Automating Production of Cross Media Content for Multi-channel Distribution

www.AXMEDIS.org

DE4.7.1.3
Content Distribution toward mobiles, update of DE4.7.1.2

Version: 1.0
Date: 03/10/2007
Responsible: UR (revised and approved by coordinator)

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: DE4.7.1.3 Contractual Date of Delivery: M36 Actual Date of Delivery: 03/10/2007 Work-Package contributing to the Deliverable: WP4.7 Task contributing to the Deliverable: WP4.7 Nature of the Deliverable: Report and Prototype Author(s): UR, UPC, DSI, DIPITA, EPFL, ILABS, FHGIGD
Abstract: This report updates and concludes the work performed by the Consortium partners based on the 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 their interoperability. The report opens with the Executive Summary and Scope section to clarify the focus of the report. There follows an analysis of content distribution towards mobiles giving an overview of the relevant issues such as general plug and play, transcoding, media adaptation and profiling. In doing this we have also provided scripts examples for transcoding content for mobiles (axcp rules for: resizing images, converting an image

from colour to black and white, converting IMS packages into AXMEDIS objects). Thereafter the aspects of ILABS content sources integration with the AXMEDIS model are described.

An update of device state-of-the-art is then provided along with trends in digital media adaptation and transcoding, delivery context representation for digital item adaptation, DRM support and interoperability (especially in relation to REL profiles); especially focusing on work already done in supporting interoperability between MPEG-21 and OMA).

Following an appropriate state-of-the-art review of the latest reported resarch by resarchers working on various aspects of DIA, the work done in digital item adaptation and MPEG 21-Conformant DIA profiling management is then reported focusing on DIA profiling adaptation decision engine, ringtone, audio, image, document, and video adaptation.

Keyword List: digital media distribution, profiling, transcoding, mobile, DRM, ringtones, proxies, MPEG-21, DIA algorithms, REL, optimisation, interoperability OMA MPEG-21, back office of AXMEDIS with OMA, MPEG21-Conformant Dynamic Profiling Adaptation Decision Engine.

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 (ALL)

The purpose of this deliverable is to provide an update to the DE4.7.1 "Distribution to Mobile" on the relevant state-of-the art and the RTD work carried out by the Partners in order to design and implement the required components for dynamic adaptation and provision of media to mobile clients to be integrated for the AXMEDIS Demonstrator. As such this report will update on the re-aligned transcoding plug-in and profiling based on MPEG-21 that will work integratively with the AXMEDIS JS Script Engine within the AXCP to deliver adaptive multimedia distribution to mobile clients as well as to provide the profiling knowledge management basis for the DIA Decision Taking Engine to be designed and implemented as a C++ layer acting above the level of the media adaptation JS scripts so as to allow the invocation of the appropriate media selection and adaptation rules under variable states of run-time profiling knowledge of the client side delivery context parameters of preference (re user, device, channel/network and other general conditions).

The completed work report here includes the following aspects:

1. Overview of the relevant issues such as general plug and play, transcoding, media adaptation and profiling; scripts examples for transcoding content for mobiles
2. Aspects of ILABS content sources integration with the AXMEDIS mode
3. An update of device state-of-the-art and trends in digital media adaptation and transcoding, delivery context representation for digital item adaptation, DRM support and interoperability (especially in relation to REL profiles)
4. Work done in supporting interoperability between MPEG-21 and OMA including:
    - Studying the adaptation of licenses from MPEG-21 REL to ODLR OMA. (FUFP)
    - Production of OMA packages via AXMEDIS CP scripts (DSI)
    - Automatation of license adaptation from MPEG-21 REL to OMA (DSI)
    - Exploitation of the OMA model and distribution and trial of OMA distribution channel (DSI)
5. State-of-the-art review of the latest reported resarch by resarchers working on various aspects of DIA Decision Engine
6. Work done on the MPEG 21-Conformant DIA profiling management and Adaptation Decision Engine

2 Work to be done I nthe period

T4.7.2 AXMEDIS Architecture for Transcoding– (started at M13) – UR responsible – planned to be completed for M36.
The AXCP platform can be used to implement an efficient Transcoding architecture and platform supporting adaptation of protected content and thus DRM as well. These features have been integrated into the AXCP java script. It has to be capable to load and save data type modelling Device Capabilities (may be in CCPP and/or MPEG-21 DIA profile), context of the device and usage, and the user preferences (the preferred formats of the user with respect to its devices and content and situations of usage, etc.).
The transcoding platform has to be capable of using suitable formatting styles for the production of content adapted off-line and on demand. What is happening now on the mobile is going to happen in terms of needs of massive production of content for distribution also on other media distribution: for instance for images and video, and thus for all AXMEDIS objects disregarding their complexity.
The AXCP Transcoding Platform developed provides the content format conversion functionality. Its modularity allows the plug-and-play installation and update of transcoding components. It is

efficient and designed for high performance. To enable smart mobile device capabilities, user profiling, new standards like MPEG-21 DIA, UAProf based on the new multi model RDF will be adopted and embedded inside the transcoding platform, etc.
UR will work on the 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.
For the older generation of mobile phones the memory constraints on the device and other design constraints were responsible for different formatting being chosen for ringtones than was available for other audio files but with the advent of the new generation of mobile phones which do not suffer from memory constraints to the same extent as previously the formats deployed for ringtones are moving towards mainstream and upcoming audio formats such as MP3, WAV, 3GP for which we have already made available appropriate transcoding within AXMEDIS.
UR will support the Distributor Partners for the AXMEDIS Transcoding Framework. We will implement any other upcoming audio formats that will emerge as required by the users, i.e. Distributor Partners, in order to ensure the full upward compatibility of the Axmedis Transcoding Framework as far as the audio transcoding is concerned

In order to ensure full backward compatibility of the AXMEDIS Framework for the older generation of devices it may be advisable to explore with the user partners concerned any other older formats which may need to be included such as Midi etc in order to ensure this backward compatibility.

- M30: Completion of Transcoding Plug-in & Profiling Management including processing of profiles to direct the transcoding activities.
- M36: Further changes and debugging Transcoding Plug-in & Profiling Management following tests
- M36: performance analysis of AXCP as transcoding platform

T4.7.3 DRM support for mobiles and interoperability – UPC responsible -- planned to be completed for M36.
Some initiatives exist from standardisation bodies and associations to provide DRM for mobile platforms, like MPEG-21 profiles and OMA DRM, which uses ODRL. In this period, we will continue the study of existing initiatives, actively participating in the definition of profiles for the mobile environment. Implementation of the DRM profiles for mobiles and initial integration of the implemented prototypes and tools with existing mobile platforms is foreseen. The work done inside this task includes the description of MPEG-21 DIA expressions for the authorisation of operations in the mobile environment, the implementation of local authorisation mechanisms for mobile devices and the use of a simplified version of event reporting. Local storage of information into mobile devices will be also evaluated, as the security mechanisms available are not the same as for other platforms. In collaboration with ILABS, integration of DRM aspects for IMS/SCORM for mobile distribution of educational content, aspects related to the authoring and to the client sides.

- M30: Study of tools for mobiles: DMP, OMA DRM, Domain support
- M32: Modification of the authorization algorithm to support mobile environment restrictions expressed with MPEG-21 DIA
- M36: Integration of protection tools implemented in WP4.3 and WP4.5 for supporting mobile distribution channel: Event reporting for mobiles, License request, Basic authorisation to be carried out in the mobile device

T4.7.4 AXMEDIS Mobile Player – TISCALI responsible in collaboration with DSI and EXITECH, and UPC (planned to be completed for M36),.
The idea is to explore the possibility of developing a pure java AXMEDIS player for mobiles. Another alternative is the development of C++ version for Symbian and/or Windows Mobile 2005.

Major requirements could be: reduced AXMEDIS model (the hierarchical level could be included or not), simpler protection processor (only one level of processing), SMIL player or a something very simple for the menu', video and audio player may be directly derived from those available on the device, access to the internet for the authentication, simple PMS client, protected communication for the protection information. Possibility of integrating this with AXMEDIS objects streamed on MPEG-2. EXITECH will work on stream server and client.

- M30: specification of the mobile player, first experiments;
- M36: first release of the AXMEDIS player for mobiles, independent client-server demonstration of the streaming;
- M36: final release of the AXMEDIS player for mobile with stream client and of the server streamer;

3 Analysis of Content Distribution Towards Mobiles (ILABS, UR)

Today's telecommunication market is extremely interested in content-oriented services as a source for potential revenue. The AXMEDIS platform provides a comprehensive business perspective for both content providers and telecom service providers. A large variety of technologies and protocols are presently in use throughout the world, many of which are proprietary and do not adhere to any standard. Consequently, telecom operators encounter major challenges when attempting to deliver content-based value-added services. The AXMEDIS platform effectively addresses these challenges, by providing content owners with user-friendly tools in a familiar environment, and enables them to offer their content to a large number of users while ensuring content protection.

In particular the capability of massively and automatically operate onto data (including not only simple resources, but also already aggregated objects) using templates and profiles open the possibility to exploit huge repositories of content presently available but not yet used in this domain. Just think of tourist information currently available in websites and CD/DVD-ROM based archives. With the adoption of AXMEDIS tools it is already possible to transform these sources into archives of protected resources that then (thanks to the profiling and adaptation capabilities) can be consistently and profitably delivered to end users.
By increasing the number of accessible and standardised types of content, the AXMEDIS platform can assist operators and content providers in realising the following goals:

- Increase end user satisfaction;
- Increase content offering to a broad range of customers supporting most diffused type of device (by streamlining content delivery to mobile handsets);
- Open new business opportunities to content providers and telecom operators.

Content distribution for mobile devices is performed using a "**content push**" method upon end user request (referred to as the "end user" or "subscriber" as the underlaying business model will be mainly centred on subscription mechanisms). In order to reach the desired content, the subscriber navigates through a set of menus (including the element referred to as "catalogue" that comprises more than simple menu choices). All formats supported by the AXMEDIS platform are stored in the AXMEDIS Platform, and supplied upon request. The most relevant and usable encodings will be identified and developed (or integrated and made accessible) within the AXMEDIS platform. The platform will be capable of accepting any custom codecs and automatically supporting newly added formats whenever the necessary codec plug-in will be available. To ensure content protection the DRM should be used to define the type of protection and licensing methods.

3.1 Overview of Content Distributions Towards Mobiles (ILABS)

A content that has been designed for fruition on a PC when presented onto a PDA/Mobile may undergo a set of changes. These changes are decided and performed at the Factory side of the Mobile distribution. The first, and probably the more apparent is that content is highly probable to be scaled down to a smaller

dimension, as even the simplest of all apparent changes occurring (that is converting from landscape to portrait or vice versa) implies a change in:

- areas sizes (to retain the same content) and
- location of some specific content.

Unless the PDA/Mobile is able to automatically flip the screen (that is presenting data in landscape rather than portrait) such a transformation has to be done prior to delivery. Now this is not the only point as passing from PCs to PDA/Mobile, the change in display size implies:

- Lower display area
- Lower computational power
- Lower image quality
- Smaller colour space
- Smaller fonts
- Worse anti-aliasing
- Scroll-bars / page tabs
- Different content distribution

This latter point may sound strange, but given the fact that the most diffused browser for Windows CE / Windows Mobile (prior to version 6) equipped PDA/Mobiles does not support CSS files and formatted HTML page will lose most of what foreseen at content design and production time.
This, in some contexts, where content quality or publisher / distributor branding is highly related to content quality may be a major limiting factor that may lead to the voluntary production of device customized content. Moreover at rendering time (either at server or client side) of a content has to be taken into account that passing from a single page to a set of pages, links between portions of the same page (anchors) are turned into links between pages (URLs), including the return to “top of page” as the way content is browsed changes from:

- Scrolling to
- Paging

The same applies vice-versa. This process, if performed manually is long, time consuming and potentially error prone as it could lead to programming problems like “missing links”… while if performed automatically requires a quite complex environment.

In the Mobile Factory all this will be made possible thanks to the combination of the solution initially introduced in our editing environment with the procedures and tools of AXMEDIS. In more detail this will be achieved as follows thanks to sets of purposely-defined rules:

- Content component identification and description,
- Content composition,
- Content formatting,
- Content adaptation and eventually
- Content distribution.

The AXCP is the way that allows adapting AXMEDIS contents on the basis of users’ devices characteristics. Given a device profile, it is possible to have a rule that “reads” the profile, extracts some relevant features like colour capability, screen size, screen resolution, sound capability and so on, and depending on these features can modify the AXMEDIS content in order to better fit to the user’s device capabilities. Basic steps of the adaptation mechanism are:

- get one or more device profile(s)
- get one or more content(s)
- for each device profile, and for each content:
    - read device profile and extract some relevant features;
    - depending on device features, modify the content;
    - store the content locally

The adaptation phase implies the use of AXMEDIS Plugins for image, audio, multimedia, text and video adaptation. Also on demand adaptation is foreseen through Query on Demand AXMEDIS module. Later contents are distributed through AXMEDIS PnP Tool.

3.1.1 General Plug and Play Transcoding and Media Adaptation

The AXDMA Adaptation must provide for efficient multimedia transcoding, any format to any format (including audio e.g. ringtones, image, text) to suit the standard characteristics of users' device(s) for distribution on-demand to existing mobile devices and new generation of mobiles and PDAs.
The potential volume of downloads by mobile devices requires an efficient transcoding architecture, thus the transcoding must be kept as efficient as possible i.e. it must minimise the complexity in the adaptation process. The requirement for optimising quality, efficiency and complexity of the transcoding process for distribution to mobiles will be increasingly important for video distribution where it could be even more critical since the videos are typically large files and more complex to for transcoding.

In order to complete the subscriber's request and deliver the requested content to the mobile device the requested files should be available either in the distributor CMS or in its AXMEDIS database. Therefore, in the latter case it is necessary for the CMS to be able to generate the requested object automatically from one (or more) source-files exploiting its interconnection with local AXMEDIS database and all other framework tools necessary for the task completion.

The following scenario illustrates typical Transcoding requirements. One WAV file has been uploaded into the system. An end user may need a version of this file in a format suitable for use on their own device. Moreover to play this file as a sample it should be clipped (to 30 seconds, for example) and converted to the low-quality WMA similar format, while to sell this file it should be in high-quality WMA similar format. In order to be acquired and played by the end-user system it should be therefore fully converted into one of the device supported formats. Summing up, for sample playing it should be clipped (e.g. to 10 seconds, for example), or supplied in full length, depending on the subscriber requested operation (pre-view or purchase).
It is taken for granted that among available possible purchase modes there is the infinite repetition bound to a specific device. This latter has to be identified along with its user in an unequivocal manner (for example [[phone-no + SIM] + IMEI] where parenthesis represent the binding hierarchy) and modifiable under explicit request to service provider following a change in device (new system of the same or different model when possible) or change in SIM due to any valid reason (loss, subtraction, obsolescence…).
In this scenario, we see that even simple interfaces will require several different file format instances at least. In order to service additional interfaces and devices, the number of required files quickly increases. The content manager or content administrator cannot generate these files manually, but the system can easily produce all of them using only the source files and the appropriate codecs for the needed conversion processes.

3.2 Profiling (UR)

The AXMEDIS platform supports cross-channel distribution, including that involving Mobiles, Satellite, Internet and Kiosks, for distribution of content in both push and pull (on-demand) modes. Each of the distribution channels in AXMEDIS are required to perform any necessary structuring and resolution of various parameters representing the user's device characteristics and additional user's preferences. One of the most widespread problems faced by these distribution channels is the issue of compatibility, support of various formats, and interoperability between different devices. This problem is prevalent for almost all types of devices irrespective of manufacturer (e.g. even the same manufacturer may produce two incompatible phones; and the incompatibility problem just multiplies given that target devices in use can come from multiple manufacturers). Hence the AXMEDIS Distribution channels must enable dynamic access to, structuring, update, storage, and integration, by way of AXMEDIS Content Processing Engine (AXCP), of personal, device and delivery context profile element values. Personalisation requirements can be seen in practice to demand a capability for profiling management, coupled with a requirement for dynamic collection and evaluation of personal, default and closure values of profile elements depending on the profile element values availability - Profiling Resolution.

Imagine a scenario in which a user wishes to download a polyphonic ring tone to his/her device. They connect to the system to browse the available content, and potentially listen to the top-level menu. They select a service: Dedications, Fun Greetings, Icons, Pictures, Ringtones, Polyphonic Ringtones, etc and then a content catalogue. Most systems present a hierarchy of content, starting, for example, with the most popular types such as pop music, folk music, classical music, ethnic music, etc. User then selects a top-level category and may have to access a lower level or two, until they can finally select their choice content. Next they must specify the mobile device to which they are uploading or sending the content. If the operator's network supports more than one type of device (as is usually the case), then either the system must select the type of device, assume a default, or ask the user to select the device from a list. Without this vital piece of information re the target device or with incorrect information, all the effort of navigating through the menus and selecting content will be pointless. The AXMEDIS platform can empower this process by storing personalised information in the database, which can be automatically retrieved. The information could include the history of the user, his preferences, and interest groups, and many other properties, as is the case with Recommender Systems. Such information can be used to modify the menu flow in order to ensure that the user's navigation and product selection is more optimised. The platform can promote content to the users and/or suggest a narrow cross-section of content to accelerate navigation and enhance choice making based on the users' history and that of other users from the same interest groups.

The AXMEDIS-compliant media objects are to be distribute-able globally over heterogeneous networks and dispatch-able to various kinds of terminals. Moreover, the people who will ultimately consume and interact with the content can all have different behaviours and preferences. Consequently, digital items should be adapted to fit their particular usage environment. For example mobile devices typically have limited terminal capabilities compared to larger devices such as PCs. Most mobile phones cannot display video while they are able to play audio files in some specific format. In such context, one should only send audio information to the device. Such adaptations are to be performed on the server side before the content is dispatched to the device. Transformations aimed at economising in network bandwidth utilisation must be performed on the server side as well. Once content is received in a form that is supported by the mobile device, a user may wish to modify it according to their own preferences. This latter adaptation is to be performed directly on the mobile side since different users with different preferences may share the device or the same user may own several devices and be using different devices depending on their usage context such as location, content type etc. Following the same idea, the device should be able to adapt the content to the environment characteristics, which are variable by nature. The seventh part of ISO/IEC 21000 (MPEG-21) specifies tools for the adaptation of Digital Items. More specifically, it proposes a set of normalised tools describing the usage environment of a digital item to command adaptation tools. According to such precedence, AXMEDIS-compliant devices should support content adaptation conformant with a number of MPEG-21 usage environment tools.

In order to fulfil these dynamic media adaptation requirements, we have envisaged the requirements for five different kinds of profiles elements derived from MPEG-21 DIA, which when deployed in combination enable enhanced personalised services. These profiles can be broadly categorised as follows:

1. **User Profiles**: This profile type represents the salient personal information about the user and his preferences, e.g. presentation preferences, auditory or visual impairment, etc.
2. **Device Profiles**: This profile represents the salient information about the user's device(s) including both hardware and software specific information, e.g. codecs, formats, I/O, etc.
3. **Network Profiles**: This profile represents the salient information about the distribution channel that the service provider must use for the delivery of the content, e.g. QoS related elements, bandwidth, error rate, security etc.
4. **Context Profiles**: This profile represents the salient delivery context information about the user and his usage environment, e.g. illumination characteristics, etc.
5. **Business User Profiles**: This profile is the Distributor's or prosumer's (producer-consumer) profile to support B2B media transaction scenarios, e.g. licenses acquired, etc.

The elements for Context Profile are covered in the Users Profile and Network Profile, while the Business User Profile can be seen as subsume-able within the User Profile in the generalised sense of the User which subsumes all types of User including users who are both producers and consumers (prosumers) of digital

media as is increasingly the case i.e. User Profiles can be extended to include the twin profile roles (procurer and distributor) required to support the global prosumer, and thus the B2B, e-media market.

3.2.1 AxmedisUser

The users profile captures and stores information related to the users, the usage environment and the user preferences. This profile type is managed at the personalisation server end and includes at least minimal details about the user to identify his personal preferences in respect of the services being offered.

Figure 2.1: Schema for AXMEDIS User Profile

3.2.2 AxmedisTerminal

This device profile type captures information related to user's devices, that the service provider must take into account in delivering the contents. Generally the minimal set required comprises those that affect the proper utilisation of the contents on the device e.g. screen resolution for images. This profile type comprises the most vital information for the content providers to transcode the content to match the content to device characteristics to enhance the usage of the content.

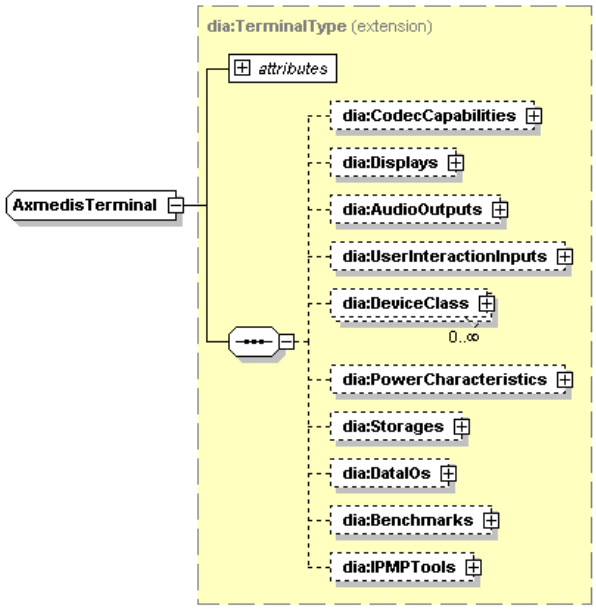

Figure 2.2: Schema for AXMEDIS Terminal Profile

3.2.3 AxmedisNetwork

The Network Profile type captures bandwidth utilisation and QoS related information for the carrier to use to optimize delivery of the content to the user's device. It is necessary for the distributors to know various characteristics of the Network so as to attempt to provide the promised QoS. The information contained in this profile can also be used for transcoding of the contents to better utilise both network and device capabilities.

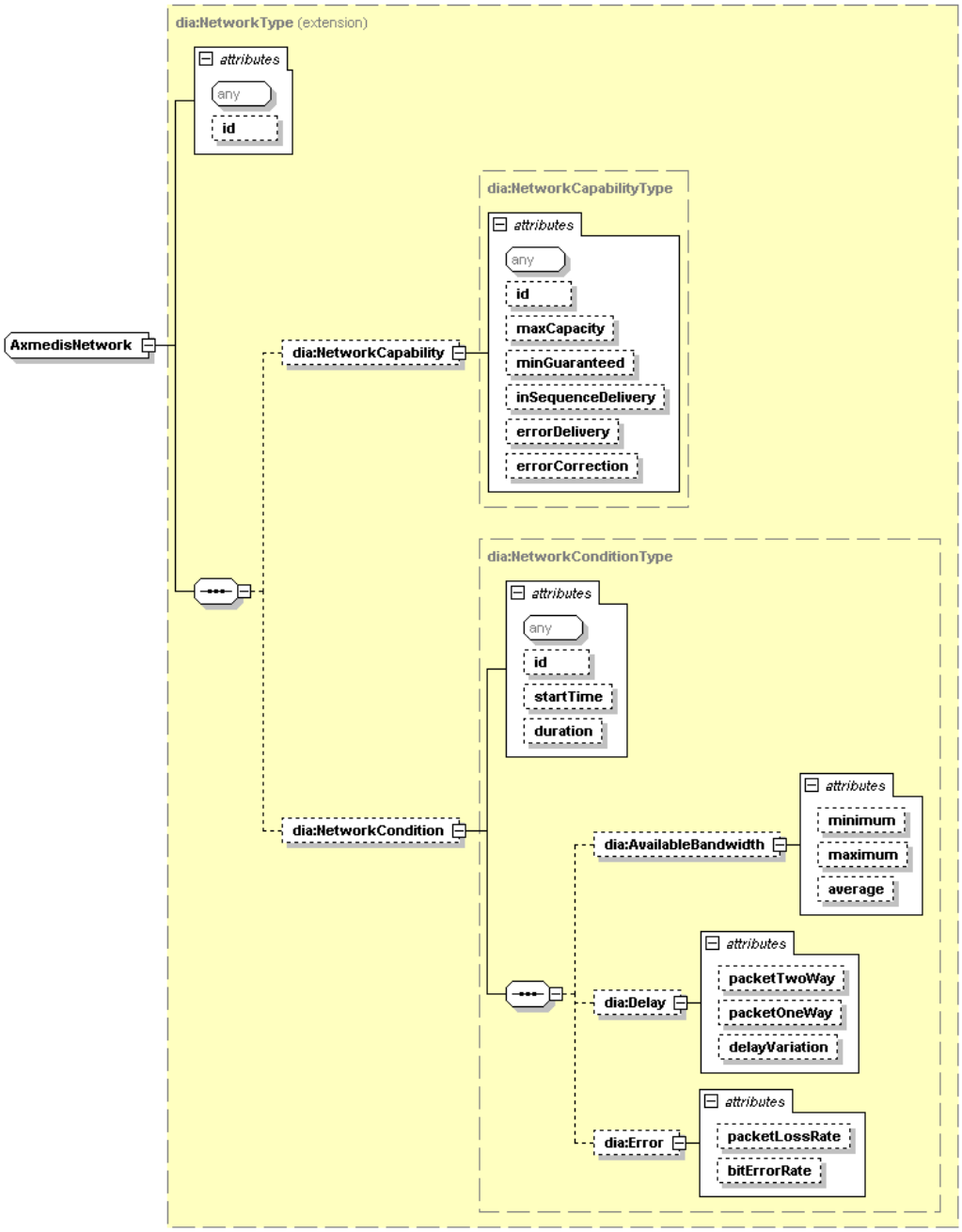

Figure 2.3: Schema for AXMEDIS Network Profile

3.3 Scripts Examples for Transcoding Content for Mobiles (ILABS, DSI)

3.3.1 AXCP Rule for Resizing Images

The following AXCP rule is used to resize set of images to match the screen size of the PDA/Mobile Device.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Rule xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="Rule_AXMEDIS.xsd">
 <Header>
  <Rule_Name>ely2</Rule_Name>
  <AXRID>axcprule:00d0eea1-16af-49bd-974b-5082513a802f</AXRID>
  <Rule_Version>
  </Rule_Version>
  <Rule_Type>AXCP</Rule_Type>
  <Software_Name>
```

```
  </Software_Name>
  <Version_of_software>
  </Version_of_software>
  <Date_of_production>2006-04-10</Date_of_production>
  <Author>
  </Author>
  <Affiliation>
  </Affiliation>
  <URL>
  </URL>
  <Comment>
  </Comment>
  <Last_Modifications>2006-07-27</Last_Modifications>
  <Terminal_ID>
  </Terminal_ID>
  <Cost>
  </Cost>
  <Work_Item_ID>
  </Work_Item_ID>
 </Header>
 <Schedule>
  <Run>
   <Date>2006-04-10</Date>
   <Time>12:30:35</Time>
   <Periodicity Unit="Day">0</Periodicity>
   <Expiration_Date>2006-04-10</Expiration_Date>
   <Expiration_Time>12:30:35</Expiration_Time>
  </Run>
  <Status>Inactive</Status>
 </Schedule>
 <Definition>
  <AXCP_Rule>
   <Arguments>
    <Parameter Name="PathToObjects" Type="String">E:\AXMEDIS\temp\source\</Parameter>
    <Parameter Name="objs" Type="String">Blood Frenzy.axm;</Parameter>
    <Parameter Name="ScreenW" Type="Integer">101</Parameter>
    <Parameter Name="ScreenH" Type="Integer">54</Parameter>
    <Parameter Name="outputPath" Type="String">E:\AXMEDIS\temp\result\</Parameter>
   </Arguments>
   <Rule_Body>
    <JS_Script name="JScript(8)"><![CDATA[function test()
{
 var AXMEDISobjs = objs.split(";");
 print("AXMEDISobjs.length="+AXMEDISobjs.length);
 for (index=0;index<AXMEDISobjs.length-1;index++)
 {
  print("AXMEDISobjs["+index+"]="+AXMEDISobjs[index]);
            var  newObj = new AXMEDISObject(PathToObjects+AXMEDISobjs[index]);
            var resources = newObj.getContent();
            print("resources.length="+resources.length);

            var h =  new Array(1);
            var w =  new Array(1);
            h[0]=0;
            w[0]=0;
            for (i=0;i<resources.length;i++)
            {
              var aRes = resources[i];
              if (aRes.mimeType!=null)
              {
                 var val = aRes.mimeType.search("image");
            (val==0)
                 {
                   ImageProcessing.GetInfo(aRes,h,w);
                   print("Original resource size is "+h+"x"+w);
                   print("Resizing the resource at "+ScreenW+"x"+ScreenH);
                   ImageProcessing.Resize(aRes,ScreenH,ScreenW,false,aRes);

                   ImageProcessing.GetInfo(aRes,h,w);
                   var ipath = outputPath+"res"+i+aRes.ref;
                   print(ipath);
                   aRes.save(ipath);
                 }
              }
```

```
                }
                print("Storing Object on disk");
                var dc = newObj.getDublinCore();
                var title = dc.getDCElementValue("title");
                print (title);
                newObj.save(outputPath+title+".axm");
 }
 return true;
}

//Entry point of the script();]]></JS_Script>
    </Rule_Body>
    <Dependencies>
     <Dependency>
       <Plug_In_name>ImageProcessing</Plug_In_name>
       <Version>1.001</Version>
     </Dependency>
    </Dependencies>
  </AXCP_Rule>
 </Definition>
</Rule>
```

Consider the following screenshot for an AXMEDIS Object containing an image:

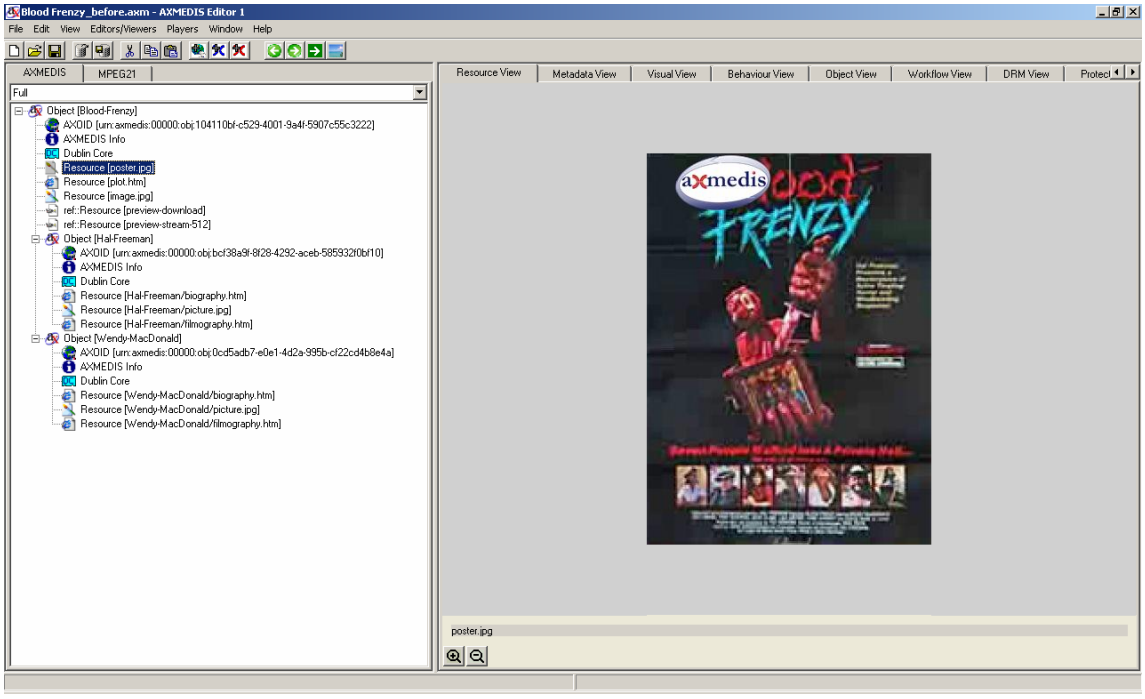

Figure 2.4 : AXMEDIS Object Showing the Original Image

After the execution of the above rule, the image in this object will be reduced to the size specified as shown in the following screenshot:

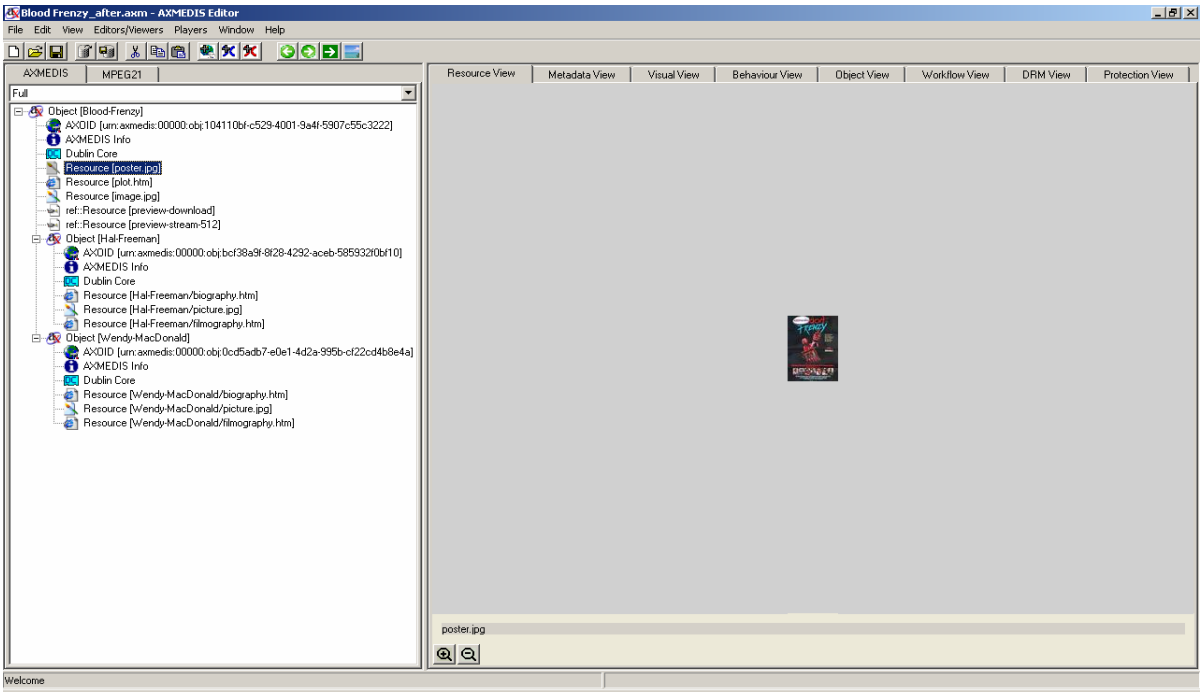

Figure 2.5 : AXMEDIS Object Showing the Resized Image

3.3.2 AXCP Rule for converting an Image from Colour to Black and White

The following AXCP rule is used to convert an image from Colour to Black and White. This rule is used for transcoding the image to be distribution on old fashion monochrome display PDA/Mobile Devices.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<Rule xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="Rule_AXMEDIS.xsd">
 <Header>
  <Rule_Name>ImageToBN</Rule_Name>
  <AXRID>axcprule:3c62ca39-dc96-44ab-8ae1-40c08c82c54a</AXRID>
  <Rule_Version></Rule_Version>
  <Rule_Type>AXCP</Rule_Type>
  <Software_Name></Software_Name>
  <Version_of_software></Version_of_software>
  <Date_of_production>2006-04-13</Date_of_production>
  <Author></Author>
  <Affiliation></Affiliation>
  <URL></URL>
  <Comment>get a colour image and convert it to B/N image</Comment>
  <Last_Modifications>2006-09-07</Last_Modifications>
  <Terminal_ID></Terminal_ID>
  <Cost></Cost>
  <Work_Item_ID></Work_Item_ID>
 </Header>
 <Schedule>
  <Run>
   <Date>2006-04-13</Date>
   <Time>14:56:10</Time>
   <Periodicity Unit="Day">0</Periodicity>
   <Expiration_Date>2006-04-13</Expiration_Date>
   <Expiration_Time>14:56:10</Expiration_Time>
  </Run>
  <Status>Inactive</Status>
 </Schedule>
 <Definition>
  <AXCP_Rule>
   <Arguments>
    <Parameter Name="ImageToProcess" Type="String">E:\titty.jpg</Parameter>

   </Arguments>
   <Rule_Body>
```

```
      <JS_Script name="JScript(4)"><![CDATA[function test()
{
 var h =  new Array(1);
 var w =  new Array(1);
 h[0]=0;
 w[0]=0;
 var aRes = new AxResource();
 aRes.load(ImageToProcess);

 print("Original resource size is "+h+"x"+w);
 ImageProcessing.Monochrome(aRes,aRes);
 var ipath = ImageToProcess.substring(0, ImageToProcess.indexOf("."));
 aRes.save(ipath+"BN");

 return true;
}

//Entry point of the script();]]>
      </JS_Script>
    </Rule_Body>
    <Dependencies>
      <Dependency>
        <Plug_In_name>ImageProcessing</Plug_In_name>
        <Version>1.001</Version>
      </Dependency>
      <Dependency>
        <Plug_In_name>AudioAdaptation</Plug_In_name>
        <Version>1.001</Version>
      </Dependency>
    </Dependencies>
  </AXCP_Rule>
 </Definition>
</Rule>
```

Consider the following coloured image. In order to distribute this image to a mobile device having a monochrome display, the above rule will be applied.

