Automating Production of Cross Media Content for Multi-channel Distribution

DE8.4.1
Editorial Format Guidelines and Basic Examples

Version: 1.1
Date: 07/09/2005
Responsible: XIM (laurence@xim.co.uk)

Project Number: IST-2001-37168
Project Title: Automating Production of Cross Media Content for Multichannel Distribution
Deliverable Type: Report
Visible to User Groups: Yes
Visible to Affiliated: Yes
Visible to the Public: Yes.

Deliverable Number: DE8.4.1
Contractual Date of Delivery: 31/08/2005
Actual Date of Delivery: 07/09/2005
Title of Deliverable: AXMEDIS Editorial Format Guidelines and basic examples
Work-Package contributing to the Deliverable: WP8
Task contributing to the Deliverable: WP8.4
Nature of the Deliverable: Report
Author(s): XIM, ILABS, DSI, SEJER, EUTELSAT

Abstract:
This document provides guidelines and basic examples for the development of formats for use with the AXMEDIS engines in the automated production of content.

Keyword List:
Editing, Formats, Layouts, Templates, Channels, Content

Table of Content

1 Executive Summary and Report Scope (XIM)

The need for high quality content in AXMEDIS for demonstration and testing is clear. A substantial amount of content is available from the partners in the project, which will be documented in DE8.1.1, evaluated for suitability via the criteria in DE8.2.1 and should also meet the test case requirements in DE2.2.1. However, before this content can be delivered to the end user it must be formatted in a manner that is:

- Suitable for the distribution channel
- Stylistically professional and high quality.

The purpose of this deliverable, DE8.4.1 is to provide the first version of guidelines for designers and content creators of how to create *editorial formats* in order for the AXMEDIS formatting engine to automatically format content. The specification and structure of the formatting rules will be defined in WP4, but the focus of this deliverable is to describe how best to use these editorial formats, and to provide actual examples suited to the various distribution channels featured in AXMEDIS. In Month 22, an updated version of these guidelines will be published as DE8.4.2 which will benefit from advancements in the AXMEDIS framework development and availability of prototypes to allow more specific real examples of editorial formats to be presented. The Annex I states the following:

WP8.4 -- Definition of distribution channel editorial formats, coordinated by XIM

The standard distribution channel is today a single distribution path for each type of content, and often, multiple proprietary systems of representation for the same content. The definition of distribution channel editorial formats would provide one way, unified and rock-solid content format for multipurpose applications. Alternative solutions such as CONTESSA supported the multi-channel distribution by using an XML model of content into the Content management systems of the content provider that also include multiple transcoding engines for transforming the XML model of content in the format suitable for the channel. This approach is not flexible enough since the transcoding of content at the source create strong limitation to the management of Digital Rights. In fact, in models such as CONTESSA the DRM can be applied only to the content in its final version. This creates relevant problems for the content providers since the content distributors are entitled to receive non protected content. This is almost unacceptable in most cases. In AXMEDIS, differently from other solutions such as CONTESSA, the channel distributors may maintain their distribution process. They can continue to use the same format for reaching the final users. In AXMEDIS, the content is distributed on the P2P tool, AXEPTool, by using an evolution of the MPEG-21 format, with the AXMEDIS contribution. This format will make it easy to contain and deliver MPEG-4, MPEGxx, PDF, HTML, SVG, images, documents, videos, audio file, etc. (in open standard format for continuation, without the use of proprietary technologies – see WP 4.7.1) on demand and for all platforms according to the final format produced by the Distributor. The received content will be formatted using AXMEDIS tools on the basis of specific editorial formats. Their structure will be defined in WP4. In this WP, a set of editorial formats will be defined and produced to cover a large number of needs, ranging from: i-TV, PDA, PC, mobiles, etc.

1.1 Responsibilities

XIM has main responsibility for the document with close coordination and contribution from ILABS as overall WP8 coordinator. Contributions are required from all partners mentioned either at chapter or section level. When a section does not specifically identify a responsible it is intended to be a general one where all partners involved in the process shall contribute.

2 Introduction (XIM, ILABS, all)

A key philosophy of AXMEDIS is to support multiple media kinds by means of MPEG21 objects rather than enforce a common media descriptor (for instance using XML, like many POPE cross-media architectures), which limits the use of DRM and restricts the functionality of content to the lowest common denominator.

In order to achieve this support for multiple media kinds, it is necessary to define editorial formats for each distribution channel/device so that content can be automatically reformatted into the optimum format and layout at distribution time, when channel-specific DRM and scheduling rules can also be applied.

These editorial formats need to sufficiently define the layout and composition of how content will appear on the target terminal device in order to enable the automatic formatting engine in AXMEDIS to function.

2.1 Media types

The types of content that will need to be formatted for distribution are:

Simple, single items, for example:

- Text (txt, html, pdf, doc, postscript)
- Images (tiff, bmp, jpg, gif, etc.)
- Audio (mpeg3, wave, aac...)
- Video (avi, mpeg, etc.)

Basic Combined items, for example:

- Text + Image
- Text + Audio
- Text + Image + Audio
- Audio + Text
- Animation + Text
- Video + Text
- Video compilations
- Audio compilations
- Image compilations
- etc.

Advanced Combined items, for example:

- Multimedia presentation embedding sets of raw assets, in particular MPEG4 and SMIL
- Multimedia presentations composed of basic combined objects, such as HTML, etc.
- MPEG21 Hierarchical sets, ie. nested media objects within a tree structure.

Full descriptions of the possible file formats and issues for these media types can be found in DE3.1.3 Content Aspect Specification.

2.2 Presentation modalities

For each of these content permutations, it is necessary to provide editorial formats to combine them:

- **Static layout** – for static output, using page layout templates
- **Time-based layout** – for temporal output, using a movie-style output, for example with transition effects, but again using a layout template where appropriate, for example where still images or pages are to be combined into a slide show.

Design guidelines for both modalities of output for each distribution channel have been produced in DE8.2.1. Some of the key constraints for specific formats on specific channels are also provided in this document to highlight the key differences in template design for each channel.

2.3 Distribution channels

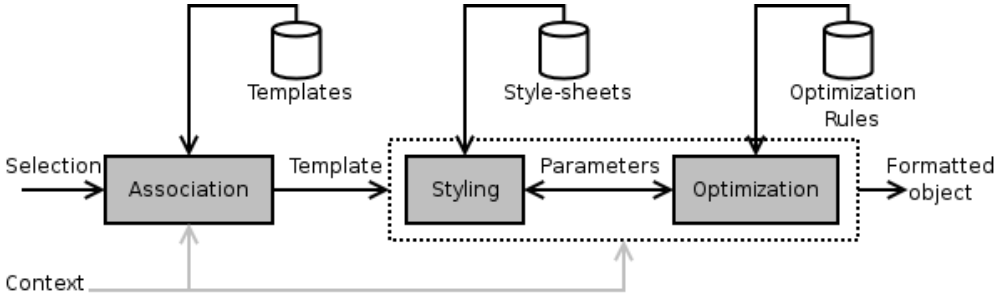
The final dimension to this work is the need to format these combinations of content (either time-based or statically) for each of the specific distribution modalities:

- Web
- PDA
- Kiosk
- mobile
- iTV for Satellite.

This will require transcoding into an appropriate delivery format (HTML, etc. if static, SMIL or MPEG4 if time-based, dependent on the optimum format for the channel), creating in turn a formatted AXMEDIS object. These formatting and transcoding processes are described in DE4.3.1, but an overview of the formatting process is provided below to explain the context for this document.

2.4 Overview of the formatting process

The AXMEDIS formatting system is composed of two main logical blocks:



The main blocks of the formatting system

1. The **Association Block** receives as input the media obtained from the selection process and some context information from the author (or from an user profile). Context information provides basic knowledge about the type of the presentation, its output format and the platform for which the presentation is targeted. The output of this block is the indication of a template that matches the input and describes the basic structure of the presentation.

   To detect the "type" of the presentation and choose a template that matches it, it is very relevant for this block to have a precise context definition. At least, the author should specify:

   - the platform for which the presentation is targeted, i.e.: PC, PDA, SmartMobile, Mobile, iTV, etc.;
   - the output format, i.e.: MPEG2, MPEG4, AXMEDIS/MPEG21, SMIL, HTML, etc.;
   - a general category for the content, i.e.: slide show, electronic book, kiosk, interactive music, interactive video, training tutorial, etc.

   The system will perform a mapping of context and selected media, and it will choose the template nearest to these criteria.

2. The **Styling/Optimization Block** select a style-sheet for the given template and adjusts its parameters to cope with the context, managing the adaptation/transcoding of the media involved. The output of this block is the formatted content.

The optimization process is a very critical and complex phase that may involve many aspects. Layout determination and media encoding are part of the optimization: to fit the context, the layout specified in the template may be radically modified and the media may necessitate transcoding or resampling; moreover, media have to be transformed (scaled, rotated, etc.) to fit the adapted layout.

The choice of the optimization algorithm is very important: the problem of finding the best combination of a potentially large number of layout parameters is NP-complete, therefore the computational time to determine an exact solution is not reasonable. A more practical approach is to search approximate solutions, with methods such as Tabu Search or Genetic Algorithms. These algorithms are still very time intensive and benefit from parallel execution. For this reason the chosen algorithm has to be easily distributed on the AXMEDIS GRID.

The system is intended to work in two modalities:

- an interactive modality, which allows the author to choose or create templates and style-sheets and control the results of the adaptation. This modality is based on the C++ AXMEDIS Formatting Engine;
- an automatic modality, that manages the whole process following a set of rules and the specified context; this modality uses the JavaScript modules that wrap the Formatting Engine.

Please refer to DE4.3.1 for a full description of the AXMEDIS Content Formatting Engine.

3 Definition of the formatting description (DSI, ILABS, XIM)

This section presents an easy to understand description of the editorial formatting description required for each channel, as specified and needed by the algorithms for the AXMEDIS Composition and Formatting Engine in WP4.3. Rather than repeat too much information developed in the technical deliverables, the aim of this section is to provide "designer-friendly" guidelines that will assist with actual creation and editing of editorial formats to be fed into the Formatting Rule Engine.

It is important to remember that both the raw content and the formatted output will be MPEG21 digital items, possibly MPEG21 containers for composite objects. The formatting process will adapt the source content according to predefined rules – including the editorial format – into the target layout, format, size and compression settings for the specific distribution channel. This will also need to accommodate integrating the more complex multimedia objects into playable, usable objects for reach channel.

