hook within which license issuers may place additional content as they find appropriate and convenient. This can be useful for conveying information that is peripherally related to, for example, authentication and authorization, but is not part of the REL core infrastructure.
  It should, however, be carefully understood that not all processors of REL licenses will understand the semantics intended by any particular use of this extensibility hook. Processors of the license may choose wholly at their own discretion to completely ignore any such content that might be present therein.

Next figure shows the structure of a REL License.

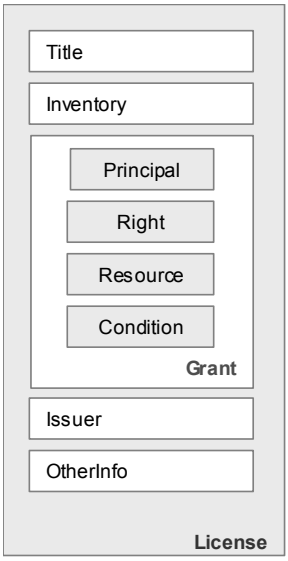


Figure REL License Structure

The most important concept within a license is the grant that conveys to a particular principal the sanction to exercise some identified right against some identified resource, possibly subject to the need for some condition to be first fulfilled. A Grant is an XML structure that is at the heart of the rights management and authorization policy semantics that REL is designed to express.

A grant is formed by four elements, a Principal that represents the unique identification of an entity involved in the granting or exercising of Rights. A Right that specifies an action or activity that a Principal may perform on, or using, some associated target Resource. A Resource that represents the object against which the Principal of a Grant has the Right to perform. The use of a digital resource in a Grant provides a means by which a sequence of digital bits can be identified within the Grant. The Condition element represents grammatical terms, conditions and obligations that a Principal must satisfy before it may take advantage of an authorization conveyed to it in a Grant. The issuer element that may contain two pieces of information, a set of issuer-specific details about the circumstances under which he issues the license, and an identification of the issuer, possibly coupled with a digital signature for the license. The optional issuer-specific details are found in the details element of the issuer. These details optionally include any of the following information the specific date and time at which this issuer claims to have effected his issuance of the license and an indication of the mechanism or mechanisms by which the Issuer of the license will, if he later Revokes it, post notice of such revocation. When checking for revocation, REL processing systems may choose to use any one of the identified mechanisms, that is, they are all considered equally authoritative as to the revocation status of the issuance of the License.

The structure of a REL license is the one described if it is in clear text, but a REL license can contain only an encryptedLicense element if the license is encrypted. The encryptedLicense element provides a mechanism by which the contents of a License may be encrypted and so hidden from view from inappropriate parties. This mechanism makes straightforward use of XML Encryption Syntax and Processing (XML Encryption). Specifically, the XML content model of a License is a choice between a sequence containing the elements previously described in this section and an encryptedLicense element that represents the encryption of the contents of the License element.

The principals, rights, resources and conditions of the REL are organized in three main groups. The first one, the Core specifies structural elements and types and how are they related. The standard extension and the multimedia extension specifies standard or multimedia principals, rights, resources and conditions.

At the heart of REL is the REL Core Schema whose elements and types define the core structural and validation semantics that comprises the essence of the specification. The REL Core Schema includes different elements and types organised in four main groups:

- Principals: Within REL, instances of the type Principal represent the unique identification of an entity involved in the granting or exercising of rights. They represent the subject that is permitted to carry out the action involved in exercising the Right. The principal element and its type are conceptually abstracts. Then, it does not indicate how a particular principal is actually identified and authenticated. Rather, this is carried out in types that are derivations of Principal. Such derived types may be defined in extensions to REL in order to provide, for example, a means by which Principals who are authenticated using some proprietary logon mechanism may be granted certain Rights using the REL License mechanism.
  There are derivations that are important and central enough to be defined within the REL core itself:
- allPrincipals: Structurally, an AllPrincipals Principal is a simple container of Principals. Semantically, an AllPrincipals represents the logical conjunct of the Principals represented by all of its children.
- keyHolder: Instances of a KeyHolder Principal represent entities which are identified by their possession of a certain cryptographic key. For example, using a KeyHolder, a Principal that uses public-key cryptography may be conceptually identified as that Principal which possesses the private key that corresponds to this-here public key.
- Rights: Within REL, instances of the type Right represent a verb that a Principal may be authorized to carry out under the authority conveyed by some authorized Grant. Typically, a Right specifies an action or activity that a Principal may perform on or using some associated target Resource. The semantic specification of each different particular kind of Right should indicate which kinds of Resource if any may be legally used in authorized Grants containing that Right.
  The element right and its type are conceptually abstract. Therefore, the type Right itself does not indicate any actual action or activity that may be carried out. Rather, such actions or activities are to be defined in types that are derivations of Right. Such derived types will commonly be defined in extensions to REL. However, the following rights are related to the domain of the REL core itself:
- issue: When an Issue element is used as the right in an authorized grant, it is required that resource against which the right is applied in fact be a grant or grantGroup. The grant then conveys the authorization for the principal to issue the resource.
  At the instant a License is issued, the issue right must be held by the issuer of the License with respect to all the grants and grantGroups directly authorized therein.
- obtain: When an obtain element is used as the right in an authorized grant, the resource must be present and be a grant or a grantGroup. The use of the obtain right can be conceptualized as an offer or advertisement for the sale of the contained grant
- possessProperty: The possessProperty right represents the right for the associated principal to claim ownership of a particular characteristic, which is listed as the resource associated with this Right.
- revoke: The authorized act of exercising the revoke right by a principal effects a retraction of a dsig:Signature that was previously issued and thus accomplishes a withdrawal of any authorization conveyed by that dsig:Signature.
- Resources: An instance of type resource represents the direct object against which the subject principal of a grant has the right to perform some verb. The actual element resource and its type are conceptually abstracts. That is, the type resource itself does not indicate any actual object against which a Right may

be carried out. Rather, such target objects are to be defined in types that are derivations of Resource. Such derived types will commonly be defined in extensions to REL. The relevant resources defined within the REL core:

- digitalResource: Use of a digitalResource resource in a grant provides a means by which an arbitrary sequence of digital bits can be identified as being the target object of relevance within the grant. Specifically, such bits are not required to be character strings that conform to the XML specification, but may be arbitrary binary data. The means by which this is accomplished breaks down into several cases. For example, the bits are to be physically present within the digitalResource or the bits are to be physically located at some external location (e.g. in a Web site).
- propertyAbstract: An instance of type propertyAbstract represents some sort of property that can be possessed by principals via possessProperty right.
- Conditions: Within REL, instances of the type Condition represent grammatical terms and conditions that a Principal must satisfy before it may take advantage of an authorization conveyed to it in a grant containing the condition instance. The semantic specification of each different particular kind of condition must indicate the details of the terms, conditions, and obligations that use of the Condition actually imposes. When these requirements are fulfilled, the Condition is said to be satisfied.
  The actual element condition and its type are conceptually abstracts. That is, the type Condition itself does not indicate the imposition of any actual term or condition. Rather, such terms and conditions are to be defined in types that are derivations of Condition. Such derived types will commonly be defined in extensions to REL. The conditions defined within the REL core that we consider relevant to detail:
  - AllConditions: Structurally, an allConditions is a simple container of conditions. Semantically, the allConditions represents a logical conjunct of the conditions represented by all of its children.
  - validityInterval: A ValidityInterval condition indicates a contiguous, unbroken interval of time. The semantics of the condition expressed is that the interval of the exercise of a right to which a validityInterval is applied must lie wholly within this interval. The delineation of the interval is expressed by the presence, as children of the condition, of up to two specific fixed time instants:
  - notBefore element, of type xsd:dateTime, indicates the inclusive instant in time at which the interval begins. If absent, the interval is considered to begin at an instant infinitely distant in the past
  - notAfter element, also of type xsd:dateTime, indicates the inclusive instant in time at which the interval ends. If absent, the interval is considered to end at an instant infinitely distant in the future.

The Standard Extension schema defines terms to extend the usability of the Core Schema, some of them are:

- Right Extensions: Right Uri.
- Resource Extensions: Property Extensions and Revocable.
- Condition Extensions: Stateful Condition, State Reference Value Pattern, Exercise Limit Condition, Transfer Control Condition, Seek Approval Condition, Track Report Condition, Track Query Condition, Validity Interval Floating Condition, Validity Time Metered Condition, Validity Time Periodic Condition, Fee Condition and Territory Condition.
- Payment Abstract and its Extensions: Payment Abstract, Rate, Payment Flat, Payment Metered, Payment per Interval, Payment per Use, Best Price Under, Call for Price and Markup.
- Service Description: WSDL and UDDI

- Country, Region and Currency Qualified Names: Namespace URI Structure, Country Qualified Names, Region Qualified Names and Currency Qualified Names.
- Matches XPath Function: Regular Expression Syntax and Flags.

The REL Multimedia Extension expands the Core Schema by specifying terms that relate to digital works. Specifically describes rights, conditions and metadata for digital works, that includes:

- Rights: Modify, Enlarge, Reduce, Move, Adapt, Extract, Embed, Play, Print, Execute, Install, Uninstall and Delete.
- Resources: Digital Item Resources.
- Conditions: Resource Attribute Conditions, Digital Item Conditions, Marking Conditions, Security Conditions and Transactional Conditions.
- Resource Attribute Set Definitions: Complement, Intersection, Set and Union.

5.1.2 ODRL

The ODRL [3] is a proposed language for the DRM community for the standardisation of expressing rights information over content. The ODRL is intended to provide flexible and interoperable mechanisms to support transparent and innovative use of digital resources in publishing, distributing and consuming of electronic publications, music, audio, movies, digital images, learning objects, computer software and other creations in digital form.
Using ODRL it is possible to specify, for a digital resource (music work, content, service, or software application), which is allowed to use that resource, the rights available to them and the terms, conditions or restrictions necessary to exercise those rights on the resource. The ODRL function is to express rights granted by some parties for specific resources and the conditions under which those rights apply.
ODRL is based on an extensible model for rights expressions, which involves three core entities and their relationships: **Party** (identifies an entity such as the person, organisation, or device to whom rights are granted), **Right** (includes permissions, which can then contain constraints, requirements, and conditions) and **Asset** (includes any physical or digital content).

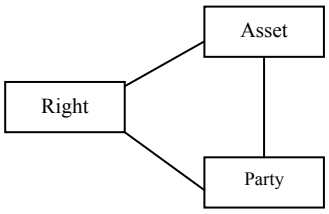


Figure Core elements of ODRL

5.1.3 OMA DRM Rights Expression Language

OMA (Open Mobile Alliance) has developed the OMA DRM Rights Expression Language versions [4] based on ODRL [3].
Rights are the collection of permissions and constraints defining under which circumstances access is granted to DRM Content. The structure of the rights expression language enables the following functionality:

1. Metadata such as version and content ID
2. The actual rights specification consisting of
    a. Linking to and providing protection information for the content, and
    b. Specification of usage rights and constraints

Models are used to group rights elements according to their functionality, and thus enable concise definition of elements and their semantics. The following models are used throughout this specification:

- Foundation model: constitutes the basis for rights. It contains the rights element bringing together meta information and agreement information. The foundation model serves as the starting point for incorporating the agreement model and the context model.
- Agreement model: expresses the Rights that are granted over an DRM Content. It consists of the agreement element connecting a set of Rights with the corresponding DRM Content specified with the asset element. The agreement model incorporates the permission model and the security model
- Context model: provides Meta information about the rights. It augments the foundation model, the agreement model, and the constraint model by expressing additional information.
- Permission model: augments the agreement model. It facilitates the expression of permissions over assets by specifying the access granted to a device. The permission model incorporates the constraint model allowing fine-grained consumption control of DRM Content. The set of permissions comprises play, display, execute, print, and export. Usage of the DRM Content MUST only be granted according to the permissions explicitly specified by the corresponding Rights Object(s). A permission that does not contain a constraint child element is unconstrained and access according to the respective permission element(s) MUST be granted. Note that the REL only specifies consumption and export rights and not management rights, e.g., install, uninstall, delete, or distribution rights. This is made possible by the separation of DRM Content and Rights Objects (although DRM Content and Rights Objects may be delivered together) freeing the REL from unnecessary complexity and overhead. Content can be stored; however, it can only be accessed if a corresponding Rights Object is available. Similarly, encrypted content can be super-distributed without unnecessarily complicating the REL; no separate distribution permissions are necessary, since DRM Content without the decryption key is of no value. The DRM Agent MUST ignore unknown or unsupported permission elements. The DRM Agent MUST NOT grant alternative, not explicitly specified rights to access Content instead. Known and supported permission elements defined by the same Rights Object MUST remain unaffected and the DRM Agent MUST grant access according to those. A Permission that is not granted due to unknown or unsupported constraints MUST NOT affect the granting of other permissions.
- Constraint model: enhances the permission model by providing fine-grained consumption control of content. Constraints are associated with one permission element at a time. For a permission to be granted all its constraints MUST be fulfilled. If a constraint is not understood or cannot be enforced by the consuming device the parent permission is invalid and must no be granted. If present, a constraint element should contain at least one of its child elements. If a constraint element does not contain any constraints such as count, datetime, etc. it is unconstrained, and a DRM Agent must grant unconstrained access according to the permission containing such an unconstrained constraint element.
- Inheritance model: describes how a parent Rights Object can specify Permissions and Constraints for one or more pieces of DRM Content each governed by a child Rights Object, using a limited subset of the ODRL inheritance model. The DRM Agent must not accept parent child Rights Objects constellations with more than one level of inheritance (i.e. parent-child). In other words, a parent Rights Object must not inherit Permissions and Constraints from another Rights Object.
- Security model: Security constitutes an important part of a DRM system. OMA DRM 2.0 provides confidentiality for the CEK of Rights Objects, integrity of the association between Rights Objects and DRM Content and Rights Object integrity and authenticity. The ODRL security model, which forms the basis for the security model of this specification, is based on XMLENC [5] and XMLSIG [6].

5.1.4 DRM Support for mobile and interoperability: The problems

Different systems define different rights expression languages. If we want to make them interoperable, or simply work with different kinds of devices we may be obliged to adapt rights expressions.

Moreover, changing versions of specifications make difficult to implement any kind of software to solve this issue.

5.1.5 DRM Support for mobile and interoperability: Work performed

In order to be able to transform rights expressions expressed from language to language, the first step to take is to study their common points. We have studied in detail the relationship between MPEG-21 REL and ODRL, based on the OMA DRM REL, as described next.

5.1.5.1 Interoperability between MPEG-21 REL and OMA DRM REL v1.0

In order to perform adaptation of rights between OMA REL and MPEG-21 REL, we propose an equivalent structure of the rights expression language of OMA DRM v1.0, but defined as a subset of MPEG-21 REL, and not as a subset of ODRL. This research work was presented in [7], [8].

The main concept in this equivalent structure is the r:license element that conceptually is a container of a r:grant or a r:grantgroup elements. A r:grant element conveys to someone the sanction to exercise a right (mx:play, mx:execute or mx:print are considered) against a resource. In this case, a resource is represented with the r:digitalResource element.

A r:license element also contains a r:otherinfo element that it is useful to include OMA DRM REL v1.0 information not considered by MPEG-21 REL, and it provides meta information about the rights.

The conditions considered are: sx:exerciseLimit that specifies the number of allowed exercises of a certain right, validityInterval that specifies an interval of time within which a right can be exercised and validityIntervalDurationPattern that specifies a period of time within which a right can be exercised.

```
<!ELEMENT r:license ( (r:grantgroup|r:grant), r:otherinfo? )>
<!ATTLIST r:license
            xmlns:r CDATA #FIXED
            "urn:mpeg:mpeg21:2003:01-REL-R-NS”
            xmlns:dsig CDATA #FIXED
            "http://www.w3.org/2000/09/xmldsig#"
            xmlns:mx CDATA #FIXED
            "urn:mpeg:mpeg21:2003:01-REL-MX-NS"
            xmlns:sx CDATA #FIXED
            "urn:mpeg:mpeg21:2003:01-REL-SX-NS">

<!ELEMENT r:grantgroup (r:grant+)>
<!ELEMENT r:grant
            ((mx:play|mx:execute|mx:print)?,
              r:digitalResource,
              r:allConditions?)>

<!ELEMENT mx:play EMPTY>
<!ELEMENT mx:execute EMPTY>
<!ELEMENT mx:print EMPTY>

<!ELEMENT r:digitalResource (r:nonSecureIndirect) >
<!ELEMENT r:nonSecureIndirect EMPTY>
<!ATTLIST r:nonSecureIndirect URI CDATA #IMPLIED>

<!ELEMENT r:allConditions
            (sx:exerciseLimit?,
             validityInterval?,
             validityIntervalDurationPattern?)>
<!ELEMENT sx:exerciseLimit (sx:count)>
```

```
<!ELEMENT sx:count (#PCDATA)>
<!ELEMENT r:validityInterval (r:notBefore?, r:notAfter?)>
<!ELEMENT r:notBefore (#PCDATA)>
<!ELEMENT r:notAfter (#PCDATA)>
<!ELEMENT sx:validityIntervalDurationPattern (sx:duration)>
<!ELEMENT sx:duration (#PCDATA)>
<!ELEMENT r:otherinfo (version?,KeyValue?)>
<!ELEMENT version (#PCDATA)>
<!ELEMENT KeyValue (#PCDATA)>
```

Figure OMA-based MPEG-21 REL DTD

5.1.5.2 Interoperability between MPEG-21 REL and OMA DRM REL v2.0

In this section, we introduce different tables with XML equivalences, between the OMA DRM REL v2.0 and the MPEG-21 REL subset, that will lead us to achieve interoperability between the MPEG-21 REL subset for the mobile domain and OMA DRM REL v2.0 specification. This work has been presented in [9],[10].