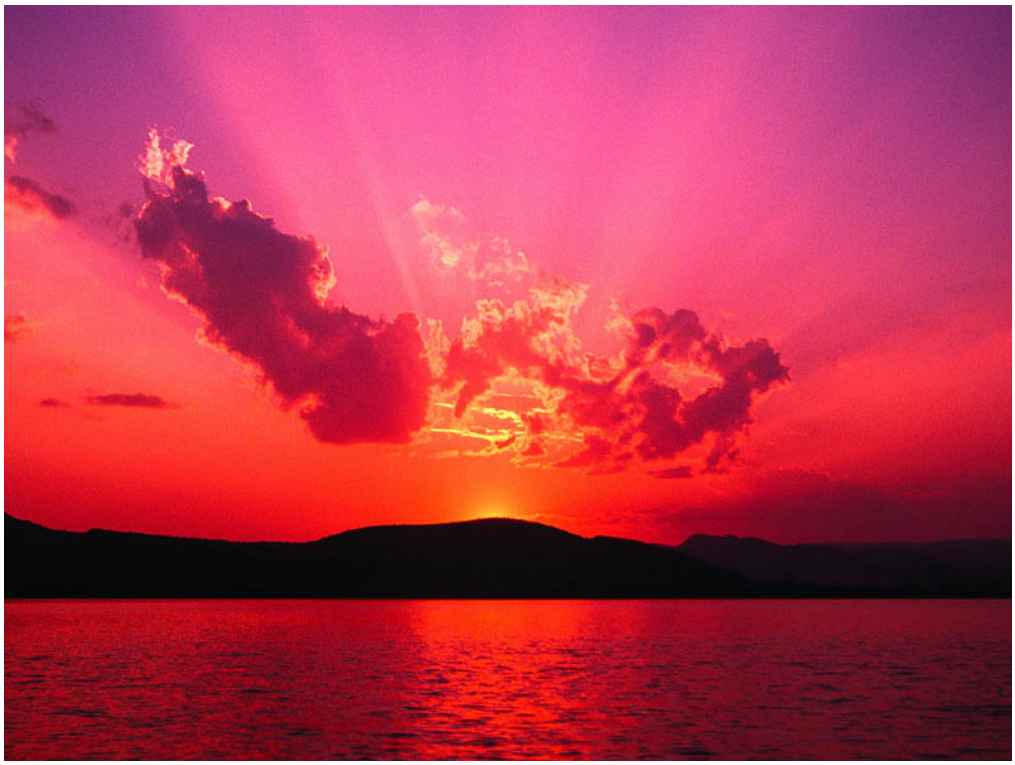

Figure 2.6 : The Original Coloured Image.

The output produce by the rule is as shown in the following of the image:

Figure 2.7: The Image Converted to Black and White

3.3.3 AXCP Rule for creating AXMEDIS object with SMIL content

This rule creates an 800 x 600 SMIL object that encompasses various image resources as well as an audio resource, using a SMIL template to format the resources in the 800 x 600 SMIL object.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<Rule xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="Rule_Axmedis.xsd">
 <Header>
  <Rule_Name>CreateSMILart_contents</Rule_Name>
  <AXRID>axcprule:afd10344-0668-44de-832a-3816934010d8</AXRID>
  <Rule_Version></Rule_Version>
  <Rule_Type>AXCP</Rule_Type>
  <Software_Name></Software_Name>
  <Version_of_software></Version_of_software>
  <Date_of_production>2007-02-07</Date_of_production>
  <Author></Author>
  <Affiliation></Affiliation>
  <URL></URL>
  <Comment></Comment>
  <Last_Modifications>2007-09-07</Last_Modifications>
  <Terminal_ID></Terminal_ID>
  <Cost></Cost>
  <Work_Item_ID></Work_Item_ID>
 </Header>
 <Schedule>
  <Run>
   <Date>2007-02-07</Date>
   <Time>10:20:38</Time>
   <Periodicity Unit="Day">0</Periodicity>
   <Expiration_Date>2007-02-07</Expiration_Date>
   <Expiration_Time>10:20:38</Expiration_Time>
  </Run>
  <Status>Inactive</Status>
```

```
  </Schedule>
  <Definition>
    <AXCP_Rule>
      <Arguments>
        <Parameter Name="smil_template_filepath" Type="String">E:\\smiltemplate\\</Parameter>
        <Parameter Name="smil_template_filename" Type="String">0index.smil</Parameter>
        <Parameter Name="resources_folder" Type="String">E:\\smilres\\</Parameter>
        <Parameter Name="smil_content_group_filepath" Type="String">E:\\smilcontentgroup\\group-original.xml</Parameter>
        <Parameter Name="output_path" Type="String">E:\\smiloutput\\</Parameter>
        <Parameter Name="N" Type="Integer">5</Parameter>
        <Parameter Name="title" Type="String">SMILtestings</Parameter>
        <Parameter Name="creator" Type="String">UR - modified version of script created by ILABS through AXCP Rule
Editor</Parameter>
        <Parameter Name="description" Type="String">Testing DIA following JSScript SMIL creation</Parameter>

      </Arguments>
      <Rule_Body>
        <JS_Script name="JScript(1)"><![CDATA[/* Lists the files (only files, no directories) in a directory
with a certain mask
returns an array with the files.
If no file is found it will return an empty array */
function getfilelist (
getfilelist_dir,        //directory
getflielist_mask        // mask e.g. "*.*"
)
{
  var list = new Array();
  if (!existsDir(getfilelist_dir))  // check that the directory exists
  {
              return (list);
  }
  var file = getFirstFile(getfilelist_dir, getflielist_mask);
  if (file != null)
  {
    list.push (file);
  }
  while (file != null)
  {
     var nextfile = getNextFile();
     if (nextfile == null)
     {
       break;
     }
     list.push (nextfile);
     nextfile = undefined;
   }
  return (list);
}

function FillSmilTemplate()
{
  var strXML = readFromFile(smil_template_filepath+smil_template_filename);
  var xmltemplate = new XML(strXML);
 // print (xmltemplate);
  //look for par and add pars inside for img+text
  for (var bodyindex in xmltemplate.body)
  {
     var bodynode = xmltemplate.body[bodyindex];
     for (var parindex in bodynode.par)
     {
       var entrypoint = bodynode.par[parindex];
       //print(entrypoint);
       //Set end time for audio
       entrypoint.audio[0].@end = ((5*N)+1)+"s";

       for (var temp=1;temp<=N;temp++)
       {
          var groupXML = readFromFile(smil_content_group_filepath);
          var parToAdd = new XML(groupXML);

          var dummy = (temp-1) * 5;
          // Set begin time
          parToAdd.@begin = dummy+"s";
```

```
            // Set image, subtitle, title and text attributes
            var imgnode = parToAdd.img[0];
            imgnode.@id = imgnode.@id+temp;
            var srcold = imgnode.@src;
            var imgval = srcold.substring(0,srcold.indexOf("."))
                    + temp
                    + srcold.substring(srcold.indexOf(".")) ;
            print("imgval="+imgval);
            imgnode.@src = imgval;

            for (var textindex in parToAdd.text)
            {
               var textnode = parToAdd.text[textindex];
               textnode.@id = textnode.@id +temp;
               var srcold = textnode.@src;
               var textval = srcold.substring(0,srcold.indexOf("."))
                    + temp
                    + srcold.substring(srcold.indexOf(".")) ;
               print("imgval="+textval);
               textnode.@src = textval;
            }

            print("parToAdd = "+parToAdd);
            entrypoint.appendChild(parToAdd);
            print(xmltemplate);
         }
         // Add namespace
         var namespaceVar = new Namespace("xmlns",'http://www.w3.org/2005/SMIL21/');
         xmltemplate.addNamespace(namespaceVar);

         var strtowrite = "<?xml version=\"1.0\" encoding=\"ISO-8859-1\"?>"
         +xmltemplate.toXMLString();
         writeToFile(resources_folder+smil_template_filename,strtowrite);
       }
     }
}

function FillTemplateAndCreateObjects()
{
   FillSmilTemplate();
   BuildAxmedisObject();
   return true;
}

FillTemplateAndCreateObjects();]]>
      </JS_Script>
      <JS_Script name="JScript(2)">

        <![CDATA[// Function for creating the Dublin Core information
function createDC(obj,title)
{
   dc = obj.getDublinCore();
   dc.addDCElement("creator",creator);
   dc.addDCElement("title",title);
   dc.addDCElement("description",description);
}
// Get resources list
function createResourceArray()
{
   var resArray = getfilelist(resources_folder,"*.*");
   return resArray;
}
function  BuildAxmedisObject()
{
  var obj = new AxmedisObject();

  var resources = createResourceArray();
  if(resources.length==0)
     return false;

  var filename_array = null;
  var ext = null;

  for(j=0; j<resources.length; j++)
```

```
{
   var resource = new AxResource();
   // if resource is smil object, its mimeType is somehow given as text/xml, which
   // has to be changed manually in the AXMEDIS Editor, for it to be played normally
   resource.load(resources[j]);
   filename_array = resources[j].split("\\");                    //set filename as contentID of resource
   resource.contentID = filename_array[filename_array.length-1]; //set filename as contentID of resource
   ext = resources[j].split(".");
   if ( ext[ext.length-1].search("smil") == 0)
      resource.mimeType = "application/smil";
   obj.addContent(resource);

   resource.flush();
   resource= null;
   filename_array = null;
}
 print("    Creating Dublin Core Metadata");
 createDC(obj,title);
 obj.save(output_path+"\\"+title+".axm");
}]]>
     </JS_Script>
   </Rule_Body>
   <Dependencies/>
  </AXCP_Rule>
 </Definition>
</Rule>
```

The following 800x600 SMIL content AXOBJECT is created by the above Rule.

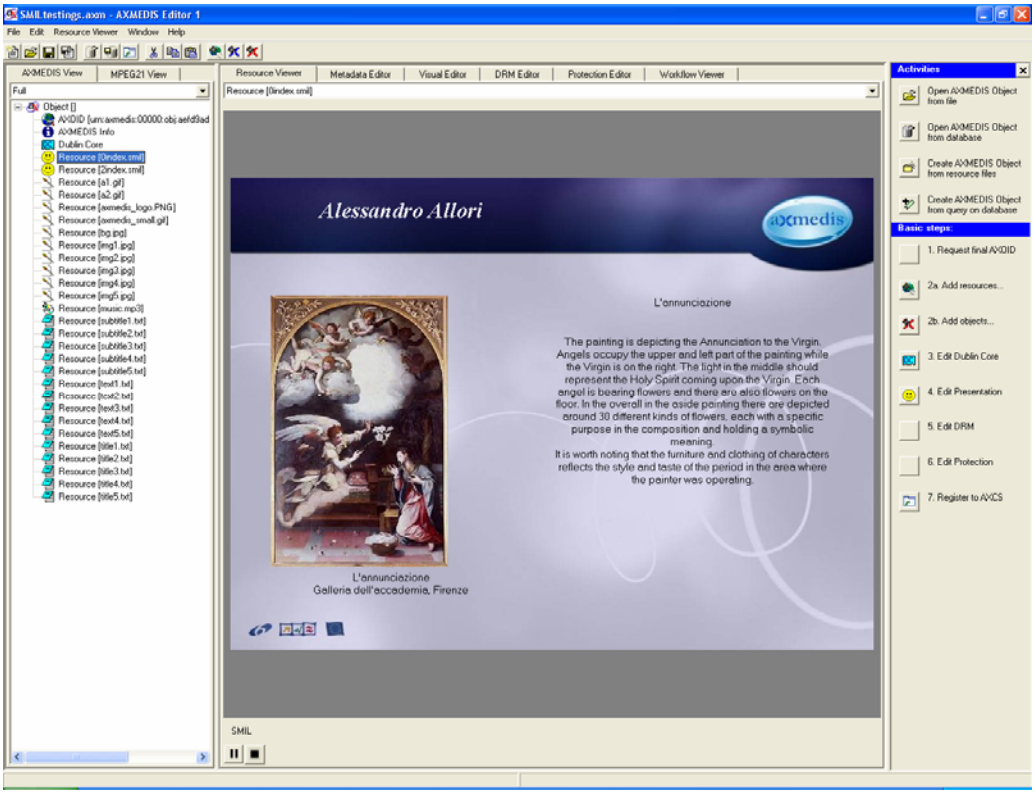

Figure: 2.8 SMIL content (image, audio, formatting) in AXOBJECT

3.3.4 AXCP Rule for adapting AXMEDIS object with SMIL content

This Rule adapts an AXOBJECT with SMIL content (800 x 600) by extracting and adapting its resources one-by-one to be used on a mobile device with screen size of 320 x 240, using a specified SMIL template to format the adapted resources in the 320 x 240 SMIL object.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<Rule xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="Rule_Axmedis.xsd">
```

```
<Header>
  <Rule_Name>CreateSMILart_contents</Rule_Name>
  <AXRID>axcprule:afd10344-0668-44de-832a-3816934010d8</AXRID>
  <Rule_Version></Rule_Version>
  <Rule_Type>AXCP</Rule_Type>
  <Software_Name></Software_Name>
  <Version_of_software></Version_of_software>
  <Date_of_production>2007-02-07</Date_of_production>
  <Author></Author>
  <Affiliation></Affiliation>
  <URL></URL>
  <Comment></Comment>
  <Last_Modifications>2007-09-07</Last_Modifications>
  <Terminal_ID></Terminal_ID>
  <Cost></Cost>
  <Work_Item_ID></Work_Item_ID>
 </Header>
 <Schedule>
  <Run>
   <Date>2007-02-07</Date>
   <Time>10:20:38</Time>
   <Periodicity Unit="Day">0</Periodicity>
   <Expiration_Date>2007-02-07</Expiration_Date>
   <Expiration_Time>10:20:38</Expiration_Time>
  </Run>
  <Status>Inactive</Status>
 </Schedule>
 <Definition>
  <AXCP_Rule>
   <Arguments>
    <Parameter Name="smil_template_filepath" Type="String">E:\\smiltemplate\\</Parameter>
    <Parameter Name="smil_template_filename" Type="String">2index.smil</Parameter>
    <Parameter Name="resources_folder" Type="String">E:\\smilres\\</Parameter>
    <Parameter Name="smil_content_group_filepath" Type="String">E:\\smilcontentgroup\\group.xml</Parameter>
    <Parameter Name="output_path" Type="String">E:\\smiloutput\\</Parameter>
    <Parameter Name="N" Type="Integer">5</Parameter>
    <Parameter Name="title" Type="String">SMILtestings</Parameter>
    <Parameter Name="resourcePath" Type="String">E:\\DIA\\</Parameter>
    <Parameter Name="profile" Type="String">Samsung-i320.xml</Parameter>
    <Parameter Name="output" Type="String">diadtatest-2.axm</Parameter>

   </Arguments>
   <Rule_Body>
    <JS_Script name="createSMILObject"><![CDATA[/* Lists the files (only files, no directories) in a directory with a certain mask
returns an array with the files. If no file is found it will return an empty array */

/*
function getfilelist (getfilelist_dir, getflielist_mask)
{
 // getfilelist_dir directory
 // getfilelist_mask e.g. "*.*"
 var list = new Array();
 if (!existsDir(getfilelist_dir))  // check that the directory exists
 {
            return (list);
 }
 var file = getFirstFile(getfilelist_dir, getflielist_mask);
 if (file != null)
 {
  list.push (file);
 }
 while (file != null)
 {
   var nextfile = getNextFile();
   if (nextfile == null)
   {
    break;
   }
   list.push (nextfile);
   nextfile = undefined;
  }
 return (list);
}
*/
```

```
function FillSmilTemplate()
{
  var strXML = readFromFile(smil_template_filepath+smil_template_filename);
  var xmltemplate = new XML(strXML);
 // print (xmltemplate);
  //look for par and add pars inside for img+text
  for (var bodyindex in xmltemplate.body)
  {
     var bodynode = xmltemplate.body[bodyindex];
     for (var parindex in bodynode.par)
     {
       var entrypoint = bodynode.par[parindex];
       //print(entrypoint);
       //Set end time for audio
       entrypoint.audio[0].@end = ((5*N)+1)+"s";

       for (var temp=1;temp<=N;temp++)
       {
          var groupXML = readFromFile(smil_content_group_filepath);
          var parToAdd = new XML(groupXML);

          var dummy = (temp-1) * 5;
          // Set begin time
          parToAdd.@begin = dummy+"s";

          // Set image, subtitle, title and text attributes
          var imgnode = parToAdd.img[0];
          imgnode.@id = imgnode.@id+temp;
          var srcold = imgnode.@src;
          var imgval = srcold.substring(0,srcold.indexOf("."))
                  + temp
                  + srcold.substring(srcold.indexOf(".")) ;
          print("imgval="+imgval);
          imgnode.@src = imgval;

          for (var textindex in parToAdd.text)
          {
             var textnode = parToAdd.text[textindex];
             textnode.@id = textnode.@id +temp;
             var srcold = textnode.@src;
             var textval = srcold.substring(0,srcold.indexOf("."))
                  + temp
                  + srcold.substring(srcold.indexOf(".")) ;
             print("imgval="+textval);
             textnode.@src = textval;
          }

          print("parToAdd = "+parToAdd);
          entrypoint.appendChild(parToAdd);
          print(xmltemplate);
       }
       // Add namespace
       var namespaceVar = new Namespace("xmlns",'http://www.w3.org/2005/SMIL21/');
       xmltemplate.addNamespace(namespaceVar);

       var strtowrite = "<?xml version=\"1.0\" encoding=\"ISO-8859-1\"?>"
       +xmltemplate.toXMLString();

       writeToFile(resources_folder+smil_template_filename,strtowrite);

       var resource = new AxResource();
       resource.load(resources_folder+smil_template_filename);
       resource.mimeType = getMimeTypeFromExt("smil");
       resource.contentID = smil_template_filename;
       return (resource);
     }
   }
}

/*
// Get resources list
function createResourceArray()
{
```

```
    var resArray = getfilelist(resources_folder,"*.*");
    return resArray;
}
function  BuildAxmedisObject()
{
  var obj = new AxmedisObject();

  var resources = createResourceArray();
  if(resources.length==0)
     return false;

  var filename_array = null;
  var ext = null;
  for(j=0; j<resources.length; j++)
  {
     var resource = new AxResource();
     // if resource is smil object, its mimeType is somehow given as text/xml, which
     // has to be changed manually in the AXMEDIS Editor, for it to be played normally
     resource.load(resources[j]);
     filename_array = resources[j].split("\\");                  //set filename as contentID of resource
     resource.contentID = filename_array[filename_array.length-1]; //set filename as contentID of resource
     ext = resources[j].split(".");
     if ( ext[ext.length-1].search("smil") == 0)
        resource.mimeType = getMimeTypeFromExt(ext[ext.length-1]);
     obj.addContent(resource);

     resource.flush();
     resource= null;
     filename_array = null;
  }
   obj.save(output_path+"\\"+title+".axm");
}

function FillTemplateAndCreateObject()
{
   FillSmilTemplate();
   BuildAxmedisObject();
   return true;
}*/]]>
        </JS_Script>
        <JS_Script name="AxProfileObjectAdapter">

          <![CDATA[/*****************************************************************************/
// AXMEDIS JSScript - Digital Item Adaptation and Profile Management
//
// 1. Load an existing AXMEDIS object with one to many resources
// 2. Sends AXMEDIS profile URL to the AXMEDIS DIA Decision Taking Engine
// 3. AXMEDIS DIA Decision Taking Engine takes appropriate decisions and makes
//    necessary changes in the profile values for the object
// 3. Extract each resource and adapt it according to the profiles values
// 4. Add each adapted resource to a new AXMEDIS object
// 5. Save new AXMEDIS object
//
// University of Reading - AXMEDIS
/*****************************************************************************/
function AxProfileObjectAdapter(inputObject,profInstance,outputObject)
{
   //Load input AxObject
   var axmedisObject = new AxmedisObject(inputObject);
   var res = new AxResource();
   //Assign AxObject Content
   var resources = axmedisObject.getContent();
   //Create new AxObject to store adapted resources
   var newAxmedisObject = new AxmedisObject();

   print("\t***AXMEDIS Digital Item Adaptation based on AXMEDIS Profiles***\n");

   //Cycle through resources, taking action depending on resource mimeType
   for(k in resources)
   {
      print("\n\t\t\t*******************\n\nResource "+k+" in AXMEDIS object \""+axmedisObject.contentID+"\" is being processed...");
      res = resources[k];

      print("\nResource "+k+" Details");
```

```
        print("================");

        print(res.contentID);
        if(resources[k].mimeType.match("image/"))
        {
           print("Resource Type = " + res.mimeType);
           //Image specific adaptations
           res = adaptImageRes(res,profInstance);
           //General adaptations
           res = adaptRes(res,profInstance);

           var h2 = new Array(1);
           var w2 = new Array(1);
           h2[0] = 0;
           w2[0] = 0;
           print("\nAdapted Resource Details");
           print("=====================");
           ImageProcessing.GetInfo(res,h2,w2);
           print("Adapted resource size is "+h2[0]+"x"+w2[0]);
        }
        if(resources[k].mimeType.match("video/"))
        {
           print("Resource Type = " + res.mimeType);
           res = adaptRes(res,profInstance);
        }

        if(resources[k].mimeType.match("audio/"))
        {
           print("Resource Type = " + res.mimeType);
           res = adaptRes(res,profInstance);
        }

        if(resources[k].mimeType.match("text/"))
        {
           print("Text Type = " + res.mimeType);
           res = adaptRes(res,profInstance);
        }
        if(resources[k].mimeType.match("application/smil"))
        {
           print("Resource Type = " + res.mimeType);
           res = FillSmilTemplate();
           res = adaptRes(res,profInstance);
        }
        newAxmedisObject.addContent(res);
    }

    cloneAxmedisObjectMetadata(axmedisObject, newAxmedisObject)

    newAxmedisObject.contentID = axmedisObject.contentID + "-adapted";
    // Store the Axmedis Object on Disk
    newAxmedisObject.save(outputObject);

    return true ;
}]]>
        </JS_Script>
        <JS_Script name="AxProfileObjectAdapter_Video">

          <![CDATA[function adaptImageRes(res,profInstance)
{
    var res1 = new AxResource();
    res1=res;

    var ap = new AxUserProfile();
    ap.loadXMLFile(profInstance);

    var h1 = new Array(1);
    var w1 = new Array(1);
    h1[0] = 0;
    w1[0] = 0;

    ImageProcessing.GetInfo(res,w1,h1);
    print("Original Resource Size: "+w1[0]+"x"+h1[0]);

    print("\nPreferences available in AXMEDIS profiles:");
```

```
    print("==================================");

            var deviceName = ap.getValue("Name", "AxmedisTerminal/DeviceClass/DeviceClass/Name");
    var screenSizeX = ap.getAttribute("AxmedisTerminal/Displays/Display/DisplayCapability/ScreenSize/horizontal");
    var screenSizeY = ap.getAttribute("AxmedisTerminal/Displays/Display/DisplayCapability/ScreenSize/vertical");
    var colorCapable = ap.getAttribute("AxmedisTerminal/Displays/Display/DisplayCapability/colorCapable");
    print ("Device: " + deviceName);
    print ("Screen size: " + screenSizeX + " x " + screenSizeY );
    print ("Color capable? " + colorCapable);

    // These need to be dynamically supplied from the SMIL template somehow
    var originalScreenX = 800;
    var originalScreenY = 600;

    if (colorCapable == "false")
    {
        print("ImageProcessing Grayscale func: "+ImageProcessing.Grayscale(res1, res1));
    }

    //print("\nAdaptation Report:");
    //print("===============");

    if ((screenSizeX != originalScreenX) || (screenSizeY != originalScreenY))
    {
        ImageProcessing.GetInfo(res1,w1,h1);
        print (w1);
        print (h1);
        var newX = (w1/originalScreenX) * screenSizeX;
        var newY = (h1/originalScreenY) * screenSizeY;
        //ImageProcessing.Resize(res1, w1[0]/2, h1[0]/2, true, res1);
        ImageProcessing.Resize(res1, newX, newY, false, res1);
    }

    ap.save();

    // Change resource contentID and add the resource in the Axmedis Object
    //res1.contentID = res.contentID+"-DIA-profiling";
    //print(res1.size);*/

    return res1;
}]]>
        </JS_Script>
        <JS_Script name="AxProfileObjectAdapter_General">

          <![CDATA[function adaptRes(res,profInstance)
{
    var res1 = new AxResource();
    res1=res;

 /*   var ap = new AxUserProfile();
    ap.loadXMLFile(profInstance);

    var resHeight = new Array(1);
    var resWidth = new Array(1);
    resHeight[0] = 0;
    resWidth[0] = 0;

    if(res1.mimeType.match("image/"))
        ImageProcessing.GetInfo(res1,resHeight,resWidth);

    //Device Capabilities
    var resX = ap.getAttribute("AxmedisTerminal/Displays/Display/DisplayCapability/Mode/Resolution/horizontal");
    var resY = ap.getAttribute("AxmedisTerminal/Displays/Display/DisplayCapability/Mode/Resolution/vertical");
    var colourCapable = ap.getAttribute("AxmedisTerminal/Display/Display/DisplayCapability/colorCapable");
    var convresX = 0;
    var convresY = 0;

    ////////////////////////////////////////////////////
    ////////////////// Start of conversion ////////////////////
    ////////////////////////////////////////////////////
    if(res1.mimeType.match("image/"))
    {
        ///convert to codecCap[j]
        if (colourCapable == "false") //Make black and white
```

```
        {
            print("ImageProcessing Plugin, Grayscale Function: " +
                        ImageProcessing.GrayScale(res1, res1));
        }

        if ((resX < resWidth)||(resY < resHeight)) //Resize image to device resolution
        {   print("ImageProcessing Plugin, Resize for Terminal: " +
                        ImageProcessing.Resize(res1, resX, resY, true, res1));
        }
    }

    ap.save();*/

    // Change resource contentID and add the resource in the Axmedis Object
    var resName = res1.contentID.split(".");
    res1.contentID = resName[0] + "." + getMimeTypeExt(res1.mimeType);
    //print(res1.size);
    return res1;
}]]>
        </JS_Script>
        <JS_Script name="utilityFunctions">

          <![CDATA[function cloneAxmedisObjectMetadata(sourceObj,targetObj)
{
        var metadata = sourceObj.getGenericMetadata();
        if(metadata!=null)
        {
            for(j in metadata)
                targetObj.addMetadata(metadata[j]);
        }
        targetObj.removeMetadata(targetObj.getDublinCore());
        targetObj.removeMetadata(targetObj.getAxInfo());
        targetObj.addMetadata(sourceObj.getAxInfo());
        targetObj.addMetadata(sourceObj.getDublinCore());

        targetObj.contentID = sourceObj.contentID+"-adapted";
}
function getMimeTypeExt(res1)
{
    switch(res1)
    {
        case "audio/mpeg":
            return "mp3";
        case "audio/x-mpeg":
            return "mp3";
        case "audio/x-wav":
            return "wav";
        case "audio/x-ms-wma":
            return "wma";
        case "audio/x-pn-realaudio":
            return "ra";
        case "audio/x-ac3":
            return "ac3";
        case "audio/x-aac":
            return "aac";
        case "audio/mp4":
            return "mp4";
        case "video/x-msvideo":
            return "avi";
        case "video/mpeg":
            return "mpg";
        case "video/avi":
            return "avi";
        case "image/jpeg":
            return "jpg";
        case "image/gif":
            return "gif";
        case "image/tiff":
            return "tif";
        case "image/png":
            return "png";
        case "image/bmp":
            return "bmp";
        case "text/html":
```

```
            return "html";
        case "text/plain":
            return "txt";
        case "application/msword":
            return "doc";
        case "application/rtf":
            return "rtf";
        case "application/postscript":
            return "ps";
        case "application/pdf":
            return "pdf";
        case "application/smil":
            return "smil";
        default:
            return "none";
    }
}
function getMimeTypeFromExt(res1)
{
    switch(res1)
    {
        case "mp3":
            return "audio/mpeg";
        case "mp3":
            return "audio/x-mpeg";
        case "wav":
            return "audio/x-wav";
        case "wma":
            return "audio/x-ms-wma";
        case "ra":
            return "audio/x-pn-realaudio";
        case "ac3":
            return "audio/x-ac3";
        case "aac":
            return "audio/x-aac";
        case "mp4":
            return "audio/mp4";
        case "avi":
            return "video/x-msvideo|";
        case "mpg":
            return "video/mpeg";
        case "avi":
            return "video/avi";
        case "jpg":
            return "image/jpeg";
        case "gif":
            return "image/gif";
        case "tif":
            return "image/tiff";
        case "png":
            return "image/png";
        case "bmp":
            return "image/bmp";
        case "html":
            return "text/html";
        case "txt":
            return "text/plain";
        case "doc":
            return "application/msword";
        case "rtf":
            return "application/rtf";
        case "ps":
            return "application/postscript";
        case "pdf":
            return "application/pdf";
        case "smil":
            return "application/smil";
        default:
            return "none";
    }
}]]>
        </JS_Script>
        <JS_Script name="main">
```

```
        <![CDATA[
//FillTemplateAndCreateObject();
//var proftemplate = AxDecisionTakingEngine.Activate(resourcePath+profile);
var profDTA = resourcePath+profile;
AxProfileObjectAdapter(output_path+title+".axm",profDTA,output_path+output);]]>
      </JS_Script>
    </Rule_Body>
    <Dependencies>
      <Dependency>
        <Plug_In_name>ImageProcessing</Plug_In_name>
        <Version>1.001</Version>
      </Dependency>
      <Dependency>
        <Plug_In_name>AudioAdaptation</Plug_In_name>
        <Version>1.001</Version>
      </Dependency>
      <Dependency>
        <Plug_In_name>MultimediaAdaptation</Plug_In_name>
        <Version>1.001</Version>
      </Dependency>
      <Dependency>
        <Plug_In_name>RingtoneAdaptation</Plug_In_name>
        <Version>1.001</Version>
      </Dependency>
      <Dependency>
        <Plug_In_name>TextDocsAdaptation</Plug_In_name>
        <Version>1.001</Version>
      </Dependency>
      <Dependency>
        <Plug_In_name>AudioDescriptor</Plug_In_name>
        <Version>1.001</Version>
      </Dependency>
      <Dependency>
        <Plug_In_name>VideoAdaptation</Plug_In_name>
        <Version>1.001</Version>
      </Dependency>
    </Dependencies>
  </AXCP_Rule>
 </Definition>
</Rule>
```