3.1 A 'template of templates'

The easiest way to describe the AXMEDIS formatting description is to think of a template of templates: a layout definition language that can be used to create templates suited to each channel.

In order to format AXMEDIS objects using these templates, it is essential that source CMS metadata related to AXMEDIS objects needs to be mapped into a set of common template elements for design layout. These elements can then be placed and scaled on the output device by the Formatting Rule Engine.

3.2 Template elements

Each layout for distribution on a terminal device will require a known set of template page layout elements:

- Title
- Subtitle
- Body – consisting one or more sub-elements for text, images etc.
- Sidebar
- Navigation – in fact two elements, one for global navigation and one for local navigation within the context of the content being viewed
- Footer

3.2.1 title

This will be extracted from the AXMEDIS object's metadata.

3.2.2 subtitle

Optional. Like TITLE, this will need to be extracted from the object's metadata.

3.2.3 body

This element will contain a set of sub-elements depending upon the number and type(s) of content to be distributed.

Sub-elements could be:

- textBox (possibly containing in-line images)
- imageBox – image in separate bordered or spaced box
- mediaBox – movie, audio or flash animation
- mediaStatus – for time-based media, displays status information ('ready', 'connecting', etc.)
- controlBox: this element will need to be present in order to control time-based media (audio, video, animation, interactive, etc.) when a mediabox is present.

3.2.4 localNavigation

Much of the sample content described in DE8.1.1 consists of sets of images/text pages. As AXMEDIS objects will store the granular objects, we will need to automatically generate a navigation element for each page and also navigation pages ('home' etc). This could be a simple set of links to pages belonging to a common container object, or could be enhanced via metadata to allow for hierarchical navigation (eg 'history', 'present day', sections, etc. each with dynamic section homepages).

3.2.5 sidebar

An optional secondary panel of content usually positioned on the right hand side of a screen, normally holding section navigation, promotional information other content related to the main body content.

3.2.6 footer

A standard footer element could be generated from the object's general and PAR metadata (author, terms of use, copyright information etc.).

3.3 Styling

As well as requiring layout templates, the formatting system will need to have style sheets defined for output. These could be at two levels:

- A preferred stylesheet from the content owner, which provides a blueprint for how the content should ideally be presented

- A channel-owner specific stylesheet provided by the distribution channel, which will take into account legibility and accessibility, as well as possibly specific portal branding or other commercial issues that may override the content owner's preferred styling.

The following styling capabilities will therefore be needed:

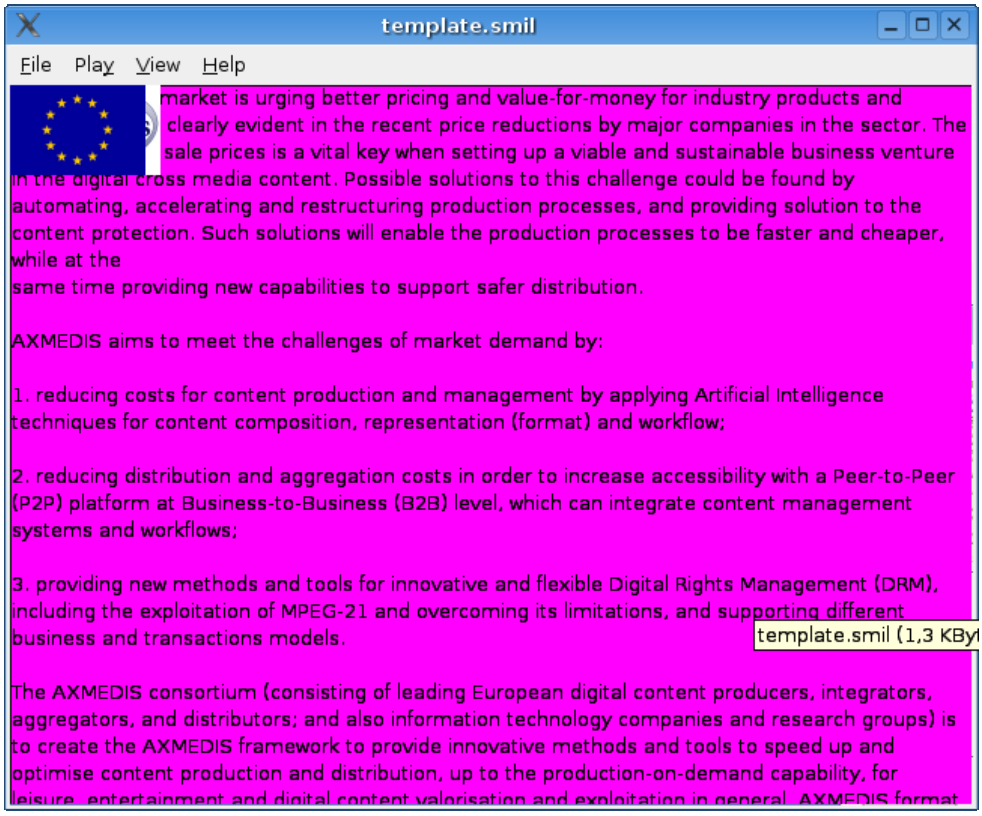
- Definition of preferred fonts, font sizes, border sizes and colours, background colours and background images, both globally for a screen as well as independently for each layout element
- Ability to override content-owner's preferred formatting, for example, to meet the constraints of a particular device or channel

- Maintaining (inheriting) elements of the original look and feel of source documents where permitted by the formatting rules

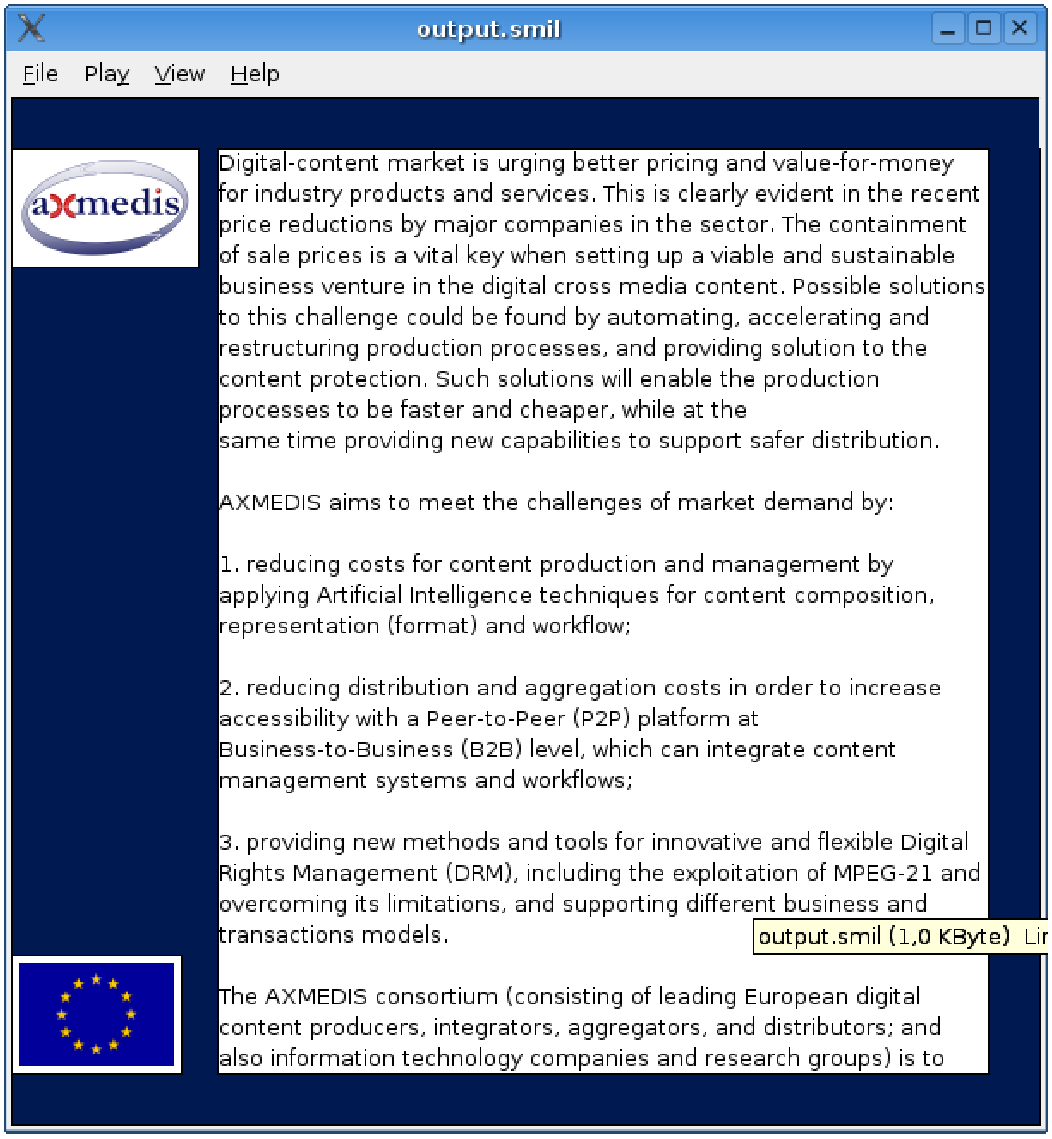
These requirements are met by using XSL as a standards-based styling description language. XSL is well documented on the web by w3c (www.w3.org/Style/XSL/).