We use different models to group the XML equivalences according to their functionality and license structure. The models described in this section are: Basic equivalences, Rights and Conditions. They define the elements that form the subset of MPEG-21 REL that fulfils OMA DRM REL v2.0 specification.

Basic equivalences

The basic equivalences constitutes the basis for licenses and includes the necessary elements in any license. The OMA DRM REL <rights> and <asset> elements are represented with the MPEG 21 REL <license> and <digitalResource> elements. The OMA <context> element provides meta information about the rights, and is represented with the MPEG-21 <otherinfo> element.

Table Basic model

OMA DRM REL v2.0	OMA-based MPEG-21 REL
<o-ex:rights>	*<r:license>*
<o-ex:context> <o-dd:version>2.0</o-dd:version> <o-dd:uid>RightsObjectID</o-dd:uid> </o-ex:context>	*<r:otherinfo> <version>1.0 </version> <uid>RightsObjectID</uid> </r:otherinfo>*
<o-ex:asset> <o-ex:context> <o-dd:uid>ContentID </o-dd:uid> </o-ex:context> </o-ex:asset>	*<r:digitalResource> <r:nonSecureIndirect URI='ContentID' /> </r:digitalResource>*

Rights

In the following table we introduce the MPEG-21 REL rights equivalent to the rights specified in OMA DRM REL. The <display> and <play> elements are represented with the <play> element, the <export - move> element with the <move> element and the <export - copy> element with the <adapt> element and <prohibitedAttributeChanges> elements.

Table Rights model

OMA DRM REL v2.0	OMA-based MPEG-21 REL
<o-dd:display/>	*<mx:play />*
<o-dd:play/>	*<mx:play />*
<o-dd:execute/>	*<mx:execute />*

<o-dd:print/>	<mx:print />
<oma-dd:export oma-dd:mode="move"> <o-ex:constraint> <oma-dd:system> <o-ex:context> <o-dd:version> 1.0 </o-dd:version> <o-dd:uid> XYZ </o-dd:uid> </o-ex:context> </oma-dd:system> </o-ex:constraint> </oma-dd:export>	<mx:move/> <r:digitalResource> <r:nonSecureIndirect URI="ContentID"/> </r:digitalResource> <r:allConditions> <mx:destination> <r:keyHolder> <r:info> <version>1.0</version> <uid>XYZ</uid> </r:info> </r:keyHolder> </mx:destination> </r:allConditions>
<oma-dd:export oma-dd:mode="copy"> <o-ex:constraint> <oma-dd:system> <o-ex:context> <o-dd:version> 1.0 </o-dd:version> <o-dd:uid> XYZ </o-dd:uid> </o-ex:context> </oma-dd:system> </o-ex:constraint> </oma-dd:export>	<mx:adapt/> <r:digitalResource> <r:nonSecureIndirect URI="ContentID1"/> </r:digitalResource> <mx:prohibitedAttributeChanges> <set definition= "urn:mpeg:mpeg21:2003:01- RDD-NS:2346"/> <set definition= "urn:mpeg:mpeg21:2003:01- RDD-NS:2347"/> </mx:prohibitedAttributeChanges> <r:keyHolder> <version>1.0</version> <uid>XYZ</uid> </r:keyHolder>

Conditions

The following tables introduce different kinds of MPEG-21 REL conditions equivalent to the ones specified in OMA DRM REL.

One kind of conditions represented are time conditions. The <datetime> element represented in MPEG-21 REL with the <validityInterval> element specifies an interval of time within which a right can be exercised. The <interval> represented in MPEG-21 REL with the <validityIntervalDurationPattern> element specifies a period of time within which a right can be exercised. Finally, the <accumulated> element represented in MPEG-21 REL with the <validityTimeMetered> specifies the maximum period of metered usage time during which the rights can be exercised.

Table Time conditions model

OMA DRM REL v2.0	OMA-based MPEG-21 REL
<o-ex:constraint> <o-dd:datetime> <o-dd:start>... </o-dd:start> <o-dd:end>... </o-dd:end>	<r:allConditions> <r:validityInterval> <r:notBefore>...</r:notBefore> <r:notAfter>...</r:notAfter> </r:validityInterval> </r:allConditions>

</o-dd:datetime> </o-ex:constraint>	
<o-ex:constraint> <o-dd:interval> </o-dd:interval> </o-ex:constraint>	<r:allConditions> <sx:validityIntervalDurationPattern> <sx:duration>...</sx:duration> </sx:validityIntervalDurationPattern> </r:allConditions>
<o-ex:constraint> <o-dd:accumulated> PT10H </ o-dd:accumulated > </o-ex:constraint>	<r:allConditions> <sx:validityTimeMetered> <sx:duration>PT10H</sx:duration> </sx:validityTimeMetered> </r:allConditions>

The next table introduces the rest of MPEG-21 REL conditions considered in the mobile subset we are defining equivalent to the ones specified in OMA DRM REL. The <count> element represented in MPEG-21 REL with the <exerciseLimit> element specifies the number of allowed exercises. The <timed-count> element specify the number of times a permission may be granted over an asset or resource, with the addition of an optional timer attribute. This timer attribute specifies the number of seconds after which the count state can be reduced. As the timer attribute is not specified in MPEG-21 REL, we have defined the <exerciseLimitTime>, that consist of <count> and <duration> elements. The <individual> represented in MPEG-21 REL with the <keyHolder> element specifies the individual to which content is bound. The <system> represented in MPEG-21 REL with the <renderer> element specifies the target system to which DRM Content and Rights Objects can be exported.

