AXMEDIS
Automated Production of Cross Media Content for Multichannel Distribution

AXMEDIS Editor User Manual
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-fulfillment of any of his obligations in respect of this License.

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

COPYRIGHT

Copyright © 2008 by AXMEDIS
For information on this book, contact:

Prof. Paolo Nesi
Via S. Marta 3
50136 Firenze (Italy)
Tel: +39 055 4796523
Fax: +39 055 4796469
http://www.axmedis.com
To report errors, please send a note to mitolo@dsi.unifi.it
# Table of Content

AXMEDIS COPYRIGHT NOTICE ................................................................................................................................... 2

COPYRIGHT .......................................................................................................................................................... 3

INTRODUCTION .................................................................................................................................................... 15

OVERVIEW .......................................................................................................................................................... 15

AXMEDIS EDITOR .................................................................................................................................................. 16

MAIN FUNCTIONALITIES .................................................................................................................................... 16

RELATIONSHIP WITH OTHER TOOLS ........................................................................................................... 16

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS ................................................... 16

REGISTER THE USER AND CERTIFICATE THE TOOL ................................................................................... 16

COMMAND LINE OPTIONS .................................................................................................................................. 20

EDIT CONFIGURATION ....................................................................................................................................... 20

CREATE A NEW AXMEDIS OBJECT .................................................................................................................. 22

THE ACTIVITIES WINDOW ................................................................................................................................ 25

THE BASIC STEPS WINDOW ............................................................................................................................. 26

ADDING OBJECT REFERENCE ......................................................................................................................... 27

ADDING EMBEDDED OBJECT ............................................................................................................................ 29

ADDING DUBLIN CORE METADATA .............................................................................................................. 31

OPEN AN EXISTING OBJECT ............................................................................................................................. 34

EXTRACT AN EMBEDDED RESOURCE ............................................................................................................ 34

MODIFYING AN AXMEDIS OBJECT STORED ON DATABASE ................................................................................. 35

THE QUERY DIALOG ......................................................................................................................................... 36

THE RESOURCE PROPERTY DIALOG ............................................................................................................. 37

THE CONTENT PROCESSING PLUG-INS ............................................................................................................. 39

THE AXMEDIS EDITOR MENUS .......................................................................................................................... 40

The File menu .................................................................................................................................................... 40

Edit menu .......................................................................................................................................................... 41

The Resource Viewer menu ............................................................................................................................... 41

AXMEDIS RESOURCE VIEWER ......................................................................................................................... 42
AXMEDIS Editor User Manual

Semantics .................................................................................................................................................. 127
Example use ........................................................................................................................................... 128
wait ......................................................................................................................................................... 128
Interface ............................................................................................................................................... 128
Semantics ............................................................................................................................................... 128
Example usage ....................................................................................................................................... 128
LOCAL OBJECTS, DIBOs AND CONSTANTS ......................................................................................... 129
DIPError .................................................................................................................................................. 129
ObjectMap ............................................................................................................................................. 130
PlayStatus ............................................................................................................................................. 132
TABLE OF AVAILABLE MPEG21 DIP FUNCTIONALITIES .................................................................... 133

AXMEDIS DRM EDITOR ......................................................................................................................... 135
Main functionalities .................................................................................................................................. 135
Relationship with other tools .................................................................................................................. 135
Detailed description of the functionalities and screenshots ...................................................................... 135

AXMEDIS WORKFLOW EDITOR ............................................................................................................... 139
Main functionalities .................................................................................................................................. 139
Relationship with other tools .................................................................................................................. 139
Detailed description of the functionalities and screenshots ...................................................................... 139

AXMEDIS PROTECTION INFORMATION EDITOR ................................................................................. 141
Main functionalities .................................................................................................................................. 141
Relationship with other tools .................................................................................................................. 141
Detailed description of the functionalities and screenshots ...................................................................... 141
Viewing of Protection Information .......................................................................................................... 141
Editing of Protection Information ............................................................................................................ 142

AXMEDIS PLUG-INS ............................................................................................................................... 143

AUDIO ADAPTATION PLUG-IN .............................................................................................................. 143
Main functionalities .................................................................................................................................. 143
Relationship with other tools .................................................................................................................. 143
Detailed description of the functionalities and screenshots ...................................................................... 143
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relationship with Other Tools</td>
<td>165</td>
</tr>
<tr>
<td>Detailed Description of the Functionalities and Screenshots</td>
<td>165</td>
</tr>
<tr>
<td>Cryptlib Plug-in</td>
<td>170</td>
</tr>
<tr>
<td>Main Functionalities</td>
<td>170</td>
</tr>
<tr>
<td>Relationship with Other Tools</td>
<td>170</td>
</tr>
<tr>
<td>Detailed Description of the Functionalities and Screenshots</td>
<td>170</td>
</tr>
<tr>
<td>Language Guesser Plug-in</td>
<td>171</td>
</tr>
<tr>
<td>Main Functionalities</td>
<td>171</td>
</tr>
<tr>
<td>Relationship with Other Tools</td>
<td>171</td>
</tr>
<tr>
<td>Detailed Description of the Functionalities and Screenshots</td>
<td>171</td>
</tr>
<tr>
<td>Plagiarism Detection Plug-in</td>
<td>172</td>
</tr>
<tr>
<td>Main Functionalities</td>
<td>172</td>
</tr>
<tr>
<td>Relationship with Other Tools</td>
<td>172</td>
</tr>
<tr>
<td>Detailed Description of the Functionalities and Screenshots</td>
<td>172</td>
</tr>
<tr>
<td>Document Descriptor Extractor Plug-in</td>
<td>174</td>
</tr>
<tr>
<td>Main Functionalities</td>
<td>174</td>
</tr>
<tr>
<td>Relationship with Other Tools</td>
<td>174</td>
</tr>
<tr>
<td>Detailed Description of the Functionalities and Screenshots</td>
<td>174</td>
</tr>
<tr>
<td>Document Adaptation Plug-in</td>
<td>176</td>
</tr>
<tr>
<td>Main Functionalities</td>
<td>176</td>
</tr>
<tr>
<td>Relationship with Other Tools</td>
<td>176</td>
</tr>
<tr>
<td>Detailed Description of the Functionalities and Screenshots</td>
<td>176</td>
</tr>
<tr>
<td>Audio FP Plug-in</td>
<td>178</td>
</tr>
<tr>
<td>Main Functionalities</td>
<td>178</td>
</tr>
<tr>
<td>Relationship with Other Tools</td>
<td>178</td>
</tr>
<tr>
<td>Detailed Description of the Functionalities and Screenshots</td>
<td>178</td>
</tr>
<tr>
<td>Fingerprint Extraction</td>
<td>178</td>
</tr>
<tr>
<td>Fingerprint Matching</td>
<td>180</td>
</tr>
<tr>
<td>Mzany - Audio FP Plug-in</td>
<td>181</td>
</tr>
</tbody>
</table>
Detailed Description of the Functionalities and Screenshots .............................................................. 201

Workflow Rule Editor Plug-in .................................................................................................................. 204

Main Functionalities ................................................................................................................................. 204
Relationship with Other Tools .................................................................................................................. 204
Detailed Description of the Functionalities and Screenshots .............................................................. 204

Workflow Engine Plug-in .......................................................................................................................... 208

Main Functionalities ................................................................................................................................. 208
Relationship with Other Tools .................................................................................................................. 209
Detailed Description of the Functionalities and Screenshots .............................................................. 209

Ringtone Adaptation Plug-in .................................................................................................................... 210

Main Functionalities ................................................................................................................................. 210
Relationship with Other Tools .................................................................................................................. 210
Detailed Description of the Functionalities and Screenshots .............................................................. 210
Convert Function ...................................................................................................................................... 210
Convert_to-MP3 Function .......................................................................................................................... 212
Convert_to_WAV Function ........................................................................................................................ 212
Resample Function ................................................................................................................................... 213
Convert_And_Resample Function ............................................................................................................. 214
getInfo Function ...................................................................................................................................... 215
Clip Function .............................................................................................................................................. 217

Image Processing Plug-in .......................................................................................................................... 218

Main Functionalities ................................................................................................................................. 218
Relationship with Other Tools .................................................................................................................. 218
Detailed Description of the Functionalities and Screenshots .............................................................. 219
Conversion ................................................................................................................................................. 220
Import ....................................................................................................................................................... 220
Resize ......................................................................................................................................................... 221
Contrast ..................................................................................................................................................... 222
Edge ......................................................................................................................................................... 222
AXMEDIS Editor User Manual

Emboss ............................................................................................................................ 223
Blur ................................................................................................................................. 224
GaussianBlur ................................................................................................................ 224
Median ............................................................................................................................. 225
Mirror ............................................................................................................................. 226
Noise ............................................................................................................................... 226
Despeckle ....................................................................................................................... 227
Equalize ........................................................................................................................ 227
Enhance ........................................................................................................................ 228
ExtractChannel .......................................................................................................... 228
Grayscale ...................................................................................................................... 229
Magnify ........................................................................................................................ 229
Minify ........................................................................................................................... 230
Modulate ....................................................................................................................... 230
Monochrome ............................................................................................................... 231
Negate .......................................................................................................................... 232
Normalize ..................................................................................................................... 232
OilPaint .......................................................................................................................... 233
Quality .......................................................................................................................... 233
Quantize ......................................................................................................................... 234
Raise .............................................................................................................................. 235
ReduceNoise ............................................................................................................... 236
Replace ........................................................................................................................ 236
FloodFill ....................................................................................................................... 237
Roll ................................................................................................................................. 238
Rotate ............................................................................................................................ 239
Scale ............................................................................................................................... 240
Shear ............................................................................................................................. 240
Shade ............................................................................................................................. 241
Spread ................................................................. 242
SetOpacity ............................................................. 243
SubImage .............................................................. 243
GetInfo ................................................................. 244
SetMaskColour ...................................................... 245
Paste ................................................................. 246
Test ................................................................. 247

REFERENCES AND LINKS ............................................. 248
AXMEDIS TUTORIALS ............................................. 248
AXMEDIS TOOLS FOR FREE DOWNLOAD .................. 248
AXMEDIS TECHNICAL NOTES .................................. 249
AXMEDIS SOLUTIONS ............................................. 249
AXMEDIS SHOWCASES ............................................ 249
AXMEDIS FRAMEWORK SPECIFICATION .................. 249
AXMEDIS FRAMEWORK DEMONSTRATORS, CASES, TRIALS, FOR DISTRIBUTION etc. ....................... 251
BROCHURES AND PRESS CUTTING (A PART) .................. 251
OTHER REFERENCES ................................................... 251
**INTRODUCTION**

Dear AXMEDIS user,

Welcome to *AXMEDIS Editor User Manual*, the official training workbook for the program that has as main aim to create rich and interactive content.

With this guide, the AXMEDIS team is hoping to help users to create easily AMEDIS objects to show how simple and powerful this tool is.

AXMEDIS is a very complex and complete world and it is very difficult to cover all the aspects in only one manual. Remember to not hesitate to contact the AXMEDIS team if you need additional information or to point out mistakes in this manual. Also don’t hesitate to visit the AXMEDIS portal where you can find many other information regarding the AXMEDIS technology, with many real examples and objects.

**OVERVIEW**

This deliverable aim is to describe the User Manuals of AXMEDIS Editor tool.

For a general overview of the other AXMEDIS tools and for their download please access to: [http://www.axmedis.org/com/index.php?option=com_content&task=view&id=83&Itemid=55](http://www.axmedis.org/)
AXMEDIS Editor User Manual

AXMEDIS EDITOR

MAIN FUNCTIONALITIES
The AXMEDIS Editor is used to create AXMEDIS Objects embedding "raw" digital resources or other AXMEDIS Objects. The AXMEDIS Editor allows to:

• create a new AXMEDIS Object
• add resources (images, videos, documents, etc.)
• add AXMEDIS Objects from file or from database
• manipulate the AXMEDIS Object structure (remove any element, move the elements)
• save to file
• upload to database
• load an AXMEDIS Object from file
• load an AXMEDIS Object from database
• view the resources with the internal viewers
• manipulate the resources using the Content Processing Plug-ins
• associate metadata to the AXMEDIS Object (using the Metadata Editor),
• define the Potential Available Rights for the Object (using the internal DRM Editor)
• create a SMIL presentation for the resources inside the Object (using the Visual Editor)
• create rules to define the object behaviour (using the Behaviour Editor)
• define the protection information for the object (using the Protection Editor)
• see the status of the object in the Workflow process (using the Workflow Viewer)
• be launched from the workflow to do a specific job and when the user finishes it can notify the activity completion and thus to proceed to the next step in the production process.

RELATIONSHIP WITH OTHER TOOLS
The AXMEDIS Editor:

• uses the AXMEDIS Database to search/retrieve for content and to store content
• can be launched form the Workflow engine and interact with it
• uses the content processing plug-ins to manipulate the digital resource

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS
The following sections aim to describe the use of the AXMEDIS Editor to create AXMEDIS objects embedding “raw” digital resource or other AXMEDIS objects.

In the next sections, it is shown how to register and certify a tool, as well as a short tutorial to create a new AXMEDIS object.

Subsequently the next sections will show all the features available in the Editor more in detail. The menus and the integration of the AXMEDIS Editor with a database will also be explained.

REGISTER THE USER AND CERTIFICATE THE TOOL
When the AXMEDIS Editor is opened for the first time, a window asking to register the new user will be showed (see the figure below).
By clicking the “Yes” button the certification procedure will start and a web page will be opened asking mandatory information to be filled in for the correct user registration. By clicking the “OK” button this information will be sent and collected by the AXCS and a confirmation email will be sent to the user at the email address specified in the field “Email”.

The email received contains an URL to be clicked for the confirmation of the procedure: this is necessary to control the correctness of the specified email address. After the confirmation, another email will be received containing the user certificate (a .p12 file) and a password necessary to import the certificate.

The User Certificate has to be saved on the hard disk before starting the importation procedure.
To import the certificate select in the AXMEDIS Editor the menu Help/Import User Certificate...
Select the .p12 file received and press the **Open** button.

Insert the password received by email to continue the registration procedure and press the **OK** button.
A message confirming the correctness of the registration procedure will be showed. Press the OK button to continue.

After the user registration it is necessary to certificate the tool. Open the menu Help | Tool certification…

Immediately a message will be showed confirming the correctness of the tool certification.
After this the user will be able to create protected AXMEDIS objects from scratch using the AXMEDIS Editor.

**COMMAND LINE OPTIONS**

It is possible to launch the axmedis editor by using the Command Pront. In this case it is sufficient to specify the installation folder of the AXMEDIS Editor and the file name to be opened.

The following screenshot shows an example of command to open an object called `accademia.axm`.

![Command Prompt Screenshot](image)

**EDIT CONFIGURATION**

The configuration menu allows editing the configuration parameters of the AXMEDIS Editor. By clicking in the File/Configuration… menu a new window will be showed containing some parameters related to different databases.

![Configuration Window](image)

The following figure shows the configuration window containing the following fields to be configured:

- the **Database** server name
- the **PMS** server name
- the **Workflow** server name
- the **AXCS** server name, username and password
- The **Proxy** with the host name, username and password.
By clicking the Advanced button a more complete window will be opened containing many different parameters to be configured.

The most significant modules and relative parameters to be configured are the following:

- **AXMEDIS_PLUGIN_MANAGER**
  - PLUGINS_PATH, the folder containing the AXMEDIS plug-ins are

- **DATABASE**
  - User, the username to access in the database
  - Passwd, the relative password
  - LoaderWSEndPoint, web service end point to load AXMEDIS objects
  - ServerWSEndPoint, web service end point to save AXMEDIS objects
  - FTPPath, ftp site where to upload the AXMEDIS objects when saving (e.g. ftp://axdbf.AXMEDIS.org)
  - LockWSEndPoint, web service end point to lock/unlock AXMEDIS objects

- **AXMEDIS_SELECTION**
  - MAIN_QUERY_SUPPORT_WSDL, the WSDL to query the database

- **OBJECT_CREATOR**
  - AXCID, the ID of the object creator. If empty the AXUID received during user registration is used
  - Name, the name of the object creator
  - URL, the URL of the object creator
  - Company, the company of the object creator
  - CompanyURL, the URL of the company of the object creator
  - Nationality, the nationality of the object creator

- **RESOURCE_PARAM**
  - IMAGE_TIMER (s), the time duration used in the slide show presentation
  - COLOUR, the background colour used in the resource viewer

- **RESOURCE_EXTENSIONS**
  - <file extension>, the mimetype associated with the file extension

- **PMSCLIENT**
  - PMSClientEndpoint, the end point used when connecting to PMS server
• CERTINFO
  o CERTPATH, the path where the certificate is stored on the hard disk
  o USERPWD, the user password received after the registration and used to read the user certificate
• AXCSOBJREG
  o AXCSObjRegEndpoint, the end point used for AXMEDIS object ID request and registration
  o AXCSObjRegUser, the user name used to request the AXMEDIS object ID
  o AXCSObjRegPsw the password
• WORKFLOW
  o workflowUrl, the URL of the workflow server
  o gatewayUrl, the URL of the workflow gateway used to request workflow information
• TOOL_RECIPIENT_PUBLISH
  o information related to the pending workflow operations
• METADATA_EDITOR_VIEWER
  o MetadataView_CSS, the CSS for the metadata viewer
  o SCHEMA_PATH, the path where to find the schema files
  o <xml namespace>, schema file associated with the namespace
• SMIL_PARAM
  o HEIGHT, WIDTH, COLOR, are the default values for the root layout element in the visual editor
  o ZOOM, default zoom factor for the behavioural view in the visual editor

Below the left and right part of the Configuration windows, a number of buttons are present.
By clicking the Add button in the left part of the window it is possible to add a new configuration module, as showed in the figure below.

The Remove button permits to delete a module.
The right part of the window contains three buttons.
To add a parameter to a selected module press the Add button. As showed in the figure below, to add a parameter it is necessary to specify the following three values:

  • the parameter name
  • the parameter type (double, int32 or string)
  • the parameter value

To edit a parameter select the parameter in the right window and press the Edit button.
To remove a parameter select the parameter in the right window and press the Remove button.

CREATE A NEW AXMEDIS OBJECT
To create a new AXMEDIS object select File/New from the Menu or use the button on the toolbar
The AXMEDIS Editor will appear divided in three main parts:

- the Tree view window on the left, that provides a view of the AXMEDIS object structure; selecting the MPEG-21 panel it is possible to see the hierarchical view of the MPEG-21 structure of the object;
- the Panels View window in the middle, which shows the selected panel; below these panels a combo box is present showing all the resources available in the object and facilitating the applications of all the different features to them;
- the Activities windows on the right, that helps to find easily the main features of the Editor and to guide the user in the basic steps to create a new objects. If necessary this windows can be closed by clicking on the \(\times\) button in the high-right corner to enlarge the Resource View window.

In the Tree window initially are present only three items:

- the AXOID, that is the unique AXMEDIS Object Identification number
- the AXMEDIS Info item, double clicking on it will show information about the object creator, the creation date, etc.
- the Dublin Core item, containing the metadata information in Dublin Core standard format

The first thing to do to create a new object is to add a number of digital resources initially stored in the hard disk as single files.

To add a resource from a file on the local hard disk, select the \(\text{\textcolor{blue}{+}}\) button from the toolbar and select the file to be added:
It is possible to add a resource also using the drag-and-drop feature, dragging the resource in the tree window. The AXMEDIS Editor is capable to automatically recognize all the most important file formats for documents (txt, doc, pdf, rtf, etc.), images (gif, tiff, png, jpg, etc.), audio (wma, mp3, wav, etc.), video (mpeg, mpg, etc.) and to play them using the internal player.

Finally it is possible to add a resource using the contextual menu with a right click on the tree window and selecting Add/Embedded Resource...

When a resource is correctly embedded, it appears in the tree window with an icon identifying the type of the resource followed by the prefix Resource and the resource name closed by square brackets. Double clicking on the resource opens the ImageViewer:
The Activities window permits a simple access to basic features using intuitive shortcuts.

If not visible, it is possible to open the Activities window by clicking in the Help/Show activities menu.

The available features in the Activities window are the following:
- Open AXMEDIS Object from file
- Open AXMEDIS Object from database
- Create AXMEDIS Object from resource files
- Create AXMEDIS Object from query on database

Clicking on **Open AXMEDIS Object from file** simply opens a window to select and load an AXMEDIS object from the disk.

Clicking on **Open AXMEDIS Object from database** the Query dialog is open. See section “The Query Dialog” for more details about the Query dialog.

Clicking on **Create Object from resource files** opens a new dialog to help a user to create a new AXMEDIS object from scratch.
The **Resources** box permits to add external resource selecting them from the disk. Pressing the + button a new resource is added, otherwise pressing the – button the resource is deleted.

The **Dublin Core metadata** box permits to add metadata of the object before his creation as AXMEDIS object. Ticking the **Upload into database** option the object is automatically uploaded into the database. Pressing OK the new object is created and, eventually, uploaded into the database.

Clicking **Create AXMEDIS Object from query on database** opens a new dialog to help a user to create a new AXMEDIS object from scratch using existing objects stored in the database.

The functioning is very similar to the previous. The only difference is in the first box now named **AXMEDIS Objects** that permits to add an object querying it from the database. Pressing the **Query** button the Query dialog is open. See section "The Query Dialog" for more details about the Query dialog.

**THE BASIC STEPS WINDOW**

Below the **Activities** window a number of numbered buttons are showed to help users to create a new object showing the necessary steps to be followed, the **Basic Steps** window.
The first button **Request final AXOID** have the aim to request the final ID of the AXMEDIS object. This procedure can be done in any moment but in any case before the step no. 7. **Register keys on AXCS**.

Press the button **2a. Add resources...** to add digital resources into the object or the button **2b. Add objects...** to add embedded AXMEDIS objects.

By pressing the button **3. Edit Dublin Core** the Metadata Editor will be showed (see section “AXMEDIS Metadata Editor” for additional information).

By pressing the button **4. Edit SMIL Presentation**, the Visual Editor will be opened (see section “AXMEDIS Visual and Behaviour Editor” for additional information).

By pressing the button **5. Edit DRM** the DRM editor will be opened (see section “AXMEDIS DRM Editor” for additional information).

By pressing the button **6. Edit Protection keys** the Protection Editor will be opened (see section “AXMEDIS Protection Information Editor” for additional information).

The button **7. Register keys on AXCS** is for the last operation to be performed before the content distribution and permits to send the protection information to the AXCS after the object protection.

The button **8. Create licenses** will open the DRM Editor to define the license.

**ADDING OBJECT REFERENCE**

Using this feature it is possible to add into an AXMEDIS object a reference to another object stored into the database. This feature is accessible in the toolbar pressing the button or **Edit/Add/Referred Object from DB** in the menu selecting.
Selecting this option, a query dialog is open. The dialog permits to execute a search into the database using a number of pre-defined fields. If you need more information about the Query dialog see section “The Query Dialog”. It is possible to search an object by using the following parameters:

- **Object creator**
- **Title**
- **Coverage**
- **Format**
- **Type**
- **Subject**
- **Description**
- **Creation Date**
- **AXInfo information** (such as Status, Distributor, Owner and Access Mode)

For example, to search into the database an AXMEDIS object containing a comedy film, it is necessary to write this parameter into the Type field, select into the Info Results box the information that will appear in the results (for example the creator and the title), and press the Submit button.

A new dialog will appear where it is necessary to insert the User Name and the password to access the database.

Pressing OK the query is sent to the database and the results are shown in a new window.

At this point it is possible to select the object in the list and press the OK button to continue. It is also possible to select more than only one object pressing the <SHIFT> or the <CTRL> keys and selecting all the needed objects.
After pressing OK, the selected objects are added in the Editor as reference objects (identified by the ref prefix).

In the Panels window, immediately below the name of each tab, a combo box is present allowing selecting the different resources available in the object.

**ADDING EMBEDDED OBJECT**

Using this feature it is possible to add into an AXMEDIS object another object stored into the database. This feature is accessible in the toolbar pressing the button or in the menu selecting **Edit/Add/Embedded Object from DB**. The procedure is very similar to what explained in the previous section.

Selecting this option, a query dialog is open. The dialog permits to execute a search into the database using a number of pre-defined fields. If you need more information about the Query dialog see section “The Query Dialog”.

It is possible to search an object using the following parameters:

- **Object creator**
- **Title**
- **Coverage**
- **Format**
- **Type**
- **Subject**
- **Description**
- **Creation Date**
- **AXInfo information** (such as Status, Distributor, Owner and Access Mode)

For example, to search into the database a comedy film, it is necessary to write this parameter into the **Type** field, select into the **Info Results** box the information that will appear in the results (for example the creator and the title), and press the Submit button.
A new dialog will appear where it is necessary to insert the User Name and the password to access into the database.

Pressing OK the query is sent to the database and the results are shown in a new window.

At this point it is possible to select the object in the list and press the OK button to continue. It is also possible to select more than only one object pressing the <SHIFT> or the <CTRL> keys and selecting all the needed objects. After the OK pressing, the selected objects are added in the Editor as embedded objects (identified by the Object prefix).
ADDING DUBLIN CORE METADATA

Double clicking on the Dublin Core item the metadata editor is opened:
To add new metadata elements, select them from the Add Sub-Element list box and then press the Add button.

To set the value of an element, e.g. the “title” element, select it from the metadata tree and enter the value in the Content’s Value textbox as highlighted below on the right.
The user can also edit the attribute(s) of an element by using the **Element's Attribute Grid** to change the Value of the attribute such as the language of the title element to English (lang, EN) as shown below:

Double clicking on the **AXMEDIS Info** the **Metadata Viewer** shows the AXMEDIS specific information associated with the object:
The Object Creator information is automatically added getting information from the configuration (File/Configuration… from the menu)

The object can be overwritten on the local hard disk using File/Save (or using the button on the toolbar) or saved as new file using File/Save As… from the menu.

Also the object can be uploaded on the AXMEDIS Database using the button on the toolbar or File/Upload into Database… from the menu.

Please note that the Metadata Editor also has its own “local toolbar” with six buttons: load, save, expand, expand all, collapse, and collapse all. The Load button is used to load metadata from a separate file XML button, and the Save button is used to save metadata into a separate XML file. This allows the user to use the current metadata as a template:

The AXMEDIS Editor permits to create very easily complex objects with both simple and nested resources. The following figure shows an example of complex object with a big number of digital resources.
OPEN AN EXISTING OBJECT

To open an existing AXMEDIS object select **File/Open...** from the Menu or use the 💾 button on the toolbar. The “Select a file” window appears where it is possible to select an AXMEDIS object to be opened.

EXTRACT AN EMBEDDED RESOURCE

The AXMEDIS Editor permits to extract a resource embedded in an object to be saved as external file. Select the resource in the tree window and right click on it to open the contextual menu. Select **Extract resource**, to save the file on the disk.
MODIFYING AN AXMEDIS OBJECT STORED ON DATABASE

To search for the AXMEDIS object on the Database select the button on the toolbar or using File/Open from Database… in the menu, the query dialog is opened and it is searched for an object with the title containing "portrait":

Pressing the Submit button, if the authentication procedure is successfully passed, the query is sent to the database and the results are shown:
Pressing OK the object is recovered from the database and opened:

THE QUERY DIALOG
The Query dialog is the window that permits to start a query into the database to search a specific AXMEDIS object using parameters defined by the user.

When the Query dialog is open, the user have to choose the sources where apply the query in the Available Sources box (1). The predefined available sources are AXEPTOOL, CMS, AXMEDIS DB and All of these. The search criteria supported by the Query dialog are organized into two main groups, (2) search by Dublin Core metadata information (with a specific sub-box for the search by creation date) and search by AxInfo information (3). Each group, along with its associated set of search criteria, is exposed on the dialog interface. Using this interface, a user enters a search string in the preferred field and chooses the correct criteria using comparison operators associated with that string (4).

The available criteria are:
GT, (>) the searched value is greater than the specified value
LT, (<) the searched value is less than the specified value
EQ, (=) the searched value is equal to the string specified by the user
GE, (>=) the searched value is greater or equal to the searched value
**LE**, (\leq) the searched value is less than or equal to the specified value  
**NE**, (\neq) the searched value is not equal to the specified value  
**STARTWITH**, the searched word start with the string specified by the user  
**ENDWITH**, the searched word end with the string specified by the user  
**CONTAINS**, the searched word contains the string specified by the user

The **Logic Operator Selector** box (5) specifies the and/or option to be applied for all the fields specified by the user.

Once all the search criteria have been specified, a user chooses the **Info Results** (6) to be showed in the **Query Result**. It is possible to select more that one Info selecting them pressing the <SHIFT> or the <CTRL> keys.

The information available for the results are divided in two groups, the **AXINFO** and the Dublin Core Metadata Information (prefix DCMI).

The **Reset** button deletes all the inserted information. The **Submit** button starts the query. After the pressing of the Submit button, it is necessary to insert the Username and the Password in the **Login on Query Support** dialog to authorize the query.

The results of the query are showed in the Query Result tab.

The information showed in the Query Result tab (1) are those selected in the Info Result box (2) followed by the UNR with the object ID (3) and the Source Channel (4).

In the previous figure the results table shows the AXMEDIS objects with Creator and Title fields as selected in the Info Results tab. The string "NULL" indicates that the info is not present in the object.

To load an object into the AXMEDIS Editor it is necessary to select it in the list and to press the **OK** button. It is also possible to select more than one object pressing the <SHIFT> or the <CTRL> keys and selecting all the needed objects. After the OK pressing again it is necessary to insert the Username and the Password to Login.