3.4 Example SMIL and XSL format

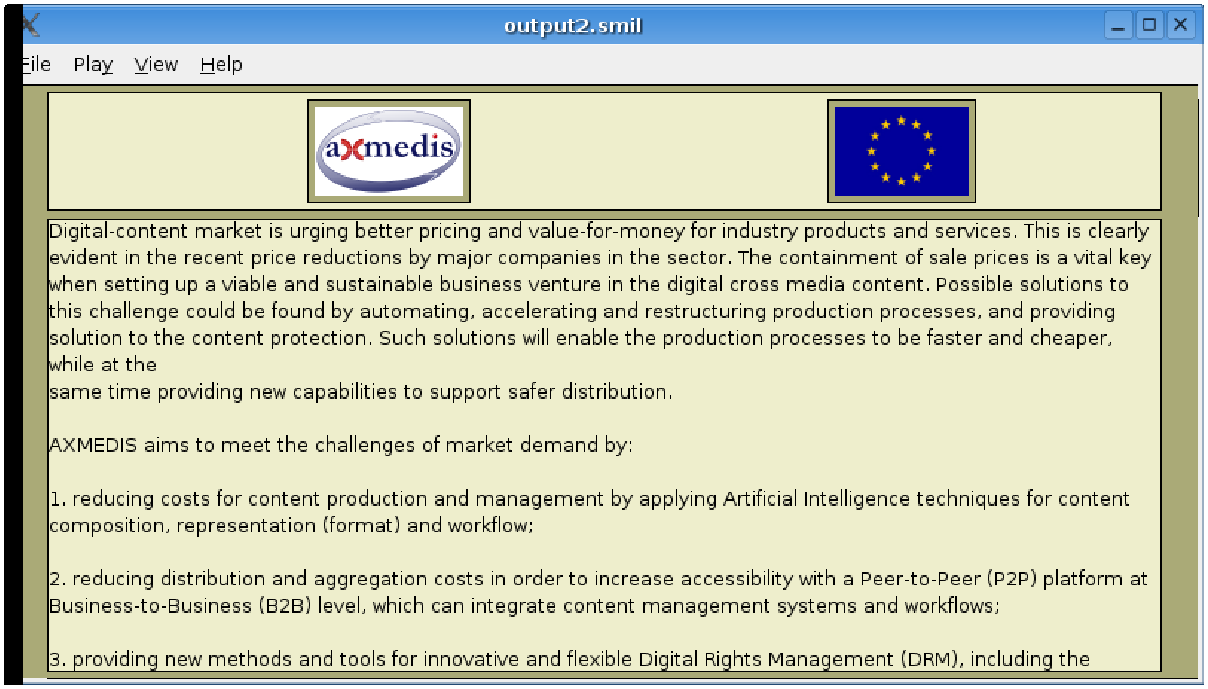
The following is an example using SMIL plus XSL to define a page layout containing images plus text.



Unformatted SMIL template.



The formatted SMIL document (template + XSL style-sheet)



The same document reformatted by an additional stylesheet (template + stylesheet1 + stylesheet2)

SMIL code for the template

```
<smil>
 <head>
  <!-- metadata -->
  <meta name="title" content="Axmedis Formatting Test" />
  <meta name="author" content="Paolo Vaccari" />
  <meta name="copyright" content="nocopyright:)" />
  <!-- regions -->
  <layout type="text/smil-basic-layout">
   <root-layout
            width="auto" height="auto"
            title="Smil test"
            />
   <region id="text_region"
              width="auto" height="auto"
              top="auto" left="auto"
              />
   <region id="axm_img_region"
              width="auto" height="auto"
              top="auto" left="auto"
              />
   <region id="eu_img_region"
              width="auto" height="auto"
              top="auto" left="auto"
              />
  </layout>
 </head>
 <body>
  <par>
   <switch>
          <text id="text1_it"
              systemLanguage="it"
              src="media/axm_it.txt"
              type="text/plain"
              region="text_region"
              />
          <text id="text1_en"
              systemLanguage="en"
              src="media/axm_en.txt"
              type="text/plain"
              region="text_region"
              />
   </switch>
   <img id="axm_img"
            src="media/logo_axm.png"
            type="image/png"
            dur="indefinite"
            region="axm_img_region"
            />
   <img id="eu_img"
            src="media/flag.png"
            type="image/png"
            dur="indefinite"
            region="eu_img_region"
            />
  </par>
 </body>
</smil>
```

XSL used for the transformation:

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

 <!-- build the basic SMIL structure -->
 <xsl:template match="smil">
  <smil>
   <head>
    <xsl:apply-templates select="head"/>
   </head>
   <body>
            <xsl:apply-templates select="body"/>
   </body>
  </smil>
 </xsl:template>

 <!-- apply templates for head -->
 <xsl:template match="head">
  <xsl:apply-templates/>
 </xsl:template>

 <!-- apply templates for body -->
 <xsl:template match="body">
  <xsl:apply-templates/>
 </xsl:template>

 <!-- apply templates for layout -->
 <xsl:template match="layout">
  <layout type="text/smil-basic-layout">
   <xsl:apply-templates/>
  </layout>
 </xsl:template>

 <!-- apply templates for switch, using my preferences -->
 <xsl:template match="switch">
  <xsl:apply-templates select="*[@systemLanguage='en']|
                               *[@systemScreenDepth='16']|
                               *[@systemCaptions='on']|
                               *[@systemOperatingSystem='linux']|
                               *[@systemScreenSize='800x600']
                               "/>
 </xsl:template>

 <!-- apply templates for par -->
 <xsl:template match="par">
  <par>
   <xsl:if test="@id">
    <xsl:attribute name="id">
     <xsl:value-of select="@id"/>
    </xsl:attribute>
   </xsl:if>
   <xsl:apply-templates/>
  </par>
 </xsl:template>

 <!-- apply templates for seq -->
 <xsl:template match="seq">
  <seq>
   <xsl:if test="@id">
    <xsl:attribute name="id">
     <xsl:value-of select="@id"/>
    </xsl:attribute>
   </xsl:if>
   <xsl:apply-templates/>
  </seq>
 </xsl:template>

 <!-- apply templates for excl -->
 <xsl:template match="excl">
  <seq>
   <xsl:if test="@id">
    <xsl:attribute name="id">
     <xsl:value-of select="@id"/>
    </xsl:attribute>
   </xsl:if>
   <xsl:apply-templates/>
  </seq>
 </xsl:template>

 <!-- media -->

 <!-- apply templates for img -->
 <xsl:template match="img">
  <img
```

XSL stylesheet used in the first transformation:

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <!-- parameters; with sablotron:
       sabcmd xsl input output '$param1=p1' '$param2=p2'
       their scope includes smilstruct.xsl
  -->
  <xsl:param name="screenWidth"/>
  <xsl:param name="screenHeight"/>
  <xsl:param name="backGround"/>
  <xsl:param name="foreGround"/>
  <xsl:param name="menuWidth"/>
  <xsl:param name="textWidth"/>


  <!-- SMIL structure -->
  <xsl:include href="smilstruct.xsl" />

  <!-- attributes for root-layout -->
  <xsl:template match="root-layout">
    <root-layout
              width="{$screenWidth}" height="{$screenHeight}"
              backgroundColor="{$foreGround}"
              title="Smil+XSL test"
              />
  </xsl:template>

  <!-- attributes for regions -->
  <xsl:template match="region">
    <xsl:choose>
      <xsl:when test="@id='axm_img_region'">
              <region
                id="{@id}"
                backgroundColor="{$backGround}"
                fit="hidden"
                regPoint="center"
                regAlign="center"
                width="110" height="70"
                top="{$screenHeight * 0.05}"
                left="0"
                z-index="1"
                />
      </xsl:when>
      <xsl:when test="@id='eu_img_region'">
              <region
                id="{@id}"
                backgroundColor="{$backGround}"
                fit="hidden"
                regPoint="center"
                regAlign="center"
                width="100" height="70"
                bottom="{$screenHeight * 0.05}"
                left="0"
                z-index="1"
                />
      </xsl:when>
      <xsl:when test="@id='text_region'">
              <region
                id="{@id}"
                backgroundColor="{$backGround}"
                fit="scroll"
                width="{$textWidth}"
                left="{$menuWidth}"
                height="{$screenHeight * 0.9}"
                top="{$screenHeight * 0.05}"
                z-index="2"
                />
      </xsl:when>
      <xsl:otherwise>
              <!-- include the region as is -->
              <xsl:copy-of select="."/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

</xsl:stylesheet>
```

Second XLS stylesheet

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

 <!-- SMIL structure -->
 <xsl:include href="smilstruct.xsl" />

 <!-- attributes for root-layout -->
 <xsl:template match="root-layout">
  <root-layout
            width="800" height="400"
            backgroundColor="#AAAA77"
            title="Smil+XSL test"
            />
 </xsl:template>

 <!-- attributes for regions -->
 <xsl:template match="region">
  <xsl:choose>
   <xsl:when test="@id='axm_img_region'">
            <region
              id="{@id}"
              backgroundColor="#AAAA77"
              fit="hidden"
              regPoint="center"
              regAlign="center"
              width="110" height="70"
              top="10"
              left="200"
              z-index="2"
              />
     <!-- additional region -->
     <region
      id="emptyReg"
      backgroundColor="#EEEECC"
      width="750" height="80"
      top="5"
      left="25"
      z-index="1"
      />
   </xsl:when>
   <xsl:when test="@id='eu_img_region'">
            <region
              id="{@id}"
              backgroundColor="#AAAA77"
              fit="hidden"
              regPoint="center"
              regAlign="center"
              width="100" height="70"
              top="10"
              left="550"
              z-index="2"
              />
   </xsl:when>
   <xsl:when test="@id='text_region'">
            <region
              id="{@id}"
              backgroundColor="#EEEECC"
              fit="scroll"
              width="750"
              left="25"
              height="305"
              top="90"
              z-index="2"
              />
   </xsl:when>
   <xsl:otherwise>
            <!-- include the region as is -->
            <xsl:copy-of select="."/>
   </xsl:otherwise>
  </xsl:choose>
 </xsl:template>