Table Other conditions model

OMA DRM REL v2.0	OMA-based MPEG-21 REL
<o-ex:constraint> <o-dd:count > 1 </o-dd count> </o-ex:constraint>	<sx:exerciseLimit> <sx:count>1</sx:count> </sx:exerciseLimit>
<o-ex:constraint> <o-dd:timed-count timer=”30”>1 </o-dd:timed-count> </o-ex:constraint>	<r:otherinfo> <exerciseLimitTime> <sx:count>1</sx:count> <sx:duration>30 </ sx:duration> </exerciseLimit> </r :otherinfo>
	<r:grant licensePartId="Asset-1"> <r:allConditions> <sx:exerciseLimit> <sx:count>1</sx:count> </sx:exerciseLimit> </r:allConditions> </r:grant licensePartId="Asset-1"> <r:otherinfo> <grant licensePartIdRef="Asset-1"> <exerciseLimitDuration> 30 </exerciseLimitDuration> </grant> </r:otherinfo>
	<sx:exerciseLimit> <r:serviceReference licensePartIdRef="externalService"/> <sx:count>1</sx:count> </sx:exerciseLimit>
<o-ex:constraint>	<r:grant>

<o-dd:individual> <o-ex:context> <odd:uid> XYZ </odd:uid> </o-ex:context> </o-dd: individual> </o-ex:constraint>	<r:keyHolder> <r:info> <uid>XYZ</uid> </r:info> </r:keyHolder> </r:grant>
<o-ex:constraint> <oma-dd:system> <o-ex:context> <odd:uid> XYZ </odd:uid> </o-ex:context> </oma-dd system> </o-ex:constraint>	<mx:renderer> <r:keyHolder> <r:info> <uid>XYZ</uid> </r:info> </r:keyHolder> </mx:renderer>

7. Appendix

Supported File Formats and Codecs by "FFmpeg"

File Formats:

FFmpeg supports the following file formats through the libavformat library:

Supported File Format	Encoding	Decoding	Comments
MPEG audio	X	X	
MPEG1 systems	X	X	muxed audio and video
MPEG2 PS	X	X	also known as VOB file
MPEG2 TS		X	also known as DVB Transport Stream
ASF	X	X	
AVI	X	X	
WAV	X	X	
Macromedia Flash	X	X	Only embedded audio is decoded
FLV	X	X	Macromedia Flash video files
Real Audio and Video	X	X	
Raw AC3	X	X	
Raw MJPEG	X	X	
Raw MPEG video	X	X	
Raw PCM8/16 bits, mulaw/Alaw	X	X	
Raw CRI ADX audio	X	X	
SUN AU format	X	X	
NUT	X	X	NUT Open Container Format

Quicktime	X	X	
MPEG4	X	X	MPEG4 is a variant of Quicktime
Raw MPEG4 video	X	X	
DV	X	X	
4xm		X	4X Technologies format, used in some games
Playstation STR		X	
Id RoQ		X	used in Quake III, Jedi Knight 2, other computer games
Interplay MVE		X	format used in various Interplay computer games
WC3 Movie		X	multimedia format used in Origin's Wing Commander III computer game
Sega FILM/CPK		X	used in many Sega Saturn console games
Westwood Studios VQA/AUD		X	Multimedia formats used in Westwood Studios games
Id Cinematic (.cin)		X	Used in Quake II
FLIC format		X	.fli/.flc files
Sierra VMD		X	used in Sierra CD-ROM games
Sierra Online		X	.sol files used in Sierra Online games
Matroska		X	
Electronic Arts Multimedia		X	used in various EA games; files have extensions like WVE and UV2
Nullsoft Video (NSV) format		X	

X means that the encoding (resp. decoding) is supported.

CODECS

Video CODECS

Supported Codec	Encoding	Decoding	Comments
MPEG1 video	X	X	
MPEG2 video	X	X	
MPEG4	X	X	Also known as DIVX4/5
MSMPEG4 V1	X	X	
MSMPEG4 V2	X	X	
MSMPEG4 V3	X	X	Also known as DIVX3
WMV7	X	X	
WMV8	X	X	Not completely working
H.261	X	X	
H.263(+)	X	X	Also known as Real Video 1.0
H.264		X	
MJPEG	X	X	
Lossless MJPEG	X	X	
Apple MJPEG-B		X	

