



Automating Production of Cross Media Content for Multi-channel Distribution

www.AXMEDIS.org

DE9.5.4

Integrated Prototype of content Production and Distribution on-demand for Mobile Phones, and new generation of PDAs

Version: 2.4

Date: 15/04/2007

Responsible: ILABS (revised and approved by coordinator)

Project Number: IST-2-511299

Project Title: AXMEDIS

Deliverable Type: Private

Visible to User Groups: No

Visible to Affiliated: No

Visible to the Public: No.

Deliverable Number: DE9.5.4

Contractual Date of Delivery: M30

Actual Date of Delivery: 15/04/2007

Work-Package contributing to the Deliverable: WP9,5

Task contributing to the Deliverable: WP9,5 all

Nature of the Deliverable: report

Author(s): ILABS, EPFL, IRC, DSI, FHGIGD, UPC

Abstract: the present document reports the detailed specifications for the development of the experimental test-bed for distribution towards mobiles.

Keyword List: mobile, PDA, content, production, distribution

Table of content

1	EXECUTIVE SUMMARY AND REPORT SCOPE	4
2	CONTENT PRODUCTION.....	5
2.1	THE MOBILE FACTORY CASE.....	6
2.2	MOBILE FACTORY OVERALL ARCHITECTURE.....	8
2.3	LEARN EXACT INSTALLATION AND CONFIGURATION.....	10
2.3.1	LEARN EXACT MANAGER	11
2.3.2	TO EDIT DOMAIN PROPERTIES	12
2.3.3	TO INSTALL A NEW DOMAIN POSSIBLY KEEPING DEFAULT SETTINGS	12
2.3.4	TO PERFORM MASSIVE IMPORT/EXPORT OF USER DATA.....	12
2.3.5	TO IMPORT USER DATA FROM A CSV FILE	13
2.3.6	TO EXPORT USER DATA TO A CSV FILE.....	14
2.3.7	TO EXPORT USER DATA TO A UNICODE CSV FILE.....	14
2.3.8	CLIENT REQUIREMENTS FOR WEB ACCESS	14
2.3.9	eXact Packager Client Installation	14
2.3.10	TYPES OF EXACT PACKAGER LICENSE.....	15
2.3.11	TO REQUEST AND INSTALL A CLIENT LICENSE.....	15
2.3.12	TO UNINSTALL EXACT PACKAGER	15
2.4	INTERFACING LEX WITH A BACKEND	15
2.4.1	WEB SERVICE INTERFACE DESCRIPTION	16
2.4.2	Error Codes	21
2.5	BACKEND PUBLISHING PROCEDURE	22
2.5.1	Querying and retrieving procedure.....	23
2.5.2	Sample of Query and Result Set in XML Format.....	24
2.5.3	Lobster WSDL File.....	24
2.6	CONTENT SELECTION & ACQUISITION FOR MOBILE E-APPLICATIONS	32
3	INTRODUCTION TO CONTENT ADAPTATION.....	34
3.1	CONTENT ADAPTATION FOR MOBILE E-APPLICATIONS	35
3.1.1	Template definition and Format adoption	38
3.1.2	Content combination and adaptation	38
3.1.3	Metadata and tagging.....	43
4	INFO ON SUPPORTIVE TECHNOLOGIES INTEGRATED	44
4.1	MEDIA DISTRIBUTION MANAGEMENT REQUIREMENTS	44
4.1.1	Target Client Device and User ID Management.....	44
4.1.2	Profiling management (profiles collection, storage, access and/or update)	44
4.1.3	Delivery Context Information	46
4.1.4	Dynamic Rights Audit Management Information.....	48
4.1.5	Personalisation (profiling resolution and inference).....	48
4.1.6	Media Selection & Bundling	52
4.1.7	Media Transcoding	52
4.1.8	Media Adaptation.....	52
4.1.9	Conclusions on Integration for mobile distribution- Requirements Update (IRC)	54
4.2	INTRODUCTION TO MOBILE TRANSCODING	55
4.2.1	Libraries for Video Transcoding	56
4.2.2	Libraries for Image Transcoding	56
4.2.3	Libraries for Audio Transcoding	57
4.3	INTRODUCTION TO DEVICE PROFILE MANAGEMENT	57
4.3.1	CC/PP	58
4.3.2	UAProf	58
4.3.3	DELI.....	58
4.3.4	WURFL.....	59
4.3.5	The GPAC suite	59

5	DISTRIBUTION TOWARDS MOBILE.....	61
5.1	MOBILE CONTENT SELECTION AND CATEGORISATION (SERVER SIDE)	61
5.2	MOBILE CONTENT DISTRIBUTION (CLIENT SIDE)	65
6	MOBILE DISTRIBUTOR DEMO FACT SHEET	68
7	INSTALLATION AND CONFIGURATION	71
8	TECHNICAL SPECIFICATION AND DETAILS	72
8.1	OVERALL USE CASES	73
8.2	OVERALL ARCHITECTURE	74
8.3	DATA TYPES	74
8.4	TEMPORAL DIAGRAMS & RELATED GUI ASPECTS	77
8.4.1	Administrative Use Cases	78
8.4.2	User System Access Use Cases	87
8.4.3	Content Fruition Use Cases	91
8.6	MOBILE ARCHITECTURE	99
8.6.1	Top Level Modules	99
8.6.2	AXMEDIS Communication Modules	106
8.6.3	Domain Management Module	111
8.6.4	User Management Modules	112
8.6.5	Content Management Module	116
8.6.6	Content Retriever Module	118
8.6.7	Catalogue Management Module	120
8.6.8	DB Management Modules	121
8.6.9	E-Commerce Module	125
8.6.10	LobsterCommunicModule Module	126
8.7	DATABASE ENTITY RELATIONSHIP	127
9	USED CONTENT SAMPLES DESCRIPTION	130
10	REFERENCES	130
11	GLOSSARY OF ACRONYMS & TERMS	131
11.1	GLOSSARY OF TERMS IN THE PUBLISHING ENVIRONMENT	131
11.2	GLOSSARY OF TERMS IN THE E-LEARNING ENVIRONMENT	132
11.3	GLOSSARY OF TERMS IN THE MOVIE, TV, MEDIA... ENVIRONMENT	132
11.4	OTHER GLOSSARIES	132

1 Executive Summary and Report Scope

This document contains a detailed specification of the demonstrators for production and distribution of content to mobile devices. Therefore we will deal with content production, protection, sharing, formatting and distribution considering the market segments and interest of the potential consumers and their needs. Final objective of the demonstrator is to validate project results and achievements in a real world application that may lead to the set up of some complete and ready to use AXMEDIS product and service. The document is structured in three sections, but the first one is only an introductory one; the second section aims at describing content production (basically the Mobile Server Side) together with the third, about content adaptation. Fourth section is an overview of the supportive integrated technologies, while the fifth one covers the actual distribution to mobiles, which technical details are given into section six.. References to used contents and link to glossaries are provided too.

2 Content Production

The starting point of the mobile demonstrator is the mobile factory. This one represents an instance of the publishing value chain enriched with AXMEDIS solutions. In the following sections we will quickly recall the production structure so to make apparent the area of intervention and the modified process. The aim is to specify most of the work by difference. This approach has been adopted as it has proved to be the most suitable for inserting new technology in highly structured and complex environment like the content production one. The content production chain follows well-codified and standardised process that can be described as follows:

Activity	Task		Involved roles	
Idea (1)		Management Marketing Department Press Office		
Market survey (2)				
Initiative design (3)				
Go / No Go decision ¹ (4)		Management		
Research of (5)	Sources (6) References (7) Contacts (8) ... Similar initiatives (10)	IPR/© clearance	Management Marketing Department Press Office Legal Department	
Draft acceptance ² <i>(if positive the next step starts if not the previous is reiterated)</i>			Management Marketing Department Legal Department	
Selection of (11)	Content (12) Communication message (13)		Marketing Department Designer Press Office Legal Department	
Formal authorisation to start production <i>(if positive the next step starts if not the previous is reiterated. In this case IPR/© clearance should have been completed if not process may be suspended / stopped / cancelled)</i>			Management Marketing Department Legal Department	
Adaptation & Finalisation of (14)	Content (15) Communication message (16)	Costs control	IPR/© Contracts management	Production department Outsourced service Press Office Marketing manager Legal department Company accountant
Marketing (17)	Promoting	Costs & revenues control		Marketing manager Legal department
Distribution (18)	Revenue management			Production department or Outsourced service Marketing manager Legal department Company accountant

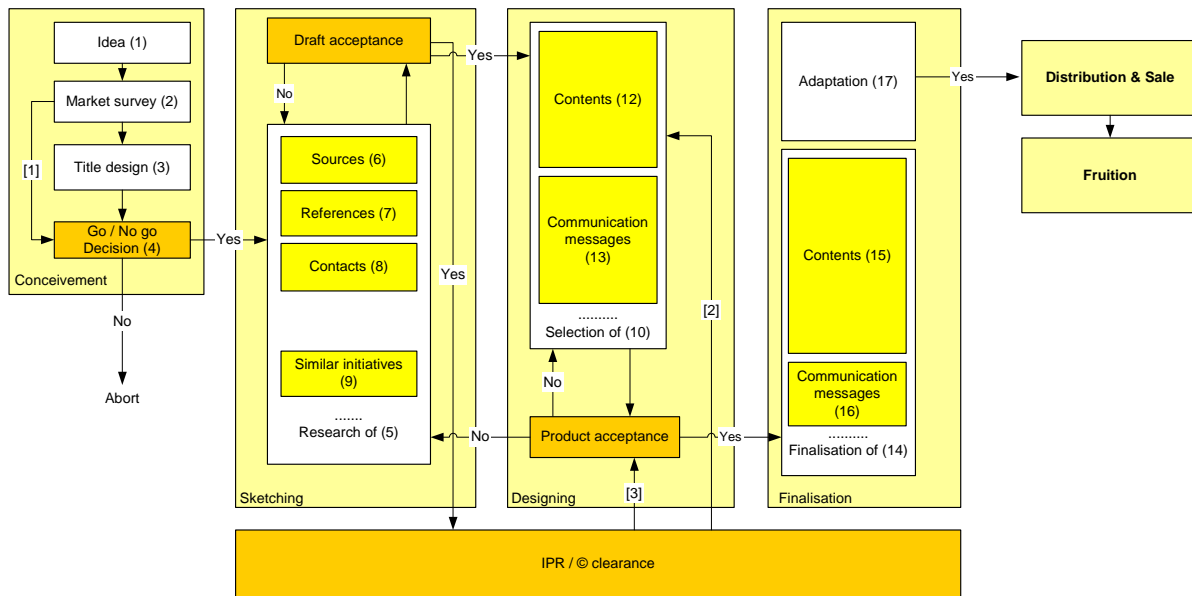
The cost control, and overall, IPR and copyright management are usually managed in a parallel stream. As far as the impacts are concerned, cost control has to monitor that process development is kept in line with expectations and budgets retaining its profitability (ore even increasing it whenever possible) and therefore is a good part of the standard management process. On the other hand IPR and copyright management main interactions occur whenever some asset cannot be cleared and therefore has to be replaced. Such event may occur at any step and the impact may be marginal or relevant depending on a set of possible combinations of factors:

- The relevance for the project of the object that could not be cleared;
- The reason for lack of clearance;
- The stage of the process:

¹ Such a decision is based on market data and production cost analysis to ensure the expected return on investment (ROI)

² At this step starts a parallel process for managing © and IPR clearance performed in an asynchronous parallel fashion.

- The availability of a replacement / equivalent object...



[1] There is no market

[2] There is a problem that may affect the Designing process and / or require product re-design

[3] There is a clearance problem so serious that is worth stopping the process

Figure 1. Content production flow.

2.1 The Mobile factory case

Given the process described before we will address now the points where the integration with AXMEDIS will introduce significant changes. The present section is aimed specifically to cover the first part of the experimentation phase; that is: the content production. In more details, we will present here how we expect to integrate AXMEDIS into the existing production chain. As already mentioned we will operate on a by difference approach and therefore here we will focus only on the steps that are affected by AXMEDIS pointing out why they are affected and how. This is initially done by highlighting the affected process steps and then describing why. The revised proves will then look as follows:

Activity	Task	Involved roles
Idea (1)		Management Marketing Department Press Office
Market survey (2)		
Initiative design (3)		
Go / No Go decision ³ (4)		Management
Research of (5)	Sources (6) References (7) Contacts (8) ... Similar initiatives (10)	Management Marketing Department Press Office Legal Department
Draft acceptance ⁴ (if positive the next step starts if not the previous is reiterated)		Management Marketing Department Legal Department
Selection of (11)	Content (12) Communication message (13)	Marketing Department Designer Press Office Legal Department

³ Such a decision is based on market data and production cost analysis to ensure the expected return on investment (ROI)

Activity	Task		Involved roles	
Formal authorisation to start production <i>(if positive the next step starts if not the previous is reiterated. In this case IPR/© clearance should have been completed if not process may be suspended / stopped / cancelled)</i>	IPR/© clearance		Management Marketing Department Legal Department	
Adaptation & Finalisation of (14)	Content (15) Communication message (16)	Costs control	IPR/© Contracts management	Production department Outsourced service Press Office Marketing manager Legal department Company accountant
Marketing (17)	Promoting	Costs & revenues control		Marketing manager Legal department
Distribution (18)	Revenue management			Production department or Outsourced service Marketing manager Legal department Company accountant

From the table is evident the impact of AXMEDIS on the overall and especially on the production process, the reason is reported hereafter also taking into account what previously mentioned about the traditional process.

Activity	Reason
Research ... (5)	During the search process the P2P infrastructure, AXEPTool and Query support will reduce efforts related to searching content and address IPR/© clearance.
Editing of (11) Finalisation of (17) Production	During these process phases the composition and formatting engines will reduce efforts related to content production/aggregation while ensuring proper IPR/© management.
Marketing	P2P infrastructure, AXEPTool and Query support will reduce efforts related to content promotion and distribution (in terms of samples). The framework will help easing up addressing IPR/© issues enabling the definition of proper rules, licenses and distribution agreement in a simpler form.
Distribution & rights selling	P2P infrastructure, AXEPTool and Query support will reduce efforts related to content distribution (broadcast, active selections, publishing rules...). The framework will help easing up addressing IPR/© issues enabling the definition of proper rules, licenses and distribution agreement (including revenue management) in a simpler form.

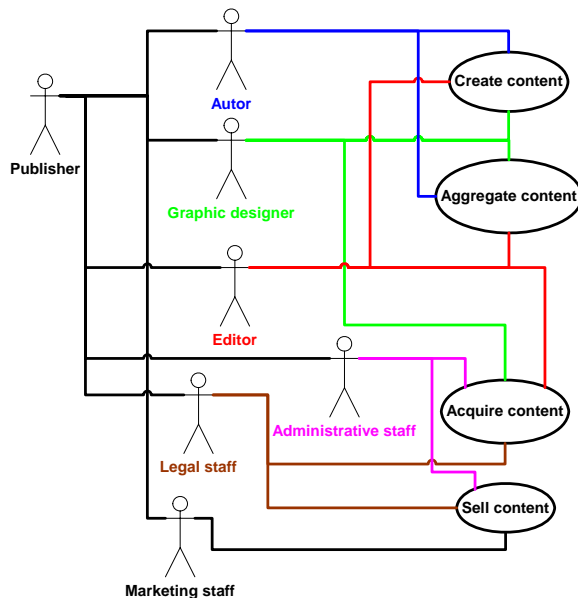
The content production process in the AXMEDIS framework for the mobile environment has as a starting point the transposition of our archive on history of arts from its original format to an intermediate format that will represent the “raw” one for our experiment in AXMEDIS. This is just a part of a more complex process of content processing, filtering and adapting that covers mainly the last part of the value chain (even if it has direct impacts on some initial steps as if certain issues are not addressed at production time it will not be possible to solve them at this stage). What just stated would be valid per se in the publishing environment but is even more in the case of the specific demonstrator as the mobile environment devoted to a large audience with tailored services and applications. Having said this let’s focus on the operations to be performed that basically are:

- Acquire content
- Aggregate content
- Sell content

What just described is therefore translated in the following UML schemas (where for simplicity we have used a colour code to identify relations between actors and objects (please see the following picture).

⁴ At this step starts a parallel process for managing © and IPR clearance performed in an asynchronous parallel fashion.
AXMEDIS Project

In this scenario the first three operations are occurring at the mobile factory and mainly foresee a set of actors operating with different level of competencies and responsibilities in the process. More specifically we have that:



Publisher – manages the overall process in terms of strategies and decision-making.

Author – creates the message conveying or accompanying the content.

Graphic designer – takes care of content graphic finalisation process and of all QA aspects related to graphic.

Editor – takes care of content finalisation process and of all QA aspects related to content.

Administrative staff – manages the administrative aspect of the process (including but not limited to provisioning)

Legal staff – manages the legal aspect of the process (including but not limited to IPR, © clearance, protection, licensing...)

Marketing staff – manages the promotional aspect of the process as well as the distribution policies in cooperation with the publisher and the legal staff.

At the mobile factory all these actors will have access to AXMEDIS framework, each for own role and activities. The result of their cooperation (given the previously mentioned impacts on the process) will lead to the generation of the content that will then be delivered (in broadcast) to the AXMEDIS users mobile devices and used by them for their daily operation as described in other sections of the document.

2.2 Mobile factory overall architecture

We have already mentioned the bi-partition of the environment in a mobile factory and Mobile Portal Application given the nature of the demonstrator and its purpose. The present section focuses on the mobile factory that will comprise two interacting elements: AXMEDIS system and *learn eXact*[®] (LEX) production one. The AXMEDIS system has been described in other relevant documents (reported also in the reference section) it is now time to provide a description of LEX system. LEX is a proprietary environment developed in GIUNTI Labs during the past ten years and devoted to cover the whole value chain of e-learning publishing and fruition in full compliancy with international standards. Basically it comprises the following elements:

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

The most relevant aspect in the D-Repository connection is the possibility to use their content either as source or destination of the packaging process as all that is needed is the compatibility with IMS packaging standard. This feature was developed originally to grant users the possibility to continue exploiting the already in place content sources or distribution facilities and infrastructure even in the occasion of a shift in either production or delivery framework. The same principle brought to the development of a set of API for interconnecting with external back-ends; this is actually the way that will be followed to exploit AXMEDIS empowered production and distribution value chain. As stated in the TA it is expected that each distributor may keep on using own distribution media, but should also take advantage of the powerful support provided by AXMEDIS. Given the layered structure of GIUNTI Authoring and Learning Environment, this has been possible operating at API level in the area previously generically marked as “backend” (for example SAP...).

More specifically it has been necessary to define a set of new API to allow AXMEDIS co-operation with the new third party environment. In more detail we have the following schema:

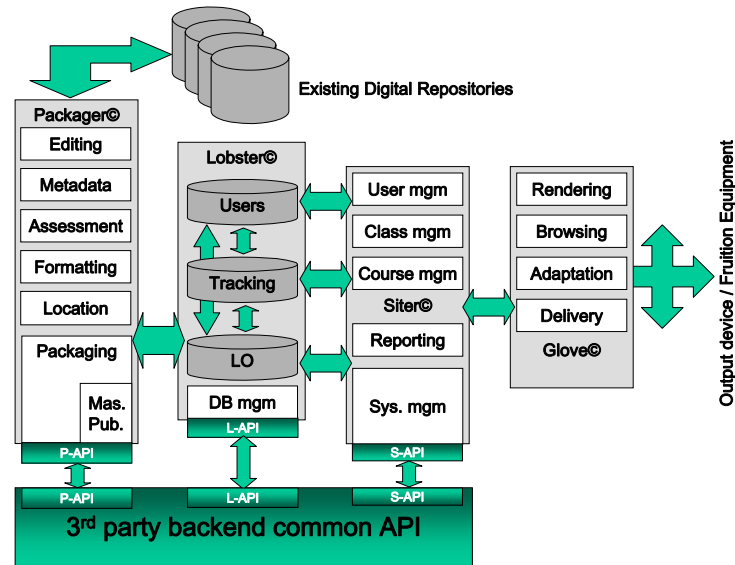


Figure 2. Mobile Factory architecture.

From the previous picture it is evident that the two environments have always a “mediated” interaction, and this is achieved through proper APIs sets. Having said so it is necessary to examine in more detail the level of functionalities covered by the various set of API in respect to the process value-chain. In the following diagram is summarised the possible level of interaction between AXMEDIS as a backend and our environment. The set of common API allow interaction mainly at packaging, distribution and fruition level (at least as far as content delivery is concerned) even though the rendering tool (Glove) is designed only for e-learning purposes and therefore is not compatible with the overall aim if the project. On the other side at design and development phase the possibility of interaction is limited (especially at design) to data import/export.

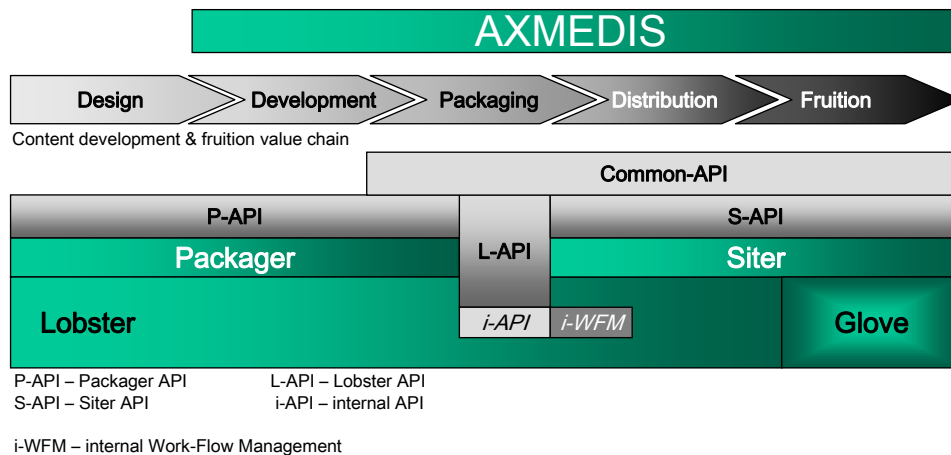


Figure 3. Internal WorkFlow Management.