**THE RESOURCE PROPERTY DIALOG**

The Properties of a resource (and for any other element) can be edited right clicking on the element and selecting **Properties…** from the contextual menu:
And the following dialog is opened and the properties can be modified:

The **ContentId** is the name identifying the resource that appears in the Tree window closed into square brackets. As default the **ContentId** value is the same as the filename.

The **Reference** field contains (if present) the URL address of an external resource

The **MimeType** field identifies the file type and the format

The **Local path** specifies the resource file name
Finally the **Type** field specifies if a resource is embedded in the AXMEDIS object or if it is an external resource not physically present into the object. If a resource is not embedded in the AXMEDIS object, it is represented in the Tree windows with the prefix “ref::”. In the following figure, the [AXMEDIS] external resource is a link to an external web site.

### THE CONTENT PROCESSING PLUG-INS

On a Resource the contextual menu enables to use content processing plug-ins. Selecting the **Content processing plug-ins...** menu option the plug-ins function list is provided with the list of applicable plug-ins functions (based on resource mimetype):

Selecting a plug-in function and pressing the **Execute** button a dialog is presented allowing to provide the arguments for the function, clicking on the **Execute** button the plug-in function is executed:

In this example we will resize an image using the Resize function in the ImageProcessing plug-in. The result is the following:
THE AXMEDIS EDITOR MENUS

THE FILE MENU
The File menu contains a set of functions related to the edited (or to be loaded) object. They are:
- **New**, to create a new AXMEDIS object
- **Open**, to open an AXMEDIS object loading it from the disk
- **Open from database**, to open an object searching it from the database
- **Save**, to overwrite the object
- **Save as...**, to save an object specifying the filename
- **Save as MPEG21**, to save the object as *.mp21 file
- **Upload on Database**, to upload an object into the database
- **Close**, to close the current opened object
- **Notify Workflow activity completion...**, to be used when the AXMEDIS Editor has been launched from the Workflow Server to notify that the activity to be done has been completed;
- **Configuration**, shows a number of dialogs to set the correct configuration of the Editor
- **Plugins...**, shows a box with the list of available plug-ins
- **Recent files**, shows a list of the recent opened objects
- **Exit**, to close the AXMEDIS Editor
EDIT MENU

The Edit menu contains a set of functionalities available for the resources in the object tree.

When a resource is selected in the tree, using the Edit menu is possible to select one of the following features:
- **Open**, open or play the resource in the Resource View panel
- **Open with**, play or view the selected resource selecting one of the viewers listed in the menu
- **Edit Properties**, view and modify the properties of the resource
- **Insert**, opens a menu with:
  - **Embedded Resource…**, add an embedded resource in the object
  - **Referred Resource…**, add an external resource
  - **New Object**, insert a new empty object in the tree
  - **New Object with Resource…**, insert an object in the tree selecting it from the disk
  - **Embedded Object from file…**, insert and embedded object from file
  - **Referred Object from file…**, insert a reference to an object choosing it from as file in the disk
  - **Embedded Object from DB…**, insert an embedded object from the database
  - **Referred Object from DB…**, insert a reference to an object choosing it from the database
- **Content processing plugins…**, opens a new window with the list of available plug-ins for the selected resource.
- **Extract resource**, to save an embedded resource in the disk as file
- **Cut, Copy, Paste** and **Delete** options
- **Move up, Move Down** to change the resource position in the tree
- **Refresh**, to force the tree update

THE RESOURCE VIEWER MENU

The Resource View menu shows a set of functionalities available according with the resource showed in the Resource View window. The list of functionalities is the same showed with a right click in the Resource View window.
AXMEDIS RESOURCE VIEWER

MAIN FUNCTIONALITIES

The Resource Viewer allows viewing and in some cases also editing the resources in the AXMEDIS Object, it is composed of:

- Image Viewer
- Audio Player
- Video Player
- Document Viewer
- SMIL Player
- MPEG4 Player

The functionalities provided are explained in the following sections.

RELATIONSHIP WITH OTHER TOOLS

The Resource viewer is integrated as a part of the AXMEDIS Editor.

IMAGE VIEWER - DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

The Image Viewer is the part of the AXMEDIS Editor visualizing the images embedded as resources in an AXMEDIS object.

Double clicking on an image resource opens the Image Viewer. Right clicking on the image opens the contextual menu.

Features in the contextual menu of the image player are:

- Load, opens an external resource;
- Fit, fits the rendered image to the windows size;
- Zoom In, enlarges the image size view;
- Zoom out, reduces the image size view;
- Zoom, from 1% to 3000% to reduce or enlarge the size of the image by the given percent;
- Autofit, fits the rendered image to the windows size automatically.
- **Fullscreen**, to zoom the image to fit the size of the screen. To return to the normal view size, right click on the image and deselect the “Fullscreen” option in the contextual menu;
- **Set background colour**, changes the colour of the background choosing it using a colour palette 
- **Rotate figure**, rotates the image by 90, 180 and 270 degrees 
- **Mirror Figure**, creates a mirror image reflecting in the horizontal or in the vertical direction
- **Select region**, permits to select a portion of the image
- **Save region to file**, to save the selected region on disk
- **Save region as Resource**, to save the selected region to be copied as new resource in the object tree
- **Copy region**, to copy the selected region in the clipboard
- **Paste region**, to paste the selected region
- **Show controls**, hides or shows additional information and functions below the image, such as
  - the “Status” with the name and the size of the image;
  - the “Zoom buttons”;
- **Players viewers**, opens a different player.

### AUDIO PLAYER - DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

The Audio Player is the part of the AXMEDIS Editor playing the audio resources embedded in an AXMEDIS object

Double clicking on an audio resource opens the Image Viewer. Right clicking on the audio player opens the contextual menu.

---

Features in the contextual menu of the audio player are:
- **Load**, opens an external resource;
- **Start Time**, sets the point to start the audio playing
- **End Time**, sets the point to stop the audio playing
- **Set background colour**, changes the colour of the background choosing it using a colour palette
- **Show controls**, hides or shows additional information and functions, such as
  - the “Status” with the name of the file and the total duration,
  - the Current Time,
  - the Play/Stop buttons
  - the Slider;
  - the History button to go forward or back in the history of the showed resources
- **Players viewers**, opens a different player.
VIDEO PLAYER - DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

The Video Player is the part of the AXMEDIS Editor playing the videos embedded as resources in an AXMEDIS object. Double clicking on a video resource opens the Video Player. Right clicking in the video player opens the contextual menu.

Features in the contextual menu of the Video Player are:
- **Load**, opens an external resource;
- **Fit**, fits the rendered video to the windows size;
- **Zoom In**, enlarges the video size view;
- **Zoom out**, reduces the video size view;
- **Zoom**, from 1% to 3000% to reduce or enlarge the size of the video by the given percent;
- **Fullscreen**, to zoom the video to fit the size of the screen. To return to the normal view size, right click on the image and deselect the “Fullscreen” option in the contextual menu;
- **Start Time**, sets the point to start the video playing;
- **End Time**, sets the point to stop the video playing;
- **Set background colour**, changes the colour of the background choosing it using a colour palette;
- **Show controls**, hides or shows additional information and functions, such as:
  - the “Status” with the name of the file,
  - the Current Time,
  - the Play/Stop buttons
  - the Slider;
  - the Zoom buttons
  - the History button to go forward or back in the history of the showed resources;
- **Players viewers**, opens a different player.

DOCUMENT VIEWER - DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

The Document Viewer is the part of the AXMEDIS Editor visualizing the documents embedded as resources in an AXMEDIS object.

Double clicking on a document resource (typically an html, pdf, .doc, .rft, or .txt resource) opens the corresponding Document Player according to the document type.
Particularly the html viewer permits to view images or play resources linked into the html code and stored as single resources into the same AXMEDIS object.

**SMIL PLAYER - DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS**

The SMIL Player is the part of the AXMEDIS Editor visualizing the SMIL resources embedded in an AXMEDIS object. Double clicking on a SMIL resource in the tree opens the SMIL Player. Right clicking in the SMIL Player opens the contextual menu.

![SMIL Player Interface](image)

Features in the contextual menu of the SMIL Player are:
- **Load**, opens an external resource;
- **Fullscreen**, to view the SMIL resource in fullscreen. To return to the normal view, right click on the image and deselect the “Fullscreen” option in the contextual menu;
- **Start Time**, sets the point to start the SMIL resource playing
- **End Time**, sets the point to stop the SMIL resource playing
- **Set background colour**, changes the colour of the background choosing it using a colour palette
- **Show controls**, hides or shows additional information and functions, such as
  - the “Status” with the name of the file,
  - the Play/Stop buttons
- **Players viewers**, opens a different player.

**MPEG4 PLAYER - DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS**

The MPEG-4 player is a part of the AXMEDIS Editor playing the MPEG-4 resources embedded in an AXMEDIS object. Functionalities are similar to those of the Video player.
AXMEDIS Editor User Manual

**AXMEDIS METADATA EDITOR**

**MAIN FUNCTIONALITIES**

The Metadata Editor allows the user to add, edit, delete and view metadata elements including Dublin Core and AXMEDIS Info (AxInfo) using a simple HCI interface with pop up menus and editing boxes.

**RELATIONSHIP WITH OTHER TOOLS**

The Metadata Editor is integrated as a part of the AXMEDIS Editor.

**DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS**

The Metadata Editor can be opened by selecting “Metadata Editor” tab or by double clicking on a metadata (e.g. AXMEDIS Info or Dublin Core) in the Hierarchical View on the left side of the main AXMEDIS Editor. The Metadata Editor consists of two areas: one area for a Tree View displaying the metadata elements and one area for the editor and viewer tabs as illustrated in Figure 1.

![Figure 1](image)

The Editor provides the following functionalities:

- Adding child metadata elements to a Metadata element
- Inserting metadata elements
- Deleting Metadata elements
- Editing Metadata elements
ADDING METADATA ELEMENTS

Adding metadata elements to the metadata is achieved by right clicking on the element the user wishes to add a child to. A popup menu appears and the user can navigate to the “Add New Child Element to...” to see a list of valid child elements from which the user can select. These elements are derived from the metadata’s associated schema.

Another method for adding a child element is to use the Add Sub-Element list box in the Metadata Editor panel highlighted in Figure 3.

Figure 2

Figure 3
INSERTING METADATA ELEMENTS

To insert elements into the metadata is similar to the Adding functionality. To insert elements, the user right clicks on an element and navigates through the popup menu to “Insert Element to … “ as shown in Figure 4. The new element will be inserted above the selected element.

DELETING METADATA ELEMENTS

The user right clicks on the element he wishes to delete and then navigates through the popup menu to “Remove … Element” (see the left image in Figure 5). In this example the Dublin Core element “date” is removed. Alternatively, the user can click on the button “Delete Element” as shown in the right image in Figure 5.

EDITING METADATA ELEMENTS

Editing metadata elements is achieved by using the “Content’s Value” textbox in the “Element’s Content” frame, the “Enumeration” drop down box and the “Attribute” grid as shown in Figure 6. “Enumeration” is the only choices which...
the user can select for the element’s content. If the schema does not allow a field to be edited, for example there are no enumerations in Figure 6, the field will be disabled and greyed out.

To edit elements with enumeration choices, the user selected the enumeration value they require from the drop down box as shown in Figure 7.
AXMEDIS Editor User Manual

AXMEDIS METADATA MAPPER EDITOR

MAIN FUNCTIONALITIES

This is a GUI interface where a user can define mapping information to enable transformation between metadata documents. This mapping can be used to generate a style sheet which can then be used to transform metadata information in the AXCP tools.

RELATIONSHIP WITH OTHER TOOLS

The resulting style sheet can be used to transform metadata documents using the AXCP tools.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

CREATING A TRANSFORMATION XSLT TO MAP METADATA

The metadata mapper is integrated inside the AXMEDIS Editor, or more specifically inside the metadata editor. The main window of the metadata mapper consists of three components, the left hand component displays a tree view of the source metadata once it has been loaded, the right hand component displays a tree view of the destination metadata and the middle component is used to show the relationship between elements on either side.

Figure: Metadata Mapper

CREATING A TRANSFORMATION XSLT TO MAP METADATA

To create an XSLT style sheet, we first need to load two metadata documents. The first should contain an instance of the metadata you want to transform from (source metadata language), the second should contain an instance of the metadata you want to transform to (destination metadata language). This is achieved by selecting the “open” or
“open from database” buttons on top of the source and destination panels and navigating to the file you want to open. These operations can also be found in the menu “File”.

![Figure: Opening the Source File](image)

Once the two metadata documents have been successfully loaded, they will be displayed in the tree components as shown in the screenshot below.

![Figure: After opening source and destination Files](image)

Relationships can now be made by selecting an element from the left hand side and then selecting the related element from the right hand side. These connections can be updated by selecting a new element on the right hand side while the same node on the left hand side is kept selected.
The user can also map two parent nodes. In this case, all the children whose names are the same will be connected automatically. In the figure below the “title” and “creator” elements are connected as a result of mapping their parent nodes (i.e. the “Description” element).
A connection can be deleted by right clicking the connected element on the right hand side and selecting “Disconnect” from the popup menu. Alternatively, the user can select the source node and double click on the connected destination node to delete the connection. Deleting connection(s) is also available via the local toolbar buttons.

Figure: Deleting a connection

Once the user has finished mapping metadata, a XSLT style sheet can be saved which contains all the connection information. This is achieved by selecting the “save” toolbar button. Viewing the XSLT style sheet is also possible via the “view mapping file” toolbar button (right next to the above “save” button).
TRANSFORMING METADATA USING THE XSLT

Now that we have a style sheet describing how to map metadata from the source language to the target language, we can use it to transform a document. Normally the style sheet is meant to be used in the AXCP tools for automatic transformation. However, transforming the current document is also possible. This is achieved by clicking the “save” toolbar button on top of the destination panel (see the figure below). The user can also view the transformation using the “view transformed metadata” toolbar button (right next to the above “save” button).
Figure: Saving the transformed file
AXMEDIS VISUAL EDITOR

MAIN FUNCTIONALITIES
The SMIL Editor allows editing of SMIL resources from the AxEditor by a visual interface. It is divided in three parts: the tree view part that shows the whole SMIL structure, the visual part that shows the regions used for resources displaying and the behaviour part that shows the timing structure and properties.

RELATIONSHIP WITH OTHER TOOLS
The SMIL Editor and Player is integrated as a part of the AXMEDIS Editor.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

GENERAL COMMANDS
You can use the functions from the AXMEDIS Editor to create a new SMIL resource, open an existing SMIL resource, save a SMIL resource, and save a SMIL resource as another SMIL resource as follows:
SMIL RESOURCE COMBO BOX

You can use this function to select the SMIL resources embedded inside the AXMEDIS object as follows.
NEW SMIL

When you have no SMIL resource loaded and select the option of "New SMIL", the SMIL editor will create a very simple SMIL resource with only rootlayout and body container as follows:

When you have a SMIL resource loaded and select the option of "New SMIL", the SMIL editor will first ask you if you want to save the current resource or not and then create a very simple SMIL resource with only rootlayout and body container:
OPEN SMIL
When you have no SMIL resource loaded and select the option of “Open SMIL”, the SMIL editor will pop up a dialog for you to select the SMIL resources embedded inside the AXMEDIS object as follows:

When you have a SMIL resource loaded and select the option of “Open SMIL”, the SMIL editor will first pop up a dialog to ask you if you want to save the current resource or not and then pop up a dialog for you to select the SMIL resources embedded inside the AXMEDIS object:

SAVE SMIL
When you have no SMIL resource loaded and select the option of “Save SMIL”, the SMIL editor will pop up a dialog for you to save the SMIL resources as a new resource embedded inside the AXMEDIS object as follows:

When you have a SMIL resource loaded and select the option of “Save SMIL”, there will pop up a dialog asking you if you want to overwrite the current one or not.

SAVE SMIL AS
When you have a SMIL resource loaded and select the option of “Save SMIL As”, there will pop up a dialog for you to save this resource as another resource.
The tree part is located at the left of the SMIL Editor.

The tree view part of the SMIL editor is used for manipulating the whole SMIL with his internal structure. It is the only view that can display/edit completely Metas, Transitions and Links.
SMIL

You can add Metas and Transitions to the SMIL by right clicking the SMIL icon as follows.

ADD METAS

<table>
<thead>
<tr>
<th>Id</th>
<th>ID of the meta. It uniquely identifies the Meta within this SMIL.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Identifies a property name. It is required for Meta elements.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>abstract</td>
<td>Gives the presentation abstract.</td>
</tr>
<tr>
<td>author</td>
<td>Lists the presentation author's name.</td>
</tr>
<tr>
<td>base</td>
<td>Sets the base URL for the source clips.</td>
</tr>
</tbody>
</table>
AXMEDIS Editor User Manual

<table>
<thead>
<tr>
<th>Copyright</th>
<th>Supplies the presentation copyright.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>Gives the presentation title.</td>
</tr>
</tbody>
</table>

**Content:** Provides the content for the `name` attribute.

**ADD TRANSITIONS**

**ADD TRANSITION**

```
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>ID of the transition (default: empty). It uniquely identifies the transition within this SMIL.</td>
</tr>
<tr>
<td>Type</td>
<td>Specifies the main transition type (default: none).</td>
</tr>
<tr>
<td>Subtype</td>
<td>Defines an optional subtype for each type.</td>
</tr>
<tr>
<td>Dur</td>
<td>Defines the length of the transition effect.</td>
</tr>
<tr>
<td>Start</td>
<td>Starts the effect at a midway point.</td>
</tr>
<tr>
<td>End</td>
<td>Ends the effect before it completes fully.</td>
</tr>
<tr>
<td>Director</td>
<td>Specifies the transition direction (default value is forward).</td>
</tr>
<tr>
<td>FadeColor</td>
<td>Sets a colour for fades.</td>
</tr>
</tbody>
</table>
```

You can choose the option of “Select Colour” to customise the colour. There will pop up the following dialog.
META

You can edit the Meta by right click the Meta. There would be three options: “Rename”, “Modify”, “Delete”
**RENAME**

You can directly rename the Meta Name as the following.

![Image of AXMEDIS Editor](image)

**MODIFY**

By choosing the option of “Modify”, you can modify the Meta with the following dialog:

![Image of MODIFY META](image)

The attributes of this dialog are exactly the same as those of “Add Meta” described in 0.
DELETE

By choosing the option of “Delete”, you can delete the Meta with the following dialog:

TRANSITION

You can edit the transition by right click the transition. There would be three options: “Rename”, “Modify”, “Delete”.

![Transition editing in AXMEDIS Editor](image)
RENAME

You can directly rename the transition name as the following:

MODIFY

By choosing the option of “Modify”, you can modify the transition with the following dialog:

The attributes of this dialog are exactly the same as those of “Add transition” described in 0.
DELETE

By choosing the option of “Delete”, you can delete the transition with the following dialog:

![Warning Dialog]

ROOTLAYOUT

You can modify the size of the root-layout by right clicking the root-layout element in the Tree View part (The same function as in the Visual View part). There will pop up a menu. One option is to modify the size and colour of the root-layout. By inputting the values of the height and width, you can modify the size. With the dialog given, you can customize the colour as well. The left top position is fixed at the left top corner of the canvas.
**MODIFY**

**MODIFY ROOTLAYOUT**

<table>
<thead>
<tr>
<th>Colour</th>
<th>Sets the window background colour (default: black).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Width</td>
<td>Sets the main window width (default: 0).</td>
</tr>
<tr>
<td>Height</td>
<td>Sets the main window height (default: 0).</td>
</tr>
</tbody>
</table>

You can select the colour by right-clicking the "Select Colour" and there will pop up the colour for choice.
ADD REGION

You can add the region into the root layout by selecting the option of “Add Region”.

- **Id**: Uniquely identifies a region within the root-layout.
- **Name**: Identified the region. It is not unique.
- **Width**: Sets the region’s width (default: auto).
- **Left**: Sets the region’s offset from the window's left side (default: auto).
- **Right**: Sets the region’s offset from the window’s right side (default: auto).
- **Height**: Sets the region’s height (default: auto).
- **Top**: Sets the region’s offset from the top of the window (default: auto).
- **Bottom**: Sets the region’s offset from the bottom of the window (default: auto).
- **Z-Index**: Sets the stacking order when the region overlaps another region (default: 0).
- **Fit**: Controls how clips fit the region (default: hidden).
- **Show Background**: Determines when the background colour appears (default: always).

The governing equation for the horizontal dimension is: \( \text{bbw (bounding-box-width)} = \text{left} + \text{width} + \text{right} \). Given that each of these three parameters can have either a value of "auto" or a defined value not "auto", then there are 8 possibilities:

<table>
<thead>
<tr>
<th>Attribute values</th>
<th>Result before clipping to the bounding box</th>
</tr>
</thead>
<tbody>
<tr>
<td>left width right</td>
<td>left width right</td>
</tr>
<tr>
<td>auto auto auto</td>
<td>0 bbw 0</td>
</tr>
</tbody>
</table>
The vertical attributes height, bottom, and top are resolved similarly. The governing equation for the vertical dimension is: \( \text{bbh} = \text{top} + \text{height} + \text{bottom} \). Given that each of these three parameters can have either a value of "auto" or a defined value not "auto", then there are 8 possibilities:

<table>
<thead>
<tr>
<th>Attribute values</th>
<th>Result before clipping to the bounding box</th>
</tr>
</thead>
<tbody>
<tr>
<td>top</td>
<td>height</td>
</tr>
<tr>
<td>auto</td>
<td>auto</td>
</tr>
<tr>
<td>auto</td>
<td>auto</td>
</tr>
<tr>
<td>auto</td>
<td>defined</td>
</tr>
<tr>
<td>auto</td>
<td>defined</td>
</tr>
<tr>
<td>defined</td>
<td>auto</td>
</tr>
<tr>
<td>defined</td>
<td>auto</td>
</tr>
<tr>
<td>defined</td>
<td>defined</td>
</tr>
<tr>
<td>defined</td>
<td>defined</td>
</tr>
</tbody>
</table>

**REGION**

You can modify the size of the region manually with dialog menu by right clicking the region element of the Tree View part (the same function of the Visual View part). There will pop up a menu.

Options available are:
- Rename this region
- Modify the size, position, index and colour of the region
- Delete this region
- Add some sub-regions inside this region
- Show media resources associated with this region

By inserting the values of the height and width, you can modify the size. With the dialog given, you can customize the colour as well.
RENAME

You can directly rename the region’s name as the following:

MODIFY

[Diagram of the MODIFY REGION window showing properties like Id, Name, Width, Height, and other settings for the content-image region.]
You can modify the region by choosing the option of “modify region”. There will pop up a dialog menu. The attributes are exactly the same as the part “add region” described in the root layout editing.

**DELETE**
You can also delete the region by choosing the option of “delete”.

1) When there is no media resource associated with this region, there will popup the following warning dialog.

![Warning Dialog 1]

2) When there are some child regions associated with this region, there will popup the following warning dialog asking if you want to delete this region and all his children.

![Warning Dialog 2]

3) When there are some media resources associated with this region, there will popup the following warning dialog asking if you want to delete or change region for these resources. By option “Delete”, you can delete the region and the resources inside.

![Warning Dialog 3]

You can choose to use other regions to host these resources as follows. There will appear a list of all the other regions for changing.

![Warning Dialog 4]
ADD REGION

You can add new region by choosing the option of “Add Region”. There will pop up a dialog menu. The attributes of “Add Region” in region editing are exactly the same as the part “Add Region” described in the root layout editing.
SHOW MEDIAS

You can check which media is associated the region by choosing the option of “Show Medias”. You can select the Media from them for editing.

BODY

The Body is also the main sequence and is a not modifiable sequence of timings. You can add Containers and Medias to it by right clicking on it and using the menu.

ADD CONTAINER

To add a Container to the Body right click on it and select the submenu “Add Container” and choose the type of container you want to add.

For example if you create a Par, a window like this will be displayed:
Now you can set the new Par properties and create it by clicking on “Ok”, you can also cancel the creation by clicking on “Cancel” or exiting the window.

**ADD MEDIAS**

To add a Media to the Body right click on it and select the submenu “Add Media” and choose the type of media you want to add.

For example if you create an Audio, a window like this will be displayed:
Now you can set the new Audio properties and create it by clicking on “Ok”, you can also cancel the creation by clicking on “Cancel” or exiting the window.

**COMMON TIMING OPERATIONS**

There are some common operations between Medias and Containers. You can see in the Menu the commands Move Up, Move Down and Rename are both in Media and Container Menus. Modify and delete are also common in the menus but loads different actions.

**MOVE UP**

The Move Up command can be loaded by right click on the Media/Container and selecting Move Up. It moves the timing selected up in the structure (in the order of his parent). It is disabled when the timing selected is at the top of his parent.

**MOVE DOWN**

The Move Down command can be loaded by right click on the Media/Container and selecting Move Down. It moves the timing selected down in the structure (in the order of his parent). It is disabled when the timing selected is at the bottom of his parent.
AXMEDIS Editor User Manual

RENAME

The Rename command can be loaded by right click on the Media/Container and selecting Rename. It opens the name editing. You can also open the editing by fast clicking on it.

When you end the editing, the name is changed.

CONTAINERS

The Containers are the logical structure of the whole timing of the SMIL. You can add Containers and Medias to a Container by right clicking on it and using the menu. Containers could be Par, Seq or Excl for parallel, sequential and exclusive playing.

ADD CONTAINER

To add a Container to a Container right click on it and select the submenu “Add Container” and choose the type of container you want to add.

For example if you create a Par, a window like this will be displayed:
Now you can set the new Par properties and create it by clicking on “Ok”, you can also cancel the creation by clicking on “Cancel” or exiting the window.

**ADD MEDIAS**

To add a Media to a Container right click on it and select the submenu “Add Media” and choose the type of media you want to add.

For example if you create an Audio, a window like this will be displayed:
Now you can set the new Audio properties and create it by clicking on “Ok”, you can also cancel the creation by clicking on “Cancel” or exiting the window.

**MODIFY**

To modify by dialog a Container right click on it and select “Modify”. A window like this will be displayed.

There you can set the following properties:

**Id:** ID of the container (default: empty)

The ID is used identify the Container in the SMIL
Begin: Begin of the container (default: not set)
The Begin set the start of the container; it is a list of beginnings. The general definition of W3C is supported:
http://www.w3.org/TR/SMIL2/smil-timing.html#adef-begin
End: End of the container (default: not set)
The End set the end of the container; it is a list of endings. The general definition of W3C is supported:
http://www.w3.org/TR/SMIL2/smil-timing.html#adef-end
Dur: Duration of the container (default: not set)
The Dur set the duration of the container; it is a single Duration value.
RepeatCount: Number of repetitions (default: not set)
The RepeatCount is used when you want to repeat the container a defined number of times.
RepeatDur: Duration of repetitions (default: not set)
The RepeatCount is used when you want to repeat the container for a defined duration.
Fill: Default Fill for Medias in this Container (default: auto)
The Fill is used to set default filling for Medias in the Container. It can have the following values: auto, remove, freeze, hold and transition.

DELETE
To delete a Container right click on it and click on “Delete”. A confirms dialog will appear:
Like this one if the Container contains something.

![Warning dialog](image1)

Like this one if not.

![Warning dialog](image2)

In both cases click Yes to delete the Container or Cancel or exit the window if you changed your mind.

MEDIAS
The Medias are the resources played in the SMIL. You can add Link to a Media or select his Region by right clicking on it and using the menu. Medias could be Audio, Image, Text and Video for respective type of files playing.
ADD LINK

To add a Link to a Media right click on it and click the menu “Add Link”, a window like this will be displayed:

There you can set the following properties of the new Link:

- **Id:** ID of the link (default: empty)
The ID is used identify the Link in the SMIL (used most of time for internal links).

- **Href:** Link to load (default: empty)
The Href is used for external Links by setting a link to another SMIL file.

- **Title:** Title of the Link (default: empty)
The Title is displayed as information when the SMIL is played.

- **Coords:** Coordinates of the Link Region (default: empty= entire Region)
The Coords are a list of coordinates to define a sub-region mouse sensitive to load this link (the entire Region is used if it is not defined).

The Resource button allows setting the Href by selecting a SMIL resource in Axom. It loads a window like this:

You simply select the resource you want to link and validate; the new Href will be set.

SELECT REGION

To select the Region where a Media is played, simply right click on it and click the menu “Show Region”. The Region of the Media will be selected. (In Tree and Visual View)

MODIFY

To modify by dialog a Media right click on it and select “Modify”. A window like this will be displayed.
There you can set the following properties:

**Id:** ID of the Media (default: empty)
The ID is used to identify the Media in the SMIL.

**Source:** Source of the Media (default: empty)
The Source is used to link the resource played by this Media.

**Region:** Region of the Media (default: first Region)
The Region defines the area where the Media will be played. You can select each region you have defined.

**In Transition:** Transition at start (default: not defined)
The In Transition defines the Transition at start where the Media will be played. You can select each Transition you have defined.

**Out Transition:** Transition at end (default: not defined)
The Out Transition defines the Transition at end where the Media will be played. You can select each Transition you have defined.

**Begin:** Begin of the Media (default: not set)
The Begin set the start of the Media; it is a list of beginnings. The general definition of W3C is supported: [http://www.w3.org/TR/SMIL2/smil-timing.html#adef-begin](http://www.w3.org/TR/SMIL2/smil-timing.html#adef-begin)

**End:** End of the Media (default: not set)
The End set the end of the Media; it is a list of endings. The general definition of W3C is supported: [http://www.w3.org/TR/SMIL2/smil-timing.html#adef-end](http://www.w3.org/TR/SMIL2/smil-timing.html#adef-end)

**Dur:** Duration of the Media (default: not set)
The Dur set the duration of the Media; it is a single Duration value.

**RepeatCount:** Number of repetitions (default: not set)
The RepeatCount is used when you want to repeat the Media a defined number of times.