Sunplus MJPEG		X	fourcc: SP5X
DV	X	X	
Huff YUV	X	X	
FFmpeg Video 1	X	X	Experimental lossless codec (fourcc: FFV1)
FFmpeg Snow	X	X	Experimental wavelet codec (fourcc: SNOW)
Asus v1	X	X	fourcc: ASV1
Asus v2	X	X	fourcc: ASV2
Creative YUV		X	fourcc: CYUV
Sorenson Video 1	X	X	fourcc: SVQ1
Sorenson Video 3		X	fourcc: SVQ3
On2 VP3		X	still experimental
Theora		X	still experimental
Intel Indeo 3		X	only works on i386 right now
FLV	X	X	Sorenson H.263 used in Flash
ATI VCR1		X	fourcc: VCR1
ATI VCR2		X	fourcc: VCR2
Cirrus Logic AccuPak		X	fourcc: CLJR
4X Video		X	used in certain computer games
Sony Playstation MDEC		X	
Id RoQ		X	used in Quake III, Jedi Knight 2, other computer games
Xan/WC3		X	used in Wing Commander III .MVE files
Interplay Video		X	used in Interplay .MVE files
Apple Animation		X	fourcc: 'rle '
Apple Graphics		X	fourcc: 'smc '
Apple Video		X	fourcc: rpza
Apple QuickDraw		X	fourcc: qdrw
Cinepak		X	
Microsoft RLE		X	
Microsoft Video-1		X	
Westwood VQA		X	
Id Cinematic Video		X	used in Quake II
Planar RGB		X	fourcc: 8BPS
FLIC video		X	
Duck TrueMotion v1		X	fourcc: DUCK
VMD Video		X	used in Sierra VMD files
MSZH		X	Part of LCL
ZLIB	X	X	Part of LCL, encoder experimental
TechSmith Camtasia		X	fourcc: TSCC
IBM Ultimotion		X	fourcc: ULTI
Miro VideoXL		X	fourcc: VIXL

QPEG		X	fourccs: QPEG, Q1.0, Q1.1

X means that the encoding (resp. decoding) is supported.

Audio CODECS

Supported Codec	Encoding	Decoding	Comments
MPEG audio layer 2	IX	IX	
MPEG audio layer 1/3	IX	IX	MP3 encoding is supported through the external library LAME
AC3	IX	IX	liba52 is used internally for decoding
Vorbis	X	X	supported through the external library libvorbis
WMA V1/V2		X	
AAC	X	X	supported through the external library libfaac/libfaad
Microsoft ADPCM	X	X	
MS IMA ADPCM	X	X	
QT IMA ADPCM		X	
4X IMA ADPCM		X	
G.726 ADPCM	X	X	
Duck DK3 IMA ADPCM		X	used in some Sega Saturn console games
Duck DK4 IMA ADPCM		X	used in some Sega Saturn console games
Westwood Studios IMA ADPCM		X	used in Westwood Studios games like Command and Conquer
SMJPEG IMA ADPCM		X	used in certain Loki game ports
CD-ROM XA ADPCM		X	
CRI ADX ADPCM	X	X	used in Sega Dreamcast games
Electronic Arts ADPCM		X	used in various EA titles
Creative ADPCM		X	
RA144		X	Real 14400 bit/s codec
RA288		X	Real 28800 bit/s codec
RADnet	X	IX	Real lowbitrate AC3 codec, liba52 is used for decoding
AMR-NB	X	X	supported through an external library
AMR-WB	X	X	supported through an external library
DV audio		X	
Id RoQ DPCM		X	used in Quake III, Jedi Knight 2, other computer games
Interplay MVE DPCM		X	used in various Interplay computer games
Xan DPCM		X	used in Origin's Wing Commander IV AVI files
Sierra Online DPCM		X	used in Sierra Online game audio files
Apple MACE 3		X	
Apple MACE 6		X	
FLAC		X	
FFmpeg Sonic	X	X	Experimental lossy/lossless codec

Windows Platform Installation

- Install the current versions of MSYS and MinGW from http://www.mingw.org/. You can find detailed installation instructions in the download section and the FAQ.
- If you want to test the FFmpeg Simple Media Player, also download the MinGW development library of SDL 1.2.x (`SDL-devel-1.2.x-mingw32.tar.gz') from http://www.libsdl.org. Unpack it in a temporary place, and unpack the archive `i386-mingw32msvc.tar.gz' in the MinGW tool directory. Edit the `sdl-config' script so that it gives the correct SDL directory when invoked.
- Extract the current version of FFmpeg (the latest release version or the current CVS snapshot whichever is recommended).
- Start the MSYS shell (file `msys.bat').
- Change to the FFMPEG directory and follow the instructions of how to compile ffmpeg (file `INSTALL'). Usually, launching `./configure' and `make' suffices. If you have problems using SDL, verify that `sdl-config' can be launched from the MSYS command line.
- You can install FFmpeg in `Program Files/FFmpeg' by typing `make install'. Don't forget to copy `SDL.dll' at the place you launch `ffplay'.

Notes:

- The target `make wininstaller' can be used to create a Nullsoft based Windows installer for FFmpeg and FFplay. `SDL.dll' must be copied in the ffmpeg directory in order to build the installer.
- By using ./configure --enable-shared when configuring ffmpeg, you can build `avcodec.dll' and `avformat.dll'. With make install you install the FFmpeg DLLs and the associated headers in `Program Files/FFmpeg'.

Visual C++ compatibility: if you used ./configure --enable-shared when configuring FFmpeg, then FFmpeg tries to use the Microsoft Visual C++ lib tool to build avcodec.lib and avformat.lib. With these libraries, you can link your Visual C++ code directly with the FFmpeg DLLs.

Supported File Formats by FFmpeg Through Libavcodec Library

Supported Codec	**Encoding**	**Decoding**	**Comments**
MPEG Audio Layer 2	IX	IX	
MPEG Audio Layer 1/3	IX	IX	MP3 encoding is supported through the external library LAME
AC3	IX	IX	Liba52 (GPL) is used internally for decoding
Vorbis	X	X	Supported through the external library libvorbis
WMA V1/V2		X	
AAC	X	X	Supported through the external library libfaac/libfaad
Microsoft ADPCM	X	X	
MS IMA ADPCM	X	X	
QT IMA ADPCM		X	
4X IMA ADPCM		X	
G. 726 ADPCM	X	X	
Duck DK3 IMA ADPCM		X	Used in some Sega Saturn console games
Duck DK4 IMA ADPCM		X	Used in some Sega Saturn console games
Westwood Studios IMA ADPCM		X	Used in Westwood Studios games like Command and Conquer