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

The setup shows a company network with an Intranet on which there are two centralized servers: one web server for the *eXact Siter* and one XML server for the *eXact Lobster*[®] (Tamino[®] XML server plus access layer). The platform users can comprise two types: Packager users (from which the authors publish Courses and LO on the *eXact Lobster*) and *eXact Siter* clients (from which the users access the platform services and courses).

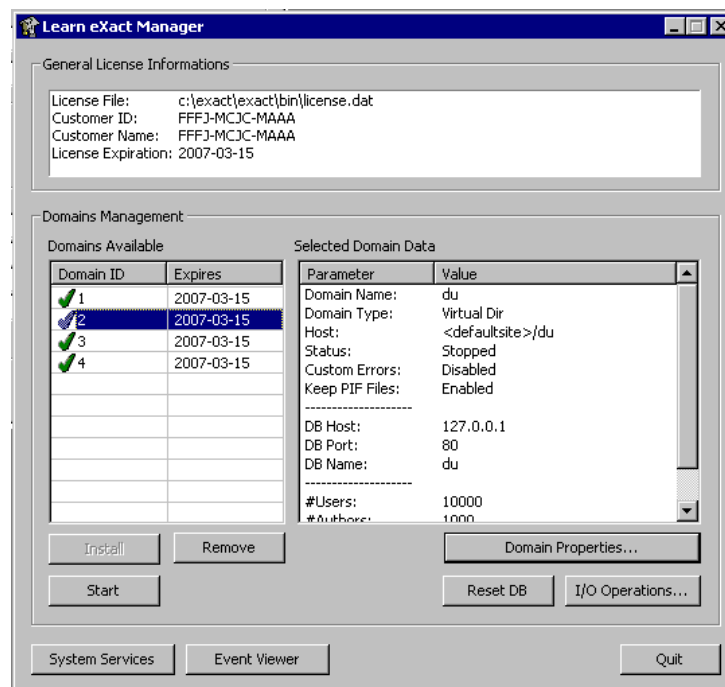
Note: Actually, a massive use of the *eXact Lobster* database and of its "Working Areas" requires a system installed on two servers. The first one is the *learn eXact* front-end, for publishing and management; the second one is specific for Tamino. Please contact the *learn eXact* technical support (www.learneXact.com) if you need any further information.

Note: In system architectures with intensive information traffic, we suggest you raise the data processing timeout limit from 90 (default) to up to 600 seconds. Please follow the procedure below:

1. From Windows *Start* menu go to *Settings > Control Panel > Administrative Tools > Internet Services Manager*
2. On the *Tree* panel in the *Internet Information Services (IIS) window*, spread all *Site* structures and locate the *eXact Virtual Directory* (usually in *Default Web Site*)
3. Select *Properties* from *eXact* right click menu. In the new window, select *Home Directory*. N.B.: If your Windows version is earlier than XP, the *Home Directory* may NOT be available at the level indicated. In this case, please right click on the *Web Site* that includes the *eXact Virtual Directory* and select *Properties* from the pop-up menu. Please select *Home Directory* In the new window
4. On the *Home Directory*, please select *Configuration...*
5. In the new window, select *App Options*, then type in a new, higher value in the *ASP script Timeout* filed (default value: 90 sec., suggested value: 600 seconds or higher). This way , the system can successfully run complex operations, even with large amounts of data.

2.3.1 *learn eXact* Manager

From the *learn eXact Manager* you can fully manage your *learn eXact*[®] domains. To access *Learn eXact Manager* please access the *Start* menu on Windows and locate *Learn eXact Manager* within the *Program Files>Interactive Labs>Learn eXact v3.0* folder:



On this dialog window you may:

- Modify Domain Properties (enable/disable Http custom errors, enable/disable blackboard link, keep PIF files)

DE9.5.4 Integrated Prototype of content Production and Distribution on-demand for Mobile Phones, and new generation of PDAs

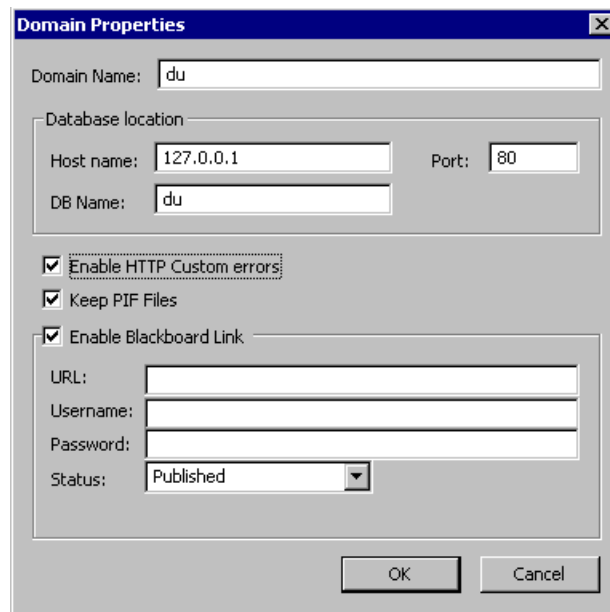
- Perform massive import/export of user data
- Reset your Lobster database (click on the *Reset DB* button). If you perform this last action, all data on current database will be lost.

It is then possible to customize some default settings when installing new domains.

Note: You need to restart IIS in case you have customized default System Settings in your *learn eXact* Web Interface (*Administration Page>Settings>System*: please refer to the section entitled "Learning Management within the eXact Siter" for more information).

2.3.2 To Edit Domain Properties

1. On the *learn eXact* Manager main page, select a domain on the *Available Domains* pane
2. Click the *Domain Properties* button
3. The *learn eXact* Manager *Domain Properties* window displays:



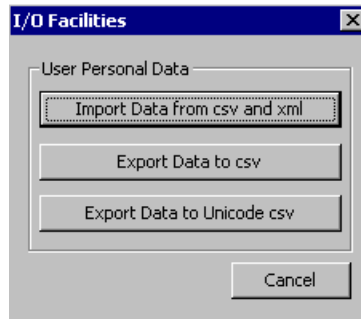
- Select the *Enable HTTP Custom Errors* checkbox to enable all HTTP custom errors report for the selected domain
- Select *Keep PIF files* to preserve pif files within the domain
- Select *Enable Blackboard Link* to activate a domain blackboard link (complete spaces below with blackboard link URL, Username and Password to access it and its status - *Locked* or *Published*)

2.3.3 To install a new domain possibly keeping default settings

Commonly, when you install a new domain from the *learn eXact Manager* window (click *Install*), you may choose to copy Siter and Glove themes after "default" theme. This will allow to correctly complete "theme.xml" and Siter and Glove "domainName" folders for image upload. If you decide not to take this option and, in any case, if you change your mind and decide to "manually" copy and overwrite these files on a second time (as PC administrator), YOU HAVE TO reload both themes as application administrator. If you don't, the system will cache info and will keep "default" theme path. The same applies to strings. As an alternative to application reload, you may restart IIS or your PC as PC administrator.

2.3.4 To perform massive import/export of user data

To import/export massive sets of users into/from a given domain, on *learn eXact Manager* select the *I/O Operations...* button.



2.3.5 To import user data from a CSV file

To import users from a Comma Separated Values (.csv) file, please select the *Import data from csv and xml* option. In order to import a set of users from a CSV file, please make sure that the CSV file conforms to the following specifications:

1. The first line of the file should contain field names, separated by a comma (no commas other than the separating ones are admitted on this line).
2. All fields should be separated by commas.
3. Each field containing any comma or colon should be in quotation marks.
 - In any other case the fields should not be necessarily in quotation marks.
 - If any field already contains quotation marks, you should insert them twice.
4. All records, except for the last one, should be terminated by an end-of-line character:
 - "\r\n" in a Windows environment
 - "\n" in a Unix environment
 - "\r" in a Mac environment

For the purpose of clarification, given a record of the following kind:

```
John Smith
Main Street, #11
"Springfield"
```

it could be transformed in the following way:

```
John Smith,"Main Street, #11","""Springfield"""
```

or else

```
"John Smith","Main Street, #11","""Springfield"""
```

More in general, a csv file valid for the output could be:

```
=====
Import.csv
=====
"First Name","Second Name","Last Name","Gender","Date of Birth","E-mail","Address","PO-
Box","Locality","Postal Code","Country","Language","Username","Password"
John,,Smith,M,17-02-1974,j.smith@giuntilabs.it,"Main Street, #11",,"""Springfield""",16030,United Kingdom,"uk","jsmith","john"
User0001,User0001,User0001,0,17-02-1974,User0001@giuntilabs.it,,,,,it,NewUser0001,
User0002,User0002,User0001,0,17-02-1974,User0002@giuntilabs.it,,,,,it,NewUser0002
"User0003","User0003","User0003","0","17-02-1974","User0003@giuntilabs.it","","","it","User0003",""
=====
```

Of course all changes are made in conformance with the above described rules are valid.

Note: The CSV import operation does not include the assignment of any user role to imported users. It is up to the System Admin to assign them a basic "User" role within the Siter. Refer to the [User and Administrator Profiles](#) section for more information.

2.3.6 To export user data to a CSV file

To export users to a Comma Separated Values (.csv) file, please select the *Export data to csv* option. Choose your destination directory, enter a valid file name and select *Save*.

2.3.7 To export user data to a Unicode CSV file

To export users to a Comma Separated Values (.csv) Unicode file, please select the *Export data to csv Unicode* option. Choose your destination directory, enter a valid file name and select *Save*.

2.3.8 Client Requirements for Web Access

Hardware Requirements:

CPU: Pentium II class

RAM: 64 MB

Software Requirements:

OS: Windows® 98 SE, ME, NT Workstation 6a, 2000 Professional (Service Pack2) or higher, XP Home/Professional or higher.

Web browser: Microsoft® Internet Explorer 5.0 5.5 6.0, Netscape® Communicator 7.2 or higher

Other: Java Virtual Machine (Microsoft JVM 5.0 or Sun JRE 1.4.2_02 recommended)
Additional plug-ins required to view animations and movies

Note:

The first release of Windows XP includes an Internet Explorer version without the virtual machine required for Java encoding (necessary for *eXact Glove*). Subsequently, after a legal dispute conducted by Sun against Microsoft, Redmond House included this component in Service Pack 1 for XP. To view *eXact Glove* on XP Platform, it is necessary to: install the JRE provided by Sun, or else install the Service Pack 1 for Win XP provided by Microsoft.

2.3.9 eXact Packager Client Installation

The *eXact Packager* client application can be installed directly from the CD-Rom on any client workstation. On the client machine, run Setup.exe in the *eXact Packager* folder of the Installation CD and follow the instructions appearing on the screen.

Hardware Requirements:

CPU: Pentium III class or higher

RAM: 128 MB minimum (256 MB suggested)

Free Disk Space: 50 MB (required for the installation)

Network Card

Software Requirements:

OS: Windows® 98 Second Edition
Windows NT Workstation Service Pack 6a (SP6a) or higher,
Windows 2000 Professional Service Pack 2 (SP2) or higher,
Windows XP Home/Professional (SP1) or higher

Web Browser: Microsoft® Internet Explorer 6.0 or higher

Other: Microsoft .NET Framework 1.1; some additional plug-ins may be needed to view animations and movies

Note: Both *eXact packager* and *learn eXact* Server (Lobster/Siter) require .Net Framework, version 1.1 (.4322). The Framework must be installed on your PC following these indications: Windows Update⁵ direct Download.⁶

⁵ <http://v4.windowsupdate.microsoft.com/en/default.asp>

⁶ <http://www.microsoft.com/downloads/details.aspx?FamilyId=262D25E3-F589-4842-8157-034D1E7CF3A3&displaylang=en>

2.3.10 Types of eXact Packager License

There are two different types of license:

- Server License: if a server fingerprint has been produced, you can install the *eXact Packager* in accordance with the number of client workstations indicated in the license file within the terms of the license agreement.
- Client License: each *eXact Packager* installation requires the fingerprint of the client workstation which permits a remote use of the Packager, without any connection to the LOBSTER archive server.

2.3.11 To request and install a Client License

1. After installing the *eXact Packager*, select *Programs > Interactive Labs > eXact Packager 2.0 > Request License* in the Windows® Start menu.

The system will automatically calculate the fingerprint code for the unique serial number relating to a set of hardware devices installed on the machine. The fingerprint will appear in a special dialog window.

2. To run an e-mail message on the predefined e-mail application, click on the *eMail Request* button.
3. Insert your data and fingerprint code.
4. Send the message to licenses@learneXact.com.

When the license file is returned to you, save it in the main folder where the application XPackager.exe has been installed.

Note: In order to produce a fingerprint of either the client or server workstations, a network board and NetBIOS must be installed.

2.3.12 To uninstall eXact Packager

To uninstall eXact Packager just perform the standard Windows application removal procedure:

- From Control Panel launch "Add/Remove programs"
- Select "eXact Packager v1.x" and click on "Change/Remove"
- Follow the displayed instructions.

Once the procedure has been performed, all files and folders created by the installation program will be removed from your disk. However, all data added after the installation will be preserved on disk (Client Licence file, Courses, Resources, log files).

2.4 Interfacing Lex with a backend

eXact Lobster is a full XML and eLearning standards native content storage solution that enables to store and retrieve contents (raw assets, learning objects and courses) and to store, query and retrieve their related metadata, in a common and interoperable digital repository. Lobster implements a Web Service that enables third party applications to use Lobster for publishing, querying and retrieving contents and their metadata by means of Web Services, SOAP, HTTP and FTP technologies. The present document describes in detail this Web Service. Lobster relies on the XML binding of the IMS Content Packaging Specification (IMS CP). Any content to be published has to be made available to Lobster as a Package Interchange Format (PIF). Therefore from now on, with the term *package*, we refer to all types of contents: asset, learning objects, and courses. A PIF is a ZIP archive which contains an xml file, named *imsmanifest.xml*, and some (or all) binary files referenced into the *imsmanifest.xml*. The *imsmanifest.xml*, schema available at www.imsglobal.org⁷, contains several information used by the Lobster to store and retrieve packages, in particular:

1. The value of the attribute **identifier** of the element **manifest**, it is used by Lobster to identify the package once published. Therefore this information must be explicitly expressed when referring to

⁷ <http://www.imsglobal.org/content/packaging/>

the package in some of the methods of the Web Services.

NOTE When publishing or updating, is the third party application responsibility to check consistency between the value of the identifier of the manifest file and the values of the identifier passed as parameter into the Web Services methods.

2. The value of the element **metadata** and of its sub-element is used by the Lobster to retrieve packages on the base of the values of their metadata.

2.4.1 Web Service Interface Description

This paragraph contains a detailed description of the Lobster Web Service. Each method of the interface is described by a table; the related information are structured as shown in the table below:

<i>Name of the method</i>	
<i>Description of the method</i>	
<i>XML of the parameters passed by the method</i>	<i>Parameters passed by the method</i>
<i>XML of the parameters returned by the method</i>	<i>Parameters returned by the method</i>
<i>Examples and/or Notes (optional)</i>	

Login()	
Performs mandatory login procedure on the Service. By providing authentication information this function opens a working session and returns a session identifier. Every operation on Lobster must start with a call to Login() and must end with a call to Shutdown().	
<pre><message name='Login'> <part name='Domain' type='xsd:string' /> <part name='User' type='xsd:string' /> <part name='Pass' type='xsd:string' /> </message></pre>	Domain name Username Password
<pre><message name='LoginResponse'> <part name='SID' type='xsd:string' /> </message></pre>	Session ID

Shutdown()	
Performs logout operation closing the specified session.	
<pre><message name='Shutdown'> <part name='SID' type='xsd:string' /> </message></pre>	Session ID
<pre><message name='ShutdownResponse'> </message></pre>	

GetVersion()	
Retrieves Lobster Web Service version number ("4.0" for the version of this document).	
<pre><message name='GetVersion'> </message></pre>	
<pre><message name='GetVersionResponse'> <part name='Version' type='xsd:string' /> </message></pre>	Version number

KeepSessionAlive()	
Keeps a session alive. Clients must call this function periodically during a session, with the frequency provided by the value of the tag <pollinginterval> returned by GetDomainInfo().	
<pre><message name='KeepSessionAlive'> <part name='SID' type='xsd:string' /> </message></pre>	Session ID
<pre><message name='KeepSessionAliveResponse'> </message></pre>	

GetDomainInfo()	
Retrieves information about the domain you have logged in. The function provides general information such as material location, FTP location, authentication data.	
Information are provided in XML format:	
<pre><message name='GetDomainInfo'> <part name='SID' type='xsd:string' /> </message></pre>	Session ID
<pre><message name='GetDomainInfoResponse'> <part name='XMLResult' type='xsd:string' /> </message></pre>	Information XML*
* this is an example of the XML format returned:	
<pre><?xml version="1.0"?> <domaininfo> <domainurl>:80/domainname</domainurl> <coursedomainpath>nameofdomain</coursedomainpath> <ftpuser>eXactFTPUser</ftpuser> <ftppassword>ftppass</ftppassword> <ftpport>21</ftpport> <pollinginterval>60</pollinginterval> </domaininfo></pre>	

FindPackage()	
OBSOLETE. Use FindPackageInfo2() instead.	

GetPackageFolder()	
Retrieves the folder where the Package specified by PackageID is deployed on the server	
<pre><message name='GetPackageFolder'> <part name='SID' type='xsd:string' /> <part name='PackID' type='xsd:string' /> </message></pre>	Session ID Package ID
<pre><message name='GetPackageFolderResponse'> <part name='Folder' type='xsd:string' /> </message></pre>	Folder name

InsertPackageRequest()	
Makes a request to store a new package on Lobster. A package unique identifier must be provided with the request; the unique identifier must be equal to the manifest identifier to be uploaded. The Object type provides information about the type of the package: Course or Learning Object.	
The return value provides the filename you have to use for PIF uploading.	
<pre><message name='InsertPackageRequest'> <part name='SID' type='xsd:string' /> <part name='PackID' type='xsd:string' /> <part name='ObjType' type='xsd:unsignedInt' /> </message></pre>	Session ID Package ID Object Type*
<pre><message name='InsertPackageRequestResponse'> <part name='FileName' type='xsd:string' /> </message></pre>	Name of the file to be uploaded via FTP
<p>* the "Object Type" parameter should have one of the following values:</p> <p>0: The package contains a Course</p> <p>1: The package contains Resources/Learning Objects</p>	

UpdatePackageRequest()	
Makes a request to update a package on Lobster.	
To perform an Update operation, the user connected must have <i>write</i> permissions on the package.	
<pre><message name='UpdatePackageRequest'> <part name='SID' type='xsd:string' /> <part name='PackID' type='xsd:string' /> </message></pre>	Session ID Package ID

<code><message name='UpdatePackageRequestResponse'></code>	Name of the file you must upload
<code> <part name='FileName' type='xsd:string' /></code>	via FTP
<code></message></code>	

NotifyUploadDone()

Notifies that an FTP upload operation has been performed successfully. This method has to be called after a call to **InsertPackageRequest()** or **UpdatePackageRequest()**.

<code><message name='NotifyUploadDone'></code>	
<code> <part name='SID' type='xsd:string' /></code>	Session ID
<code></message></code>	
<code><message name='NotifyUploadDoneResponse'></code>	
<code></message></code>	

DeletePackage()

Deletes a package from Lobster. To perform a Delete operation, the user connected must have *write* permissions on the package.

<code><message name='DeletePackage'></code>	
<code> <part name='SID' type='xsd:string' /></code>	Session ID
<code> <part name='PackID' type='xsd:string' /></code>	Package ID
<code></message></code>	
<code><message name='DeletePackageResponse'></code>	
<code></message></code>	

GetWAList()

Returns the list of the Working Areas of the user.

<code><message name='GetWAList'></code>	
<code> <part name='SID' type='xsd:string' /></code>	Session ID
<code></message></code>	
<code><message name='GetWAListResponse'></code>	
<code> <part name='Result' type='xsd:string' /></code>	
<code></message></code>	

* this is an example of the XML format returned:

```
<XML>
  <WA>
    <ID>100</ID>
    <TITLE>My WA</TITLE>
    <DESCR>My WA Description</DESCR>
  </WA>
</XML>
```

SetInsertOptions()

Specifies if the package to be published must be tagged as “public accessible” and/or must be inserted into a particular Working Area.

<code><message name='SetInsertOptions'></code>	Session ID
<code> <part name='SID' type='xsd:string' /></code>	WA identifier for automatic
<code> <part name='TargetWAID' type='xsd:string' /></code>	enrollment into a WA.
<code> <part name='AllowPublicAccess' type='xsd:int' /></code>	1 for public access material, 0
<code></message></code>	otherwise.
<code><message name='GetWAListResponse'></code>	
<code> <part name='Result' type='xsd:string' /></code>	
<code></message></code>	

* this is an example of the XML format returned:

```
<XML>
  <WA>
    <ID>100</ID>
    <TITLE>My WA</TITLE>
    <DESCR>My WA Description</DESCR>
  </WA>
</XML>
```

GetPackageInfo()	Available in version 2.3
Retrieve status information about the specified Package.	
<pre><message name='LexSOAPConnector.GetPackageInfo'> <part name='SID' type='xsd:string' /> <part name='PackID' type='xsd:string' /> </message></pre>	Session ID Package ID
<pre><message name='LexSOAPConnector.GetPackageInfoResponse'> <part name='Result' type='xsd:string' /> </message></pre>	XML Result
<p>This is an example of the XML format returned:</p> <pre><package packid="Course1" status="locked" islastmodifier="yes"> <userrights canupdate="yes" /> <allowedstatus>unlocked</allowedstatus> </package></pre> <p>The <code>status</code> attribute shows the current Package status. The <code>islastmodifier</code> is true when the current user is the last modifier of the status. If the status is locked and the last modifier is the current user, it means that the current user is the user that has lock the material. The <code>canupdate</code> attribute is true if the user has write privilege on the Package. The list of the <code>allowedstatus</code> tags shows a list of the allowed status that the current user can set.</p>	

SetStatus()	Available in version 2.3
Change the current status of a Package.	
<pre><message name="LexSOAPConnector.SetStatus"> <part name="SID" type="xsd:string" /> <part name="PackID" type="xsd:string" /> <part name="Status" type="xsd:string" /> </message></pre>	Session ID Package ID The new Package status. Can be "locked" or "unlocked".
<pre><message name="LexSOAPConnector.SetStatusResponse"> </message></pre>	
<p>The SetStatus() can easily throw and E_PERMISSION exception if the current user has not the privilege to change the status. Use GetPackageInfo() before calling SetStatus() in order to check if the user has the correct rights.</p>	

FindPackageInfo()	Available in version 3.1
OBSOLETE. Use FindPackageInfo2() instead.	