</xsl:stylesheet>
```

3.5 Formatting description for each distribution channel

The general format to describe layout will be HTML for static layouts and SMIL in combination with XSL for time-based layouts, given that all channels support HTML and MPEG4 to a greater or lesser extent. However, each channel has specific constraints and optimum delivery formats which need to be considered.

3.5.1 Internet (TISCALI, XIM, DSI)

The web is perhaps the most flexible channel, but due to the wide range of potential client configurations, requires careful planning in order to support the widest range of browsers, connection speeds and plugins.

Guidelines for content delivered across the Internet are generally well defined, so where relevant, references are given to standards.

The required delivery formats are summarized in the following table:

AXMEDIS object content type	**Optimum delivery format for this content on this channel**	**Reference to guidelines for this format**
Simple		
Image (Static)	JPG or GIF	http://www.w3.org/Graphics/
Text (Static)	HTML or PDF	www.w3c.org; PDF would require compatible viewer/plugin
Audio (Time-based)	MPEG4 with audio track or simple MP3	www.m4if.org
Video (Time-based)	MPEG4 with video and audio tracks	www.m4if.org
Image sequence (Time-based slideshow)	HTML or MPEG4 with image sequence track	www.m4if.org
Combination		
Text + Image (Static)	HTML + JPG/GIF	www.w3c.org
Text + Audio (Time-based)	MPEG4 with HTML and audio tracks or FLASH	www.m4if.org / www.macromedia.com
Text + Image + Audio (Time based)	MPEG4/SMIL with HTML, audio and image sequence tracks or FLASH	www.m4if.org / www.macromedia.com
Animation + Text (Time-based)	MPEG4/SMIL with HTML and image sequence tracks or FLASH	www.m4if.org / www.macromedia.com
Video + Audio + Text (Time-based)	MPEG4/SMIL or FLASH multiple tracks	www.m4if.org / www.macromedia.com
Video compilations (Time-based)	MPEG4 or FLASH multiple tracks	www.m4if.org / www.macromedia.com
Audio compilations (Time-based)	MPEG4 or FLASH multiple tracks	www.m4if.org / www.macromedia.com
Advanced combined assets		
Multimedia presentation/ game/ eLearning application etc. (Time-based)	SMIL / MPEG4 or FLASH; multiple tracks including interactivity	www.w3.org/TR/REC-smil / www.macromedia.com

3.5.2 Kiosk (ILABS, XIM, DSI)

The available formats are a subset of those for Internet delivery.

AXMEDIS object content type	**Optimum delivery format for this content on this channel Or state 'not supported'**	**Constraint**
Simple		
Image (Static)	JPG, PNG or GIF	JPG to save bandwidth when image nature does not cause too much degradation during encoding, PNG for better quality and still acceptable size, GIF for animation inside HTML / VRML avoiding to use heavier tools like FLASH
Text (Static)	HTML or FLASH	HTML is preferred as it is easier to handle and often ensures self adaptation at client size with lover limitation than the one presently encountered when porting content across devices/platforms.
Audio (Time-based)	MP3 or WAW	MP3 is preferred due to higher compression at comparable quality therefore ensuring less bandwidth usage
Video (Time-based)	WindowsMedia or QuickTime	WM solution is preferred as embedded in the OS and therefore does not require a specific plug-in download and installation
Image sequence (Time-based slideshow)	FLASH or HTML	Flash solution unless specifically designed for adaptation are difficult to adapt (resize may case in format alteration and loss of info)
Combination		
Text + Image (Static)	HTML + JPG/GIF	Frame / tabular based pages may present problems in terms of accessibility unless this has been properly taken into account especially as far as impaired people access is concerned, for example for blind people is necessary to add extra info to the image caption (the description of the image) and ensure that the adopted solution is compatible with features supported by internet pages screen readers (AAA W3C compliancy)
Text + Audio (Time-based)	HTML and MP3 audio tracks or FLASH	

Text + Image + Audio (Time based)	HTML + JPG/GIF, MP3 audio or FLASH	Frame / tabular based pages may present problems in terms of accessibility unless this has been properly taken into account especially as far as impaired people access is concerned, for example for blind people is necessary to add extra info to the image caption (the description of the image) and ensure that the adopted solution is compatible with features supported by internet pages screen readers (AAA W3C compliancy). As far as audio is concerned is necessary to ensure both auto-play at load time and user-controlled-play, audio caching is required to avoid disturbing jerky effects in rendering
Animation + Text (Time-based)	HTML + animated GIF or FLASH	Flash solution, unless specifically designed for adaptation, are difficult to adapt (resize may case in format alteration and loss of info). Animated GIF is heavy if quality has to be high
Video + Audio + Text (Time-based)	FLASH or SVG + WindowsMedia or QuickTime	OGG or Matrioska (Video + text)
Video compilations (Time-based)	MPEG4 multiple tracks	
Audio compilations (Time-based)	MPEG4 multiple tracks	
Advanced combined assets		
Multimedia presentation/ game/ eLearning application etc. (Time-based)	FLASH or HTML	Flash solution unless specifically designed for adaptation are difficult to adapt (resize may case in format alteration and loss of info)

3.5.3 PDA (ILABS, COMVERSE, XIM, DSI)

The available formats are a subset of those for Internet and Kiosk delivery therefore here only additional constraints are reported.

AXMEDIS object content type	**Optimum delivery format for this content on this channel Or state 'not supported'**	**Constraint (refer to constraints table)**
Simple		
Image (Static)	JPG or GIF	The main limitation here is size, other formats may be too heavy or lacking a rendering tool. Previously reported constraints still apply.

Text (Static)	HTML	Some feature implementations of HTML pages can be dependant on the specific browser and this can cause limitations even for Windows © as the PDA version of Internet Explorer is different form the PC one. Previously reported constraints still apply.
Audio (Time-based)	MPEG4 with audio track or simple MP3	The main limitation here is size. Previously reported constraints still apply.