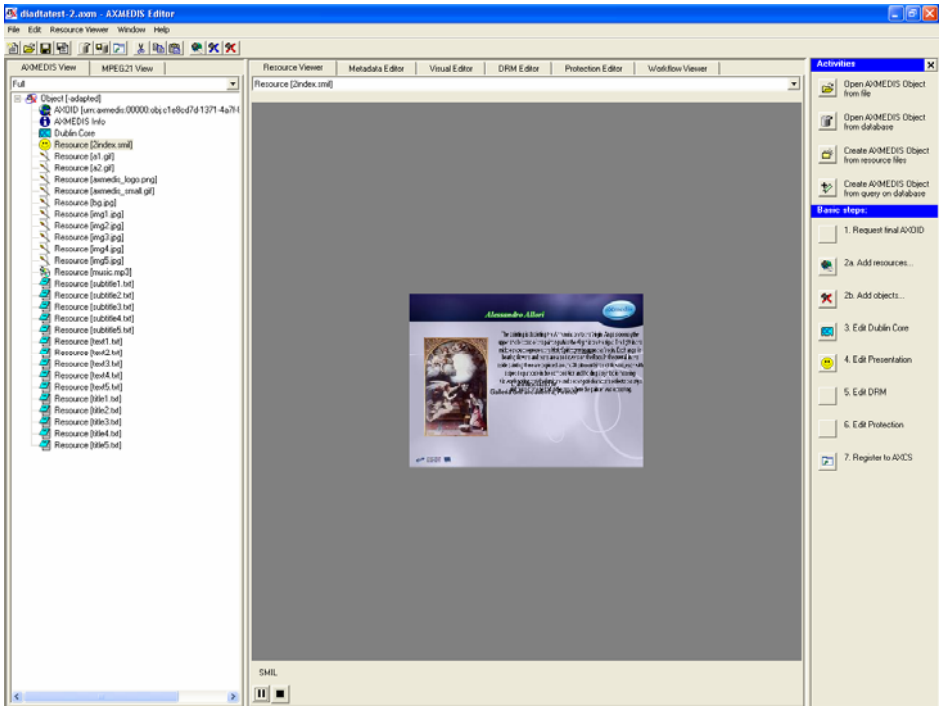

Figure 2.9: Adapted SMIL content (image, audio, formatting) in AXOBJECT

3.3.5 AXCP Rule for converting IMS packages into AXMEDIS Objects

The following AXCP rule is used to convert an IMS packages into AXMEDIS Objects.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<Rule xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="Rule_AXMEDIS.xsd">
 <Header>
  <Rule_Name>Upload_test</Rule_Name>
  <AXRID>axcprule:e8caaae1-1bc9-4699-9184-bb9a1918ffe9</AXRID>
  <Rule_Version></Rule_Version>
  <Rule_Type>AXCP</Rule_Type>
  <Software_Name></Software_Name>
  <Version_of_software></Version_of_software>
  <Date_of_production>2006-09-11</Date_of_production>
  <Author></Author>
  <Affiliation></Affiliation>
  <URL></URL>
  <Comment>A script for contents creation that: takes a starting folder inside this folder there are some subfolders, each one has inside the content of an ims content package: an imsmanifest.xml + all its resource files for each folder inside the main folder, create a new AXMEDIS object having as its resources all the files inside the folder; get some information from imsmanifest and insert into DC; save on disk the newly created object.: since importing an html page with images inside with relative path does not work properly, a workaround has been used to modify html file.: since loading a file into a reosurce, the resource looses the extension, a workaround has been used to add file extension to contentid property of the resource.</Comment>
  <Last_Modifications>2006-09-20</Last_Modifications>
  <Terminal_ID></Terminal_ID>
  <Cost></Cost>
  <Work_Item_ID></Work_Item_ID>
 </Header>
 <Schedule>
  <Run>
   <Date>2006-09-11</Date>
   <Time>16:37:20</Time>
   <Periodicity Unit="Day">0</Periodicity>
   <Expiration_Date>2006-09-11</Expiration_Date>
   <Expiration_Time>16:37:20</Expiration_Time>
  </Run>
  <Status>Inactive</Status>
 </Schedule>
 <Definition>
  <AXCP_Rule>
   <Arguments>
    <Parameter Name="CurrentLanguage" Type="String">en-US</Parameter>
    <Parameter Name="startfolder" Type="String">C:\Documents and Settings\ElisabettaP\Desktop\axcont_ArtistEN\</Parameter>
    <Parameter Name="default_image_path" Type="String">C:\default_img.jpg</Parameter>
    <Parameter Name="destinationfolder" Type="String">c:\contents_ax\v2\</Parameter>

   </Arguments>
   <Rule_Body>
    <JS_Script name="JScript(1)"><![CDATA[// Extract manifest title and put in into DCsetTitle(titleItem,obj)
{
 for (var i=0; i<titleItem.childElements.length; i++)
 {
  var item = titleItem.childElements[i];
  if (item.name=="langstring")
  {
   var dc = obj.getDublinCore();
   if (item.text!=null && item.text!="")
   {
    dc.addDCElement("title",item.text);
    titleOfCurrentContent = item.text;
    //print ("TITLE is "+item.text);
   }
  }
 }
}

// Extract manifest title and put in into DCsetDescription(descrItem,obj)
{
 for (var i=0; i<descrItem.childElements.length; i++)
 {
  var item = descrItem.childElements[i];
  if (item.name=="langstring")
```

```
  {
    var dc = obj.getDublinCore();
    if (item.text!=null && item.text!="")
    {
      dc.addDCElement("description",item.text);
      //print ("DESCRIPTION is "+item.text);
    }
  }
 }
}

// Extract manifest title and put in into DCsetCoverage(covItem,obj)
{
  for (var i=0; i<covItem.childElements.length; i++)
 {
   var item = covItem.childElements[i];
   if (item.name=="langstring")
   {
     var dc = obj.getDublinCore();
     if (item.text!=null && item.text!="")
     {
       dc.addDCElement("coverage",item.text);
       //print ("DESCRIPTION is "+item.text);
     }
   }
 }
}

// Set languagesetLanguage(obj)
{
 var dc = obj.getDublinCore();
 dc.addDCElement("language",CurrentLanguage);
}

// browse metadatabrowseMetadata(startNode, indent,obj)
{
   for (var i=0; i<startNode.childElements.length; i++)
   {
       var item = startNode.childElements[i];
       //print(indent +  item.type + " "+item.name );
       if (item.name == "title")
       {
         setTitle(item,obj);
       }
       else if (item.name == "description")
       {
         setDescription(item,obj);
       }
       else if (item.name == "coverage")
       {
         setCoverage(item,obj);
       }
       if (startNode.name!="general")
       {
         browseMetadata(item, indent +"   ",obj);
       }
   }
}

// read manifestparseManifest(xmlfile,obj)
{
 var strXML = readFromFile(xmlfile);
 var xmlDoc = new REXML(strXML);

  for (var i=0; i<xmlDoc.rootElement.childElements.length; i++)
   {
       var item = xmlDoc.rootElement.childElements[i];
       //print("Child element of type " + item.type + " "+item.name );
       if (item.name == "metadata")
       browseMetadata(item,"   ", obj);
   }
   setLanguage(obj);
}
```

```
// Workaround for bad import of resources; flatten image path
// NOW it flattens only first image, TODO: flatten all images sourcesflattenImagesIn(aFile,axobj)
{
  var fileContent = readFromFile(aFile);
  var index = fileContent.indexOf("src=\"./");
  var firstpiece = "";
  var sndpiece = "";

  if (index >0)
  {
  // flatten image path
    index = index +"src=\"".length;
    firstpiece = fileContent.substring(0, index );
    sndpiece   = fileContent.substring(index+5,fileContent.length );
    var end    = sndpiece.substring(sndpiece.indexOf("\""), sndpiece.length);
    var ref    = sndpiece.substring(0,sndpiece.indexOf("\""));
    ref        = ref.substring(ref.lastIndexOf("/")+1,ref.length);
    print ("-- ref="+ref);
    var result = firstpiece+ref +end;
    removeFile(aFile);
    writeToFile(aFile, result);
  }
  else
  {
    // put default image filename
    index = fileContent.indexOf("src=\".");
    print("NO IMAGE index="+index);
    firstpiece = fileContent.substring(0, index +"src=\"".length);
    sndpiece   = fileContent.substring(index+5,fileContent.length );
    //print ("firstpiece="+firstpiece);
    //print("sndpiece="+sndpiece);

    var end    = sndpiece.substring(sndpiece.indexOf("\""), sndpiece.length);
    var ref    = default_image_path.substring(default_image_path.lastIndexOf("\\")+1,default_image_path.length);

    //print ("-- ref="+ref);
    var result = firstpiece +ref +end;
    //print("result="+result);

    removeFile(aFile);
    writeToFile(aFile, result);

    // add default image as resource to the AXMEDIS object
    var res = new AxResource();
    res.load(default_image_path);
    res.contentID = res.contentID+mimeTypeToExt(res.mimeType);
    axobj.addContent( res );
  }
}

// Get recursively all files in a folder and its subfolders, adding them to axobj asd reosurcesbrowseFolders(axobj, folder,
indent)
{
  //print( indent + "folder =" + folder  );
  var f = getFirstFile( folder );

  while(f!=null)
  {
    //print(indent + f);
    // add f as resource
    var res = new AxResource();
    // workaround for bad images import
    if (f.indexOf(".htm")!=-1)
    {
      flattenImagesIn(f,axobj);
    }
    else if (f.indexOf("imsmanifest.xml")!=-1)
    {
      print ("Found manifest file");
      parseManifest(f,axobj) ;
    }
    res.load(f);
    res.contentID = res.contentID+mimeTypeToExt(res.mimeType);
    axobj.addContent( res );
```

```
      f = getNextFile();
    }
    var l = listDir(folder); //It provides the list of folders
    for (i in l)
    {
      browseFolders(axobj, l[i], indent + "  ");
    }
}
test()
{
 var indexpage = "<html><head><title>Index of contents</title></head><body><h1>Contents</h1><ul>";

 var l = listDir(startfolder); //It provides the list of folders

 for (i in l)
 {
   var aContentDir = l[i];
   // create new empty AXMEDIS object
   var axobj = new AXMEDISObject();
   var contid =  aContentDir.substring( aContentDir.lastIndexOf("\\")+1, aContentDir.length);
   print("contid = "+contid);
   // add resources
   browseFolders(axobj, aContentDir,  "  ");
   indexpage +="<li><a href=\""+ contid + ".axm\" title=\""+titleOfCurrentContent +"\">"+titleOfCurrentContent+"</a></li>";
   var savepath = destinationfolder + contid + ".axm";
   print("  -> saved "+ savepath);
   axobj.save(savepath);
   delete(axobj);
 }
 indexpage += "</ul></body></html>";
 writeToFile(destinationfolder +"index.html", indexpage);

 return true;
}
titleOfCurrentContent = "";();]]>
      </JS_Script>
      <JS_Script name="JScript(2)">

        <![CDATA[////////// JSXML XML Tools              ////////////////////
////////// Ver 1.2 Jun 18 2001          ////////////////////
////////// Copyright 2000 Peter Tracey   ////////////////////
////////// http://jsxml.homestead.com/   ////////////////////
////
////:
////
////
////    Regular Expression-based XML parser
////
////
////    Iterates through the tree structure without recursion
////
////
////    Loads xml into a linear structure and provides
////for adding and removing elements
////setting attributes, generates XML
////
////functions:
////
////
////    Takes string of attibutes and attribute name
////    Returns attribute value
////
////_Remove
////    Removes element in array
////
////_Add
////    Adds element to array
////
////
////    Repeats string specified number of times
////
//////////////////////////////////////////////////////////////REXML(XML) {
.XML = XML;
.rootElement = null;
```

```
.parse = REXML_parse;
(this.XML && this.XML != "") this.parse();
}
REXML_parse() {
   reTag = new RegExp("<([^>/ ]*)([^>]*)>","g"); // matches that tag name $1 and attribute string $2
   reTagText = new RegExp("<([^>/ ]*)([^>]*)>([^<]*)","g"); // matches tag name $1, attribute string $2, and text $3
   strType = "";
   strTag = "";
   strText = "";
   strAttributes = "";
   strOpen = "";
   strClose = "";
   iElements = 0;
   xmleLastElement = null;
   (this.XML.length == 0) return;
   arrElementsUnparsed = this.XML.match(reTag);
   arrElementsUnparsedText = this.XML.match(reTagText);
   i=0;
   (arrElementsUnparsed[0].replace(reTag, "$1") == "?xml") i++;
   (; i<arrElementsUnparsed.length; i++) {
        = arrElementsUnparsed[i].replace(reTag,"$1");
        = arrElementsUnparsed[i].replace(reTag,"$2");
        = arrElementsUnparsedText[i].replace(reTagText,"$3").replace(/[\r\n\t ]+/g, " "); // remove white space
        = "";
        (strTag.indexOf("![CDATA[") == 0) {
             = "<![CDATA[";
             = "";
             = "cdata";
             } else if (strTag.indexOf("!--") == 0) {
             = "<!--";
             = "-->";
             = "comment";
             } else if (strTag.indexOf("?") == 0) {
             = "<?";
             = "?>";
             = "pi";
             } else strType = "element";
        (strClose != "") {
             = "";
             (arrElementsUnparsedText[i].indexOf(strClose) > -1) strText = arrElementsUnparsedText[i];
             {
                  (; i<arrElementsUnparsed.length && arrElementsUnparsedText[i].indexOf(strClose) == -1; i++) {
                       += arrElementsUnparsedText[i];
                       }
                  += arrElementsUnparsedText[i];
                  }
             (strText.substring(strOpen.length, strText.indexOf(strClose)) != "")    {
                  .childElements[xmleLastElement.childElements.length] = new REXML_XMLElement(strType,
"","",xmleLastElement,strText.substring(strOpen.length, strText.indexOf(strClose)));
                  (strType == "cdata") xmleLastElement.text += strText.substring(strOpen.length, strText.indexOf(strClose));
                  }
             (strText.indexOf(strClose)+ strClose.length < strText.length) {
                  .childElements[xmleLastElement.childElements.length] = new REXML_XMLElement("text",
"","",xmleLastElement,strText.substring(strText.indexOf(strClose)+ strClose.length, strText.length));
                  (strType == "cdata") xmleLastElement.text += strText.substring(strText.indexOf(strClose)+ strClose.length,
strText.length);
                  }
             ;
             }
        (strText.replace(/ */, "") == "") strText = "";
        (arrElementsUnparsed[i].substring(1,2) != "/") {
             (iElements == 0) {
                  = this.rootElement = new REXML_XMLElement(strType, strTag,strAttributes,null,strText);
                  ++;
                  (strText != "") xmleLastElement.childElements[xmleLastElement.childElements.length] = new
REXML_XMLElement("text", "","",xmleLastElement,strText);
                  } else if (arrElementsUnparsed[i].substring(arrElementsUnparsed[i].length-2,arrElementsUnparsed[i].length-1) !=
"/") {
                  = xmleLastElement.childElements[xmleLastElement.childElements.length] = new
REXML_XMLElement(strType, strTag,strAttributes,xmleLastElement,"");
                  ++;
                  (strText != "") {
                       .text += strText;
```

```
                    .childElements[xmleLastElement.childElements.length] = new REXML_XMLElement("text",
"","",xmleLastElement,strText);
                    }
                } else {
                .childElements[xmleLastElement.childElements.length] = new REXML_XMLElement(strType,
strTag,strAttributes,xmleLastElement,strText);
                (strText != "") xmleLastElement.childElements[xmleLastElement.childElements.length] = new
REXML_XMLElement("text", "","",xmleLastElement,strText);
                }
            } else {
            = xmleLastElement.parentElement;
            -;
            (xmleLastElement && strText != "") {
                .text += strText;
                .childElements[xmleLastElement.childElements.length] = new REXML_XMLElement("text",
"","",xmleLastElement,strText);
                }
            }
        }
    }
REXML_XMLElement(strType, strName, strAttributes, xmlParent, strText) {
    .type = strType;
    .name = strName;
    .attributeString = strAttributes;
    .attributes = null;
    .childElements = new Array();
    .parentElement = xmlParent;
    .text = strText; // text of element
    .getText = REXML_XMLElement_getText; // text of element and child elements
    .childElement = REXML_XMLElement_childElement;
    .attribute = REXML_XMLElement_attribute;
    }
    REXML_XMLElement_getText() {
        (this.type == "text" || this.type == "cdata") {
            this.text;
            } else if (this.childElements.length) {
            L = "";
            (var i=0; i<this.childElements.length; i++) {
                += this.childElements[i].getText();
                }
            L;
            } else return "";
        }

    REXML_XMLElement_childElement(strElementName) {
        (var i=0; i<this.childElements.length; i++) if (this.childElements[i].name == strElementName) return this.childElements[i];
        null;
        }
    REXML_XMLElement_attribute(strAttributeName) {
        (!this.attributes) {
            reAttributes = new RegExp(" ([^= ]*)="","g"); // matches attributes
            (this.attributeString.match(reAttributes) && this.attributeString.match(reAttributes).length) {
                arrAttributes = this.attributeString.match(reAttributes);
                (!arrAttributes.length) arrAttributes = null;
                for (var j=0; j<arrAttributes.length; j++) {
                    [j] = new Array(
                            (arrAttributes[j]+"").replace(/[= ]/g,""),
                        (this.attributeString, (arrAttributes[j]+"").replace(/[= ]/g,""))
                                    );
                    }
                .attributes = arrAttributes;
                }
            }
        (this.attributes) for (var i=0; i<this.attributes.length; i++) if (this.attributes[i][0] == strAttributeName) return
this.attributes[i][1];
        "";
        }JSXMLBuilder() {
.XML = "";
.elements = new Array();
.prototype.remove = Array_Remove;
.prototype.add = Array_Add;
.load = JSXMLBuilder_load;
.element = JSXMLBuilder_element;
.addElementAt = JSXMLBuilder_addElementAt;
```

```
.insertElementAt = JSXMLBuilder_insertElementAt;
.removeElement = JSXMLBuilder_removeElement;
.generateXML = JSXMLBuilder_generateXML;
.moveElement = JSXMLBuilder_moveElement;
}
JSXMLBuilder_load(strXML, xmleElem) {
  .XML = strXML;
  (!xmleElem) {
      (strXML.length) xmleElem = (new REXML(strXML)).rootElement;
      return false;
      }
  xmlBuilder = new JSXMLIterator(xmleElem);
  (true) {
      (xmlBuilder.xmleElem.type == "element") {
          (xmlBuilder.xmleElem.attributes) {
              .addElementAt(xmlBuilder.xmleElem.name,xmlBuilder.xmleElem.attributes, xmlBuilder.xmleElem.text,
this.elements.length, xmlBuilder.iElemLevel);
              } else {
              .addElementAt(xmlBuilder.xmleElem.name,xmlBuilder.xmleElem.attributeString, xmlBuilder.xmleElem.text,
this.elements.length, xmlBuilder.iElemLevel);
              }
          }
      (!xmlBuilder.getNextNode(false)) break;
      }
  (var i=0; i<this.elements.length; i++) this.elements[i].index = i;
  }
JSXMLBuilder_element(iIndex) {
  this.elements[iIndex];
  }
JSXMLBuilder_addElementAt(strElement,Attributes,strText,iElemIndex,iElemLevel) {
  = parseInt(iElemIndex);
  = parseInt(iElemLevel);
  (iElemIndex < 0 || typeof(iElemIndex) != "number" || isNaN(iElemIndex)) iElemIndex = (this.elements.length>0) ?
this.elements.length-1 : 0;
  (iElemLevel < 0 || typeof(iElemLevel) != "number" || isNaN(iElemLevel)) iElemLevel = this.elements[iElemIndex-1].level;
  (!Attributes) Attributes = "";
  Elem = new Array();
  iAddIndex = iElemIndex;
  (iElemIndex > 0) {
      (var i=iElemIndex; i<this.elements.length; i++) if (this.elements[i].level > iElemLevel) iAddIndex++;
      if (this.elements[i].level <= this.elements[iElemIndex].level) break;
      = new JSXMLBuilder_XMLElement(strElement,Attributes,strText,iElemLevel+1,this);
      } else {
      = new JSXMLBuilder_XMLElement(strElement,Attributes,strText,1,this);
      }
  .elements = this.elements.add(iAddIndex,Elem);
  (var i=iAddIndex; i<this.elements.length; i++) this.elements[i].index = i;
  }
JSXMLBuilder_insertElementAt(strElement,Attributes,strText,iElemIndex,iElemLevel) {
  = parseInt(iElemIndex);
  = parseInt(iElemLevel);
  (iElemIndex < 0 || typeof(iElemIndex) != "number" || isNaN(iElemIndex)) iElemIndex = (this.elements.length>0) ?
this.elements.length-1 : 0;
  (iElemLevel < 0 || typeof(iElemLevel) != "number" || isNaN(iElemLevel)) iElemLevel = this.elements[iElemIndex-1].level;
  (!Attributes) Attributes = "";
  Elem = null;
  iAddIndex = iElemIndex;
  (iElemIndex > 0 && iElemLevel > 0) {
      = new JSXMLBuilder_XMLElement(strElement,Attributes,strText,iElemLevel+1,this);
      } else {
      = new JSXMLBuilder_XMLElement(strElement,Attributes,strText,1,this);
      }
  .elements = this.elements.add(iAddIndex,Elem);
  (var i=iAddIndex; i<this.elements.length; i++) this.elements[i].index = i;
  }
JSXMLBuilder_removeElement(iElemIndex) {
  = parseInt(iElemIndex);
  (var iAfterElem=iElemIndex+1; iAfterElem<this.elements.length; iAfterElem++) if (this.elements[iAfterElem].level <
this.elements[iElemIndex].level+1) break;
  .elements = this.elements.slice(0,iElemIndex).concat(this.elements.slice(iAfterElem,this.elements.length));
  (var i=iElemIndex; i<this.elements.length; i++) this.elements[i].index = i;
  }
JSXMLBuilder_moveElement(iElem1Index,iElem2Index) {
  arrElem1Elements = new Array(this.elements[iElem1Index]);
```

```
  arrElem2Elements = new Array(this.elements[iElem2Index]);
  (var i=iElem1Index; i<this.elements.length; i++) if (this.elements[i].level > this.elements[iElem1Index].level)
arrElem1Elements[arrElem1Elements.length] = this.elements[i]; else if (i>iElem1Index) break;
  (var i=iElem2Index; i<this.elements.length; i++) if (this.elements[i].level > this.elements[iElem2Index].level)
arrElem2Elements[arrElem2Elements.length] = this.elements[i]; else if (i>iElem2Index) break;
  arrMovedElements = new Array();
  (iElem1Index < iElem2Index) {
      (i=0; i<iElem1Index; i++) arrMovedElements[arrMovedElements.length] = this.elements[i]; // start to the 1st element
      (i=iElem1Index+arrElem1Elements.length; i<iElem2Index+arrElem2Elements.length; i++)
arrMovedElements[arrMovedElements.length] = this.elements[i]; // end of 1st element to end of 2nd element
      (i=0; i<arrElem1Elements.length; i++) arrMovedElements[arrMovedElements.length] = arrElem1Elements[i]; // 1st
element and all child elements
      (i=iElem2Index+arrElem2Elements.length; i<this.elements.length; i++) arrMovedElements[arrMovedElements.length] =
this.elements[i]; // end of 2nd element to end
      .elements = arrMovedElements;
      } else {
      (i=0; i<iElem2Index; i++) arrMovedElements[arrMovedElements.length] = this.elements[i]; // start to the 2nd element
      (i=0; i<arrElem1Elements.length; i++) arrMovedElements[arrMovedElements.length] = arrElem1Elements[i]; // 1st
element and all child elements
      (i=iElem2Index; i<iElem1Index; i++) arrMovedElements[arrMovedElements.length] = this.elements[i]; // 2nd element to
1st element
      (i=iElem1Index+arrElem1Elements.length; i<this.elements.length; i++) arrMovedElements[arrMovedElements.length] =
this.elements[i]; // end of 1st element to end
      .elements = arrMovedElements;
      }
  (var i=0; i<this.elements.length; i++) this.elements[i].index = i;
  }
JSXMLBuilder_generateXML(bXMLTag) {
  strXML = "";
  arrXML = new Array();
  (bXMLTag) strXML += '<?xml version="1.0"?>\n\n'
  (var i=0; i<this.elements.length; i++) {
      += RepeatChar("\t",this.elements[i].level-1);
      += "<" + this.element(i).name // open tag
      (this.element(i).attributes) {
          (var j=0; j<this.element(i).attributes.length; j++) { // set attributes
              (this.element(i).attributes[j]) {
                  += ' ' + this.element(i).attributes[j][0] + '="' + this.element(i).attributes[j][1] + '"';
                  }
              }
          } else strXML += this.element(i).attributeString.replace(/[\/>]$/gi, "");
      (((this.elements[i+1] && this.elements[i+1].level <= this.elements[i].level) || // next element is a lower or equal to
              (!this.elements[i+1] && this.elements[i-1])) // no next element, previous element
              && this.element(i).text == "") {
          += "/";
          }
      += ">";
      (this.element(i).text != "") strXML += this.element(i).text;
      strXML += "\n";
      (((this.elements[i+1] && this.elements[i+1].level <= this.elements[i].level) || // next element is a lower or equal to
              (!this.elements[i+1] && this.elements[i-1])) // no next element, previous element
              && this.element(i).text != "") strXML += "</" + this.element(i).name + ">\n";
      (!this.elements[i+1]) {
          = i;
          (var j=i; j>-1; j--) {
              (this.elements[j].level >= this.elements[i].level) continue;
              {
                  (this.elements[j].level < this.elements[lastelem].level) {
                      += RepeatChar("\t",this.elements[j].level-1) + "</" + this.element(j).name + ">\n";
                      = j;
                      }
                  }
              }
          } else {
          (this.elements[i+1].level < this.elements[i].level) {
              = i;
              (var j=i; this.elements[j].level>=this.elements[i+1].level; j--) {
                  (this.elements[i] && this.elements[j] && this.elements[j].level < this.elements[i].level && this.elements[j].level
< this.elements[lastelem].level) {
                      += RepeatChar("\t",this.elements[j].level-1) + "</" + this.element(j).name + ">\n";
                      = j;
                      }
                  }
              }
```

```
            }
        (strXML.length > 1000) {
            [arrXML.length] = strXML;
            = "";
            }
        }
    [arrXML.length] = strXML;
    arrXML.join("");
    }
JSXMLBuilder_XMLElement(strName,Attributes,strText,iLevel,xmlBuilder) {
    .type = "element";
    .name = strName;
    .attributes = (typeof(Attributes) != "string") ? Attributes : null;
    .attributeString = (typeof(Attributes) == "string") ? Attributes : "";
    .text = strText;
    .level = iLevel;
    .index = -1;
    .xmlBuilder = xmlBuilder;
    .parseAttributes = JSXMLBuilder_XMLElement_parseAttributes;
    .attribute = JSXMLBuilder_XMLElement_attribute;
    .setAttribute = JSXMLBuilder_XMLElement_setAttribute;
    .removeAttribute = JSXMLBuilder_XMLElement_removeAttribute;
    .parentElement = JSXMLBuilder_XMLElement_parentElement;
    .childElement = JSXMLBuilder_XMLElement_childElement;
    }
    JSXMLBuilder_XMLElement_parseAttributes() {
        (!this.attributes) {
            reAttributes = new RegExp(" ([^= ]*)="","g"); // matches attributes
            (this.attributeString.match(reAttributes) && this.attributeString.match(reAttributes).length) {
                arrAttributes = this.attributeString.match(reAttributes);
                (!arrAttributes.length) arrAttributes = null;
                for (var j=0; j<arrAttributes.length; j++) {
                    [j] = new Array(
                            (arrAttributes[j]+"").replace(/[= ]/g,""),
                        (this.attributeString, (arrAttributes[j]+"").replace(/[= ]/g,""))
                                    );
                    }
                .attributes = arrAttributes;
                }
            }
        }

    JSXMLBuilder_XMLElement_attribute(AttributeName) {
        (!this.attributes) this.parseAttributes();
        (this.attributes) for (var i=0; i<this.attributes.length; i++) if (this.attributes[i][0] == AttributeName) return
this.attributes[i][1];
        "";
        }
    JSXMLBuilder_XMLElement_setAttribute(AttributeName,Value) {
        (!this.attributes) this.parseAttributes();
        (this.attributes) for (var i=0; i<this.attributes.length; i++) if (this.attributes[i][0] == AttributeName) {
            .attributes[i][1] = Value;
            ;
            }
        .attributes[this.attributes.length] = new Array(AttributeName,Value);
        }
    JSXMLBuilder_XMLElement_removeAttribute(AttributeName,Value) {
        (!this.attributes) this.parseAttributes();
        (this.attributes) for (var i=0; i<this.attributes.length; i++) if (this.attributes[i][0] == AttributeName) {
            .attributes = this.attributes.remove(i);
            ;
            }
        }
    JSXMLBuilder_XMLElement_parentElement() {
        (var i=this.index; this.xmlBuilder.element(i) && this.xmlBuilder.element(i).level != this.level-1; i--);
        this.xmlBuilder.element(i);
        }
    JSXMLBuilder_XMLElement_childElement(Child) {
        iFind = -1;
        (var i=this.index+1; i<this.xmlBuilder.elements.length; i++) {
            (this.xmlBuilder.elements[i].level == this.level+1) {
                ++;
                (iFind == Child || this.xmlBuilder.elements[i].name == Child) return this.xmlBuilder.elements[i];
                } else if (this.xmlBuilder.elements[i].level <= this.level) break;
```

```
            }
        null;
        }JSXMLIterator(xmleElem) {
.xmleElem = xmleElem;