GetPackageHistory()	Available in version 4.0
Provides information about the version history of the specified package.	
<pre><message name="LexSOAPConnector.GetPackageHistory"> <part name="SID" type="xsd:string" /> <part name="PackID" type="xsd:string" /> </message></pre>	Session ID Package ID

</message>	
<pre><message name="LexSOAPConnector.GetPackageHistoryResponse"> <part name="Result" type="xsd:string"/> </message></pre>	
XML Result	
This is an example of the XML format returned:	
<pre><materialhistory gid="100001" packid="TEST"> <version datetime="2006-05-03T13:59:05.698+00:00" label="1.0" number="1" oldstatus="draft" status="draft" user="00" useremail="admin@host.example.net" userfn="System Admin"> <comment>Fist Version</comment> </version> <version datetime="2006-05-03T13:59:44.449+00:00" label="1.1" number="2" oldstatus="draft" status="draft" user="00" useremail="admin@host.example.net" userfn="System Admin"> <comment>Some fixing...</comment> </version> <version datetime="2006-05-03T14:00:30.621+00:00" label="2.0" number="3" oldstatus="draft" status="published" user="00" useremail="admin@host.example.net" userfn="System Admin"> <comment>This is the final release.</comment> </version> </materialhistory></pre>	

NotifyUploadDone2()	Available in version 4.0
Like NotifyUploadDone(), but provides the possibility to specify a comment and a label for the new version that you are publishing.	
<pre><message name="LexSOAPConnector.NotifyUploadDone2"> <part name="SID" type="xsd:string"/> <part name="VersionLabel" type="xsd:string"/> <part name="VersionComment" type="xsd:string"/> </message></pre>	
	Session ID Label Comment
<pre><message name="LexSOAPConnector.NotifyUploadDone2Response"> <part name="Result" type="xsd:unsignedInt"/> </message></pre>	
	New version number
<i>This function also returns the incremental number of the new version that has been created.</i>	

SetVersionStatus()	Available in version 4.0
Set the status of a particular version of a package.	
<pre><message name="LexSOAPConnector.SetVersionStatus"> <part name="SID" type="xsd:string"/> <part name="PackID" type="xsd:string"/> <part name="VersionNumber" type="xsd:unsignedInt"/> <part name="VersionStatus" type="xsd:string"/> </message></pre>	
	Session ID Package ID Version Number Version Status
<pre><message name="LexSOAPConnector.SetVersionStatusResponse"> </message></pre>	
Version status can be one of:	
<ul style="list-style-type: none"> draft review 	

- *final*
- *published*
- *obsolete*

GetPackageTypes()	Available in version 4.0
Provides the list of the custom package types supported by the Lobster server.	
<pre><message name="LexSOAPConnector.GetPackageTypes"> <part name="SID" type="xsd:string"/> </message></pre>	Session ID
<pre><message name="LexSOAPConnector.GetPackageTypesResponse"> <part name="Result" type="xsd:string"/> </message></pre>	XML Result
This is an example of the XML format returned:	
<pre><packtypes> <packtype code="course" preview="glove" aggregationlevel="course" delivery="glove" icon="" name="" /> <packtype code="resource" preview="asset" aggregationlevel="resource" delivery="asset" icon="" name="" /> <packtype code="lo" preview="asset" aggregationlevel="resource" delivery="asset" icon="" name="" /> <packtype code="qti" preview="asset" aggregationlevel="resource" delivery="asset" icon="" name="" /> <packtype code="wctcourse" preview="no" aggregationlevel="course" delivery="no" icon="" name="" /> <packtype code="mycustomtype" preview="no" aggregationlevel="resource" delivery="glove" icon="customtype.gif" name="My Custom Type"/> </packtypes></pre>	
<i>Please note that the first five packtypes are the standard Learn eXact package types and they are always present.</i>	

FindPackageInfo2()	Available in version 4.0
From Lobster 4.0 substitutes FindPackage() and FindPackageInfo() to performs XPath find operation.	
<pre><message name="LexSOAPConnector.FindPackageInfo2"> <part name="SID" type="xsd:string"/> <part name="Query" type="xsd:string"/> <part name="PackTypes" type="xsd:string"/> <part name="Start" type="xsd:unsignedInt"/> <part name="Size" type="xsd:unsignedInt"/> </message></pre>	Session ID XPath Query Package type list Starting element Result page size
<pre><message name="LexSOAPConnector.FindPackageInfo2Response"> <part name="Result" type="xsd:string"/> </message></pre>	XML Result
<i>Details on the usage of this method are available in the "Details on the query procedure" section.</i>	

2.4.2 Error Codes

The table below describes the errors codes returned by method calls:

<u>ERROR CODE</u>	<u>HEX VALUE</u>	<u>DESCRIPTION</u>
--------------------------	-------------------------	---------------------------

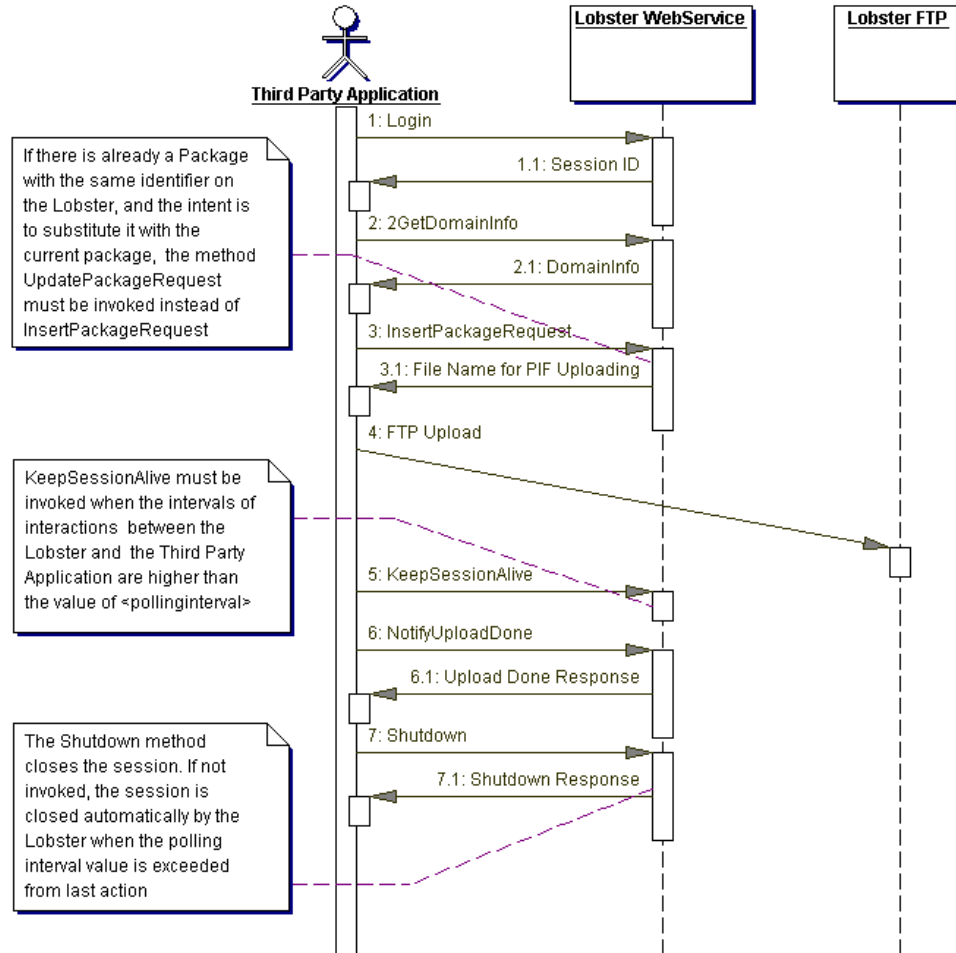
E_FAIL	0x80004005	Generic exception
E_INVALIDSESSION	0x80050001	Invalid session ID
E_ALREADYEXISTS	0x80050002	The element already exists
E_DONOTEXISTS	0x80050003	The element does not exists
E_PASSWORD	0x80050004	Invalid password
E_USERNOTFOUND	0x80050005	User not found
E_PERMISSION	0x80050006	User does not have permissions
E_INVALIDXML	0x80050007	Invalid XML file
E_NOOPERATION	0x80050008	No operation requested
E_FILENOTEXISTS	0x80050009	File does not exists
E_MANIFNOTEXISTS	0x8005000A	Manifest file does not exists
E_INVALIDCOURSEID	0x8005000B	Package/Course ID is not valid
E_LOCKED	0x8005000C	Package is locked
E_TOOMANYUSERS	0x8005000D	Too many users connected
E_USERALREADYLOGGED	0x8005000E	User already logged
E_UNZIPPING	0x8005000F	Error during PIF unzipping
E_DOMAINNOTFOUND	0x80050A01	Domain not found
E_SESSIONNOTOPENED	0x80050A10	Session not opened
E_INVALIDPACKID	0x80050A16	Invalid Package ID format

2.5 Backend Publishing Procedure

The present paragraph describes, step by step, the procedure that a third party application has to follow to publish (insert or update) a package into Lobster. Packages are published using FTP protocol.

1. Invoke Login() in a Lobster Domain with valid authentication data to authenticate and receive a LoginResponse Message
2. Invoke GetDomainInfo() in order to obtain FTP authentication info and receive a GetDomainInfoResponse Message
3. Invoke either InsertPackageRequest() or UpdatePackageRequest() for an Insert or an Update operation. In both cases the identifier of the package to be uploaded is requested.
NOTE: Each package (Course, LearningObject, Asset) published on Lobster, has a unique identifier inside the repository: this identifier is represented by the IMS manifest identifier attribute
4. In case of new package insertion, call SetInsertOptions() in order to specify if the material should be treated as a public access material, or to insert the material into a particular Working Area. By default, new material are private and are not enrolled into a Working Area. Use GetWAList() in order to obtain a list of the Working Areas where the user is enrolled.
5. Perform an FTP transfer or an HTTP(S) transfer of the PIF file. If you choose FTP, use FTP authentication info provided by GetDomainInfo() and upload a file with the name provided by InsertPackageRequest() or UpdatePackageRequest().
NOTE: The PIF file must be in Zip archive format. Call KeepSessionAlive() periodically (the period is provided by GetDomainInfo) if the file transfer is longer than the session timeout (usually about 120 seconds)
6. Invoke NotifyUploadDone() to notify the Lobster when the transfer is finished and receive a NotifyUploadDoneResponse() Message. This response is an implicit confirmation of a successful registration of the package on the repository
7. Invoke Shutdown() to close the connection and receive the ShutdownResponse message.

The following sequence diagram describes the call sequence for a publish operation



2.5.1 Querying and retrieving procedure

To perform a query on Lobster materials, a third party application has to express its query with the Xpath language syntax. The result set is represented in XML format and could be paged. Packages are retrieved using HTTP protocol.

1. Invoke Login() in a Lobster Domain with valid authentication data and receive a LoginResponse Message
2. Invoke GetDomainInfo() in order to obtain some information about content location.
3. Invoke FindPackage() to find the identifier of the packages that satisfy the constraints imposed into the Xpath expression passed as parameter into the FindPackage() method. To locate the package the third party application there are two possibilities to create the URL:

a) **domainurl** + "/" + **coursebasepath** + "/cppid/" + **packageid** + "/" + **version** + "/imsmanifest.xml"

- The **domainurl** returned by DomainInfo() is an absolute path, a string like <http://127.0.0.1:80/domainsample>
- **coursebasepath** is returned by GetDomainInfo().
- "/cppid/" is a fixed string.
- **packageid** is the package identifier, returned by the FindPackageInfo().
- **version** is the wanted version of the package. It can be a positive integer number or one of:
 - "last": request the last version of the package.
 - "lastpub": request the last version of the package in the "published" status.
 - "lastfinal": request the last version of the package in the "final" status.

- Also, when a specific version number is used, this parameter can be followed by the character "f" in order to force the download of the exact version even if it is marked "obsolete". If "f" is not specified and the requested version is obsolete, a more non obsolete version of the package will be provided by the system.
- "imsmanifest.xml" can be used to retrieve the imsmanifest file, but can be any other complete file path inside the package.

b) `protocol+ "/" + host + ":" + port + "/" + domainurl + "/" + coursebasepath + "/" + cppid + "/" + packageid + "/" + version + "/" + imsmanifest.xml`

- protocol is either http or https (usually http) and host is the IP of the machine hosting the WebService, port is the port where the transfer protocol service is available.

NOTE These information can be retrieved by the third party application from the url of the wsdl file, e.g. `http://127.0.0.1:80/LobsterWebService/Lobster.wsdl`

- The domainurl returned by DomainInfo() is an relative path, a string like `:80/domainsample`
- The rest of the URL is the same as the case a).

4. Perform http(s) transfer of the imsmanifest.xml and possibly (all) the binary files referenced by it.
5. Invoke Shutdown() to close the connection and receive the ShutdownResponse message.

2.5.2 Sample of Query and Result Set in XML Format

XPath Query: `[tf:containsText(general/title/langstring, "Package")]`

FindPackage() response:

```
<?xml version="1.0"?>
<ino:response xmlns:ino="http://namespaces.softwareag.com/tamino/response2"
  xmlns:xql="http://metalab.unc.edu/xql/" ino:sessionId="162" ino:sessionkey="546882039">
  <ino:message ino:returnvalue="0">
    <ino:messageLine>XQuery Request processing</ino:messageLine>
  </ino:message>
  <xq:result xmlns:xq="http://namespaces.softwareag.com/tamino/XQuery/result">
    <lexresult identifier="Package_1" publicaccess="true" status="unlocked"
      type="course" format="application/zip" >
      <title>Package 1</title>
      <descr>Package 1 Description</descr>
    </lexresult>
    <lexresult identifier="Package_2" publicaccess="true" status="unlocked"
      type="resource" format="image/jpeg">
      <title>Package 2</title>
      <descr>Package 2 Description</descr>
    </lexresult>
  </xq:result>
  <ino:message ino:returnvalue="0">
    <ino:messageLine>XQuery Request processed</ino:messageLine>
  </ino:message>
</ino:response>
```

NOTE: In relation to the version of Tamino XML Server installed, the response XML format could change.

2.5.3 Lobster WSDL File

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="Lobster" targetNamespace="http://www.giuntlabs.com/Lobster/wsdl/"
  xmlns:wsdl="http://www.giuntlabs.com/Lobster/wsdl/" xmlns:types="http://www.giuntlabs.com/Lobster/type/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:stk="http://schemas.microsoft.com/soap-toolkit/wsdl-extension"
  xmlns:dime="http://schemas.xmlsoap.org/ws/2002/04/dime/wsdl/"
  xmlns:ref="http://schemas.xmlsoap.org/ws/2002/04/reference/" xmlns:content="http://schemas.xmlsoap.org/ws/2002/04/content-type/"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns="http://schemas.xmlsoap.org/wsdl/">
  <types>
    <schema targetNamespace="http://www.giuntlabs.com/Lobster/type/" xmlns="http://www.w3.org/2001/XMLSchema"
      xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
      elementFormDefault="qualified">
      <import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
      <import namespace="http://schemas.xmlsoap.org/wsdl/" />
      <import namespace="http://schemas.xmlsoap.org/ws/2002/04/reference/" />
      <import namespace="http://schemas.xmlsoap.org/ws/2002/04/content-type/" />
    </schema>
  </types>
  <message name="LexSOAPConnector.Login">
```

```
<part name="Domain" type="xsd:string"/>
<part name="User" type="xsd:string"/>
<part name="Pass" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.LoginResponse">
  <part name="Result" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.GetVersion">
</message>
<message name="LexSOAPConnector.GetVersionResponse">
  <part name="Result" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.GetDomainInfo">
  <part name="SID" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.GetDomainInfoResponse">
  <part name="Result" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.FindPackage">
  <part name="SID" type="xsd:string"/>
  <part name="Query" type="xsd:string"/>
  <part name="Start" type="xsd:unsignedInt"/>
  <part name="Size" type="xsd:unsignedInt"/>
</message>
<message name="LexSOAPConnector.FindPackageResponse">
  <part name="Result" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.DeletePackage">
  <part name="SID" type="xsd:string"/>
  <part name="PackID" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.DeletePackageResponse">
</message>
<message name="LexSOAPConnector.InsertPackageRequest">
  <part name="SID" type="xsd:string"/>
  <part name="PackID" type="xsd:string"/>
  <part name="ObjType" type="xsd:unsignedInt"/>
</message>
<message name="LexSOAPConnector.InsertPackageRequestResponse">
  <part name="Result" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.UpdatePackageRequest">
  <part name="SID" type="xsd:string"/>
  <part name="PackID" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.UpdatePackageRequestResponse">
  <part name="Result" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.NotifyUploadDone">
  <part name="SID" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.NotifyUploadDoneResponse">
</message>
<message name="LexSOAPConnector.GetPackageFolder">
  <part name="SID" type="xsd:string"/>
  <part name="PackID" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.GetPackageFolderResponse">
  <part name="Result" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.Shutdown">
  <part name="SID" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.ShutdownResponse">
</message>
<message name="LexSOAPConnector.KeepSessionAlive">
  <part name="SID" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.KeepSessionAliveResponse">
</message>
```

```
<message name="LexSOAPConnector.GetLicense">
  <part name="SID" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.GetLicenseResponse">
  <part name="Result" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.GetWAList">
  <part name="SID" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.GetWAListResponse">
  <part name="Result" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.SetInsertOptions">
  <part name="SID" type="xsd:string"/>
  <part name="TargetWAID" type="xsd:string"/>
  <part name="AllowPublicAccess" type="xsd:int"/>
</message>
<message name="LexSOAPConnector.SetInsertOptionsResponse">
</message>
<message name="LexSOAPConnector.SetStatus">
  <part name="SID" type="xsd:string"/>
  <part name="PackID" type="xsd:string"/>
  <part name="Status" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.SetStatusResponse">
</message>
<message name="LexSOAPConnector.GetPackageInfo">
  <part name="SID" type="xsd:string"/>
  <part name="PackID" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.GetPackageInfoResponse">
  <part name="Result" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.FindPackageInfo">
  <part name="SID" type="xsd:string"/>
  <part name="Query" type="xsd:string"/>
  <part name="Start" type="xsd:unsignedInt"/>
  <part name="Size" type="xsd:unsignedInt"/>
</message>
<message name="LexSOAPConnector.FindPackageInfoResponse">
  <part name="Result" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.GetPackageHistory">
  <part name="SID" type="xsd:string"/>
  <part name="PackID" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.GetPackageHistoryResponse">
  <part name="Result" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.NotifyUploadDone2">
  <part name="SID" type="xsd:string"/>
  <part name="VersionLabel" type="xsd:string"/>
  <part name="VersionComment" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.NotifyUploadDone2Response">
  <part name="Result" type="xsd:unsignedInt"/>
</message>
<message name="LexSOAPConnector.SetVersionStatus">
  <part name="SID" type="xsd:string"/>
  <part name="PackID" type="xsd:string"/>
  <part name="VersionNumber" type="xsd:unsignedInt"/>
  <part name="VersionStatus" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.SetVersionStatusResponse">
</message>
<message name="LexSOAPConnector.GetPackageTypes">
  <part name="SID" type="xsd:string"/>
</message>
<message name="LexSOAPConnector.GetPackageTypesResponse">
  <part name="Result" type="xsd:string"/>
</message>
```

```
</message>
<message name="LexSOAPConnector.FindPackageInfo2">
  <part name="SID" type="xsd:string"/>
  <part name="Query" type="xsd:string"/>
  <part name="PackTypes" type="xsd:string"/>
  <part name="Start" type="xsd:unsignedInt"/>
  <part name="Size" type="xsd:unsignedInt"/>
</message>
<message name="LexSOAPConnector.FindPackageInfo2Response">
  <part name="Result" type="xsd:string"/>
</message>
<portType name="LobsterSoapPort">
  <operation name="Login" parameterOrder="Domain User Pass">
    <input message="wsdlIns:LexSOAPConnector.Login"/>
    <output message="wsdlIns:LexSOAPConnector.LoginResponse"/>
  </operation>
  <operation name="GetVersion" parameterOrder="">
    <input message="wsdlIns:LexSOAPConnector.GetVersion"/>
    <output message="wsdlIns:LexSOAPConnector.GetVersionResponse"/>
  </operation>
  <operation name="GetDomainInfo" parameterOrder="SID">
    <input message="wsdlIns:LexSOAPConnector.GetDomainInfo"/>
    <output message="wsdlIns:LexSOAPConnector.GetDomainInfoResponse"/>
  </operation>
  <operation name="FindPackage" parameterOrder="SID Query Start Size">
    <input message="wsdlIns:LexSOAPConnector.FindPackage"/>
    <output message="wsdlIns:LexSOAPConnector.FindPackageResponse"/>
  </operation>
  <operation name="DeletePackage" parameterOrder="SID PackID">
    <input message="wsdlIns:LexSOAPConnector.DeletePackage"/>
    <output message="wsdlIns:LexSOAPConnector.DeletePackageResponse"/>
  </operation>
  <operation name="InsertPackageRequest" parameterOrder="SID PackID ObjType">
    <input message="wsdlIns:LexSOAPConnector.InsertPackageRequest"/>
    <output message="wsdlIns:LexSOAPConnector.InsertPackageRequestResponse"/>
  </operation>
  <operation name="UpdatePackageRequest" parameterOrder="SID PackID">
    <input message="wsdlIns:LexSOAPConnector.UpdatePackageRequest"/>
    <output message="wsdlIns:LexSOAPConnector.UpdatePackageRequestResponse"/>
  </operation>
  <operation name="NotifyUploadDone" parameterOrder="SID">
    <input message="wsdlIns:LexSOAPConnector.NotifyUploadDone"/>
    <output message="wsdlIns:LexSOAPConnector.NotifyUploadDoneResponse"/>
  </operation>
  <operation name="GetPackageFolder" parameterOrder="SID PackID">
    <input message="wsdlIns:LexSOAPConnector.GetPackageFolder"/>
    <output message="wsdlIns:LexSOAPConnector.GetPackageFolderResponse"/>
  </operation>
  <operation name="Shutdown" parameterOrder="SID">
    <input message="wsdlIns:LexSOAPConnector.Shutdown"/>
    <output message="wsdlIns:LexSOAPConnector.ShutdownResponse"/>
  </operation>
  <operation name="KeepSessionAlive" parameterOrder="SID">
    <input message="wsdlIns:LexSOAPConnector.KeepSessionAlive"/>
    <output message="wsdlIns:LexSOAPConnector.KeepSessionAliveResponse"/>
  </operation>
  <operation name="GetLicense" parameterOrder="SID">
    <input message="wsdlIns:LexSOAPConnector.GetLicense"/>
    <output message="wsdlIns:LexSOAPConnector.GetLicenseResponse"/>
  </operation>
  <operation name="GetWAList" parameterOrder="SID">
    <input message="wsdlIns:LexSOAPConnector.GetWAList"/>
    <output message="wsdlIns:LexSOAPConnector.GetWAListResponse"/>
  </operation>
  <operation name="SetInsertOptions" parameterOrder="SID TargetWAID AllowPublicAccess">
    <input message="wsdlIns:LexSOAPConnector.SetInsertOptions"/>
    <output message="wsdlIns:LexSOAPConnector.SetInsertOptionsResponse"/>
  </operation>
  <operation name="SetStatus" parameterOrder="SID PackID Status">
    <input message="wsdlIns:LexSOAPConnector.SetStatus"/>
  </operation>
```