Video (Time-based)	MPEG4 with video and audio tracks	The main limitation here is size. Previously reported constraints still apply.
Image sequence (Time-based slideshow)	MPEG4 with image sequence track	The main limitation here are: 1) size plus 2) tools used for packaging may introduce limitation to adaptability Previously reported constraints still apply.
Combination		
Text + Image (Static)	HTML + JPG/GIF	The main limitation here is size. Previously reported constraints still apply.
Text + Audio (Time-based)	MPEG4 with HTML and audio tracks	The main limitation here is size especially as far as PDA or smartphones are concerned. Previously reported constraints still apply.
Text + Image + Audio (Time based)	MPEG4/SMIL with HTML, audio and image sequence tracks	The main limitation here is size. SMIL players are not yet so widespread and usable as the others (Media Player, Quicktime, RealPlayer…) Previously reported constraints still apply.
Animation + Text (Time-based)	MPEG4/SMIL with HTML and image sequence tracks	The main limitation here is size. SMIL players are not yet so widespread and usable as the others (Media Player, Quicktime, RealPlayer…) Previously reported constraints still apply.
Video + Audio + Text (Time-based)	MPEG4/SMIL multiple tracks	The main limitation here is size. SMIL players are not yet so widespread and usable as the others (Media Player, Quicktime, RealPlayer…) Previously reported constraints still apply.
Video compilations (Time-based)	MPEG4 multiple tracks	The main limitation here is size. Previously reported constraints still apply.

Audio compilations (Time-based)	MPEG4 multiple tracks	The main limitation here is size. Previously reported constraints still apply.
Advanced combined assets		
Multimedia presentation/ game/ eLearning application etc. (Time-based)	SMIL / MPEG4; multiple tracks including interactivity	The main limitation here is size. SMIL players are not yet so wide-spread and usable as the others (Media Player, Quicktime, RealPlayer…) Previously reported constraints still apply.

Content type	**Constraint**	**Alternative format**	**Comments**
Video	Max bitrate Max dimension Max resolution	Mediaplayer, Realplayer, Quicktime	Adaptation "on-device", usually, is not feasible due to required storage space and computational load (at least for PDA and smart-phones) The same applies to for-mat conversions (quick-time to …)
Flash	Adaptability	XHTML	A ".swf" file is a binary embedding used re-sources therefore unless adaptation has been fore-seen at design stage it is not possible.

3.5.4 Mobile (IRC, COMVERSE, XIM, DSI)

The available formats are a subset of those for Internet and Kiosk delivery.

AXMEDIS object content type	**Optimum delivery format for this content on this channel Or state 'not supported'**	**Constraint (refer to constraints table)**
Simple		
Image (Static)	JPG or GIF	Main limitation is screen size (see constraints table below).
Text (Static)	HTML	Main limitation is screen size, as above
Audio (Time-based)	MPEG4 with audio track or sim-ple MP3	Audio quality can now be very good on newer mobile terminals, especially those designed for ste-reo headphones for mp3 playing.
Video (Time-based)	MPEG4 with video and audio tracks	Main limitation is screen size, as above
Image sequence (Time-based slideshow)	MPEG4 with image sequence track	Main limitation is screen size, as above
Combination		
Text + Image (Static)	HTML + JPG/GIF	Main limitation is screen size, as above

Text + Audio (Time-based)	MPEG4 with HTML and audio tracks	Main limitation is screen size, as above
Text + Image + Audio (Time based)	MPEG4/SMIL with HTML, audio and image sequence tracks	Main limitation is screen size, as above
Animation + Text (Time-based)	MPEG4/SMIL with HTML and image sequence tracks	Main limitation is screen size, as above
Video + Audio + Text (Time-based)	MPEG4/SMIL multiple tracks	Main limitation is screen size, as above
Video compilations (Time-based)	MPEG4 multiple tracks	Main limitation is screen size, as above
Audio compilations (Time-based)	MPEG4 multiple tracks	
Advanced combined assets		
Multimedia presentation/ game/ eLearning application etc. (Time-based)	SMIL / MPEG4; multiple tracks including interactivity	Main limitation is screen size, as above.

Content type	Constraint	Alternative format	Comments
Video, still images, text, combined	Screen size. Smallest phone displays are 90 pixels width/ height. Growing adoption of 3G however means that the larger 240x230 pixel displays are increasingly used.	Not applicable..	Need formatting and adaptation strategies to chunk, scroll, shrink or reject as appropriate content that is too large to view in native format.

3.5.5 iTV/ Satellite (EUTELSAT, XIM, DSI)

For EUTELSAT's network the available formats are identical of those for broadband Internet delivery as the delivery device is a PC with a high network bandwidth. EUTELSAT's OSClient supports all regular PC media formats, the only unique action being that it can uncompress zip-compressed files automatically before they are opened.

4 Guidelines on the development of editorial formats for each channel (XIM, ILABS, TISCALI , ANSC, SEJER …)

This section presents guidance to content developers and other users of the development of formats.

4.1 General guidelines(all)

The AXMEDIS editors and viewers include built-in capabilities for a set of leading formats (as defined in DE3.1.2 part B):

- video (AVI, MPEG, etc.)
- images (single and multi-image TIFF, BMP, JPG, GIF, etc.)
- audio (MP3, WAV, WMA, AAC...)
- document (HTML, PDF, MS Word Doc, Postscript)
- SMIL (XML-based multimedia language that enables scripting of time-based media)
- MPEG4 (can include interactivity, controls, etc via SMIL, as well as audio, video, image, etc. digital items)
- Other formats, dependent on external editors associated via MIME types.

As well as the source content itself, these formats should also be used for the editorial formats, ie the layout templates for the delivery of content. This will allow easy management, previewing and editing within the AXMEDIS tools. Therefore the editorial formats should be designed to be delivered in one or more of the above formats, the actual selection of format being optimised for the specific channel.