.iElemIndex = 0;
.arrElemIndex = new Array(0);
.iElemLevel = 0;
.iElem = 0;
.arrElemIndex[this.iElemLevel] = -1;
.getNextNode = JSXMLIterator_getNextNode;
}
JSXMLIterator_getNextNode() {
   (!this.xmleElem || this.iElemLevel<0)false;
   (this.xmleElem.childElements.length) {  // move up
        .arrElemIndex[this.iElemLevel]++;
        .iElemIndex++;
        .iElemLevel++;
        .arrElemIndex[this.iElemLevel] = 0;
        .xmleElem = this.xmleElem.childElements[0];
        } else { // move next
        .iElemIndex++;
        .arrElemIndex[this.iElemLevel]++;
        (this.xmleElem.parentElement && this.xmleElem.parentElement.childElements.length &&
this.arrElemIndex[this.iElemLevel] < this.xmleElem.parentElement.childElements.length) this.xmleElem =
this.xmleElem.parentElement.childElements[this.arrElemIndex[this.iElemLevel]];
        {
            (this.iElemLevel>0) { // move down
                (; this.iElemLevel > 0; this.iElemLevel--) {
                    (this.xmleElem.parentElement &&
this.xmleElem.parentElement.childElements[this.arrElemIndex[this.iElemLevel]]) {
                        .xmleElem = this.xmleElem.parentElement.childElements[this.arrElemIndex[this.iElemLevel]];
                        .iElemLevel++;
                        .arrElemIndex = this.arrElemIndex.slice(0,this.iElemLevel+1);
                        ;
                        } else {
                        .xmleElem = this.xmleElem.parentElement;
                        }
                    }
                .iElemLevel--;
                } else {
                false;
                }
            }
        }
   (typeof(this.xmleElem) == "object" && this.iElemLevel > -1);
   }ParseAttribute(str,Attribute) {
str = str +  ">";
(str.indexOf(Attribute + "=")>-1) var Attr = new RegExp(".*" + Attribute + "='([^']*)'.*>");
if (str.indexOf(Attribute + '=')>-1) var Attr = new RegExp(".*" + Attribute + '="([^"]*)".*>');
str.replace(Attr, "$1");
}Array_Remove(c) {
tmparr = new Array();
(var i=0; i<this.length; i++) if (i!=c) tmparr[tmparr.length] = this[i];
tmparr;
}Array_Add(c, cont) {
tmparr = new Array();
(var i=0; i<this.length; i++) {
   (i==c) tmparr[tmparr.length] = cont;
   [tmparr.length] = this[i];
   }
(!tmparr[c]) tmparr[c] = cont;
tmparr;
}RepeatChar(sChar,iNum) {
L = "";
(var i=0; i<iNum; i++) L += sChar;
L;
  }]]>
      </JS_Script>
    </Rule_Body>
    <Dependencies/>
  </AXCP_Rule>
 </Definition>
</Rule>
```

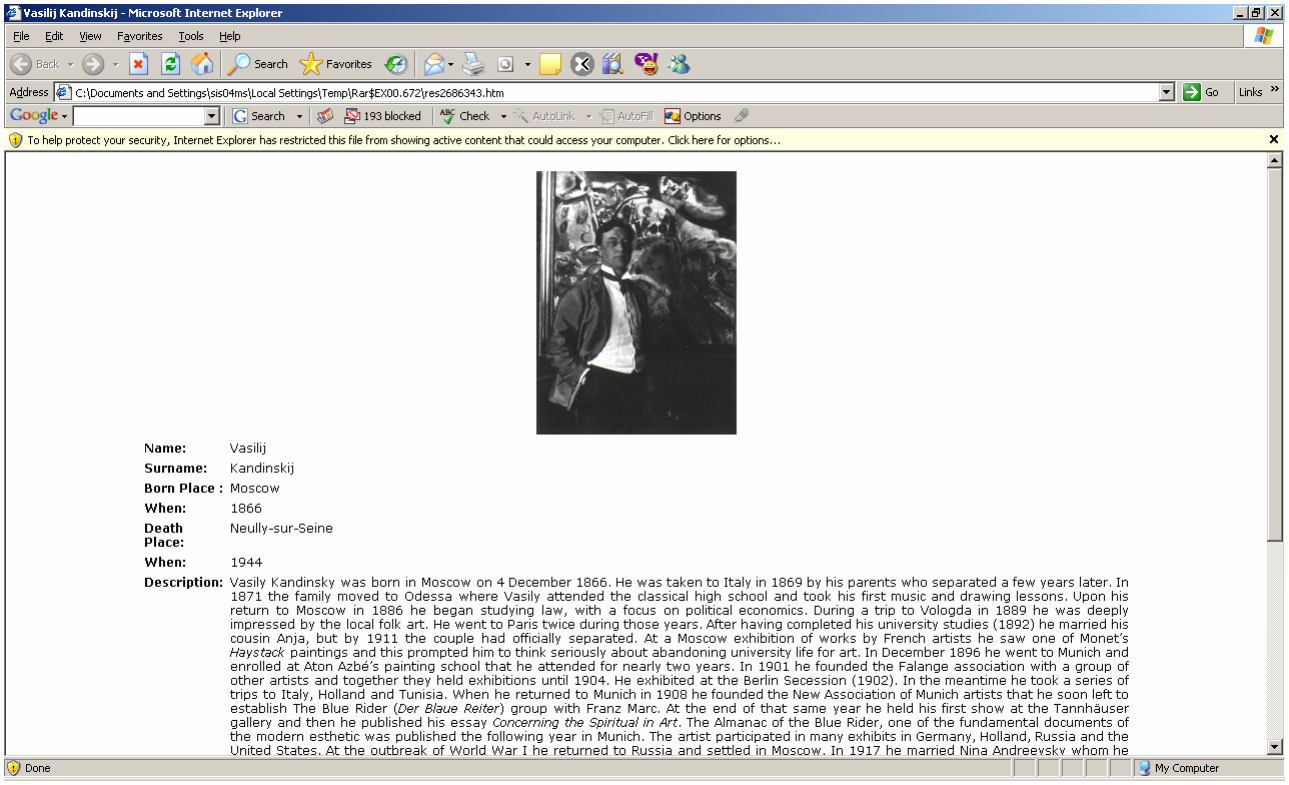

Figure 2.10: The original IMS Learning Object

The following is the AXMEDIS Object produced by executing the rule on the IMS Object Above.

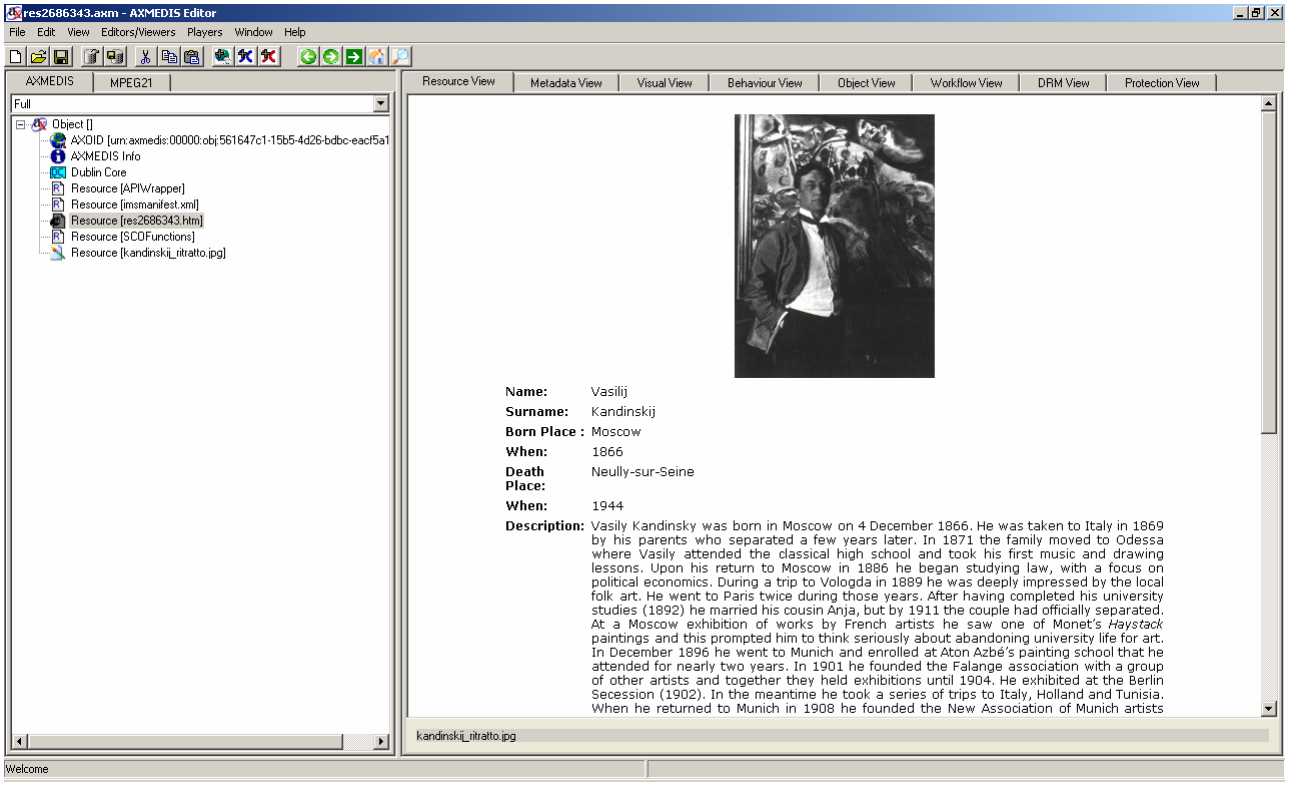

Figure 2.11: AXMEDIS Object produced from the IMS Object

3.4 Integration of ILABS Content Sources to the AXMEDIS Model (ILABS)

The present section focuses on the integration between AXMEDIS system and *learn eXact® Mobile* (LEX-Mobile). The AXMEDIS system has been described in other relevant documents (reported also in the reference section) it is now time to provide a description of LEX system. LEX is a proprietary environment developed in GIUNTI Interactive Labs during the past ten years and devoted to cover the whole value chain of e-learning publishing and fruition in full compliancy with international standards. The Mobile edition allows management of delivery to mobile devices. Basically it comprises the following elements:

- *Packager©* - the authoring and packaging facility of the *learn eXact®* platform;
- *Lobster©* - the data manager that handles all published content of the *learn eXact®* platform;
- *Siter©* - the e-learning management module of the *learn eXact®* platform;
- *Glove©* - the e-learning rendering and fruition module of the *learn eXact®* platform;
- D-Repository - any repository compliant to the *Digital Repository* standard (WebCT, BlackBoard…)
- Back-end - any existing backend compliant with the set of API exposed by *learn eXact®* platform.

The most relevant aspect in the D-Repository connection is the possibility of using their content either as source or destination of the packaging process as all that is needed is the compatibility with IMS packaging standard. This feature was developed originally to grant users the possibility of continuing to exploit the already existing content sources or distribution facilities and infrastructure even in the case of a shift in either production or delivery framework. T he same principle is brought to the development of a set of API for interconnecting with external back-ends; this is actually the way that will be followed to allow AXMEDIS empowered production and distribution value chain to be exploited. As stated in the TA it is expected that each distributor may keep on using own distribution media, but should also take advantage of the powerful support provided by AXMEDIS. Given the layered structure of GIUNTI Authoring and Learning Environment, this has been possible operating at API level in the area previously generically marked as "backend" (for example SAP…). More specifically it has been necessary to define a set of new API to allow AXMEDIS co-operation with the new third party environment. The following schema explains this in more detail:

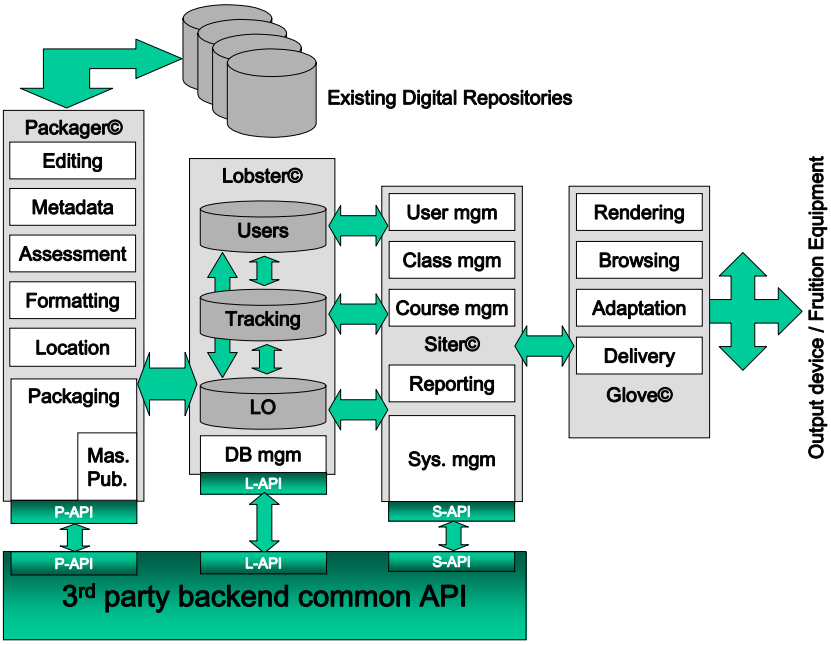

Figure 2.12: Block Diagram for Mobile Distribution

From the above diagram it is evident that the two environments always have a "mediated" interaction, and this is achieved through proper API sets. Having said this it is necessary to examine in more detail the level of functionalities covered by the various set of API in respect to the process value-chain. In the following

diagram the possible level of interaction between AXMEDIS as a backend and our environment is summarised. The set of common API allows interaction mainly at packaging, distribution and fruition level (at least as far as content delivery is concerned) even though the rendering tool (Glove) is designed only for e-learning purposes and therefore is not compatible with the overall aim of the project. On the other hand during the design and development phase the possibility of interaction is limited (especially at the design phase) to data import/export.

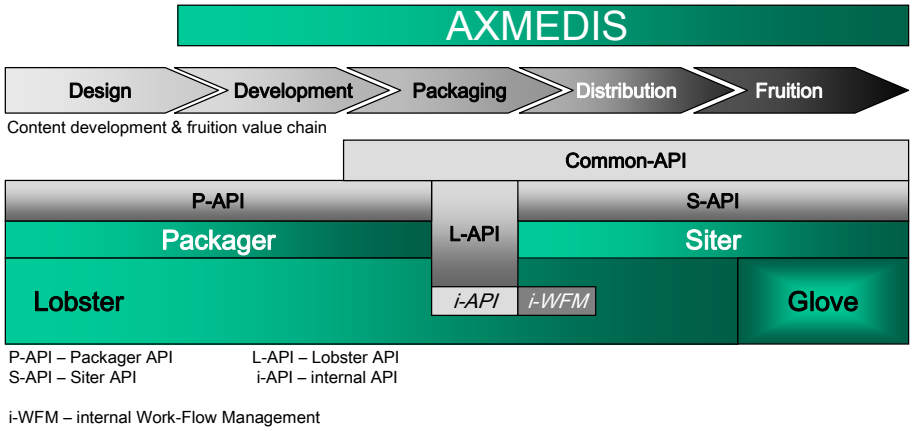

Figure 2.13: Global Workflow Management

From the above schema it is evident that the Lobster embeds a workflow management system used to handle cooperative editing of content. The packager can toggle, via internal API, the status of a package (LO(for Learning Object or course) between the status of "locked" and "unlocked" while the Siter can manage, via internal API, all states of the package. The internal API is implemented via web-service and it is not exposed to the backend. Having stated this it is now necessary to see how the overall architecture will look like at the end of the integration phase.

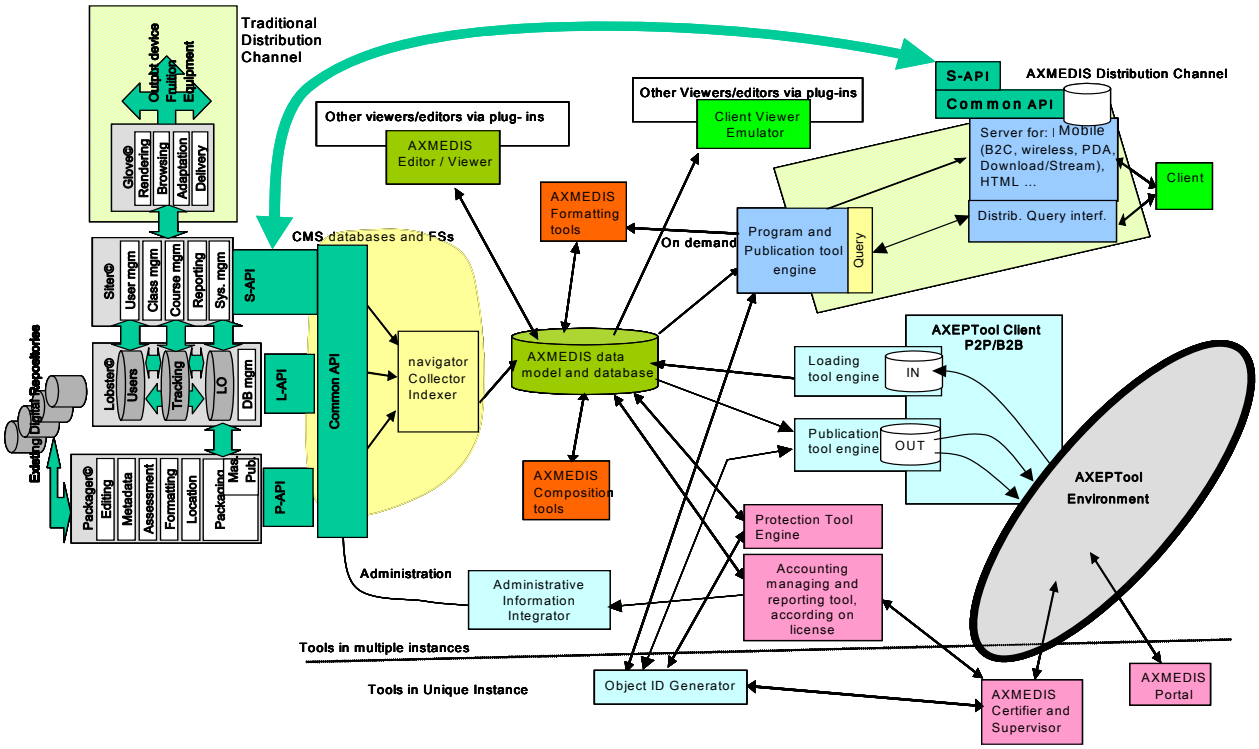

Figure 2.14: Global Architecture for Mobile Distribution

Interaction will basically occur at the level of the CMS and file system thanks to the 3rd party backend common API as highlighted in the picture above. It is worth mentioning that the schema is an adaptation of the one describing AXMEDIS architecture and reported both in the TA and in the framework specifications. The only relevant change is the one depicting the interaction between AXMEDIS (seen here as a backend) and the GIUNTI Authoring and Learning Environment.

4 State-Of-The-Art Device Update (UR, FUPF)

4.1 Trends in Digital Media Adaptation & Transcoding (UR)

Over the period the trend towards the adoption of the MPEG-21 standard re digital item adaptation has been more established and the relevant RTD in AXMEDIS has responded to that as described in [1].
The main thrust of the other developments in this area over the recent period has tended to focus on automated on-the-fly Service Level Agreement (SLA) negotiation and dynamic transcoding mainly to optimise network bandwidth utilisation and security protection. This is to enhance of the capability of telecommunication networks for secure traffic management on the one hand, and, to optimise tasking and load balancing of service provisioning in Grid-enabled networks.

All these trends are predicated on advancing the semantic expressiveness of the representations that together constitute the media delivery context in terms of a number of usage conditions relating to the network session and client device (both generic and individual) capabilities, and, user preferences including location-based requirements. As such there has been a continuing interest in enhancing the standards to allow more expressive integrative and open semantic parameterisation of the delivery context including the deployment of generalisation ontologies, for example as advocated earlier in [2], that would enable proxy-enabled SLA negotiation and advanced personalisation particularly in the context of security and trust services.
To the extent that the adoption of an appropriately partitioned and standards-compliant set of user preferences hierarchies is fundamental to all the above, our work in this area has built on the latest state-of – the art and extended it to provide increased semantic parameterisation of the media delivery context through adopting the MPEG-21 DIA for Digital Item Adaptation to which we have added additional semantic elements for our own profiling management to serve the Digital Media Adaptation, Transcoding Services to be provided by the AXMEDIS Framework

MPEG-21 is a multimedia standards framework devised to support a simplified, open and interoperable multimedia delivery and consumption chain and thus motivate an open market for content creators, producers, distributors, service providers and content consumers.
MPEG-21 is based on two essential concepts namely the definition of a fundamental unit of distribution and transaction (the Digital Item) and the concept of Users interacting with the Digital Items. Its goal is to define the technology needed to support Users to exchange, access, consume, trade and otherwise manipulate Digital Items in an efficient, transparent and interoperable way.
The Digital Item Adaptation (DIA) section is one of the 17 constituent parts of the MPEG-21 standard. This is elaborated by defining the syntax and semantics of the various aspects relevant to the media delivery context as deployed in our work in this period, the main elements of which are outlined below.

4.1.1 Digital Item Adaptation for Delivery Context Representation

MPEG-21 motivates the shielding of users from issues relating to network and terminal installation, management and implementation by offering interoperable transparent access to distributed advanced multimedia content. The MPEG-21-enabled vision is to enable on-demand adaptation of the Digital Item for delivery to client terminals so that multimedia content can be created and ubiquitously shared, always with the agreed/contracted quality, reliability and flexibility. Digital Items are subject to a resource adaptation engine, as well as a descriptor adaptation engine, which together produce the adapted Digital Items.
The target for this part of the standard is to specify tools that provide input to the adaptation engine, so that any constraints on the delivery and consumption of resources can be satisfied, and the quality of the user experience can be guaranteed.

Digital Item adaptation comprises of the User Characteristics, Terminal Capabilities, Network Characteristics, Natural Environment Characteristics, Resource Adaptability and Session Mobility.

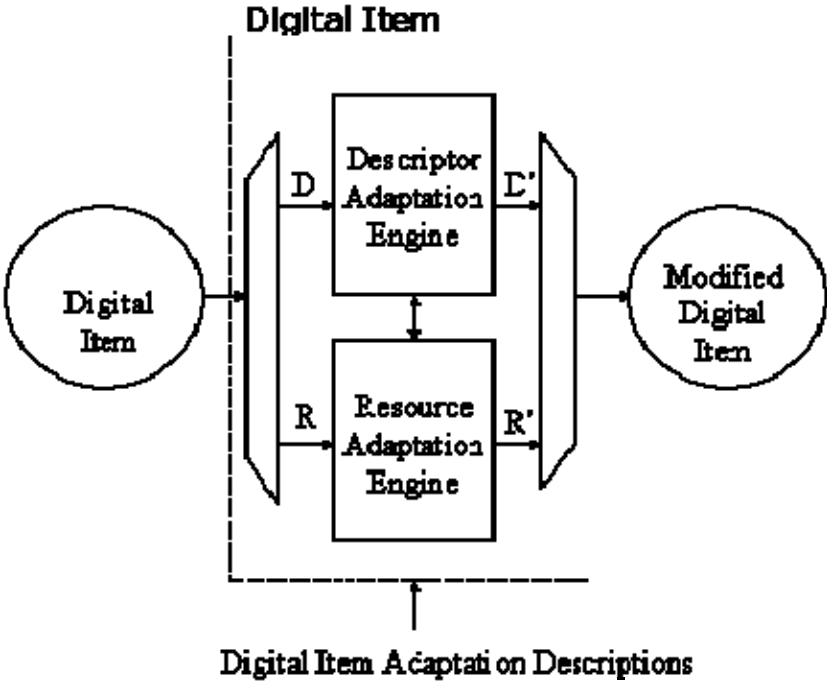

Figure 3.1: Digital Item Adaptation

MPEG-21 has provided a standard descriptor for multimedia content access and allowed standards-compatible technologies to be used for adapting multimedia content. Jannach et al [49] and Timmerer et al [50] have highlighted the importance of an adequate semantic framework, as incoroprated within MPEG21, to repreesent the profiling knowledge supporting Digiatl Items Adaptation and various aspects of multimedia delivery matchmaking. MPEG-21 seeks to achieve interoperable multimedia communication across networks and devices. It addresses device and format coding independence by standardising descriptors of usage environment and bit-stream syntax. Through detailed examples involving streaming of audio-video resources and adapting of images according to terminal capabilities, MPEG-21 solves the challenging and important problem of universal multimedia access.

Pellen et al [51] present a method to complement existing video, audio and image adaptation techniques in order to improve the perceived quality of interactive rich media applications in streaming scenarios. This approach involves dynamically adapting multimedia services at the presentation level according to media level adaptation decisions. Firstly, an enhanced decision-taking engine decides on the best presentation form based on client-side constraints. Then, presentation updates dynamically modify the user display according to fluctuating environment constraints. A fine-grain control of the synchronisation of these updates with adapted media streams is used to enable seamless switching of media bitstreams with different coding (transcoding) or switching to media streams with different modalities (transmoding). They finally describes an application test-bed using MPEG-4 and MPEG-21 to implement the proposed adaptation approach and illustrates the user experience improvement. Similarly Timerer et al [52] provide a description of their media daptation platform including the the Decision Taking Engine.

The seamless access to rich multimedia content on any device and over any network, usually known as Universal Multimedia Access, requires interoperable description tools and adaptation techniques to be developed. To address this, MPEG-21 introduces an Adaptation Framework, which provides several mechanisms for making adaptation decisions according to usage environment and adapting multimedia contents in a coding-format-independent way. Ransburg [53] gives an overview of the European FP6

project DANAE which not only implements and extends the existing MPEG-21 adaptation mechanisms but has also initiated several new standardisation activities in the area of dynamic and distributed adaptation and resource conversion. MPEG-21 DIP enables static stream selection which is a first step in a series of adaptations. The existing BSD-based adaptation mechanisms enable the efficient implementation of generic adaptation engines, which can be used for existing and future coding formats. These mechanisms were extended to enable dynamic and distributed adaptation. As an alternative to the BSD-based adaptation, resource conversion was investigated which does not rely on scalable media and allows adaptations at the scene level.
The paper esentially covers the three different adaptation mechanisms, na,ely, DIP-based static stream selection, resource conversion and BSD-based adaptation and shows how DANAE extends the BSD-based adaptation to distributed scenarios where the adaptation can happen anywhere along the delivery chain.

Pervasive computing applications allow users to access information from anywhere while traveling and using variety of devices. Heterogeneity and limitation of resources involved in this application demand adaptation of content according to the current context (device, user, network etc.). The dynamic nature of adaptation mechanisms together with emerging opportunities of Web Service technology provides a new approach o adaptation which is service-based. While this approach would provide a valuable service for the end customer, the service provider, and the content provider, it is important to have an architectural framework which is simple, scalable, flexible and interoperable. Moreover, in order to provide a complete service-based content negotiation and adaptation solution, we must have a model, or a tool, that allows defining environmental constraints, mapping them to appropriate adaptation service requirements and finding an optimal service configuration. Tong et al [54] present a service-based content adaptation architecture, enabling the use of third-party adaptation services and a novel content negotiation and adaptation model. The proposed architectural framework is validated through a prototype.

Due to the rapid pace of technological change, emerging standards and the constant evolution of mobile devices, the use of flexible and adaptable technology and methodologies is essential for pervasive applications. Berhe et al [55] set out the first goal of their proposed architecture as providing an all round solution to problems of content adaptation in wide area systems which were not well addressed by existing adaptation approaches. The second goal is to exploit the potential of emerging technologies such as Web Services for the purpose of adaptation. Web Services have demonstrated their flexibility, extensibility and interoperability; however their application for the purpose of content adaptation in a distributed environment has not been investigated. Berhre et al state that there project was initiated to develop distributed content adaptation as a solution that combines several of these technologies and extends them to alleviate the drawbacks of existing adaptation approaches. They conclude that by doing this, content adaptation can be developed as any value-added services provided by third party and used through subscription by users or content providers. Their longer term goal is developing the heuristics required to solve the equation system proposed for optimal path selection.

In order to cater to the diversity of terminals and networks, efficient, and flexible adaptation of multimedia content in the delivery path to the end consumers is required. To this end, it is necessary to associate the content with the metadata that provides the relationship between feasible adaptation choices and various media characteristics obtained as a function of these choices. Furthermore, adaptation is driven by the specification of terminal, network, user preference or rights-based constraints on media characteristics that are to be satisfied by the adaptation process. Using the metadata and the constraint specification, an adaptation engine can take an appropriate decision for adaptation, efficiently and flexibly. MPEG-21 Part 7 entitled Digital Item Adaptation standardises among other things the metadata and constraint specifications that act as interfaces to the decision-taking component of an adaptation engine. Mukhrejee et al [56] present the concepts behind these tools in the standard and shows universal methods based on pattern search to process the information in the tools to make decisions, and presents some adaptation use cases where these tools can be used to enable Terminal and Network QoS.

Casting the adaptation decision-taking problem as a generic constrained optimization problem involving adaptation variables has allowed Mukhrejee et al to aim for a universal decision-taking engines that takes

decisions based on terminal, network, and preference constraints, irrespective of their semantics and content type. In this way decision-taking by search over discrete variables covers the vast majority of practical adaptation scenarios existing today - some of these have been set out in the paper. Viable strategies when one or more variables are continuous are also presented in the context of exploring the management of continuous variable adaptation decisions

Simiarly H. Zhang [57] described a framework for adaptive content delivery in heterogeneous environments, aiming at improving content accessibility under changing network and viewer conditions. The framework included content adaptation schemes, client capability and network bandwidth discovery methods, and a Decision Engine for adaptation selection. These included the discovery modules for detecting client capabilities, user preferences and network characteristics, various content adaptation techniques to improve web accessibility and information delivery, and a general decision-making framework for optimising adaptive content delivery over the Internet.

In Conclusion Timerer et al [52] have been a leading contributor to the research in th area of Digital Item Adaptation and their work has motivated our approach within the Axmedis Multi media Adapatation Platform as described later in this document.

4.2 DRM Support and Interoperability (FUPF)

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

4.2.1 REL Profiles

An MPEG REL profile consists of a subset of the MPEG REL with some application oriented extensions in terms of its types, elements, attributes and authorisation model, and is usually defined as the result of first extending (when needed) and then profiling the language with the extensions. Extending the REL enables users of the REL to define new types, elements and attributes specific to their needs. This includes extending the REL schematic elements with new ones for new basic entities of the REL, namely, principal, right, resource and condition, to improve efficiency in a specific domain. Profiling the REL enables users to select only those language elements, as a subset of the language, required to meet a specific application need. This optimises payload of digital items and computation requirements of MPEG terminals. An MPEG REL profile, resulting from extending and profiling the REL, can thus be used to optimise the applicability of the REL to one specific application.

4.2.2 The REL MAM Profile

The MPEG-21 REL MAM profile is for distributing and consuming content delivered via channels including mobile communications and pre-recorded optical media to devices like mobile phones and high-definition DVD players.
To support applications in the mobile and optical media domains, the MAM profile extends the MPEG REL with the following elements:

- *Principals* identified with identifiers from a given identification system.
- *Rights* to copy and move digital resources according to pre-determined rules defined by some governance organisation.
- *Resources* protected by some form of (symmetric key and/or public key) encryption.
- *Conditions* to derive digital resources based on their relations with other resources when creating a collection of resources like a play-list and a customised movie, conditions to copy, move and output digital resources to other DRM systems, and conditions to seek dynamic permissions at the time of using digital resources.

The MAM profile consists of elements for:

- *Principals* that can be identified by either an identifier in some identification system or a cryptographic key.

- *Rights* to play, print and execute resources, to copy and move according to rules defined by a governance body, and to possess attribute properties (related to, say, subscribers or authorisation domains).
- *Resources* whose confidentiality and integrity are protected or not protected.
- *Conditions* related to time intervals, usage counts and territories, conditions to copy, move and output digital resources to other DRM systems, and conditions to seek dynamic permissions at the time of using digital resources.

The MAM profile supports usage models such as:

- Unrestricted usage
- Usage tied to individuals
- Preview
- Super-distribution
- Subscription based usage