```
<output message="wsdl:LexSOAPConnector.SetStatusResponse"/>
</operation>
<operation name="GetPackageInfo" parameterOrder="SID PackID">
  <input message="wsdl:LexSOAPConnector.GetPackageInfo"/>
  <output message="wsdl:LexSOAPConnector.GetPackageInfoResponse"/>
</operation>
<operation name="FindPackageInfo" parameterOrder="SID Query Start Size">
  <input message="wsdl:LexSOAPConnector.FindPackageInfo"/>
  <output message="wsdl:LexSOAPConnector.FindPackageInfoResponse"/>
</operation>
<operation name="GetPackageHistory" parameterOrder="SID PackID">
  <input message="wsdl:LexSOAPConnector.GetPackageHistory"/>
  <output message="wsdl:LexSOAPConnector.GetPackageHistoryResponse"/>
</operation>
<operation name="NotifyUploadDone2" parameterOrder="SID VersionLabel VersionComment">
  <input message="wsdl:LexSOAPConnector.NotifyUploadDone2"/>
  <output message="wsdl:LexSOAPConnector.NotifyUploadDone2Response"/>
</operation>
<operation name="SetVersionStatus" parameterOrder="SID PackID VersionNumber VersionStatus">
  <input message="wsdl:LexSOAPConnector.SetVersionStatus"/>
  <output message="wsdl:LexSOAPConnector.SetVersionStatusResponse"/>
</operation>
<operation name="GetPackageTypes" parameterOrder="SID">
  <input message="wsdl:LexSOAPConnector.GetPackageTypes"/>
  <output message="wsdl:LexSOAPConnector.GetPackageTypesResponse"/>
</operation>
<operation name="FindPackageInfo2" parameterOrder="SID Query PackTypes Start Size">
  <input message="wsdl:LexSOAPConnector.FindPackageInfo2"/>
  <output message="wsdl:LexSOAPConnector.FindPackageInfo2Response"/>
</operation>
</portType>
<binding name="LexSOAPConnectorSoapBinding" type="wsdl:LexSOAPConnectorSoapPort">
  <stk:binding preferredEncoding="UTF-8"/>
  <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="Login">
    <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.Login"/>
    <input>
      <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Domain User Pass"/>
    </input>
    <output>
      <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
    </output>
  </operation>
  <operation name="GetVersion">
    <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.GetVersion"/>
    <input>
      <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </input>
    <output>
      <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
    </output>
  </operation>
  <operation name="GetDomainInfo">
    <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.GetDomainInfo"/>
    <input>
      <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID"/>
    </input>
    <output>
      <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
    </output>
  </operation>
  <operation name="FindPackage">
    <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.FindPackage"/>
    <input>
```

DE9.5.4 Integrated Prototype of content Production and Distribution on-demand for Mobile Phones, and new generation of PDAs

```
<soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID Query Start Size"/>
</input>
<output>
  <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
</output>
</operation>
<operation name="DeletePackage">
  <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.DeletePackage"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID PackID"/>
  </input>
  <output>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
  </output>
</operation>
<operation name="InsertPackageRequest">
  <soap:operation
soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.InsertPackageRequest"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID PackID ObjType"/>
  </input>
  <output>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
  </output>
</operation>
<operation name="UpdatePackageRequest">
  <soap:operation
soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.UpdatePackageRequest"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID PackID"/>
  </input>
  <output>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
  </output>
</operation>
<operation name="NotifyUploadDone">
  <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.NotifyUploadDone"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID"/>
  </input>
  <output>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
  </output>
</operation>
<operation name="GetPackageFolder">
  <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.GetPackageFolder"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID PackID"/>
  </input>
  <output>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
  </output>
</operation>
<operation name="Shutdown">
  <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.Shutdown"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID"/>
  </input>
```

```
</input>
</output>
  <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding"/>
</output>
</operation>
<operation name="KeepSessionAlive">
  <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.KeepSessionAlive"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID"/>
  </input>
  <output>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding"/>
  </output>
</operation>
<operation name="GetLicense">
  <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.GetLicense"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID"/>
  </input>
  <output>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
  </output>
</operation>
<operation name="GetWAList">
  <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.GetWAList"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID"/>
  </input>
  <output>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
  </output>
</operation>
<operation name="SetInsertOptions">
  <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.SetInsertOptions"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID TargetWAID AllowPublicAccess"/>
  </input>
  <output>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding"/>
  </output>
</operation>
<operation name="SetStatus">
  <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.SetStatus"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID PackID Status"/>
  </input>
  <output>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding"/>
  </output>
</operation>
<operation name="GetPackageInfo">
  <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.GetPackageInfo"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID PackID"/>
  </input>
  <output>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
  </output>
</operation>
```

```
</output>
</operation>
<operation name="FindPackageInfo">
  <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.FindPackageInfo"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID Query Start Size"/>
  </input>
  <output>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
  </output>
</operation>
<operation name="GetPackageHistory">
  <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.GetPackageHistory"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID PackID"/>
  </input>
  <output>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
  </output>
</operation>
<operation name="NotifyUploadDone2">
  <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.NotifyUploadDone2"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID VersionLabel VersionComment"/>
  </input>
  <output>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
  </output>
</operation>
<operation name="SetVersionStatus">
  <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.SetVersionStatus"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID PackID VersionNumber VersionStatus"/>
  </input>
  <output>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
  </output>
</operation>
<operation name="GetPackageTypes">
  <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.GetPackageTypes"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID"/>
  </input>
  <output>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
  </output>
</operation>
<operation name="FindPackageInfo2">
  <soap:operation soapAction="http://www.giuntlabs.com/Lobster/action/LexSOAPConnector.FindPackageInfo2"/>
  <input>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="SID Query PackTypes Start Size"/>
  </input>
  <output>
    <soap:body use="encoded" namespace="http://www.giuntlabs.com/Lobster/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" parts="Result"/>
  </output>
</operation>
</binding>
<service name="Lobster">
```