MPEG21 Hierarchy
It is possible for an AXMEDIS object to contain a tree structure of child media objects, so it is necessary to consider this both in terms of an input/source format but also as a possible delivery format.

It should also be possible to imply navigation between objects belonging to the same the same container object by nature of their hierarchy.

SMIL and MPEG4 adoption
The AXMEDIS viewers and editors will include full support for visual and behavioral editing of SMIL format content interactivity, so it is recommended to use SMIL-compliant layouts (preferably with MPEG4 compliant components) as a useful time-based format that can have both static and dynamic content objects placed within them.

Metadata considerations
The automated formatting process may need to adjust associated content metadata accordingly between the source object and the formatted. For example, there is a difference between a computer with a 1024x768 resolution and a computer between a 1600x1200 resolution and content that has been re-scaled for one should contain metadata to indicate for which aspect ratio it is optimized to. Where this is likely, this should be registered within the editorial formats.

4.1.1 Editorial formats for static layouts (Still Image and Text)

Orientation
A basic starting point for still images lays in the two reference format adopted for images, nominally: "**portrait**" or "**landscape**". Given the variety of dimensions and resolutions of screens and rendering devices the optimum format template for image rendering is a black square with a suitable dimension ration in respect to the overall page structure. Images presented therefore will always present a double black stripe either in the upper / lover part on in the left/right one.

Fixed and scaleable layouts
There are two different kinds of formatting for static content on the web:

- the layout has a fixed width and the content is positioned inside once and for all,
- the layout adjusts to the size of the window and so the content can be repositioned when resizing

The former approach is the easiest to display the content and a lot of web sites are using it: www.times.com, www.lemonde.fr, www.msn.com, www.yahoo.fr. The page cannot be too wide so the text is easier to read and you will never risk having paragraphs made of single lines. Such paragraphs give the illusion that there is not a lot of content.

The latter approach is the hardest as you have a lot of parameters to take into account in order to have a nice display of the information. The homepage site msdn.microsoft.com is divided in two columns, the first column being twice as large as the second. When the window is resized, each column grows or shrinks proportionally.

In general, it is preferable to avoid very long documents on a single browser page as the end-user might feel overwhelmed. The "fixed-width" layout, with the fact that the paragraphs have a restricted width is more tolerant against that problem: you have readable paragraphs and you are not discouraged by a screen full of characters.

In the “variable width” layout, it is often good practice to include illustration between some paragraphs. It has the double advantage of allowing the reader to breathe and to synthesize or illustrate what is written. Illustrations are also helpful as a reference point when scrolling.

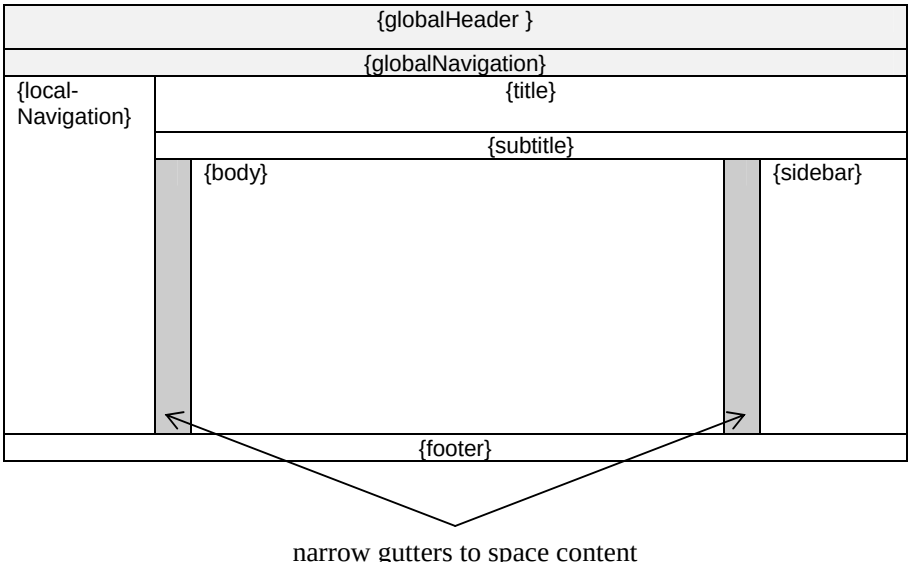
The recommended general guidelines therefore for AXMEDIS static layouts are:

- Define where possible fixed layouts for each output device – this is in fact easier for 3G mobile, PDA, etc. where a small screen is already a fixed constraint, and for PC/Kiosk a standard page width can be agreed.

- Limit page lengths to reasonable size, and build in the capability to ‘chunk’ long pages into screen-size sections with an internal index (‘home previous-next’ links at the bottom of each section) as necessary.

A proposed general static layout is illustrated below.

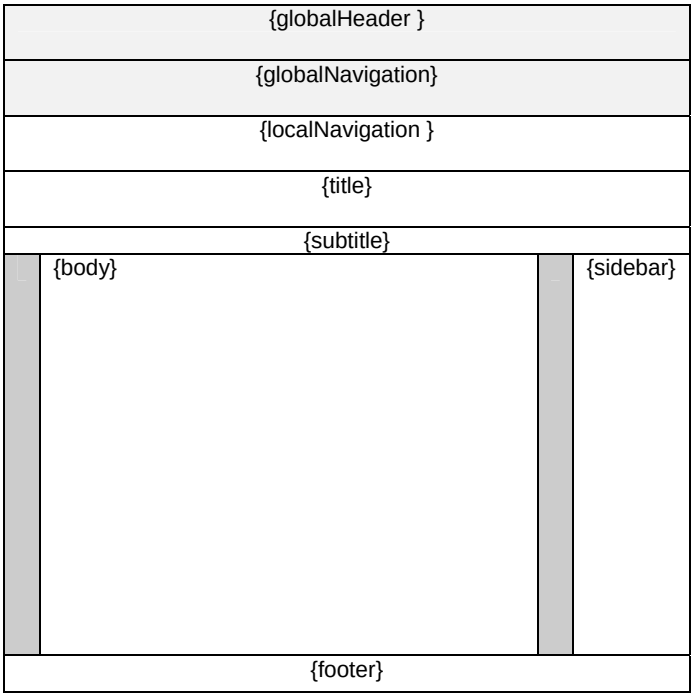
General '3 column' editorial format for landscape-oriented devices viewing static content



Key:

Region/ element	Description
{globalHeader}	masthead region containing main title and/or branding, usually from the distribution channel
{globalNavigation}	main navigation for the distribution channel
{localNavigation}	navigation local to the viewed content (could be derived from object hierarchy)
{title}	title specific to the object being viewed
{subtitle}	optional subtitle or long title specific to the object being viewed.
{body}	main container for actual page content, which could be simple item or complex combination of aggregated items for static display
{sidebar}	optional secondary content container either specific to the object (or object hierarchy) or generated by the distribution channel (eg containing purchase information, or other promotions)
{footer}	legal, authorship, contact and terms of use information, could be generated from object metadata.

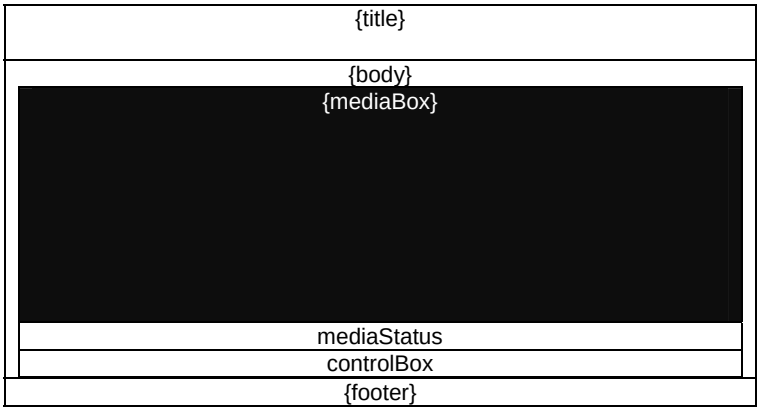
General editorial format for portrait-oriented devices viewing static content



This layout combines the same layout elements as for landscape layout, but optimizing the height of the display by stacking local navigation vertically.

4.1.2 Editorial formats for time-based layouts (Audio, Video, Animation, Interactive and Multimedia)

General 'Movie Player' editorial format for landscape-oriented devices viewing time-based content



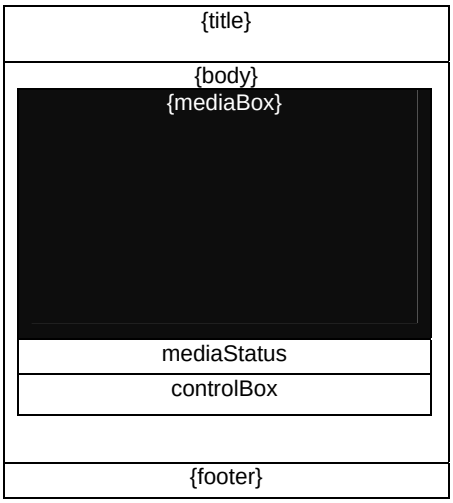
Key:

Region/ element	Description
{title}	title specific to the media object being viewed

{body}	main container for actual media content, which could be simple time-based item or complex combination of aggregated items for static display
{mediaBox}	Contains video or animation. In the case of audio-only content this box will not appear.
{mediaStatus}	Displays status information (‘ready’, ‘connecting’, HH:MM:SS etc.)
{controlBox}	Contains buttons for play/pause, stop, rewind to control the media content
{footer}	legal, authorship, contact and terms of use information, could be generated from object metadata.

This general layout can be resized for portrait orientation as below.