- Usage within authorized domains

5 Work Done in Digital Item Adaptation (UR, DSI, DIPITA, EPFL, FHGIGD)

5.1 JSPROFILING (UR)

JSPROFILING consists of Java Script classes for User, Device and Network profiles which act as wrappers for corresponding C++ classes. These classes contain functions responsible for creating and managing the profiles with respect to different scenarios.

5.1.1 JSAxUserProfile

The user's profile captures and stores information related to the users, the usage environment and the user preferences. This profile should be managed at the personalisation server and be able to store minimal details about the user to identify his personal preferences in respect to the services being offered. The elements in this profile are the minimal set of elements that are required to meet the AXMEDIS User's personalisation needs. Each of these elements is based on the definitions provided by the MPEG21 schema [ISO MPEG-21, Part 7 - Digital Item Adaptation, ISO/IEC JTC1/SC29/WG11/N5231, and (Oct 2002)]. The schema for this profile is contained in the AXMEDIS-user-profile.xsd. The JSUserProfile is a JS class to automatically create and manage user profiles as a part of rule scripts for the AXCP engine.

The functionalities and values for the user profiles can be accessed through the methods given below. These functions are only wrappers for the functions in the C++ class UserProfile.

- UserProfile: Constructor for Initialising the Profile
- loadXmlFile: Loads the Xml from file
- loadXmlString: Loads Xml from an Xml String
- createXmlfile:Creates an Xml file where the file is specified as uri
- getXml: Returns the contents of the XML file in String form
- setXml: Writes the string as the contents of the XML file
- getAttribute: It returns the corresponding value of the attributes from the xml file. Attributes for the respective profiles are specified in the Profiles documentation provided in [1]
- setAttribute: Sets individual values to the attributes of the xml file
- getValue: Returns the corresponding value of the element from the xml file. Possible Values for the respective profiles are specified in the Profiles documentation provided in [1]
- setValue: Sets individual values to the elements of the xml file
- createElement: Creates an xml element using the DomPath and Element Name
- deleteElement: Deletes an xml element based on the DomPath
- close: Closes the xml file. If any element is changed then the changes are saved and then the xml file is closed.

5.1.2 JSAxDeviceProfile

The device profile captures information related to the user's device, that the service provider must take into account while delivering the multimedia contents. Generally the minimal set of elements required are those

that affect the proper utilisation of the contents on the device e.g. screen resolution for images. This profile represents the characteristics of the user's personal device and hence contains the most vital information for the content providers that can be used to transcode the content to match the device characteristics and can be better utilised by the user. Each of these elements is based on the definitions provided by MPEG21 schema [ISO MPEG-21, Part 7 - Digital Item Adaptation, ISO/IEC JTC1/SC29/WG11/N5231, and (Oct 2002)]. The schema for this profile is contained in the AXMEDIS-device-profile.xsd. The JSDeviceProfile is a JS-class to automatically create and manage Device profiles as a part of rule scripts for the AXCP engine.
The functionalities and values for the device profiles can be accessed through the methods similar to those outlined above for the User Profile. These functions are only wrappers for the functions in the c++ class DeviceProfile.

5.1.3 JSAxNetworkProfile

Network profile captures information related to the carrier that is used for delivery of the content to the user's device. It is necessary for the distributors to know various characteristics of the Network so as to provide the promised QoS. The information contained in this profile can also be used to for transcoding of the content to better utilise both network and device capabilities. The elements in this profile are the minimal set of elements that are required to meet the AXMEDIS User's personalisation needs and to meet the distributor's needs in terms of the network capabilities.
Each of these elements is based on the definitions provided by MPEG21 schema [ISO MPEG-21, Part 7 - Digital Item Adaptation, ISO/IEC JTC1/SC29/WG11/N5231, and (Oct 2002)].
The schema for this profile is contained in the AXMEDIS-network-profile.xsd. The JSNetworkProfile is a JS class to automatically create and manage user profiles as a part of rule scripts for the AXCP engine. The JS_NetworkProfile updates/retrieves the Network Profile stored in the database. The functionalities and values for the Network Profile can be accessed through methods similar to those outlined above for the User Profile. Essentially, 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, UAProf or MPEG-21 and thereafter invoking a suitable transcoding algorithm to adapt the delivery of the media to the client device.

Following extensive discussions within the AXMEDIS Consortium regarding User Profiling for Multimedia Distribution, it was concluded that the MPEG-21 standard should be adopted so as to accommodate all the possible scenarios. Accordingly each and every aspect of the MPEG-21 DIA was analysed in depth and suitable fields were adopted for the AXMEDIS profiles and in some cases some AXMEDIS specific values were added to the existing profile to enrich it to accommodate all AXMEDIS-related requirements. The schemas were defined in terms of three categories, namely User Profiles, Device Profiles and Network Profiles and were circulated amongst the partners for their feedback. This supported the schema specification for user profiling as finalised by way of the detailed schema specification as appeared in the document AXMEDIS-DE3-1-2-2-6-Spec-of-AX-Content-Processing-upC-v1-5). For the entire schema file please refer to the relevant xsd files. During this project period we have also been engaged in the process of developing the respective Javascript and C++ classes for creating and managing the profiles. The Javascript classes have been based on the mozilla spider monkey scripts and the C++ classes have been constructed as re-usable object models.

5.2 DIA Profile Management

In this last reporting period relating to this research, the extension of the MPEG 21-conformant profiling schema and the development of the AXMEDIS media Adaptation Decision Engine was undertaken to provide an integrated adaptation platform in conjunction with the AXCP JS Engine as will operate within the Axmedis Demonstrator. Work began with additional schema amendments followed by partitioning of the profile management to facilitate streamlined Adaptation Decision Taking according to the framework that will be described in the next section. The schematic below illustrates the various parametrics of the media adaptation process.

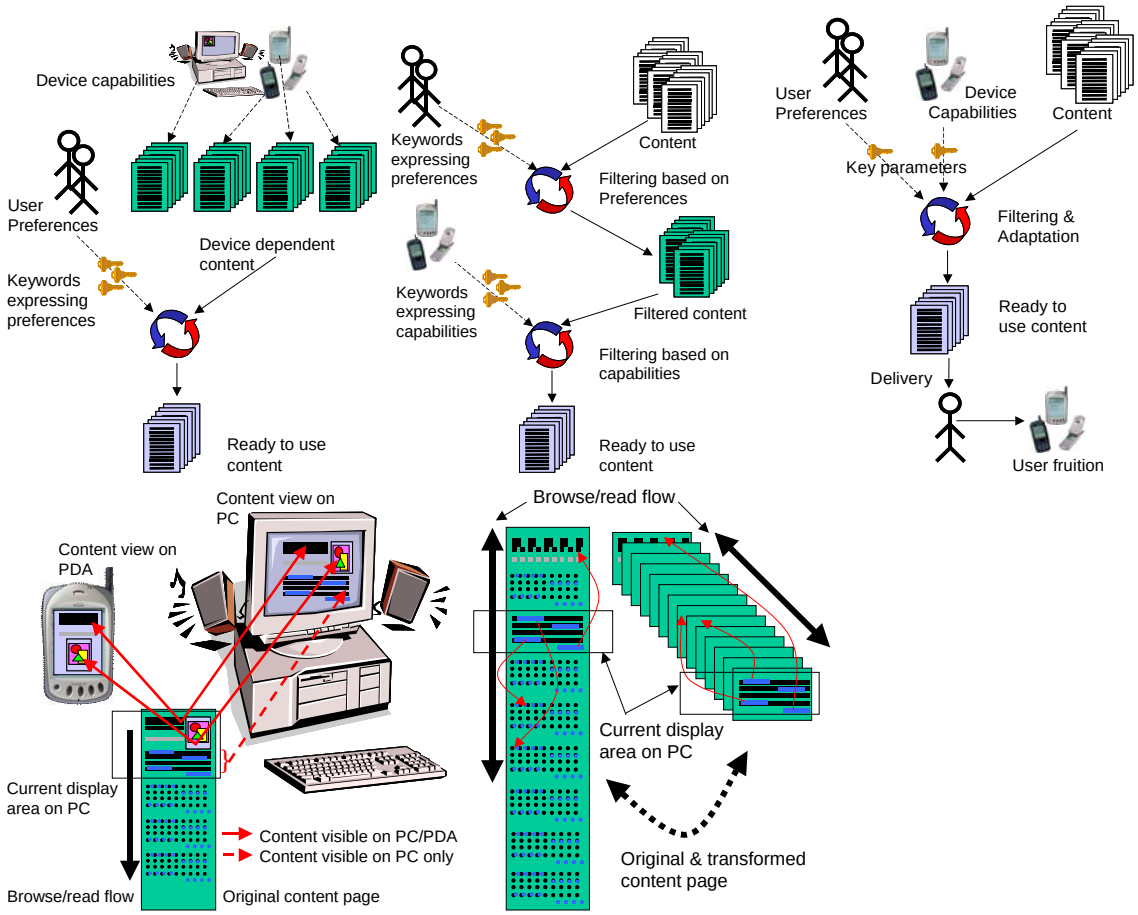

Figure 3.2 A Typical Media Adapatation Process Flow and Profiling Knowledge Basis (ILABS)

Ideally the Adaptation Decision Engine Platform would need to be able to rely on the following resources to be available to it in order to ensure optimally matched and efficient delivery of media in every case:

1. Relevant prototypical profiles (for user's media adaptation needs and preferences and the same for the client device and the network)
2. Target Media Resource Parameters (e.g. size, font etc)
3. Target delivery-context-sensitive Adaptation Guidelines
4. Fully operational Transcoding plug-ins for various media types e.g. audio, text, video, etc
5. Situated Delivery Context Over-rides (environmentally-triggered exceptions)
6. QoS Metrics or Accessibility Guidelines (see Appendix) that may optionally be provided by the client-side to be available for use in selection/rejection or refinement of the adapted resource.

However whilst the adapatation platform will not be able to operate in the absence of appropriate transcoding plug-ins for various media, it should still be able to intelligently manage the adaptation process should some of the profiling knowledge be unavailable. This in essence requires the invocation of appropriate context-sensitive heuristics to infer the opitimal media delivery template in the absence of full profiling information about the four categories of profiles as partitioned within the AXMEDIS DIA Profiling Management framework.

The needs and preferences of a user that may arise from the user's context or environment, the technical requirements of the user's device, the tools available (e.g., assistive technologies), the user's background, or a disability in the traditional sense, need to be considered while making decisions for match making of media delivery to the user and dynamic media adaptation to suit individualised purposes, perefrences and accessibility.

Essentially an AXMEDIS object can be analysed to be adapted to the destination device (in this case, a mobile device), in light of the preferences set out in the AXMEDIS User, AXMEDIS Terminal, AXMEDIS Network and AXMEDIS Natural Environment profiles.

Elements from these four profiles are grouped into four categories:

- audio-only elements that affect the presentation of audio on the device (e.g. number of audio channels),
- video-only elements that affect the presentation of video on the device (e.g. monochrome or colour display),
- text-only elements that affect the presentation of text on the device,
- general elements that specify general characteristics, regardless of media type (e.g. network bandwidth).

In this way, the approach to perform the required adaptation is intended to minimise the number of comparisons between parameters of User, Terminal, Network and Natural Environment profiles and the parameters of resource metadata.

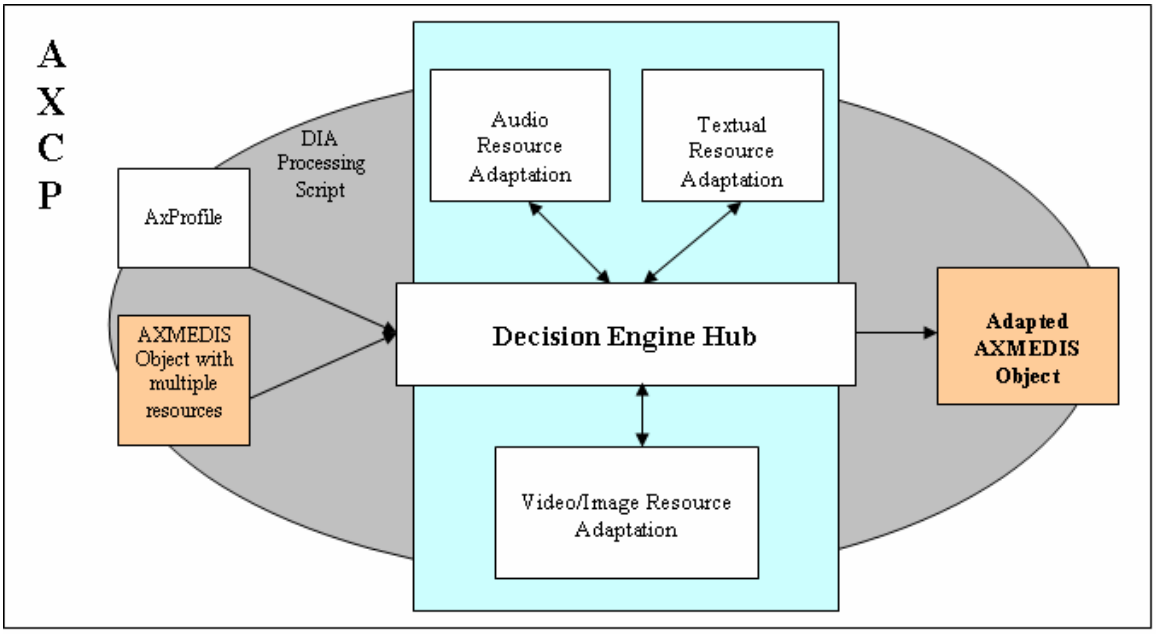

Figure 3.3 : DIA based on categorised profile elements

The input to the adaptation process may be a simple AXMEDIS object having a single resource for example, or a complex AXMEDIS object with multiple resources of various types such as audio, video, image, text etc.

This object is analysed and for each single resource within it, the functionalities provided for the adaptation process are invoked. These adaptation functionalities take the profiles as their input, and for each resource within the object, ascertain its type (audio, video, image etc) and based on the type, match them against the relevant profile elements. Accordingly if the resource is audio, its parameters are compared with a list of audio-related parameters (as well as General parameters, which apply to all kinds of resource types), in case of a video resource, the comparisons are made with video-related list of parameters as well as general

elements selected from the four AXMEDIS profiles; and so on. If there are mismatches in any characteristics of the resource (regarding presentation to the user on destination device) and the profile element values, then appropriate modification/adaptation is made and the target value is stored. If there are multiple parameters affecting the same object characteristics, then based on the priority an appropriate adaptation parameter is stored.

Once the analyses, comparisons and necessary modifications to the parameter values have been performed according to the elements of the profiles (categorised into Audio Only, Video Only, Text Only and General Elements, as mentioned above), transcoding algorithms are invoked for adaptation of the constituent resource(s) using the list of necessary modifications and adaptations to be performed on the resource to arrive at an adapted AXMEDIS object as output.
These functionalities can be accessed via JS in the AXCP engine to create and to manage the profiles dynamically and adapt objects (resources).

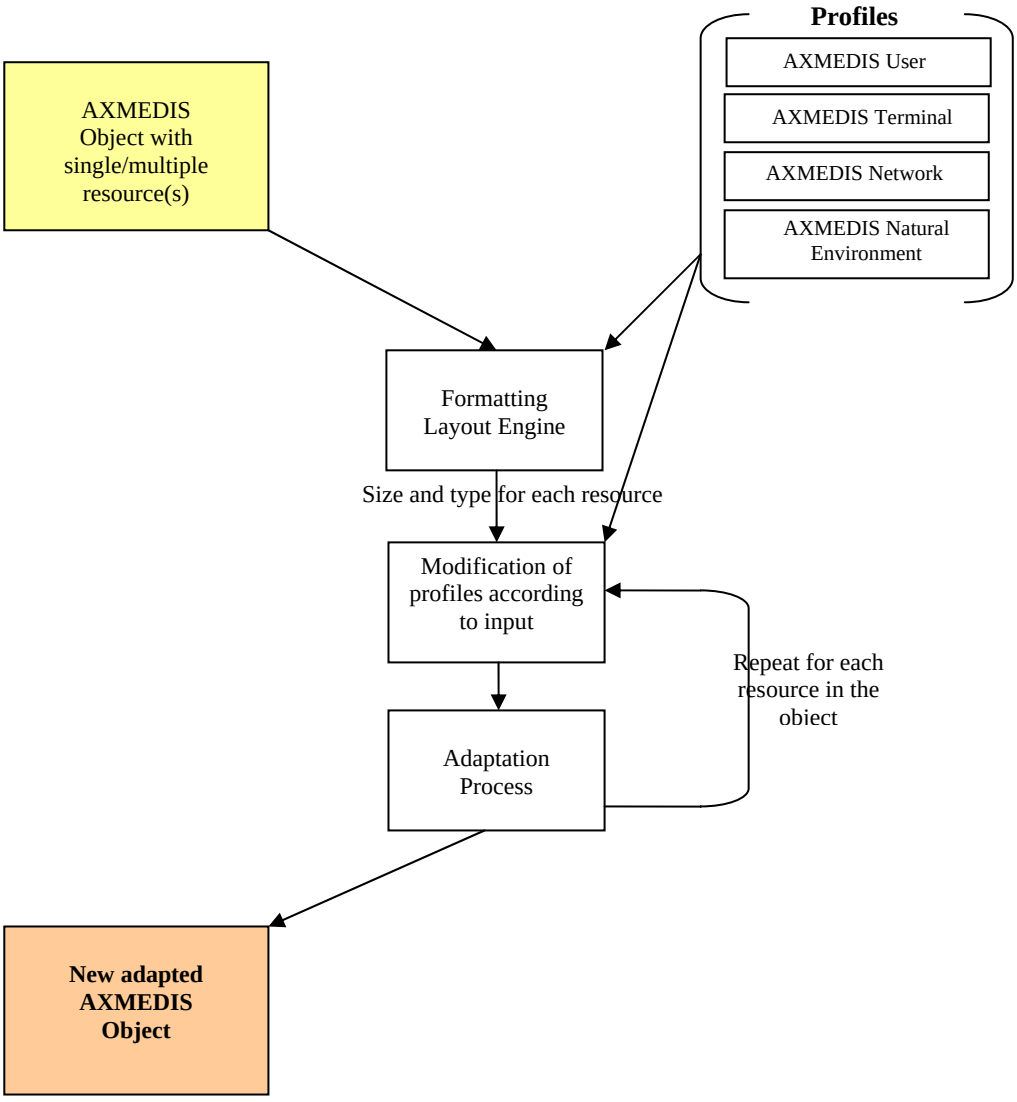

Figure 3.4: AXMEDIS DIA Process flow

JSAxDecisionTakingEngine

JSAxDecisionTakingEngine (underlying C++ class AxDecisionTakingEngine) takes the path of the device profile being used and uses this profile for decision making to arrive at the conclusion which available template profile matches the best.

- If there is an exact match e.g. Samsung i320 is the target device, the profile is loaded by the AxDecisionTakingEngine, if a template exists in the set of available templates, a link to that template is sent to the AXCP script for subsequent adaptation via calls to various AXMEDIS adaptation plugins.
- If an exact match is not available, profile templates by same manufacturer are searched within the available set of templates, and the least capability one is sent to the AXCP.
- If no profile template for device by same manufacturer is available, the lowest capability device profile template from the available template set is given as output, e.g. a device template profile with no color/image capability.

5.3 Ringtone Adaptation (UR)

Ringtone Adaptation refers to the adaptation of ringtones of popular formats to enhance the usability and manage the variable delivery to cater for heterogeneous client devices and user requirements on-demand. It uses external libraries like FFMPEG and TIMIDITY to convert a given ringtone from one format to another and to re-sample it based on the client device.

The following are the main functions described in the Ringtone Adapatation Module

GetInfo: Gets all the information about the input Ringtone like Mimetype, Sampling Rate, Bit Rate, Number of Channels etc

Convert: Converts a ringtone to different formats. (E.g. from Wav to MP3)

Re-sample: Re-samples the input file (i.e. changing frequency, bitrate etc)

Clip: Clips the file to size it for the specified time (e.g. reducing it to a 30-second clip)

During this project period, we have updated all the functions for Ringtone transcoding and made sure that the ffmpeg libraries used were aligned with those deployed by other partners. Since the ffmpeg versions had their basic structures changed for creating the required new versions, re-aligning the libraries with other partners posed a bigger problem than had been anticipated. It involved re-writing of already completed code so as to make it compatible with the corresponding library. Naturally this required a lot of effort to make the functions consistent with the specifications. The other problems encountered were due to the lack of support for Timidity. Since its last version was written in 1997 the support available online currently is negligible and in our opinion, using a library with limited support is not advisable. Accordingly it is planned to exclude the Timidity from our list of supported functions in the next release.

5.4 Audio Adaptation (EPFL)

The audio adaptation functionalities have been integrated with the axeditor as plug-ins through the AXCP interface. The plug-in simply consists of a DLL and an XML file describing the functionalities of the DLL. Both the DLL and the XML description are installed in the plug-in directory of the AXCP compliant tool using the plug-in. The FFMPEG library and the LIBSNDFILE library have been used for the audio transcoding functionalities.

5.4.1 FFmpeg Audio Transcoding

The FFmpeg Audio Transcoding functions are used to convert an audio file into a different format and/or codec. Apart from the bit rate reduction depending on the selected codec, one can further reduce the size of the resulting audio file by changing its sample rate and its number of audio channels. Moreover one can select only a specific portion of the input file to produce the resulting output file by specifying starting and ending points in the input file.

5.4.1.1 Formal description of FFmpeg transcoding algorithm

Description: encode an audio file in another format or another codec and change its sample rate and number of audio channels if needed.

The generic syntax is:

string Trancoding(*AxResource* InputResource, *string* MimeType, *AxResource* OutputResource, *UINT32* OutputSamplingRate, *UINT16*, OuputNumChannels, *UINT16* OutputBitRate, *float* ReadStartingTime, *float* ReadEndingTime, *string* OutputCodec)

Parameter List:

Name: InputResource
Description: the resource to be converted
Parameter Type: *AxResource*
Default Value:
Constraints:
Resource Type: audio
Resource Format: x-mpeg (.mp3), x.aiff (.aif, .aiff), x-wav (.wav), basic (.au, .snd), x-ms-wma (.wma), x-vorbis (.ogg), x-pn-realaudio (.ra, .ram)
Ranges:

Name: MimeType
Description: MimeType for the output resource
Parameter Type: *string*
Default Value:
Constraints:
Resource Type: audio
Resource Format: x-mpeg, x-aiff, x-wav, basic, x-vorbis, x-ac3
Ranges:

Name: OutputResource
Description: Where the output resource will be stored
Parameter Type: *AxResource*
Default Value:
Constraints:
Range:

Name: OutputSamplingRate
Description: The sampling rate of the output resource in Hertz
Parameter Type: *uint32*
Default Value: by default, the sampling rate of the input resource is used
Constraints:
Range:

Name: OutputNumChannels
Description: The number of channels of the audio resource after transcoding
Parameter Type: *uint16*
Default Value: by default, the number of channels of the input resource is used
Constraints:
Range:

Name: OutputBitRate
Description: The bit rate of the audio resource after transcoding in kilo-Bytes (this parameter is used when transcoding towards a compressed audio format such as MP3)
Parameter Type: *uint16*
Default Value: by default, the bit rate is set to 64 kB
Constraints:
Range:

Name: ReadStartingTime
Description: set the beginning of the output resource to *ReadStartingTime* seconds from the beginning of the input resource
Parameter Type: *float*
Default Value: by default, the read starting time is set to 0 seconds which means that the input resource is considered from the beginning
Constraints:
Range:

Name: ReadEndingTime
Description: set the end of the output resource at *ReadEndingTime* seconds from the beginning of the input resource
Parameter Type: *float*
Default Value: by default, the read ending time is set to the end of the input resource
Constraints:
Range:

Name: OutputCodec
Description: set the codec of the output resource; depending on the mime type selected for the output resource, only a certain subset of codec will be supported (the following table shows the possible codecs according to the possible mime types)
Parameter Type: *string*
Default Value: the default codec depend on the mime type selected for the output resource (the following table shows the default codec according to the possible mime types)
Constraints:
Range:

Result: Result
Result Type: string
Result Description: the result of conversion, SUCCESS if ok, ERROR followed by a message in case of error

5.4.1.2 FFmpeg Audio Transcoding Plug-in

The following demonstrates the FFmpeg audio adaptation transcoding function as a plug-in integrated with for the AXMEDIS editor.
The plug-in must be applied on an audio resource of an AXMEDIS object. The adaptation plug-in is called by right-clicking on the interesting resource and selecting the ‘Plugin…’ command:

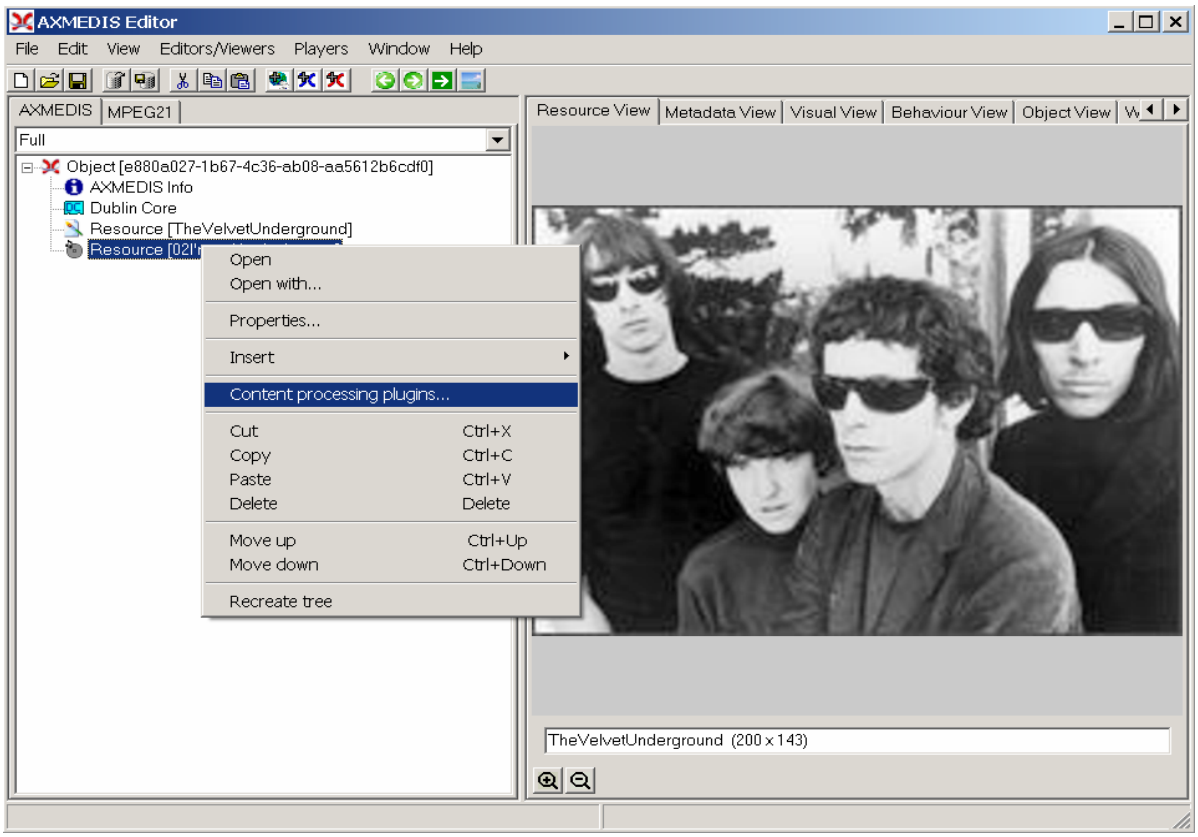

Figure 4.1 : AXMEDIS Transcoding parameters selection screen 1

A window showing the functionalities available for the kind of resource selected appears:

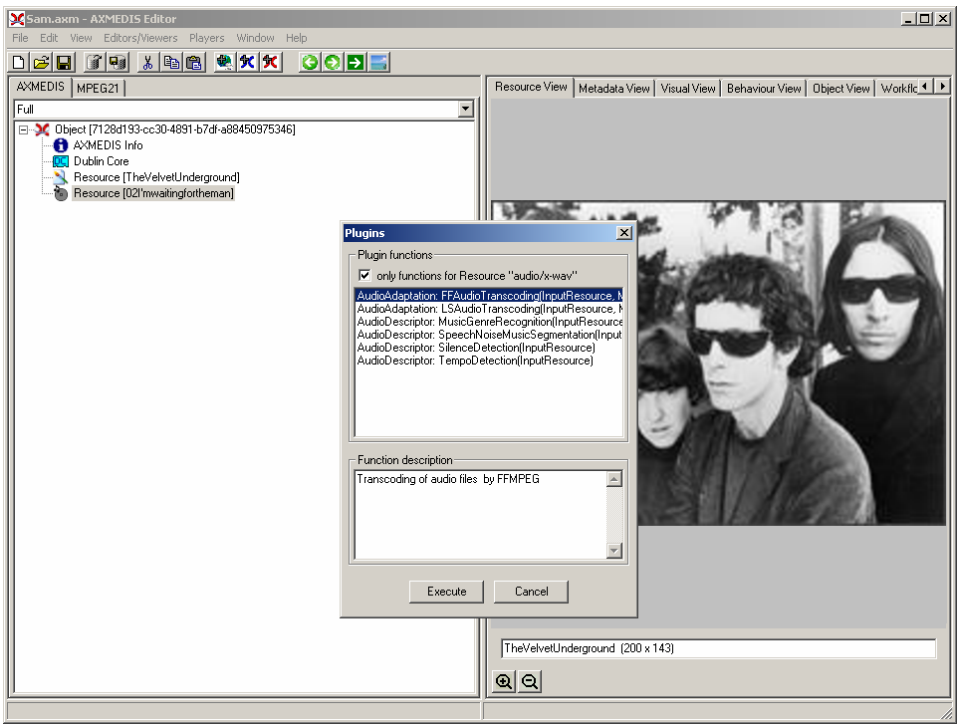

Figure 4.2 : AXMEDIS Transcoding Parameters Selection Screen 2

The first audio adaptation function available is the FFmpeg transcoding function which is selected by clicking on **FFAudioAdaptation: FFAudioTranscoding**. A new window appears showing the interface to the audio transcoding function. In the example of the following figure, the transcoding function is used to create a 10 second snapshot with reduced bit rate of the input audio file:

- Mp3 compression is selected with a bit rate of 64 kB (which corresponds to a low quality)
- Further bit rate reduction is achieved by using a lower sampling rate for the output (22050 Hz) and mixing audio channels into a single mono channel
- Only a portion of 10 seconds of the input resource is selected (starting at time 10 seconds and ending at time 20 seconds)

A snapshot with reduced bit rate is particularly useful to allow a customer to pre-view an item before purchasing the corresponding high quality object.

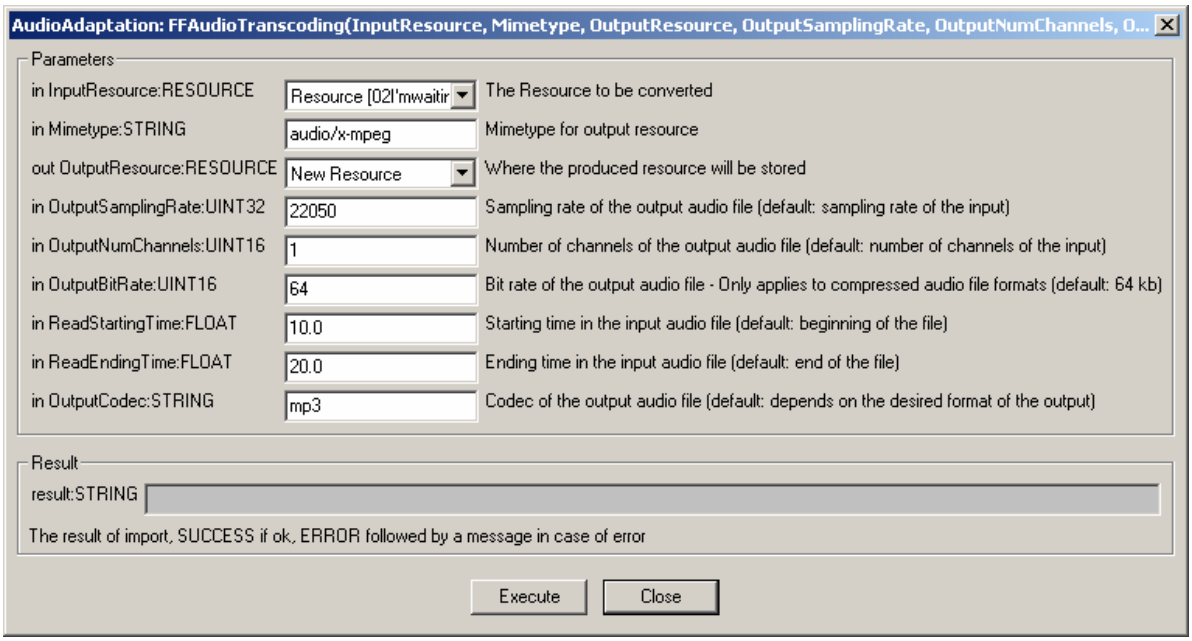

Figure 4.3 : AXMEDIS Audio Adaptation Data Screen 1

5.4.2 Libsndfile

Libsndfile Audio Transcoding

The libsndfile Audio Transcoding function are used to convert an audio file into a different format and/or codec. Apart from the bit rate reduction depending on the selected codec, one can further reduce the size of the resulting audio file by changing its sample rate and its number of audio channels. Moreover one can select only a specific portion of the input file to produce the resulting output file by specifying starting and ending points in the input file.

5.4.2.1 Formal description of libsndfile transcoding algorithm

Description: encode an audio file in another format or another codec and change its sample rate and number of audio channels if needed.

Signature:

string Trancoding(*AxResource* InputResource, *string* MimeType, *AxResource* OutputResource, *float* ReadStartingTime, *float* ReadEndingTime, *string* OutputCodec)

Parameter List:

Name: InputResource

Description: the resource to be converted
Parameter Type: *AxResource*
Default Value:
Constraints:
Resource Type: audio
Resource Format: x-mpeg (.mp3), x.aiff (.aif, .aiff), x-wav (.wav), basic (.au, .snd), x-ms-wma (.wma), x-vorbis (.ogg), x-pn-realaudio (.ra, .ram)
Ranges:

Name: MimeType
Description: MimeType for the output resource
Parameter Type: *string*
Default Value:
Constraints:
Resource Type: audio
Resource Format: x-mpeg, x-aiff, x-wav, basic, x-vorbis, x-ac3
Ranges:

Name: OutputResource
Description: Where the output resource will be stored
Parameter Type: *AxResource*
Default Value:
Constraints:
Range:

Name: ReadStartingTime
Description: set the beginning of the output resource to *ReadStartingTime* seconds from the beginning of the input resource
Parameter Type: *float*
Default Value: by default, the read starting time is set to 0 seconds which means that the input resource is considered from the beginning
Constraints:
Range:

Name: ReadEndingTime
Description: set the end of the output resource at *ReadEndingTime* seconds from the beginning of the input resource
Parameter Type: *float*
Default Value: by default, the read ending time is set to the end of the input resource
Constraints:
Range:

Name: OutputCodec
Description: set the codec of the output resource; depending on the mime type selected for the output resource, only a certain subset of codec will be supported (the following table shows the possible codecs according to the possible mime types)
Parameter Type: *string*
Default Value: the default codec depend on the mime type selected for the output resource (the following table shows the default codec according to the possible mime types)
Constraints:
Range:

Result: Result
Result Type: string

Result Description: the result of conversion, SUCCESS if ok, ERROR followed by a message in case of error

Libsndfile supported types and codecs:

For a list of codecs and formats supported by the Libsndfile library, please refer to section 34.5.

Mime Type accepted :