**RepeatDur:** Duration of repetitions (default: not set)
The RepeatCount is used when you want to repeat the Media for a defined duration.

**Fill:** Filling of the Media (default: auto)
The Fill is used to set the filling for the Media. It can have the following values: auto, remove, freeze, hold and transition.

The Resource button allows setting the Source by selecting a resource in Axom (selected by mime type). It loads a window like this:

![Select Resource dialog](image)

You simply select the resource you want to link and validate; the new Source will be set.

**DELETE**

To delete a Media right click on it and click on “Delete”. A confirms dialog will appear:

![WARNING dialog](image)

Click Yes to delete the Media or Cancel or exit the window if you changed your mind.

**LINKS**

The Links are the way to jump from a Media. They can be renamed, modified and deleted by menu.

**MODIFY**

To add a Link to a Media right click on it and click the menu “Modify”, a window like this will be displayed:

![MODIFY LINK dialog](image)

There you can set the following properties of the new Link:

**Id:** ID of the link (default: empty)

The ID is used identify the Link in the SMIL (used most of time for internal links).

**Href:** Link to load (default: empty)

The Href is used for external Links by setting a link to another SMIL file.

**Title:** Title of the Link (default: empty)

The Title is displayed as information when the SMIL is played.
**Coords:** Coordinates of the Link Region (default: empty= entire Region)
The Coords are a list of coordinates to define a sub-region mouse sensitive to load this link (the entire Region is used if it is not defined)
The Resource button allows setting the Href by selecting a SMIL resource in Axom. It loads a window like this:

![Select Resource](image)

You simply select the resource you want to link and validate; the new Href will be set.

**RENAME**
The Rename command can be loaded by right click on the Link and selecting Rename. It opens the name editing. You can also open the editing by fast clicking on it.

![Rename](image)

When you end the editing, the name is changed.

**DELETE**
To delete a Link right click on it and click on “Delete”. A confirms dialog will appear:

![WARNING](image)

Click Yes to delete the Link or Cancel or exit the window if you changed your mind.

**VISUAL VIEW PART**
The visual part is located at the right top of the SMIL Editor.
The visual view part of the SMIL editor is used for manipulating the SMIL layout which including the Rootlayout, Regions, Sub-regions of regions.
ROOT-LAYOUT EDITING

You can modify the size of the root-layout by using the mouse to change the size directly or manually with dialog menu by right clicking the root-layout. There will pop up a menu for you. One option is to modify the size and colour of the root-layout. By inputting the values of the height and width, you can modify the size. With the dialog given, you can customize the colour as well. The left top position is fixed at the left top corner of the canvas.

MODIFY

You modify the size and colour of the root-layout by inputting the values of the height and width.
You can select the colour by right-clicking the “select colour” and there will pop up the colour for choice.

**ADD REGION**

You can add the region into the root layout by selecting the option of “add region”.

---

**Section: AXMEDIS Visual Editor**
REGION EDITING
You can modify the size of the region by using the mouse to change the size directly or manually with dialog menu by right clicking the corresponding region. There will pop up a menu for you. One option is to modify the size, position, index, and colour of the region. One option is to delete this region. One option is add some sub-regions inside this region, one option is to show the media resources which are associated with this region. By inputting the values of the height and width, you can modify the size. With the dialog given, you can customize the colour as well. By directly editing the region with mouse, you can move the region by holding the left key of the mouse and moving it. You can also drag the border of the region to resize it. The sub regions cannot exceed the borders of their parent regions. Also they cannot be moved out of the borders of their parent regions.
MODIFY

You can modify the region by choosing the option of “Modify”. There will pop up a dialog menu for you.

DELETE

You can also delete the region by choosing the option of “Delete”.

1) When there is no media resource associated with this region, there will popup the following warning dialog.

2) When there are some child regions associated with this region, there will popup the following warning dialog asking if you want to delete this region and all his children.

3) When there are some media resources associated with this region, there will popup the following warning dialog asking if you want to delete or change region for these resources. By option “Delete”, you can delete the region and the resources inside.
You can choose to use other regions to host these resources as follows. There will appear a list of all the other regions for changing.

You can add new region by choosing the option of “add region”. There will pop up a dialog menu for you. The attributes of “add region” in region editing are exactly the same as the part “add region” described in the root layout editing.
SHOW MEDIAS

You can check which media is associate the region by choosing the option of “Show Medias”. You can select the media from them for editing.

BEHAVIOUR VIEW PART

The behaviour part is located at the right bottom of the SMIL Editor.

DESCRIPTION OF COMPONENTS

At the top of the behaviour view you can see the time ruler. This ruler gives timing information and a scaling view for the whole SMIL timing.
At the centre you can see and modify the timing attributes of the SMIL. At the top there is always a not modifiable anonymity red bar that represents the main sequence that is the basic timing structure of the SMIL file. Under it, that means “in it”, you can add/modify/delete others containers like “Par” for parallel playing (all the contained together), “Seq” for Sequential playing (all the contained one after another) and “Excl” for Exclusion playing (only one of the contained upon a time). You can also add/modify/delete the media that will be played. If the timings are too big navigation scroll bars appears.

At the bottom, you can see and modify the zoom scale used for timing display.

**TIME RULER AND ZOOM**

Time ruler and zoom show and allow seeing the global timing. One important thing to remember is that if durations are not defined for the whole SMIL file, the timing you will get in a player will be different from the one displayed because all not known durations are set by default to 20 seconds in the behaviour view. (See below for more details).

**TIMINGS**

Timings are both Main Sequence and other Medias and Containers (boxes in the middle). The meanings of the colours are:

- Red: Sequence Container
- Blue: Parallel Container
- Yellow: Exclusive Container
- Pink: Audio Media
Green: Image Media
Cyan: Video Media
White: Text Media
When a colour is shining the duration of the Media/Container is set.
When a colour is greyed the duration of the Media/Container is not set.

MAIN SEQUENCE

The main sequence is a not modifiable sequence of timings. You can add Containers and Medias to it by right clicking on it and using the menu.

Add Container
To add a Container to the Main Sequence right click on it and select the submenu “Add Container” and choose the type of container you want to add.

For example if you create a Par, a window like this will be displayed:
Now you can set the new Par properties and create it by clicking on “Ok”, you can also cancel the creation by clicking on “Cancel” or exiting the window.

**Add Medias**
To add a Media to the Main Sequence right click on it and select the submenu “Add Media” and choose the type of media you want to add.

For example if you create an Audio, a window like this will be displayed:
Now you can set the new Audio properties and create it by clicking on “Ok”, you can also cancel the creation by clicking on “Cancel” or exiting the window.

**COMMON TIMING OPERATIONS**

There are some common operations between Medias and Containers. You can see in the Menu the commands Move Up, Move Down and Undefine Duration are both in Media and Container Menus. Modify and delete are also common in the menus but loads different actions.

The mouse modifications are also in common.
Move Up
The Move Up command can be loaded by right click on the Media/Container and selecting Move Up. It moves the timing selected up in the structure (in the order of his parent). It is disabled when the timing selected is at the top of his parent.

Move Down
The Move Down command can be loaded by right click on the Media/Container and selecting Move Down. It moves the timing selected down in the structure (in the order of his parent). It is disabled when the timing selected is at the bottom of his parent.

Undefine Duration
The Undefine Duration command can be loaded by right click on the Media/Container and selecting Undefine Duration. It removes a definition of Duration of the Timing (don’t remove them all). In order (when available) : Dur, Ends in the order of apparition.

Mouse modifications
The Begin, End and Duration attributes of timings can also be modified by mouse. To modify media/container, left click on it. If you keep the left down drag & drop is activated. The resource can only be moved horizontally. When the resource is dropped it takes the new time offset for begin.

When the media/container is left clicked, a selection appears. When you drag and drop the black squares it changes durations and start (when using the left one).

Please note that Seq doesn’t accept negative values, so the resource will be adjusted to be correct. Another thing to note is that if the duration of your timing was not defined, it changes his colour to the “duration set” one.
CONTAINERS

The Containers are the logical structure of the whole timing of the SMIL. You can add Containers and Medias to a Container by right clicking on it and using the menu. You can also set timings properties by mouse and by the menu. Containers could be Par, Seq or Excl for parallel, sequential and exclusive playing.

Add Container

To add a Container to a Container right click on it and select the submenu “Add Container” and choose the type of container you want to add.

For example if you create a Par, a window like this will be displayed:

Now you can set the new Par properties and create it by clicking on “Ok”, you can also cancel the creation by clicking on “Cancel” or exiting the window.
Add Medias
To add a Media to a Container right click on it and select the submenu “Add Media” and choose the type of media you want to add.

For example if you create an Audio, a window like this will be displayed:
Now you can set the new Audio properties and create it by clicking on “Ok”, you can also cancel the creation by clicking on “Cancel” or exiting the window.

Modify
To modify by dialog a Container right click on it and select “Modify”. A window like this will be displayed.

There you can set the following properties:

**Id:** ID of the container (default: empty)
The ID is used identify the Container in the SMIL

**Begin:** Begin of the container (default: not set)
The Begin set the start of the container; it is a list of beginnings. The general definition of W3C is supported: [http://www.w3.org/TR/SMIL2/smil-timing.html#adef-begin](http://www.w3.org/TR/SMIL2/smil-timing.html#adef-begin)

**End:** End of the container (default: not set)
The End set the end of the container; it is a list of endings. The general definition of W3C is supported: [http://www.w3.org/TR/SMIL2/smil-timing.html#adef-end](http://www.w3.org/TR/SMIL2/smil-timing.html#adef-end)

**Dur:** Duration of the container (default: not set)
The Dur set the duration of the container; it is a single Duration value.

**RepeatCount:** Number of repetitions (default: not set)
The RepeatCount is used when you want to repeat the container a defined number of times.

**RepeatDur:** Duration of repetitions (default: not set)
The RepeatCount is used when you want to repeat the container for a defined duration.

**Fill:** Default Fill for Medias in this Container (default: auto)
The Fill is used to set default filling for Medias in the Container. It can have the following values: auto, remove, freeze, hold and transition.

Note: the offset value for Begin, End and Dur can be also set by mouse.

Delete
To delete a Container right click on it and click on “Delete”. A confirms dialog will appear: Like this one if the Container contains something.
Like this one if not.

In both cases click Yes to delete the Container or Cancel or exit the window if you changed your mind.

**MEDIAS**

The Medias are the resources played in the SMIL. You can add Link to a Media or select his Region by right clicking on it and using the menu. You can also set timings properties by mouse and by the menu. Medias could be Audio, Image, Text and Video for respective type of files playing.

**Add Link**

To add a Link to a Media right click on it and click the menu “Add Link”, a window like this will be displayed:

There you can set the following properties of the new Link:

- **Id**: ID of the link (default: empty)
  The ID is used identify the Link in the SMIL (used most of time for internal links).

- **Href**: Link to load (default: empty)
  The Href is used for external Links by setting a link to another SMIL file.

- **Title**: Title of the Link (default: empty)
  The Title is displayed as information when the SMIL is played.

- **Coords**: Coordinates of the Link Region (default: empty= entire Region)
  The Coords are a list of coordinates to define a sub-region mouse sensitive to load this link (the entire Region is used if it is not defined)

The Resource button allows setting the Href by selecting a SMIL resource in Axom. It loads a window like this:

You simply select the resource you want to link and validate; the new Href will be set.

**Select Region**

To select the Region where a Media is played, simply right click on it and click the menu “Show Region”. The Region of the Media will be selected (in Tree and Visual View)
Modify
To modify by dialog a Media right click on it and select “Modify”. A window like this will be displayed.

There you can set the following properties:

**Id:** ID of the Media (default: empty)
The ID is used to identify the Media in the SMIL

**Source:** Source of the Media (default: empty)
The Source is used to link the resource played by this Media

**Region:** Region of the Media (default: first Region)
The Region defines the area where the Media will be played. You can select each region you have defined.

**In Transition:** Transition at start (default: not defined)
The In Transition defines the Transition at start where the Media will be played. You can select each Transition you have defined.

**Out Transition:** Transition at end (default: not defined)
The Out Transition defines the Transition at end where the Media will be played. You can select each Transition you have defined.

**Begin:** Begin of the Media (default: not set)
The Begin set the start of the Media; it is a list of beginnings. The general definition of W3C is supported:
http://www.w3.org/TR/SMIL2/smil-timing.html#adef-begin

**End:** End of the Media (default: not set)
The End set the end of the Media; it is a list of endings. The general definition of W3C is supported:
http://www.w3.org/TR/SMIL2/smil-timing.html#adef-end

**Dur:** Duration of the Media (default: not set)
The Dur set the duration of the Media; it is a single Duration value.

**RepeatCount:** Number of repetitions (default: not set)
The RepeatCount is used when you want to repeat the Media a defined number of times.

**RepeatDur:** Duration of repetitions (default: not set)
The RepeatCount is used when you want to repeat the Media for a defined duration.

**Fill:** Filling of the Media (default: auto)
The Fill is used to set the filling for the Media. It can have the following values: auto, remove, freeze, hold and transition.

The Resource button allows setting the Source by selecting a resource in Axom (selected by mime type). It loads a window like this:

![Select Resource Window]

You simply select the resource you want to link and validate; the new Source will be set.

**Note:** the offset value for Begin, End and Dur can be also set by mouse.

**Delete**
To delete a Media right click on it and click on “Delete”. A confirms dialog will appear:

![Delete Confirmation Dialog]

Click Yes to delete the Media or Cancel or exit the window if you changed your mind.

---

**EXAMPLE**

**INTRODUCTION**
For this small example we will create a little SMIL presentation from scratch. It will demonstrate how to use this editor for a simple interactive presentation.

**BASIC FILES**
A SMIL play other files. So we will load the createSMILexample.axm file:
CREATE THE SMIL
Now we have to create a new SMIL Resource. So we use the menu VisualEditor:

The New empty SMIL appears:

EDIT SMIL
Now we can edit our new SMIL resource.

MODIFY ROOT LAYOUT
The Root Layout dimensions are not the one we want to use. So we change them:
And also by menu...

... we change the Colour and Adjust the Sizes

ADD REGIONS
To add the first Region we right click on Layout.

And add a new Region: Audio for Audio playing
And also another one named Main for Main window.

Note that this time we have set the Fit to “fill”.

We finally add another Region in Main:
To be a little region in the right bottom part.
TIMING

With the Region defined we can go to the Timing part.

Containers

First we will define some Containers.

A Par to play sound and images in parallel
And an Excl in it, 2 Par in the Excl named opt1 and opt2 and finally one Seq in the new Par.

Medias
Now we will add the Medias in the containers.
First we add an Audio Media named sound to play the mp3 file we have in the main Par.
We set the mp3 link by clicking on Resource. And select the Region to be Audio.

In the two Pars we add the Image groupe.jpg with names skip1 and skip2 in the Skipping region.
Creating Transitions

We haven’t defined the Transitions since yet. We add a new transition in the SMIL:

A fading one:

```
ADD TRANSITION

Id: fading
Type: fade
Subtype: crossfade
Dur: 1.000
Start: 0
End: 1
Direction: forward
FadeCol: #000000
```

And a fanning one:

```
ADD TRANSITION

Id: fanning
Type: doubleFanWipe
Subtype: fanInHorizontal
Dur: 1.000
Start: 0
End: 1
Direction: forward
FadeCol: #000000
```

Now we can use these Transitions.
Setting timing
Now we will set Timing properties to our images and a Transition between them.

We set all durations to 10 to the images in the Seq and fading transitions between images of the first Seq and fanning to the second one.

By using the Zoom we obtain a behaviour view like this:
Now we have only to define the start of Par in the Excl. We want that opt1 to be the default and to switch by the skipping. We also take the opportunity to cycle the Par by setting RepeatDur to indefinite.

So we edit opt1:

![Modify Par Dialog]

And opt2
And we add an end to the Excl:

**SAVING SMIL**

Now we can save our work.

And find an original name.
Now you can see it in the Resource Viewer. The image at right bottom is interactive, as set before.

Ok, it is not perfect (and the sound in French), but now can modify it again... It's up to you now!
AXMEDIS Editor User Manual

AXMEDIS BEHAVIOUR EDITOR

AXMEDIS objects embedded scripting support aims to add dynamic behaviour and intelligence to these objects. AXMEDIS object scripting capabilities combine MPEG-21 DIP scripting support with some of the already present AXMEDIS scripting extensions.

This may lead to enforce different degrees of intelligence and dynamic behaviour of the objects, leaving at the authors’ creativity the option of proposing to final users new experiences for interaction and thus new business models.

The user interface of the Behaviour Editor is divided in three main parts:

- The Method text part, where the rule in Javascript has to be written
- The Arguments part, where edit and add arguments
- The Output part, the some messages during the script execution is reported.

When a Javascript rule has been written in the Method text part, it is possible to save it by pressing the save button Immediately the new rule will be added in the resource tree list identified by the Method word followed by the method name in brackets.
AXMEDIS supports and extends the MPEG-21 DIP standard set of functionalities. AXMEDIS already has a significant scripting support used for automating some tasks like content production. These capabilities are easily adaptable to be included in the DIP Architecture. Therefore, the functionalities suitable in DIP field could be converted and reused. It is possible to access to the current Axmedis Object using two ways:

1) DIP/MPEG21 standard way via `didDocument` global object (it is a DOM object)
2) AXMEDIS way using `axDocument` global object (it is an AxmedisObject)

The next samples show how to implement a javascript SlideShow using the above mentioned objects:

**Access using global object didDocument (MPEG21 based)**

```javascript
function slideshowDIP()
{
    var domResources=didDocument.getElementsByTagName("Resource")
    var audioStatus = null
    for(var i=0; i< domResources.length;i++)
    {
        var mimetype=domResources[i].getAttribute("mimeType")
        if(mimetype.toLowerCase().match("audio"))
        {
            audioStatus=DIP.play(domResources [i],true)
            break;
        }
    }
    for(var i=0; i<domResources.length;i++)
    {
        var mimetype=domResources[i].getAttribute("mimeType")
        if(mimetype!="application/mp21-method" && !mimetype.toLowerCase().match("audio"))
        {
            var playStatus=DIP.play(domResources[i],true)
            DIP.wait(1000)
            DIP.release(playStatus)
        }
    }
    if(audioStatus != null)
    DIP.release(audioStatus)
}
```

**Access using global object axDocument (AXMEDIS based)**

```javascript
function slideshowAX()
{
    var content = axDocument.getContent()
    var audioStatus = null
    for(var i=0; i<content.length;i++)
    {
        if(content[i] instanceof AxResource)
        {
            var mimetype=content[i].mimeType
            if(mimetype.toLowerCase().match("audio"))
            {
                audioStatus=DIP.play(content[i],true)
                break;
            }
        }
    }
    for(var i=0; i<content.length;i++)
    {
```
if(content[i] instanceof AxResource)
{
    var mimetype=content[i].mimeType
    if(mimetype!="application/mp21-method" &&
    !mimetype.toLowerCase().match("audio"))
    {
        var playStatus=DIP.play(content[i],true)
        DIP.wait(3000)
        DIP.release(playStatus)
    }
}
}
if(audioStatus != null)
DIP.release(audioStatus)
N.B.: the axoid is not the full axoid but just the last part as shown in the following example.

**Example of full axoid:**

```plaintext
urn:axmedis:00000:obj:42fc9a46-4a61-43a2-af34-f615abe3aab7
```

**Part used in to links:**

```plaintext
42fc9a46-4a61-43a2-af34-f615abe3aab7
```

Only the root object and the child containing the resources/methods axoids are required. To maintain the new structure backward compatible all the resources and methods are reported into the root object as well.

If the object absolute url looks like the following:

```plaintext
axmedis://42fc9a46-4a61-43a2-af34-f615abe3aab7/68bcd4c8-464d-458b-8d40-b30a780dd257/resource-localpath
```

it is also possible to use this syntax (even if it's not advised):

```plaintext
axmedis://42fc9a46-4a61-43a2-af34-f615abe3aab7/resource-localpath
```

as the resource was into the root object.

N.B.: if more than one object have the same link, just the first found into the axmedis file is accessible with this syntax.

### DIP RELATED OPERATIONS

These DIBOs provide access to DIP related functionality. The DIBOs are provided as function properties of a DIP object. DIML includes such a DIP object as a property of the global.

**EXAMPLE**

**Calling a DIP DIBO**

```plaintext
function foo(arg1)
{
    ...
    DIP.Play( component, false );
    ...
}
```

---

### ALERT

**INTERFACE**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>alert(message, messageType)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Alerts the User with a message.</td>
</tr>
</tbody>
</table>
| Parameters      | message
|                 | A string value containing the message to be displayed. |
|                 | messageType
|                 | A number value indicating the generic nature of the message. |
|                 | Valid values are MSG_INFO, MSG_WARNING, MSG_ERROR and MSG_PLAIN which are defined as value properties of the DIML global object. |
| Return value    | None |
| Exceptions      | DIPError
|                 | With DIP error code INVALID_PARAM
|                 | — If message is not a string value; or
|                 | — If messageType does not specify a valid value. |
**SEMANTICS**

This DIBO provides simple textual feedback to the User.

EXAMPLE  
To notify the User of an error condition.

The textual message is specified by the message parameter. It is an error to invoke this DIBO if the message parameter is not a string value, in which case an invalid parameter exception is generated.

The messageType parameter indicates the nature of the message. It is an error to invoke this DIBO if the messageType parameter is not a valid number value, in which case an invalid parameter exception is generated (a valid number value is one of the values MSG_INFO, MSG_WARNING, MSG_ERROR, or MSG_PLAIN defined as number properties of the DML global object).

NOTE  The implementation of the alert is a DIBO implementation decision. For example, one DIBO implementation might present the message in a GUI to a human user. Another implementation might be intended to process a DIL without human intervention and log the message to a log file.

---

**GETEXTERNALDATA**

**INTERFACE**

<table>
<thead>
<tr>
<th>Syntax:</th>
<th>getExternalData(mimeTypes, requestMessages)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description:</td>
<td>Requests the User to select resources located external to the DI.</td>
</tr>
<tr>
<td>Parameters:</td>
<td>mimeTypes</td>
</tr>
<tr>
<td></td>
<td>An array of arrays of string values specifying the media formats required.</td>
</tr>
<tr>
<td></td>
<td>requestMessages</td>
</tr>
<tr>
<td></td>
<td>An array whose elements are either string values or null.</td>
</tr>
<tr>
<td>Return value:</td>
<td>Returns an array whose elements are string values specifying the URLs that describe the location of the resources.</td>
</tr>
<tr>
<td>Exceptions:</td>
<td>DIPError</td>
</tr>
<tr>
<td></td>
<td>With DIP error code INVALID_PARAMETER</td>
</tr>
<tr>
<td></td>
<td>— if mimeTypes is not an array of arrays of string values, or</td>
</tr>
<tr>
<td></td>
<td>— if requestMessages is not null and is not an array the same size as mimeTypes with each array member being a string value or null.</td>
</tr>
</tbody>
</table>

---

**SEMANTICS**

This DIBO requests the User to select resources external to the current DID.

EXAMPLE  
An, audio or video resource.

The mimeTypes parameter specifies allowable MIME types of the resources to be selected. It is an array with an element for each resource which is to be selected. Each element is another array of string values. The string values in one array specify the allowed MIME types for a single resource for which the User is to select.

The MIME type values are given in the form type/subtype. The values for the type and subtype of the MIME type should be valid values.

EXAMPLE  
A value of image/jpeg specifies a JPEG image is allowed to be selected.

It is an error to invoke this DIBO if the mimeTypes parameter is not an array where each array entry is itself an array of string values, in which case an invalid parameter exception is generated.

The requestMessages parameter is an array of string values that contains a request message for each of the resource selections indicated by the members of the mimeTypes array. The requestMessages parameter may be null, in which case no explicit request messages are specified. If the requestMessages parameter is not null, it is an error if the number of members in the requestMessages parameter is not equal to the number of members in the mimeTypes parameter. However one or more of the members of the requestMessages parameter may be null. It is an error if a member of the requestMessages parameter is not null and is not a string value.
The selected resources are identified to the invoking DIM by the URLs returned by the DIBO. A null return value indicates that no resource was selected.

NOTE The presentation and selection of the resource is a DIBO implementation decision. For example, one DIBO implementation might present the resource selection in a GUI to a human allowing the human to select the resource. Another implementation might be intended to process a DI without human intervention and implement some form of automated resource selection.

**EXAMPLE USE**

An example where this DIBO can be used is when a RESOURCE element is created and a DIM needs to set the ref attribute and requests the User to locate a resource to which the ref attribute will refer.

### GETOBJECTMAP

#### INTERFACE

<table>
<thead>
<tr>
<th>Syntax:</th>
<th><code>getObjectMap(document)</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>Description:</td>
<td>Retrieves the Object Map from a DID instance document.</td>
</tr>
<tr>
<td>Parameters:</td>
<td><code>document</code> A DOM Document representing the DID instance document containing the Object Type information.</td>
</tr>
<tr>
<td>Return value:</td>
<td>An ObjectMap object representing the Object Map of the DID instance document.</td>
</tr>
<tr>
<td>Exceptions:</td>
<td>DIPError With DIP error code INVALID_PARAMETER if document is not a DOM Document object representing a DIDL instance document.</td>
</tr>
</tbody>
</table>

#### SEMANTICS

This DIBO allows the retrieval of an Object Map from a DID instance document. The `document` parameter shall be a DOM Document object representing the DIDL instance document with the Object Type information needed to construct the Object Map. The Object Map is represented in DIML by an ObjectMap object. It is an error to invoke this DIBO if the `document` parameter is not a DOM Document object representing a DIDL document, in which case an invalid parameter exception is generated.

**EXAMPLE USE**

An example where this DIBO can be used is if a DIM author wants to process all DID Objects of a particular Object Type. For example, in a music album Digital Item, to get all Objects associated with a music track Object Type, the DIM author can use the `getObjects` operation of the ObjectMap object. Prior to this, the DIM author would need to call the `getObjectMap` DIBO to first retrieve the ObjectMap.

### GETOBJECTS

#### INTERFACE

<table>
<thead>
<tr>
<th>Syntax:</th>
<th><code>getObjects(objectTypes, requestMessages)</code></th>
</tr>
</thead>
</table>
Description: Provides the User with one or more selections of Objects of given Object Types. The Objects for each selection are those elements identified in the Object Map contained in the DI to be of the given Object Types.

Parameters:  
objectTypes  
An Array of string values specifying the Object Type names.

requestMessages  
An array whose elements are either string values or null.

Return value: Returns an array whose elements are either a DOM Element object or null. A DOM Element object represents the selected element of the corresponding Object Type as specified in the objectTypes array.

Exceptions:  
DIPError  
With DIP error code INVALID_PARAMETER
  — if .objectTypes is not an array of string values, or
  — if .requestMessages is not null and is not an array the same size as objectTypes with each array member being a string value or null.

SEMANTICS  
This DIBO requests the User to select DIDL elements of specified Object Types from the current DID containing or referencing the DIM from which the DIBO is invoked.

DIM authors should provide DID Objects, representing DIDL elements, to be operated on in a DIM as DIM Arguments declared by the DIM declaration. DIM authors should only use the getObjects DIBO when it is not possible to provide a DID Object as an Argument.

NOTE 1 Use of the getObjects DIBO prevents DIP engines that, for example, present the User with a list of DID Objects, and then a list of DIMs that accept those Objects as Arguments, from identifying DIMs that operate on Objects that are not specified as the Arguments of a DIM in the DIM declaration.

The objectTypes parameter is an array of string values that specifies the Object Type for each element that the User will be requested to select. The Object Type is an Object Type that is found in the Object Map for the current DID. It is an error to invoke this DIBO if the objectTypes parameter is not an array of string values, in which case an invalid parameter exception is generated.

For each Object Type specified in the objectTypes parameter, the DIBO allows the User to select one element that maps to that Object Type.

The requestMessages parameter is an array of string values that contains a request message for each of the element selections indicated by the members of the objectTypes array. The requestMessages parameter may be null, in which case no explicit request messages are specified. If the requestMessages parameter is not null, it is an error if the number of members of the requestMessages parameter is not equal to the number of messages in the objectTypes parameter. However one or more of the members of the requestMessages parameter may be null. It is an error if a member of the requestMessages parameter is not null and is not a string value.

The selected elements of the specified Object Types are available to the invoking DIM via the returned array of Element objects. These can then be used as parameters to other DIBOs.

NOTE The presentation and selection of the elements is a DIBO implementation decision. For example, one DIBO implementation might present the element selection in a GUI to a human allowing the human to select the elements. Another implementation might be intended to process a DI without human intervention and implement some form of automated element selection.

GETVALUES
### INTERFACE

**Syntax:**    
getValues(dataTypes, requestMessages)

**Description:**  
Requests the User for input data of one of the primitive data types.

**Parameters:**
- dataTypes
  
  An Array of string values specifying the data type of each datum.

- requestMessages

  An array whose elements are either string values or null.

**Return value:**

Returns an array whose elements are values of type boolean, string or number, corresponding to the data type specified in the dataTypes array.

**Exceptions:**

DIPError  
With DIP error code INVALID_PARAMETER if

- dataTypes contains an invalid data type (not one of Boolean, String, or Number); or

- requestMessages is not null and is not an array the same size as objectTypes with each array member being a string value or null.

### SEMANTICS

This DIBO requests the User to enter data values of one of the primitive DIML data types. The DataTypes parameter is an array of string values that specifies primitive data type for each data value that the User will be requested to enter. The data type shall be one of: Boolean, String, or Number. It is an error to specify an invalid data type, in which case an invalid parameter exception is generated.