SMJPEG IMA ADPCM		X	Used in certain Loki game ports
CD-ROM XA ADPCM		X	
CRI ADX ADPCM	X	X	Used in Sega Dreamcast games
Electronic Arts ADPCM		X	Used in various EA titles
Creative ADPCM		X	
RA 144		X	Real 14400 bit/s codec
RA 288		X	Real 28800 bit/s codec
RADNET	X	IX	Real lowbitrate AC3 codec, liba52 (GPL) is used for decoding
AMR-NB	X	X	Supported through an external library
AMR-WB	X	X	Supported through an external library
DV audio		X	
Id RoQ DPCM		X	Used in Quake III, Jedi Knight II, other computer games
Interplay MVE DPCM		X	Used in various Interplay computer games
Xan DPCM		X	Used in Origin's Wing Commander IV AVI files
Sierra Online DPCM		X	Used in Sierra Online game audio files
Apple MACE 3		X	
Apple MACE 6		X	
FLAC		X	
FFmpeg Sonic	X	X	Experimental lossly/lossless codec

Supported Audio File Formats by Libnsdfile Library

	Micro-soft WAV	SGI / Apple AIFF / AIFC	Sun / DEC / NeXT AU / SND	Header-less RAW	Paris Audio File PAF	Commo-dore Amiga IFF / SVX	Sphere Nist WAV	IRCAM SF	Creative VOC	Sound forge W64	GNU Octave 2.0 MAT4	GNU Octave 2.1 MAT5	Portable Voice Format PVF	Fasttracker 2 XI	HMM Tool Kit HTK
Unsigned 8 bit PCM	R/W	R/W		R/W					R/W	R/W		R/W			
Signed 8 bit PCM		R/W	R/W	R/W	R/W	R/W	R/W						R/W		
Signed 16 bit PCM	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W		R/W
Signed 24 bit PCM	R/W	R/W	R/W	R/W	R/W		R/W	R/W		R/W					
Signed 32 bit PCM	R/W	R/W	R/W	R/W			R/W	R/W		R/W	R/W	R/W	R/W		
32 bit float	R/W	R/W	R/W	R/W				R/W		R/W	R/W	R/W			
64 bit	R/W	R/W	R/W	R/W						R/W	R/W	R/W			

double														R/W	
u-law encoding	R/W	R/W	R/W	R/W			R/W	R/W	R/W	R/W					
A-law encoding	R/W	R/W	R/W	R/W			R/W	R/W	R/W	R/W					
IMA ADPCM	R/W									R/W					
MS ADPCM	R/W									R/W					
GSM 6.10	R/W	R/W		R/W						R/W					
G721 ADPCM 32kbps			R/W												
G723 ADPCM 24kbps			R/W												
G723 ADPCM 40kbps			R/W												
12 bit DWVW		R/W		R/W											
16 bit DWVW		R/W		R/W											
24 bit DWVW		R/W		R/W											
Ok Dialogic ADPCM				R/W											
8 bit DPCM														R/W	
16 bit DPCM														R/W	