```
audio/x-wav
audio/x-basic
audio/x-paris
audio/x-svx
audio/x-nist
audio/x-voc
audio/x-ircam
audio/x-w64
audio/x-sd2
audio/x-flac
application/x-pcm
application/x-pagerecall
```

Here's an example on how to use the libsndfile audio adaptation transcoding function as a plug-in with for the AXMEDIS editor.

5.4.2.2 Libsndfile AudioTranscoding Plug-in

The following demonstrates the libsndfile audio adaptation transcoding function as a plug-in integrated with for the AXMEDIS editor.
The plug-in must be applied on an audio resource of an AXMEDIS object. The adaptation plug-in is called by right-clicking on the interesting resource and selecting the 'Plugin…' command:

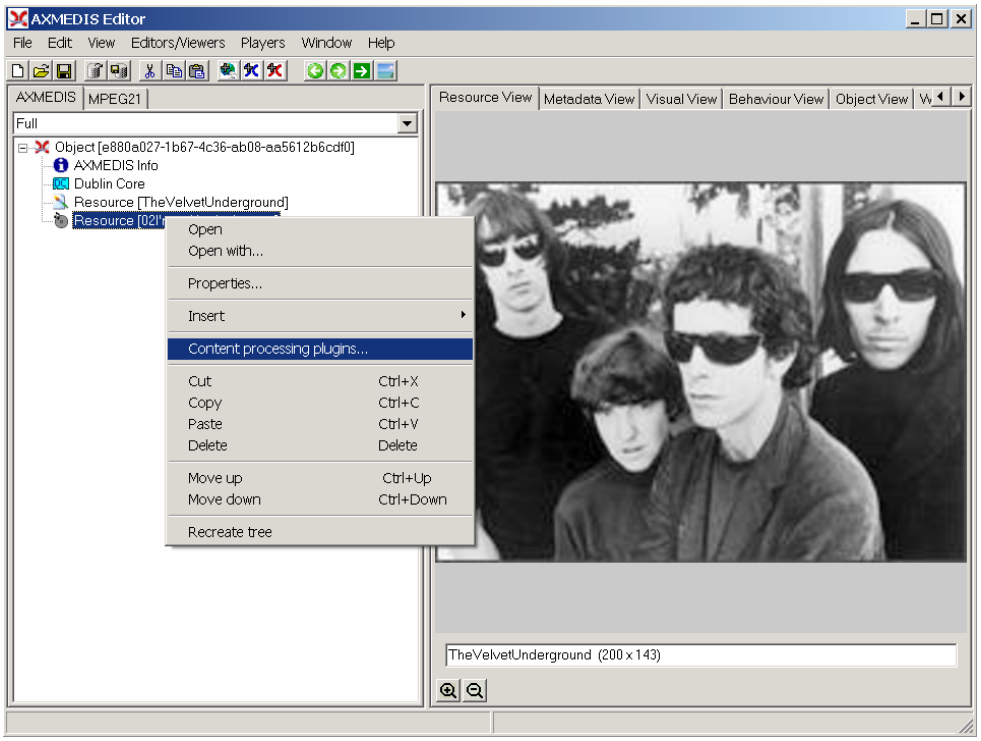

Figure 4.4 :: AXMEDIS Transcoding Parameters Selection Screen 3

A window showing the functionalities available for the kind of resource selected appears:

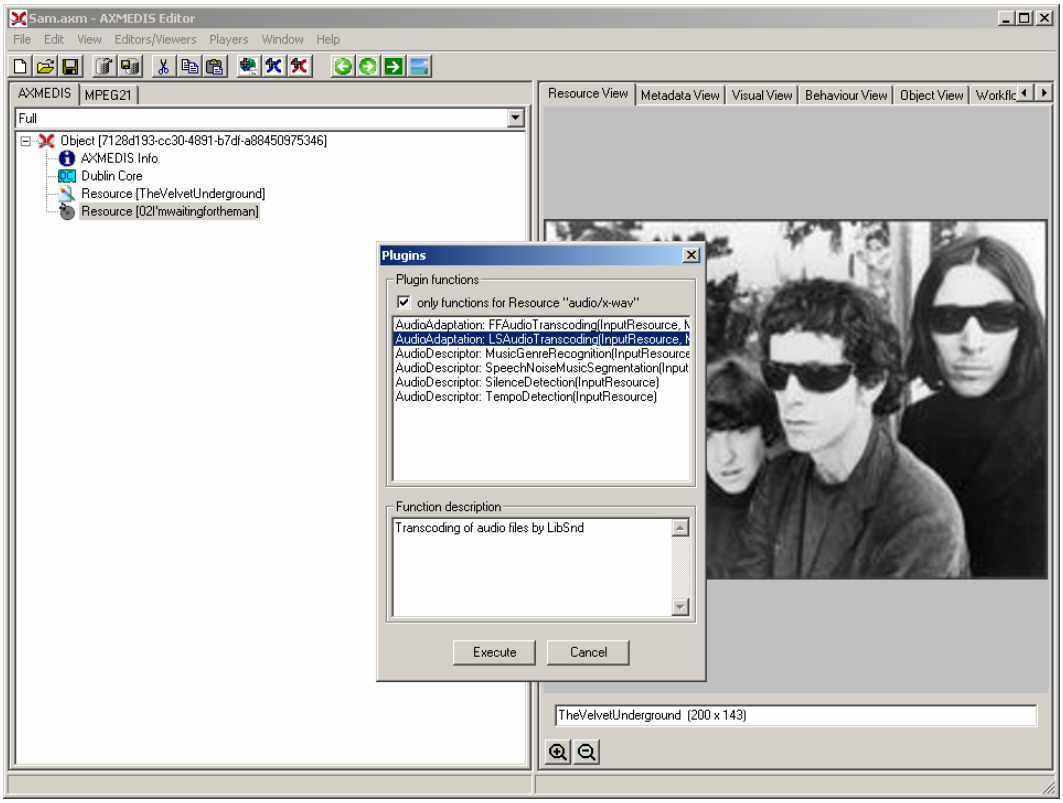

Figure 4.5 : AXMEDIS Transcoding Parameters Selection Screen 4

The first audio adaptation function available is the libsndfile transcoding function which is selected by clicking on **LSAudioAdaptation: LSAudioTranscoding**. A new window appears showing the interface to the audio transcoding function. In the example of the following figure, the transcoding function is used to create a 10 second snapshot with reduced bit rate of the input audio file:

- AIFF format
- Only a portion of 8 seconds of the input resource is selected (just the beginning of the sound track)

Such a snapshot could be useful for small audio sampling.

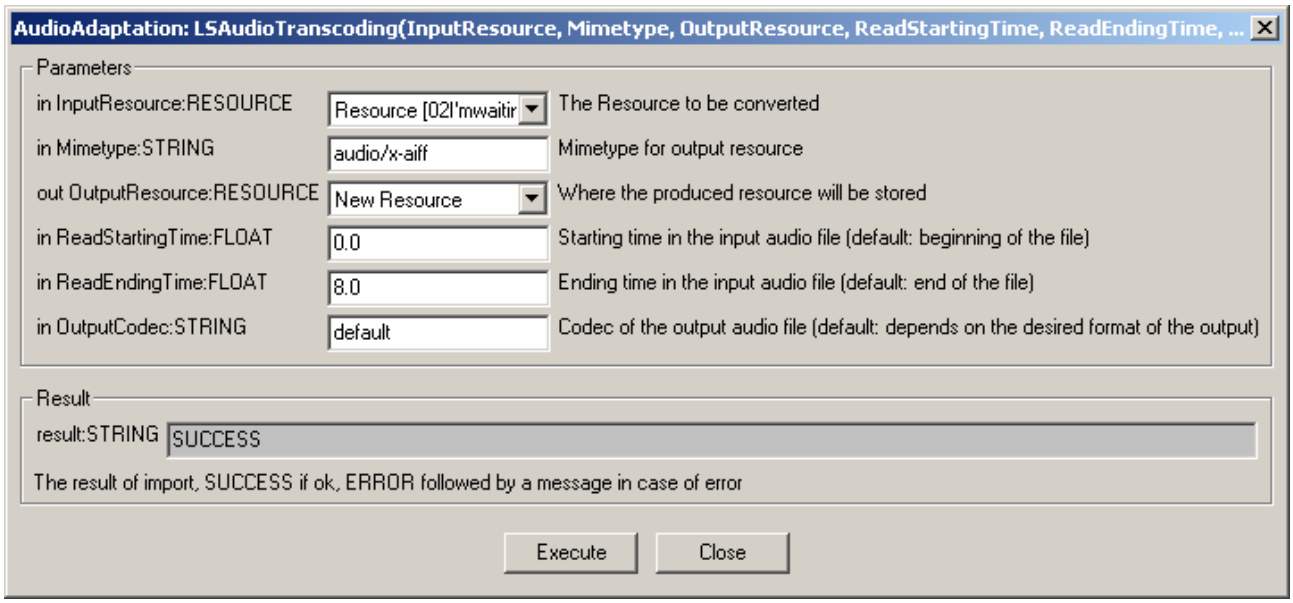

Figure 4.6 :AXMEDIS Audio Adaptation Data Screen 2

Major Features accessible:

- Audio transcoding with FFMEPG library
- Audio transcoding with Libsndfile library

5.5 Image Adaptation (DSI)

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.

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

Major Features accessible:
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.
AVI	R	Microsoft Audio/Visual Interleaved	
AVS [44]	RW	AVS X image	
BMP[45]	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	Requires Ghostscript to read.

Tag	Mode	Description	Notes
		PostScript	
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	

Tag	Mode	Description	Notes
		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).

Tag	Mode	Description	Notes
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	Use `-size` and `-depth` to specify

Tag	Mode	Description	Notes
		blue samples	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

Tag	Mode	Description	Notes
			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	

Table 1: Axmedis Supported Adaptation Formats

5.6 Document Adaptation (DIPITA)

The following conversion libraries are the ones that will be used for text document conversion, as they obtain (according to the literature) the best performances compared to the other similar ones:

1. DOCFRAC (http://docfrac.sourceforge.net/), that allows conversions from RTF to HTML, from RTF to TXT, from HTML to RTF, from HTML to TXT, from TXT to RTF and from TXT to HTML.
The exploitation of the DOCFRAC features library will be particularly useful for converting active web pages, converting many documents at a time and converting output from Microsoft's Internet Explorer RTF control to HTML.
The supported platforms are Windows, Linux (command line) and programming kit (ActiveX and DLL). DocFrac is free. It is released under the LGPL.

2. GNU Ghostscript (http://www.cs.wisc.edu/~ghost/)
Ghostscript is the name of a set of software that contains an interpreter for the PostScript (TM) language and the Adobe Portable Document Format, and a set of C procedures (the Ghostscript library) that implement the graphics and filtering (data compression / decompression / conversion) capabilities that appear as primitive operations in the PostScript language and in PDF. Versions entitled "GNU Ghostscript" are distributed with the GNU General Public License.

3. XPDF (http://www.foolabs.com/xpdf/)
Xpdf is an open source viewer for Portable Document Format (PDF) files. The Xpdf project also includes a PDF text extractor, PDF-to-PostScript converter, and various other utilities.
Xpdf runs under the X Window System on UNIX, VMS, and OS/2. The non-X components (pdftops, pdftotext, etc.) also run on Win32 systems and should run on pretty much any system with a decent C++ compiler.
Xpdf is designed to be small and efficient. It can use Type 1, TrueType, or standard X fonts.
Xpdf is licensed under the GNU General Public License (GPL), version 2.

4. HTMLDOC (http://www.easysw.com/htmldoc/)
HTMLDOC converts HTML source files into indexed HTML, PostScript, or Portable Document Format (PDF) files that can be viewed online or printed.
The program is free software and is distributed under GPL.

5. PDF2HTML (http://pdftohtml.sourceforge.net/)
pdftohtml is a utility which converts PDF files into HTML and XML formats. It's based on the xpdf 2.02 by Derek Noonburg.
The program is free software and is distributed under GPL.

6. a2ps (http://gnuwin32.sourceforge.net/packages/a2ps.htm)
GNU a2ps is an Any to PostScript filter. Of course it processes plain text files, but also pretty prints quite a few popular languages. Its slogan is precisely `` Do The Right Thing '', which means that though it is highly configurable, everything was made so that a novice user can do complicated PostScript manipulations. For instance, it has the ability to delegate the processing of some files to other filters (such as groff, texi2dvi, dvips, gzip etc.), what allows a uniform treatment (n-up, page selection, duplex etc.) of heterogeneous files. It supports a wide number of encodings, and a very good handling of Latin 2-6 should be noted, thanks to Ogonkify (by Juliusz Chroboczek). Needed fonts are automatically downloaded. The interface is internationalized, the output is customizable and there are as many options as users had wishes (table of content, headings, virtual page layout etc. etc.).
The program is free software and is distributed under GPL.
To test the first results of the adaptation tool, it has been created a test corpus for content adaptation. It contains several documents, each one in different textual format, and precisely:

a) Digital music report 2005 (a state of the art on digital music; formats: pdf, txt)
b) European constitution (the EU constitution in Italian, formats: pdf, rtf, txt)
c) Hibernate documentation (documentation on an object/relational mapping tool for Java environments; formats: pdf, html, txt)
d) Html specification (specification from the w3c on html formatting; formats: pdf, ps, html, txt).

5.7 Video Adaptation (FHGIGD)

The work performed focused on the (integration of) transcoding algorithms. To show AXMEDIS' capabilities in terms of content adaptation as well as extensibility the FFMPEG library has been selected for the implementation/integration of the firstly available video adaptation functionality.
The via command line version of FFMPEG available video options are:

-b bitrate	set video bitrate (in kbit/s)
-r rate	set frame rate (Hz value, fraction or abbreviation)
-s size	set frame size (WxH or abbreviation)
-aspect aspect	set aspect ratio (4:3, 16:9 or 1.3333, 1.7777)
-croptop size	set top crop band size (in pixels)
-cropbottom size	set bottom crop band size (in pixels)
-cropleft size	set left crop band size (in pixels)
-cropright size	set right crop band size (in pixels)
-vn	disable video
-bt tolerance	set video bitrate tolerance (in kbit/s)
-maxrate bitrate	set max video bitrate tolerance (in kbit/s)
-minrate bitrate	set min video bitrate tolerance (in kbit/s)
-bufsize size	set ratecontrol buffere size (in kByte)
-vcodec codec	force video codec ('copy' to copy stream)
-sameq	use same video quality as source (implies VBR)
-pass n	select the pass number (1 or 2)
-passlogfile file	select two pass log file name

Besides the basic functionality so called “advanced options” are available:

-pix_fmt format	set pixel format
-g gop_size	set the group of picture size
-intra	use only intra frames
-qscale q	use fixed video quantiser scale (VBR)
-qmin q	min video quantiser scale (VBR)
-qmax q	max video quantiser scale (VBR)
-mbqmin q	min macroblock quantiser scale (VBR)
-mbqmax q	max macroblock quantiser scale (VBR)
-qdiff q	max difference between the quantiser scale (VBR)
-qblur blur	video quantiser scale blur (VBR)
-qcomp compression	video quantiser scale compression (VBR)
-rc_init_cplx	complexity initial complexity for 1-pass encoding
-b_qfactor factor	qp factor between p and b frames
-i_qfactor factor	qp factor between p and i frames
-b_qoffset offset	qp offset between p and b frames
-i_qoffset offset	qp offset between p and i frames
-rc_eq equation	set rate control equation
-rc_override	override rate control override for specific intervals
-me method	set motion estimation method
-dct_algo algo	set dct algo
-idct_algo algo	set idct algo
-er n	set error resilience
-ec bit_mask	set error concealment
-bf frames	use 'frames' B frames
-mbd mode	macroblock decision
-mbcmp	cmp function macroblock compare function
-ildctcmp	cmp function ildct compare function
-subcmp	cmp function subpel compare function
-cmp	cmp function fullpel compare function
-4mv	use four motion vector by macroblock (MPEG4)
-obmc	use overlapped block motion compensation (h263+)
-part	use data partitioning (MPEG4)
-bug param	workaround not auto detected encoder bugs
-strict strictness	how strictly to follow the standarts
-deinterlace	deinterlace pictures
-ildct	force interlaced dct support in encoder (MPEG2/MPEG4)
-ilme	force interlacied me support in encoder MPEG2
-psnr	calculate PSNR of compressed frames
-vstats	dump video coding statistics to file
-vhook module	insert video processing module
-aic	enable Advanced intra coding (h263+)
-aiv	enable Alternative inter vlc (h263+)
-umv	enable Unlimited Motion Vector (h263+)
-alt	enable alternate scantable (mpeg2)
-trell	enable trellis quantization
-scan_offset	enable SVCD Scan Offset placeholder
-intra_matrix	matrix specify intra matrix coeffs
-inter_matrix	matrix specify inter matrix coeffs
-top	top=1/bottom=0/auto=-1 field first
-nr	noise reduction

The functionality available within FFMPEG addresses two categories of video adaptation:

- Considering the (fixed) needs of the content creator or service providers: The content can be adapted according to the requirements of the content creator or service providers. The simplest case is the adaptation of the spatial resolution as supported by FFMPEG to a fixed resolution without changing the aspect ratio.
- Dynamic content creation: FFMPEG also allows changing the aspect ratio of the video content. Thus, a basic functionality required to dynamically adapt content according to the receiving device is available.

5.7.1 Video-Adaptation plug-in

Current version of the video adaptation plug-in implements a sub-set of the potential parameters as shown in the next diagram respectively as described below:

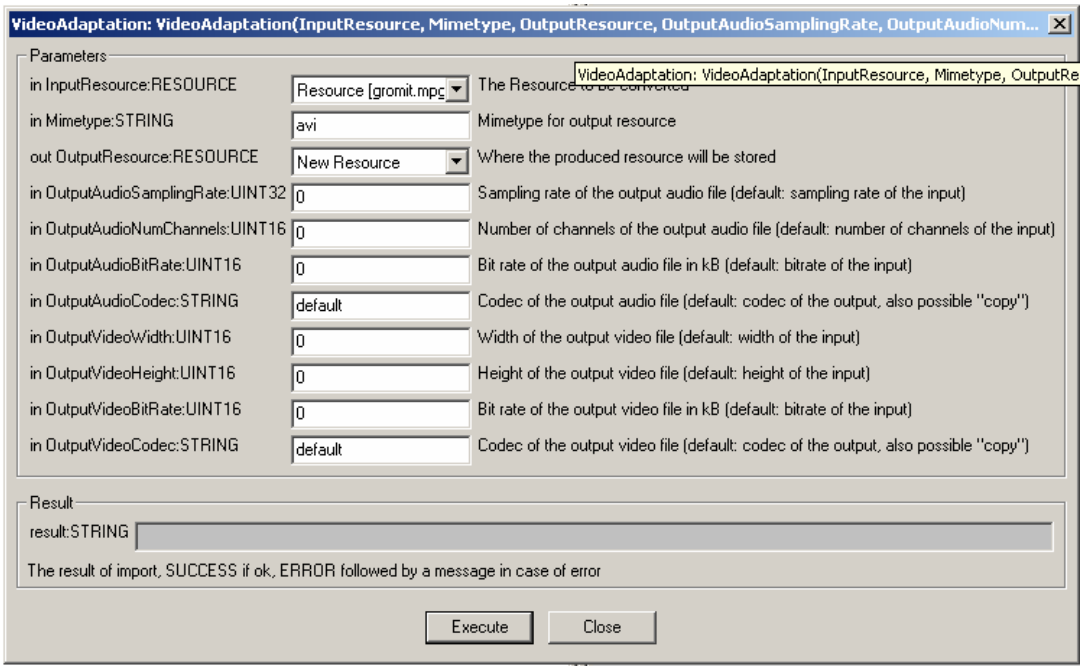

Figure 4.7 : : AXMEDIS Audio Adaptation Data Screen 3

- AudioSamplingRate: Sampling rate of the output audio.
- AudioNumChannels: Number of Channels of the output audio file.
- AudioBitRate: Bit rate of the output audio file in kB.
- AudioCodec: Codec of the output audio file.
- VideoWidth; Width of the output video file.
- VideoHeight: Height of the output video file.
- VideoBitRate: Bit rate of the output video file in kB.
- VideoCodec: Codec of the output video file.

5.8 DIA Adaptation Decision Engine

Input:

1. Resource
2. Resource description
3. Constraints (profiles)

Output:

1. Adapted Resource
2. Adapted Resource description

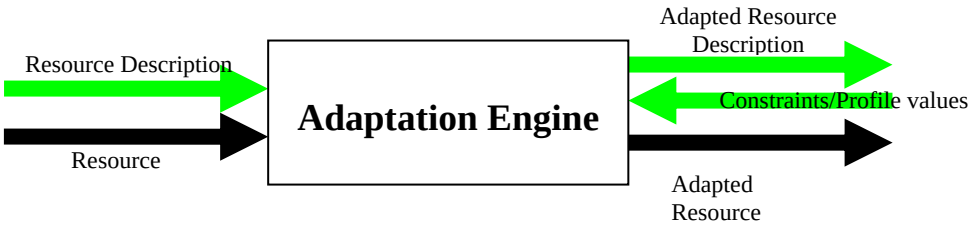

Figure 4.8 : Adaptation Engine (reproduced from [46])

Adaptation Engine Constituents:
The widely acknowledged approach (e.g. as advocated in [46] is for the DIA Adaptation Engine to be conceptually divided into two parts:

1. ADTE: Adaptation Decision Taking Engine
2. BDAE: Bit-stream and Description Adaptation Engine

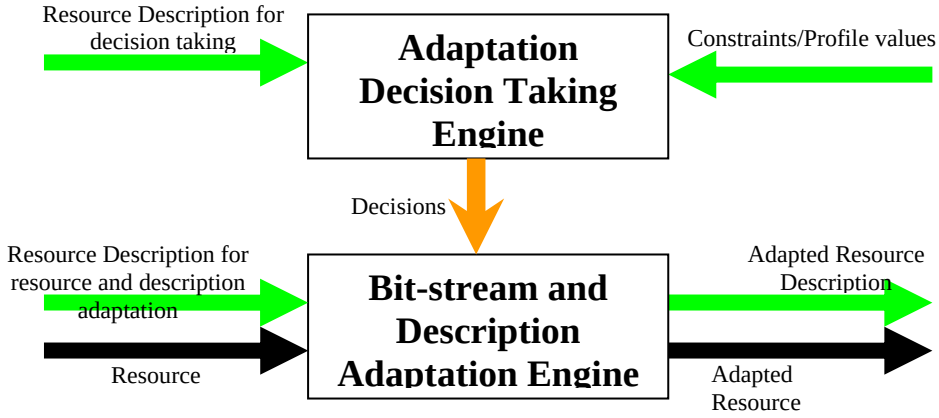

Figure 4.9: Adaptation Engine Components (reproduced from [46])

In AXMEDIS, the ADTE is implemented as a C++ layer with a specified set of profile templates ranging from very low capability to very high capability. Once the decisions have been taken and a template profile has been identified to be used for adaptation of AXOBJECT, this is passed on to the BDAE which is essentially AXCP scripts that use various AXMEDIS plugins and JS functionalities to convert the resources of the AXOBJECT and its metadata.

6 Work Done in DRM Support for Mobile and Interoperability (UPC)

Different systems define different rights expression languages. If it is desired to make heterogeneous systems interoperable we may have to adapt the rights expressions.
Moreover, changing versions of the specifications could make it difficult to implement any kind of software to resolve the interoperability challenges.
Not only different systems and versions are problematic re DRM interoperability, but also, the devices present in the mobile environment involve different ways of programming. For instance, some PDAs

supporting mobile communication use Windows based systems, while other kind of mobile phones, use systems based on Symbian or other systems. The implementation of a software system as a generic solution to support rights expression languages for such a heterogeneous environments is thus a challenging task.

6.1 Rights Transcoding/Adaptation Implementation and Results

Rights transcoding/adaptation is explained in detail in section 14 of "DE4.3.1.2 Content Composition and Formatting, 1st update". It provides a thorough explanation on how licenses can be translated from MPEG-21 REL to OMA DRM REL and viceversa, following different approaches, such as translations based on UML models or translations based on relational models.

The Rights Expression Translator is a module to input license in xml and convert to another. The mapping is done getting the original XML license in its model format. Depending on the destination language and origin language, the algorithm calls the appropriate functions to transform each object of the original license model into the new one. After that, the system calls a method that generates the new XML license from the one in the license model.
Rights Expression Translator library, which involves the translation of rights expressions from MPEG-21 REL to OMA DRM REL and viceversa, has been already developed. Its specifications can be found in section 16 of "DE3.1.2.3.14 Specification of AXMEDIS Protection Support – Update".

The following features regarding transcoding of PAR and Licenses are currently available:

- License and PAR Database for MPEG 21 REL
- Libraries to support the creation and management of license and PAR in MPEG 21 REL format
- License Database for OMA DRM REL
- Libraries to support the creation and management
- Implementation of the tool for transcoding PAR and Licenses, the Rights Expression Translator

6.2 Protection tools for supporting mobile distribution channel

To be able to connect to PMS Server, a specific PMS Client has to be developed in J2ME. The PMS Client has to be secure (use SSL) to connect to PMS Server. Since there are no libraries that implement SSL client authentication in Mobile environment, we have to solve this issue creating a specific architecture to access the PMS Server.

The architecture to connect to PMS Server from a Mobile is the following:

PMS Client (J2ME Midlet Library) → PMS Servlet Gateway → PMS Server

PMS Client for mobile is a Java class that implements a Servlet Client. It contains the necessary operations to invoke the PMS Server. The servlet client can connect to the PMS Servlet Gateway using SSL, with server authentication, according to the literature (see http://developers.sun.com/mobility/midp/articles/https/). The Servlet (programmed in J2SE) implements the complete WS secure call to the PMS Server with both SSL client and server authentication.

Although it is possible, according to the literature, to implement SSL server authentication in a servlet client, it has not been possible up to the moment to make it operative. Thus, the communication between the client and servlet is currently not secure, being secured only the connection between the servlet and PMS Server.

The functions implemented in PMS Client are the following:

getLicense(String licenseId)

certifyForMobile(String axid, String axrtid, String AXDOM, String toolFingerprint, String regDeadline)

authorise(String logID, String AXOID, String objectVersion, String protectionStamp, String AXWID, String AXDOM, String AXUID, String AXDID, String AXCID, String ownerName, String AXTID, String AXLID, String AXCSID, String location, String operationDetails, String operation, String registrationTimestamp, String executionTimestamp, String estimatedHwFingerprint, int timesUsed, String territoryOfEmission, boolean protectionInfoRequired)

Ping(int x)

reverifyForMobile(String axid, String AXTID, String AXDOM, String toolFingerprint, String lastFPPA)

verifyUser(String axid, String AXDOM)

verifyForMobile(String axid, String AXTID, String AXDOM, String toolFingerprintDigest, String lastFPPA)

The functions which named with the word "Mobile" (e.g. verifyForMobile, certifyForMobile, …) are slightly different from the original ones developed for the PC devices. Whereas original functions have some parameters of type byte array (byte[]), for servlets this types are not permited, so they have been transformed into String, where each String holds the base64 encoding of the byte array.

Due to the restrictions imposed by mobile platforms, the Secure Cache module has not been implemented for the mobile platform. Thus, is is not possible for a user to consume content in an unconnected scenario, as there is no possibility for tracking offline operations or storing protection information on mobile devices for a future usage. Regarding AXMEDIS Action Logs (i.e. the AXMEDIS version of Event Reporting), all the necessary information involving the content usage is sent to the PMS Server when requesting the authorisation. PMS Server will generate and send the corresponding Action Log to AXCS when needed.

6.3 Support for MPEG-21 DIA conditions in the authorisation process

When a resource has to be used in a mobile device, it usually needs some kind of adaptation in order to fit the limited capabilities of the device. In these cases, the adaptation conditions, if any, have to be represented using a standard notation that allows the system to recognise them as new constraints. For that reason, the language used to represent the adaptation has to be compatible with the one used in current licenses, the MPEG-21 REL, in order to be able to express both things together and make the authorisation process easier.
MPEG-21 DIA defines a set of rules that can be used to represent the adaptation possibilities in a standard way, being natively compatible with the MPEG-21 REL licenses, as DIA and REL are parts of the same standard.
Although DIA enables a huge number of rules, in a first step, we have added three different kinds of MPEG-21 DIA adaptation conditions in the licenses, which enable controlling the format, the bitrate or even the resolution changes done over a digital content.
To enable the adaptation control using current license model, it has been necessary to change a few things on it. One example of these changes is in the MPEG-21 license representation, where MPEG-21 DIA information has been added. The MPEG-21 standard defines two new parts in the licenses. The first one indicates the number and type of adaptations that the user will be granted to do, whereas the second one defines the limits of each possible adaptation.
Another change that has to be done to current license model involves the authorisation module, which has to be modified in order to recognise and be able to apply these new kinds of conditions. Two new entry parameters have been added to the authorisation module, both represented using the MPEG-7 XML standard for describing the digital content attributes. The first one contains the information about the original content codification, and the other one expresses the changes that the user wants to apply to it.
With this information and the rights expressed in the license, the authoriser is able to give a result indicating if the adaptation of the content is possible according to the provided attributes.

7 Work Done on Supporting Interoperability between MPEG-21 and OMA (DSI)

Details regarding this part are included into Annex 1 to this document. The Annex I is a document in Italian language that presents an experiment win which the AXMEDIS Content processing platform has been used to proof the interoperability and multichannel management among AXMEDIS based channel (MPEG-21) and OMA channel. For this purpose the AXMEDIS java script of AXCP has been used as a back end for:

- Licence transformation (similarly to what has been described by FUPF in the previous chapter)
- Adaptation of content
- Access to AXMEDIS content and resources in general
- Production of OMA packages by means of the SonyEricsson OMA packager
- Production of ODRL OMA licenses
- Posting of OMA packages to an OMA like distribution server
- Posting of OMA ODRL license to an OMA like distribution server

A front end OMA like distribution server has been realised providing:

- The OMA protocol for communicating with clients

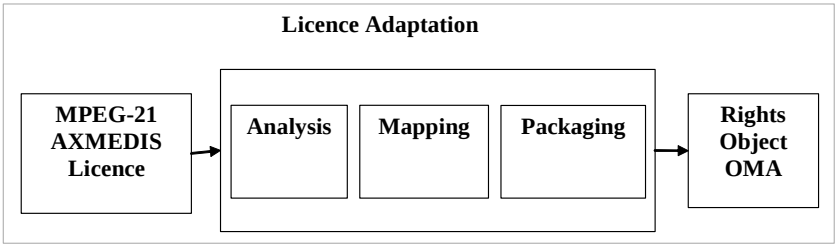

Figure 6.1: Licence Adaptation

7.1 Protocol description

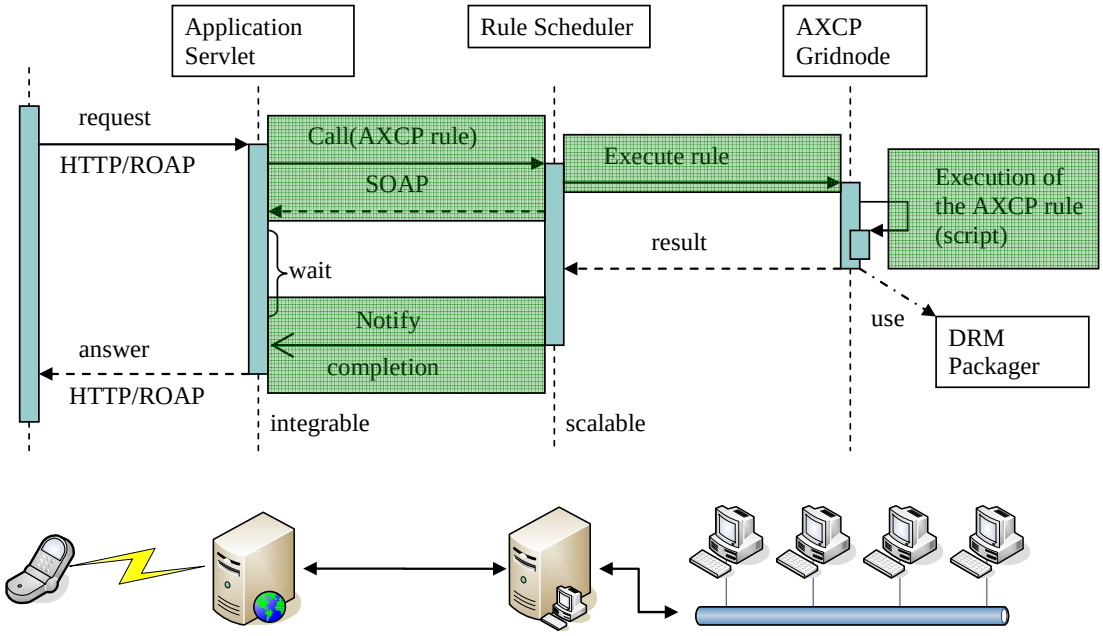

Figure 6.2: Protocol Description

7.2 Script: ISSUE.XML RULE

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
- <Rule xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="Rule_Axmedis.xsd">
- <Header>
  <Rule_Name>issue</Rule_Name>
  <AXRID>axcprule:5e0215fc-501a-422f-ace3-959326da62b7</AXRID>
  <Rule_Version />
  <Rule_Type>AXCP</Rule_Type>
  <Software_Name />
  <Version_of_software />
  <Date_of_production>2006-08-17</Date_of_production>
  <Author />
  <Affiliation />
  <URL />
  <Comment />
  <Last_Modifications>2006-08-18</Last_Modifications>
  <Terminal_ID />
  <Cost />
  <Work_Item_ID />
    </Header>
- <Schedule>
- <Run>
  <Date>2006-08-17</Date>
  <Time>01:02:38</Time>
  <Periodicity Unit="Day">0</Periodicity>
  <Expiration_Date>2006-08-17</Expiration_Date>
  <Expiration_Time>01:02:38</Expiration_Time>
    </Run>
  <Status>Inactive</Status>
    </Schedule>
- <Definition>
- <AXCP_Rule>
- <Arguments>
  <Parameter Name="axoid" Type="String" />
  <Parameter Name="user" Type="String" />
  <Parameter Name="uriRights" Type="String" />
  <Parameter Name="uriContents" Type="String" />
  <Parameter Name="uriResources" Type="String" />
  <Parameter Name="uriLicenses" Type="String" />
  <Parameter Name="uriPackager" Type="String" />
    </Arguments>
- <Rule_Body>
  - <JS_Script name="main">
  - <![CDATA[
  function sample(){
    if (authenticate()){
      var axobject = getObject();
      if(axobject != null){
        var axresource = getResource(axobj);
        if(axresource != null){
          var resourceURI = uriResources + axresource.contentID;
          axresource.save(resourceURI);

          var licenseURI = getLicenseURI();
          if(licenseURI != null){
```

```
                var omaobject = parseLicense(licenseURI);
                var i = 0;
                for(i in omaobject.resources){
                   var resource = omaobject.resources[i];
                   if(resource.ID == axresource.contentID){
                      packageResource(resource, axresource.mimeType);
                   }
                }
             }
          }
       }
    }
}
   ]]>
   </JS_Script>
- <JS_Script name="authentication">
- <![CDATA[
// override this for validating users.
// function reads user ID from global parameter 'user'.
function authenticate(){
    if(user != ""){
       return true;
    }else{
       return false;
    }
}
   ]]>
   </JS_Script>
- <JS_Script name="axmedis">
- <![CDATA[
// override this to search and return an AxObject by AXOID, null otherwise.
// function reads AXOID from global parameter 'axoid'.
function getObject(){
    return null;
}
// extract the resource in an AxObject.
// assume that there is just one resource in each object.
function getResource(axobj){
    var contents = obj.getContent();

    if(contents != null && contents.length > 0){
       var content = contents[0];
       if(content instanceof AxResource){
          return content;
       }
    }

    return null;
}
   ]]>
   </JS_Script>
```

```
- <JS_Script name="oma">
- <![CDATA[
function omaObject(){
    this.issuer = null;
    this.resources = new Array();
    this.print = omaprint;
}
function omaResource(){
    this.ID = null;
    this.right = null;
    this.count = null;
    this.from = null;
    this.to = null;
    this.principal = null;//redundant
    this.print = resourceprint;
}
function omaprint(){
    print("issuer: " + this.issuer);

    var i = 0;
    for(i in this.resources){
        this.resources[i].print();
    }
}
function resourceprint(){
    print("ID: " + this.ID);
    print("    right: " + this.right);
    print("    count: " + this.count);
    print("    from: " + this.from);
    print("    to: " + this.to);
    print("    principal: " + this.principal);
}
  ]]>
  </JS_Script>
- <JS_Script name="license">
- <![CDATA[
var omaobj;
// override this to return the user's license file path.
function getLicenseURI(){
    var licenseURI = uriLicenses + user + ".xml";
    if(existFile(licenseURI)){
        return licenseURI;
    }else{
        return  null;
    }
}
// parse an AxLicense stored in an XML file.
// return an 'oma' object filled with the information retrieved form license.
function parseLicense(licenseURI){
    var xml = readFromFile(licenseURI);
```

```
    var xmldoc = new REXML(xml);

    omaobj = new omaObject();
    visit(xmldoc.rootElement);

    return omaobj;
}
function visit(element){
    switch(element.name){
        case "r:issuer":
            omaobj.issuer = resolveIssuer(element);
            break;
        case "mx:diReference":
            omaobj.resources[omaobj.resources.length] = resolveResource(element);
            break;
    }

    var i = 0;
    for(i in element.childElements){
        var current = element.childElements[i];
        if(current.type == "element")
            visit(current);
    }
}
function resolveResource(element){
    var res = new omaResource();

    // ID
    res.ID = resolveID(element);

    // properties
    i = 0;
    var parent = element.parentElement;
    for(i in parent.childElements){
        current = parent.childElements[i];
        switch(current.name){
            case "r:keyHolder":
                res.principal = resolvePrincipal(current);
                break;
            case "r:allConditions":
                var conditions = resolveConditions(current);
                res.count = conditions.count;
                res.from = conditions.from;
                res.to = conditions.to;
                break;
            case "mx:play":
            case "mx:print":
            case "mx:execute":
                res.right = current.name;
```

```
                break;
            }
        }

        return res;
    }
    function resolveIssuer(element){
        var keyholder = element.childElement("r:keyHolder");
        var info = keyholder.childElement("r:info");
        var keyname = info.childElement("dsig:KeyName");
        return keyname.text;
    }
    function resolveID(element){
        var identifier = element.childElement("mx:identifier");
        return identifier.text;
    }
    function resolvePrincipal(element){
        var info = element.childElement("r:info");
        var keyname = info.childElement("dsig:keyName");
        return keyname.text;
    }
    function resolveConditions(element){
        var conditions = new Object();

        conditions.count = null;
        conditions.from = null;
        conditions.to = null;

        var i = 0;
        for(i in element.childElements){
            var current = element.childElements[i];
            switch(current.name){
                case "sx:exerciseLimit":
                    var count = current.childElement("sx:count");
                    conditions.count = count.text;
                    break;
                case "r:validityInterval":
                    var from = current.childElement("r:notBefore");
                    var to = current.childElement("r:notAfter");
                    conditions.from = from.text;
                    conditions.to = to.text;
                    break;
            }
        }

        return conditions;
    }
    ]]>
    </JS_Script>
```

```
- <JS_Script name="jsxml">
- <![CDATA[
////////////////////////////////////////////////////////////
////////// JSXML XML Tools              ////////////////////
////////// Ver 1.2 Jun 18 2001          ////////////////////
////////// Copyright 2000 Peter Tracey  ////////////////////
////////// http://jsxml.homestead.com/  ////////////////////
////
//// Objects:
////
//// REXML
////   Regular Expression-based XML parser
////
//// JSXMLIterator
////   Iterates through the tree structure without recursion
////
//// JSXMLBuilder
////   Loads xml into a linear structure and provides
//// interface for adding and removing elements
//// and setting attributes, generates XML
////
//// Utility functions:
////
//// ParseAttribute
////   Takes string of attibutes and attribute name
////   Returns attribute value
////
//// Array_Remove
////   Removes element in array
////
//// Array_Add
////   Adds element to array
////
//// RepeatChar
////   Repeats string specified number of times
////
////////////////////////////////////////////////////////////
function REXML(XML) {
 this.XML = XML;
 this.rootElement = null;
 this.parse = REXML_parse;
 if (this.XML && this.XML != "") this.parse();
}
 function REXML_parse() {
    var reTag = new RegExp("<([^>/ ]*)([^>]*)>","g"); // matches that tag name $1 and attribute string $2
    var reTagText = new RegExp("<([^>/ ]*)([^>]*)>([^<]*)","g"); // matches tag name $1, attribute string $2, and text $3
    var strType = "";
    var strTag = "";
    var strText = "";
```

```
var strAttributes = "";
var strOpen = "";
var strClose = "";
var iElements = 0;
var xmleLastElement = null;
if (this.XML.length == 0) return;
var arrElementsUnparsed = this.XML.match(reTag);
var arrElementsUnparsedText = this.XML.match(reTagText);
var i=0;
if (arrElementsUnparsed[0].replace(reTag, "$1") == "?xml") i++;
for (; i<arrElementsUnparsed.length; i++) {
 strTag = arrElementsUnparsed[i].replace(reTag,"$1");
 strAttributes = arrElementsUnparsed[i].replace(reTag,"$2");
 strText = arrElementsUnparsedText[i].replace(reTagText,"$3").replace(/[\r\n\t ]+/g, " "); // remove white space
 strClose = "";
 if (strTag.indexOf("![CDATA[") == 0) {
      strOpen = "<![CDATA[";
      strClose = "";
      strType = "cdata";
 } else if (strTag.indexOf("!--") == 0) {
      strOpen = "<!--";
      strClose = "-->";
      strType = "comment";
 } else if (strTag.indexOf("?") == 0) {
      strOpen = "<?";
      strClose = "?>";
      strType = "pi";
 } else strType = "element";
 if (strClose != "" && strType!="comment") { // added by Ivan Bruno
      strText = "";
      if (arrElementsUnparsedText[i].indexOf(strClose) > -1) strText = arrElementsUnparsedText[i];
      else {
       for (; i<arrElementsUnparsed.length && arrElementsUnparsedText[i].indexOf(strClose) == -1; i++) {
           strText += arrElementsUnparsedText[i];
       }
       strText += arrElementsUnparsedText[i];
      }
      if (strText.substring(strOpen.length, strText.indexOf(strClose)) != "") {
       xmleLastElement.childElements[xmleLastElement.childElements.length] = new REXML_XMLElement(strType,
"","",xmleLastElement,strText.substring(strOpen.length, strText.indexOf(strClose)));
       if (strType == "cdata") xmleLastElement.text += strText.substring(strOpen.length, strText.indexOf(strClose));
      }
      if (strText.indexOf(strClose)+ strClose.length < strText.length) {
       xmleLastElement.childElements[xmleLastElement.childElements.length] = new REXML_XMLElement("text",
"","",xmleLastElement,strText.substring(strText.indexOf(strClose)+ strClose.length, strText.length));
       if (strType == "cdata") xmleLastElement.text += strText.substring(strText.indexOf(strClose)+ strClose.length,
strText.length);
      }
      continue;
```

```
        }
        if (strText.replace(/ */, "") == "") strText = "";
        if (arrElementsUnparsed[i].substring(1,2) != "/") {
            if (iElements == 0) {
             xmleLastElement = this.rootElement = new REXML_XMLElement(strType, strTag,strAttributes,null,strText);
             iElements++;
             if (strText != "") xmleLastElement.childElements[xmleLastElement.childElements.length] = new
    REXML_XMLElement("text", "","",xmleLastElement,strText);
            } else if (arrElementsUnparsed[i].substring(arrElementsUnparsed[i].length-2,arrElementsUnparsed[i].length-1) != "/") {
             xmleLastElement = xmleLastElement.childElements[xmleLastElement.childElements.length] = new
    REXML_XMLElement(strType, strTag,strAttributes,xmleLastElement,"");
             iElements++;
             if (strText != "") {
                 xmleLastElement.text += strText;
                 xmleLastElement.childElements[xmleLastElement.childElements.length] = new REXML_XMLElement("text",
    "","",xmleLastElement,strText);
              }
            } else {
             xmleLastElement.childElements[xmleLastElement.childElements.length] = new REXML_XMLElement(strType,
    strTag,strAttributes,xmleLastElement,strText);
             if (strText != "") xmleLastElement.childElements[xmleLastElement.childElements.length] = new
    REXML_XMLElement("text", "","",xmleLastElement,strText);
            }
         } else {
            xmleLastElement = xmleLastElement.parentElement;
            iElements--;
            if (xmleLastElement && strText != "") {
             xmleLastElement.text += strText;
             xmleLastElement.childElements[xmleLastElement.childElements.length] = new REXML_XMLElement("text",
    "","",xmleLastElement,strText);
            }
         }
        }
    }
    function REXML_XMLElement(strType, strName, strAttributes, xmlParent, strText) {
        this.type = strType;
        this.name = strName;
        this.attributeString = strAttributes;
        this.attributes = null;
        this.childElements = new Array();
        this.parentElement = xmlParent;
        this.text = strText; // text of element
        this.getText = REXML_XMLElement_getText; // text of element and child elements
        this.childElement = REXML_XMLElement_childElement;
        this.attribute = REXML_XMLElement_attribute;
    }
        function REXML_XMLElement_getText() {
         if (this.type == "text" || this.type == "cdata") {
            return this.text;
```

```
    } else if (this.childElements.length) {
        var L = "";
        for (var i=0; i<this.childElements.length; i++) {
         L += this.childElements[i].getText();
        }
        return L;
    } else return "";
  }

  function REXML_XMLElement_childElement(strElementName) {
    for (var i=0; i<this.childElements.length; i++) if (this.childElements[i].name == strElementName) return this.childElements[i];
    return null;
  }
  function REXML_XMLElement_attribute(strAttributeName) {
    if (!this.attributes) {
        var reAttributes = new RegExp(" ([^= ]*)="","g"); // matches attributes
        if (this.attributeString.match(reAttributes) && this.attributeString.match(reAttributes).length) {
         var arrAttributes = this.attributeString.match(reAttributes);
         if (!arrAttributes.length) arrAttributes = null;
         else for (var j=0; j<arrAttributes.length; j++) {
            arrAttributes[j] = new Array(
                (arrAttributes[j]+"").replace(/[= ]/g,""),
                ParseAttribute(this.attributeString, (arrAttributes[j]+"").replace(/[= ]/g,""))
                    );
         }
         this.attributes = arrAttributes;
        }
    }
    if (this.attributes) for (var i=0; i<this.attributes.length; i++) if (this.attributes[i][0] == strAttributeName) return this.attributes[i][1];
    return "";
  }
function JSXMLBuilder() {
 this.XML = "";
 this.elements = new Array();
 Array.prototype.remove = Array_Remove;
 Array.prototype.add = Array_Add;
 this.load = JSXMLBuilder_load;
 this.element = JSXMLBuilder_element;
 this.addElementAt = JSXMLBuilder_addElementAt;
 this.insertElementAt = JSXMLBuilder_insertElementAt;
 this.removeElement = JSXMLBuilder_removeElement;
 this.generateXML = JSXMLBuilder_generateXML;
 this.moveElement = JSXMLBuilder_moveElement;
}
 function JSXMLBuilder_load(strXML, xmleElem) {
    this.XML = strXML;
    if (!xmleElem) {
      if (strXML.length) xmleElem = (new REXML(strXML)).rootElement;
      else return false;
```

```
    }
    var xmlBuilder = new JSXMLIterator(xmleElem);
    while (true) {
      if (xmlBuilder.xmleElem.type == "element") {
          if (xmlBuilder.xmleElem.attributes) {
           this.addElementAt(xmlBuilder.xmleElem.name,xmlBuilder.xmleElem.attributes, xmlBuilder.xmleElem.text,
    this.elements.length, xmlBuilder.iElemLevel);
          } else {
           this.addElementAt(xmlBuilder.xmleElem.name,xmlBuilder.xmleElem.attributeString, xmlBuilder.xmleElem.text,
    this.elements.length, xmlBuilder.iElemLevel);
          }
       }
      if (!xmlBuilder.getNextNode(false)) break;
    }
    for (var i=0; i<this.elements.length; i++) this.elements[i].index = i;
}
function JSXMLBuilder_element(iIndex) {
    return this.elements[iIndex];
}
function JSXMLBuilder_addElementAt(strElement,Attributes,strText,iElemIndex,iElemLevel) {
    iElemIndex = parseInt(iElemIndex);
    iElemLevel = parseInt(iElemLevel);
    if (iElemIndex < 0 || typeof(iElemIndex) != "number" || isNaN(iElemIndex)) iElemIndex = (this.elements.length>0) ?
    this.elements.length-1 : 0;
    if (iElemLevel < 0 || typeof(iElemLevel) != "number" || isNaN(iElemLevel)) iElemLevel = this.elements[iElemIndex-1].level;
    if (!Attributes) Attributes = "";
    var Elem = new Array();
    var iAddIndex = iElemIndex;
    if (iElemIndex > 0) {
      for (var i=iElemIndex; i<this.elements.length; i++) if (this.elements[i].level > iElemLevel) iAddIndex++;
      else if (this.elements[i].level <= this.elements[iElemIndex].level) break;
      Elem = new JSXMLBuilder_XMLElement(strElement,Attributes,strText,iElemLevel+1,this);
    } else {
      Elem = new JSXMLBuilder_XMLElement(strElement,Attributes,strText,1,this);
    }
    this.elements = this.elements.add(iAddIndex,Elem);
    for (var i=iAddIndex; i<this.elements.length; i++) this.elements[i].index = i;
}
function JSXMLBuilder_insertElementAt(strElement,Attributes,strText,iElemIndex,iElemLevel) {
    iElemIndex = parseInt(iElemIndex);
    iElemLevel = parseInt(iElemLevel);
    if (iElemIndex < 0 || typeof(iElemIndex) != "number" || isNaN(iElemIndex)) iElemIndex = (this.elements.length>0) ?
    this.elements.length-1 : 0;
    if (iElemLevel < 0 || typeof(iElemLevel) != "number" || isNaN(iElemLevel)) iElemLevel = this.elements[iElemIndex-1].level;
    if (!Attributes) Attributes = "";
    var Elem = null;
    var iAddIndex = iElemIndex;
    if (iElemIndex > 0 && iElemLevel > 0) {
      Elem = new JSXMLBuilder_XMLElement(strElement,Attributes,strText,iElemLevel+1,this);
```

```
    } else {
      Elem = new JSXMLBuilder_XMLElement(strElement,Attributes,strText,1,this);
    }
    this.elements = this.elements.add(iAddIndex,Elem);
    for (var i=iAddIndex; i<this.elements.length; i++) this.elements[i].index = i;
}
function JSXMLBuilder_removeElement(iElemIndex) {
    iElemIndex = parseInt(iElemIndex);
    for (var iAfterElem=iElemIndex+1; iAfterElem<this.elements.length; iAfterElem++) if (this.elements[iAfterElem].level <
    this.elements[iElemIndex].level+1) break;
    this.elements = this.elements.slice(0,iElemIndex).concat(this.elements.slice(iAfterElem,this.elements.length));
    for (var i=iElemIndex; i<this.elements.length; i++) this.elements[i].index = i;
}
function JSXMLBuilder_moveElement(iElem1Index,iElem2Index) {
    var arrElem1Elements = new Array(this.elements[iElem1Index]);
    var arrElem2Elements = new Array(this.elements[iElem2Index]);
    for (var i=iElem1Index; i<this.elements.length; i++) if (this.elements[i].level > this.elements[iElem1Index].level)
    arrElem1Elements[arrElem1Elements.length] = this.elements[i]; else if (i>iElem1Index) break;
    for (var i=iElem2Index; i<this.elements.length; i++) if (this.elements[i].level > this.elements[iElem2Index].level)
    arrElem2Elements[arrElem2Elements.length] = this.elements[i]; else if (i>iElem2Index) break;
    var arrMovedElements = new Array();
    if (iElem1Index < iElem2Index) {
      for (i=0; i<iElem1Index; i++) arrMovedElements[arrMovedElements.length] = this.elements[i]; // start to the 1st element
      for (i=iElem1Index+arrElem1Elements.length; i<iElem2Index+arrElem2Elements.length; i++)
    arrMovedElements[arrMovedElements.length] = this.elements[i]; // end of 1st element to end of 2nd element
      for (i=0; i<arrElem1Elements.length; i++) arrMovedElements[arrMovedElements.length] = arrElem1Elements[i]; // 1st element
    and all child elements
      for (i=iElem2Index+arrElem2Elements.length; i<this.elements.length; i++) arrMovedElements[arrMovedElements.length] =
    this.elements[i]; // end of 2nd element to end
      this.elements = arrMovedElements;
    } else {
      for (i=0; i<iElem2Index; i++) arrMovedElements[arrMovedElements.length] = this.elements[i]; // start to the 2nd element
      for (i=0; i<arrElem1Elements.length; i++) arrMovedElements[arrMovedElements.length] = arrElem1Elements[i]; // 1st element
    and all child elements
      for (i=iElem2Index; i<iElem1Index; i++) arrMovedElements[arrMovedElements.length] = this.elements[i]; // 2nd element to 1st
    element
      for (i=iElem1Index+arrElem1Elements.length; i<this.elements.length; i++) arrMovedElements[arrMovedElements.length] =
    this.elements[i]; // end of 1st element to end
      this.elements = arrMovedElements;
    }
    for (var i=0; i<this.elements.length; i++) this.elements[i].index = i;
}
function JSXMLBuilder_generateXML(bXMLTag) {
    var strXML = "";
    var arrXML = new Array();
    if (bXMLTag) strXML += '<?xml version="1.0"?>\n\n'
    for (var i=0; i<this.elements.length; i++) {
      strXML += RepeatChar("\t",this.elements[i].level-1);
      strXML += "<" + this.element(i).name // open tag
```

```
        if (this.element(i).attributes) {
            for (var j=0; j<this.element(i).attributes.length; j++) { // set attributes
             if (this.element(i).attributes[j]) {
                strXML += ' ' + this.element(i).attributes[j][0] + '="' + this.element(i).attributes[j][1] + '"';
             }
            }
        } else strXML += this.element(i).attributeString.replace(/[\/>]$/gi, "");
        if (((this.elements[i+1] && this.elements[i+1].level <= this.elements[i].level) || // next element is a lower or equal to
            (!this.elements[i+1] && this.elements[i-1])) // no next element, previous element
            && this.element(i).text == "") {
            strXML += "/";
        }
        strXML += ">";
        if (this.element(i).text != "") strXML += this.element(i).text;
        else strXML += "\n";
        if (((this.elements[i+1] && this.elements[i+1].level <= this.elements[i].level) || // next element is a lower or equal to
            (!this.elements[i+1] && this.elements[i-1])) // no next element, previous element
            && this.element(i).text != "") strXML += "</" + this.element(i).name + ">\n";
        if (!this.elements[i+1]) {
            lastelem = i;
            for (var j=i; j>-1; j--) {
             if (this.elements[j].level >= this.elements[i].level) continue;
             else {
                if (this.elements[j].level < this.elements[lastelem].level) {
                    strXML += RepeatChar("\t",this.elements[j].level-1) + "</" + this.element(j).name + ">\n";
                    lastelem = j;
                }
             }
            }
        } else {
            if (this.elements[i+1].level < this.elements[i].level) {
             lastelem = i;
             for (var j=i; this.elements[j].level>=this.elements[i+1].level; j--) {
                if (this.elements[i] && this.elements[j] && this.elements[j].level < this.elements[i].level && this.elements[j].level <
    this.elements[lastelem].level) {
                    strXML += RepeatChar("\t",this.elements[j].level-1) + "</" + this.element(j).name + ">\n";
                    lastelem = j;
                }
             }
            }
        }
        if (strXML.length > 1000) {
            arrXML[arrXML.length] = strXML;
            strXML = "";
        }
    }
    arrXML[arrXML.length] = strXML;
    return arrXML.join("");
}
```

```
function JSXMLBuilder_XMLElement(strName,Attributes,strText,iLevel,xmlBuilder) {
    this.type = "element";
    this.name = strName;
    this.attributes = (typeof(Attributes) != "string") ? Attributes : null;
    this.attributeString = (typeof(Attributes) == "string") ? Attributes : "";
    this.text = strText;
    this.level = iLevel;
    this.index = -1;
    this.xmlBuilder = xmlBuilder;
    this.parseAttributes = JSXMLBuilder_XMLElement_parseAttributes;
    this.attribute = JSXMLBuilder_XMLElement_attribute;
    this.setAttribute = JSXMLBuilder_XMLElement_setAttribute;
    this.removeAttribute = JSXMLBuilder_XMLElement_removeAttribute;
    this.parentElement = JSXMLBuilder_XMLElement_parentElement;
    this.childElement = JSXMLBuilder_XMLElement_childElement;
}
    function JSXMLBuilder_XMLElement_parseAttributes() {
      if (!this.attributes) {
            var reAttributes = new RegExp(" ([^= ]*)="","g"); // matches attributes
            if (this.attributeString.match(reAttributes) && this.attributeString.match(reAttributes).length) {
             var arrAttributes = this.attributeString.match(reAttributes);
             if (!arrAttributes.length) arrAttributes = null;
             else for (var j=0; j<arrAttributes.length; j++) {
                 arrAttributes[j] = new Array(
                     (arrAttributes[j]+"").replace(/[= ]/g,""),
                     ParseAttribute(this.attributeString, (arrAttributes[j]+"").replace(/[= ]/g,""))
                             );
             }
             this.attributes = arrAttributes;
            }
      }
    }

    function JSXMLBuilder_XMLElement_attribute(AttributeName) {
      if (!this.attributes) this.parseAttributes();
      if (this.attributes) for (var i=0; i<this.attributes.length; i++) if (this.attributes[i][0] == AttributeName) return this.attributes[i][1];
      return "";
    }
    function JSXMLBuilder_XMLElement_setAttribute(AttributeName,Value) {
      if (!this.attributes) this.parseAttributes();
      if (this.attributes) for (var i=0; i<this.attributes.length; i++) if (this.attributes[i][0] == AttributeName) {
            this.attributes[i][1] = Value;
            return;
      }
      this.attributes[this.attributes.length] = new Array(AttributeName,Value);
    }
    function JSXMLBuilder_XMLElement_removeAttribute(AttributeName,Value) {
      if (!this.attributes) this.parseAttributes();
      if (this.attributes) for (var i=0; i<this.attributes.length; i++) if (this.attributes[i][0] == AttributeName) {
```

```
            this.attributes = this.attributes.remove(i);
            return;
         }
        }
        function JSXMLBuilder_XMLElement_parentElement() {
         for (var i=this.index; this.xmlBuilder.element(i) && this.xmlBuilder.element(i).level != this.level-1; i--);
         return this.xmlBuilder.element(i);
        }
        function JSXMLBuilder_XMLElement_childElement(Child) {
         var iFind = -1;
         for (var i=this.index+1; i<this.xmlBuilder.elements.length; i++) {
            if (this.xmlBuilder.elements[i].level == this.level+1) {
             iFind++;
             if (iFind == Child || this.xmlBuilder.elements[i].name == Child) return this.xmlBuilder.elements[i];
            } else if (this.xmlBuilder.elements[i].level <= this.level) break;
         }
         return null;
        }
       function JSXMLIterator(xmleElem) {
        this.xmleElem = xmleElem;

        this.iElemIndex = 0;
        this.arrElemIndex = new Array(0);
        this.iElemLevel = 0;
        this.iElem = 0;
        this.arrElemIndex[this.iElemLevel] = -1;
        this.getNextNode = JSXMLIterator_getNextNode;
       }
        function JSXMLIterator_getNextNode() {
         if (!this.xmleElem || this.iElemLevel<0)     return false;
         if (this.xmleElem.childElements.length) {  // move up
          this.arrElemIndex[this.iElemLevel]++;
          this.iElemIndex++;
          this.iElemLevel++;
          this.arrElemIndex[this.iElemLevel] = 0;
          this.xmleElem = this.xmleElem.childElements[0];
         } else { // move next
          this.iElemIndex++;
          this.arrElemIndex[this.iElemLevel]++;
          if (this.xmleElem.parentElement && this.xmleElem.parentElement.childElements.length && this.arrElemIndex[this.iElemLevel] <
         this.xmleElem.parentElement.childElements.length) this.xmleElem =
         this.xmleElem.parentElement.childElements[this.arrElemIndex[this.iElemLevel]];
          else {
            if (this.iElemLevel>0) { // move down
             for (; this.iElemLevel > 0; this.iElemLevel--) {
               if (this.xmleElem.parentElement && this.xmleElem.parentElement.childElements[this.arrElemIndex[this.iElemLevel]]) {
                  this.xmleElem = this.xmleElem.parentElement.childElements[this.arrElemIndex[this.iElemLevel]];
                  this.iElemLevel++;
                  this.arrElemIndex = this.arrElemIndex.slice(0,this.iElemLevel+1);
```

```
                    break;
                } else {
                    this.xmleElem = this.xmleElem.parentElement;
                }
            }
            this.iElemLevel--;
           } else {
            return false;
           }
       }
      }
      return (typeof(this.xmleElem) == "object" && this.iElemLevel > -1);
 }
 function ParseAttribute(str,Attribute) {
  var str = str +  ">";
  if (str.indexOf(Attribute + "=")>-1) var Attr = new RegExp(".*" + Attribute + "='([^']*)'.*>");
  else if (str.indexOf(Attribute + '=')>-1) var Attr = new RegExp(".*" + Attribute + '="([^"]*)".*>');
  return str.replace(Attr, "$1");
 }
 function Array_Remove(c) {
  var tmparr = new Array();
  for (var i=0; i<this.length; i++) if (i!=c) tmparr[tmparr.length] = this[i];
  return tmparr;
 }
 function Array_Add(c, cont) {
  var tmparr = new Array();
  for (var i=0; i<this.length; i++) {
      if (i==c) tmparr[tmparr.length] = cont;
      tmparr[tmparr.length] = this[i];
  }
  if (!tmparr[c]) tmparr[c] = cont;
  return tmparr;
 }
 function RepeatChar(sChar,iNum) {
  var L = "";
  for (var i=0; i<iNum; i++) L += sChar;
  return L;
 }
   ]]>
   </JS_Script>
- <JS_Script name="packaging">
- <![CDATA[
 function packageResource(resource, mimeType){
    if((resource.right == play) &&
       (mimeType.indexOf("text") > -1 || mimeType.indexOf("image") > -1)){
       resource.right = "display";
    }

    // it must distinguish between play and display
    var rights = "--" + resource.right + " ";
```

```
        if(resource.count != null){
            rights += resource.count;
        }
        rights += ":";

        if(resource.from != null){
            rights += resource.from;
        }
        rights += ":";

        if(resource.to != null){
            rights += resource.to;
        }
        rights += ":";

        var params = rights + " " + uriResources + resource.ID + mimeTypetoExt(mimeType) + " " + uriContents + " " + uriRights;

        execute(uriPackager + params);
    }
    ]]>
    </JS_Script>
    </Rule_Body>
  <Dependencies />
    </AXCP_Rule>
    </Definition>
    </Rule>
```