For each data type specified in the DataTypes parameter, the DIBO allows the User to enter a data value of that data type. The DIBO implementation should execute some basic validation on the value entered, at least to ensure the value is of the correct data type. It is an error to invoke this DIBO if the DataTypes parameter is not an array of string values, in which case an invalid parameter exception is generated.

The requestMessages parameter is an array of string values that contains a request message for each of the data values indicated by the members of the DataTypes array. The requestMessages parameter may be null, in which case no explicit request messages are specified. If the requestMessages parameter is not null, it is an error if the number of members of the requestMessages parameter is not equal to the number of messages in the DataTypes parameter. However one or more of the members of the requestMessages parameter may be null. It is an error if a member of the requestMessages parameter is not null and is not a string value.

The User entered data values of the specified data types are available to the invoking DIM via the returned array of values. If a value was not entered for a requested datum then the corresponding entry in the returned array of values will be null. If no data values at all were entered then a null is returned by the DIBO.

**NOTE**  
The data entry mechanism for the requested data is a DIBO implementation decision. For example, one DIBO implementation might present a data entry GUI to a human allowing the human to enter the values. Another implementation might be intended to process a DI without human intervention and implement some form of automated data entry.

### PLAY

**INTERFACE**

**Syntax:**    
play(element, async)

**Description:**  
Plays a specified COMPONENT, or DESCRIPTOR and optionally waits for completion before returning control to the calling DIM.

**Parameters:**
- element
A DOM Element object representing the COMPONENT, or DESCRIPTOR element to be played.

**async**
A boolean value indicating if the COMPONENT, or DESCRIPTOR should be played asynchronously or not. If true then the element is played asynchronously and the DIBO should return control immediately to the calling DIM after playing of the element is initiated. If false then the element is played synchronously and control is not returned to the calling DIM until the element media end time has been reached (if applicable).

**Return value:** Returns a PlayStatus object to identify the playing element. This is included so that the playing instance of the elements can be released at a later time.

**Exceptions:**
- DIPError
  - WITH DIP error code
  - INVALID_PARAMETER if
    - element is not a COMPONENT, or DESCRIPTOR; or
  - async is not a boolean value; or
  - PLAYBACK FAILED if an error occurs during playback.

### SEMANTICS
This DIBO causes the DIDL element represented by the element parameter to be rendered into a transient and directly perceivable representation.

The element parameter shall be a DOM Element object representing a COMPONENT or DESCRIPTOR to be played. It is an error to invoke this DIBO if the element parameter is not a DOM Element object representing a COMPONENT or DESCRIPTOR, in which case an invalid parameter exception is generated.

The manner of playing the element, appropriate to its content, is left as an implementation choice of the DIBO implementer.

The async parameter shall be a boolean value indicating whether the element is to be played asynchronously or synchronously. It is an error to invoke this DIBO if the async parameter is not a boolean value, in which case an invalid parameter exception is generated.

For an asynchronous playing, once playing of the element is initiated, the DIBO should return execution control to the invoking DIM.

For synchronous playing, the DIBO should not return execution control to the invoking DIM until the play status transitions to STATICPLAY or RELEASED.

For time based media the play status on successful playing of the element will be TIMEPLAY. When the media end time is reached, the status will transition to STATICPLAY.

For non time based media the play status on successful playing of the element will be STATICPLAY.

If the element could not be successfully played the play status will be RELEASED.

The DIBO returns a PlayStatus object which provides a handle to the playing instance of the element and its status. This handle is used as a parameter to other DIBOs to control the status of the playing instance of the element.

**NOTE 1** Time based media is media that changes over time, such as animations, audio clips, and video clips. The MIME media type of resources associated with the element can be retrieved from the mimeType attribute of any DIDL Resource elements. In many cases this will allow the DIBO implementation to determine whether the resource is time based or not. For example, resources with a top-level media type of audio or video (such as audio/mpeg for MP3) are generally time based.

**NOTE 2** For a synchronous call to the play DIBO for time based media, the play status transition through the TIMEPLAY state is effectively hidden from the invoking DIM, since the play DIBO will not return until the play status has transitioned to STATICPLAY.

**NOTE 3** It is the responsibility of the DIBO implementation to locate the appropriate technology for playing of the element and to inform the User if this cannot be found. For example the resource could be played internally or a mechanism to locate and use some external 'third party' software to play the resource could be provided. The MIME
media type of resources associated with the element can be retrieved from the MIMEType attribute of any DIDL Resource elements. This will enable the DIBO implementation to determine how to play the associated resources.

NOTE 4 While the manner of playing the element is a DIBO implementation choice, it could be guided by additional available information, for example information contained within descriptors associated with the element to be played.

## RELEASE

### INTERFACE

<table>
<thead>
<tr>
<th>Syntax</th>
<th>release(playStatus)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Stop playback of a COMPONENT or DESCRIPTOR and release related playback state information.</td>
</tr>
<tr>
<td>Parameters</td>
<td>playStatus</td>
</tr>
<tr>
<td></td>
<td>The PlayStatus object that was generated as a return value when playback of the DIDL element began.</td>
</tr>
<tr>
<td>EXAMPLE</td>
<td>As the result of an asynchronous call to the play DIBO.</td>
</tr>
<tr>
<td>Return value</td>
<td>None.</td>
</tr>
<tr>
<td>Exceptions</td>
<td>None.</td>
</tr>
</tbody>
</table>

### SEMANTICS

This DIBO causes playing of the DIDL element associated with the specified playStatus to be stopped and any state information to be released. The playStatus parameter shall be an object of type PlayStatus that was returned by a call to the play DIBO to play the associated element which is to be stopped. After calling this DIBO the play status will transition to RELEASED. If the current status is already RELEASED, then this DIBO does nothing.

### EXAMPLE USE

EXAMPLE In some cases a DIM might play a component asynchronously, do some other operations, and then release the component before exiting. For example, if a Digital Item represents an electronic travel brochure, and an Item contains several components that contain information about a specific destination, a DIM could be authored such that it starts playing some background audio asynchronously, displays synchronously the information components, then prior to exiting, releases the asynchronously playing component containing the audio resource.

## RUNDIM

### INTERFACE

<table>
<thead>
<tr>
<th>Syntax</th>
<th>runDIM(itemIdType, itemId, componentIdType, componentId, arguments)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Runs a DIM declared in an identified COMPONENT.</td>
</tr>
<tr>
<td>Parameters</td>
<td>itemIdType</td>
</tr>
<tr>
<td></td>
<td>A string value indicating the type of identifier (DII Identifier or URI) that is given by the itemId parameter.</td>
</tr>
<tr>
<td></td>
<td>Valid values are dii to indicate a DII Identifier, or uri to inciate a URI.</td>
</tr>
</tbody>
</table>
itemId
A string value identifying the ITEM that contains the DIM declaration of the DIM to be run or null.

componentIdType
A string value indicating the type of identifier (DII Identifier or URI) that is given by the componentId parameter.
Valid values are dii to indicate a DII Identifier, or uri to indicate a URI.

componentId
A string value identifying the COMPONENT that contains the DIM declaration of the DIM to be run or null.

arguments
An array of zero or more objects that are to be the arguments to be passed on to the invoked DIM.

Return value: None.

Exceptions: DIPError
With DIP error code
   INVALID_PARAM if
   either the itemIdType or componentIdType parameters do not specify a valid value;
   both the itemId and componentId parameters are null;
   the itemId or componentId are not string values or null values;
   either the itemId parameter, if not null, identifies an element that is not an ITEM, or the componentId parameter, if not null, identifies an element that is not a COMPONENT;
   both an ITEM is identified and a COMPONENT is identified, but the COMPONENT is not a child of the ITEM; or
   if the arguments array does not contain objects of the required Argument Types for this DIM; or
   NOT_FOUND if, given an identified ITEM and/or COMPONENT, a DIM to run cannot be determined; or
   GENERAL_EXCEPTION if the DIM cannot be run for any other reason.

SEMANTICS
This DIBO allows a DIM to be invoked from within another DIM.
Without the runDIM DIBO it would not be possible to call one DIM from within another DIM. The runDIM DIBO provides the DIM author with this capability.
EXAMPLE
An example where this capability could be used, is the calling of a PlayTrack DIM from a PlayTracks DIM. The runDIM DIBO can then also be used from other DIMs that require playing a track.
The itemIdType parameter shall contain a value of either dii or uri to indicate the type of identifier given by the itemId parameter. A value of dii indicates the itemId parameter contains the value of a DII Identifier. A value of uri indicates the itemId parameter contains the value of a URI.
The itemId parameter shall contain a value that identifies the ITEM in which the DIM to be invoked is declared. If the value of the itemIdType parameter is dii, then the value of the itemId parameter shall match the value contained in a DII Identifier identifying the ITEM. If the value of the itemIdType parameter is uri, then the value of the itemId parameter shall be a URI identifying the ITEM.
The DIM to be invoked shall be directly declared within the identified ITEM, not in a sub-ITEM. If the DIM to be invoked is declared in a sub-ITEM, then that sub-ITEM should be the identified ITEM.
If the itemId parameter is null, then only the componentId is used.
The componentIdType parameter shall contain a value of either dii or uri to indicate the type of identifier given by the componentId parameter. A value of dii indicates the componentId parameter contains the value of a DII Identifier. A value of uri indicates the componentId parameter contains the value of a URI.
The **componentId** parameter shall contain a value that identifies the COMPONENT in which the DIM to be invoked is declared. If the value of the **componentIdType** parameter is dii, then the value of the **componentId** parameter shall match the value contained in a DII Identifier identifying the COMPONENT. If the value of the **componentIdType** parameter is uri, then the value of the **componentId** parameter shall be a URI identifying the COMPONENT. If both the **itemId** and the **componentId** parameters are not null, then the identified COMPONENT shall be a child of the identified ITEM, and the DIM declared in the identified COMPONENT shall be run.

If the **itemId** parameter is null and the **componentId** parameter is not null, then the DIM declared in the identified COMPONENT shall be run.

If the **itemId** parameter is not null and the **componentId** parameter is null, then the DIBO implementation may choose any DIM declared directly in the identified ITEM (i.e., in a COMPONENT that is a child of the ITEM).

If both the **itemId** and the **componentId** parameters are null, then a DIPError exception is generated.

If the DIM to be invoked requires arguments, then the **arguments** parameter is an array of objects containing the arguments required by the DIM.

If the DIM cannot be invoked or an unhandled exception occurs during execution of the DIM, then a DIPError exception is generated.

### EXAMPLE USE

**EXAMPLE** An example where this DIBO can be used is when a DI author would like to have a message, included as a RESOURCE in the DI, displayed prior to the playing of any one of a number of other RESOURCES. The display of the message might be determined by certain conditions. A DIM to check the conditions and if required display the message can be authored. This DIM can then be invoked by the runDIM DIBO from other DIMs prior to playing any other RESOURCES for which the message might be required.

### INTERFACE

<table>
<thead>
<tr>
<th>Syntax:</th>
<th>wait(timeInterval)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description:</td>
<td>Waits for a given length of time.</td>
</tr>
<tr>
<td>Parameters:</td>
<td>timeInterval</td>
</tr>
<tr>
<td></td>
<td>A number value indicating the time in milliseconds for the Wait operation to pause execution of the DIM.</td>
</tr>
<tr>
<td>Return value:</td>
<td>None.</td>
</tr>
<tr>
<td>Exceptions:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DIPError</td>
</tr>
<tr>
<td></td>
<td>With DIP error code INVALID_PARAM</td>
</tr>
<tr>
<td></td>
<td>— if timeInterval is not a positive number value</td>
</tr>
</tbody>
</table>

### SEMANTICS

This DIBO causes a pause in execution of the invoking DIM. On invocation of the DIBO, execution control is not returned to the DIM until the specified time interval has elapsed. It is an error to invoke this DIBO if the **waitInterval** parameter is not a positive number value, in which case an invalid parameter exception is generated.

The pause applies only to the execution of the invoking DIM.

The DIBO implementer should ensure the actual elapsed interval is as close as possible to the specified time interval, but is not obliged to ensure the exact specified time interval.

### EXAMPLE USAGE

**EXAMPLE** An example where this DIBO can be used is when several resources are played asynchronously, and then the author wants to pause the execution of the DIM for an interval, before stopping one or more of the resources. For example, a Digital Item representing a travel brochure might contain an audio resource, a video resource, and an HTML resource representing a “brochure” of a travel destination. A DIM can be authored such that it...
plays all three resources asynchronously, and then waits for an interval of time, after which the audio and video resources are released.

**LOCAL OBJECTS, DIBOS AND CONSTANTS**

The following additional supporting object types are also normatively included in the DIML specification. The DIML object types are specified by the object properties (including primitive values, other objects, and functions), and attributes of those properties.

**DIPERROR**

This DIML object inherits from the ECMAScript Error object and is thrown as an exception whenever a runtime error specific to DIP occurs. In addition to the properties inherited from the Error object it has the properties specified below.

Other exceptions may also be thrown during execution of a DIM. These include ECMAScript native errors and exceptions derived from DIPError that are specified in this or other parts of ISO/IEC 21000.

**EXAMPLE**

A DIML syntax error will generate an ECMAScript native SyntaxError exception.

<table>
<thead>
<tr>
<th>Value Properties:</th>
<th>GENERAL_EXCEPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A number value of 1.</td>
</tr>
<tr>
<td></td>
<td>This indicates a general DIP error not covered by any other error code has occurred.</td>
</tr>
<tr>
<td></td>
<td>This property has the attributes { ReadOnly, DontEnum, DontDelete }.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value Properties:</th>
<th>INVALID_PARAM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A number value of 2.</td>
</tr>
<tr>
<td></td>
<td>This indicates a parameter passed to a DIBO is invalid.</td>
</tr>
<tr>
<td></td>
<td>EXAMPLE A DIPError returning this value from the getDIPErrorCode() DIBO will be generated if the element parameter passed in a call to the Play DIBO is not an Element object representing a DIDL ITEM, COMPONENT, or DESCRIPTOR element.</td>
</tr>
<tr>
<td></td>
<td>This property has the attributes { ReadOnly, DontEnum, DontDelete }.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value Properties:</th>
<th>INVALID_PERMISSION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A number value of 3</td>
</tr>
<tr>
<td></td>
<td>This indicates that an attempt was made to execute an operation for which required permission in the host environment is unavailable.</td>
</tr>
<tr>
<td></td>
<td>EXAMPLE If the host environment does not allow execution of resources in Digital Item, a DIPError returning this value from the getDIPErrorCode() DIBO will be generated if the Execute DIBO is called.</td>
</tr>
<tr>
<td></td>
<td>This property has the attributes { ReadOnly, DontEnum, DontDelete }.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value Properties:</th>
<th>NOT_FOUND</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A number value of 4</td>
</tr>
<tr>
<td></td>
<td>This indicates something required to complete an operation was not found. The missing entity is dependent on the operation that was attempted.</td>
</tr>
<tr>
<td></td>
<td>EXAMPLE A DIPError returning this value from the getDIPErrorCode() DIBO will be generated if a DIM with the required name is not found in the identified Item when the RunDIM DIBO is called.</td>
</tr>
<tr>
<td></td>
<td>This property has the attributes { ReadOnly, DontEnum, DontDelete }.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value Properties:</th>
<th>ADAPTATION_FAILED</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A number value of 5.</td>
</tr>
<tr>
<td></td>
<td>This indicates an error occurred during an attempt to adapt a resource.</td>
</tr>
<tr>
<td></td>
<td>This property has the attributes { ReadOnly, DontEnum, DontDelete }.</td>
</tr>
</tbody>
</table>
### OBJECTMAP

This DIML object represents the DIP Object Map. It has the properties specified below.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value Properties:</td>
<td>None</td>
</tr>
<tr>
<td>Object Properties:</td>
<td>None.</td>
</tr>
<tr>
<td>DIBO Properties:</td>
<td>getArgumentList(index)</td>
</tr>
<tr>
<td></td>
<td>This DIBO returns an array of string values representing the list of Argument Types.</td>
</tr>
<tr>
<td></td>
<td>The index parameter is a number value that indicates the index of the Argument list in the Object Map. The order of the Argument lists in the Object Map corresponds to the document order of the lists of DIP Argument children of all MethodInfo descriptors in the DIDL document and with subsequent instances of duplicate lists removed.</td>
</tr>
<tr>
<td></td>
<td>This DIBO generates a DIPError exception with error code INVALID_PARAM if the index parameter is not a number value; or not a valid index (a valid index is a number ranging from 0 to one less than the value returned from getArgumentListCount()).</td>
</tr>
<tr>
<td></td>
<td>getArgumentListCount()</td>
</tr>
<tr>
<td></td>
<td>This DIBO returns a number value and is the number of Argument lists with Arguments in a specific order that are defined in the Object Map. This corresponds to the number of unique permutations for all lists of DIP Argument children of all MethodInfo descriptors in the DIDL document.</td>
</tr>
<tr>
<td></td>
<td>getMethodInfo(argumentList)</td>
</tr>
<tr>
<td></td>
<td>This DIBO returns a number value and is the number of DIMs that are defined to accept as parameters the Arguments listed in argumentList. This corresponds to the number of DIM declarations in the DIDL document that have a DIP MethodInfo descriptor with zero or more child Argument elements such that</td>
</tr>
</tbody>
</table>
the number and values of those Argument elements match the number and names of the specified Argument Types.

The argumentList parameter is an array of string values that indicates the Argument Types. A zero length array passed as the argumentList parameter indicates DIMs declared to accept no Arguments.

This DIBO generates a DIPError exception with error code INVALID_PARAM if the argumentList parameter is not an array of string values.

<table>
<thead>
<tr>
<th>getMethodsWithArgs(argumentList)</th>
</tr>
</thead>
</table>
| This DIBO returns an array of Element objects representing the DIDL COMPONENT elements representing the DIM declaration for DIMs that accept as parameters the Arguments listed in argumentList. This corresponds to the list of DIDL Component elements, in document order, in the DIDL document that have a DIP MethodInfo descriptor with zero or more child Argument elements such that the number and values of those Argument elements match the number and names of the specified Argument Types.
| The argumentList parameter is an array of string values that indicates the Argument Types. A zero length array passed as the argumentList parameter indicates DIMs declared to accept no arguments.
| This DIBO generates a DIPError exception with error code INVALID_PARAM if the argumentList parameter is not an array of string values.

<table>
<thead>
<tr>
<th>getMethodWithArgs(argumentList, index)</th>
</tr>
</thead>
</table>
| This DIBO returns an Element object representing the DIDL COMPONENT element representing the DIM declaration for a DIM that accepts as parameters the Arguments listed in argumentList.
| The argumentList parameter is an array of string values that indicates the Argument Types. A zero length array passed as the argumentList parameter indicates DIMs that are declared to accept no Arguments.
| The index parameter is a number value that indicates the index of the DIM in the list of DIMs that accept as parameters the Arguments listed in argumentNames. The order of the DIMs in the list of DIMs corresponds to the document order of the DIM declarations in the DIDL document that have a DIP MethodInfo descriptor with zero or more child Argument elements such that the number and values of those Argument elements match the number and names of the specified Argument Types.
| This DIBO generates a DIPError exception with error code INVALID_PARAM if the argumentList parameter is not an array of string values; the index parameter is not a number value; or the index parameter is not a valid index (a valid index is a number ranging from 0 to one less than the value returned from getMethodCount(argumentList)).

<table>
<thead>
<tr>
<th>getObjectOfType(typeName, index)</th>
</tr>
</thead>
</table>
| This DIBO returns an Element object of a given Object Type.
| The typeName parameter is a string value that indicates the Object Type name.
| The index parameter is a number value that indicates the index of the Object in the list of Objects of the given Object Type. The order of the Objects in the list of Objects corresponds to the document order of the elements in the DIDL document that have a child descriptor containing a DIP ObjectType descriptor with a value matching the specified Object Type.
| This DIBO generates a DIPError exception with error code INVALID_PARAM if the typeName parameter is not a string value;
the `index` parameter is not a number value; or
the `index` parameter is not a valid index (a valid index is a number ranging from
0 to one less than the value returned from
`getObjectsOfTypeCount(typeName)`).

`getObjectsOfType(typeName)`

This DIBO returns an array of Element objects of a given Object Type. This
corresponds to the list of elements, in document order, in the DIDL document
that have a child DIP `ObjectType` descriptor with value matching the specified
Object Type. If there are no objects of the given Object Type, a zero length array
shall be returned.

The `typeName` parameter is a string value that indicates the Object Type name.
This DIBO generates a DIPError exception with error code INVALID_PARAM if
the `typeName` parameter is not a string value.

`getObjectsOfTypeCount(typeName)`

This DIBO returns a number value and is the number of Objects that are defined
in the Object Map to be of a given Object Type. This corresponds to the number
of elements in the DIDL document that have a child DIP `ObjectType` descriptor
with value matching the specified Object Type.

The `typeName` parameter is a string that indicates the Object Type name.
This DIBO generates a DIPError exception with error code INVALID_PARAM if
the `typeName` parameter is not a string value.

`getObjectTypeCount()`

This DIBO returns a number value and is the number of Object Types that are
defined in the Object. This corresponds to the number of unique values for
all DIP `ObjectType` descriptors in the DIDL document.

`getObjectTypeName(index)`

This DIBO returns a string value representing the Object Type name.

The index parameter is a number value that indicates the index of the Object
Type in the Object Map. The order of the Object Types in the Object Map
corresponds to the document order of the DIP `ObjectType` descriptors in the
DIDL document and with subsequent instances of duplicate values removed
from the list.

This DIBO generates a DIPError exception with error code INVALID_PARAM if
the index parameter is not a number value; or
not a valid index for an Object Type in the Object Map (a valid index is a number
ranging from 0 to one less than the value returned from
`getObjectTypeCount()`).

### PLAYSTATUS

This object is returned by the Play DIBO to enable subsequent operations on the particular instance of the Act
invoked. It has the properties specified below.

<table>
<thead>
<tr>
<th>Value Properties</th>
<th>RELEASED</th>
</tr>
</thead>
<tbody>
<tr>
<td>A number value of 0.</td>
<td></td>
</tr>
</tbody>
</table>

This value indicates the associated resource is not currently playing.

NOTE A resource can be in the RELEASED state because an error
prevented a request to play the resource from successfully completing, or as a
result of an explicit request to release the resource.

A RELEASED resource has no other preserved state information. Playing a
RELEASED resource will commence with relevant state information in an initial
state.

EXAMPLE For a resource with time based media, playing a RELEASED
resource will commence from the media start time.
This property has the attributes {ReadOnly, DontEnum, DontDelete}.

<table>
<thead>
<tr>
<th>Section: AXMEDIS Behaviour Editor</th>
<th>Resource will commence from the media start time. This property has the attributes {ReadOnly, DontEnum, DontDelete}.</th>
</tr>
</thead>
<tbody>
<tr>
<td>STATICPLAY</td>
<td>A number value of 1. This value indicates the associated resource is currently playing. Time-based state information related to playing the resource, if relevant, is paused for a STATICPLAY resource. NOTE: For time-based media, this status is equivalent to a state typically considered as 'paused'. This property has the attributes {ReadOnly, DontEnum, DontDelete}.</td>
</tr>
<tr>
<td>TIMEPLAY</td>
<td>A number value of 2. This value indicates the associated resource is currently playing. Time-based state information related to playing the resource, if relevant, is advancing for a TIMEPLAY resource. NOTE: Only time-based media can have a status of TIMEPLAY. This property has the attributes {ReadOnly, DontEnum, DontDelete}.</td>
</tr>
</tbody>
</table>

Object Properties: None.

DIBO Properties: getStatus()
This DIBO returns a number value indicating the status of the resource associated with the PlayStatus object. The returned value is one of the value properties specified above for the PlayStatus prototype.

### TABLE OF AVAILABLE MPEG21 DIP FUNCTIONALITIES

<table>
<thead>
<tr>
<th>Object</th>
<th>DIBO Function</th>
<th>Implemented/not implemented</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIA</td>
<td>adapt(element, metadata)</td>
<td>no</td>
</tr>
<tr>
<td>DID</td>
<td>areConditionsSatisfied(element)</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>configureChoice(choice)</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>setSelection(selection, state)</td>
<td>no</td>
</tr>
<tr>
<td>DII</td>
<td>getElementsByIdentifier(sourceDID, value)</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>getElementsByRelatedIdentifier(sourceDID, value)</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>getElementsByType(sourceDID, value)</td>
<td>no</td>
</tr>
<tr>
<td>DIP</td>
<td>alert(message, messageType)</td>
<td>implemented</td>
</tr>
<tr>
<td></td>
<td>execute(element)</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>getExternalData(mimeTypes, requestMessages)</td>
<td>implemented</td>
</tr>
<tr>
<td></td>
<td>getObjectMap(document)</td>
<td>implemented</td>
</tr>
<tr>
<td></td>
<td>play(element, async)</td>
<td>implemented</td>
</tr>
<tr>
<td></td>
<td>print(element)</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>release(playStatus)</td>
<td>implemented</td>
</tr>
<tr>
<td></td>
<td>runDIM(itemIdType, itemId, componentIdType)</td>
<td>implemented</td>
</tr>
<tr>
<td>AXMEDIS Editor User Manual</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>componentId,arguments)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>runJDIXO(itemIdType,itemId,componentIdType, componentId,className,arguments) no</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wait(timeInterval) implemented</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>DOM Level 3 Core Specification</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getElementsByTagNameNS(in DOMString namespaceURI, in DOMString localName) implemented</td>
<td></td>
<td></td>
</tr>
<tr>
<td>getElementsByTagName(in DOMString name) implemented</td>
<td></td>
<td></td>
</tr>
<tr>
<td>getAttributeNS(in DOMString namespaceURI, in DOMString localName) implemented</td>
<td></td>
<td></td>
</tr>
<tr>
<td>getAttribute(in DOMString name) implemented</td>
<td></td>
<td></td>
</tr>
<tr>
<td>getElementById(in DOMString elementId) implemented</td>
<td></td>
<td></td>
</tr>
<tr>
<td>hasChildNodes() implemented</td>
<td></td>
<td></td>
</tr>
<tr>
<td>getAttributes() implemented</td>
<td></td>
<td></td>
</tr>
<tr>
<td>getDocumentElement() implemented</td>
<td></td>
<td></td>
</tr>
<tr>
<td>getTagName() implemented</td>
<td></td>
<td></td>
</tr>
<tr>
<td>getChild() implemented</td>
<td></td>
<td></td>
</tr>
<tr>
<td>getChildNodes() implemented</td>
<td></td>
<td></td>
</tr>
<tr>
<td>getFirstChild() implemented</td>
<td></td>
<td></td>
</tr>
<tr>
<td>getLastChild() implemented</td>
<td></td>
<td></td>
</tr>
<tr>
<td>getParentNode() implemented</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>DOM Level 3 Load and Save Specification</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getLicense(resource) no</td>
<td></td>
<td></td>
</tr>
<tr>
<td>queryLicenseAuthorization(license,resource, rightNs,rightLocal,addionalInfo) no</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>List of ECMAScript bindings for DIBOs of local objects</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>object</th>
<th>DIBO</th>
<th>implemented/not implemented</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIPError</td>
<td>getDIPErrorCode()</td>
<td>implemented</td>
</tr>
<tr>
<td>ObjectMap</td>
<td>getArgumentList(index)</td>
<td>implemented</td>
</tr>
<tr>
<td></td>
<td>getArgumentListCount()</td>
<td>implemented</td>
</tr>
<tr>
<td></td>
<td>getMethodCount(argumentNames)</td>
<td>implemented</td>
</tr>
<tr>
<td></td>
<td>getMethodWithArgs(argumentNames,index)</td>
<td>implemented</td>
</tr>
<tr>
<td></td>
<td>getMethodsWithArgs(argumentNames)</td>
<td>implemented</td>
</tr>
<tr>
<td></td>
<td>getObjectType(typeName, index)</td>
<td>implemented</td>
</tr>
<tr>
<td></td>
<td>getObjectsOfType(typeName)</td>
<td>implemented</td>
</tr>
<tr>
<td></td>
<td>getObjectsOfTypeCount(typeName)</td>
<td>implemented</td>
</tr>
<tr>
<td></td>
<td>getObjectsOfTypeCount(typeName)</td>
<td>implemented</td>
</tr>
<tr>
<td></td>
<td>getObjectType(index)</td>
<td>implemented</td>
</tr>
<tr>
<td></td>
<td>getObjectTypeCount()</td>
<td>implemented</td>
</tr>
<tr>
<td>PlayStatus</td>
<td>getStatus()</td>
<td>implemented</td>
</tr>
</tbody>
</table>
AXMEDIS DRM EDITOR

MAIN FUNCTIONALITIES

“DRM Editor and Viewer” is a software application to view and eventually edit MPEG-21 REL Licenses. This user manual refers to its version 2.6. “DRM Editor and Viewer” comes in the form of two different applications. Its main features are:

<table>
<thead>
<tr>
<th>Feature</th>
<th>DRM editor</th>
<th>DRM viewer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load a license from a XML file</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Load license from a remote host (PMS Server)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Create a new license from scratch</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Display graphically the license</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Store a license in a XML file</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Store a license in a remote host (PMS Server)</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Edit (modify) a License</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

RELATIONSHIP WITH OTHER TOOLS

This tool can be embedded as a component in other tools (i.e. axeditor). To start the DRM Editor and Viewer Standalone Application, there are two possible ways.