General ‘Movie Player’ editorial format for portrait-oriented devices viewing time-based content



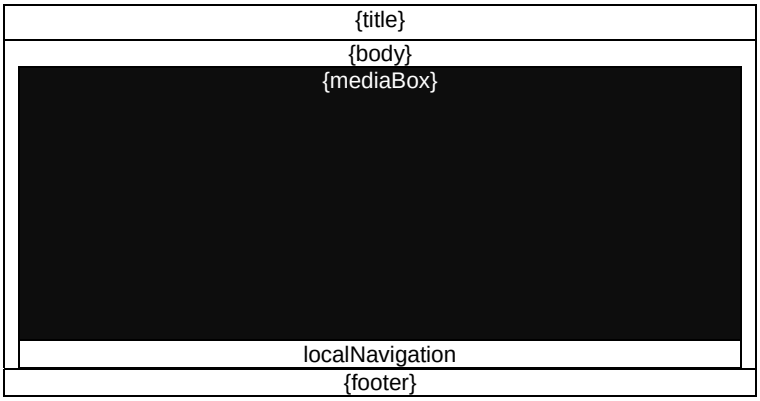
Slide show format
A slide show player can be considered close to a movie player but with navigation instead of playback controls. As the slideshow concept is the presentation of a sequence of still images, the navigation is required to allow the user to:

- advance to the next image
- return to the previous image
- return to the first image (start)
- close or quit.

Also, as the content is a series of still images, streaming is not required and therefore the media status box is not necessary.

This general format is illustrated in the diagram below.

General 'Slide Show Player' editorial format for landscape-oriented devices viewing still image sequence content



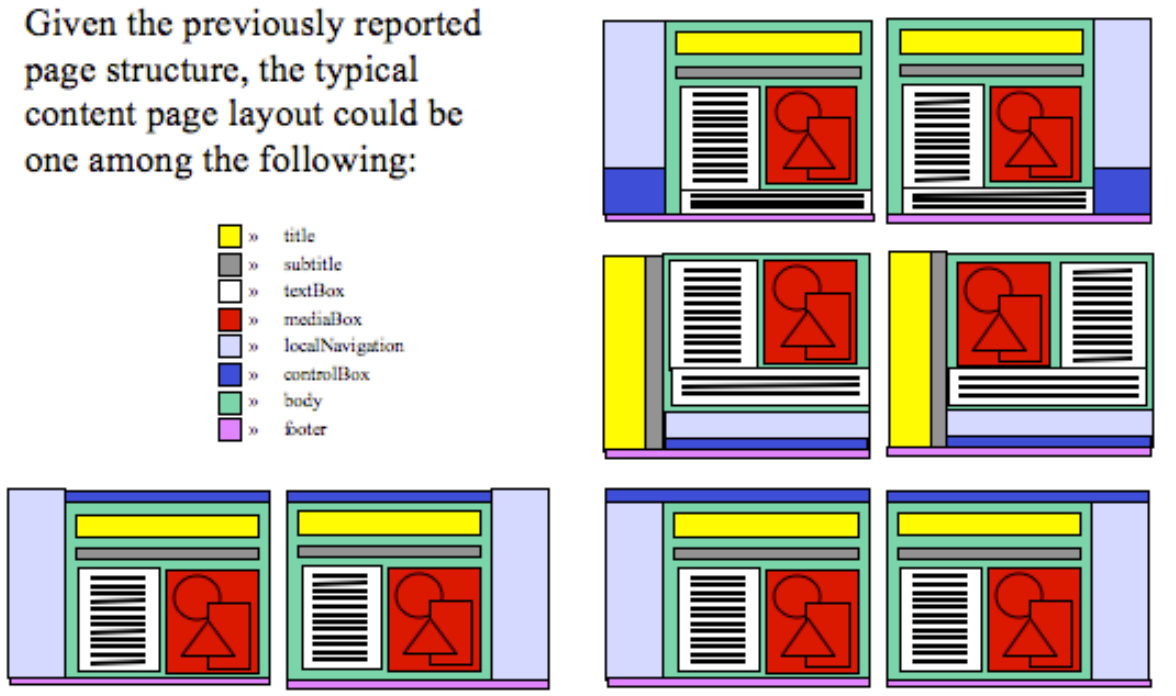
Key:

Region/ element	Description
{title}	title specific to the media object being viewed
{body}	main container for actual media content, which could be simple time-based item or complex combination of aggregated items for static display
{mediaBox}	Contains video or animation. In the case of audio-only content this box will not appear.
{localNavigation}	navigation local to the viewed content (could be derived from object hierarchy)
{footer}	legal, authorship, contact and terms of use information, could be generated from object metadata.

4.1.3 Hybrid editorial formats combining media, text and images

By combining static and dynamic page elements, it is possible to create a flexible range of possible layouts to suit different platforms, content, and channels.

The illustration below presents 8 example variations.



Possible hybrid editorial formats combining static and dynamic content elements

4.2 Guidelines specific to Web, Kiosk and iTV (XIM, TISCALI, OD2, SEJER, ANSC…)

The web, Kiosk and interactive TV (for EUTELSAT's network) formats can be considered together as these three formats rely on PC client hardware and software. The general guidelines above apply directly to all three channels. Size and layout constraints are those that generally apply to PCs plus the specific constraints and characteristics of each channel as described in section 3.

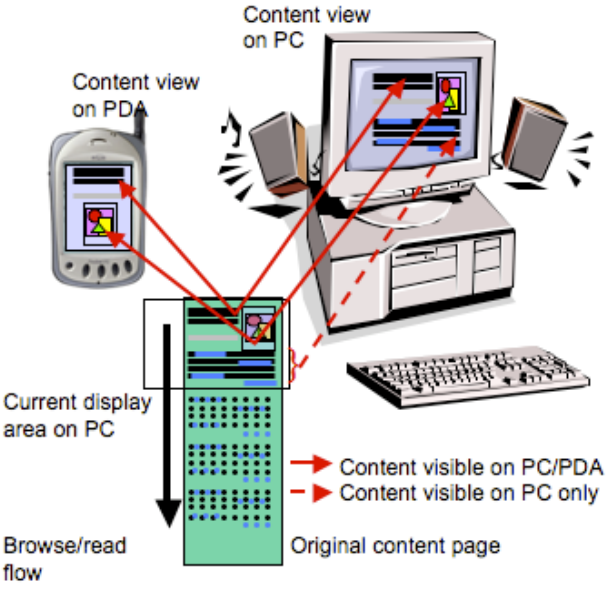
For both static and dynamic formats, PC users can enjoy a relatively large display area and familiar user interface. However, it is important to take into account familiar interface design conventions for the PC, as users will expect the application to behave along the same lines as other media players/browsers, and will feel more comfortable with editorial formats that resemble web site conventions such as the 3 column layout for static content.

4.3 Editorial formats for PDA and mobile (ILABS, COMVERSE)

Passing from PC optimsed format to either a PDA or mobile 'smart phone', the change in display size implies:

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

The diagram below illustrates the problem.



Viewing the same content on PC and PDA/mobile

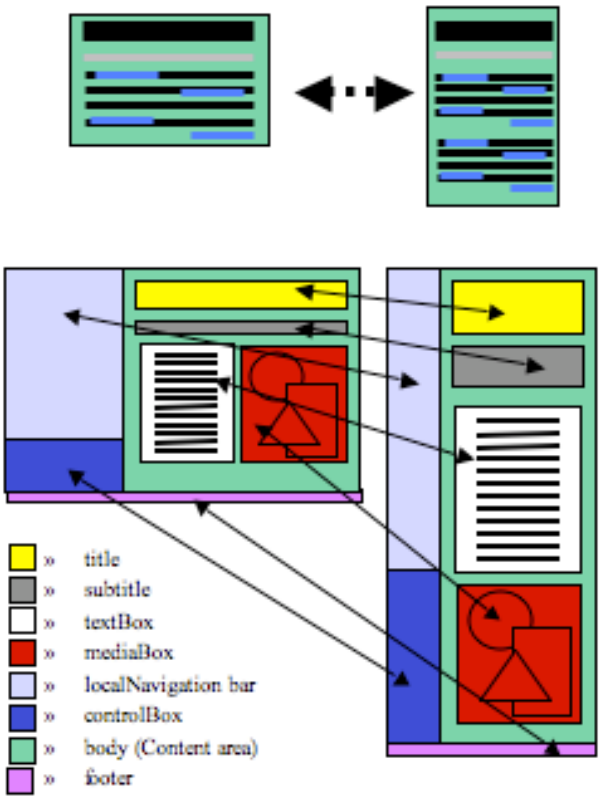
There are two basic formatting strategies to enable the usable transformation of content to the PDA/mobile platform.

4.3.1 Orientation conversion

The simplest modification of the format is converting from landscape to portrait or vice versa
This implies:

- a change in areas sizes (to retain the same content) and possibly
- a change in location of some specific content

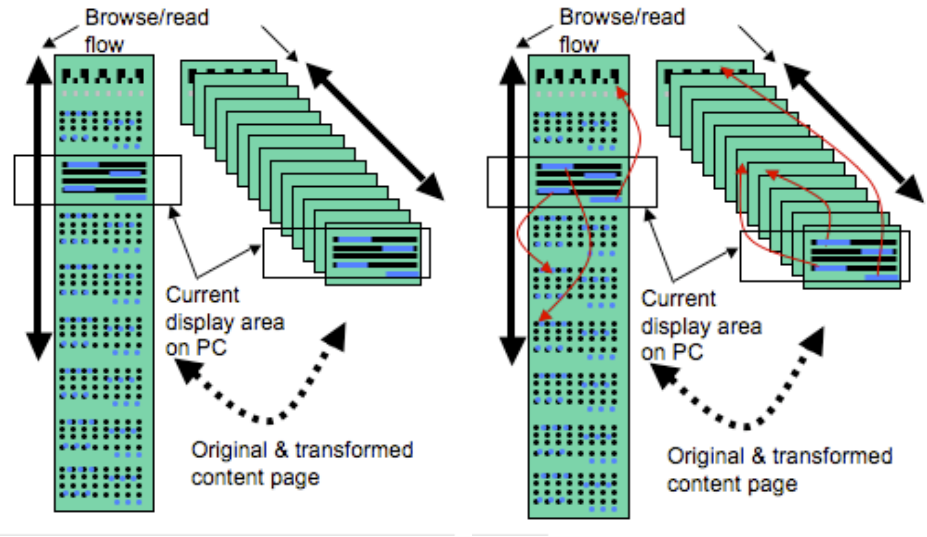
The diagram below illustrates this transformation by relocating and resizing the standard page elements introduced in the previous section.



Transforming content between portrait and landscape

4.3.2 Chunking

Passing from a single page to a set of pages, the way content is browsed changes from scrolling to Paging. The same applies vice-versa. The breaking up of a long sroll-based page into small elements is illustrated in the first diagram below.



Unlinked and linked chunked screens from a single, long page

Without enhanced navigation, this requires the user to sequentially step up and down the document. An improved version would transform internal anchor points into links between pages (URLs), including the return to “top of page”, as shown in the second diagram above.

5 Example formats (XIM, ILABS, TISCALI, COMVERSE, EUTELSAT …)

This section presents sample editorial formats for each channel. To avoid duplication, this section presents a high-level only presentation of sample formats. The actual content described is available for test case use in the AXMEDIS content repository and is described in deliverable DE8.1.1.

5.1 Computer based (XIM, ILABS, TISCALI, OD2, SEJER, ANSC…)

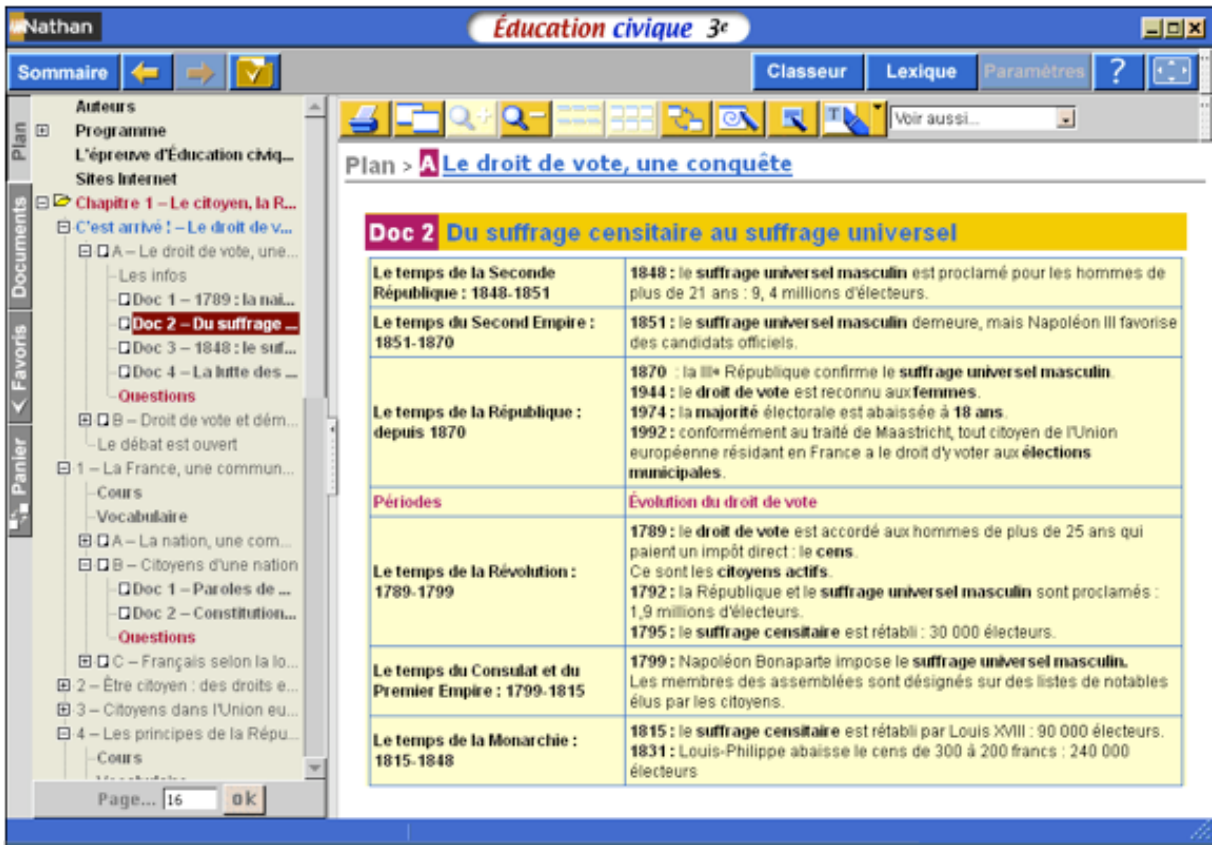
These examples apply equally to web, kiosk and EUTELSAT's iTV format as all three distribution channels involve PC-based client terminals.

Static formats

For delivery to PC, static content is typically viewed via a web browser. The samples below show real-world variations of the suggested grid-style page layout.

The above sample from XIM shows the typical '3 column' grid layout, with main/global title element in the top corner, local title along the top centre, left-hand side navigation element, footer element, central body element for content, plus an optional right-hand column for search, section-level content, etc. For this example, the elements, layout and formatting are defined by CSS, although these could be easily mapped onto XSLT. For accessibility and crawling reasons, no images are used for navigation or title text, all formatting is defined in CSS/HTML.

This sample from ILABS is from a training course. Here the navigation is provided along the bottom of the screen, the main content therefore is given a wider space suited to larger images, etc.



This sample from SEJER follows the side navigation convention, with main title element along the top and main content space filling the rest of the screen. The more complex hierarchical navigation requires a wider column space.

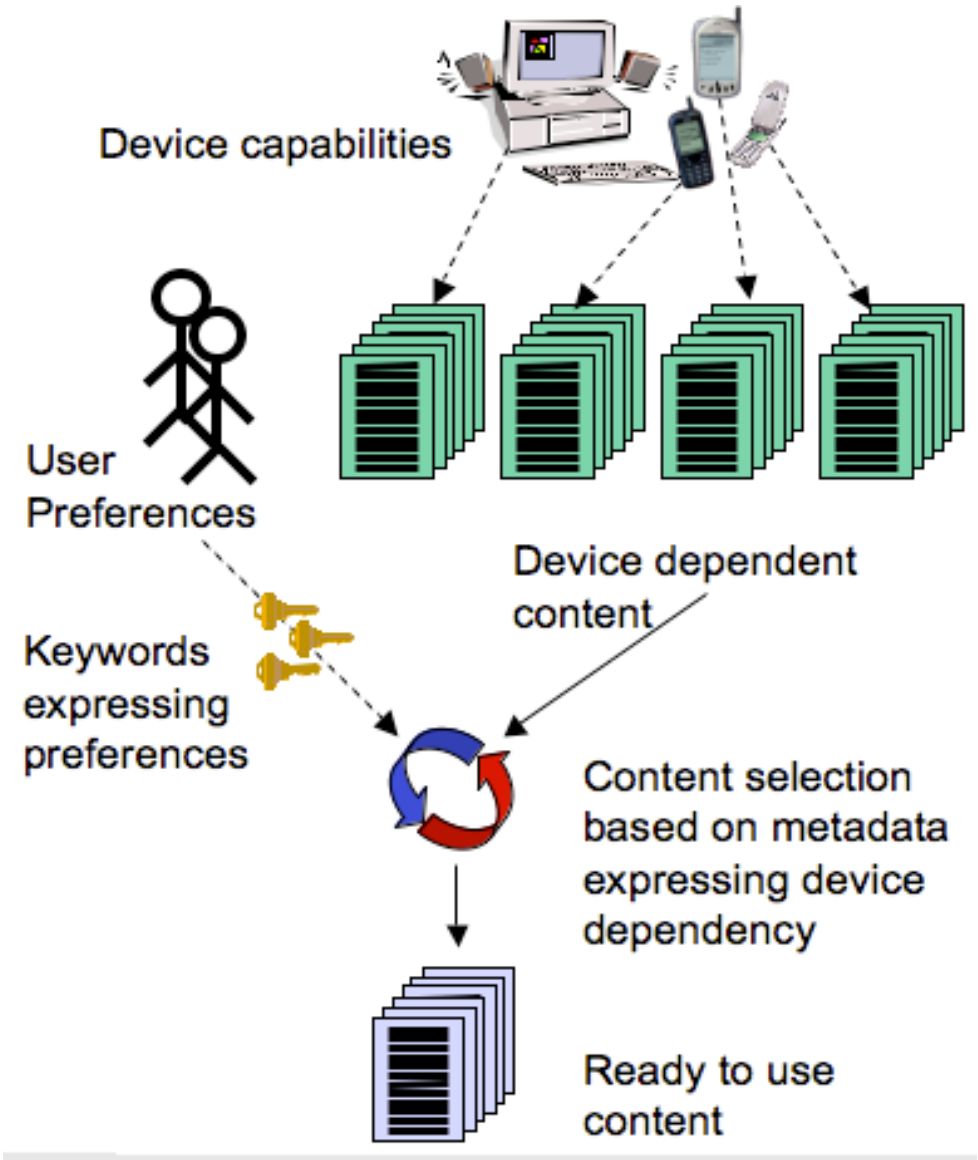
Temporal formats

Temporal content is normally viewed either in a media player mode or embedded within a content layout such as a web page. From an editorial point of view, the embedded mode should only be used where the rest of the screen layout is strongly related to the media content.