7.3 AXMEDIS CP GRID Script: License_parser.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
- <Rule xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="Rule_Axmedis.xsd">
- <Header>
  <Rule_Name>license_parser</Rule_Name>
  <AXRID>axcprule:2fee4e75-7d59-4fa7-b13f-4191ad5db577</AXRID>
  <Rule_Version />
  <Rule_Type>AXCP</Rule_Type>
  <Software_Name />
  <Version_of_software />
  <Date_of_production>2006-04-24</Date_of_production>
  <Author />
  <Affiliation />
  <URL />
  <Comment />
  <Last_Modifications>2006-08-13</Last_Modifications>
  <Terminal_ID />
  <Cost />
  <Work_Item_ID />
    </Header>
- <Schedule>
- <Run>
  <Date>2006-04-24</Date>
  <Time>03:53:46</Time>
  <Periodicity Unit="Day">0</Periodicity>
  <Expiration_Date>2006-04-24</Expiration_Date>
  <Expiration_Time>03:53:46</Expiration_Time>
    </Run>
  <Status>Inactive</Status>
```

```
    </Schedule>
- <Definition>
- <AXCP_Rule>
- <Arguments>
  <Parameter Name="filename"
     Type="String">E:\Projects\JScript\AXMEDIS\license\axlicense.xml</Parameter>
    </Arguments>
- <Rule_Body>
  - <JS_Script name="main">
      - <![CDATA[
        viewer();
        resolver();
    ]]>
    </JS_Script>
  - <JS_Script name="xml">
      - <![CDATA[
        ////////////////////////////////////////////////////////////
        ////////// JSXML XML Tools            ////////////////////
        ////////// Ver 1.2 Jun 18 2001        ////////////////////
        ////////// Copyright 2000 Peter Tracey ////////////////////
        ////////// http://jsxml.homestead.com/ ////////////////////
        ////
        //// Objects:
        ////
        //// REXML  ////   Regular Expression-based XML parser
        ////
        //// JSXMLIterator ////   Iterates through the tree structure without recursion
        ////
        //// JSXMLBuilder ////   Loads xml into a linear structure and provides
        //// interface for adding and removing elements //// and setting attributes, generates XML
        ////
        //// Utility functions:////
        //// ParseAttribute
        ////   Takes string of attibutes and attribute name
        ////   Returns attribute value
        ////
        //// Array_Remove
        ////   Removes element in array
        ////
        //// Array_Add
        ////   Adds element to array
        ////
        //// RepeatChar
        ////   Repeats string specified number of times
        ////
        ////////////////////////////////////////////////////////////
        function REXML(XML) {
          this.XML = XML;
          this.rootElement = null;
          this.parse = REXML_parse;
          if (this.XML && this.XML != "") this.parse();
        }
```

```
function REXML_parse() {
 var reTag = new RegExp("<([^>/ ]*)([^>]*)>","g"); // matches that tag name $1 and attribute string $2
 var reTagText = new RegExp("<([^>/ ]*)([^>]*)>([^<]*)","g"); // matches tag name $1, attribute string $2, and text $3
 var strType = "";
 var strTag = "";
 var strText = "";
 var strAttributes = "";
 var strOpen = "";
 var strClose = "";
 var iElements = 0;
 var xmleLastElement = null;
 if (this.XML.length == 0) return;
 var arrElementsUnparsed = this.XML.match(reTag);
 var arrElementsUnparsedText = this.XML.match(reTagText);
 var i=0;
 if (arrElementsUnparsed[0].replace(reTag, "$1") == "?xml") i++;
 for (; i<arrElementsUnparsed.length; i++) {
     strTag = arrElementsUnparsed[i].replace(reTag,"$1");
     strAttributes = arrElementsUnparsed[i].replace(reTag,"$2");
     strText = arrElementsUnparsedText[i].replace(reTagText,"$3").replace(/[\r\n\t ]+/g, " "); // remove white space
     strClose = "";
     if (strTag.indexOf("![CDATA[") == 0) {
         strOpen = "<![CDATA[";
         strClose = "";
         strType = "cdata";
     } else if (strTag.indexOf("!--") == 0) {
         strOpen = "<!--";
         strClose = "-->";
         strType = "comment";
     } else if (strTag.indexOf("?") == 0) {
         strOpen = "<?";
         strClose = "?>";
         strType = "pi";
     } else strType = "element";
     if (strClose != "") {
         strText = "";
         if (arrElementsUnparsedText[i].indexOf(strClose) > -1) strText = arrElementsUnparsedText[i];
         else {
             for (; i<arrElementsUnparsed.length && arrElementsUnparsedText[i].indexOf(strClose) == -1; i++) {
               strText += arrElementsUnparsedText[i];
             }
             strText += arrElementsUnparsedText[i];
         }
         if (strText.substring(strOpen.length, strText.indexOf(strClose)) != "") {
             xmleLastElement.childElements[xmleLastElement.childElements.length] = new REXML_XMLElement(strType,
"","",xmleLastElement,strText.substring(strOpen.length, strText.indexOf(strClose)));
             if (strType == "cdata") xmleLastElement.text += strText.substring(strOpen.length, strText.indexOf(strClose));
         }
         if (strText.indexOf(strClose)+ strClose.length < strText.length) {
```

```
                xmleLastElement.childElements[xmleLastElement.childElements.length] = new REXML_XMLElement("text",
"","",xmleLastElement,strText.substring(strText.indexOf(strClose)+ strClose.length, strText.length));
                if (strType == "cdata") xmleLastElement.text += strText.substring(strText.indexOf(strClose)+ strClose.length,
strText.length);
            }
            continue;
        }
        if (strText.replace(/ */, "") == "") strText = "";
        if (arrElementsUnparsed[i].substring(1,2) != "/") {
            if (iElements == 0) {
                xmleLastElement = this.rootElement = new REXML_XMLElement(strType, strTag,strAttributes,null,strText);
                iElements++;
                if (strText != "") xmleLastElement.childElements[xmleLastElement.childElements.length] = new
REXML_XMLElement("text", "","",xmleLastElement,strText);
            } else if (arrElementsUnparsed[i].substring(arrElementsUnparsed[i].length-2,arrElementsUnparsed[i].length-1) != "/")
{
                xmleLastElement = xmleLastElement.childElements[xmleLastElement.childElements.length] = new
REXML_XMLElement(strType, strTag,strAttributes,xmleLastElement,"");
                iElements++;
                if (strText != "") {
                  xmleLastElement.text += strText;
                  xmleLastElement.childElements[xmleLastElement.childElements.length] = new REXML_XMLElement("text",
"","",xmleLastElement,strText);
                }
            } else {
                xmleLastElement.childElements[xmleLastElement.childElements.length] = new REXML_XMLElement(strType,
strTag,strAttributes,xmleLastElement,strText);
                if (strText != "") xmleLastElement.childElements[xmleLastElement.childElements.length] = new
REXML_XMLElement("text", "","",xmleLastElement,strText);
            }
        } else {
            xmleLastElement = xmleLastElement.parentElement;
            iElements--;
            if (xmleLastElement && strText != "") {
                xmleLastElement.text += strText;
                xmleLastElement.childElements[xmleLastElement.childElements.length] = new REXML_XMLElement("text",
"","",xmleLastElement,strText);
            }
        }
    }
  }
  function REXML_XMLElement(strType, strName, strAttributes, xmlParent, strText) {
    this.type = strType;
    this.name = strName;
    this.attributeString = strAttributes;
    this.attributes = null;
    this.childElements = new Array();
    this.parentElement = xmlParent;
    this.text = strText; // text of element
```

```
    this.getText = REXML_XMLElement_getText; // text of element and child elements
    this.childElement = REXML_XMLElement_childElement;
    this.attribute = REXML_XMLElement_attribute;
  }
    function REXML_XMLElement_getText() {
        if (this.type == "text" || this.type == "cdata") {
            return this.text;
        } else if (this.childElements.length) {
            var L = "";
            for (var i=0; i<this.childElements.length; i++) {
                L += this.childElements[i].getText();
            }
            return L;
        } else return "";
    }

    function REXML_XMLElement_childElement(strElementName) {
        for (var i=0; i<this.childElements.length; i++) if (this.childElements[i].name == strElementName) return
this.childElements[i];
        return null;
    }
    function REXML_XMLElement_attribute(strAttributeName) {
        if (!this.attributes) {
            var reAttributes = new RegExp(" ([^= ]*)=","g"); // matches attributes
            if (this.attributeString.match(reAttributes) && this.attributeString.match(reAttributes).length) {
                var arrAttributes = this.attributeString.match(reAttributes);
                if (!arrAttributes.length) arrAttributes = null;
                else for (var j=0; j<arrAttributes.length; j++) {
                  arrAttributes[j] = new Array(
                    (arrAttributes[j]+"").replace(/[= ]/g,""),
                    ParseAttribute(this.attributeString, (arrAttributes[j]+"").replace(/[= ]/g,""))
                            );
                }
                this.attributes = arrAttributes;
            }
        }
        if (this.attributes) for (var i=0; i<this.attributes.length; i++) if (this.attributes[i][0] == strAttributeName) return
this.attributes[i][1];
        return "";
    }
function JSXMLBuilder() {
  this.XML = "";
  this.elements = new Array();
  Array.prototype.remove = Array_Remove;
  Array.prototype.add = Array_Add;
  this.load = JSXMLBuilder_load;
  this.element = JSXMLBuilder_element;
  this.addElementAt = JSXMLBuilder_addElementAt;
  this.insertElementAt = JSXMLBuilder_insertElementAt;
```

```
  this.removeElement = JSXMLBuilder_removeElement;
  this.generateXML = JSXMLBuilder_generateXML;
  this.moveElement = JSXMLBuilder_moveElement;
}
  function JSXMLBuilder_load(strXML, xmleElem) {
   this.XML = strXML;
   if (!xmleElem) {
       if (strXML.length) xmleElem = (new REXML(strXML)).rootElement;
       else return false;
   }
   var xmlBuilder = new JSXMLIterator(xmleElem);
   while (true) {
       if (xmlBuilder.xmleElem.type == "element") {
           if (xmlBuilder.xmleElem.attributes) {
               this.addElementAt(xmlBuilder.xmleElem.name,xmlBuilder.xmleElem.attributes, xmlBuilder.xmleElem.text,
this.elements.length, xmlBuilder.iElemLevel);
           } else {
               this.addElementAt(xmlBuilder.xmleElem.name,xmlBuilder.xmleElem.attributeString, xmlBuilder.xmleElem.text,
this.elements.length, xmlBuilder.iElemLevel);
           }
       }
       if (!xmlBuilder.getNextNode(false)) break;
   }
   for (var i=0; i<this.elements.length; i++) this.elements[i].index = i;
  }
  function JSXMLBuilder_element(iIndex) {
   return this.elements[iIndex];
  }
  function JSXMLBuilder_addElementAt(strElement,Attributes,strText,iElemIndex,iElemLevel) {
   iElemIndex = parseInt(iElemIndex);
   iElemLevel = parseInt(iElemLevel);
   if (iElemIndex < 0 || typeof(iElemIndex) != "number" || isNaN(iElemIndex)) iElemIndex = (this.elements.length>0) ?
this.elements.length-1 : 0;
   if (iElemLevel < 0 || typeof(iElemLevel) != "number" || isNaN(iElemLevel)) iElemLevel = this.elements[iElemIndex-1].level;
   if (!Attributes) Attributes = "";
   var Elem = new Array();
   var iAddIndex = iElemIndex;
   if (iElemIndex > 0) {
       for (var i=iElemIndex; i<this.elements.length; i++) if (this.elements[i].level > iElemLevel) iAddIndex++;
       else if (this.elements[i].level <= this.elements[iElemIndex].level) break;
       Elem = new JSXMLBuilder_XMLElement(strElement,Attributes,strText,iElemLevel+1,this);
   } else {
       Elem = new JSXMLBuilder_XMLElement(strElement,Attributes,strText,1,this);
   }
   this.elements = this.elements.add(iAddIndex,Elem);
   for (var i=iAddIndex; i<this.elements.length; i++) this.elements[i].index = i;
  }
  function JSXMLBuilder_insertElementAt(strElement,Attributes,strText,iElemIndex,iElemLevel) {
   iElemIndex = parseInt(iElemIndex);
```