- From the PAR Editor, embedded in the Axeditor application, it is possible going to the menu and selection “Launch Standalone DRM Editor and Viewer”
- From the AXMEDIS Tools package, select the DRM Editor and Viewer application

The licenses created with the AXMEDIS DRM Editor can be stored locally or in a remote server (AXMEDIS PMS Server). In order to be able to store licenses in the remote server, a connection to it must be available, as explained in next section.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

The application is a window application. (see figure DRM Editor View). As in a common window application, the following elements appear: menu bar, button bar, status bar. The status bar shows an icon informing whether connection to the server is available or not. If it does not exist, only file operations are possible

The main window shows two areas. In the left a hierarchically structure is displayed (also called tree), which is used to navigate through the license elements; in the right side there is a panel showing information related to the element marked in the tree. Functionalities can be accessed through the menu options, the buttons in the button bar, or the buttons that appear in the panel. Next we summarise the DRM Editor and Viewer main functionalities:

- DRM Editor and Viewer. Open a license from a file (see next figure). The panel displays the file path or an alternative description.
• DRM Editor. Create a new license. When a new license is created, it is empty. The tree appears with an “issuer” element and with a “Grant Group” element, but they are initially empty.
• DRM Editor. Modify a license.
  o When the tree is clicked on its “issuer element”, the Editor shows the issuer panel, where the issuer and the date of issuance can be set (see next figure).

  o When the tree is clicked on its “GrantGroup element”, a Grant Group panel is displayed. (see next figure). From this panel it is possible to add or delete grants. The first icon creates an empty grant, the second deletes it. Once a grant has been created, it will be displayed in the tree.
When the tree is clicked on its “Grant” element (see next figure), the Grant panel is shown. From this panel, the following actions can be performed:

- To change the following fields: principal, right, resource.
  - The resource is a free text, that should point to a Axmedis Object ID
  - The right is one of the string selectable in the combo-box
  - The principal is a free text that should point to an Axmedis User or an Axmedis Domain. The Principal is the user who will receive the rights granted in a license. The Principal appears as a text field that for each grant, and it can be filled in with the proper text describing the beneficiary of the grant. Usually, it will be an Axmedis User ID, but it can also be a Domain ID. In order to select a Domain, check the proper item besides Principal (see Figure below).
To add and remove conditions. The conditions that can be set are: number, interval, territory and/or fee. Each of them can be modified.

Figure: DRM Editor License Grant view

- When the issue right is chosen, the license becomes a “distributor license”, and the Grants underneath are Grants to be given by the distributor. This is reflected in the tree, where a new sublevel is displayed containing additional grants.
  - DRM Editor. Store a license to a file. This option is performed when selection the “File -> Save to File” option of the Editor.
  - DRM Editor. Store a license to a remote server.

Limitations to be fixed in subsequent versions:
  - In the current version only up to 8 grants can be added to the same grant group.
  - Only a condition of each type can be added to a grant.

If the DRM Editor and Viewer are used in its classic version (the one which displays a simple tree) these limitations are overcome.
AXMEDIS WORKFLOW EDITOR

MAIN FUNCTIONALITIES

The AXMEDIS workflow Editor/Viewer provides user with the workflow information for the object being edited/viewed. It also shows the Workflow Engine’s Interface for the user to see his worklist. The Workflow information that is displayed is as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>The</td>
</tr>
<tr>
<td>Process</td>
<td>The workflow process that is being executed for the selected AXOID</td>
</tr>
<tr>
<td>Activity</td>
<td>The Activity that is currently being executed for the selected AXOID</td>
</tr>
<tr>
<td>Priority</td>
<td>The priority of the Workflow Process</td>
</tr>
<tr>
<td>Status</td>
<td>The status of the Workflow Process</td>
</tr>
<tr>
<td>Actor</td>
<td>The Actor responsible for the current Activity</td>
</tr>
<tr>
<td>AXRQID</td>
<td>The Workflow Request ID issued by the workflow engine for current request.</td>
</tr>
</tbody>
</table>

RELATIONSHIP WITH OTHER TOOLS

This tool is embedded as a part of AxEditor. It uses the workflow editorPlugin to communicate with the Workflow Engine through Workflow Gateways (with OpenFlow). With Biztalk, it will communicate directly.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

In order to view the workflow information, the user should select the “Workflow View” tab in the Axeditor. On activation the editor will show empty fields for the parameters.

Meanwhile it will retrieve the “Work List” for the logged user from the workflow engine.
When the user clicks on the “Request Workflow Information” button, the editor retrieves the workflow parameters from the Workflow Engine and displays in the upper half of the viewer. If the information cannot be retrieve an error message is displayed (as showed in the picture below).

To configure appropriately the Workflow parameters it is necessary to select the menu **File/Configuration**.
AXMEDIS PROTECTION INFORMATION EDITOR

MAIN FUNCTIONALITIES
The Protection Information Editor and Viewer provides the functionalities to view and edit protection information. The main features are:
- The user can browse the protection information, the list of protection operations that were applied to the selected part of an AXMEDIS object.
- The user can view detailed information about a specific protection operation including all parameters and the protection target.
- The user can alter the order of different protection operations.
- The user can delete one of the protection operations from the list of protection steps.
- The user can select one of the available tools for protection, e.g., encryption, scrambling or compression, and add an additional protection operation to a specific part of an AXMEDIS object.

RELATIONSHIP WITH OTHER TOOLS
The Protection Information Editor and Viewer uses the Protection Processor to access protection information, to apply protection operations to a specific part of an AXMEDIS object.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS
Here’s an example on how to use the AXMEDIS Protection Information Editor and Viewer. A part of an object is selected in the tree view of the object shown on the left side of the editor. For editing and viewing of protection information the tab “Protection Information” must be selected on the right side of the editor window.

The current user interface for the protection of resources and the viewing of protection information is shown in the following picture.

![Figure. Protection Information Editor and Viewer user interface](image)

VIEWING OF PROTECTION INFORMATION
After the user selected a specific part of an object and switched to the protection view (as described above) a window appears which shows on the right side a list of the protections steps that were applied to this part of the object. When the user selects one of these operations more detailed information is shown in the lower right part of the window (“Protection Information Details”). The user can open an additional window by double clicking one of the protection operations. This window displays the name of the tool that was applied, a list of the different parameters and the target to which this protection operation was applied to (see figure below).
EDITING OF PROTECTION INFORMATION

When a user wants to change the protection information he has to select a part of an AXMEDIS and view the protection information as described above. The order of the different protection operations can be changed by selecting a specific operation and clicking on the up or down button on the right side.

If a user wants to add an additional protection step, he can select a protection tool from the list on the left side of the protection window and click on the green arrow in the middle to add it to the list of protection steps. A new window appears in which the specific parameters for this new protection operation, e.g. key length, can be edited and saved by the user.

A user can also edit existing protection operations by double clicking on an operation in the protection view window. A window appears that displays the different parameters for this protection operation which can be edited and saved by the user.
AXMEDIS PLUG-INS

The following sections will show in details the capabilities of the plug-ins available in the AXMEDIS Editor.

AUDIO ADAPTATION PLUG-IN

MAIN FUNCTIONALITIES

The audio_adaptation_plug-in allows adapting audio content to various use cases. For example, it can be used for transcoding applications to transform a high-quality audio file into a low bit rate audio file well suited for distribution on a network with reduced bandwidth. This document shows the main functionalities provided by a first prototype of the tool forming a minimal user guide.

RELATIONSHIP WITH OTHER TOOLS

This tool is implemented as a plug-in. Like other plug-ins, its functionality is available via the AXMEDIS Editor and the AXMEDIS Processing Engine.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

Descriptions of the functionalities available in the audio adaptation plug-in are showed in the next sections.

FFMPEG

Here’s an example on how to use the FFmpeg audio adaptation transcoding function as a plug-in 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:

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

Here follows a more complete description of the parameters of the FFmpeg audio transcoding function:

**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:

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:

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

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:

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:

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:

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:

**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:

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

---

**FILE FORMATS**
For a list of codecs and formats supported by FFmpeg, please see http://ffmpeg.mplayerhq.hu/ffmpeg-doc.html#SEC21

Mime type accepted:
- audio/x-wav
- audio/x-ms-wma
- audio/basic
- audio/x-mpeg
- audio/x-vorbis
- audio/x-pn-realaudio
- audio/x-ac3
- audio/x-dv
- audio/x-mace
- audio/x-adpcm
- audio/x-aac
- audio/32KADPCM
- audio/amr
- video/x-mpeg
- video/x-mpeg2
- video/mp4
- video/x-raw
- video/x-h263
Here’s an example on how to use the libsndfile audio adaptation transcoding function as a plug-in 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:

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

**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:**

**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:

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:

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

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:

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:

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 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**

Mime Type accepted:
- audio/x-wav
- audio/x-basic
- audio/x-paris
- audio/x-svx
- audio/x-nist
- audio/x-voc
- audio/x-URam
- audio/x-w64
- audio/x-sd2
- audio/x-flac
- application/x-pcm
- application/x-pagerecall

**TRICKS/ERRORS**

Some tricks for plug-in testing:
- Don’t use the same file for input and output (generates an unknown error, due to the editor ?)
- Don’t try to use vorbis related files with these actual FFMPEG dlls (.ogg)
- We endured problems with AMR and it could still fail.

If the reply is that this is Unknown output mime type, check the corresponding mime-type table.
If you don’t have to trunk the audio files don’t change ReadStartingTime and ReadEndingTime options. (the same caution may be used for other FFMPEG advanced options)
AUDIODESCRIPTOR PLUGIN

MAIN FUNCTIONALITIES
The audiodescriptorplugin aims at extracting automatically metadata from audio signals by audio signal analysis. The functionalities implemented include a segmentation algorithm, a music genre recognizer, a tempo detection algorithm plus a set of low level descriptors extraction algorithms. This document describes these functionalities forming a minimal user guide.

RELATIONSHIP WITH OTHER TOOLS
This tool is implemented as a plug-in. Like other plug-ins, its functionality is available via the AXMEDIS Editor and the AXMEDIS Processing Engine.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS
Here’s an example on how to use the plug-in with the AXMEDIS editor. The plug-in must be applied on an audio resource of an AXMEDIS object. The descriptor extraction plug-in is called by right-clicking on the interesting resource and selecting the ‘Plugin...’ command (see figure 1).

A window showing the functionalities available for the kind of resource selected appears (see figure 2). In the following parts, we discuss independently each available functionality.

LOW LEVEL DESCRIPTORS EXTRACTION
The Low Level Descriptors extraction algorithm allows extracting MPEG-7 compliant descriptors of the audio signal. Such descriptors are said to be “low-level” since they describe the shape and the properties of the audio signal but cannot be directly interpreted by humans (as opposed to “high-level” descriptors such as music genre or tempo). The Low-Level Descriptors extraction algorithm is launched by selecting the AudioDescriptor: LowLevelDescriptors function:
Figure 2 – Selecting the low level descriptors extraction function

A new window appears showing the interface to the low level descriptors extraction algorithm (see figure 3).

Figure 3 – The low level descriptors extraction function

The input audio file to be analysed is selected with first parameter. The last parameter allows specifying where the resulting MPEG-7 compliant description will be saved. The other parameters allow selecting which low level descriptors should be extracted. The analysis is launched by clicking on the **Execute** button. Once the analysis completed, one can display the resulting MPEG-7 description by double-clicking on the resource in which the description was saved (figure 4).
Here follows a more complete description of the parameters of the tempo estimation function:

- **InputResource**: the audio resource to be analysed; reading of audio resources is supported for the following mime types (corresponding to uncompressed audio formats):
  1. audio/x-aiff (.aiff, .aiff)
  2. audio/x-wav (.wav)
  3. audio/x-basic (.au, .snd)
- **HopSize**: the HopSize defines the temporal distance (in seconds) between two successive analyses (set to 10 ms by default).
- **AudioPower**: computes the AudioPower descriptor if set to 1. The AudioPower descriptor describes the temporally-smoothed instantaneous power of the audio signal. Instantaneous power is a useful measure of the amplitude of a signal as a function of time.
- **SpectralCentroid**: computes the SpectralCentroid descriptor if set to 1. The SpectralCentroid descriptor describes the centre of gravity of the log-frequency power spectrum. It is an economical description of the shape of the power spectrum. It indicates whether the power spectrum is dominated by low or high frequencies and, additionally, it is correlated with a major perceptual dimension of timbre, i.e. sharpness.
- **SpectralSpread**: computes the SpectralSpread descriptor if set to 1. The SpectralSpread descriptor describes the second moment of the log-frequency power spectrum. SpectralSpread is an economical descriptor of the shape of the power spectrum that indicates whether it is concentrated in the vicinity of its centroid, or else spread out over the spectrum. It allows differentiating between tone-like and noise-like sounds.
- **SpectralEnvelope**: computes the SpectralEnvelope descriptor if set to 1. The SpectralEnvelope descriptor describes the spectrum of the audio according to a logarithmic frequency scale. A logarithmic frequency axis is used to conciliate requirements of concision and descriptive power. Peripheral frequency analysis in the ear roughly follows a frequency axis.
- **EnvLoEdge**: set lower edge of logarithmically-spaced frequency bands for the extraction of the SpectralEnvelope descriptor (62.5 Hz by default).
- **EnvHiEdge**: set higher edge of logarithmically-spaced frequency bands for the extraction of the SpectralEnvelope descriptor (16000.0 Hz by default).
- **BandsPerOctave**: frequency resolution of logarithmic spectrum for the extraction of the SpectralEnvelope descriptor (width of each spectrum band between EndLoEdge and EnvHiEdge).
- **SpectralFlatness**: computes the SpectralFlatness descriptor if set to 1. The SpectralFlatness descriptor properties of the spectrum of an audio signal within a given number of frequency bands. This descriptor expresses the deviation of the signal’s power spectrum over frequency from a flat shape (corresponding to a noise-like or impulse-like signal). A high deviation from a flat shape may indicate the presence of tonal components.
- **FlatLoEdge**: set lower edge of logarithmically-spaced frequency bands for the extraction of the SpectralFlatness descriptor (250.0 Hz by default).
- **FlatHiEdge**: set higher edge of logarithmically-spaced frequency bands for the extraction of the SpectralFlatness descriptor (16000.0 Hz by default).
- **ScaleRatio**: the ScaleRatio is the number of original samples represented by each scaled sample when using a scaling operation (such as mean or variance of the descriptors)
- **EvalMeans**: computes the mean of the descriptors if set to 1.
- **EvalVariances**: computes the variance of the descriptors if set to 1.
- **OutputResource**: the MPEG-7 description of the audio resource.

### SEGMENTATION INTO SILENCE / SPEECH / NOISE / MUSIC

The Silence / Speech / Noise / Music segmentation algorithm allows segmenting the audio stream into 4 kind of semantically coherent segments:

- **Silence segment**: silence segments are defined as regions of the audio file in which no significant sound is heard.
- **Speech segment**: speech segments are defined as regions of the audio file in which spoken content is dominant.
- **Music segment**: music segments are defined as regions of the audio file in which music content is dominant.
- **Noise segment**: noise segments are defined as regions of the audio file in which noise is dominant; noise is loosely defined as audio content which is not speech, music nor silence.
- **The segmentation algorithm is called by selecting the AudioDescriptor: Segmentation function:**

![Figure 5 – Selecting the segmentation function](image-url)
A new window appears showing the interface to the segmentation function (see figure 6).

Figure 6 – The speech/noise/music segmentation function

The input audio file to be analysed is selected with first parameter while the second parameter allows specifying where the resulting MPEG-7 compliant description will be saved. The analysis is launched by clicking on the **Execute** button. Once the analysis completed, one can display the resulting MPEG-7 description by double-clicking on the resource in which the description was saved (figure 7).

![Figure 7 – The resulting MPEG-7 description](image)

Here follows a more complete description of the parameters of the segmentation function:

- **InputResource**: the audio resource to be analysed; reading of audio resources is supported for the following mime types (corresponding to uncompressed audio formats):
  1. audio/x-aiff (.aif, .aiff)
  2. audio/x-wav (.wav)
  3. audio/x-basic (.au, .snd)
- **OutputResource**: the MPEG-7 description of the audio resource.

### MUSIC GENRE RECOGNITION

The Music Genre recognizer allows characterizing music segments in terms of music genres. The provided model classifies music into one of the following categories:

- Classical
- Jazz
- Rap
Rock

The Music Genre recognizer is called by selecting the **AudioDescriptor: MusicGenreEstimation** function:

![Figure 8 – Selecting the music genre recognition function](image)

A new window appears showing the interface to the music genre recognizer (see figure 9).

![Figure 9 – The music genre recognition function](image)

The input audio file to be analysed is selected with first parameter while the second parameter allows specifying where the resulting MPEG-7 compliant description will be saved. The analysis is launched by clicking on the **Execute** button. Once the analysis completed, one can display the resulting MPEG-7 description by double-clicking on the resource in which the description was saved (figure 10).

![Figure 10 – The resulting MPEG-7 description](image)
Here follows a more complete description of the parameters of the music genre recognition function:

- **InputResource**: the audio resource to be analysed; reading of audio resources is supported for the following mime types (corresponding to uncompressed audio formats):
  4. audio/x-aiff (.aif, .aiff)
  5. audio/x-wav (.wav)
  6. audio/x-basic (.au, .snd)
- **OutputResource**: the MPEG-7 description of the audio resource.

### TEMPO DETECTION

The tempo detection algorithm allows detecting the tempo in beats per minute of a music segment. It is launched by selecting the **AudioDescriptor: TempoEstimation** function:

![Figure 11 – Selecting the tempo estimation function](image1)

A new window appears showing the interface to the music genre recognizer (see figure 12).

![Figure 12 – The tempo estimation function](image2)

The input audio file to be analysed is selected with first parameter. The last parameter allows specifying where the resulting MPEG-7 compliant description will be saved. The parameters **BpmLoLimit** and **BpmHiLimit** allow to set boundaries to the estimation of tempo, i.e. the estimated tempo will fit in between these limits. The analysis is launched by clicking on the **Execute** button. Once the analysis completed, one can display the resulting MPEG-7 description by double-clicking on the resource in which the description was saved (figure 13).
Here follows a more complete description of the parameters of the tempo estimation function:

- **InputResource**: the audio resource to be analysed; reading of audio resources is supported for the following mime types (corresponding to uncompressed audio formats):
  1. audio/x-aiff (.aif, .aiff)
  2. audio/x-wav (.wav)
  3. audio/x-basic (.au, .snd)

- **BpmLoLimit**: the minimum acceptable tempo in beats per minute (BPM).

- **BpmHiLimit**: the maximum acceptable tempo in beats per minute (BPM).

- **OutputResource**: the MPEG-7 description of the audio resource.
MULTIMEDIA ADAPTATION PLUG-IN

MAIN FUNCTIONALITIES
The multimedia_adaptation_plug-in allows adapting multimedia content to various use cases. For example, it can be used to transcode an MP4 file into a 3GP file or to extract the media resources embedded into a complex multimedia file. The plug-in is composed by five functions that are: Extract Media Track,Mp4 To 3GP, Mp4 to Isma, Add Media files and Cat Media Files. This document shows the main functionalities provided by a first prototype of the tool forming a minimal user guide.

RELATIONSHIP WITH OTHER TOOLS
This tool is implemented as a plug-in. Like other plug-ins, its functionality is available via the AXMEDIS Editor and the AXMEDIS Processing Engine.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS
Here is an example on how to use the plug-in with the AXMEDIS editor.
The plug-in must be applied on an mp4 resource of an AXMEDIS object. The adaptation plug-in is called by right-clicking on the interesting resource and selecting the ‘Content processing plugins...’ command (see figure 1).

A window showing the functionalities available for the kind of resource selected appears (see figure 2).

The following section summarizes the multimedia adaptation functions.
**EXTRACT MEDIA TRACK**

This function extracts one track from the original source (without deleting this track) and leaves it into a separated new file. The supported output mime types are: video/x-cmp, video/x-msvideo, video/mp4 and audio/x-gsm.

This function is selected by clicking on MultimediaAdaptation: ExtractMediaTrack. After clicking “execute”, a new window appears showing the interface to the extraction function (see figure 3).

![Figure 3 – The Extract Media Track function](image)

Here follows a brief description of the parameters of the Extract Media Track function:

- **InputResource**: the multimedia resource where the track is going to be extracted (not deleted). Its use is only allowed for the video/mp4 (.mp4) files.
- **OutputResource**: the multimedia resource after the extraction, it is to say, the track already extracted into a new file. It can be part of all the allowed mpeg-4 compliant formats.
- **TrackID**: track to be extracted from the InputResource. If the track does not exist, the result will show “ERROR: Bad parameter”.
- **MimeType**: mime type of the OutputResource. Supports video/x-cmp, video/x-msvideo, video/mp4 and audio/x-gsm.

**MP4 TO 3GP**

This function will translate the input resource that is supposed to be .mp4 into a new file with the 3gp format.

This function is selected by clicking on MultimediaAdaptation: Mp4To3GP. After clicking “execute”, a new window appears showing the interface to the Mp4 to 3GP function (see figure 4).

![Figure 4 – The Mp4 to 3GP function](image)

Here follows a brief description of the parameters of the Mp4 to 3GP function:

- **InputResource**: The multimedia resource to be translated into 3gp. At first, it is only allowed to use an mp4 resource.
- **OutputResource**: The multimedia output that is obtained after the transformation performed by the function. The obtained file is an .3gp file.
- **KeepSys**: If it should keep system tracks within the translation.

**CAT MULTIMEDIA FILES**

This function concatenates two whole multimedia resources and gives a new file containing the result of the concatenation.

This function is selected by clicking on MultimediaAdaptation: CatMultimediaFiles. After clicking “execute”, a new window appears showing the interface to the concatenation function (see figure 5).
Here follows a brief description of the parameters of the Cat Media Files function:

- **InputResourceA**: It is one of the multimedia sources to concatenate. It will be the first in the timeline of the output file. By the moment, it is only allowed to introduce .mp4 files.
- **InputResourceB**: It is one of the multimedia sources to concatenate. It will be included after the InputResourceA into the new output resource. By the moment, it is only allowed to introduce .mp4 files.
- **OutputResource**: Is the result of the concatenation of InputResourceA and InputResourceB. The format of this file is .mp4.

**MP4 TO ISMA**

This function converts the input resource to the ISMA specification.
The function is selected by clicking on Multimedia Adaptation: Mp4ToISMA. After clicking “execute”, a new window appears showing the interface to the conversion function (see figure 6).

Here follows a brief description of the parameters of the Mp4 to ISMA function:

- **InputResource**: Mp4 file to be converted into the ISMA specification.
- **OutputResource**: Output file of the conversion to ISMA of the InputResource.

**ADD MULTIMEDIA FILES**

This function takes multimedia resources and adds them as new tracks into new or already existing mp4 file. It must specify the size (amount of seconds) of the multimedia resource that is imported and when should it begin inside the destination file, it is to say, the delay of the new track.
The function is selected by clicking on Multimedia Adaptation: AddMultimediaFiles. After clicking “execute”, a new window appears showing the interface to the Add Multimedia Files function (see figure 7).

Here follows a brief description of the parameters of the Add Media Files function:

- **InputResource**: File to be included into a new MP4 file.
- **BaseResource**: Base MP4 file where to add the InputResource
• **Delay**: Delay in milliseconds to be applied at the track to be included into the MP4 output file.
• **ImportLength**: Number of seconds to import from the input file starting from the beginning.
• **TrackID**: Track to extract in the file. If empty take the whole file.
• **FPS**: Frames per sample of the new track. 0 means source file FPS.
• **Lang**: Optional: Language code of the new track
• **OutputResource**: Output file where the track is included

### TO MP4

This function converts the input resource to Mp4.

The function is selected by clicking on MultimediaAdaptation: ToMp4. After clicking “execute”, a new window appears showing the interface to the conversion function (see figure 8).

![Figure 8 – The To Mp4 function](image)

Here follows a brief description of the parameters of the To Mp4 function:

- **InputResource**: File to be converted to Mp4.
- **OutputResource**: Output file of the conversion to Mp4 of the InputResource.

### DELAY TRACK

This function set the delay to a track from a mp4 file.

The function is selected by clicking on MultimediaAdaptation: DelayTrack. After clicking “execute”, a new window appears showing the interface to the Delay Track function (see figure 9).

![Figure 9 – The Delay Track function](image)

Here follows a brief description of the parameters of the Delay Track function:

- **InputResource**: Mp4 file where to delay a track.
- **Delay**: New delay in milliseconds applied at the track of the MP4 output file.
- **TrackID**: Track to be delayed in the file.
- **OutputResource**: Output Mp4 file where the track is included delayed.

### REMOVE TRACK

This function removes a track from a mp4 file.

The function is selected by clicking on MultimediaAdaptation: RemoveTrack. After clicking “execute”, a new window appears showing the interface to the Remove Track function (see figure 10).
Here follows a brief description of the parameters of the Remove Track function:

- **InputResource**: Mp4 file where to remove a track.
- **TrackID**: Track to be removed from the file.
- **OutputResource**: Output mp4 file where the track is removed.

**Extract From Start To End**

This function extracts a new mp4 file from a mp4 file by time limitation.
The function is selected by clicking on MultimediaAdaptation: ExtractFromStartToEnd. After clicking “execute”, a new window appears showing the interface to the Extract from Start to End function (see figure 11).

Here follows a brief description of the parameters of the Extract from Start to End function:

- **InputResource**: File where to extract the new mp4 file.
- **Start**: Start of extraction in seconds
- **End**: End of extraction in seconds
- **OutputResource**: Output mp4 file limited by time

**MP4 TO AVI**

This function will translate the input resource that is supposed to be .mp4 pure BIFS file into a new file with the avi format.
The function is selected by clicking on MultimediaAdaptation: Mp4ToAvi. After clicking “execute”, a new window appears showing the interface to the Mp4 to Avi function (see figure 12).

Here follows a brief description of the parameters of the Mp4 to Avi function:

- **InputResource**: The Mp4 pure BIFS file to be translated to Avi format.
- **FPS**: Extraction frame rate (0 compute from the BIFS track duration )
- **Width**: Width of the bifs scene (0 takes original size)
- **Height**: Height of the bifs scene (0 takes original size)
• **OutputResource**: The multimedia output that is obtained after the transformation performed by the function. The obtained file is an .avi file.
**DOCUMENT CRYPTLIB PLUG-IN**

**MAIN FUNCTIONALITIES**
Cryptlibplugin is a tool that can encrypt/decrypt AXMEDIS object. This document shows the main functionalities provided by a first prototype of the tool forming a minimal user guide.

**RELATIONSHIP WITH OTHER TOOLS**
This tool is implemented as a plug-in. Like other plug-ins, its functionality is available via the AXMEDIS Editor and the AXMEDIS Processing Engine.

**DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS**
Here’s an example on how to use the plug-in with Axeditor.
The plug-in can be applied to all AXMEDIS objects. In the resourcePath directory, there is a sample AXMEDIS file. Select one of the AXMEDIS object.

Then select ‘Content Processing Plug-in...’ command; the following window should appear:
There is one function available: Select Cryptography Adaptation. The function accepts the parameters showed in the next figure.

In detail:

- **InputResource**: it describes the resource input
- **Mimetype**: it describes the object type
- **OutputResource**: it is the destination object
- **AlgorithmName**: it is the encryption/decryption algorithm that the user wants to use. The algorithms supported are AES, DES, 3DES, CAST-128 and Blowfish. The key size is related to the algorithm selected.
- **Keysize**: it is the number of byte of the input key. The DES algorithm wants, for instance, 8 bytes (64 bit key). The AES works with 128 bits.
- **KeyInput**: it is the input key
- **AlgorithmMode**: it is the encryption/decryption mode. If the user wants to use a mode different from ECB, he has to provide the KeyIVInput parameter.
- **KeyIVInput**: it is the key needed for the mode different from ECB
- **EncDec**: it is the flag to select the encryption or decryption operation

Clicking on execute makes the plug-in run. Output is given in the ‘out’ field:
If you try to open the "new" AXMEDIS resource an error appears.

Due to the encryption of the object the image cannot be read.
Doing the "Decryp" operation the resource can be open again.

The result:

The encryption can be seen better using a text resource. Add the Introduction.txt file that you can find in the resourcePath folder.
Follow the steps previously described for the plug in.

As the user can see the text file now is a "random" sequence of chars.
CRYPTLIB PLUG-IN

MAIN FUNCTIONALITIES
Cryptlib plug-in is a protection tool that brings into AXMEDIS the ability to exploit cryptlib functionalities, this way any kind of file can be encrypted/decrypted with several symmetric cryptography algorithms.
This plug-in is an IPMP tool. It cannot be used directly from a user but it should rather be used by developers.
Implemented algorithms:
• BlowFish
• AES
• 3DES
• CAST128

RELATIONSHIP WITH OTHER TOOLS
This tool is implemented as a plug-in. Like other plug-ins, its functionality is available via the AXMEDIS Editor and the AXMEDIS Processing Engine.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS
Please refer to the cryptlib manual for further implementation details. (www.cryptlib.com)
Parameters:
Every encryption tool should be created with the following parameters, described in cryptlib.xml:
• Key: the key for the encryption
• Iv: the initialization vector for the encryption
• Keylength: the length of the key
• Ivlength: the length of the initialization vector
If a mismatch is detected between the length and the corresponding string an exception is raised.
If length is not specified the plug-in defaults to an appropriate length which is different for each algorithm.
**LANGUAGE GUESSER PLUG-IN**

**MAIN FUNCTIONALITIES**

Language_guesser_plug-in is a tool that can detect the main language of a text document. Supported languages are: German, English, Spanish, French and Italian. This document shows the main functionalities provided by a first prototype of the tool forming a minimal user guide.

**RELATIONSHIP WITH OTHER TOOLS**

This tool is implemented as a plug-in. Like other plug-ins, its functionality is available via the AXMEDIS Editor and the AXMEDIS Processing Engine.

**DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS**

Here’s an example on how to use the plug-in with Axeditor.

The plug-in can be applied only to plain text. In the resourcePath directory, you can choose any of the .txt files (e.g. en_redcap.txt, which is the well known Red Cap tale).