CC/PP Profile for Device "Sony Ericsson T39"

```
<?xml version="1.0" ?>
- <RDF xmlns="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:prf="http://www.wapforum.org/UAPROF/ccppschema-20000405#">
- <rdf:Description ID="Profile">
- <prf:component>
- <rdf:Description ID="HardwarePlatform">
  <prf:ScreenSize>101x54</prf:ScreenSize>
  <prf:Model>T39m</prf:Model>
- <prf:InputCharSet>
- <rdf:Bag>
  <rdf:li>ISO-8859-1</rdf:li>
```

```
<rdf:li>US-ASCII</rdf:li>
<rdf:li>UTF-8</rdf:li>
<rdf:li>ISO-10646-UCS-2</rdf:li>
</rdf:Bag>
</prf:InputCharSet>
<prf:ScreenSizeChar>15x5</prf:ScreenSizeChar>
<prf:BitsPerPixel>2</prf:BitsPerPixel>
<prf:ColorCapable>No</prf:ColorCapable>
<prf:TextInputCapable>Yes</prf:TextInputCapable>
<prf:ImageCapable>Yes</prf:ImageCapable>
<prf:Keyboard>PhoneKeypad</prf:Keyboard>
<prf:NumberOfSoftKeys>0</prf:NumberOfSoftKeys>
<prf:Vendor>Ericsson Mobile Communications AB</prf:Vendor>
- <prf:OutputCharSet>
- <rdf:Bag>
<rdf:li>ISO-8859-1</rdf:li>
<rdf:li>US-ASCII</rdf:li>
<rdf:li>UTF-8</rdf:li>
<rdf:li>ISO-10646-UCS-2</rdf:li>
</rdf:Bag>
</prf:OutputCharSet>
<prf:SoundOutputCapable>Yes</prf:SoundOutputCapable>
<prf:StandardFontProportional>Yes</prf:StandardFontProportional>
<prf:PixelsAspectRatio>1x0,9</prf:PixelsAspectRatio>
</rdf:Description>
</prf:component>
- <prf:component>
- <rdf:Description ID="SoftwarePlatform">
<prf:AcceptDownloadableSoftware>No</prf:AcceptDownloadableSoftware>
</rdf:Description>
</prf:component>
- <prf:component>
- <rdf:Description ID="NetworkCharacteristics">
<prf:SecuritySupport>WTLS class 1/2/3/signText</prf:SecuritySupport>
- <prf:SupportedBearers>
- <rdf:Bag>
<rdf:li>TwoWaySMS</rdf:li>
<rdf:li>CSD</rdf:li>
<rdf:li>GPRS</rdf:li>
</rdf:Bag>
</prf:SupportedBearers>
</rdf:Description>
</prf:component>
- <prf:component>
- <rdf:Description ID="BrowserUA">
<prf:BrowserName>Ericsson</prf:BrowserName>
- <prf:CcppAccept>
- <rdf:Bag>
<rdf:li>application/vnd.wap.wmlc</rdf:li>
<rdf:li>application/vnd.wap.wbxml</rdf:li>
<rdf:li>application/vnd.wap.wmlscriptc</rdf:li>
<rdf:li>application/vnd.wap.multipart.mixed</rdf:li>
<rdf:li>text/x-vCard</rdf:li>
<rdf:li>text/x-vCalendar</rdf:li>
<rdf:li>text/x-vMel</rdf:li>
<rdf:li>text/x-eMelody</rdf:li>
```

```
<rdf:li>image/vnd.wap.wbmp</rdf:li>
<rdf:li>image/gif</rdf:li>
<rdf:li>application/vnd.wap.wtls-ca-certificate</rdf:li>
<rdf:li>application/vnd.wap.sic</rdf:li>
<rdf:li>application/vnd.wap.slc</rdf:li>
<rdf:li>application/vnd.wap.coc</rdf:li>
<rdf:li>application/vnd.wap.sia</rdf:li>
</rdf:Bag>
</prf:CcppAccept>
- <prf:CcppAccept-Charset>
- <rdf:Bag>
<rdf:li>US-ASCII</rdf:li>
<rdf:li>ISO-8859-1</rdf:li>
<rdf:li>UTF-8</rdf:li>
<rdf:li>ISO-10646-UCS-2</rdf:li>
</rdf:Bag>
</prf:CcppAccept-Charset>
- <prf:CcppAccept-Encoding>
- <rdf:Bag>
<rdf:li>base64</rdf:li>
</rdf:Bag>
</prf:CcppAccept-Encoding>
<prf:FramesCapable>No</prf:FramesCapable>
<prf:TablesCapable>Yes</prf:TablesCapable>
</rdf:Description>
</prf:component>
- <prf:component>
- <rdf:Description ID="WapCharacteristics">
<prf:WapDeviceClass>C</prf:WapDeviceClass>
<prf:WapPushMsgSize>3000</prf:WapPushMsgSize>
<prf:WapVersion>1.2.1/June 2000</prf:WapVersion>
- <prf:WmlVersion>
- <rdf:Bag>
<rdf:li>1.2.1/June 2000</rdf:li>
<rdf:li>1.1</rdf:li>
</rdf:Bag>
</prf:WmlVersion>
<prf:WmlDeckSize>3000</prf:WmlDeckSize>
- <prf:WmlScriptVersion>
- <rdf:Bag>
<rdf:li>1.2.1/June 2000</rdf:li>
<rdf:li>1.1</rdf:li>
</rdf:Bag>
</prf:WmlScriptVersion>
- <prf:WmlScriptLibraries>
- <rdf:Bag>
<rdf:li>Lang</rdf:li>
<rdf:li>Float</rdf:li>
<rdf:li>String</rdf:li>
<rdf:li>URL</rdf:li>
<rdf:li>WMLBrowser</rdf:li>
<rdf:li>Dialogs</rdf:li>
</rdf:Bag>
</prf:WmlScriptLibraries>
- <prf:WtaiLibraries>
- <rdf:Bag>
```

```
<rdf:li>WTA.Public.makeCall</rdf:li>
<rdf:li>WTA.Public.sendDTMF</rdf:li>
<rdf:li>WTA.Public.addPBEntry</rdf:li>
</rdf:Bag>
</prf:WtaiLibraries>
</rdf:Description>
</prf:component>
</rdf:Description>
</RDF>
```

References

1. http://www.imagemagick.org/.
2. http://www.mega-nerd.com/libsndfile/
3. http://sky.prohosting.com/oparviai/soundtouch/.
4. http://www.mobile-phones-uk.org.uk/
5. http://www.pdatoday.com/comments/2934_0_1_0_C/
6. http://www.w3.org/Mobile/CCPP/
7. http://www.hpl.hp.com/personal/marbut/DeliUserGuideWEB.htm.
8. http://arbor.ee.ntu.edu.tw/~jlhuang/publications/IMMCN03_ICC.pdf
9. http://wurfl.sourceforge.net/
10. http://sourceforge.net/projects/delicon/
11. http://cocoon.apache.org/2.1/developing/deli.html
12. http://www.w3.org/TR/NOTE-CCPP/
13. http://www.hpl.hp.com/personal/marbut/DeliUserGuideWEB.htm
14. [1] ISO/IEC, ISO/IEC IS 21000-5 – Rights Expression Language.
15. [2] XrML, http:// www.xrml.org/.
16. [3] Open Digital Rights Language (ODRL). http://odrl.net.
17. [4] OMA DRM Rights Expression Language, OMA-Download-DRMREL-V2_0-20041210-C. 10 December 2004.
18. [5] XML Encryption Syntax and Processing, W3C Candidate Recommendation 10 December 2002, http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/
19. [6] XML Signature Syntax and Processing, W3C Recommendation 12 February 2002, http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/
20. [7] Prados, J., Rodríguez, E., Delgado, J., Profiles for interoperability between MPEG-21 REL and OMA DRM, CEC 2005, Munich (Germany), 19 – 22 July 2005, ISBN 0-7695-2277-7.
21. [8] Delgado, J., Prados, J., Rodríguez, E., Interoperability between MPEG 21 REL and OMA DRM: A profile?, ISO/IEC JTC1/SC29/WG11 MPEG2005/M11580, January 2005.
22. [9] Delgado, J., Prados, J., Rodríguez, E., Interoperability between different Rights Expression Languages and Protection Mechanisms, AXMEDIS 2005, Florence (Italy), 30 November – 2 December 2005, To be published.
23. [10] Delgado, J., Prados, J., Rodríguez, E., A subset of MPEG-21 REL for interoperability with OMA DRM v2.0, ISO/IEC JTC 1/SC 29/WG 11/ M11893, April 2005.