At present, current examples of temporal content use Content Selection to support different channels, connection speeds, terminal devices and platforms. Content Selection works in the following manner:

- Device dependent content is prepared and stored in a structure reflecting device dependencies for proper access (broadband, dial-up, wap …)
- Device dependent content is selected matching user preferences and connection profile

This is illustrated in the following diagram:



The screen shots below show the *What Women Want* movie trailer viewed in QuickTime player. These media files were encoded to QuickTime via capture from BetaSP by XIM using Discrete Cleaner. Three versions were created to allow either automatic selection (where network connection speed has been detected by the media player) or user choice over download time versus quality and size.

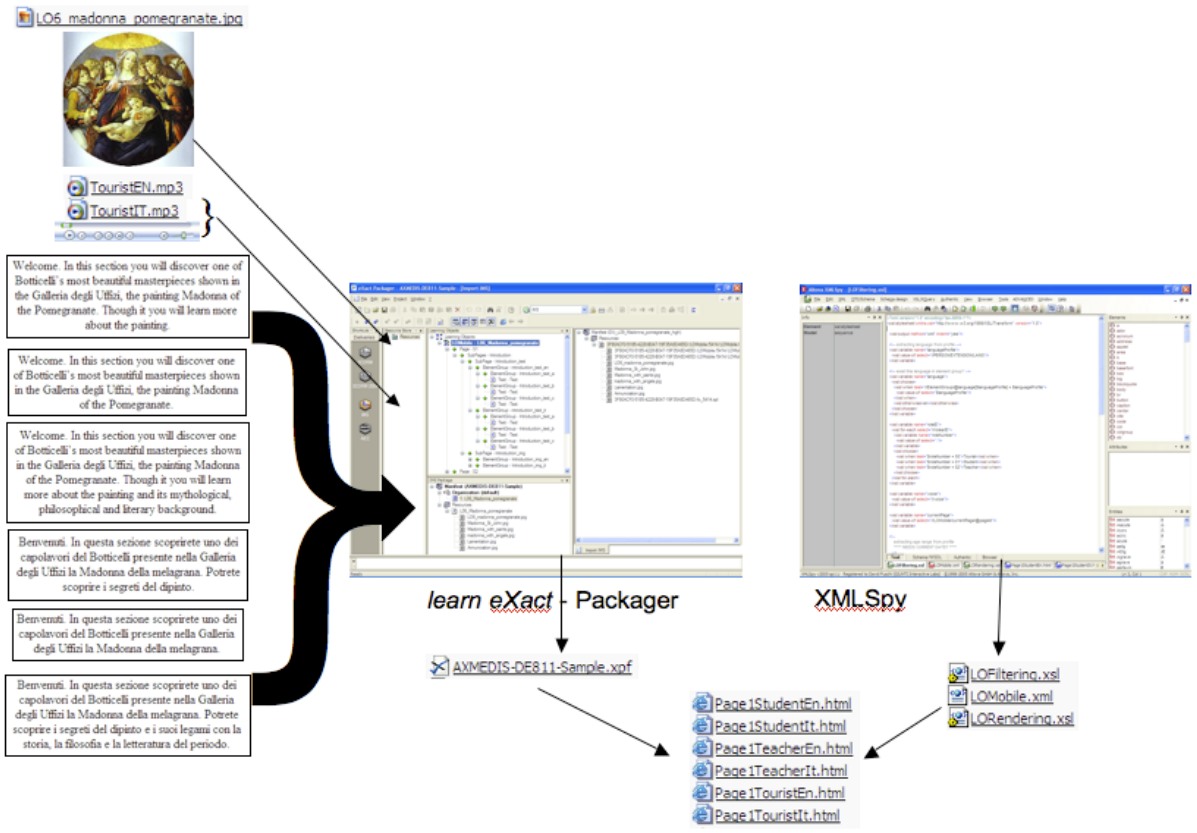
Format:160x89 pixels, 2.11Mb. Sorensen codec, 150kbits/sec rate, 13FPS.

Format: 240x134 pixels, 5.67Mb. Sorensen codec, 400kbits/sec rate, 13FPS.

Format: 480x268 pixels, 14.86Mb. Sorensen codec, 1050kbits/sec rate, 13FPS.

5.2 Smart phone (ILABS, COMVERSE)

This example from ILABS applies both to PDAs and mobile phones.



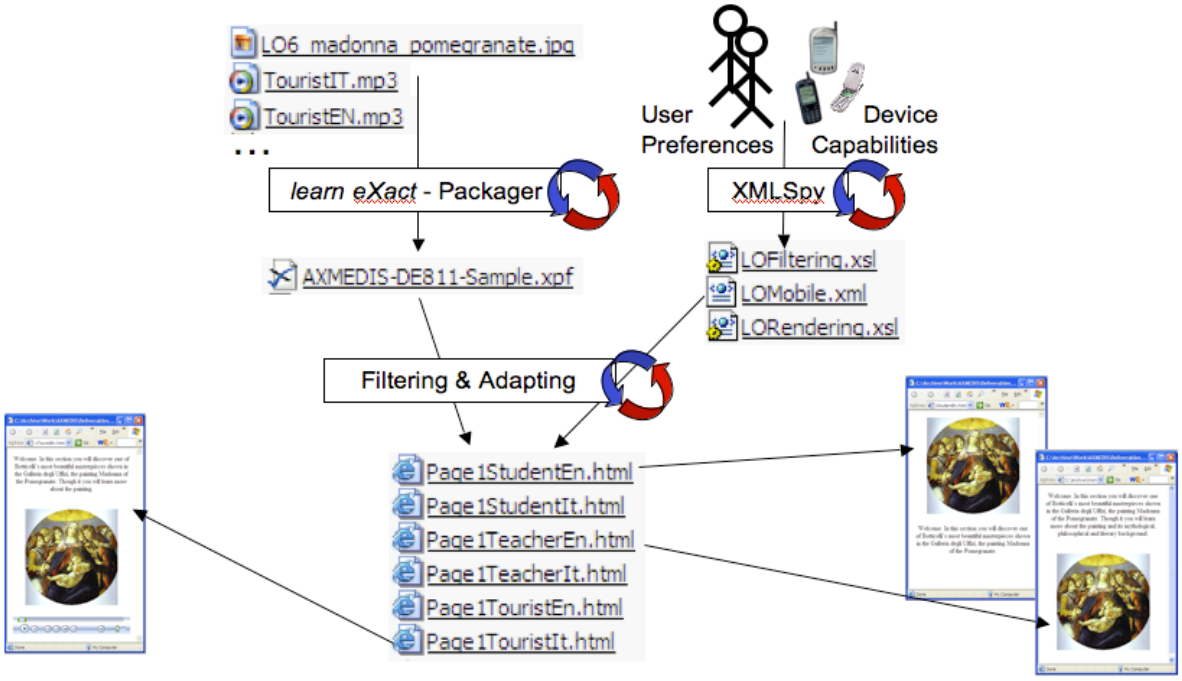
The above diagram is summarizes the process of construction and formatting of a set of contents (image, text and audio) into a single object (in this case a Learning Object or LO) that can be delivered both on a PC or a mobile platform (PDA or smart phone). The schematic representation can be summed up in the following steps:

1) Exploiting the IMS packaging standard (see ref. 5), it is possible to assemble into the same object not only different media, but also different language versions of media.

2) The assembled content can then be filtered and rendered on the basis of what described in XSL files that are either produced by tools like XMLSpy or by AXMEDIS tools.

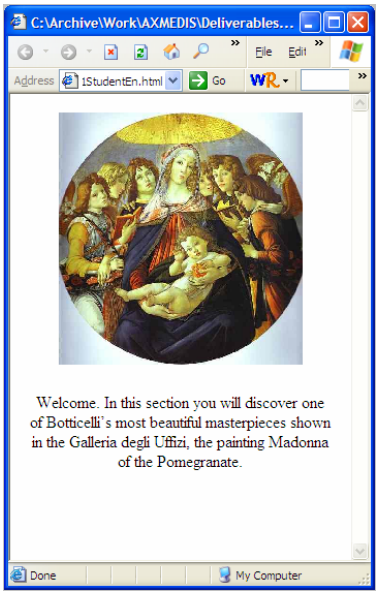
3) The combination of the LO (represented by a XML file) and formatting XSL files is then used to generate (at server side) pages differentiated on the basis of the controlling parameters received at run time (kind of user, terminal device, location or fruition methods, etc.)

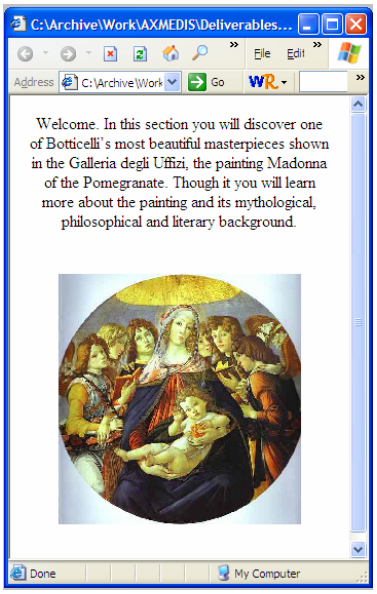
This provides a good example of a starting point that could be used to develop a set of easily adaptable content. Moreover this could be further reinforced by the adoption of special additional tags to be used in a "content description" file that will complement the original object. In the following diagram this approach is presented in a more "flow oriented" manner pointing out the tools performing the various processes that could either be for processing or filtering and adapting. In more detail the picture shows how starting from a set of raw assets, thanks to the LEX authoring environment it is possible to achieve a product (LO) that has a structure describing components, relations, formats (basic ones), purposes and further metadata info. In parallel with such a process there should be another process that leads to content filtering and formatting XSL descriptions that will be used along with a content XML description to handle the real content in the subsequent steps of the process.

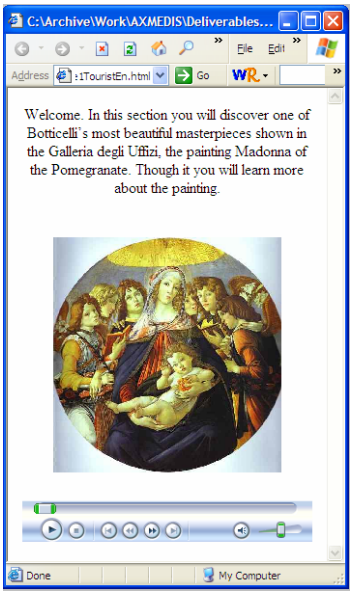


The last step will be the processing of all such inputs to generate "ready to use" content that can be delivered and used at the end-user device. The single pages that will be delivered will in essence bear the same origin but present only the subset of content that is deemed to be shown after taking into account the set of constraints coming either from the user or device.

The following images are the same as the above samples but presented in a more visible manner.







6 On demand considerations (XIM, DSI, COMVERSE…)

This section considers the specific implications on formats of the need for dynamic formatting on demand applications. It is particularly relevant to the mobile platform for personalized content presentation.

The implication of on-demand formatting is the requirement to support end-user preferences. These have been taken into account in the design of the formatting engine, and also in the planning of the templates. Typical user preferences that impact formatting and layout would be:

- **Media quality vs download speed** (allowing the possibility to view content faster but at a lower quality). This could either require dynamic compression of media, or at least selection from multiple media versions.

- **Accessibility** preferences such as preferred font size and preferred font, even a personal CSS. This could be best implemented by supporting a customised CSS for the end-user, which by default would be empty.

- **Custom layout**, although more challenging, this could allow preferred default layout for content (eg where to position the navigation, options to disable elements such as title and subtitle for experienced users to maximise media viewing size on a small device, etc.)

7 References & Standards (all)

[1] Glover, F. and Laguna, F. - “Tabu Search” - Kluwer Academic Publishers – 1997

[2] Forrest, S. - “Genetic algorithms” - ACM Comput. Surv. 28, 1 – 1996

[3] Bulterman et al. - "Synchronized Multimedia Integration Language (SMIL 2.0)" – W3C Recommendation - 2005 - http://www.w3.org/TR/2005/REC-SMIL2-20050107/

[4] Clark, J. - “XSL Transformations (XSLT) Version 1.0” - 1999 - W3C Recommendation - http://www.w3.org/TR/xslt

[5] IMS Global Learning Consortium Inc. - http://www.imsglobal.org/

8 Terminology (all)

Term	Explanation (including source if available)
Editorial format	In AXMEDIS, an editorial format is a combination of layout template and stylesheet that enable a set of source content to be formatted for a given distribution channel and output device. It therefore includes both format (fonts, colours, etc.) as well as layout (sizes, spacing, positioning of elements, etc.)
LO	Learning Object (see IMS in references)
POPE	Produce Once, Publish Everywhere
SMIL	Synchronized Media Integration Language
XSL	Extensible Stylesheet Language