Create a new AXMEDIS object and add the txt file as an embedded resource.

Then select ‘Content Processing Plug-in...’ command; the following window should appear:

Select LanguageGuesser and press execute.

Press Execute again. In the out field a string of the detected language should appear. Possible values are:

<table>
<thead>
<tr>
<th>Output string</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>de</td>
<td>German</td>
</tr>
<tr>
<td>en</td>
<td>English</td>
</tr>
<tr>
<td>es</td>
<td>Spanish</td>
</tr>
<tr>
<td>fr</td>
<td>French</td>
</tr>
<tr>
<td>it</td>
<td>Italian</td>
</tr>
</tbody>
</table>
PLAGIARISM DETECTION PLUG-IN

MAIN FUNCTIONALITIES

The plagiarism plug-in is meant to detect possible plagiarism of textual documents. It is based on an algorithm that takes into account the plagiarist behaviour. This behaviour is modelled as a set of actions like insertion, deletion, substitution or transposition and gives as a result a similarity value which is normalized between 0 and 1. Next we will show the main functionalities provided by a first prototype of the tool.

RELATIONSHIP WITH OTHER TOOLS

This tool is implemented as a plug-in. Like other plug-ins, its functionality is available via the AXMEDIS Editor and the AXMEDIS Processing Engine.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

Here’s an example on how to use the plug-in within the Axeditor. The plug-in can be applied only to plain text resource. Create a new AXMEDIS object and add the text files to be checked (original and suspect plagiarized) as embedded resources.

Then selecting the ‘Content Processing Plug-in...’ command; the following window should appear:

Select the plagiarism plug-in the following window should appear:
Select the source and target file to be compared and then click on execute.

Clicking on execute makes the plug-in run. The algorithm compares two plain text documents and gives as a result a similarity value which is normalized between 0 and 1. Output is given in the ‘out’ field.
DOCUMENT DESCRIPTOR EXTRACTOR PLUG-IN

MAIN FUNCTIONALITIES

Descriptor_extractor_plug-in is a tool that can extract high-level metadata from text documents. So far, metadata include single and multi-word keywords. This document shows the main functionalities provided by a first prototype of the tool forming a minimal user guide.

RELATIONSHIP WITH OTHER TOOLS

This tool is implemented as a plug-in. Like other plug-ins, its functionality is available via the AXMEDIS Editor and the AXMEDIS Processing Engine.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

Here’s an example on how to use the plug-in with Axeditor.

The plug-in can be applied only to plain text resources and will give meaningful results to the following five supported languages: English, French, German, Italian and Spanish. In the resourcePath directory, there is a sample file to test: en_redcap.txt. It’s the well known Red Cap tale.

Create a new AXMEDIS object and add the txt file as an embedded resource.

Then select ‘Content Processing Plug-in...’ command; the following window should appear:

There are 2 functions available:

- **KWFromComparisons**: extracts single and multi-words making a statistical comparison against reference corpora.
- **KWFromSemanticAnalysis**: extracts single and multi-words making a further analysis with the help of a semantic resource (WordNet): this function is available only for English and it doesn’t give satisfactory results, so it won’t be extended to other languages.

The first function accepts three parameters:

- the number of requested keyword
- the language of the document (supported values are: de, en, es, fr, it)
- a true/false parameter used to get numeric values beside keywords
Clicking on execute makes the plug-in run. Output is given in the ‘out’ field as a carriage-return separated list of words/multi-words:
**DOCUMENT ADAPTATION PLUG-IN**

**MAIN FUNCTIONALITIES**

Document_adaptation _plug-in is a tool that can transcode text documents between various formats. The following text formats are supported: Adobe© PDF, Rich Text Format (RTF), plain text, Postscript, Hyper-Text Markup Language (HTML). It is possible any type of transcoding among these formats.

In order to exploit the plug-in MIME types for supported formats must be known:

<table>
<thead>
<tr>
<th>Format</th>
<th>MIME type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plain text</td>
<td>text/plain</td>
</tr>
<tr>
<td>HTML</td>
<td>text/html</td>
</tr>
<tr>
<td>Postscript</td>
<td>application/postscript</td>
</tr>
<tr>
<td>PDF</td>
<td>application/pdf</td>
</tr>
<tr>
<td>RTF</td>
<td>Application/rtf</td>
</tr>
</tbody>
</table>

This document shows the main functionalities provided by a first prototype of the tool forming a minimal user guide.

**RELATIONSHIP WITH OTHER TOOLS**

This tool is implemented as a plug-in. Like other plug-ins, its functionality is available via the AXMEDIS Editor and the AXMEDIS Processing Engine.

**DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS**

Here’s an example on how to use the plug-in with Axeditor.

In the package there is a sample PDF file to test: AXMEDIS-pres-eng-v1-7-short.pdf.

Create a new AXMEDIS object and add the PDF file as an embedded resource.

Then select ‘Content Processing Plug-in...’ command; the following window should appear:
There is only one function available:

- DocumentConversion: it will make the transcoding to the format specified as the requested parameter.

For example, let’s convert the document to plain text, so let’s write in the ConversionFormat text box: text/plain

Make output a new resource, and click execute

Here’s the plain text version of the file:
**MAIN FUNCTIONALITIES**

Audio-Fingerprinting Plug-in is a tool that extracts an audio fingerprint of a given audio stream within a multimedia file. A fingerprint is perceptual digest of the given multimedia content. Therefore it is also called a perceptual hash value.

The audio stream can be embedded either in a standard audio file (mpg, wav, wma, etc...) or within a video file (mpeg, wmv, avi, etc...). This document shows the main functionalities provided by the prototype of the tool forming a minimal user guide. For testing purposes the prototype also includes a basic matching function.

**RELATIONSHIP WITH OTHER TOOLS**

This tool is implemented as a plug-in. Like other plug-ins, its functionality is available via the AXMEDIS Editor and the AXMEDIS Processing Engine.

**DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS**

Information related to the installation can be found in file README.txt.

Here is an example on how to use the plug-in with AXMEDIS Editor (AXEditor).

**FINGERPRINT EXTRACTION**

The plug-in can be applied to any audio or video resource, provided this was declared in the MIME type attribute of the resource. The output is a binary file containing the fingerprint itself and an image file of any supported format (more than 90 MIME type formats). The PNG format is recommended.

For demonstration purposes the package contains a 10 second sample wave file: “test_full.wav” and a 3 second short extract of this file: “test_snippet.wav”.

Create a new AXMEDIS object and add the wave files as embedded resources by right clicking on the object.

Right click on the “test_full.wav” resource and select ‘Content Processing Plug-ins’
Select ‘AudioFingerprintExtraction’ and the following window appears:

The algorithm parameters are:

- **nFeatures** defines the number of frequency bands in which the input stream is divided into. This affects the size of the resulting perceptual hash (default value: 18).
- **frameSize** defines the number of sample values that are considered for one perceptual hash (default value: 512).
- **frameShift** defines the shift of the window (default value: 128).
- **offset** defines the point in seconds where the plug-in starts extracting the fingerprint (default value: 0).
- **endPoint** defines the end point in seconds up till which the plug-in extracts the fingerprint (default value: 0 means end of audio file).

**Choose for both outputs a new resource and click execute.**

After receiving the “success” message, close the window and you should have a new image resource and a new resource of the MIME type “fingerprint/audio” in the AXEditor. It is advised to edit the object’s properties and set the resource filename.

To view it, just double-click on the image resource.

Here is the graphical display of the fingerprint:
FINGERPRINT MATCHING

For demonstration purposes a basic matching function was implemented.
Repeat the extraction steps for the “test_snippet.wav” file.
After the extraction process has finished right click on any “fingerprint/audio” object, select the “AXAFPCompare” function and execute it:

On the following window just select the two fingerprint files to be compared and click on execute
The order is not important; the algorithm automatically decides the candidate and reference object order:

The result shows the length in seconds of the original candidate and the reference objects (the audio files the fingerprint was extracted from).

The parameter MINBER (minimum bit error rate) is the minimal bit error rate (BER) found during the comparison. The TIMEPOS is the time position for the minimal BER.
The result value will show “Success!” if the comparison was performed without errors. Otherwise an error message will be displayed. Please note, that the “Success!” message indicates the technical success of the comparison process and not the matching probability.
M2ANY - AUDIO FP PLUG-IN

MAIN FUNCTIONALITIES
M2ANYAudio_fingerprint_plug-in is a tool that extracts an audio fingerprint of a given audio stream within a WAV file. This document shows the main functionalities provided by a first prototype of the tool forming a minimal user guide. A matching function is still in development.

RELATIONSHIP WITH OTHER TOOLS
This tool is implemented as a plug-in. Like other plug-ins, its functionality is available via the AXMEDIS Editor and the AXMEDIS Processing Engine.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS
CAVEAT: This guide assumes you licensed the corresponding technology from M2ANY! The licensed executables (cfymain.exe, xtrmain.exe, cfy.dll, xtr.dll and asign.dll) have to be installed in the plug-in directory! You also need to indicate the path to an existing fingerprint database which is also provided by M2ANY.
Information related to the installation can be found in file README.txt.
The plug-in can be applied to WAV files, provided this was declared in the MIME type attribute of the resource. The output is a binary file containing the fingerprint in binary format.
Create a new AXMEDIS object and add a wave file as an embedded resource by right clicking on the object.
The provided audio file excerpt (“nene.wav”) is used here as a complete fingerprint of the complete song is contained within the M2ANY Database.
Right click on the WAV resource and select ‘Content Processing Plug-ins’

Select ‘M2ANYAudioFingerprintExtraction’ and click “Execute” the following window should appear:

Choose for the output a new resource and click execute.

A console window will shortly appear indicating the extraction process. After receiving the “success”-message, close the window and you should have a new resource of the MIME type “fingerprint/m2anyAfp” in the AXMEDIS editor. It is advised to edit the objects properties with the original resource filename.

For demonstration purposes a basic matching function was included. For this section the fingerprint previously generated will be used.

After the extraction process has finished right click on any “fingerprint/ m2anyAfp” object, select the “M2ANYAXAFPCompare” function and execute it:
On the following window select a new resource as the output resource and provide the path to an existing database directory (containing at least 10 fingerprint files in it) and click on execute:

The result value will show “Success!” if the comparison was performed without errors. Else an error message will display. The resulting resource will be a text file. Please edit its properties and add an arbitrary content ID. This prevents the AXEditor from crashing. This issue is soon to be fixed.

The matching TUID (Track unique ID) for the excerpt should be the ID: 1038200000000002 which belongs to the song “7 Seconds” by Neneh Cherry. (Unfortunately the provided binary demonstration package does not allow resolving the TUIDs to their source files.)
VIDEO FP PLUG-IN

MAIN FUNCTIONALITIES

Video_fingerprint_plug-in is a tool that extracts a fingerprint of a given video stream. The video stream can be embedded in a video file (mpeg, wmv, avi, etc...).

This document shows the main functionalities provided by a first prototype of the tool forming a minimal user guide.

RELATIONSHIP WITH OTHER TOOLS

This tool is implemented as a plug-in. Like other plug-ins, its functionality is available via the AXMEDIS Editor and the AXMEDIS Processing Engine.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

FINGERPRINT EXTRACTION

The plug-in can be applied to any video resources, provided this was declared in the MIME type attribute of the resource. The output can be any mimetype within the image/* format. The PNG format is recommended. In the package there are two sample avi files to do a test: “test.avi” and “test_snippet.avi”.

Create a new AXMEDIS object and with a right click, add the avi files as embedded resources.

With a right Click on a resource, select ‘Content Processing Plug-ins’. Then you should search for the option ‘VideoFingerprintExtraction’ and click execute.

After the selection, the following window should appear:

Make output a new resource, select the desired number of frames to be processed and click execute.
After receiving the “success” message, close the window and you should have a new image resource and a new resource of the MIME type “fingerprint/video” in the AXEditor. It is advised to edit the objects properties to change the resource filename.
With a double click in the resource, the following graphical of the fingerprint will be shown:

---

**FINGERPRINT MATCHING**

For demonstration purposes a basic matching function was implemented.

Repeat the extraction steps for the “test_snippet.avi” file.
After the extraction process has finished right click on any “fingerprint/video” object, select the “AxVFPCoCompare” function and execute it:

On the following window just select the two fingerprint files to be compared and click on execute
The order is not important; the algorithm automatically decides the candidate and reference object order:

The result shows the length in seconds of the original candidate and reference objects (the video files the fingerprint was extracted from).
The parameter MINBER (minimum bit error rate) is the minimal bit error rate (BER) found during the comparison. The TIMEPOS is the time position for the minimal BER.

The result value will show “Success!” if the comparison was performed without errors. Else an error message will display. Please note, that the “Success!” message indicates the technical success of the comparison process and not the matching probability.
GENERIC RESOURCE FILES FP PLUG-IN

MAIN FUNCTIONALITIES
GenericFiles_fingerprint_plug-in is a tool that calculates a fingerprint (a cryptographic hash) for a given arbitrary file. This document shows the main functionalities provided by a first prototype of the tool forming a minimal user guide.

RELATIONSHIP WITH OTHER TOOLS
This tool is implemented as a plug-in. Like other plug-ins, its functionality is available via the AXMEDIS Editor and the AXMEDIS Processing Engine.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS
Information related to the installation can be found in file README.txt. The plug-in can be applied to any content type. The output is a string. In the package there is a test file: test.pdf. Create a new AXMEDIS object and add the wav file as an embedded resource.
Right Click on the resource and select ‘Content Processing Plug-ins’. The following window should appear:

Unselect the check-box (“only functions for Resource XXX”):

Select ‘GenericRessource::ProcessResourceMessageDigest’
Select the available algorithm (5: MD5, 6: SHA-1):

So far, the result is calculated and shown in a dialog:
VIDEO DESCRIPTOR PLUG-IN

MAIN FUNCTIONALITIES

MP7 Videodescriptor Extraction plug-in is a tool that extracts a MPEG-7 XML descriptor of a provided MPEG video file. This document shows the main functionalities provided by a first prototype of the tool forming a minimal user guide. The MPEG-7 eXperimental Model (XM) was used as a base library for this plug-in (see readme.txt for download link).

RELATIONSHIP WITH OTHER TOOLS

This tool is implemented as a plug-in. Like other plug-ins, its functionality is available via the AXMEDIS Editor and the AXMEDIS Processing Engine.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

Taken from the MPEG-7 eXperimental Model (XM) documentation. For a more detailed description please refer to it.

GOF/GOP COLOUR DESCRIPTOR

<table>
<thead>
<tr>
<th>Name</th>
<th>Santhana Krishnamachari, Mufti Ferman</th>
</tr>
</thead>
<tbody>
<tr>
<td>E-Mail</td>
<td><a href="mailto:santhana.krishnamachari@philips.com">santhana.krishnamachari@philips.com</a></td>
</tr>
<tr>
<td>Input</td>
<td>Video</td>
</tr>
<tr>
<td>Summary</td>
<td>This library implements the GoF/GoP Colour descriptor which is used to describe the colour characteristics of a collection of video frames (and a collection of images). It consists of one primary and four secondary attributes. Since the feature vector is short, a simple absolute distance or squared distance criterion can be used for matching.</td>
</tr>
<tr>
<td>Strong Points</td>
<td>None</td>
</tr>
<tr>
<td>Limitations</td>
<td>Use mean or median aggregation for matching</td>
</tr>
<tr>
<td>Known Problems</td>
<td>None</td>
</tr>
</tbody>
</table>

DOMINANT COLOR DESCRIPTOR

<table>
<thead>
<tr>
<th>Name</th>
<th>Jungmin Song, Heon Jun Kim, Leszek Cieplinski, Prof. Manjunath</th>
</tr>
</thead>
<tbody>
<tr>
<td>E-Mail</td>
<td><a href="mailto:jmsong73@mail.lgcit.com">jmsong73@mail.lgcit.com</a>, <a href="mailto:hjk@lge.co.kr">hjk@lge.co.kr</a>, <a href="mailto:Leszek.Cieplinski@vil.ite.mee.com">Leszek.Cieplinski@vil.ite.mee.com</a>, <a href="mailto:manj@ece.ucsb.edu">manj@ece.ucsb.edu</a></td>
</tr>
<tr>
<td>Contact</td>
<td>Jungmin Song(general), Leszek Cieplinski(color variance), Prof. Manjunath(dominant color extraction and search algorithm)</td>
</tr>
<tr>
<td>Input</td>
<td>Images</td>
</tr>
<tr>
<td>Summary</td>
<td>The dominant color descriptor is useful for image and video retrieval. It targets content-based retrieval for color, either for the whole image or for any arbitrary shaped region (rectangular or irregular). It is a very compact descriptor, requiring less than 6-8 colors per region. Since colors are not pre-quantized as in the histogram type color descriptors, the representation is more accurate. To accomplish high accuracy in retrieval, Spatial Coherency and/or Color Variance can be utilized. It is intended for applications that use object based representations (objects or regions in an image).</td>
</tr>
<tr>
<td>Limitations</td>
<td>The maximum allowed number of dominant colors is 8.</td>
</tr>
<tr>
<td>Known Problems</td>
<td>None.</td>
</tr>
<tr>
<td>Parameters</td>
<td>Color space and color quantization parameters (implementation preset to standard values)</td>
</tr>
<tr>
<td>HomogeneousTexture Descriptor</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>yanglim Choi, M/M Lab, Samsung Advanced Institute of Technology</td>
</tr>
<tr>
<td>E-Mail</td>
<td><a href="mailto:yanglimc@samsung.com">yanglimc@samsung.com</a></td>
</tr>
</tbody>
</table>
Input
Summary Images/Regions (JPEG, BMP, etc)
This component generates a texture descriptor for a homogeneously textured Image/Region for Search and Retrieval using texture features. The descriptor components are the average and the standard deviation of the image together with the energies and the energy deviations of Gabor filtered response of each (predefined) frequency channels.

Strong Points Statistically very precise description of homogeneously textured region.
Limitations Relatively large in size (32 components for the basic layer and 62 components for the enhanced layer).

Known Problems NONE
Parameters Plug-in implementation set to standard values.

COLOR STRUCTURE DESCRIPTOR

Name Jim Errico, Sharp Labs of America; Dean Messing, Sharp Labs of America
E-Mail jerrico@sharplabs.com; deanm@sharplabs.com
Input Images
Summary This component is the implementation of the extraction and search functionality for the ColorStructure Descriptor.

Strong Points NA
Limitations NA

Usage Example

GoF/GoP Color Descriptor
The plug-in can be applied to video MPEG resources, provided this was declared in the mime type attribute of the resource. The output is a XML file containing the descriptor data.

Create a new AXMEDIS object and add a MPEG file as a embedded resource by right clicking on the new created object.

Right click on the MPEG resource and select ‘Content Processing Plug-ins’
In the upcoming menu select ‘MP7 Videodescriptor Extraction: AxMPEG7GoFGop’. Press “Execute” and the following window should appear:

The preset configuration should be optimal for most use cases. Choose as the output a new resource and click “Execute”. A console window will appear and stay for a few seconds. This is normal due to the nature of the used original Extraction module.

After receiving the “success”-message, close the window and you should have a text resource in the AXMEDIS editor containing the XML descriptor data. It is advised to edit the objects properties and add the original resource filename.
Dominant Color Descriptor

The plug-in can be applied to image resources, provided this was declared in the mime type attribute of the resource. The output is a XML file containing the descriptor data.

Create a new AXMEDIS object and add an image resource as a embedded resource by right clicking on the new created object. Right click on the MPEG resource and select ‘Content Processing Plug-ins’.

In the upcoming menu select ‘MP7 Videodescriptor Extraction: AxDominantColor’.

Press “Execute” and the following window should appear:

This descriptor does not take any specific parameters. Choose as the output a new resource and click “Execute”. A console window will appear and stay for a few seconds. This is normal due to the nature of the used original Extraction module.

After receiving the “success”-message, close the window and you should have a text resource in the AXMEDIS editor containing the XML descriptor data. It is advised to edit the objects properties and add the original resource filename.
Homogeneous Texture Descriptor
The plug-in can be applied to image resources, provided this was declared in the mime type attribute of the resource. The output is a XML file containing the descriptor data.
Create a new AXMEDIS object and add an image resource as a embedded resource by right clicking on the new created object. Right click on the MPEG resource and select ‘Content Processing Plug-ins’

In the upcoming menu select ‘MP7 Videodesciptor Extraction: AxHomoTexture’.

Press “Execute” and the following window should appear:

This descriptor does not take any specific parameters. Choose as the output a new resource and click “Execute”. A console window will appear and stay for a few seconds. This is normal due to the nature of the used original Extraction module.
After receiving the “success”-message, close the window and you should have a text resource in the AXMEDIS editor containing the XML descriptor data. It is advised to edit the objects properties and add the original resource filename.

**Color Structure Descriptor**

The plug-in can be applied to image resources, provided this was declared in the mime type attribute of the resource. The output is a XML file containing the descriptor data.

Create a new AXMEDIS object and add an image resource as a embedded resource by right clicking on the new created object. Right click on the MPEG resource and select ‘Content Processing Plug-ins’

In the upcoming menu select ‘MP7 Videodescriptor Extraction: AxColorStructure’.

Press “Execute” and the following window should appear:
Choose as the output a new resource and click “Execute”. A console window will appear and stay for a few seconds. This is normal due to the nature of the used original Extraction module.

After receiving the “success”-message, close the window and you should have a text resource in the AXMEDIS editor containing the XML descriptor data. It is advised to edit the objects properties and add the original resource filename.
VIDEO ADAPTATION PLUG-IN

MAIN FUNCTIONALITIES

FFmpegTranscoder.dll_plug-in is a tool that can process video objects. Operations such as size scaling, format transcoding, changing of such parameters as the aspect ratio, video frame rate, bitrate, audio bit rate, audio sampling rate, the number of audio channels, and disabling the audio, video or both streams can be performed. The plug-in makes use of the FFmpeg library and can process a wide range of audio and video formats contained within video files (mpeg, wmv, avi, mp2, mp3, PCM, etc…). This document shows the main functionalities provided by the prototype of the tool forming a minimal user guide. For testing purposes the demonstration package includes a small video file.

RELATIONSHIP WITH OTHER TOOLS

This tool is implemented as a plug-in. Like other plug-ins, its functionality is available via the AXMEDIS Editor and the AXMEDIS Processing Engine.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

The plug-in makes use of the following parameters. The default settings will not change the input in any way. Changing any parameter will activate its usage in the transcoding process.

- **OutExtension**: Mimetype for output video.
- **vidAspectRatio**: The aspect ratio (Width/Height) either as a relation (4:3, 16:9) or as a floating point number (1.3333, 1.7777)
- **vidBitrate**: Sets the video bitrate (in kbit/s)
- **vidFRate**: Sets the frame rate (Hz value). Please note that some video formats do not support all frame rate values.
- **vidXFSize vidYFSize**: Sets frame size. Both parameters have to be provided in order for this option to be used.
- **audioBitrate**: Sets the audio bitrate (in kbit/s)
- **audioSamplingrate**: Sets the audio sampling rate (in Hz)
- **audioChannels**: Sets the number of audio channels
- **disableVid**: Disables the video stream
- **disableAudio**: Disables the audio stream
- **sameQ**: Use same video quality as source (implies VBR)

USAGE EXAMPLE

Here’s an example on how to use the plug-in with AXMEDIS Editor (AXEditor).

**Video format transcoding**

The plug-in can be applied to any video resources, provided this was declared in the MIME type attribute of the resource. The output is a video file containing the result of the transcoding process.

For demonstration purposes the package contains a 5 second sample video file: “test.avi”.

This file can not be played by the player included within the AXMEDIS Editor. In this guide the file will be transcoded to the MPEG format, which can be played by the built-in video player.

Create a new AXMEDIS object and add the video file as embedded resource by right clicking on the object.
Right click on the “test.avi” resource and select ‘Content Processing Plug-ins’

Select “VideoAdaptation:AX_ffmpegTranscoder”, click “Execute” and the following window will appear:

Select “New Resource” as the output resource and change the output Mimetype to the desired format, in this case “video/mpeg”. Since no other transcoding processing is needed just click on “Execute”.

AXMEDIS Editor User Manual
After receiving the “success”-message, close the window and you should have a new video resource of the MIME type “video/mpeg” in the AXEditor. To view the video just double click on it.

Video Resizing
In this guide the sample video will be transcoded to the MPEG format and it also will be resized from the original 352x288 to 112x112 pixels.
Follow the steps in section “Video format transcoding” up to the plug-in parameter window.
Select the desired output format (“video/mpeg”) and enter the desired dimensions for the video.

Please note that you have to provide both dimensions for the new frame in order to resize a video. For example: only providing the height and leaving the width at “0” will lead to both parameters being ignored.
After clicking on execute the new video resource will appear in the AXMEDIS Editor and can be viewed by double clicking on it.

For a complete list of supported codecs visit: http://ffmpeg.mplayerhq.hu/ffmpeg-doc.html#SEC21
WORKFLOW EDITOR PLUG-IN

MAIN FUNCTIONALITIES

The Workflow Editor Plug-in is a library used by AXMEDIS Editor to enable communication with AXMEDIS workflow engine. It exposes following functions:

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notification of Completion</td>
<td>This method is invoked by the editor to send back the notification towards workflow engine for the completion of previously issues asynchronous request.</td>
</tr>
<tr>
<td>Get Workflow Information</td>
<td>This method is invoked by the application to request workflow information from workflow engine.</td>
</tr>
<tr>
<td>Editing of Object</td>
<td>This method is invoked through a webservice call coming from Workflow Request Gateway. This method will load the specified object (AXOID) and will allow the user to edit the object in the AXMEDIS Editor.</td>
</tr>
<tr>
<td>View Object Attributes</td>
<td>This method is invoked through a webservice call coming from Workflow Request Gateway. This method will allow the workflow engine to retrieve the object attributes for the object specified by the AXOID.</td>
</tr>
<tr>
<td>Add History Information</td>
<td>This method is invoked through a webservice call coming from Workflow Request Gateway. This method will allow the workflow engine to add the object history for the object specified by the AXOID.</td>
</tr>
</tbody>
</table>

RELATIONSHIP WITH OTHER TOOLS

The Workflow Editor Plug-in is loaded by the AXMEDIS Editor. This plug-in communicates directly with AXMEDIS Workflow Request Gateway and AXMEDIS Workflow Response Gateway. For Microsoft Biztalk server, this plug-in communicates directly with the workflow engine. There are currently two versions of the plug-in – one for communicating with OpenFlow, and one for Biztalk. These will eventually be unified.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

The functions of this plug-in are invoked by AXMEDIS Editor. Also the Functions can be invoked by the workflow engine automatically. The following Screenshots shows the empty AXMEDIS Editor.

Figure 1: Empty AXMEDIS Editor
When the Workflow Editor Plug-in receives request from Workflow Engine for the editing of Object, it will load the specified object from the AXMEDIS Database or if the AXOID is not specified then it will create a new object as shown in the following figure:

![Figure 2: Creation of new Object following the Workflow Request](image)

The user can also select the “Workflow View” tab from the Editor. This will display the workflow related information for the object currently being edited/viewed as shown in following figure.

![Figure 3: The Workflow View for the Object](image)

When the editing of Object is completed, the user can notify workflow by selecting “Notify Workflow Activity Completion” command from the File Menu as shown in the following figure. This will send a notification signal to workflow engine.
Figure 4: Sending Notification of Completion to Workflow Engine
WORKFLOW RULE EDITOR PLUG-IN

MAIN FUNCTIONALITIES

The Workflow Rule Editor Plug-in is a library used by AXCP Rule Editor and AXMEDIS PnP Editor to enable communication with AXMEDIS workflow engine. It exposes following functions:

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notification of Completion</td>
<td>This method is invoked by the editor to send back the notification towards workflow engine for the completion of previously issues asynchronous request.</td>
</tr>
<tr>
<td>Editing of AXCP Rule</td>
<td>This method is invoked through a webservice call coming from Workflow Request Gateway. This method will load the AXCP Rule editor with the specified Rule Header and will allow the user to edit the AXCP Rule in the AXCP Rule Editor.</td>
</tr>
<tr>
<td>Edit PnP Programme</td>
<td>This method is invoked through a webservice call coming from Workflow Request Gateway. This method will load the AXMEDIS PnP editor with the specified Programme Header and will allow the user to edit the AXMEDIS PnP Programme in the PnP Editor.</td>
</tr>
</tbody>
</table>

RELATIONSHIP WITH OTHER TOOLS

The Workflow Rule Editor Plug-in is loaded by the AXCP Rule Editor and PnP Editor. This plug-in communicates directly with AXMEDIS Workflow Request Gateway and AXMEDIS Workflow Response Gateway. For Microsoft Biztalk server, this plug-in communicates directly with the workflow engine. There are currently two versions of the plug-in – one for communicating with OpenFlow, and one for Biztalk. These will eventually be unified.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

The functions of this plug-in are invoked by AXCP Rule Editor. Also the Functions can be invoked by the workflow engine automatically. The following Screenshots shows the empty AXCP Rule Editor.

![Figure 1: Empty AXCP Rule Editor](image)

When the Workflow Rule Editor Plug-in receives request from Workflow Engine for the editing of a Rule, it will load the specified XML Rule from the Workflow Request as shown in the following figure:

![Figure 2: Rule Loaded from Workflow Engine](image)

When the editing of the rule is completed, the user can notify workflow by selecting “Notify Workflow Activity Completion” command from the Workflow Menu as shown in the following figure. This will send a notification signal to workflow engine.

![Figure 3: Sending of Notification of Completion](image)

The User can confirm the notification command by selecting ‘OK’ from the message box.
The functions of this plug-in can also be invoked by PnP Editor. Also the Functions can be invoked by the workflow engine automatically. The following Screenshots shows the empty PnP Editor.