```
    iElemLevel = parseInt(iElemLevel);
    if (iElemIndex < 0 || typeof(iElemIndex) != "number" || isNaN(iElemIndex)) iElemIndex = (this.elements.length>0) ?
this.elements.length-1 : 0;
    if (iElemLevel < 0 || typeof(iElemLevel) != "number" || isNaN(iElemLevel)) iElemLevel = this.elements[iElemIndex-1].level;
    if (!Attributes) Attributes = "";
    var Elem = null;
    var iAddIndex = iElemIndex;
    if (iElemIndex > 0 && iElemLevel > 0) {
        Elem = new JSXMLBuilder_XMLElement(strElement,Attributes,strText,iElemLevel+1,this);
    } else {
        Elem = new JSXMLBuilder_XMLElement(strElement,Attributes,strText,1,this);
    }
    this.elements = this.elements.add(iAddIndex,Elem);
    for (var i=iAddIndex; i<this.elements.length; i++) this.elements[i].index = i;
  }
  function JSXMLBuilder_removeElement(iElemIndex) {
    iElemIndex = parseInt(iElemIndex);
    for (var iAfterElem=iElemIndex+1; iAfterElem<this.elements.length; iAfterElem++) if (this.elements[iAfterElem].level <
this.elements[iElemIndex].level+1) break;
    this.elements = this.elements.slice(0,iElemIndex).concat(this.elements.slice(iAfterElem,this.elements.length));
    for (var i=iElemIndex; i<this.elements.length; i++) this.elements[i].index = i;
  }
  function JSXMLBuilder_moveElement(iElem1Index,iElem2Index) {
    var arrElem1Elements = new Array(this.elements[iElem1Index]);
    var arrElem2Elements = new Array(this.elements[iElem2Index]);
    for (var i=iElem1Index; i<this.elements.length; i++) if (this.elements[i].level > this.elements[iElem1Index].level)
arrElem1Elements[arrElem1Elements.length] = this.elements[i]; else if (i>iElem1Index) break;
    for (var i=iElem2Index; i<this.elements.length; i++) if (this.elements[i].level > this.elements[iElem2Index].level)
arrElem2Elements[arrElem2Elements.length] = this.elements[i]; else if (i>iElem2Index) break;
    var arrMovedElements = new Array();
    if (iElem1Index < iElem2Index) {
        for (i=0; i<iElem1Index; i++) arrMovedElements[arrMovedElements.length] = this.elements[i]; // start to the 1st element
        for (i=iElem1Index+arrElem1Elements.length; i<iElem2Index+arrElem2Elements.length; i++)
arrMovedElements[arrMovedElements.length] = this.elements[i]; // end of 1st element to end of 2nd element
        for (i=0; i<arrElem1Elements.length; i++) arrMovedElements[arrMovedElements.length] = arrElem1Elements[i]; // 1st
element and all child elements
        for (i=iElem2Index+arrElem2Elements.length; i<this.elements.length; i++) arrMovedElements[arrMovedElements.length]
= this.elements[i]; // end of 2nd element to end
        this.elements = arrMovedElements;
    } else {
        for (i=0; i<iElem2Index; i++) arrMovedElements[arrMovedElements.length] = this.elements[i]; // start to the 2nd element
        for (i=0; i<arrElem1Elements.length; i++) arrMovedElements[arrMovedElements.length] = arrElem1Elements[i]; // 1st
element and all child elements
        for (i=iElem2Index; i<iElem1Index; i++) arrMovedElements[arrMovedElements.length] = this.elements[i]; // 2nd element
to 1st element
        for (i=iElem1Index+arrElem1Elements.length; i<this.elements.length; i++) arrMovedElements[arrMovedElements.length]
= this.elements[i]; // end of 1st element to end
        this.elements = arrMovedElements;
    }
```

```
 for (var i=0; i<this.elements.length; i++) this.elements[i].index = i;
}
function JSXMLBuilder_generateXML(bXMLTag) {
 var strXML = "";
 var arrXML = new Array();
 if (bXMLTag) strXML += '<?xml version="1.0"?>\n\n'
 for (var i=0; i<this.elements.length; i++) {
    strXML += RepeatChar("\t",this.elements[i].level-1);
    strXML += "<" + this.element(i).name // open tag
    if (this.element(i).attributes) {
        for (var j=0; j<this.element(i).attributes.length; j++) { // set attributes
            if (this.element(i).attributes[j]) {
              strXML += ' ' + this.element(i).attributes[j][0] + '="' + this.element(i).attributes[j][1] + '"';
            }
        }
    } else strXML += this.element(i).attributeString.replace(/[\/>]$/gi, "");
    if (((this.elements[i+1] && this.elements[i+1].level <= this.elements[i].level) || // next element is a lower or equal to
        (!this.elements[i+1] && this.elements[i-1])) // no next element, previous element
        && this.element(i).text == "") {
        strXML += "/";
    }
    strXML += ">";
    if (this.element(i).text != "") strXML += this.element(i).text;
    else strXML += "\n";
    if (((this.elements[i+1] && this.elements[i+1].level <= this.elements[i].level) || // next element is a lower or equal to
        (!this.elements[i+1] && this.elements[i-1])) // no next element, previous element
        && this.element(i).text != "") strXML += "</" + this.element(i).name + ">\n";
    if (!this.elements[i+1]) {
        lastelem = i;
        for (var j=i; j>-1; j--) {
            if (this.elements[j].level >= this.elements[i].level) continue;
            else {
              if (this.elements[j].level < this.elements[lastelem].level) {
                strXML += RepeatChar("\t",this.elements[j].level-1) + "</" + this.element(j).name + ">\n";
                lastelem = j;
              }
            }
        }
    } else {
        if (this.elements[i+1].level < this.elements[i].level) {
            lastelem = i;
            for (var j=i; this.elements[j].level>=this.elements[i+1].level; j--) {
              if (this.elements[i] && this.elements[j] && this.elements[j].level < this.elements[i].level && this.elements[j].level
< this.elements[lastelem].level) {
                strXML += RepeatChar("\t",this.elements[j].level-1) + "</" + this.element(j).name + ">\n";
                lastelem = j;
              }
            }
        }
```

```
        }
        if (strXML.length > 1000) {
            arrXML[arrXML.length] = strXML;
            strXML = "";
        }
    }
    arrXML[arrXML.length] = strXML;
    return arrXML.join("");
}
function JSXMLBuilder_XMLElement(strName,Attributes,strText,iLevel,xmlBuilder) {
    this.type = "element";
    this.name = strName;
    this.attributes = (typeof(Attributes) != "string") ? Attributes : null;
    this.attributeString = (typeof(Attributes) == "string") ? Attributes : "";
    this.text = strText;
    this.level = iLevel;
    this.index = -1;
    this.xmlBuilder = xmlBuilder;
    this.parseAttributes = JSXMLBuilder_XMLElement_parseAttributes;
    this.attribute = JSXMLBuilder_XMLElement_attribute;
    this.setAttribute = JSXMLBuilder_XMLElement_setAttribute;
    this.removeAttribute = JSXMLBuilder_XMLElement_removeAttribute;
    this.parentElement = JSXMLBuilder_XMLElement_parentElement;
    this.childElement = JSXMLBuilder_XMLElement_childElement;
}
    function JSXMLBuilder_XMLElement_parseAttributes() {
        if (!this.attributes) {
            var reAttributes = new RegExp(" ([^= ]*)=","g"); // matches attributes
            if (this.attributeString.match(reAttributes) && this.attributeString.match(reAttributes).length) {
                var arrAttributes = this.attributeString.match(reAttributes);
                if (!arrAttributes.length) arrAttributes = null;
                else for (var j=0; j<arrAttributes.length; j++) {
                  arrAttributes[j] = new Array(
                    (arrAttributes[j]+"").replace(/[= ]/g,""),
                    ParseAttribute(this.attributeString, (arrAttributes[j]+"").replace(/[= ]/g,""))
                            );
                }
                this.attributes = arrAttributes;
            }
        }
    }

    function JSXMLBuilder_XMLElement_attribute(AttributeName) {
        if (!this.attributes) this.parseAttributes();
        if (this.attributes) for (var i=0; i<this.attributes.length; i++) if (this.attributes[i][0] == AttributeName) return
this.attributes[i][1];
        return "";
    }
    function JSXMLBuilder_XMLElement_setAttribute(AttributeName,Value) {
```

```
        if (!this.attributes) this.parseAttributes();
        if (this.attributes) for (var i=0; i<this.attributes.length; i++) if (this.attributes[i][0] == AttributeName) {
            this.attributes[i][1] = Value;
            return;
        }
        this.attributes[this.attributes.length] = new Array(AttributeName,Value);
    }
    function JSXMLBuilder_XMLElement_removeAttribute(AttributeName,Value) {
        if (!this.attributes) this.parseAttributes();
        if (this.attributes) for (var i=0; i<this.attributes.length; i++) if (this.attributes[i][0] == AttributeName) {
            this.attributes = this.attributes.remove(i);
            return;
        }
    }
    function JSXMLBuilder_XMLElement_parentElement() {
        for (var i=this.index; this.xmlBuilder.element(i) && this.xmlBuilder.element(i).level != this.level-1; i--);
        return this.xmlBuilder.element(i);
    }
    function JSXMLBuilder_XMLElement_childElement(Child) {
        var iFind = -1;
        for (var i=this.index+1; i<this.xmlBuilder.elements.length; i++) {
            if (this.xmlBuilder.elements[i].level == this.level+1) {
                iFind++;
                if (iFind == Child || this.xmlBuilder.elements[i].name == Child) return this.xmlBuilder.elements[i];
            } else if (this.xmlBuilder.elements[i].level <= this.level) break;
        }
        return null;
    }
function JSXMLIterator(xmleElem) {
  this.xmleElem = xmleElem;

  this.iElemIndex = 0;
  this.arrElemIndex = new Array(0);
  this.iElemLevel = 0;
  this.iElem = 0;
  this.arrElemIndex[this.iElemLevel] = -1;
  this.getNextNode = JSXMLIterator_getNextNode;
}
  function JSXMLIterator_getNextNode() {
   if (!this.xmleElem || this.iElemLevel<0)  return false;
   if (this.xmleElem.childElements.length) {  // move up
        this.arrElemIndex[this.iElemLevel]++;
        this.iElemIndex++;
        this.iElemLevel++;
        this.arrElemIndex[this.iElemLevel] = 0;
        this.xmleElem = this.xmleElem.childElements[0];
    } else { // move next
        this.iElemIndex++;
        this.arrElemIndex[this.iElemLevel]++;
```

```
        if (this.xmleElem.parentElement && this.xmleElem.parentElement.childElements.length &&
this.arrElemIndex[this.iElemLevel] < this.xmleElem.parentElement.childElements.length) this.xmleElem =
this.xmleElem.parentElement.childElements[this.arrElemIndex[this.iElemLevel]];
        else {
            if (this.iElemLevel>0) { // move down
                for (; this.iElemLevel > 0; this.iElemLevel--) {
                  if (this.xmleElem.parentElement &&
this.xmleElem.parentElement.childElements[this.arrElemIndex[this.iElemLevel]]) {
                    this.xmleElem = this.xmleElem.parentElement.childElements[this.arrElemIndex[this.iElemLevel]];
                    this.iElemLevel++;
                    this.arrElemIndex = this.arrElemIndex.slice(0,this.iElemLevel+1);
                    break;
                  } else {
                    this.xmleElem = this.xmleElem.parentElement;
                  }
                }
                this.iElemLevel--;
            } else {
                return false;
            }
        }
    }
    return (typeof(this.xmleElem) == "object" && this.iElemLevel > -1);
  }
function ParseAttribute(str,Attribute) {
  var str = str +  ">";
  if (str.indexOf(Attribute + "=")>-1) var Attr = new RegExp(".*" + Attribute + "='([^']*)'.*>");
  else if (str.indexOf(Attribute + '=')>-1) var Attr = new RegExp(".*" + Attribute + '="([^"]*)".*>');
  return str.replace(Attr, "$1");
}
function Array_Remove(c) {
  var tmparr = new Array();
  for (var i=0; i<this.length; i++) if (i!=c) tmparr[tmparr.length] = this[i];
  return tmparr;
}
function Array_Add(c, cont) {
  var tmparr = new Array();
  for (var i=0; i<this.length; i++) {
   if (i==c) tmparr[tmparr.length] = cont;
   tmparr[tmparr.length] = this[i];
  }
  if (!tmparr[c]) tmparr[c] = cont;
  return tmparr;
}
function RepeatChar(sChar,iNum) {
  var L = "";
  for (var i=0; i<iNum; i++) L += sChar;
  return L;
}
```

```
]]>
</JS_Script>
- <JS_Script name="oma">
    - <![CDATA[
    function oma(){
      this.issuer = null;
      this.resources = new Array();
      this.print = omaprint;
    }
    function resource(){
      this.ID = null;
      this.right = null;
      this.count = null;
      this.from = null;
      this.to = null;
      this.principal = null;//redundant
      this.print = resourceprint;
    }
    function omaprint(){
      print("issuer: " + this.issuer);

      var i = 0;
      for(i in this.resources){
        this.resources[i].print();
      }
    }
    function resourceprint(){
      print("ID: " + this.ID);
      print("    right: " + this.right);
      print("    count: " + this.count);
      print("    from: " + this.from);
      print("    to: " + this.to);
      print("    principal: " + this.principal);
    }
]]>
</JS_Script>
- <JS_Script name="viewer">
    - <![CDATA[
    var level;
    var tab;
    function viewer(){
      var xml = readFromFile(filename);
      var xmldoc = new REXML(xml);

      level = 0;
      tab = "  ";
      print(xmldoc.rootElement.name)
      show(xmldoc.rootElement);
    }
    function show(element){
      var i = 0;
```

```
            for(i in element.childElements){
               var current = element.childElements[i];
               switch(current.type){
                  case "element":
                     print(RepeatChar(tab, level+1) + current.name);
                     level++;
                     show(current);
                     level--;
                     break;
                  case "text":
                     print(RepeatChar(tab, level+1) + current.text);
                  case "comment":
                  case "pi":
                  case "cdata":
                     break;
               }
            }
         }
      ]]>
      </JS_Script>
   - <JS_Script name="resolver">
      - <![CDATA[
         var omaobj;
         function resolver(){
            var xml = readFromFile(filename);
            var xmldoc = new REXML(xml);

            omaobj = new oma();
            visit(xmldoc.rootElement);

            omaobj.print();
         }
         function visit(element){
            switch(element.name){
               case "r:issuer":
                  omaobj.issuer = resolveIssuer(element);
                  break;
               case "mx:diReference":
                  omaobj.resources[omaobj.resources.length] = resolveResource(element);
                  break;
            }

            var i = 0;
            for(i in element.childElements){
               var current = element.childElements[i];
               if(current.type == "element")
                  visit(current);
            }
         }
         function resolveResource(element){
```

```
    var res = new resource();

    // ID
    res.ID = resolveID(element);

    // properties
    i = 0;
    var parent = element.parentElement;
    for(i in parent.childElements){
       current = parent.childElements[i];
       switch(current.name){
          case "r:keyHolder":
             res.principal = resolvePrincipal(current);
             break;
          case "r:allConditions":
             var conditions = resolveConditions(current);
             res.count = conditions.count;
             res.from = conditions.from;
             res.to = conditions.to;
             break;
          case "mx:play":
          case "mx:print":
          case "mx:execute":
             res.right = current.name;
             break;
       }
    }

    return res;
}
function resolveIssuer(element){
    var keyholder = element.childElement("r:keyHolder");
    var info = keyholder.childElement("r:info");
    var keyname = info.childElement("dsig:KeyName");
    return keyname.text;
}
function resolveID(element){
    var identifier = element.childElement("mx:identifier");
    return identifier.text;
}
function resolvePrincipal(element){
    var info = element.childElement("r:info");
    var keyname = info.childElement("dsig:keyName");
    return keyname.text;
}
function resolveConditions(element){
    var conditions = new Object();

    conditions.count = null;
```

```
            conditions.from = null;
            conditions.to = null;

            var i = 0;
            for(i in element.childElements){
               var current = element.childElements[i];
               switch(current.name){
                  case "sx:exerciseLimit":
                     var count = current.childElement("sx:count");
                     conditions.count = count.text;
                     break;
                  case "r:validityInterval":
                     var from = current.childElement("r:notBefore");
                     var to = current.childElement("r:notAfter");
                     conditions.from = from.text;
                     conditions.to = to.text;
                     break;
               }
            }

            return conditions;
         }
      ]]>
      </JS_Script>
      </Rule_Body>
   <Dependencies />
      </AXCP_Rule>
      </Definition>
      </Rule>
```

7.4 AXMEDIS CP GRID Script:resource_extract_o1.xml rule

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
- <Rule xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="Rule_Axmedis.xsd">
- <Header>
   <Rule_Name>test01</Rule_Name>
   <AXRID>axcprule:23d701f8-480e-423c-9deb-ce465e59b491</AXRID>
   <Rule_Version />
   <Rule_Type>AXCP</Rule_Type>
   <Software_Name />
   <Version_of_software />
   <Date_of_production>2006-03-22</Date_of_production>
   <Author />
   <Affiliation />
   <URL />
   <Comment />
   <Last_Modifications>2006-08-17</Last_Modifications>
   <Terminal_ID />
   <Cost />
   <Work_Item_ID />
      </Header>
- <Schedule>
```

```
- <Run>
  <Date>2006-03-22</Date>
  <Time>21:47:13</Time>
  <Periodicity Unit="Day">0</Periodicity>
  <Expiration_Date>2006-03-22</Expiration_Date>
  <Expiration_Time>21:47:13</Expiration_Time>
    </Run>
  <Status>Inactive</Status>
    </Schedule>
- <Definition>
- <AXCP_Rule>
- <Arguments>
  <Parameter Name="output" Type="String">axobj01.xml</Parameter>
  <Parameter Name="input" Type="String">axobj01.xml</Parameter>
  <Parameter Name="bmp" Type="String">tile.bmp</Parameter>
  <Parameter Name="path"
    Type="String">E:\Projects\JScript\AXMEDIS\axobj\</Parameter>
  <Parameter Name="packager" Type="String">C:\Program Files\Sony Ericsson\DRM
    Packager\DRMPackager.exe</Parameter>
    </Arguments>
- <Rule_Body>
  - <JS_Script name="main">
      - <![CDATA[
      var level;

      function load(){
        var axobj = new AxmedisObject(path + input);
        return axobj;
      }

      function extract(obj){
        var contents = obj.getContent();
        print(level + " = " + obj.AXOID);
        if(contents != null){
          var index;
          for(index in contents){
            var content = contents[index];
            if(content instanceof AxmedisObject){
              level++;
              extract(content);
              level--;
            }
            else{
              var filename = path + level + "_" + content.contentID;
              print(filename);
```

```
                content.save(filename);
            }
        }
    }
}

function make(){
    var axobj = new AxmedisObject();
    if(axobj != null){
        var bmpres = new AxResource();
        bmpres.load(path + bmp);
        axobj.addContent(bmpres);

        //var axres = load();
        //axobj.addContent(axres);

        axobj.save(path + output);
    }
}

function main(){
    //make();

    var axobj = load();
    //level = 0;
    //extract(axobj);

    //execute(packager);
    return 0;
}

main();
]]>
</JS_Script>
</Rule_Body>
<Dependencies />
</AXCP_Rule>
</Definition>
</Rule>
```

7.5 MyPackager.bat

```
@echo off
C:
cd "\Program Files\Sony Ericsson\DRM Packager"
DRMPackager -p "Profile\main.dpr" -q -x --file "Profile\tmpTemplate.xml" --xml --uri "" --iconuri ""
--name $fn --cid $id@dsi.unifi.it %1
rem pause
DRMPackager -f %2 -p "Profile\main.dpr" -q -rsep -g "Profile\tmpTemplate.xml" -d %3 -y %4
```

8 Work Done on AXMEDIS mobile player (DSI)

See DE4-1-1-3-content-modeling-and-managing for the activity performed and DE3-1-2-3-4-spec-of-AXMEDIS-Editor-and-Viewers for the specification.

References

[1] AXMEDIS-DE3-1-2-2-6-Spec-of-AX-Content-Processing-upC-v1-5).

[2] W.-T. Balke, A. Badii. Assessing Web Services Quality for Call-by-Call Outsourcing. In Proc. of the Int Workshop on Web Services Quality (WQW'03), Rome, Italy, 2003.

[3] MPEG 21, http://www.chiariglione.org/mpeg/standards/mpeg-21/mpeg-21.htm

[4] ISO/IEC, ISO/IEC 2nd Edition TR 21000-2 – Vision, Technologies and Strategy.

[5] ISO/IEC, ISO/IEC 2nd Edition FCD 21000-2 – Digital Item Declaration.

[6] ISO/IEC, ISO/IEC FDIS 21000-3 – Digital Item Identification.

[7] ISO/IEC, ISO/IEC CD 21000-4 – Intellectual Property Management and Protection.

[8] ISO/IEC, ISO/IEC IS 21000-5 – Rights Expression Language.

[9] ISO/IEC, ISO/IEC IS 21000-6 – Rights Data Dictionary.

[10] ISO/IEC, ISO/IEC IS 21000-7 – Digital Item Adaptation.

[11] ISO/IEC, ISO/IEC FDIS 21000-8 – MPEG-21 Reference Software.

[12] ISO/IEC, ISO/IEC IS 21000-9 – File Format.

[13] ISO/IEC, ISO/IEC FCD 21000-10 – Digital Item Processing.

[14] ISO/IEC, ISO/IEC TR 21000-11– Evaluation Methods for Persistent Association Technologies.

[15] ISO/IEC, ISO/IEC TR 21000-12 – Test Bed for MPEG-21 Resource Delivery.

[16] ISO/IEC, ISO/IEC CD 21000-14 – Conformance Testing.

[17] ISO/IEC, ISO/IEC WD 21000-15 – Event Reporting.

[18] ISO/IEC, ISO/IEC FDIS 21000-16 – Binary Format.

[19] ISO/IEC, ISO/IEC WD 21000-17 – Fragment Identification of MPEG Resources.

[20] ISO/IEC, ISO/IEC IS 21000-5 – Rights Expression Language.

[21] XrML, http:// www.xrml.org/.

[22] Open Digital Rights Language (ODRL). http://odrl.net.

[23] OMA DRM Rights Expression Language, OMA-Download-DRMREL-V2_0-20041210-C. 10 December 2004.

[24] XML Encryption Syntax and Processing, W3C Candidate Recommendation 10 December 2002, http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/

[25] XML Signature Syntax and Processing, W3C Recommendation 12 February 2002, http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/

[26] 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.

[27] 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.

[28] 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.

[29] 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.

[30] Open Mobile Alliance (OMA). DRM Rights Expression Language.http://www.openmobilealliance.org/release_programm/docs/DRM/V2_0-20060303-A/OMA-ERP-DRM-V2_0-20060303-A.zip

[31] http://www.imagemagick.org/.

[32] http://www.mega-nerd.com/libsndfile/

[33] http://sky.prohosting.com/oparviai/soundtouch/.

[34] http://www.mobile-phones-uk.org.uk/

[35] http://www.pdatoday.com/comments/2934_0_1_0_C/

[36] http://www.w3.org/Mobile/CCPP/

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

[38] http://arbor.ee.ntu.edu.tw/~jlhuang/publications/IMMCN03_ICC.pdf

[39] http://wurfl.sourceforge.net/

[40] http://sourceforge.net/projects/delicon/

[41] http://cocoon.apache.org/2.1/developing/deli.html

[42] http://www.w3.org/TR/NOTE-CCPP/

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

[44] http://www.jmcgowan.com/avi.html

[45] http://msdn.microsoft.com/library/sdkdoc/gdi/bitmaps_9c6r.htm

[46] D Mukherjee (2004) MPEG-21 DIA: Objectives and Concepts, HP Labs. Lecture slides as part of ECE 289J – Multimedia Networking at UC Davis taught by Prof. Mihaela van der Schaar

[47] MPEG-21 DIA: http://www.chiariglione.org/mpeg/tutorials/technologies/mp21-dia/index.htm >

[48] WY Lum, FCM Lau (2002) A Context-Aware Decision Engine for Content Adaptatio n, 2002 IEEE, University of Hong Kong

[49] Dietmar Jannach, Klaus Leopold, Christian. Timmerer, and Hermann Hellwagner. W3C Workshop on Frameworks for Semantics in Web Services Semantic Multimedia Adaptation Services for MPEG-21 Digital Item Adaptation

[50] Christian Timmerer, Hermann Hellwagner, "Interoperable Adaptive Multimedia Communication," IEEE MultiMedia, vol. 12, no. 1, pp. 74-79, January/March, 2005

[51] B. Pellan, C. Concolato, "Media-Driven Dynamic Scene Adaptation," wiamis, p. 67, Eight International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS '07), 2007

[52] TIMMERER, C., HELLWAGNER, H. 2005. Interoperable Adaptive Multimedia Communication. IEEE MultiMedia. January - March 2005, pp 74-79

[53] M. Ransburg et al. Dynamic and Distributed Adaptation of Scalable Multimedia Content in a Context-Aware Environment Proc. of the European Symposium on Mobile Media Delivery (EuMob 2006), Alghero, Italy, September 2006 [ITEC/TR]

[54] Ming-Wen Tong, Zong-Kai Yang, Qing-Tang Liu, Xiao-Ning Liu, A Novel Content Adaptation Model under E-learning Environment, 36th ASEE/IEEE Frontiers in Education Conference October 28 – 31, 2006, San Diego, CA

[55] Girma Berhe, Lionel Brunie, Jean-Marc Pierson, Modeling Service-Based Multimedia Content Adaptation in Pervasive Computing CF'04, April 14–16, 2004, Ischia, Italy.

[56] Debargha Mukherjee, Eric Delfosse, Jae-Gon Kim,Yong Wang, Optimal Adaptation Decision-Taking for Terminal and Network Quality-of-Service IEEE Transactions On Multimedia, Vol. 7, No. 3, June 2005

[57] Hongjiang Zhang, Adaptive Content Delivery: A New Application Area Formedia Computing Research, China, 2006

9 Appendices

9.1 AXMEDIS DIA Decision Engine User Guide

Structure
There are four folders as follows:

- smilres
- smiloutput
- smiltemplate
- smilcontentgroup

smilres
- contains resources to be put in the original SMIL AXMEDIS object (800 x 600)

smiloutput
- contains the two scripts CreateSMIL and AdaptSMIL, CreateSMIL creates a 800 x 600 SMIL object and AdaptSMIL loads this object, adapts its resources for Samsung i320 and uses the specified SMIL templates to format the adapted resources in a 320 x 240 SMIL object
- contains the output (original object and adapted object) of the two scripts

smiltemplate
- contains SMIL templates used by the scripts to create the original and the adapted SMIL objects

smilcontentgroup
- contains content group template files used by the scripts to create the original and the adapted SMIL objects

Functionality

Unzip SMIL-work-v1.zip file in E:\ drive (else, change the globals in the scripts to the drive/path of your choice). This will create the above mentioned 4 folders. As mentioned, if the file is extracted in a different path, the global variables of the scripts will need to be updated.

Currently, the adaptation script makes use of a specified profile path which is assumed to have been passed by the C++ layer. This is due to the fact that the AXCP Rule Editor needs to be updated with the latest C++ library to have access to the Activate() function of the AxDecisionTakingEngine class which essentially computes the best possible match for the device in hand, from a set of available template profiles.

The up-to-date code and library will be uploaded on the SVN repository.

The available templates list:

Devices for which profile templates exist:
1. Nokia
    a. 1110i
    b. 3110
    c. N95
    d. N70
2. HTC
    a. S310 Mobile Phone

   b. P3300 Mobile Phone
   c. TyTN Mobile Smartphone
   d. Advantage X7500
3. Samsung
   a. C300
   b. D900
   c. i320
   d. F500
4. iMate
   a. JASJAR
   b. JAM
   c. JAMIN
5. Motorola
   a. V3i
   b. W375
   c. Z3
   d. K1

Please note the Information included in these template profiles:
Device name, Image Capability, Color Capability and Screen Size

Adaptation of a SMIL object including images and audio resources
The SMIL object – Alessandro Allori, has been used for experimentation with formatting and presentation of adapted resources. The result (diadtatest-2.axm) can be found under smiloutput folder.

Input:
- AXOBJECT (Alessandro Allori 800x600 SMIL),
- Device profile (Samsung i320 – information included: device name, screen size, image capability, color capability),
- SMIL template (modified manually)

Output:
- AXOBJECT (Alessandro Allori 320x240 SMIL)

As Samsung i320 exists in the available templates (see list above), the template is returned to the AXCP which loads it, as well as the specified SMIL template, and adapts the resources one by one, eventually saving them in a new object formatted according to the SMIL template.

This template is modified manually before hand for appropriate formatting and placement of adapted resources on the target device screen.

The C++ class AxDecisionTakingEngine (associated JS wrapper: JSAxDecisionTakingEngine) contains a function Activate(profilepath) which takes as argument the path of the device profile being used. This profile is then used for decision making to arrive at the conclusion which available template matches the best.

- If there is an exact match e.g. Samsung i320 is the target device, the profile is loaded by the AxDecisionTakingEngine, if a template exists in the set of available templates, a link to that template is sent to the AXCP script for subsequent adaptation via calls to various AXMEDIS adaptation plugins.

- If an exact match is not available, profile templates by same manufacturer are searched within the available set of templates, and the least capability one is sent to the AXCP.

- If no profile template for device by same manufacturer is available, the lowest capability device profile template from the available template set is given as output, which currently is Nokia 1110i with no color/image capability.

Things that need to be done:

- Update AXMEDIS Content Processing Rule Editor with the new library so that the AxDecisionTakingEngine class function Activate() is available through AXCP.
- Further population of template profiles and inclusion of those elements in the process of decision taking
- QoS parameters for selection or rejection of adapted resource

Issues:

- The text size and placement in the adapted DIA image output need adjustment
- It will be helpful for the client side to be able to provide some QoS metrics data and/or Accessibility Guidelines that could be used in selection/rejection or refinement of the adapted resource.

9.2 Accessibility Guidelines

Guidelines about appropriate conversion to ensure high accessibility in cases such as user impairments (special needs) must be available as a consensus base for optimal accessibility assurance; however such specific guidelines do not as yet exist. Whilst it is clear that for someone who is colour blind we can have a media adaptation decision rule that essentially states that:

If colour vision deficiency exists, then for delivery to mobile, for example, reduce the resource (image, video) to basic visible colour values (e.g. black & white)

There are only a few cases where the decision rule is fairly well indicated as in the above case; however in many other cases arising, for example, from various degrees and combinations of perceptual impairment such as differential visual impairment in each eye; there exists no established rules about what would constitute the best decision rule.

Such difficulty is universally acknowledged and, at least as far as item number 3 above is concerned, the subject of a relevant standard which has emerged as the ISO IEC JTC1 24751-1 for Individualised Adaptability and Accessibility in E-Learning, education and Training (publication expected Christmas 2007). The Scope of the multipart standard is to provide a common framework to facilitate matching of user accessibility needs and preferences with appropriate learning resources and user interfaces. The common framework includes two complimentary sets of information:

- The description of a user's accessibility needs and preferences including:
    - how resources are to be displayed and structured
    - how resources are to be controlled and operated, and
    - what supplementary or alternative resources are to be supplied.
- The description of the characteristics of the resources that affect how they can be perceived, understood or interacted with by a user, including:
    - what sensory modalities are used in the resource,
    - the ways in which the resource is adaptable (i.e. whether text can be transformed automatically)
    - the methods of input the resource accepts, and
    - the available alternatives.

This standard is expected to provide a way to describe and specify user needs and preferences on the one hand and the corresponding description of the resources on the other hand so the appropriate user interface, tools and resources can be matched with individual preferences and needs.

Parts 2 and 3 provide Metadata definitions for description of the Personal Needs and Preferences and Digital Resource descriptions respectively for the delivery of resources in an accessible, computer-mediated environment.

It may seem unusual to base the user referencing and resource description models on something which is primarily designed for users with disability but in practice it is often the case that Accessibility requirements drive the solution of a more general need because they are more critical - the absence of some feature could for example prevent any access at all. It is often the case that features provided for accessibility benefit everyone and at some point in our lives we all will suffer some degree of perceptual impairment as we get older.

Thus the move to establish the above standard is another example of the universal acknowledgement of the fact that:

Since we cannot ask for what we do not know exists there is often repressed or hidden demand and content/system designers who simply do not know who requires which features - unless we enable them to ask.

This is the approach the above Standard takes and it is an essential way to work towards the resolution of the difficulties regarding development of a Media Adaptation Decision Engine that has been enabled to incorporate all the relevant and encodable adaptation decision rules. The Standard can be seen as a means by which a user is put in touch with a user interface relevant to their needs and requirements.

In this multi-part standard, it is recognised that users experience a disability when there is a mismatch between the user's needs (or preferences) and the experience delivered by the media content. Idiosyncratic user perceptual impairments and preferences are thus not viewed as a consequence of the situated relationship between a user and the delivery context.

A person who is blind is not disabled when the lesson is delivered in audio, but an old person listening to a monologue is disabled only in a noisy environment or under the so-called cocktail-party-effect whereby only the presence of other people talking in the background render the otherwise perfectly audible delivery inaudible for the old person who due to degradation of their hearing capability may be unable to resolve the frequencies in certain regions as keenly as younger users could (environmentally –triggered exceptions).

Accessibility is thus determined by the flexibility of the delivery environment (with respect to presentation, control methods, structure, access mode, and user supports for example) and the availability of adequate alternative-but-equivalent content and activities.

Related Standards

Using the AccessForAll Individualised Adaptability approach means integrating it with other specifications and standards. Examples of relevant standards work might include (but not be limited to)

- W3C Web Accessibility Initiative Web Content Accessibility Guidelines (2.0) (WCAG)
    - which provides universal design guidelines and technical checkpoints that can be used to deal with some accessibility aspects of content objectively and generate Metadata. Currently in final draft for version 2.0
- Web Accessibility Initiative User Agent Accessibility Guidelines (UAAG)
    - Essential to implement with any customer/user-facing tools (such as browsers)
- W3C Composite Capability Preference Profiles (CC-PP - device profiles)
    - One standard for modelling device characteristics (which are part of part of any delivery context).

There are other relevant standards which relate to:

- Aggregation formats such as RSS/Atom[1], MPEG 21[2] and particularly eLearning-community-specific formats such as IMS Content Packaging[3]
    - Content delivery involves at least virtual aggregation (if not actual). Note that IMS Content Packaging, a widely used format in eLearning, is currently undergoing revision and accessibility features are being incorporated (in version 1.2).
- Metadata standards such as IEEE Learning Objects Metadata Standard (LOM)[4], Dublin Core[5] Metadata Initiative
    - Both of these are widely used to describe formal Metadata vocabularies for Learning Objects
- Relevant Ontologies, such as IEEE Resource Aggregation Model for Learning, Education and Training (RAMlet)[6]
    - This work aims to produce an ontology enabling automation (possibly semi-automation) of translation from one aggregation format to another for eLearning aggregation formats

- IMS Learning Design[7]
  - An educational "glue" standard that permits description of learning activities in a structured machine-executable plan. This has a fast-growing community of use in Higher and Further Education research and practice groups.
- IMS ePortfolio[8]
  - A standard for moving portfolios from one system to another.

References

A full report on relevant standards would include many more, including material on Semantic Web technologies, web service technologies, such as SOAP, WSDL, REST etc., AJAX etc; the list below represent some of the most relevant sources.

1. http://www.rss-specifications.com/rss-specifications.htm
2. http://www.chiariglione.org/mpeg/standards/mpeg-21/mpeg-21.htm
3. http://www.imsglobal.org/content/packaging/
4. http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf
5. http://dublincore.org/
6. http://www.ieeeltsc.org/working-groups/wg11CMI/ramlet
7. http://www.imsglobal.org/learningdesign/
8. http://www.imsglobal.org/ep/index.html
9. http://www.imsglobal.org/accessibility

Latest drafts of the standard can be found at:

10. http://jtc1sc36.org/doc/36N1139.pdf
11. http://jtc1sc36.org/doc/36N1140.pdf
12. http://jtc1sc36.org/doc/36N1141.pd f