```
<port name="LobsterSoapPort" binding="wsdl:LexSOAPConnectorSoapBinding">
  <soap:address location="http://myserver/LobsterWebService/Lobster.wsdl"/>
</port>
</service>
</definitions>
```

2.6 Content Selection & Acquisition for mobile e-applications

In the context of the distribution to mobile demonstrator it is crucial to point out that the content is usually acquired for adaptation and delivery rather than purposely produced, yet in some cases, content is aggregated or purposely produced. Usually the delivered content stems out of some other content production value chains as a further exploitation of already developed content led by customer request. This is for example the case of audio video content that is usually developed for a specific distribution channel and then (due to its success) made available for mobile distribution (soccer, big-brother, music...). A twofold process usually leads the adaptation: **adapting to user needs** and **adapting to user device capabilities**. This has brought in time to the definition of standardised formats for describing both device and user profile. In the rest of this document we will refer to both of them and to the standard formats we have selected (DELI for devices and LIP for users). Both standard are XML based and have been extensively used. In respect to the first we will integrate what developed by IRC (that is DELI and some extensions). For the second we will presently exploit only a subset of the potentialities yet, given the completeness of the LIP format we will be able to easily extend coverage according to needs that may emerge during the experimentation phase. It is worth noting that in any case there is also a set of contents that are purposely prepared to support the consumption of content on mobile device, namely the promotional content that is usually delivered to potential users via mass media and advertisement, but also sometimes directly on the device in the form of a promotional sample. Aim of the present section is to give a basic idea on how to exploit authoring resources available to derive effective content completed by a good set of metadata. A set of relevant info related to these aspects have already been addressed in other documents, namely DE3.1.3 and DE8.2.1, in any case in those documents were primarily addressed general issues while here we focus on the specific instance of the process related to production for mobile devices. As stated in several AXMEDIS documents, is well known that selecting or producing multimedia content for mobile delivery is a rather complex, time consuming and costly task as it requires taking into account that:

- Images are available usually in formats that are not the best suited for mobile delivery;
- Audio is available usually in formats that are not the best suited for mobile delivery;
- Multimedia content presents even more copyright problems than images and audio;
- Content suitability for mobile-based applications may imply adequate post-processing.

Aside to constraints there are market trends and user consumption statistics to take into account in content selection process. In this respect same basic principles are: ***users will tend to select content based on their interest, content relevance (in respect of their interests), content price and availability either of offers or of extraordinarily appealing (according to ones interest) content.*** To achieve the maximum profitability of available content and plan for future acquisition the marketing managers usually keep under control the following variables:

- Content ranking in terms of revenue, acquisition cost, category, user fruition and acceptance;
- Content ranking in terms of balance between cost and revenue (absolute and per category).

Certainly there are many other factors that will be taken into account like the targeted audience in respect of the actual one, age class distributions among target audience... and all sorts of other information that may emerge for statistical analysis of available data. Most managers will also look for unexpected patterns or results and in such case a specific data-mining initiatives are launched. Having said so it is worth adding that in most cases content offers are often set up selecting among least used or requested content (this in other business sector is also extended to left over items) and therefore marketing managers setting up promotional campaigns will be interested in a set of data similar to the previously mentioned ones but from the opposite perspective (namely least requested/used content). In this later case is also possible to cross-merge these data with known user preferences and records of purchase to even prepare personalised offers. To this extent we assume here that in the mobile content factory will be available facilities for generating report and statistic analysis at least on the following info:

Most used/acquired contents	Least used/acquired contents
<ul style="list-style-type: none"> • per user • per distributor • per category 	<ul style="list-style-type: none"> • per user • per distributor • per category
Most used/acquired contents per category	Least used/acquired contents per category
<ul style="list-style-type: none"> • per user • per distributor • per distributor, user 	<ul style="list-style-type: none"> • per user • per distributor • per distributor, user

Furthermore the Statistical analysis of the content usage is also very useful for tuning the service and the structure of the ready to use proposed objects. Among these statistics it would be advisable to have the following ones:

- Content sorted by usage rate
- Content used sorted by category
- Content used sorted by category and usage rate
- Top 10 used Content sorted by category
- Top 20 used Content sorted by category
- Bottom 10 used Content sorted by category
- Bottom 20 used Content sorted by category

It may also be necessary to compute statistics about the access, utilisation, distribution etc. of objects based on the event reports previously generated like the following ones:

- Content sorted by access rate
- Content accessed sorted by category
- Content accessed sorted by category and access rate
- Top 10 accessed Content sorted by category
- Top 20 accessed Content sorted by category
- Bottom 10 accessed Content sorted by category
- Bottom 20 accessed Content sorted by category

Once a content selection has been drafted and there is an overall list of content of interest (based on the previously mentioned info and on market statistics) the legal department performs content acquisition and copyright clearance. These are all the preliminary steps; once they have been accomplished and content has been acquired and stored basic adaptation is performed.

3 Introduction to Content Adaptation

AXMEDIS media objects are to be distributed over heterogeneous networks and towards different kind of terminals. Moreover, the people who will ultimately consume and interact with the content may have different behaviour and preferences. Consequently, digital items should be adapted to fit their particular usage environment - this is the goal of AXMEDIS adaptation tools. More specifically, the adaptation tools should be able to modify content to fit:

- Terminal capabilities (codec, formats, input-output, etc supported by the terminal)
- Network characteristics (for example the minimum guaranteed bandwidth of a network)
- User characteristics (presentation preferences, auditory or visual impairment etc)
- Natural environment characteristics (for example the illumination characteristics that may affect the perceived display of visual information)

Mobile devices typically have limited terminal capabilities compared to larger device such as PCs. For example, most mobile phones cannot display video while they are able to play audio files in some specific format. In that context, one should only send audio information to the device. Such adaptations are to be performed on the server side before sending content to the device. Transformations aiming at saving network bandwidth (such as compression or down-sampling) must be performed on the server side as well. Once content received in a form that is supported by the mobile device, a user may wish to modify it according to its own preferences. Such adaptation must be performed directly on the mobile side since different users with different preferences may share the device. Following the same idea, the device should be able to adapt the content to the environment characteristics, which are variable by nature. The seventh part of ISO/IEC 21000 (MPEG-21) specifies tools for the adaptation of Digital Items. More specifically, it proposes a set of normalized tools describing the usage environment of a digital item to command adaptation tools. According to what precedes, AXMEDIS compliant mobile devices should support content adaptation according to a number of MPEG-21 usage environment tools including notably:

User characteristics:

- **AudioPresentationPreferences:** specifies the preferences of a User regarding the presentation or rendering of audio resources. Specifically, descriptions such as the preferred volume, frequency equalizer settings, and audible frequency ranges are specified. Such attributes may affect the way in which the delivered audio resource is encoded, e.g., allocating more bits to specific components in the given frequency range. Additionally, for limited capability devices that may not have equalization functionality, equalization may be performed prior to transmission given the designated preferences.
- **DisplayPresentationPreferences:** specifies a tool for describing the preferences of a User regarding the presentation or rendering of images and videos. This tool includes descriptions of preferences related to the colour and the conversion of stereoscopic video. The `ColorTemperaturePreference`, the `BrightnessPreference`, the `SaturationPreference` and the `ContrastPreference` describe the preferences of a User regarding the colour of the displayed visual contents in terms of colour temperature, brightness, saturation and contrast, each of which is a usual colour attribute of images. `StereoscopicVideoConversion` describes the preferences of a User related to the conversion of 2D video to 3D stereoscopic video and also the conversion of 3D stereoscopic video to 2D video.
- **ColorPreference:** describes preferences related to colour attributes. This sub-clause specifies the `ColorPreferenceType`, which is used as a type definition for the `ColorTemperaturePreference`, the `BrightnessPreference`, the `SaturationPreference` and `ContrastPreference`. Each of these elements that reference `ColorPreferenceType` include a pair of colour attribute values (`PreferredValue`, `ReferenceValue`) that guide how the colour attribute of a displayed image should be controlled so that the perceived colour conforms to the preference of a User. The definition of the attribute value depends on the colour attribute to which the preference description is related.
- **AuditoryImpairment:** describes the characteristics of a particular User's auditory deficiency. The description can be used by the audio resource adaptation engine to optimize the experience of audio contents for the User. The hearing threshold shift of a User is described. The description can be used to compensate the User's auditory impairment such as hearing loss during the adaptation. The tool is meant to be used in case of small hearing loss. For people with more than minimal impairments, gain changes to match the audiogram are normally not sufficient.

- **VisualImpairment:** covers a wide range of conditions. The various forms of visual impairment include difficulty to read the fine print, low vision that cannot be corrected by standard glasses, total blindness, colour vision deficiency, i.e., the inability to recognize certain colours. The low vision conditions, due to their wide variety, are described by the User's symptoms, but the names of conditions are not described.
- **ColorVisionDeficiency:** describes the characteristics of a particular User's colour vision deficiency. The description can be used by adaptation engine to deliver recognizable color image or video data for a User with color vision deficiency.

Natural environment characteristics:

- **AudioEnvironment:** describes the natural audio environment of a particular User in terms of the measured noise level and noise frequency spectrum. These descriptions can be used by audio resource adaptation engine to deliver the best experience of audio contents.
- **IlluminationCharacteristics:** specifies a tool to describe the overall illumination in the environment where a User's display device is located. The tool includes two elements, each of which describes an attribute of the overall illumination. In particular, `TypeOfIllumination` describes the type of illumination and `Illuminance` describes the illuminance of illumination. These attributes are related to the effect of illumination on the perceived colour of a displayed image and can be used to control the colour so as to present the original colour that is intended.

3.1 Content Adaptation for mobile e-applications

What just reported gives a clear understanding of the complexity involved in the adaptation step. Usually by most mobile distributors is preferred to have content ready to deliver (at least for most popular devices) rather than to adapt it on the fly (especially to minimise end-user waiting/on-air time). To this purpose is worth reporting hereafter the main needs and expected possibilities that have to be taken into account in this process that could occur either at the ingestion of new content or at the addition/removal of supported devices.

Action	performed process
Configuration	Definition of adaptation rules for ingestion of a single content
	Definition of adaptation rules for all content to support new devices
	Definition of default adaptation to perform on all content
	Definition of default adaptation to perform on single content
	Definition of content scope for adaptation in case of change in device support
Ingestion of content	Adaptation to all devices
	Adaptation to default devices
Insertion of a new device profile	Adaptation of all content
	Adaptation of content specified in device scope

a) Format support for Windows Mobile 5 PDA based SmartPhones

Text (Where U/A = UNICODE / ASCII)		Image		Audio		Video		Animation	
<i>DOC</i> (U/A)	x	<i>JPEG</i>	x	<i>WAV</i>	x	<i>AVI</i>	x	<i>Flash</i>	x
<i>ODT</i> (U/A)		<i>GIF</i>	x	<i>MP3</i>	x	<i>RealVideo</i>	x	<i>QuickTime</i>	x
<i>TXT</i> (U/A)	x	<i>PNG</i>	x	<i>AIF</i>	x	<i>QuickTime</i>	x	<i>Active Movie</i>	
<i>RTF</i> (U/A)		<i>TIF</i>	x	<i>PCM</i>	x	<i>MPEG 2/4</i>	4	<i>GIF</i>	x
<i>XLS</i> (U/A)	x	<i>TGA</i>		<i>RealAudio</i>	x	<i>Active Movie</i>		<i>QT-VR</i>	x
<i>ODS</i> (U/A)		<i>DIB</i>		<i>AMR</i>	x	<i>Indeo</i>		<i>SVG</i>	x
<i>WAP</i> (U/A)	x	<i>PCX</i>		<i>WMA</i>	x	<i>DVI</i>		<i>PPT</i>	x
<i>HTML</i> (U/A)	x	<i>AI</i>				<i>Cinepak</i>		<i>ODP</i>	(x)
<i>XHTML</i> (U/A)		<i>EPS</i>				<i>FLV</i>		<i>SMIL</i>	x
<i>SGML</i> (U/A)		<i>PSD</i>				<i>WMF</i>	x		
<i>XML</i> (U/A)	x	<i>SWF</i>							
<i>PDF</i>	x	<i>SVG</i>	x						
		<i>BMP</i>	x						

x = Needed, (x) = Optional

DE9.5.4 Integrated Prototype of content Production and Distribution on-demand for Mobile Phones, and new generation of PDAs

b) Format support for non Windows Mobile 5 SmartPhones

Text (Where U/A = UNICODE / ASCII)		Image		Audio		Video		Animation	
<i>DOC</i> (U/A)		<i>JPEG</i>	x	<i>WAV</i>	x	<i>AVI</i>		<i>Flash Light</i>	x
<i>ODT</i> (U/A)		<i>GIF</i>	x	<i>MP3</i>	x	<i>RealVideo</i>	x	<i>QuickTime</i>	
<i>TXT</i> (U/A)	x	<i>PNG</i>	x	<i>AIF</i>	x	<i>QuickTime</i>		<i>Active Movie</i>	
<i>RTF</i> (U/A)		<i>TIF</i>		<i>PCM</i>		<i>MPEG 2/4</i>	MP4	<i>GIF</i>	x
<i>XLS</i> (U/A)		<i>TGA</i>		<i>RealAudio</i>	x	<i>Active Movie</i>		<i>QT-VR</i>	
<i>ODS</i> (U/A)		<i>DIB</i>		<i>AMR</i>	x	<i>Indeo</i>		<i>SVG</i>	
<i>WAP</i> (U/A)	x	<i>PCX</i>		<i>WMA</i>		<i>DVI</i>		<i>PPT</i>	x
<i>HTML</i> (U/A)	x	<i>AI</i>				<i>Cinepak</i>		<i>ODP</i>	
<i>XHTML</i> (U/A)		<i>EPS</i>				<i>FLV</i>		<i>SMIL</i>	(x)
<i>SGML</i> (U/A)		<i>PSD</i>				<i>WMF</i>			
<i>XML</i> (U/A)		<i>SWF</i>							
<i>PDF</i>		<i>SVG</i>							
		<i>BMP</i>							

c) Screen resolutions classification

Small		Regular		Wide		Special	
Resolution	Ratio	Resolution	Ratio	Resolution	Ratio	Resolution	Ratio
160x120	4/3	640x480	4/3	1280x768	5/3	160x160	1/1
320x240	4/3	1024x768	4/3	1280x1024	5/4	200x200	1/1
		1152x864	4/3			240x320	3/4
		1280x960	4/3			300x300	1/1
						320x320	1/1
						320x480	2/3
						600x600	1/1

d) Screen resolutions conversions

Conversions (resize) from 4/3 to 4/3

4/3 -> 4/3	160x120	320x240	640x480	1024x768	1152x864	1280x960	1600x1200	1920x1440	2048x1536
320x240	x	-	-	-	-	-	-	-	-
640x480	x	x	-	-	-	-	-	-	-
1024x768	x	x	x	-	-	-	-	-	-
1152x864	x	x	x	x	-	-	-	-	-
1280x960	x	x	x	x	(x)	-	-	-	-
1600x1200	x	x	x	x	(x)	(x)	-	-	-
1920x1440	x	x	x	x	(x)	(x)	(x)	-	-
2048x1536	x	x	x	x	(x)	(x)	(x)	(x)	-

Conversions (resize) from wide format to wide formats

w -> w	1280x768	1280x1024	1600x900	1920x1080	1600x1024
1280x1024	x	-	-	-	-
1600x900	x	x	-	-	-
1920x1080	x	x	(x)	-	-
1600x1024	x	x	(x)	(x)	-

Conversions (resize) among special sizes

Special	160x160	200x200	240x320	300x300	320x320	320x480	600x600	1920x1200
200x200	x	-	-	-	-	-	-	-
240x320	x	(x)	-	-	-	-	-	-
300x300	x	(x)		-	-	-	-	-

x = Needed, (x) = Optional, - = Not needed

DE9.5.4 Integrated Prototype of content Production and Distribution on-demand for Mobile Phones, and new generation of PDAs

Conversions (resize) among special sizes (continued)

Special	160x160	200x200	240x320	300x300	320x320	320x480	600x600	1920x1200
320x320	x	(x)	x	(x)	-	-	-	-
320x480	x	(x)	x	(x)	x	-	-	-
600x600	x	(x)	x	(x)	x	x	-	-
1920x1200	(x)	(x)	(x)	(x)	(x)	(x)	(x)	-

x = Needed, (x) = Optional, - = Not needed

e) Screen format ratio conversions

Ratio Adaptation (letterbox...)

From	To				
	4/3	5/3	5/4	16/9	25/16
4/3	-	(x)	(x)	x	x
5/3	x	-	(x)	x	x
5/4	x	(x)	-	(x)	(x)

Ratio Adaptation (letterbox...)

From	To				
	4/3	5/3	5/4	16/9	25/16
8/5	x	(x)	(x)	(x)	(x)
16/9	x	(x)	(x)	-	(x)
25/16	x	(x)	(x)	(x)	-

x = Needed, (x) = Optional, - = Not needed

f) Screen format ratio and size conversions from all format to most common ones

Resize and Adaptation (letterbox...)

All - > Common		To								
		4/3							5/3	5/4
	From	160x120	320x240	640x480	1024x768	1152x864	1280x960	1600x1200	1280x768	1280x1024
1/1	200x200	x	-	-	-	-	-	-	-	-
	300x300	x	x	-	-	-	-	-	-	-
	320x320	x	x	-	-	-	-	-	-	-
	600x600	x	x	x	-	-	-	-	-	-
16/9	1600x900	x	x	x	x	(x)	(x)	-	x	x
	1920x1080	x	x	x	x	(x)	(x)	(x)	x	x
2/3	320x480	-	-	-	-	-	-	-	-	-
25/16	1600x1024	x	x	x	x	(x)	(x)	-	-	-
4/3	320x240	x	-	-	-	-	-	-	-	-
	640x480	x	x	-	-	-	-	-	-	-
	1024x768	x	x	x	-	-	-	-	-	-
	1152x864	x	x	x	x	-	-	-	-	-
	1280x960	x	x	x	x	(x)	-	-	-	-
	1600x1200	x	x	x	x	(x)	(x)	-	x	x
	1920x1440	x	x	x	x	(x)	(x)	(x)	x	x
	2048x1536	x	x	x	x	(x)	(x)	(x)	x	x
5/3	1280x768	x	x	x	x	(x)	(x)	(x)	-	-
5/4	1280x1024	x	x	x	x	(x)	(x)	(x)	(x)	-
8/5	1920x1200	x	x	x	x	(x)	(x)	(x)	(x)	(x)

x = Needed, (x) = Optional, - = Not needed

g) Screen format ratio and size conversions from all format to wide ones

Resize and Adaptation (letterbox...)

All - > Wide		To					
		16/9		25/16	5/3	5/4	8/5
From		1600x900	1920x1080	1600x1024	1280x768	1280x1024	1920x1200
16/9	1600x900	-	-	-	x	x	-
	1920x1080	x	-	(x)	x	x	-
25/16	1600x1024	x	-	-	-	-	-
4/3	1600x1200	(x)	(x)	(x)	x	x	-
	1920x1440	(x)	(x)	(x)	x	x	x
	2048x1536	(x)	(x)	(x)	x	x	x
5/4	1280x1024	-	-	-	(x)	-	-
8/5	1920x1200	(x)	(x)	(x)	(x)	(x)	-

x = Needed, (x) = Optional, - = Not needed

h) Screen format ratio and size conversions from all format to special ones

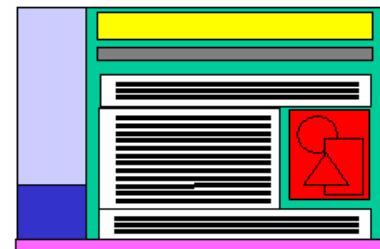
Resize and Adaptation (letterbox...)

All -> Special		To									
		1/1					16/9		2/3	25/16	3/4
From		160x160	200x200	300x300	320x320	600x600	1600x900	1920x1080	320x480	1600x1024	240x320
1/1	200x200	x	-	-	-	-	-	-	-	-	-
	300x300	x	x	-	-	-	-	-	-	-	-
	320x320	x	x	x	-	-	-	-	-	-	-
	600x600	x	x	x	x	-	-	-	x	-	-
16/9	1600x900	x	x	x	x	x	-	-	x	-	x
	1920x1080	x	x	x	x	x	x	-	x	(x)	x
2/3	320x480	x	x	x	x	-	-	-	-	-	-
25/16	1600x1024	x	x	x	x	x	x	-	x	-	x
3/4	240x320	x	x	-	(x)	-	-	-	-	-	-
4/3	320x240	x	x	x	x	-	-	-	-	-	x
	640x480	x	x	x	x	x	-	-	-	-	x
	1024x768	x	x	x	x	x	-	-	-	-	x
	1152x864	x	x	x	x	x	-	-	-	-	x
	1280x960	x	x	x	x	x	-	-	-	-	x
	1600x1200	x	x	x	x	x	(x)	(x)	x	(x)	x
	1920x1440	x	x	x	x	x	(x)	(x)	x	(x)	x
	2048x1536	x	x	x	x	x	(x)	(x)	x	(x)	x
5/3	1280x768	x	x	x	x	x	-	-	x	-	x
5/4	1280x1024	x	x	x	x	x	-	-	x	-	x
8/5	1920x1200	x	x	x	x	x	(x)	(x)	x	(x)	x

x = Needed, (x) = Optional, - = Not needed

3.1.1 Template definition and Format adoption

In the mobile distribution environment is crucial to properly define policies for content formatting either at acquisition or ingestion time as this will greatly simplify and speed up the adaptation process either at ingestion time or in case of post processing due to insertion/modification of supported devices. Therefore defining/adopting templates that reflect the need of the device/moment/customer are also a good way to allow easier adaptation procedures. We have to take into account that each content producer has developed a well-defined company image based on a highly studied combination of page layout, colours, type-fonts, styles... this is usually reflected in the defined templates (just as mentioned before). In some cases the same producers may have different product lines each with a well-defined image reflecting the expectations of the consumers to which it is addressed. It is usual to have products devoted to children (colourful, with big fonts and plenty of drawings...) and others for scholars (very dim colours, tiny and packed fonts...). In both cases there is a "grey" area of superposition that sees content and formats that are generically acceptable to the widest possible public. This is actually the segment we are referring to, as this is the one best suitable at the moment for automatic composition and adaptation of content. For the time being we assume to consider web-based content that should be made available to mobiles. Therefore a typical content page holds a set of content comprising: title, sub-title (if needed), body text (resizable and scrollable in respect to display area), some multimedia content (with related controls when needed), a quick navigation bar (if needed to allow content browsing) and potentially comprising or associated with a control bar (to allow overall content management: print, save...) and a footer usually holding copyright information. The usual structure is the one in the picture aside. What just mentioned could be synthetically expressed by the following formula:



- » Title
- » Sub-title
- » A body text
- » Multimedia content
- » Quick navigation bar
- » Control bar
- » Content area
- » Footer

$$\exists ! \text{ Title} + \{ \text{sub-title} \} + \{ \text{body text} + \{ \text{scroll bar} \} \} + \{ \text{multimedia content} + \{ \{ \text{control bar} \} \} \} + \{ \text{quick navigation menu} + \{ \text{control bar} \} \} + \{ \text{control bar} \} + \text{Footer}$$

3.1.2 Content combination and adaptation

There are several approaches for managing the combination of possibilities coming from the combination of user-preferences and delivery-device-constraints. Usually the process is based on either differentiated

content production or on selection process based on some way to express user preferences and device capabilities (usually via set of keywords).

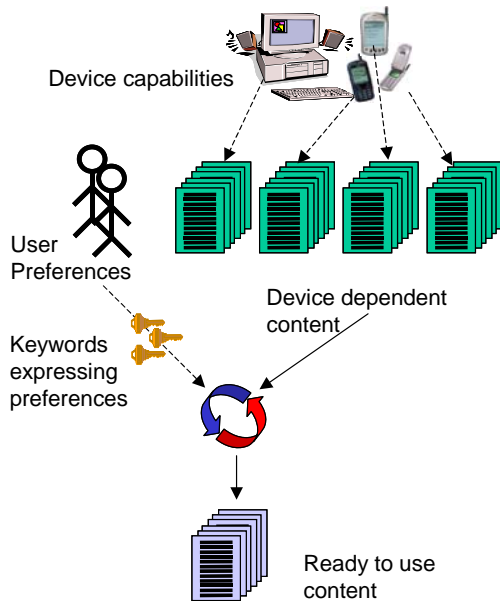


Figure 6. Device dependent content production.

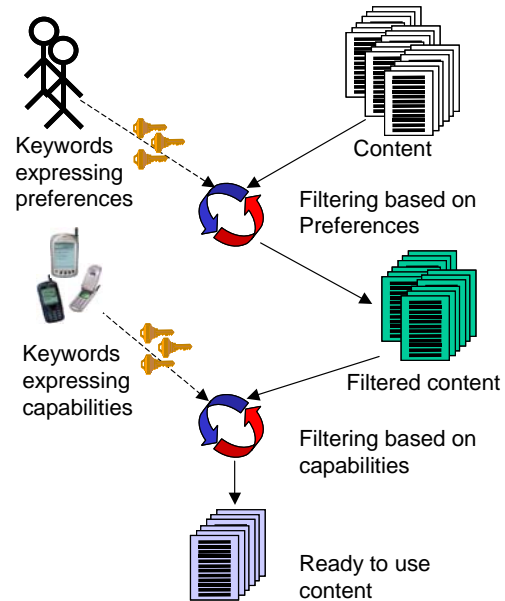


Figure 7. Two step filtering.

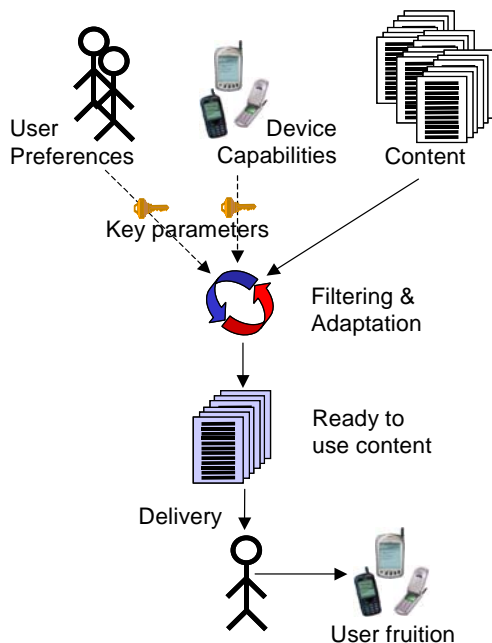


Figure 8. The ideal process.

Publishers often prefer to have device dependent version of content to simply perform preference related adaptation to content as all is demanded at production level and the user simply has to select the desired format for the chosen content. This approach is certainly more time consuming and costly, yet the most diffused. In this case device dependent content is prepared and stored in a structure reflecting device dependencies, this latter stage is made for making it easier to the user to proper access to the desired content (wap ...) as usually such services are on a “pay per use” basis and the customers have signed a specific contract comprising an agreement on the quality of service (QoS). Then device dependent content is selected matching user preferences. It is worth mentioning that usually the content available to the user following this approach is less than the one possibly available in the overall.

An alternative solution adopted is the one based on adaptation via filtering. In essence the process changes slightly from the just mentioned one as in this case both user preferences and device dependencies are used as keywords for performing a two consecutive filtering steps as described in the picture aside.

Actually, if possible, it would be far better if the ready to use content could be derived directly from the original content simply via filtering (on the basis of user preferences) and adapting (on the basis of device dependencies). This is actually one of the scopes of the project and of the present demonstrator. The last diagram, where the same terms of the previous ones are used for consistency, could represent the desired process. In such an approach is foreseen a single step where are combined filtering and adaptation. To make this possible is essential to examine the issues that are at the back of the device dependencies and that have

direct impacts on the content production process. We have already said that typically content is developed “ad hoc” for each device. This was, and often still is, basically due to a fact: *it is far easier to manually optimize the rendering aspect of content rather than making it generically adaptable via a programming approach*. This stems from the possibility to fine tune a content that has a high degree of probability to remain stable on a reasonably long time span (basically comparable with the object expected life time), this makes it more convenient to pursue the just described approach.

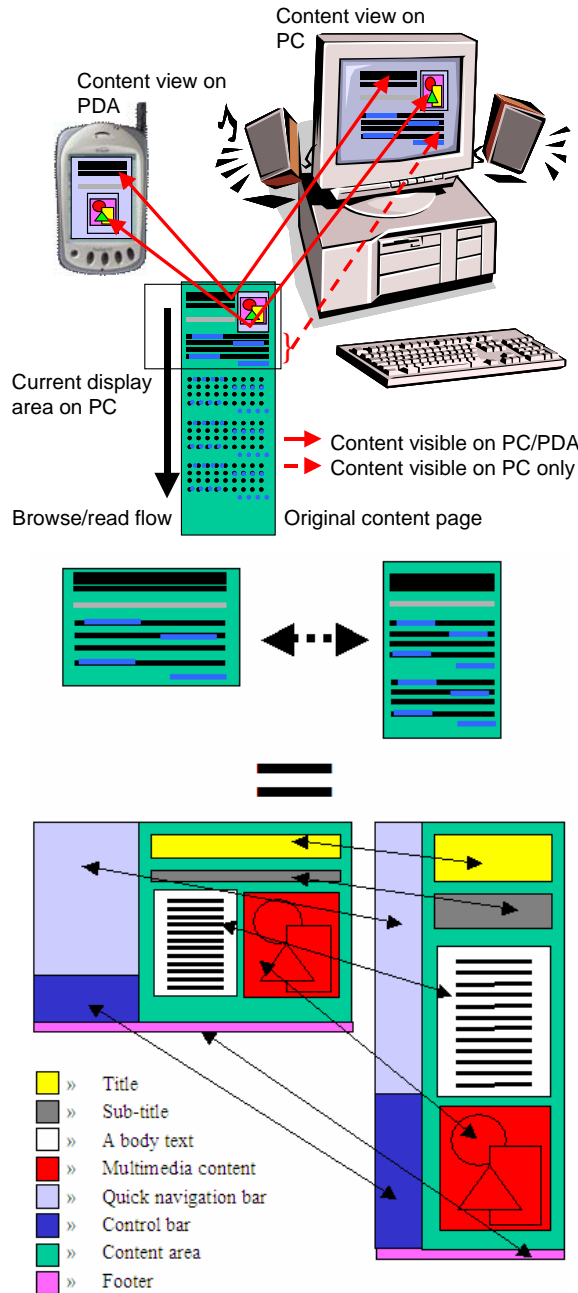


Figure 9. Page adaptation in page conversion.

The most frequent and essential change in content rendering is the orientation. Unless the PDA is able to automatically flip the screen (that is presenting data in landscape rather than portrait) such a transformation has to be done prior to delivery. Now this is not the only point as passing from PCs to PDA, 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

What just stated implies that a page that has been designed for fruition on a PC when presented onto a PDA may undergo a set of changes.

The first, and probably the more apparent is that content is highly probable to be shrunk to a smaller dimension, as even the simplest of all apparent changes occurring (converting from landscape to portrait or vice versa) implies a change in:

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

This latter point may sound strange, but given the fact that the most diffused browser for Windows CE / Windows Mobile equipped PDA does not support CSS files a formatted HTML page will lose most of what foreseen at content design and production time.

This, in some contexts, where content quality or publisher / distributor branding is highly related to content quality may be a major limiting factor that may lead to the voluntary production of device customized content production.

Moreover at rendering time (either at server or client side) of a content has to be taken into account that passing from a single page to a set of pages, links between portions of the same page (anchors) are turned into links between pages (URLs), including the return to “top of page” as the way content is browsed changes from:

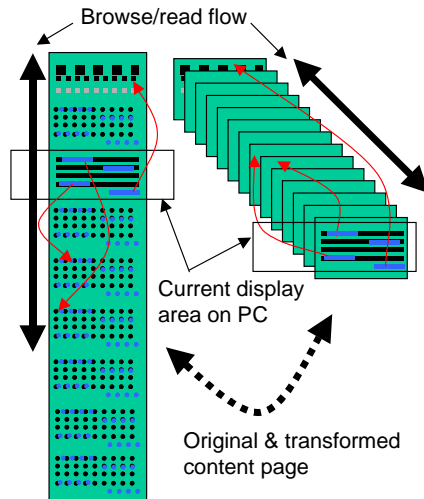


Figure 10. Navigation structure change due to adaptation.

- Content adaptation and eventually
- Content distribution.

Such a schematic approach could be better detailed as follows taking into account that each set of defined rules has a specific purpose and should be used consistently. The approach is based on what presently developed for handling IMS/SCORM/AICC content delivery in LEX.

Original content belonging to different locations (D_x) is processed with the support of a set of descriptive rules $\{R_d\}$ that enable deriving an “enriched” content holding info about single content components typology and characteristics as simplified in the example reported here where the extended description format can therefore look somehow similar to what follows depending on the set of extended information provided⁸.

```
<COMPONENT type= title LANGUAGE=en ...>
  <ID=1> (insert the XML coding for additional info on object ID) </ID>
  <CONTENT=...> (insert the XML coding for the content) </CONTENT>
  <POSITION=...> (insert the XML coding for the position) </POSITION>
  <ORIENTATION=90dl ...> (insert the XML coding for additional orientation info) </ORIENTATION>
  ...
  (insert the XML coding for additional descriptive and format related info)
  ...
</COMPONENT>
```

Having done so the new content is suitable for storage into the AXMEDIS database (A) from where it can be extracted for composition and aggregation. Such operation will be supported by a set of composition rules $\{R_c\}$ that can benefit from the “enriched” description format reported before. Is given for granted that newly produced, aggregated or composed content will be stored back in the AXMEDIS databases either as a new version of the same object or as a new object depending on the kind of operation performed onto it. A similar approach applies for subsequent formatting and adaptation steps where the $\{R_f\}$ and $\{R_a\}$ sets of rules will be respectively used to achieve the desired result. The last step is the distribution one where a last set of rules

- Scrolling to
- Paging

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

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

- Content component identification and description,
- Content composition,
- Content formatting,

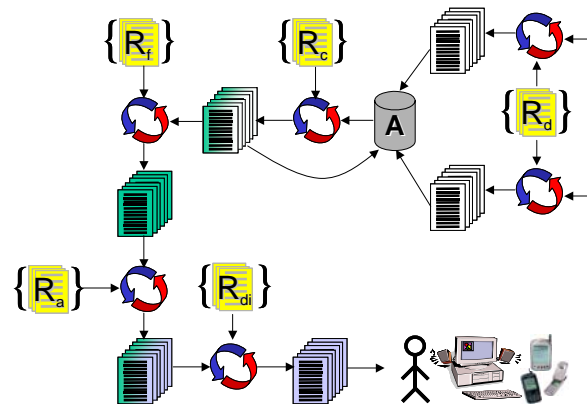


Figure 11. The mobile content processing operation schema.

⁸ The final grammar represents the work in hand
AXMEDIS Project

{R_{di}} will be used to apply constraints, limitation or other peculiarities needed to properly handle the process. As previously said this is based on what presently achieved in LEX to handle a similar procedure related, so far only, to learning objects. The following is an example of XML code representing what presently possible and implemented in our environment following IMS standard and that will therefore represent for us the starting point for future developments in the field.