When the Workflow Rule Editor Plug-in receives request from Workflow Engine for the editing of a PnP Programme, it will load the specified XML from the Workflow Request as shown in the following figure:
When the editing of the Programme is completed, the user can notify workflow by Activating the Programme in the PnP Engine as shown in the following figure. This will send a notification signal to workflow engine.
WORKFLOW ENGINE PLUG-IN

The Workflow Engine Plug-in is loaded by the AXCP Scheduler and by PnP Engine. Generally the functions offered by this plug-in are invoked automatically by these tools without the intervention of the User.

MAIN FUNCTIONALITIES

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notification of Completion</td>
<td>This method is invoked by the editor to send back the notification towards workflow engine for the completion of previously issues asynchronous request.</td>
</tr>
<tr>
<td>Workflow Process Request</td>
<td>This method is invoked by the PnP engine to send request for activation of a workflow process identified the supplied processID.</td>
</tr>
<tr>
<td>Install and Activate</td>
<td>This method allows the workflow engine to install a new rule in the AXCP engine. The ruleID of the newly installed rule will be returned as ruleID.</td>
</tr>
<tr>
<td>Run Rule</td>
<td>This method allows the workflow engine to run a rule in the AXCP engine as per the supplied parameters.</td>
</tr>
<tr>
<td>Deactivate Rule</td>
<td>This method allows the workflow engine to deactivate a rule in the AXCP engine as per the supplied parameters.</td>
</tr>
<tr>
<td>Suspend Rule</td>
<td>This method allows the workflow engine to suspend a rule in the AXCP engine as per the supplied parameters.</td>
</tr>
<tr>
<td>Pause Rule</td>
<td>This method allows the workflow engine to pause a rule in the AXCP engine as per the supplied parameters.</td>
</tr>
<tr>
<td>Kill Rule</td>
<td>This method allows the workflow engine to kill a rule in the AXCP engine as per the supplied parameters.</td>
</tr>
<tr>
<td>Remove Rule</td>
<td>This method allows the workflow engine to remove a rule from the AXCP engine as per the supplied parameters.</td>
</tr>
<tr>
<td>Resume Rule</td>
<td>This method allows the workflow engine to resume a rule in the AXCP engine as per the supplied parameters.</td>
</tr>
<tr>
<td>Get Rule Status</td>
<td>This method allows the workflow engine to know the status of the a rule in the AXCP engine as per the supplied parameters.</td>
</tr>
<tr>
<td>Get Rule Log</td>
<td>This method allows the workflow engine to know the run log for a rule in the AXCP engine as per the supplied parameters.</td>
</tr>
<tr>
<td>Get List of Rules</td>
<td>This method allows the workflow engine to retrieve the list of currently installed rules in the AXCP engine as per the supplied parameters.</td>
</tr>
<tr>
<td>Get Rule</td>
<td>This method allows the workflow engine to retrieve the rule schema from the AXCP engine as per the supplied parameters.</td>
</tr>
<tr>
<td>Status Request to PnP Engine</td>
<td>This method allows the workflow engine to retrieve the status of the PnP engine.</td>
</tr>
<tr>
<td>Suspend PnP Programme</td>
<td>This method allows the workflow engine to suspend a program in the PnP engine.</td>
</tr>
<tr>
<td>Abort PnP Programme</td>
<td>This method allows the workflow engine to abort a program in the PnP engine.</td>
</tr>
<tr>
<td>Resume PnP Programme</td>
<td>This method allows the workflow engine to resume a program in the PnP engine.</td>
</tr>
<tr>
<td>Activate PnP Programme</td>
<td>This method allows the workflow engine to activate a program in the PnP engine.</td>
</tr>
<tr>
<td>Workflow Notification</td>
<td>This method allows the workflow engine to send the notification to the PnP engine for the previously issues request to activate a process.</td>
</tr>
</tbody>
</table>
RELATIONSHIP WITH OTHER TOOLS

The Workflow Engine Plug-in is loaded by the AXCP Scheduler and PnP Engine. This plug-in communicates directly with AXMEDIS Workflow Request Gateway and AXMEDIS Workflow Response Gateway. For Microsoft Biztalk server, this plug-in communicates directly with the workflow engine. There are currently two versions of the plug-in – one for communicating with OpenFlow, and one for Biztalk. These will eventually be unified.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

The functionality offered by this plug-in is hidden from the user. Hence there are no screenshots for it.
RINGTONE ADAPTATION PLUG-IN

MAIN FUNCTIONALITIES

Ringtone Adaptation refers to the adaptation of ringtones of popular formats to enhance usability and manage the variable delivery to cater for heterogeneous client devices and user requirements on-demand. It can be used to transcode the ringtones depending on the client devices e.g. some mobile phones may support only low bit rate ringtones while others will be having restrictions on the size of the ringtone.

RELATIONSHIP WITH OTHER TOOLS

This tool is implemented as a plug-in. Like other plug-ins, its functionality is available via the AXMEDIS Editor and the AXMEDIS Processing Engine.

DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

A step by step Example showing how to use the functions is given below.

CONVERT FUNCTION

Description: Used to Convert a ringtone to different formats. The formats supported currently are x-mp3 (.mp3), x.aiff (.aif, .aiff), x.wav (.wav), basic (.au, .snd), x-ms-wma (.wma), x-vorbis (.ogg), x-pn-realaudio (.ra, .ram)

How to use:

Load an embedded resource (audio/ringtone file) into the AXMEDIS Editor
Right click on the resource and select Content processing plug-ins

It will pop up a new window showing the different content processing plug-ins available for the particular resource, in our case it is ringtone.
Select the Convert function to convert the ringtone to any popular format. The formats supported are 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)

It will take you to the next screen where you can specify the various parameters for converting the ringtone. Once you enter the parameters and click execute, it will convert the ringtone to the appropriate format. If the ringtone conversion is successful then in result’s space you can see SUCCESS or else it will return Error along with an error message.

The description of each parameter is given below.

**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)

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

**OutputResource**
- Description: Where the produced resource will be stored
- Parameter Type AxResource
- Default Value:
- Constraints:

**Result**
- Result type: string
- Result Description: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error

---

**CONVERT_TO_MP3 FUNCTION**

**Description:** Used to convert a ringtone to MP3 format. The input formats supported currently are 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)

![Convert_to_MP3 Function Screenshot]

**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)

**OutputResource**
- Description: Where the produced resource will be stored
- Parameter Type AxResource
- Default Value:
- Constraints:

**Result**
- Result type: string
- Result Description: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error

---

**CONVERT_TO_WAV FUNCTION**

**Description:** Used to convert a ringtone to WAV format. The input formats supported currently are 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)
AXMEDIS Editor User Manual

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

**OutputResource**
- **Description:** Where the produced resource will be stored
- **Parameter Type:** AxResource
- **Default Value:**
- **Constraints:**

**Result**
- **Result type:** string
- **Result Description:** The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error

---

**RESAMPLE FUNCTION**

**Description:** Resamples the input file (i.e. changing frequency, bitrate, sampling rate etc)

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

**Mimetype**
- **Description:** Mimtype for output resource
- **Parameter Type:** STRING
- **Default Value:**
Constraints:
  
  Resource Type: audio
  Resource Format: x-mpeg, x-aiff, x-wav, basic, x-vorbis, x-ac3

**OutputResource**
Description: Where the produced resource will be stored
Parameter Type RESOURCE
Default Value:
Constraints:

**OutputSamplingRate**
Description: Sampling rate of the output audio file (default: sampling rate of the input)
Parameter Type UINT32
Default Value:
Constraints:
  
  Resource Type:

Ranges:

**OutputNumChannels**
Description: Number of channels of the output audio file (default: number of channels of the input)
Parameter Type UINT16
Default Value:
Constraints:
  
  Resource Type:

Ranges:

**OutputBitRate**
Description: Bit rate of the output audio file - Only applies to compressed audio file formats (default: 64 kb)
Parameter Type UINT16
Default Value:
Constraints:
  
  Resource Type:

Ranges:

**Result**
Result type: STRING
Result Description: The result of import, SUCCESS if ok, ERROR followed by a message in case of error

---

**CONVERT AND RESAMPLE FUNCTION**

**Description:** Converts the file into any supporting formats and resample it (i.e. changing frequency, bitrate, sampling rate etc) at the same time. Please note that some values of the Sampling rates and frequencies cannot exist together according to the ffmpeg library used and hence if the plug-in shows an unknown exception then the user has to restart the plug-in and give different values.

![Convert And Resample Function Screenshot](image-url)
InputResource
Description: The Resource to be converted
Parameter Type RESOURCE
Default Value:
Constraints:
  Resource Type: audio
  Resource Format: x-mpeg (.mp3), x.aiff (.aiff), x-wav (.wav), basic (.au, .snd), x-ms-wma (.wma), x-vorbis (.ogg), x-pn-realaudio (.ra, .ram)
Ranges:

Mimetype
Description: Mimetype for output resource
Parameter Type STRING
Default Value:
Constraints:
  Resource Type: audio
  Resource Format: x-mpeg, x-aiff, x-wav, basic, x-vorbis, x-ac3

OutputResource
Description: Where the produced resource will be stored
Parameter Type RESOURCE
Default Value:
Constraints:

OutputSamplingRate
Description: Sampling rate of the output audio file (default: sampling rate of the input)
Parameter Type UINT32
Default Value:
Constraints:
  Resource Type:
Ranges:

OutputNumChannels
Description: Number of channels of the output audio file (default: number of channels of the input)
Parameter Type UINT16
Default Value:
Constraints:
  Resource Type:
Ranges:

OutputBitRate
Description: Bit rate of the output audio file - Only applies to compressed audio file formats (default: 64 kb)
Parameter Type UINT16
Default Value:
Constraints:
  Resource Type:
Ranges:

Result
Result type: STRING
Result Description: The result of import, SUCCESS if ok, ERROR followed by a message in case of error

GETINFO FUNCTION
Description: Get all the information about the input Ring Tone
**InputResource**

Description: The Resource to be converted
Parameter Type RESOURCE
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:

**SamplingRate**

Description: Sampling rate of the input ring tone
Parameter Type UINT32
Default Value:
Constraints:
  Resource Type:
Ranges:

**NumChannels**

Description: Number of channels of the input ring tone
Parameter Type UINT16
Default Value:
Constraints:
  Resource Type:
Ranges:

**BitRate**

Description: Bit rate of the input ring tone - (default: 64 kb)
Parameter Type UINT16
Default Value:
Constraints:
  Resource Type:
Ranges:

**Duration**

Description: Duration of the Ringtone File (In the format Hours: Mins: Secs: milliseconds)
Parameter Type STRING
Default Value:
Constraints:
  Resource Type:
Ranges:

**Result**

Result type: STRING
Result Description: The result of import, SUCCESS if ok, ERROR followed by a message in case of error
**CLIP FUNCTION**

**Description:** Clips the file for the specified time (for e.g. reducing it to a 30 sec clip)

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

**OutputResource**
- **Description:** Where the produced resource will be stored
- **Parameter Type:** RESOURCE
- **Default Value:**
- **Constraints:**

**Mimetype**
- **Description:** Mimetype for output resource
- **Parameter Type:** STRING
- **Default Value:**
- **Constraints:**
  - Resource Type: audio
  - Resource Format: x-mpeg, x-aiff, x-wav, basic, x-vorbis, x-ac3

**ReadStartingTime**
- **Description:** Starting time for the clip (default: beginning of the file)
- **Parameter Type:** FLOAT
- **Default Value:**
- **Constraints:**
  - Resource Type:

**ReadEndingTime**
- **Description:** Ending time for the clip (default: end of the file)
- **Parameter Type:** FLOAT
- **Default Value:**
- **Constraints:**
  - Resource Type:

**Result**
- **Result type:** STRING
- **Result Description:** The result of import, SUCCESS if ok, ERROR followed by a message in case of error
IMAGE PROCESSING PLUG-IN

MAIN FUNCTIONALITIES

The image processing plug-in allows adapting image resources to various use cases. For example, it can be used to convert different image formats, apply various effects, resize, mirror, etc. In total, the plug-in is composed of forty-one functions that are:

- **Conversion**, to convert the image
- **Import**, to import an image
- **Resize**, to resize the image
- **Contrast**, to change the image contrast
- **Edge**, to highlight edges of the image
- **Emboss**, to highlight edges with 3D effect
- **Blur**, to blur the image
- **GaussianBlur**, to apply a Gaussian Blur to the image
- **Median**, to apply a median filter to the image
- **Mirror**, to mirror the image
- **Noise**, to add noise in the image
- **Despeckle**, to reduce the noise from the image using the despeckle filter
- **Equalize**, to apply an histogram equalization to the image
- **Enhance**, to minimize the noise of the image
- **ExtractChannel**, to extract a channel from the image
- **GrayScale**, to convert a coloured image to grayscale
- **Magnify**, to scale up the image
- **Minimize**, to scale down the image
- **Modulate**, to modulate hue, saturation, and brightness of the image
- **Monochrome**, to create a monochrome image
- **Negate**, to negate colours in the image
- **Normalize**, to increase contrast by normalizing the pixel values
- **OilPaint**, to create a image looks like oil painting
- **Quality**, to change the JPEG/MIFF/PNG compression
- **Quantize**, to set the preferred number of colours in the image
- **Raise**, to highlight or dark the edges of an image to give a 3D raised or lowered effect
- **ReduceNoise**, to reduce the noise of the image
- **Replace**, to replace the image
- **FloodFill**, to apply a flood-fill texture
- **Roll**, to roll the image by a specified number of columns and rows
- **Rotate**, to rotate the image specifying a number of degrees
- **Scale**, to scale the image by using a specified ratio
- **Shear**, to create a parallelogram by sliding the image by X of Y axis
- **Shade**, to shade the image using distant light source
- **Spread**, to spread pixels randomly
- **SetOpacity to set the opacity of the image**
- **SubImage**,
- **GetInfo**, to see information about the image
- **SetMaskColour**,
- **Paste**, to paste the image
- **Test**, to test the image

RELATIONSHIP WITH OTHER TOOLS

This tool is implemented as a plug-in. Like other plug-ins, its functionality is available via the AXMEDIS Editor and the AXMEDIS Processing Engine.
DETAILED DESCRIPTION OF THE FUNCTIONALITIES AND SCREENSHOTS

Here’s an example on how to use the plug-in with AXMEDIS Editor.

The plug-in can be applied to all images resources in all formats embedded into an AXMEDIS object. Selecting one resource in the tree and right clicking, select Content Processing plugins…

A new dialog will appear with the list of available functionalities. Selecting a functionality will appear a brief description in the Function description box.

Selecting the appropriate function and pressing the Execute button a new dialog appears with a number of fields to be filled-in. The aspect of the dialog and the number of fields is different for each function.

Please, refer to next sections for a detailed description of the values needed for each functionality.

In the Output Resource cascading menu is possible to decide if the function will produce a new resource or will overwrite the old one.

Here’s a brief analysis of image processing functionalities.

Since the image processing plug-in is based on the GPL source code of ImageMagick, for a more detailed description of these functionalities, please refer to the following links:


Examples of ImageMagick usage are available here: http://www.cit.gu.edu.au/~anthony/graphics/imagick6

---

CONVERSION

STRING Conversion ( RESOURCE InputResource, STRING Mimetype, RESOURCE OutputResource )

Version: 1.0
Description: Convert an image

Parameter List

Name: InputResource
Description: The Resource to be converted
Parameter Type: RESOURCE
Default Value:
Constraints:
  Resource Type: image
  Resource Format: jpeg gif png
Rangess:

Name: Mimetype
Description: Mimetype for output resource
Parameter Type: STRING
Default Value:
Constraints:

Name: OutputResource
Description: Where the produced resource will be stored
Parameter Type: RESOURCE
Default Value:
Constraints:

Result: Result
Result type: STRING
Result Description: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

---

IMPORT

STRING Import ( STRING Path, RESOURCE OutputResource, STRING MimeTyple )

Version: 1.0
Description: Import an image

Parameter List

Name: Path
Description: Path to the image
Parameter Type: STRING
Default Value:
Constraints:

Name: OutputResource
Description: Where the imported resource will be stored
Parameter Type: RESOURCE
Default Value:
Constraints:
Axmedis Editor User Manual

Resource Type: image  
Resource Format: jpeg gif png  
Ranges:  
Name: Mime Type  
Description: Test  
Parameter Type: STRING  
Default Value:  
Constraints:  
Result: Result  
Result Type: STRING  
Result Description: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

**RESIZE**

STRING Resize ( RESOURCE InputResource, INT32 Width, INT32 Height, BOOLEAN KeepAspectRatio, RESOURCE OutputResource )

Version: 1.0  
Description: Resizes an image  
Parameter List  
Name: InputResource  
Description: The Resource to be resized  
Parameter Type: RESOURCE  
Default Value:  
Constraints:  
Resource Type: image  
Resource Format: jpeg gif png  
Ranges:  
Name: Width  
Description: The new image width  
Parameter Type: INT32  
Default Value:  
Constraints:  
Name: Height  
Description: The new image height  
Parameter Type: INT32  
Default Value:  
Constraints:  
Name: KeepAspectRatio  
Description: Indicates to preserve image aspect ratio or not  
Parameter Type: BOOLEAN  
Default Value:  
Constraints:  
Name: OutputResource  
Description: Where the resized resource will be stored  
Parameter Type: RESOURCE  
Default Value:  
Constraints:  
Result: Result  
Result Type: STRING
**Result Description:** The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

---

**CONTRAST**

`STRING Contrast ( RESOURCE InputResource, INT32 AMOUNT, RESOURCE OutputResource )`

- **Version:** 1.0
- **Description:** Change image contrast

**Parameter List**

- **Name:** InputResource
  - **Description:** The Resource to be manipulated
  - **Parameter Type:** RESOURCE
  - **Default Value:**
  - **Constraints:**
    - **Resource Type:** image
    - **Resource Format:** jpeg gif png
  - **Ranges:**
- **Name:** AMOUNT
  - **Description:** The contrast amount
  - **Parameter Type:** INT32
  - **Default Value:**
  - **Constraints:**
- **Name:** OutputResource
  - **Description:** Where the manipulated resource will be stored
  - **Parameter Type:** RESOURCE
  - **Default Value:**
  - **Constraints:**

**Result:** Result

- **Result type:** STRING
- **Result Description:** The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

---

**EDGE**

`STRING Edge ( RESOURCE InputResource, INT32 ORDER, RESOURCE OutputResource )`

- **Version:** 1.0
- **Description:** Edge image (highlight edges in image). The radius is the radius of the pixel neighbourhood. Specify a radius of zero for automatic radius selection.

**Parameter List**

- **Name:** InputResource
  - **Description:** The Resource to be manipulated
  - **Parameter Type:** RESOURCE
  - **Default Value:**
  - **Constraints:**
    - **Resource Type:** image
    - **Resource Format:** jpeg gif png
  - **Ranges:**
- **Name:** ORDER
  - **Description:** The Order Edge
Parameter Type INT32
Default Value:
Constraints:
Name: OutputResource
Description: Where the manipulated resource will be stored
Parameter Type RESOURCE
Default Value:
Constraints:
Result: Result
Result type: STRING
Result Description: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

EMBOSS
STRING Emboss ( RESOURCE InputResource, INT32 RADIUS, INT32 SIGMA, RESOURCE OutputResource )

Version: 1.0
Description: Emboss image (highlight edges with 3D effect). The radius_ parameter specifies the radius of the Gaussian, in pixels, not counting the centre pixel. The sigma_ parameter specifies the standard deviation of the Laplacian, in pixels.

Parameter List
Name: InputResource
Description: The Resource to be manipulated
Parameter Type RESOURCE
Default Value:
Constraints:
Resource Type: image
Resource Format: jpeg gif png
Ranges:
Name: RADIUS
Description: The Radius Emboss
Parameter Type INT32
Default Value:
Constraints:
Name: SIGMA
Description: The sigma Emboss
Parameter Type INT32
Default Value:
Constraints:
Name: OutputResource
Description: Where the manipulated resource will be stored
Parameter Type RESOURCE
Default Value:
Constraints:
Result: Result
Result type: STRING
Result Description: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.
Blur
STRING Blur (RESOURCE InputResource, INT32 RADIUS, INT32 SIGMA, RESOURCE OutputResource )

Version: 1.0
Description: Blur image. The radius parameter specifies the radius of the Gaussian, in pixels, not counting the centre pixel. The sigma parameter specifies the standard deviation of the Laplacian, in pixels.

Parameter List
Name: InputResource
Description: The Resource to be manipulated
Parameter Type RESOURCE
Default Value:
Constraints:
Resource Type: image
Resource Format: jpeg gif png
Ranges:
Name: RADIUS
Description: The Radius Blur
Parameter Type INT32
Default Value:
Constraints:
Name: SIGMA
Description: The sigma Blur
Parameter Type INT32
Default Value:
Constraints:
Name: OutputResource
Description: Where the manipulated resource will be stored
Parameter Type RESOURCE
Default Value:
Constraints:
Result: Result
Result type: STRING
Result Description: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

GaussianBlur
STRING GaussianBlur (RESOURCE InputResource, INT32 RADIUS, INT32 SIGMA, RESOURCE OutputResource )

Version: 1.0
Description: GaussianBlur the image
Parameter List
Name: InputResource
Description: Gaussian blur image. The number of neighbour pixels to be included in the convolution mask is specified by 'width_'. For example, a width of one gives a (standard) 3x3 convolution mask. The standard deviation of the gaussian bell curve is specified by 'sigma'.
Parameter Type RESOURCE
Default Value:
Constraints:
Resource Type: image
**Resource Format:** jpeg gif png

**Ranges:**

**Name:** RADIUS
**Description:** The Radius GaussianBlur
**Parameter Type:** INT32
**Default Value:**
**Constraints:**

**Name:** SIGMA
**Description:** The sigma GaussianBlur
**Parameter Type:** INT32
**Default Value:**
**Constraints:**

**Name:** OutputResource
**Description:** Where the manipulated resource will be stored
**Parameter Type:** RESOURCE
**Default Value:**
**Constraints:**

**Result:** Result
**Result type:** STRING
**Result Description:** The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

---

**MEDIAN**

STRING Median ( RESOURCE InputResource, INT32 RADIUS, RESOURCE OutputResource )

**Version:** 1.0

**Description:** Median the image

**Parameter List**

**Name:** InputResource
**Description:** The Resource to be manipulated
**Parameter Type:** RESOURCE
**Default Value:**
**Constraints:**

**Name:** RADIUS
**Description:** The Radius Median
**Parameter Type:** INT32
**Default Value:**
**Constraints:**

**Name:** OutputResource
**Description:** Where the manipulated resource will be stored
**Parameter Type:** RESOURCE
**Default Value:**
**Constraints:**

**Result:** Result
**Result type:** STRING
**Result Description:** The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

---

**MIRROR**

STRING Mirror ( RESOURCE InputResource, BOOLEAN KeepDirection, RESOURCE OutputResource )

**Version:** 1.0

**Description:** Mirror the image

**Parameter List**

- **Name:** InputResource
  - **Description:** The Resource to be manipulated
  - **Parameter Type:** RESOURCE
  - **Default Value:**
  - **Constraints:**
    - **Resource Type:** image
    - **Resource Format:** jpeg, gif, png
  - **Ranges:**
- **Name:** KeepDirection
  - **Description:** The KeepDirection Mirror
  - **Parameter Type:** BOOLEAN
  - **Default Value:**
  - **Constraints:**
- **Name:** OutputResource
  - **Description:** Where the manipulated resource will be stored
  - **Parameter Type:** RESOURCE
  - **Default Value:**
  - **Constraints:**

**Result:** Result

- **Result type:** STRING
- **Result Description:** The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

---

**NOISE**

STRING Noise ( RESOURCE InputResource, INT32 TYPE, RESOURCE OutputResource )

**Version:** 1.0

**Description:** Noise the image

**Parameter List**

- **Name:** InputResource
  - **Description:** The Resource to be manipulated
  - **Parameter Type:** RESOURCE
  - **Default Value:**
  - **Constraints:**
    - **Resource Type:** image
    - **Resource Format:** jpeg, gif, png
  - **Ranges:**
- **Name:** TYPE
  - **Description:** The Type Noise
  - **Parameter Type:** INT32
**Default Value:**

**Constraints:**

**Name:** OutputResource

**Description:** Where the manipulated resource will be stored

**Parameter Type** RESOURCE

**Default Value:**

**Constraints:**

**Result:** Result

**Result type:** STRING

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

---

**DESPECKLE**

STRING Despeckle ( RESOURCE InputResource, RESOURCE OutputResource )

**Version:** 1.0

**Description:** Despeckle image (reduce speckle noise)

**Parameter List**

**Name:** InputResource

**Description:** The Resource to be manipulated

**Parameter Type** RESOURCE

**Default Value:**

**Constraints:**

**Name:** OutputResource

**Description:** Where the manipulated resource will be stored

**Parameter Type** RESOURCE

**Default Value:**

**Constraints:**

**Result:** Result

**Result type:** STRING

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

---

**EQUALIZE**

STRING Equalize ( RESOURCE InputResource, RESOURCE OutputResource )

**Version:** 1.0

**Description:** Equalize image (histogram equalization)

**Parameter List**

**Name:** InputResource

**Description:** The Resource to be manipulated

**Parameter Type** RESOURCE

**Default Value:**

**Constraints:**

**Name:** OutputResource

**Description:** Where the manipulated resource will be stored

**Parameter Type** RESOURCE

**Default Value:**

**Constraints:**

**Result:** Result

**Result type:** STRING

**Result Description:** The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.
AXMEDIS Editor User Manual

**ENHANCE**

**STRING**  
Enhance ( RESOURCE InputResource, RESOURCE OutputResource )

**Version**: 1.0

**Description**: Enhance image (minimize noise)

**Parameter List**

- **Name**: InputResource
  - **Description**: The Resource to be manipulated
  - **Parameter Type**: RESOURCE
  - **Default Value**:  
  - **Constraints**:  
    - **Resource Type**: image
    - **Resource Format**: jpeg gif png

- **Name**: OutputResource
  - **Description**: Where the manipulated resource will be stored
  - **Parameter Type**: RESOURCE
  - **Default Value**:  
  - **Constraints**:  

**Result**: Result

- **Result type**: STRING

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

**EXTRACTCHANNEL**

**STRING**  
ExtractChannel ( RESOURCE InputResource, INT32 CHANNEL, RESOURCE OutputResource )

**Version**: 1.0

**Description**: ExtractChannel the image

**Parameter List**

- **Name**: InputResource
  - **Description**: The Resource to be manipulated
  - **Parameter Type**: RESOURCE
  - **Default Value**:  
  - **Constraints**:  
    - **Resource Type**: image
    - **Resource Format**: jpeg gif png

- **Name**: OutputResource
  - **Description**: Where the manipulated resource will be stored
  - **Parameter Type**: RESOURCE
  - **Default Value**:  
  - **Constraints**:  

**Result**: Result

- **Result type**: STRING

**Result Description**: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.
**Name:** CHANNEL  
**Description:** The Channel ExtractChannel  
**Parameter Type:** INT32  
**Default Value:**  
**Constraints:**

**Name:** OutputResource  
**Description:** Where the manipulated resource will be stored  
**Parameter Type:** RESOURCE  
**Default Value:**  
**Constraints:**

**Result:** Result  
**Result type:** STRING  
**Result Description:** The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

---

**GRAYSCALE**

**Description:** Grayscale the image  
**Parameter List**  
**Name:** InputResource  
**Description:** The Resource to be manipulated  
**Parameter Type:** RESOURCE  
**Default Value:**  
**Constraints:**  
- **Resource Type:** image  
- **Resource Format:** jpeg, gif, png  
**Ranges:**  
**Name:** OutputResource  
**Description:** Where the manipulated resource will be stored  
**Parameter Type:** RESOURCE  
**Default Value:**  
**Constraints:**

**Result:** Result  
**Result type:** STRING  
**Result Description:** The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

---

**MAGNIFY**

**Description:** Magnify image by integral size  
**Parameter List**  
**Name:** InputResource  
**Description:** The Resource to be manipulated  
**Parameter Type:** RESOURCE  
**Default Value:**
Constraints:
- **Resource Type**: image
- **Resource Format**: jpeg, gif, png
Ranges:
- **Name**: OutputResource
  - **Description**: Where the manipulated resource will be stored
- **Parameter Type**: RESOURCE
- **Default Value**: 
- **Constraints**: 
- **Result**: Result
  - **Result type**: STRING
  - **Result Description**: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

MINIFY

**STRING** Minify ( RESOURCE InputResource, RESOURCE OutputResource )

**Version**: 1.0
**Description**: Reduce image by integral size

**Parameter List**
- **Name**: InputResource
  - **Description**: The Resource to be manipulated
  - **Parameter Type**: RESOURCE
  - **Default Value**: 
  - **Constraints**: 
    - **Resource Type**: image
    - **Resource Format**: jpeg, gif, png
  
- **Name**: OutputResource
  - **Description**: Where the manipulated resource will be stored
  - **Parameter Type**: RESOURCE
  - **Default Value**: 
  - **Constraints**: 
  - **Result**: Result
    - **Result type**: STRING
    - **Result Description**: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

MODULATE

**STRING** Modulate ( RESOURCE InputResource, INT32 BRIGHTNESS, INT32 SATURATION, INT32 HUE, RESOURCE OutputResource )

**Version**: 1.0
**Description**: Modulate percent hue, saturation, and brightness of an image. Modulation of saturation and brightness is as a ratio of the current value (1.0 for no change). Modulation of hue is an absolute rotation of -180 degrees to +180 degrees from the current position corresponding to an argument range of 0 to 2.0 (1.0 for no change).