```
<?xml version="1.0" encoding="utf-8"?>
<!-- XML file generated by eXact Packager -->
<!-- eXact Packager licensed to: monbile2.giuntlabs.com -->
<LOMobile xp:deliverytype="SCORM" xlink:label="L5414" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xp="http://www.giuntlabs.com/exact/xp_v1d0" href="LOMobile.html" xp:description="L06_Madonna_pomegranate" Tracking="AICC-
HACP" id="L06_Madonna_pomegranate">
  <style xmlns="http://www.giuntlabs.com/exact/xp_v1d0">
    <colors c1="#000000" c2="#FFFFFF" c3="#D4D4D6" c4="#005FAE" c5="#003060" c6="#01D4FA"/>
    <linkcolors normal="#0000FF" visited="#800080" active="#FF0000"/>
    <tracking>AICC-HACP</tracking>
    <pagesize width="760" height="640" type="1"/>
    <textstyles>
      <textstyle type="Title">
        <normal fontface="Arial" fontsize="28" color="#000000" bold="false" italic="false" underline="false"/>
        <emph fontface="Arial" fontsize="28" color="#000000" bold="false" italic="true" underline="false"/>
        <moreemph fontface="Arial" fontsize="28" color="#000000" bold="true" italic="false" underline="false"/>
      </textstyle>
      <textstyle type="Subtitle">
        <normal fontface="Arial" fontsize="24" color="#000000" bold="false" italic="false" underline="false"/>
        <emph fontface="Arial" fontsize="24" color="#000000" bold="false" italic="true" underline="false"/>
        <moreemph fontface="Arial" fontsize="24" color="#000000" bold="true" italic="false" underline="false"/>
      </textstyle>
      <textstyle type="Subsubtitle">
        <normal fontface="Arial" fontsize="20" color="#000000" bold="false" italic="false" underline="false"/>
        <emph fontface="Arial" fontsize="20" color="#000000" bold="false" italic="true" underline="false"/>
        <moreemph fontface="Arial" fontsize="20" color="#000000" bold="true" italic="false" underline="false"/>
      </textstyle>
      <textstyle type="TextIntro">
        <normal fontface="Arial" fontsize="10" color="#000000" bold="false" italic="false" underline="false"/>
        <emph fontface="Arial" fontsize="10" color="#000000" bold="false" italic="true" underline="false"/>
        <moreemph fontface="Arial" fontsize="10" color="#000000" bold="true" italic="false" underline="false"/>
      </textstyle>
      <textstyle type="TextBody">
        <normal fontface="Arial" fontsize="12" color="#000000" bold="false" italic="false" underline="false"/>
        <emph fontface="Arial" fontsize="12" color="#000000" bold="false" italic="true" underline="false"/>
        <moreemph fontface="Arial" fontsize="12" color="#000000" bold="true" italic="false" underline="false"/>
      </textstyle>
      <textstyle type="TextEnd">
        <normal fontface="Arial" fontsize="10" color="#000000" bold="false" italic="false" underline="false"/>
        <emph fontface="Arial" fontsize="10" color="#000000" bold="false" italic="true" underline="false"/>
        <moreemph fontface="Arial" fontsize="10" color="#000000" bold="true" italic="false" underline="false"/>
      </textstyle>
      <textstyle type="References">
        <normal fontface="Arial" fontsize="8" color="#000000" bold="false" italic="false" underline="false"/>
        <emph fontface="Arial" fontsize="8" color="#000000" bold="false" italic="true" underline="false"/>
        <moreemph fontface="Arial" fontsize="8" color="#000000" bold="true" italic="false" underline="false"/>
      </textstyle>
      <textstyle type="Fig-caption">
        <normal fontface="Arial" fontsize="8" color="#000000" bold="false" italic="false" underline="false"/>
        <emph fontface="Arial" fontsize="8" color="#000000" bold="false" italic="true" underline="false"/>
        <moreemph fontface="Arial" fontsize="8" color="#000000" bold="true" italic="false" underline="false"/>
      </textstyle>
      <textstyle type="Footnote">
        <normal fontface="Arial" fontsize="8" color="#000000" bold="false" italic="false" underline="false"/>
        <emph fontface="Arial" fontsize="8" color="#000000" bold="false" italic="true" underline="false"/>
        <moreemph fontface="Arial" fontsize="8" color="#000000" bold="true" italic="false" underline="false"/>
      </textstyle>
      <textstyle type="Button Text">
        <normal fontface="Arial" fontsize="8" color="#000000" bold="false" italic="false" underline="false"/>
        <emph fontface="Arial" fontsize="8" color="#000000" bold="false" italic="true" underline="false"/>
        <moreemph fontface="Arial" fontsize="8" color="#000000" bold="true" italic="false" underline="false"/>
      </textstyle>
    </textstyles>
  </style>
  <Page xp:deliverytype="SCORM" xlink:label="L5415" xp:description="S1" id="Page01">
    <SubPages xp:deliverytype="SCORM" xlink:label="L5416" xp:description="Introduction" kind="R">
```

```
<SubPage xp:deliverytype="SCORM" xlink:label="L5417" xp:description="Introduction_text">
  <ElementGroup xp:deliverytype="SCORM" xlink:label="L5418" xp:description="introduction_text_en" language="en">
    <ElementGroup xp:deliverytype="SCORM" xlink:label="L5419" xp:description="Introduction_text_a" type="Tourist">
      <Text xlink:label="L5420">Welcome. In this section you will discover one of Botticelli most beautiful masterpieces
shown in the Galleria degli Uffizi, the painting Madonna of the Pomegranate.</Text>
      <audio src="L06_Madonna_Pomegranate_high/L06_Madonna_Pomegranate_high_Page01_en_tourist.wav"/>
    </ElementGroup>
    <ElementGroup xp:deliverytype="SCORM" xlink:label="L5421" xp:description="Introduction_text_b" type="Student">
      <Text xlink:label="L5422">Welcome. In this section you will discover one of Botticelli most beautiful masterpieces
shown in the Galleria degli Uffizi, the painting Madonna of the Pomegranate. Though it you will learn more about the painting.</Text>
      <audio src="L06_Madonna_Pomegranate_high/L06_Madonna_Pomegranate_high_Page01_en_student.wav"/>
    </ElementGroup>
    <ElementGroup xp:deliverytype="SCORM" xlink:label="L5423" xp:description="Introduction_text_c" type="Teacher">
      <Text xlink:label="L5424">Welcome. In this section you will discover one of Botticelli most beautiful masterpieces
shown in the Galleria degli Uffizi, the painting Madonna of the Pomegranate. Though it you will learn more about the painting and its mythological,
philosophical and literary background.</Text>
      <audio src="L06_Madonna_Pomegranate_high/L06_Madonna_Pomegranate_high_Page01_en_teacher.wav"/>
    </ElementGroup>
  </ElementGroup>
  <ElementGroup xp:deliverytype="SCORM" xlink:label="L5425" xp:description="introduction_text_it" language="it">
    <ElementGroup xp:deliverytype="SCORM" xlink:label="L5426" xp:description="Introduction_text_a" type="Tourist">
      <Text xlink:label="L5427">Benvenuti. In questa sezione scoprirete uno dei capolavori del Botticelli presente nella Galleria
degli Uffizi la Madonna della melagrana.<br/>
      <Text>
        <audio src="L06_Madonna_Pomegranate_high/L06_Madonna_Pomegranate_high_Page01_it_tourist.wav"/>
      </Text>
    </ElementGroup>
    <ElementGroup xp:deliverytype="SCORM" xlink:label="L5428" xp:description="Introduction_text_b" type="Student">
      <Text xlink:label="L5429">Benvenuti. In questa sezione scoprirete uno dei capolavori del Botticelli presente nella Galleria
degli Uffizi la Madonna della melagrana.<b> </b>Potrete scoprire i segreti del dipinto.<br/>
      <Text>
        <audio src="L06_Madonna_Pomegranate_high/L06_Madonna_Pomegranate_high_Page01_it_student.wav"/>
      </Text>
    </ElementGroup>
    <ElementGroup xp:deliverytype="SCORM" xlink:label="L5430" xp:description="Introduction_text_c" type="Teacher">
      <Text xlink:label="L5431">Benvenuti. In questa sezione scoprirete uno dei capolavori del Botticelli presente nella Galleria
degli Uffizi la Madonna della melagrana.<b> </b>Potrete scoprire i segreti del dipinto e i suoi legami con la storia, la filosofia e la letteratura del
periodo.<br/>
      <Text>
        <audio src="L06_Madonna_Pomegranate_high/L06_Madonna_Pomegranate_high_Page01_it_teacher.wav"/>
      </Text>
    </ElementGroup>
  </ElementGroup>
</SubPage>
</SubPages>
</Page>
</LOMobile>
<!-- End of XML file generated by eXact Packager -->
```

3.1.3 Metadata and tagging

This process is extremely important as the whole management infrastructure for the Mobile Application is strictly dependent on the availability and quality of the information that accompany the object, regardless of the kind. Basically it would be necessary to fill in the Mandatory and part of the Optional metadata described in the LOM standard if all was limited to our environment. Given the fact that AXMEDIS has its own metadata it is necessary to see where superposition allows automatic replication/transferring of data and where it will be necessary to specifically insert relevant metadata. In practice we have:

- **Data common between AXInfo and LOM** – this data will be possibly automatically filled in at creation time.
- **Data common between Dublin Core and LOM** – this data will be possibly automatically filled in at creation time.
- **Data specific to AXInfo** – this data will be filled in at authoring time.
- **Data specific to LOM** – this data should have been possibly filled in at authoring time.
- **Data specific to Dublin Core** – this data will be possibly filled in at authoring time for classification and management purposes.

There are two types of element subsets defined here: the elements that should be filled in every metadata instance (mandatory elements) and the elements that would be very useful to be filled (recommended elements). Both sets have been extensively described in other document that are reported in the reference section and therefore will not be further described here.

4 Info on supportive technologies integrated

4.1 Media Distribution Management Requirements

This requires the following sets of operations based on the delivery context information

1. **Target Client Device and User ID management**
2. **Profile Data Management (profiles collection, storage, access and/or update)**
3. **Delivery Context Information**
4. **Dynamic Rights Audit Management Information**
5. **Personalisation (Profiling Resolution)**
6. **Media Selection & Bundling**
7. **Media Transcoding**
8. **Media Adaptation**

The requirements for each of the above are set out as follows:

4.1.1 Target Client Device and User ID Management

In order to ensure that rights are appropriately audited and paid for by the correct user for the correct client device belonging to that user, the Certifier-Supervisor will be required to specify, store and update the following information to uniquely identify a customer and his client device to which the media is delivered.

Currently the AXCS stores some information regarding the device for certified tools. That is, when a new client tool is certified in AXCS, a new entry is created for that tool. That entry contains a fingerprint of the tool which contains information regarding the tool as well as the device where it is installed.

4.1.2 Profiling management (profiles collection, storage, access and/or update)

For each subset of the profile we need to provide: the profiling “maintenance overhead” for the on-demand, or other scenarios as follows:

- i. (near)real-time (synchronous and asynchronous, direct/indirect) means of collection of each profile element
- ii. Storage of each profile element
- iii. Access to /Dynamic update of each profile element
- iv. Resolution of the profile element

The above must include the address of default value holders, and, resolution or catch-all rules (“graceful closure” heuristics) for each profile element so that the personalisation facility can deal with anomalous situations for example even trivial anomalies such as unexpected null sets or empties that may be encountered in a particular customer’s profile set etc. Thus user profiling can be known or inferred from the specific profile elements or overall user profile which should include some or all of the elements below as may be implemented commensurate with the workpackage scope and resources within the AXMEDIS Project. The Composite Delivery Context Profile Management (CDCPM) requires the resolution of at least three distinct sets or bags of profile each consisting of several profile elements. The three sets comprise the User’s Personal Profile, User’s Device(s) Profile(s) and Delivery Context Information with the requirements for each of the constituencies of the CDCPM as set out below :

A) User’s Personal Profile - User’s personal profile can consist of the elements, each defined with respect to their required maintenance entailments (collection, storage, access, update) as follows:

Name:

Collection: must take place during user registration with the personalisation server to open a media delivery or other service account.

Storage: must be stored in the personalisation server profiling knowledge repository (user profiles database and/or customer accounts database linked to of the PMS-Database).

Access: accessible to the user, system and the Administrator.

Update: once the registration process with the personalisation serve is complete, user's name can no longer remain change-able by the user for the same account; can be updated by the Administrator if required.

Device Identities:

Collection: can take place during user registration with the personalisation server to open a media delivery or other service account. Specific unique identifiers can be targeted for evaluation for user and device.

Storage: must be stored in the personalisation server profiling knowledge repository (user profiles database or a customer accounts database).

Access: accessible to the user, system and the Administrator.

Update: once the registration process with the personalisation serve is complete, device identities can no longer remain change-able by the user for the same account; can be updated by the Administrator if required.

Number of Devices Used

Collection: can take place during user registration with the personalisation server to open a media delivery or other service account; this can be evaluated directly or indirectly from the user.

Storage: must be stored in the personalisation server profiling knowledge repository (user profiles database or customer accounts database).

Access: accessible to the user and system

Update: once the registration process with the personalisation serve is complete, the number of devices in use by any user will be inferred and updated by the system.

Gender (Title)

Collection: can optionally take place during user registration with the personalisation server to open a media delivery or other service account, it can be entered either directly or indirectly or inferred from the users' personal Title e.g. Mr, Miss or Mrs etc.

Storage: must be stored in the personalisation server profiling knowledge repository (user profiles database or customer accounts database).

Access: accessible to the user and system

Update: once the registration process with the personalisation serve is complete, user's Gender can no longer be changed by the user for the same account; can be updated by the Administrator thereafter if ever required.

Age /Date-of-Birth

Collection: must take place during user registration with the personalisation server to open a media delivery or other service account.

Storage: must be stored in the personalisation server profiling knowledge repository (user profiles database or customer accounts database)

Access: accessible to the user and system.

Update: once the registration process with the personalisation serve is complete, user's name can no longer be change-able by the user for the same account; can be corrected by the Administrator thereafter if ever required.

User's Preferences (Desired Media Types)

Collection: can take place during user registration or later interaction with the personalisation server to open a media delivery or other service account. Collection can be effected directly or indirectly, via Q&A/menu.

Storage: must be stored in the personalisation server profiling knowledge repository (user profiles database or customer accounts database).

Access: accessible to the user and system.

Update: update-able or appendable by the user and or system directly or indirectly

B) User's Device Profile

The CC/PP or increasingly now UAProf are used to give the capabilities and preferences of user agents, specifically that of the client device and their browsers. This would require an XML/RDF-based encoding of the device capabilities as well as a Deli-like profiles resolution package to dynamically structure and resolve the profile elements. The mobile device profiles can be cached or dynamically collected using UAProf and/or whichever other collection source will become more dominant in the future depending on the convergence to particular emerging standards for Dynamic Media Adaptation (e.g. as per MPEG21 recommendations) or other Open sources (e.g. WURFL). For distribution to mobile devices, the relevant device profile elements can be evaluated using the following list of 22 device characteristics as follows. It must be noted that although all the following elements can be interrogated by CC/PP-based description as may be resolved by a Deli-like profile resolution package; not all of the following elements of device characteristics as may have been specified by the manufacturer are made available in the same way, or operationally amount to the same play quality as experienced by the user. So that some elements of device characteristics for some devices may remain dynamically unavailable and thus a core of essential device profile elements should be selected for device profiling as well as appropriate resolution rules included to ensure graceful closure of the dynamic device profiles even if some device profile elements may happen to be dynamically unavailable/unknown.

Mobile Device Profile Elements

- | | |
|-----------------------------|--------------------|
| a. Screen Type | l. Interface |
| b. Colour Screen | m. Infrared |
| c. Colours | n. Bluetooth |
| d. Screen size | o. WLAN |
| e. Screen resolution | p. Battery Type |
| f. Memory Size | q. Battery Time |
| g. ROM Size | r. Weight |
| h. CPU Speed | s. Built-in Camera |
| i. Operating System version | t. Width |
| j. MP3 Player | u. Height |
| k. Voice Recording | v. Thickness |

4.1.3 Delivery Context Information

This profile set can consist of the elements as set out below; each defined with respect to their required maintenance overheads (collection, storage, access, and update).

Location (Country of residence to which the user's delivery device is registered)

Location information is significant since many media distributors adopt marketing policies and accordingly licensing regimes which are generally zone/country-specific and stipulate specific rights packages or media bundling/special-offers/sales marketing/promotion campaigns and licensing regimes available only to users resident in certain countries during certain time windows.

Collection: must take place during user registration with the personalisation serve to open a media delivery or other service account.

Storage: must be stored in the personalisation server profiling knowledge repository (user profiles database or customer accounts database)

Access: accessible to the user and system

Update: can be updated or appended by the user or system directly or indirectly

User's currently requested media if known specifically

Collection: This is expected to be included in the user's Content Request which may either specify exactly the Media Product_ID/ Special offer bundle_ID that the user would wish to purchase (e.g. by

the user highlighting some entries in a Recommendation List) or the user may have been less specific and simply asked for/ticked a particular type/genre of media or even “music, entertainment”. In practice menu-led prompting as e.g. by hierarchical menus will allow users to identify their interests at some level.

Storage: must be stored in the personalisation server as a Content Request listing to be first resolved if non-specific (e.g. by substituting any non-specific user initiated requested items with user-profile-matched default values from the current offers/etc personalised Recommendation List) and then passing the resulting specific Content Request onto the Certifier-Supervisor for licensing status/conditions query (i.e. purchase/clearance) and transaction recording and settlement before any dispatch can take place.

Access: accessible to the user and system

Update: can be updated or appended by the user or system directly or indirectly

User’s previously requested media (including unsuccessful requests)

Collection: must take place by the Certifier-Supervisor Engine recording the detail of any licenses as they are purchased by each customer such that the full identity and media classification particulars (metadata information) constituting the User’s Content Requests History (UCRH) can be made available to the personalisation server at any time - including at least Media type and Genre (e.g. entertainment(audio(music(garage; mobile-media(audio(music(ringtone(classic; Educational(video(natural-science(bird-migration.....where (denotes object hierarchies) Thus the UCRH should include:

- i. The details of media rights ever purchased by any user (whether still unspent or already fully spent) and their un-used/remaining rights
- ii. The unsuccessful requests which constitute useful additional information re user’s interests and although not relevant for the purpose of keeping audits on those user’s rights purchased/used audit, they should ideally be recorded somewhere in the Certifier-Supervisor linked to the customer accounts database; such that they can be queried with reference to a user name_ID or in general.

Storage: must be stored in the customer accounts database or other linked to the PMS-Database of the Certifier-Supervisor indexable through Query Support

Access: accessible to the user(s), system and the Administrator

Update: can be updated or appended by the system

User’s Differentiated Service/License Terms and/or Market Segmentation Classification (if maintained by the distributor – optional)

Collection: this is an optional additional element of the Delivery Context that can be useful for personalisation but which is best explicitly collectable from the customer accounts database linked to Certifier-Supervisor or it could be implicitly maintained and inferred using customer classification rules executed by the personalisation server on the basis of querying customer’s history of media purchases and content requests to-date as may need be queried and considered to allow a potentially better matched Recommendation List to be compiled and presented to the user via the user interface.

Storage: must be stored in the customer accounts database or other linked to the PMS-Database of the Certifier-Supervisor indexable through Query Support

Access: accessible to the system and the Administrator

Update: can be updated by the system or the Administrator

Zone/market/customer- category-specific promotional offers (bundles) currently available

Collection: This is that pre-compiled broad classification of the user population on the basis of criteria such as:

- i. User’s Content Request History (UCRH) i.e. previous demand pattern
- ii. User’s explicitly stated media types of interest i.e. user’s preferred media type
- iii. User’s Age, per-annum amount spent, post-code or other factors (e.g. inferred social class)
- iv. Zonal Market segmentation-based sales promotion campaigns

This information is either computed by the system dynamically or set periodically by the system or the Administrator.

Storage: must be stored in the customer accounts database or other linked to the PMS-Database of the Certifier-Supervisor indexable through Query Support

Access: accessible to the system and the Administrator

Update: can be updated by the system and the Administrator directly or indirectly

Current popularity chart for classified media or Top-10% and Bottom-10% of popularity ratings for media-on-demand

Collection: This is computed periodically (e.g. classified weekly charts) on the basis of demand pattern and volume for each media type or media type or menu item (or other appropriate content indexing).

Storage: must be stored in the personalisation server product profiling knowledge repository

Access: accessible to the system or the Administrator

Update: can be updated by the system

4.1.4 Dynamic Rights Audit Management Information

For the distribution of any object the trigger for the transcoding process is required to be at several levels:

- i. Media file and its type
- ii. Acquired adaptation license that allows the distributor to adapt the content for the final user.

The experimental scenario envisages users being able to download various learning and edutainment content types involving on-demand distribution of composite content. It is clear that the composite content would require multiple DRM-audits and transcoding jobs. In the context of the Demonstrators scenario, the multiple-licenses-checked resolution of the required DRM permissions would require combined rights resolution rules within the certifier-supervisor to be computed before any transcoding can take place as the transcoding engine will need to receive a go-nogo signal re target conversions to be performed on each component of the composite media (this is expected to be computed within the Certifier-Supervisor based on queries put to the PMS-DB, via Query Support on the extent of the remaining granted rights for the user re various components of the target composite content).

4.1.5 Personalisation (profiling resolution and inference)

This process comprises taking the various elements of the user's personal and user's device profiles as input either by reading cached profiles, or querying/computing them dynamically and evaluate their personal value, accessing/computing a set default value or otherwise bringing to a conclusive closure the computation of all personal profile elements in any profile set so that the profile values are then actionable i.e. ready to be used by the product selection or Recommender Rules. The above process is referred to as full profile resolution and involves the following stages:

- i. **Computing the profile element personal value:** this establishes the actual individualised value of a personal profile element by using the value previously input by the user at the registration time or Content Request Time or computing the value on the basis of one or several previous user inputs or other user-specific information as may be inferred directly or indirectly by the personalisation rules.
- ii. **Default evaluation:** If none or insufficient information is available to compute the value of a profile element on a specific personal basis or when it is deemed inappropriate or unnecessary either to seek a value from the user or compute it on an individual basis because it is possible to establish default value(s) for the profile element on a customer group-differentiation or classificatory basis then such default values are set by the system or the Administrator using the appropriate rules/heuristics on the basis of social-group-based demand pattern criteria.
- iii. **Resolve-infer-getclosure(catch-all):** This is the process of finally arriving at a closure re establishing a value for a profile element such that it can be used for personalised Recommendation List Compilation i.e. selection and supply of suitable content as may be suggested to the user on-demand. This uses the following stages:

a) Resolve

This gets the profile element (by look-up/query) and applies any updating rules necessary to ensure the integrity of the value(s) for that profile element as will be used by the personalisation rules; the updating distinguishes between three types of profile element on the basis of the admissibility logic of their variability, uniqueness and/or permanency and security/privacy regarding whether or not they can be amended or appended as follows:

- i. **Locked profile element:** this is the type of profile element that has a unique and permanent value for each user and can not be possibly amended or updated dynamically and stays as a permanent value for the particular user's profile e.g. Name or Date-of-Birth.
- ii. **Over-ride-able Profile Element:** this is a profile element that has a unique value that can change from time to time and can thus be dynamically over-written i.e. the latest (inline) value is compared with the pre-existing value in the profile set and if the two are different then the pre-existing value is replaced by a new value either on a one-time substitution basis or to establish a new value.
- iii. **Appendable Profile Element:** this is the type of profile element that can not necessarily have a unique value i.e. may have multiple values that have to be taken into account serially for successive selection and supply of products/services at any time e.g. selling (playing/sending on-demand) both the English and the Italian version of the same song one after another etc i.e. this profile element can take a list of values. Therefore for appendable profile elements the rule is that the latest (inline) value is compared with the pre-existing value in the profile set and if the two are different then there exists a new addition to the list of values that that profile element can take (sequentially or alternatively as may be inferable from the user's Content Request (e.g. both or only the first/second) and/or the delivery Context Logic(e.g. always both). So the new value is appended to the list of values for the profile element.

b) infer-getclosure(catch-all): Once all the profile elements, whose latest resolved value is essential for computing an ordered selection (menu) of products to be offered to the customer, have been subjected to a resolution attempt, if any profile element has failed to be resolved i.e. the resolution look-up or inline-value check-returns a null (empty/unavailable/unknown) result then appropriate rules are used to select a default value once this has been evaluated and if a default value can not be usefully computed then an appropriate body of rules are executed to either achieve a one-time heuristic value or determine what is the best guess (best possible match on the basis or poor evidence) to add to the Recommendation List, or requesting the user to supply the missing information (i.e. insert a rule to plan to ask the user directly or indirectly online). The above personalisation processes are now visited for each of the personal and delivery context profile elements. The user's device profile elements are generally also similarly resolved by a Deli-like package.

A) User's Personal Profile Elements

Name:

Computing this profile element: this is a locked element; use it to query the relevant user profile elements from the personalisation server and the relevant delivery context elements from the PMS-Database via Query Support

Default evaluation: no default value needed to be maintained for this element.

Resolve-infer-getclosure (catch-all): assumed always to be readily available and resolvable

Device Identities:

Computing this profile element: this is a locked element, use it to query the relevant user device and rights usage profile elements from the PMS-Database via Query Support.

Default evaluation: no default value needed to be maintained for this element.

Resolve-infer-getclosure (catch-all): assumed always to be readily available and resolvable.

Number of Devices Used

Computing this profile element: this is an overwrite-able element; it is evaluated as the number of entries in the User's Devices list. It may be used as one of the inputs to user classification rules for the purpose of matching possible media type or promotional offers/bundles and DRM control re device/user-specific "infinite repetition" rights.

Default evaluation: default value is always 1.

Resolve-infer-getclosure (catch-all): assumed always to be readily resolvable

Gender (Title):

Computing this profile element: This is generally a locked element; it can be available directly from previous user input or inferred from user's title.

Default evaluation: If ever both the Gender field and the Title field are found to be empty/unknown for any user or if Gender is unknown or it was not to be explicitly queried for ethical sensitivity reasons then if in such cases the Title is either unknown or a gender-neutral title such as Dr or Pr then the default gender rule would be activated to set the value for this element to a code (0 or 1, c for common or gn for gender-neutral) to activate gender-neutral rather than gender-specific profile element (default) value to be integrated with the other elements for the purpose of product recommendation to the user.

Resolve-infer-getclosure (catch-all): Resolve this element if readily available or use default value as above so as to use a gender-invariant value as the closure value.

Age (Date of Birth)

Computing this profile element: this is a locked element; obviously age is computed from Date-of-Birth. The value of age may be used as one of the inputs to user personalisation rules for the purpose of matching possible media type/genre or promotional offers/bundles as suitable recommended products for the user.

Default evaluation: This is often a mandatory user input as supplementary identification information so it is very rare for the registration process to run to completion without this information having been supplied by the user. If ever this element is unknown then it can more often than not be set to a default value of "young-ish".

Resolve-infer-getclosure (catch-all): Resolve this element if readily available; should it ever be found to be unknown, set it to the default value of youngish as closure value.

User's Preferences (Desired Media Types)

Computing this profile element: This is an unlocked element; the user can be prompted to enter as specific as possible his/her product preferences within the Content of the Request or at the user initial registration time the preferences can be elicited hierarchically i.e. categorically in terms of media types and genre within each type as applicable and/or a specific product IDs. Such preferences can at any time be queried from the customer accounts/marketing/rights database/Profiles Knowledge Repository/PMS-Database via Query Support. Depending on the degree of specificity of the values known for the user preferences such values can be given lower or higher weighting as inputs to the personalisation rules for the purpose of matching possible media type/ genre or promotional offers/bundles as suitable recommended products for the user. If an actual product ID is supplied as a user preference this obviously makes for a trivial personalisation task as it will satisfy the first rule of personalisation which is that if a specific product is demanded, supply it plus see if any cross-selling or up-selling is possible via relevant horizontal or vertical offers/promotional bundles as may be appropriate.

Default evaluation: this can be computed by reference to the classificatory customer-differentiation of the particular user in terms of criteria for which there are relevant and sufficient profile elements available for example social class or personal-profile-based (age, User's previously requested media if known specifically for that differentiated classificatory user group to which the user belongs, User's Content Request History, user's job/professional status if known, post-code, etc).

Resolve-infer-getclosure (catch-all): If available it can be either over-written or appended depending on user's content request(s) or other delivery context information. If unavailable use the default value as computed above as the closure value or use any product from the general top 10% or recommend any product for the user selected from the top 10% of the most popular products for the classificatory group to which the user belongs and/or select any product from the bottom 10% or the least popular product for the age group as additional free offers and/or recommend both of the above

as long as check – there is–no–duplicated–offer rule succeeds when run against both the above two sets and the record of previous content purchased by the user.

B) User's Device Profile

Computing/Resolving these profile elements: These elements are generally locked for the same device; use cached values for the device or get the values from the CC/PP or UAProf RDF-XML-based structures supplied by the manufacturer and use a Deli-like package to resolve them.

C) Delivery Context Information

Location

Computing this profile element: This is generally locked for the same account but can be changed by the Administrator; use it to query/combine the other user profile elements from the personalisation server and delivery context elements from the PMS-Database via Query Support

Default evaluation: not applicable

Resolve-infer-getclosure (catch-all): not applicable

User's currently requested media if known specifically

Computing this profile element: This is appendable, use its value(s) to compile the Recommendation List and to query other additional offers that might be recommended (related, supplementary or complementary items etc)

Default evaluation: not applicable

Resolve-infer-getclosure (catch-all): not applicable

User's previously requested media if known specifically