**Parameter List**
- **Name**: InputResource
**Description:** The Resource to be manipulated
**Parameter Type** RESOURCE
**Default Value:**
**Constraints:**
  - **Resource Type:** image
  - **Resource Format:** jpeg gif png
**Ranges:**
**Name:** BRIGHTNESS
  - **Description:** Brightness modulate
  - **Parameter Type** INT32
  - **Default Value:**
  - **Constraints:**
**Name:** SATURATION
  - **Description:** Saturation modulate
  - **Parameter Type** INT32
  - **Default Value:**
  - **Constraints:**
**Name:** HUE
  - **Description:** Hue modulate
  - **Parameter Type** INT32
  - **Default Value:**
  - **Constraints:**
**Name:** OutputResource
  - **Description:** Where the manipulated resource will be stored
  - **Parameter Type** RESOURCE
  - **Default Value:**
  - **Constraints:**
**Result:** Result
  - **Result type:** STRING
  - **Result Description:** The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

---

**MONOCHROME**

STRING Monochrome ( RESOURCE InputResource, RESOURCE OutputResource )

**Version:** 1.0
**Description:** Monochrome the image

**Parameter List**
**Name:** InputResource
  - **Description:** The Resource to be manipulated
  - **Parameter Type** RESOURCE
  - **Default Value:**
  - **Constraints:**
    - **Resource Type:** image
    - **Resource Format:** jpeg gif png
  - **Ranges:**
**Name:** OutputResource
  - **Description:** Where the manipulated resource will be stored
  - **Parameter Type** RESOURCE
AXMEDIS Editor User Manual

**Default Value:**

**Constraints:**

**Result:** Result

**Result type:** STRING

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

**NEGATE**

STRING Negate ( RESOURCE InputResource, BOOLEAN GRAYSCALE, RESOURCE OutputResource )

**Version:** 1.0

**Description:** Negate colours in image. Replace every pixel with its complementary color (white becomes black, yellow becomes blue, etc.). Set grayscale to only negate grayscale values in image.

**Parameter List**

- **Name:** InputResource
  - **Description:** The Resource to be manipulated
  - **Parameter Type:** RESOURCE
  - **Default Value:**
  - **Constraints:**
    - **Resource Type:** image
    - **Resource Format:** jpeg gif png

- **Name:** GRAYSCALE
  - **Description:** Where the manipulated resource will be stored
  - **Parameter Type:** BOOLEAN
  - **Default Value:**
  - **Constraints:**

- **Name:** OutputResource
  - **Description:** Where the manipulated resource will be stored
  - **Parameter Type:** RESOURCE
  - **Default Value:**
  - **Constraints:**

**Result:** Result

**Result type:** STRING

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

**NORMALIZE**

STRING Normalize ( RESOURCE InputResource, RESOURCE OutputResource )

**Version:** 1.0

**Description:** Normalize image (increase contrast by normalizing the pixel values to span the full range of colour values)

**Parameter List**

- **Name:** InputResource
  - **Description:** The Resource to be manipulated
  - **Parameter Type:** RESOURCE
  - **Default Value:**
  - **Constraints:**
**Resource Type:** image  
**Resource Format:** jpeg gif png  

**Ranges:**

**Name:** OutputResource  
**Description:** Where the manipulated resource will be stored  
**Parameter Type** RESOURCE  
**Default Value:**  
**Constraints:**

**Result:** Result  
**Result type:** STRING  
**Result Description:** The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

---

**OILPAINT**

```c
STRING OilPaint ( RESOURCE InputResource, INT32 RADIUS, RESOURCE OutputResource )
```

**Version:** 1.0  
**Description:** Oilpaint image (image looks like oil painting)  
**Parameter List**

**Name:** InputResource  
**Description:** The Resource to be manipulated  
**Parameter Type** RESOURCE  
**Default Value:**  
**Constraints:**

**Name:** RADIUS  
**Description:** the radius OilPaint  
**Parameter Type** INT32  
**Default Value:**  
**Constraints:**

**Name:** OutputResource  
**Description:** Where the manipulated resource will be stored  
**Parameter Type** RESOURCE  
**Default Value:**  
**Constraints:**

**Result:** Result  
**Result type:** STRING  
**Result Description:** The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

---

**QUALITY**

```c
STRING Quality ( RESOURCE InputResource, INT32 LEVEL, RESOURCE OutputResource )
```

**Version:** 1.0  
**Description:** JPEG/MIFF/PNG compression level (default 75).  
**Parameter List**

**Name:** InputResource
**Description**: The Resource to be manipulated  
**Parameter Type** RESOURCE  
**Default Value**:  
**Constraints**:  
  - **Resource Type**: image  
  - **Resource Format**: jpeg gif png  
**Ranges**:  
**Name**: LEVEL  
**Description**: the quality of the compress level  
**Parameter Type** INT32  
**Default Value**:  
**Constraints**:  
**Name**: OutputResource  
**Description**: Where the manipulated resource will be stored  
**Parameter Type** RESOURCE  
**Default Value**:  
**Constraints**:  
**Result**: Result  
**Result type**: STRING  
**Result Description**: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

**QUANTIZE**

STRING Quantize ( RESOURCE InputResource, INT32 NCOLORS, RESOURCE OutputResource )

**Version**: 1.0  
**Description**: Preferred number of colours in the image. The actual number of colours in the image may be less than your request, but never more. Images with less unique colours than specified with this option will have any duplicate or unused colours removed.

**Parameter List**  
**Name**: InputResource  
**Description**: The Resource to be manipulated  
**Parameter Type** RESOURCE  
**Default Value**:  
**Constraints**:  
  - **Resource Type**: image  
  - **Resource Format**: jpeg gif png  
**Ranges**:  
**Name**: NCOLORS  
**Description**: the number of color  
**Parameter Type** INT32  
**Default Value**:  
**Constraints**:  
**Name**: OutputResource  
**Description**: Where the manipulated resource will be stored  
**Parameter Type** RESOURCE  
**Default Value**:  
**Constraints**:  
**Result**: Result
Result type: STRING
Result Description: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

RAISE
STRING Raise ( RESOURCE InputResource, INT32 WIDTH, INT32 HEIGHT, INT32 XOFFSET, INT32 YOFFSET, BOOLEAN RISED, RESOURCE OutputResource )

Version: 1.0
Description: Raise image (lighten or darken the edges of an image to give a 3-D raised or lowered effect)

Parameter List
Name: InputResource
   Description: The Resource to be manipulated
   Parameter Type: RESOURCE
   Default Value:
   Constraints:
       Resource Type: image
       Resource Format: jpeg gif png
   Ranges:
Name: WIDTH
   Description: The width is parts of the geometry specification are measured in pixels
   Parameter Type: INT32
   Default Value:
   Constraints:
Name: HEIGHT
   Description: The height is parts of the geometry specification are measured in pixels
   Parameter Type: INT32
   Default Value:
   Constraints:
Name: XOFFSET
   Description: The left edge of the object is to be placed xoffset pixels in from the left edge of the image.
   Parameter Type: INT32
   Default Value:
   Constraints:
Name: YOFFSET
   Description: The top edge of the object is to be yoffset pixels below the top edge of the image.
   Parameter Type: INT32
   Default Value:
   Constraints:
Name: RISED
   Description: raisedFlag
   Parameter Type: BOOLEAN
   Default Value:
   Constraints:
Name: OutputResource
   Description: Where the manipulated resource will be stored
   Parameter Type: RESOURCE
   Default Value:
   Constraints:
**Result:** Result
**Result type:** STRING
**Result Description:** The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

---

### REDUCENOISE

STRING ReduceNoise ( RESOURCE InputResource, INT32 ORDER, RESOURCE OutputResource )

**Version:** 1.0
**Description:** Reduce noise in image using a noise peak elimination filter.

**Parameter List**

- **Name:** InputResource
  - **Description:** The Resource to be manipulated
  - **Parameter Type:** RESOURCE
  - **Default Value:**
  - **Constraints:**
    - **Resource Type:** image
    - **Resource Format:** jpeg gif png
  - **Ranges:**
- **Name:** ORDER
  - **Description:** order
  - **Parameter Type:** INT32
  - **Default Value:**
  - **Constraints:**
- **Name:** OutputResource
  - **Description:** Where the manipulated resource will be stored
  - **Parameter Type:** RESOURCE
  - **Default Value:**
  - **Constraints:**

**Result:** Result
**Result type:** STRING
**Result Description:** The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

---

### REPLACE

STRING Replace ( RESOURCE InputResource, INT32 R1, INT32 G1, INT32 B1, INT32 R2, INT32 G2, INT32 B2, RESOURCE OutputResource )

**Version:** 1.0
**Description:** Replace the image

**Parameter List**

- **Name:** InputResource
  - **Description:** The Resource to be manipulated
  - **Parameter Type:** RESOURCE
  - **Default Value:**
  - **Constraints:**
    - **Resource Type:** image
    - **Resource Format:** jpeg gif png
  - **Ranges:**
Name: R1  
  Description: r1  
  Parameter Type INT32  
  Default Value:  
  Constraints:  
Name: G1  
  Description: g1  
  Parameter Type INT32  
  Default Value:  
  Constraints:  
Name: B1  
  Description: b1  
  Parameter Type INT32  
  Default Value:  
  Constraints:  
Name: R2  
  Description: r2  
  Parameter Type INT32  
  Default Value:  
  Constraints:  
Name: G2  
  Description: g2  
  Parameter Type INT32  
  Default Value:  
  Constraints:  
Name: B2  
  Description: b2  
  Parameter Type INT32  
  Default Value:  
  Constraints:  
Name: OutputResource  
  Description: Where the manipulated resource will be stored  
  Parameter Type RESOURCE  
  Default Value:  
  Constraints:  
Result: Result  
  Result type: STRING  
  Result Description: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

FLOODFILL  

Version: 1.0  
Description: Flood-fill texture across pixels that match the colour of the target pixel and are neighbours of the target pixel. It uses current fuzz setting when determining colour match.

Parameter List  
  Name: InputResource
**Description:** The Resource to be manipulated  
**Parameter Type:** RESOURCE  
**Default Value:**  
**Constraints:**  
  - **Resource Type:** image  
  - **Resource Format:** jpeg, gif, png  
**Ranges:**  
**Name:** X  
  - **Description:** x  
  - **Parameter Type:** INT32  
  - **Default Value:**  
  - **Constraints:**  
**Name:** Y  
  - **Description:** y  
  - **Parameter Type:** INT32  
  - **Default Value:**  
  - **Constraints:**  
**Name:** B  
  - **Description:** b  
  - **Parameter Type:** INT32  
  - **Default Value:**  
  - **Constraints:**  
**Name:** R  
  - **Description:** r  
  - **Parameter Type:** INT32  
  - **Default Value:**  
  - **Constraints:**  
**Name:** G  
  - **Description:** g  
  - **Parameter Type:** INT32  
  - **Default Value:**  
  - **Constraints:**  
**Name:** OutputResource  
  - **Description:** Where the manipulated resource will be stored  
  - **Parameter Type:** RESOURCE  
  - **Default Value:**  
  - **Constraints:**  
**Result:** Result  
  - **Result type:** STRING  
  - **Result Description:** The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

**ROLL**  
STRING Roll ( RESOURCE InputResource, INT32 X, INT32 Y, RESOURCE OutputResource )

**Version:** 1.0  
**Description:** Roll image (rolls image vertically and horizontally) by specified number of columns and rows  
**Parameter List**  
  - **Name:** InputResource
**Description:** The Resource to be manipulated  
**Parameter Type:** RESOURCE  
**Default Value:**  
**Constraints:**  
  - **Resource Type:** image  
  - **Resource Format:** jpeg gif png  
**Ranges:**  
**Name:** X  
  **Description:** x  
  **Parameter Type:** INT32  
  **Default Value:**  
  **Constraints:**  
**Name:** Y  
  **Description:** y  
  **Parameter Type:** INT32  
  **Default Value:**  
  **Constraints:**  
**Name:** OutputResource  
  **Description:** Where the manipulated resource will be stored  
  **Parameter Type:** RESOURCE  
  **Default Value:**  
  **Constraints:**  
**Result:** Result  
  **Result type:** STRING  
  **Result Description:** The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

**ROTATE**

STRING Rotate ( RESOURCE InputResource, INT32 ANGLE, RESOURCE OutputResource )

**Version:** 1.0  
**Description:** Rotate image counter-clockwise by specified number of degrees.  
**Parameter List**
  **Name:** InputResource  
    **Description:** The Resource to be manipulated  
    **Parameter Type:** RESOURCE  
    **Default Value:**  
    **Constraints:**  
      - **Resource Type:** image  
      - **Resource Format:** jpeg gif png  
    **Ranges:**  
  **Name:** ANGLE  
    **Description:** Number of the degrees  
    **Parameter Type:** INT32  
    **Default Value:**  
    **Constraints:**  
  **Name:** OutputResource  
    **Description:** Where the manipulated resource will be stored  
    **Parameter Type:** RESOURCE
SCALE

STRING Scale ( RESOURCE InputResource, INT32 WIDTH, INT32 HEIGHT, INT32 MODE, RESOURCE OutputResource )

Version: 1.0
Description: Resize image by using simple ratio algorithm
Parameter List
   Name: InputResource
     Description: The Resource to be manipulated
     Parameter Type RESOURCE
     Default Value:
     Constraints:
       Resource Type: image
       Resource Format: jpeg gif png
   Name: WIDTH
     Description: Width
     Parameter Type INT32
     Default Value:
     Constraints:
   Name: HEIGHT
     Description: Height
     Parameter Type INT32
     Default Value:
     Constraints:
   Name: MODE
     Description: Mode
     Parameter Type INT32
     Default Value:
     Constraints:
   Name: OutputResource
     Description: Where the manipulated resource will be stored
     Parameter Type RESOURCE
     Default Value:
     Constraints:
Result: Result
   Result type: STRING
   Result Description: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

SHEAR

STRING Shear ( RESOURCE InputResource, INT32 XSHEAR, INT32 Yshear, RESOURCE OutputResource )
**Version**: 1.0  

**Description**: Shear image (create parallelogram by sliding image by X or Y axis). Shearing slides one edge of an image along the X or Y axis, creating a parallelogram. An X direction shear slides an edge along the X axis, while a Y direction shear slides an edge along the Y axis. The amount of the shear is controlled by a shear angle. For X direction shears, x degrees is measured relative to the Y axis, and similarly, for Y direction shears y degrees is measured relative to the X axis. Empty triangles left over from shearing the image are filled with the colour defined as borderColor.

**Parameter List**

- **Name**: InputResource  
  **Description**: The Resource to be manipulated  
  **Parameter Type**: RESOURCE  
  **Default Value**:  
  **Constraints**:  
  - Resource Type: image  
  - Resource Format: jpeg, gif, png  
  **Ranges**:  
- **Name**: XSHEAR  
  **Description**: XSHEAR  
  **Parameter Type**: INT32  
  **Default Value**:  
  **Constraints**:  
- **Name**: Yshear  
  **Description**: Yshear  
  **Parameter Type**: INT32  
  **Default Value**:  
  **Constraints**:  
- **Name**: OutputResource  
  **Description**: Where the manipulated resource will be stored  
  **Parameter Type**: RESOURCE  
  **Default Value**:  
  **Constraints**:  

**Result**

- **Result type**: STRING  
  **Result Description**: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

### SHADE

```plaintext```
STRING Shade ( RESOURCE InputResource, INT32 AZIMUTH, INT32 ELEVATION, BOOLEAN COLOR, RESOURCE OutputResource )
```

**Version**: 1.0  

**Description**: Shade image using distant light source. Specify azimuth_ and elevation_ as the position of the light source. By default, the shading results as a grayscale image. Set colorShading_ to true to shade the red, green, and blue components of the image.

**Parameter List**

- **Name**: InputResource  
  **Description**: The Resource to be manipulated  
  **Parameter Type**: RESOURCE  
  **Default Value**:  
  **Constraints**:  

Resource Type: image
Resource Format: jpeg gif png

Names:
Name: AZIMUTH
Description: AZIMUTH
Parameter Type: INT32
Default Value:
Constraints:
Name: ELEVATION
Description: ELEVATION
Parameter Type: INT32
Default Value:
Constraints:
Name: COLOR
Description: COLOR
Parameter Type: BOOLEAN
Default Value:
Constraints:
Name: OutputResource
Description: Where the manipulated resource will be stored
Parameter Type: RESOURCE
Default Value:
Constraints:
Result: Result
Result type: STRING
Result Description: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

```
SPREAD
STRING Spread ( RESOURCE InputResource, INT32 AMOUNT, RESOURCE OutputResource )
```

Version: 1.0
Description: Spread pixels randomly within image by specified amount.

Parameter List
Name: InputResource
Description: The Resource to be manipulated
Parameter Type: RESOURCE
Default Value:
Constraints:
Resource Type: image
Resource Format: jpeg gif png
Ranges:
Name: AMOUNT
Description: AMOUNT
Parameter Type: INT32
Default Value:
Constraints:
Name: OutputResource
Description: Where the manipulated resource will be stored
Parameter Type: RESOURCE
Default Value:
Constraints:
Result: Result
Result type: STRING
Result Description: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

SETOPACITY
STRING SetOpacity ( RESOURCE InputResource, INT32 LEVEL, RESOURCE OutputResource )

Version: 1.0
Description: Set the opacity of the image.
Parameter List
Name: InputResource
Description: The Resource to be manipulated
Parameter Type: RESOURCE
Default Value:
Constraints:
Resource Type: image
Resource Format: jpeg gif png
Ranges:
Name: LEVEL
Description: LEVEL
Parameter Type: INT32
Default Value:
Constraints:
Name: OutputResource
Description: Where the manipulated resource will be stored
Parameter Type: RESOURCE
Default Value:
Constraints:
Result: Result
Result type: STRING
Result Description: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

SUBIMAGE

Version: 1.0
Description: SubImage image.
Parameter List
Name: InputResource
Description: The Resource to be manipulated
Parameter Type: RESOURCE
Default Value:
Constraints:
Resource Type: image
Resource Format: jpeg gif png
Ranges:
Name: X
Description: x coordinate of the top-level corner of the rectangle
Parameter Type INT32
Default Value:
Constraints:
Name: Y
Description: y coordinate of the top-level corner of the rectangle
Parameter Type INT32
Default Value:
Constraints:
Name: WIDTH
Description: Width member
Parameter Type INT32
Default Value:
Constraints:
Name: HEIGHT
Description: Height member
Parameter Type INT32
Default Value:
Constraints:
Name: OutputResource
Description: Where the manipulated resource will be stored
Parameter Type RESOURCE
Default Value:
Constraints:
Result: Result
Result type: STRING
Result Description: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

GETINFO
STRING GetInfo ( RESOURCE InputResource, INT32 WIDTH, INT32 HEIGHT )

Version: 1.0
Description: Return the size of the image.
Parameter List
Name: InputResource
Description: The Resource under analysis
Parameter Type RESOURCE
Default Value:
Constraints:
Resource Type: image
Resource Format: jpeg gif png
Ranges:
Name: WIDTH
Description: The width of the Image
Parameter Type INT32
Default Value:
Constraints:
Name: HEIGHT
Description: The height of the Image
Parameter Type INT32
Default Value:
Constraints:
Result: Result
Result type: STRING
Result Description: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

**SETMASKCOLOUR**


Version: 1.0
Description: Sets the colour
Parameter List
Name: InputResource
   Description: The Resource to be manipulated
   Parameter Type RESOURCE
   Default Value:
   Constraints:
      Resource Type: image
      Resource Format: jpeg gif png
Ranges:
Name: R
   Description: Red
   Parameter Type INT32
   Default Value:
   Constraints:
Name: G
   Description: Green
   Parameter Type INT32
   Default Value:
   Constraints:
Name: B
   Description: Blue
   Parameter Type INT32
   Default Value:
   Constraints:
Name: OutputResource
   Description: Where the manipulated resource will be stored
   Parameter Type RESOURCE
   Default Value:
   Constraints:
Result: Result
Result type: STRING
**Result Description**: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

**PASTE**

```plaintext
STRING Paste ( RESOURCE InputResource1, RESOURCE InputResource2, INT32 X, INT32 Y, INT32 COMPOSE, RESOURCE OutputResource )
```

**Version**: 1.0

**Description**: Paste image

**Parameter List**

- **Name**: InputResource1
  - **Description**: The Resource to be manipulated
  - **Parameter Type**: RESOURCE
  - **Default Value**: 
  - **Constraints**: 
    - **Resource Type**: image
    - **Resource Format**: jpeg, gif, png
  - **Ranges**: 

- **Name**: InputResource2
  - **Description**: The Resource paste
  - **Parameter Type**: RESOURCE
  - **Default Value**: 
  - **Constraints**: 
    - **Resource Type**: image
    - **Resource Format**: jpeg, gif, png
  - **Ranges**: 

- **Name**: X
  - **Description**: X
  - **Parameter Type**: INT32
  - **Default Value**: 
  - **Constraints**: 

- **Name**: Y
  - **Description**: Y
  - **Parameter Type**: INT32
  - **Default Value**: 
  - **Constraints**: 

- **Name**: COMPOSE
  - **Description**: Compose
  - **Parameter Type**: INT32
  - **Default Value**: 
  - **Constraints**: 

- **Name**: OutputResource
  - **Description**: Where the manipulated resource will be stored
  - **Parameter Type**: RESOURCE
  - **Default Value**: 
  - **Constraints**: 

**Result**: Result

**Result type**: STRING
Result Description: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.

TEST
RESOURCE Test ( RESOURCE InputResource, AXOM Axom )
  Version: 1.0
  Description: Test an image
Parameter List
  Name: InputResource
    Description: The Resource to be tested
    Parameter Type: RESOURCE
    Default Value: 
    Constraints: 
  Name: Axom
    Description: The object
    Parameter Type: AXOM
    Default Value: 
    Constraints: 
Result: Result
  Result type: RESOURCE
  Result Description: The result of conversion, SUCCESS if ok, ERROR followed by a message in case of error.
REFERENCES AND LINKS

AXMEDIS TUTORIALS

- General Tutorial and Overview (November 2007, Barcelona, Spain)
  - Video on [part 1](http://www.axmedis.org/documenti/view_documenti.php?doc_id=3716)
  - Video on [part 2](http://www.axmedis.org/documenti/view_documenti.php?doc_id=2652)
  - Video on [part 3](http://www.axmedis.org/documenti/view_documenti.php?doc_id=3767)
  - Video on [part 4](http://www.axmedis.org/documenti/view_documenti.php?doc_id=3841)
  - Video on [part 5](http://www.axmedis.org/documenti/view_documenti.php?doc_id=3842)
  - Video on [part 6](http://www.axmedis.org/documenti/view_documenti.php?doc_id=21)

- Content Production Tutorial (AXMEDIS 2007 Conference)

- Content Distribution Tutorial (AXMEDIS 2006 Conference)

- Content Processing Tutorial (AXMEDIS 2006 Conference)

- Workflow Tutorial (AXMEDIS 2006 Conference)

- AXMEDIS general overview and content production tutorial, March 2008

AXMEDIS TOOLS FOR FREE DOWNLOAD

- AXMEDIS content production tools include:

Free Download of AXMEDIS Content Production Tools (editor and GRID AXCP tools, PnP, DRM editor, etc.), all what you need to create AXMEDIS objects and process any kind of content automatically: SMIL, HTML, MPEG-21, content adaption,, fingerprint, crawling, indexing, cms, search, retrieval, control of P2P, etc. and much more. See documentation included

- AXMEDIS players for PC:

AXMEDIS PC player version 1.2 January 2008. Free download, AXMEDIS player, MPEG-21 player, cross media player, SMIL, HTML, MPEG-4, etc

- AXMEDIS multiskin player for PC:

AXMEDIS MultiSkin PC player version 1.2 January 2008. Free download, AXMEDIS player, MPEG-21 player, cross media player, SMIL, HTML, MPEG-4, with different skins available.

- AXMEDIS Active X Player for PC for Web Pages, AXMEDIS .Net Player, MPEG-21 player, SMIL, HTML, MPEG-4, cross media, more than 200 file formats:

- AXMEDIS player plus EUTELSAT OPENSKY client integrated

- AXEPT tools: P2P client tool for establishing connection with the AXMEDIS P2P B2B network as Business User:

- AXMEDIA: P2P client tool for establishing connection with the AXMEDIS P2P B2B network as final users:

- AXMEDIS PDA player for Windows Mobiles 5 and 6:

It is capable to play AXMEDIS objects based on SMIL, HTML, video, audio, MPEG-4 files, etc. AXMEDIS PDA player for AXMEDIS MPEG-21 content including resources with presentations layer based on MPEG-4, HTML and SMIL

- Unzip the file, copy the CAB file and execute it on the PDA

- Collection of Objects for AXMEDIS player for PDA (Jan 2008):
AXMEDIS TECHNICAL NOTES

- **AXMEDIS Content Model and Tools, Authoring Tools, Players for MPEG-21, PC, PDA, Mobile, STB, PVR, HDR, etc.** (in English)
- **AXMEDIS Content Model and Tools, Authoring Tools, Players for MPEG-21, PC, PDA, Mobile, STB, PVR, HDR, etc.** (in Italian)
- **AXMEDIS Content Processing GRID all features listed** (in English)
- **AXMEDIS Content Processing GRID Tutte le caratteristiche descritte** (in Italian)
- **AXMEDIS P2P Controlled network all features listed with cases** (in English)
- **AXMEDIS P2P Controlled network tutte le caratteristiche, con alcune casi** (in Italian)
- **AXMEDIS DRM, MPEG-21 DRM, Interoperable DRM** (in English)
- **AXMEDIS DRM, MPEG-21 DRM, DRM interoperabile** (in Italian)
- Technical note on how to integrate the AXMEDIS DRM into an e-commerce portal and content distribution solution for content on demand and subscription
- Come integrare AXMEDIS DRM in un portale per la distribuzione di contenuti digitali (in Italian)
- **AXMEDIS Show Case, AXMEDIS Mpeg-21 Content distribution via satellite data broadcast, EUTELSAT OPENSKY**
- Technical note on the ELION AXMEDIS content on demand trial and solution, how to exploit AXMEDIS framework to create a cross media content distribution with DRM and automated production, and connection with P2P
- Technical note on the TEO IPTV AXMEDIS trial and solution, how to exploit AXMEDIS framework to create an IPTV with DRM and automated production, and connection with P2P.

AXMEDIS SOLUTIONS

- Content Management Automation, AXCP:
- AXMEDIS controlled P2P:
- AXMEDIS production tools and players:
- AXMEDIS DRM, MPEG-21 DRM:

AXMEDIS SHOWCASES

- **Content Distribution to Licensed Domains via DVB-T and P2P** (BBC)
- Protected Video on Demand Distribution via P2P toward PC (Tiscali)
- Protected Video on Demand (VOD) Distribution to PC (ELION)
- Content Distribution via Satellite Data Broadcast (DVB-S) to PC and STB (EUTELSAT)
- Content Distribution to Kiosks (ILABS)
- Video on Demand (VOD) Distribution to Set Top Box (TEO)
- Content Posting Tool, for Final User content production/publication/DRM (SIAE)
- Variazioni: Enrichment of Cultural Content
- **AXMEDIS Content and Tools: Automatic Production**
- **AXMEDIS Controlled P2P Network**

AXMEDIS FRAMEWORK SPECIFICATION

AXMEDIS External Processing Algorithms:
AXMEDIS CMS Crawling capabilities:
AXMEDIS Database and query support:
AXMEDIS AXEPTool and AXMedia Tools:
AXMEDIS Programme and Publication Tools:
AXMEDIS Workflow Tools:
AXMEDIS Certifier and Supervisor and networks of AXCS
AXMEDIS Protection Support
AXMEDIS Accounting Support:

AXMEDIS reports on basic enabling technologies
Content Model and Managing, MPEG-21, authoring, etc.
Content indexing and querying:
Content processing, Composition and Formatting
Content sharing and Production on P2P:
Content Protection and Supervision
Content Distribution via Internet
Content Distribution via Mobile
Content Distribution via Satellite data broadcast
Usability issues
AXMEDIS vs DMP MPEG21 Analysis
AXMEDIS Framework Infrastructure, guidelines and some tools
AXMEDIS Framework Validation and integration

Basic knowledge reports
User requirements
Use Cases
Test Case

Content Modeling and Test Cases
Content Aspect Specification
Content Aspect Specification Appendix
Content for Test Cases and Validation
Content Selection Guidelines
Multilingual Guidelines and Technical Solutions
AXMEDIS FRAMEWORK DEMONSTRATORS, CASES, TRIALS, FOR DISTRIBUTION ETC.

- requirements and use cases of AXMEDIS ELTEO of the content distribution for DVB-T to STB of Telecom Lithuania, and content distribution of Telecom Estonia
- requirements and use cases of the 4HOME take up, demonstrators of BBC, TI, SDAE, including domains, AXMEDIS for broadcasting, and OMA integration and distribution
- Integrated prototype: automated content production and formatting:
- Integrated Distribution on demand via Internet
- Integrated distribution via satellite data broadcast:
- Integrated distribution towards mobiles:
- Integrated distribution towards PDA via Kiosks:
- Content Posting Portal, Content Posting for Final User publication, SIAE Trial presentation:
- VARIAZIONI project portal: http://www.variazioniproject.org/

BROCHURES AND PRESS CUTTING (A PART)

- AXMEDIS Project Brochure
- Digital Media in Italy presentation http://www.AXMEDIS.org/documenti/view_documenti.php?doc_id=1669

OTHER REFERENCES

- Open Mobile Alliance (OMA), http://www.openmobilealliance.com/
- OMA DRM Rights Expression Language version 2 (OMA DRM REL v.2), http://www.openmobilealliance.com/