Computing this profile element: This is appendable, use it to query/combine other user profile element values from the personalisation server and delivery context elements from the PMS-Database via Query Support.

Default evaluation: not applicable

Resolve-Infer-getclosure (catch-all): not applicable

User's Differentiated Service Classification if maintained by the distributor

Computing this profile element: This is over-write-able, use it to query/combine other user profile element values from the personalisation server and delivery context elements from the PMS-Database via Query Support

Default evaluation: not applicable

Resolve-infer-getclosure (catch-all): not applicable

Zone/market/customer-category-specific promotional offers (bundles) currently available

Computing this profile element: This is locked but periodically updateable/appendable by the system or the Administrator; use it to query/combine other user profile element values from the personalisation server and the delivery context elements from the PMS-Database via Query Support

Default evaluation: not applicable

Resolve-infer-getclosure (catch-all): not applicable

Current popularity chart for classified media or Top-10% and Bottom-10% of popularity ratings for media-on-demand

Computing this profile element: This is locked periodically updateable/appendable by the system or the Administrator use its value(s) to compile the Recommendation List and to query other additional offers that might be recommended (related, supplementary or complementary items etc).

Default evaluation: not applicable.

Resolve-infer-getclosure (catch-all): not applicable.

4.1.6 Media Selection & Bundling

This is the end stage process of using match-making rules to build up the final Recommendation List of items for the user based on elicited specific choices made by the user during the current content request and adding other best matched items to these based on the consideration of the user profile, device profile and delivery context information such as available and relevant promotional offers, current media popularity charts etc

4.1.7 Media Transcoding

For this process we require to consider:

- i. Format to which content must be converted for: Audio (Ringtones), Images, Text
- ii. Additional client-device-type-profile-specific information
- iii. Single or multiple pass conversion

For the Ringtones, it is required to include conversion from any format to any format; a recommended conversion is the one from Midi to other formats and thus should amount to a fully-fledged audio transcoding capability.

4.1.8 Media Adaptation

This is the final stage process of transforming content to match the play quality capability of the actual target device and its operational needs as expressed in the device profile. Once the adaptation has been achieved the new content can be sent to the end-user with the related fruition licence.

All adaptation should be able to use a managed mix of cached (previously adapted to certain formats and stored on the distribution server) and on-the-fly adaptation responsively depending on the requirements for the particular device, the mix of business models (offer types), distributions regimes (on-demand, and/or multi-cast) that may be operating.

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

Elements from these four profiles are grouped into four categories:

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

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

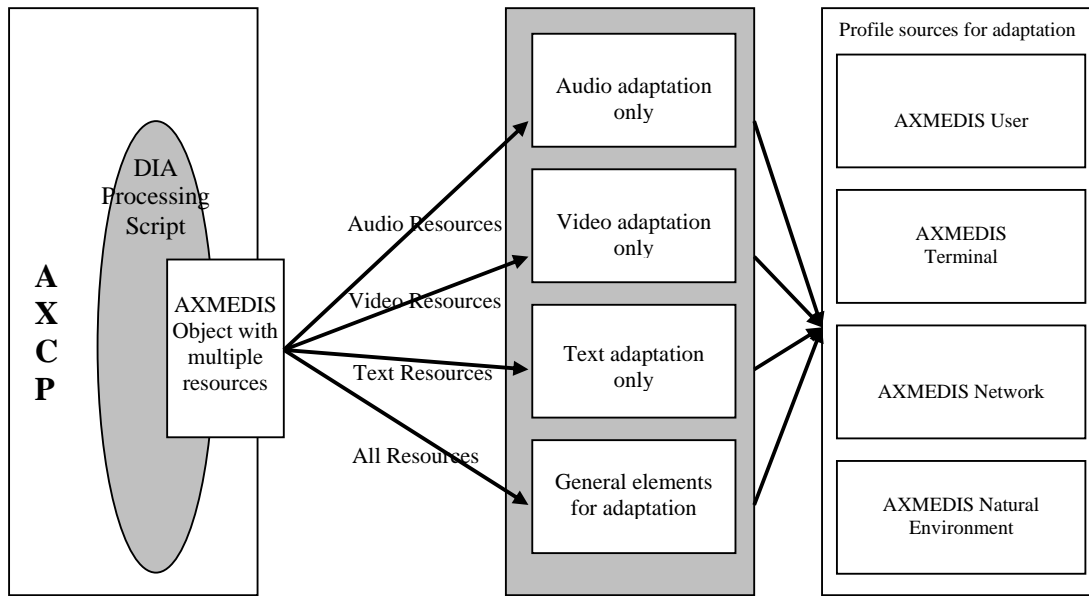


Figure 12: Categorisation of profile elements

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

This object is analysed and for each single resource within it, the functionalities provided for the adaptation process are invoked. These adaptation functionalities take as input the profiles, and for each resource within the object, ascertain its type (audio, video, image etc). and based on the type, match them against the relevant profiles elements. Accordingly if the resource is audio, its parameters are compared with a list of audio-related parameters (as well as General parameters, which apply to all kinds of resource types), in case of a video resource, the comparisons are made with video-related list of parameters as well as general elements selected from the four AXMEDIS profiles; and so on. If there are mismatches in any characteristics of the resource (regarding presentation to the user on destination device) and the profile element values, then appropriate modification/adaptation is made and the target value is stored. If there are multiple parameters affecting same object characteristics, then based on the priority appropriate adaptation parameter is stored.

Once the analyses, comparisons and necessary modification to the parameter values have been performed according to the elements of the profiles (categorised into Audio Only, Video Only, Text Only and General Elements, as mentioned above), transcoding algorithms are invoked for adaptation of the constituent resource(s) using the list of necessary modifications and adaptations to be performed on the resource to arrive at an adapted AXMEDIS object as output.

These functionalities can be accessed via JS in the AXCP engine to create and to manage the profiles dynamically and adapt objects (resources).

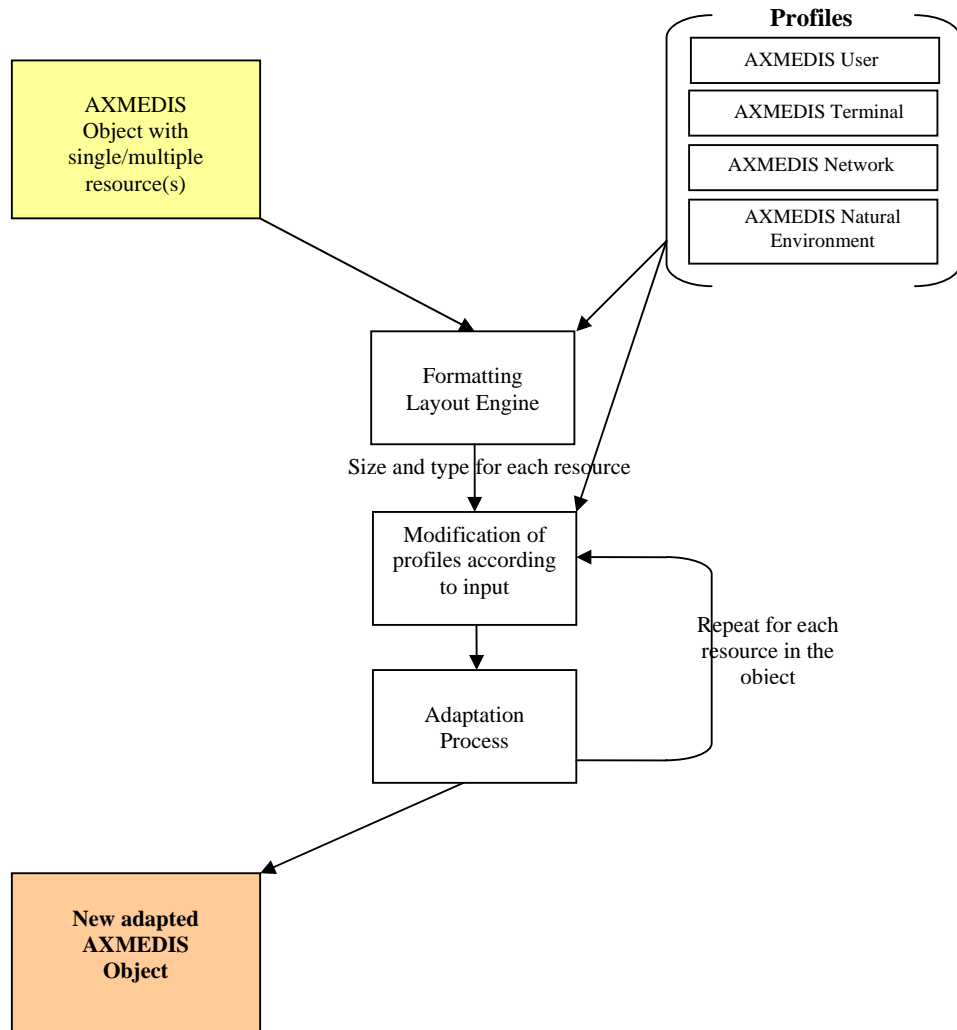


Figure 13 : DIA Process flow

4.1.9 Conclusions on Integration for mobile distribution- Requirements Update (IRC)

The main requirements for the production of content distribution towards mobile are as follows:

- i. Media adaptation for mobiles is required to cover transcoding, formatting algorithms for distribution on-demand to existing mobile devices and new generation of mobiles and PDAs.
- ii. The media adaptation must address all the various media types: audio, video, text etc and images and provide algorithms for transcoding from any format to any format. As far as audio transformation is concerned, the dynamic adaptation of ringtones has to be specifically included by the transcoding engine.
- iii. In the case of distribution to mobile phones in order to make available the media files (typically audio) to be used as a ringtone on various devices, the media are transcoded from one file format to another suitable format having regard also to operational constraints of different handsets in playing the same format. This is required because although market competition demands mean there is a large degree of convergence amongst say the top 10 mobile phones in terms of their play and display capabilities, it is still the case that even for mobile devices accepting the same file format,

- they could require adaptation of the media to accommodate their differing individual characteristics (e.g. bit-rate, sampling rate, etc) for the media.
- iv. Transcoding must carry out adaptation to suit the user, device and delivery context profiles i.e. compliance with acceptable type, content and play-display quality levels in the resulting transcoded media as experienced by the user of the client device.
 - v. The transcoding must be kept as efficient as possible i.e. it must minimise the complexity in the adaptation process. The potential volume of downloads by mobile devices requires an efficient transcoding architecture. The requirement for optimising quality, efficiency and complexity of the transcoding process for distribution to mobiles will be increasingly important for video distribution where it could be even more critical since the videos are typically large files and more complex to for transcoding.
 - vi. The transcoding integration must offer specific flexible modular solution sub-systems thus facilitating plug-and-play installation and update of transcoding components. The transcoding platform has to be extensible with regards to the number of supported media types, RAM size etc.
 - vii. The Media adaptation must incorporate separate profiling channels to allow dynamic media adaptation optionally to cover any managed mix of the various elements of the delivery context such as device, network session and personal profiles.
 - viii. To enable smart mobile device profiling, the system must possess some dynamic profile collection and update capability using new standards like UAProf based on the new multi- model RDF as can be adopted and embedded inside the transcoding platform.
 - ix. The integration of a profiling tool in the AXMEDIS Content Processing Engine has to be such as to remain open to any type of distribution models. The adaptation algorithms have to be thus directly accessible from both Axmedis authoring tools and the AXCP Java script Engine.
 - x. The AXMEDIS-integrated platform will be required also to support DRM since DRM-capable transcoding is presently largely non-existent. This implies a requirement for dynamic data exchange with the AXMEDIS certifier and supervisor module both to keep an audit on the acquired rights usage restriction and DRM compliance as well as for content selection for personalised delivery.

4.2 Introduction to Mobile Transcoding

Mobile Transcoding refers to the adaptation of multimedia information to enhance usability and manage the variable delivery to cater for heterogeneous client devices and user requirements on-demand. Content adaptation refers to the modification of the parameters of a specific media type. There are multiple facets to developing such a framework foremost of which is a generalised representation of content to allow access to individuals with varying disabilities. Transcoding can be defined as the conversion of one coded signal to another. While this definition can be interpreted quite broadly, it should be noted that research on audio/video transcoding is usually focused on adaptation either to a reduced availability of transport channel bandwidth or to a smaller display or to both these constraints. In the earliest work on transcoding, the majority of interest focused on reducing the bit rate to meet an available channel capacity. Additionally, researchers investigated conversions between constant bit-rate (CBR) streams and variable bit-rate (VBR) streams to facilitate more efficient transport of audio/video. In recent times as mobile devices with limited display and processing power became prevalent, transcoding objectives included achieving spatial resolution reduction, as well as temporal resolution reduction. Transcoding techniques are used to serve the same underlying multimedia object at different quality levels to different users to accommodate users' operational constraints. The main effect of transcoding, which is data reduction helps to reduce network end-to-end delay, loss, and delay jitter for the multimedia packets. Transcoding can be done in a number of ways including Spatial transcoding (to reduce the frame size), Temporal transcoding (to decrease the frame rate by dropping less significant frames), Colour transcoding (to reduce the data size by decreasing the colour depth). Design of the transcoding algorithm and the placement of the gateway are important issues in transcoder based adaptation schemes. During transcoding, the transcoder must receive the information about the receiver's as well as the sender's device type and their basic specifications. The information about the device types are saved in xml files based on either CC/PP profile or UAProf protocol. The key design goals of transcoding include two aspects:

- To maintain the perceived quality during the transcoding process.

- To keep the transcoding process complexity as low as possible.

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

4.2.1 Libraries for Video Transcoding

In the following paragraphs are briefly reported the most important supportive utilities to be integrated to handle transcoding. Please note that all transcoding operation will be performed via AXCP.

4.2.1.1 FFmpeg

FFmpeg is a complete solution to record, convert and stream audio and video. It includes libavcodec, the leading audio/video codec library. FFmpeg is developed under Linux, but it can be compiled under most operating systems, including Windows. FFmpeg is a very fast video and audio converter. It can also grab from a live audio/video source. The command line interface is designed to be intuitive, in the sense that ffmpeg tries to figure out all the parameters, when possible. You have usually to give only the target bitrate you want. FFmpeg can also convert from any sample rate to any other, and resize video on the fly with a high quality polyphase filter.

4.2.1.2 FOBS

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

4.2.2 Libraries for Image Transcoding

In the following paragraphs are briefly reported the most important supportive utilities to be integrated to handle transcoding. Please note that all transcoding operation will be performed via AXCP.

4.2.2.1 ImageMagick Library

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

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

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

4.2.3 Libraries for Audio Transcoding

In the following paragraphs are briefly reported the most important supportive utilities to be integrated to handle transcoding. Please note that all transcoding operation will be performed via AXCP.

4.2.3.1 FFmpeg

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

4.2.3.2 Libsndfile

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

libsndfile has the following main features :

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

4.2.3.3 SoundTouch

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

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

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

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

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

4.3 Introduction to device profile management

As already mentioned in the previous section we are going to integrate a relevant amount of standardised technologies provided by the AXMEDIS framework. Hereafter we report just some major outlines for reference purposes.

4.3.1 CC/PP

CC/PP stands for Composite Capabilities/Preferences Profiles, and is a way to specify what exactly a user agent (web browser) is capable of doing. A CC/PP profile is a description of device capabilities and user preferences that can be used to guide the adaptation of content presented to the particular device. Here profile does not refer to a subset of a particular specification, for example the CSS Mobile profile, but refers to the document(s) exchanged between devices that describe the capabilities of a device.

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

4.3.2 UAProf

The UAProf specification is based on the CC/PP specification. Like CC/PP, a UAProf profile is a two level hierarchy composed of components and attributes. Unlike CC/PP, the UAProf specification also proposes a vocabulary - a specific set of components and attributes - to describe the next generation of WAP phones. The specification also describes two protocols for transmitting the profile from the client to the server. Most mobile devices now use UAProf because this is capable of preserving the ordering of profile elements and resolution of how to treat the profiles as new profiles are added (device, personal and channel profiles).

4.3.3 DELI

There is an open-source library called DELI developed at HP Labs that allows Java servlets to resolve HTTP requests containing delivery context information from CC/PP or UAProf capable devices and query the resolved profile. Currently DELI only supports the W-HTTP protocol. Profiles using the UAProf vocabulary consist of six components: HardwarePlatform, SoftwarePlatform, NetworkCharacteristics, BrowserUA, WapCharacteristics and PushCharacteristics. These components contain attributes. In DELI each attribute has a distinct name and has an associated collection type, attribute type and resolution rule. In UAProf there are three collection types:

- *Simple* contains a single values;
- *Bag* contains multiple un-ordered values;
- *Seq* contains multiple ordered values.

Some collections of properties and property values may be common to a particular component. For example: a specific model of a smart phone may come with a specific CPU, screen size and amount of memory by default. Gathering these "default" properties together as a distinct RDF resource makes it possible to independently retrieve and cache those properties. A collection of "default" properties is not mandatory, but it may improve network performance, especially the performance of relatively slow wireless networks. From the point of view of any particular network transaction the only property or capability information that is important is whatever is "current". The network transaction does not care about the differences between defaults or persistent local changes; it only cares about the capabilities and preferences that apply to the current network transaction. Because this information may originate from multiple sources and because different parts of the capability profile may be differentially cached, the various components must be explicitly described in the network transaction. DELI provides support for legacy devices so that the proprietary delivery context descriptions currently used by applications can be replaced by standardised CC/PP descriptions. DELI can read vocabularies described using RDF schemas. The WAP Forum has published two such schemas to describe the two versions of UAProf currently in use. However the RDF Schema in its previous form did not provide an easy way of describing attributeType or resolution rules. Therefore the UAProf schemas stores this information in the comments field for each attribute. Therefore DELI also parses the comments fields to create the vocabulary. This is not an ideal solution so hopefully a more robust way of representing this information will be used in later versions of UAProf.

4.3.4 WURFL

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

4.3.5 The GPAC suite

GPAC is a multimedia framework based on the MPEG-4 Systems standard (ISO/IEC 14496-1) developed from scratch in ANSI C. As of version 0.4.0 GPAC is licensed under the GNU Lesser General Public License. The GPAC toolset is composed by several tools both for content creation (authoring tools) and for content consumption (players). The two most complete tools are MP4Box for content creation and the OSMO player. Concerning the OSMO player, its “core” decoding functionality are implemented in the so-called “Extra libs” among which FFmpeg is the most complete in terms of supported formats (a wide range of MPEG formats are supported). The OSMO player provides the widest support of media decoding possibilities. The supported format are:

- MP4 and 3GPP file reading, both local and through http download (QuickTime FastStart).
- MP3 (local and http) and ShoutCast/ICECast radios.
- AAC file reading http streaming (AAC/aacPlus radios) .
- Media Codecs: MPEG-4 Visual Simple Profile, MPEG-4 Audio AAC, JPEG, PNG, AMR/AMR-WB audio and all codecs supported by the FFMPEG library (including AVC/H264).
- All media containers supported by the FFMPEG library: avi, mpeg, vob, etc...
- Xiph.org Media: Ogg file format (including http read and Icecast), Vorbis audio and Theora video.
- 3GPP Timed Text / MPEG-4 Streaming Text.

Other features of the OSMO player related to connectivity, protection and multimedia format include:

- Streaming support: RTP and RTSP/SDP for MPEG-4 Visual/Audio (including simple LATM), MPEG-1/2 audio and video, 3GPP timed text, AVC/H264, H263 video, AMR.
- Simple ISMA E&A Decryption support.
- Multichannel audio, multichannel to stereo mapper.
- MPEG-4 scenes (2D, 3D and mixed 2D/3D scenes) - read from binary format (BIFS) and textual format (BT/XMT-A).
- VRML 2.0 (VRML97) scenes (without GEO or NURBS extensions).
- X3D scenes (not complete) - supports both X3D (XML format) and X3DV (VRML format).
- JavaScript support for MPEG4/X3D/VRML.
- Compressed description (GZip) supported for all textual formats of MPEG4/X3D/VRML.
- Simple SVG scenes (not complete).
- Simple SWF (Macromedia Flash) scenes (no ActionScript, no clipping, etc).
- HTTP reading of all scene descriptions.

According to the GPAC web site the “Extra libs” can be compiled using Embedded Visual C 4.0. On the other hand, FFmpeg will not compile as Visual C++ project since it has too many dependencies on the GCC compiler to make a port viable. However, if the FFmpeg libraries need to be used on Windows platforms developers can still compile their applications using Visual C++. An important restriction to this is the use of the dynamically linked versions of the FFmpeg libraries (i.e. the DLLs), and the creation of Visual-C++-compatible import libraries during the FFmpeg build process. In order to produce the necessary DLLs it necessary to install the current versions of MSYS and MinGW from <http://www.mingw.org/>. These tools basically provide a GCC like environment for compiling software developed on UNIX/Linux and heavily relying on GCC dependencies.

The support of state-of-the art video format is very limited, but its availability for free may be considered for the distribution of evaluation AXMEDIS players to end users for personal use and as a first access to the AXMEDIS

Table I - Comparison among the most popular media player for mobile devices

Software	PocketTv	MS Media Player	TCPMP/ Betaplayer	Resco Picture Viewer	Pict Pocket Cinema
----------	----------	-----------------	-------------------	----------------------	--------------------

DE9.5.4 Integrated Prototype of content Production and Distribution on-demand for Mobile Phones, and new generation of PDAs

Software	PocketTv	MS Media Player	TCPMP/ Betaplayer	Resco Picture Viewer	Pict Pocket Cinema
Video File Format(s)	Standard MPEG-1	Proprietary: Windows Media Video with MS Audio and ASF Multiplex.	Proprietary: DivX, Xvid, WMV Standard: MPEG-1	Standard MPEG-1	Standard MPEG-1
File extension	.mpg; .mpeg	.wmv; .asf	.avi; divx; wmv; .mpg	.mpg; .mpeg	.mpg; .mpeg
Can Stream Video	Yes	Yes	No	No	No
ATI support (1)	Yes	?	Yes	Yes	No
2700G support (4)	Yes	No	Yes	Yes	No
Dither	Yes	No	Yes	Yes	Yes
Step (2)	Yes	No	No	No	No
Microdrive Optimization	Yes	No	Yes	Yes	No
Smartphone support	Yes	Yes	Yes	Yes	No
Jpeg capture	Yes	No	No	No	No (.bmp)
Support "odd"sizes (3)	Yes	n/a	Yes	Yes	Yes

(1) ATI support: Includes optimizations for the ATI Imageon 3200-series video accelerator when available (e.g. iPaq hx4700, Toshiba e800/e805, O2 XDA II).

(2) Step: Allows stepping frame-by-frame to observe individual frames.

(3) Supports "odd" sizes: Some MPEG players do not play properly MPEG with "odd" size, i.e. image size not multiple of 16x16.

(4) 2700G support: Includes optimizations for the Intel Marathon 2700G video accelerator when available (e.g. Dell Axim x50v).

5 Distribution towards Mobile

How does it work distribution to Mobiles in AXMEDIS? Well this is a specialised distribution channel operating at B2C level and covering a quite dispersed set of user needs and activities. In more details it represents a point of service where users will connect to search and retrieve content related to their activities and interests with a common usage of technology and distribution tools. Users will have a very simple and self-explanatory interface that will present content by category, where each category basically represents a specific query that will return a list of objects matching the criteria used to define the category. Contents will be adapted to end user device in order to maximise the efficiency in the delivery phase. User preferences will be taken into account at content production and organisation level so to define proper categories in the light of end-user selections.

Initially focus was placed on the possibility to simplify the management of the process from a client perspective as the present number of constraints for mobile-based delivery of content is quite relevant. The designed application covers primarily two aspects of the process, namely the management side and the end user side. The management side is primarily devoted to the distributor that has to manage the mobile delivery channel and basically covers aspects like content categories selection and activation (this includes the definition of the query that will be executed at run-time and that is associated to each category), device and user profile management.

The end user side covers the search and retrieval of content and is clearly relaying on the category definition performed by the distributor. Each category (at client side) is basically representing a special query that is placed at server side when the user selects the category. The query returns the list of objects that the user can access (for preview or acquisition). Once the selection is performed the user can purchase the content and, once completed the process, also access to it using the specific player (eventually downloaded in advance).

To deliver AXMEDIS objects using through mobile devices, there are two scenarios:

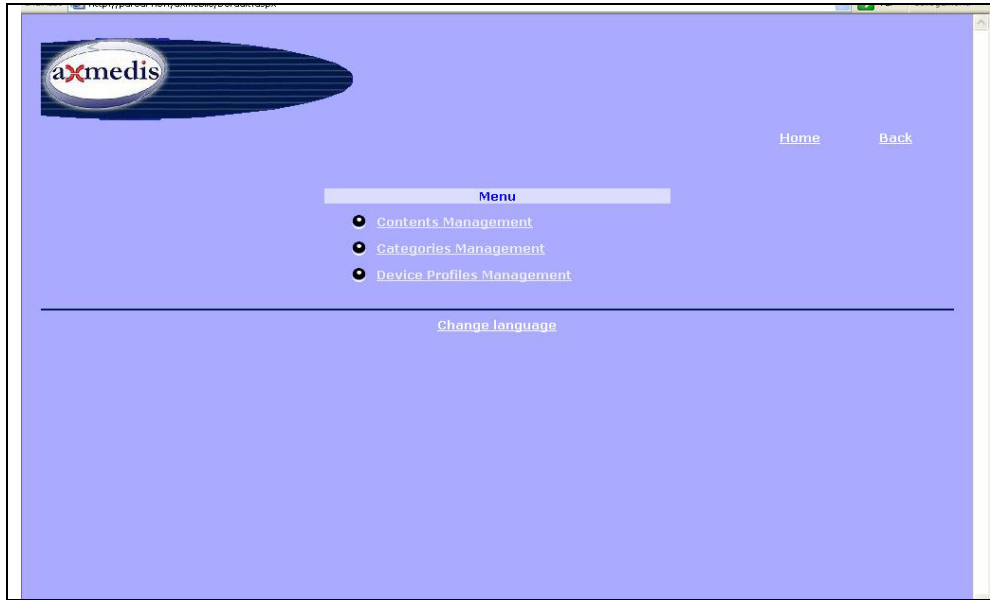
Scenario 1. In the first scenario a publisher prepares content that wants to distribute to mobiles. The publisher needs to select a list of objects, arrange them into categories, define the query needed to retrieve such objects, create the needed categories and activate them. User performed operations (download and acquisition) will have to be monitored and then used to further enhance the content selection and offer step.

Scenario 2. In the second (On-Demand) scenario, a user selects an object in the AXMEDIS Distribution Area. The request from the distribution area starts a series of events behind the scenes where the user sees the object requested.

5.1 Mobile content selection and categorisation (server side)

A publisher prepares some contents that wants to distribute via mobile. The publisher needs to select a list of objects, arrange them into categories, define the query needed to retrieve such objects, create the needed categories and activate them. User performed operations (download and acquisition) will have to be monitored and then used to further enhance the content selection and offer step. The publisher needs to select a list of objects, create/modify the categories to be seen by the end-user, create the query to connect to the created category. Last but not least activate the category.

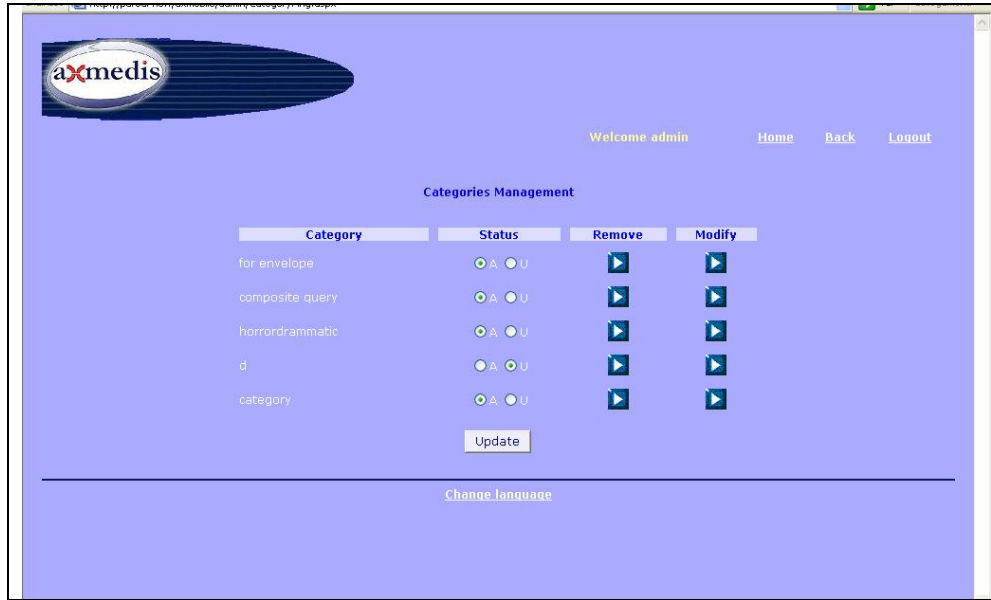
So operatively the publisher, by using the AXMEDIS Query support, searches the desired objects and selects the one to be used for the distribution via mobile. By using the AXMEDIS Editor and a specific application the publisher produces the categories that the user will see and the related queries. On completion, the publisher activates the desired categories that from that moment on will be available to the end-user for selection.



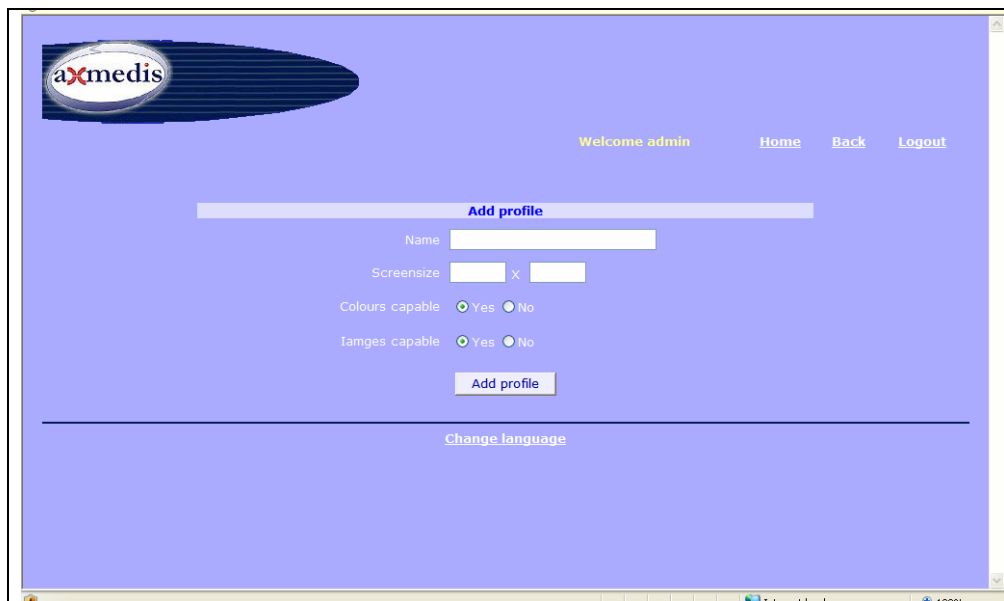
Publisher's application home page.

The screenshot shows the 'Add new category' form. At the top, it says 'Welcome admin' and has links for 'Home', 'Back', and 'Logout'. The form title is 'Add new category'. Below it, a note says 'Fields marked by * are mandatory.' The form includes: a '*Identifier:' field with the value 'ART_ILABSO'; a '*Name:' section with two rows: 'Italiano (IT)' with the value 'Arte' and 'English (US)' with the value 'Art'; a '*Query:' section with a table for defining search criteria. The table has columns for 'Field', 'Operator', 'Value', and 'Condition'. It contains two rows: one for 'Description' with operator 'contains' and value 'art', and another for 'Title' with operator 'contains' and an empty value field. At the bottom of the form are 'View query' and 'Create category' buttons, and a 'Change language' link at the very bottom.

Adding a new category, the publisher has to establish a category key, a category name for each one of the supported languages and the related retrieving query, that is, the classification criteria for the objects belonging to that category.



From the category management section it is possible to change the status of the categories (visible to users or not), to remove and modify the categories.



The Publisher has also the responsibility of inserting into the system the data of the supported device profiles.

DE9.5.4 Integrated Prototype of content Production and Distribution on-demand for Mobile Phones, and new generation of PDAs

axmedis

Welcome admin [Home](#) [Back](#) [Logout](#)

STEP 1/4

Look into

☒ AXDB ☐ AXEPTOOL

Field	Operator	Value	Condition
DCMI: contributor	CONTAINS		
DCMI: contributor	CONTAINS		

[Change language](#)

Once the publisher has setted up the system inserting data for some categories and some device profiles, he can start the process of content adaptation and publishing. The first step is the search of some contents...

axmedis

Welcome admin [Home](#) [Back](#) [Logout](#)

STEP 2/4

Contents from query

Please choose contents to prepare for Mobile and confirm through buttons below.

- ☒ [L'Erba di Grace](#)
- ☒ [Love Laughs at Andy Hardy](#)
- ☒ [McIntock!](#)
- ☒ [The Inspector General](#)
- ☐ [The perils of Pauline](#)
- ☐ [The Ring - Vinci per me!](#)
- ☐ [The sin of Harold Diddlebock](#)

Page 2 of 3

[\[First Page\]](#) [\[Previous Page\]](#) [\[Next Page\]](#) [\[Last Page\]](#)

[Next](#)

[Change language](#)

... then a selection among these contents...

axmedis

Welcome admin Home Back Logout

STEP 3/4

Categories Management

Select one or more categories into which selected contents will be put.

☐ for envelope

☐ composite query

☐ horror dramatic

☐ d

☒ category

OK

Change language

... then the publisher decides into which categories the contents will be inserted...

axmedis

Welcome admin Home Back Logout

STEP 4/4

Devices Selection

Please select for which devices contents gave to be adapted.

☒ ERICSSON007

☒ LG20000

☐ NOKIAxyz

☐ SONYalfa


OK

Change language

... and for which devices the contents will be adapted.

5.2 Mobile content distribution (client side)

So what happens when a mobile distribution channel is used?



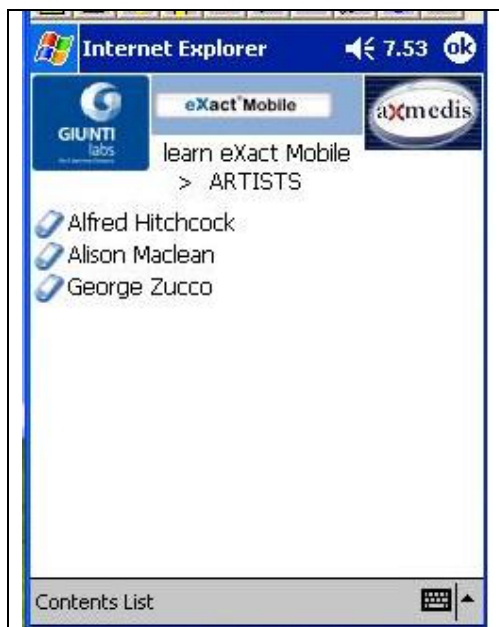
The user shall register...



... or log on to be recognised and authorised to access to provided services.



The user will be able to browse the available categories...



... and select one. This will automatically activate the related query for content retrieval, delivery and ...

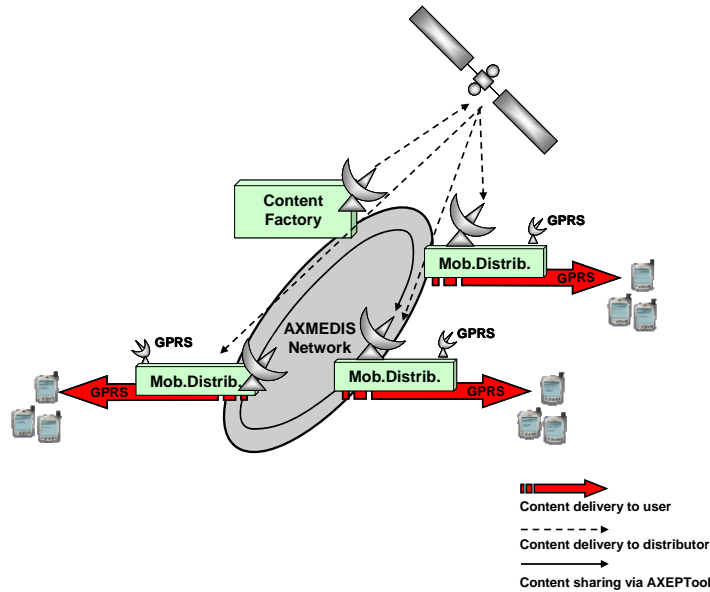


... fruition.

6 Mobile distributor demo fact sheet

Within AXMEDIS the mobile demonstrator has a specific role and aim, to demonstrate the benefits coming from the combination of several technologies in a well established, yet rather new environment. Usually when referring to mobile distribution people is biased to “voice”, “text” or MMS content, but actually there is much more from music (MP3) to games and recently also DTV that have added up to already available services. In AXMEDIS a the mobile channel is aimed primarily to show how all these contents can be combined and complemented thanks to the possibilities added by the interaction with personalised adaptation services and interoperable DRM.

The mobile within AXMEDIS Architecture is divided in two components: the “factory” where content are produced and the “mobile distributor” where users have access to services and contents. Distribution from the factory to mobile distributor is achieved via satellite so to optimise bandwidth and data transfer rate when updating (in broadcast) distribution servers that may be geographically dispersed on the territory, while content access, selection, acquisition and fruition will be performed by mobile terminals (true points of service) represented by new generation of mobiles or PDA based smartphones. The following diagram gives a better view of the involved components and their relations.



- Mobile architecture integration with AXMEDIS
 - Content Acquisition / provision:
 - From partners' databases, from partners granting access, through Query Support and LoaderSaver Web Services
 - Internally produced
 - Content Production / Processing (SMIL, HTML...):
 - From Mobile factory, through either the AXMEDIS Editor or the AXCP Editor through rules and templates.
 - Number of content items produced/processed per month will largely depend on the expected number of events that will be held at the mobile distributor location in the same period, but in line of principle they will range from 30 to 50.
 - number of content items produced/processed at the same time will derive directly from the previous procedure as once defined the target for the incoming month production, related templates will be selected or produced, rules will be selected or adapted from available ones and then content will be aggregated consequently via batch process.
 - Content protection will be achieved from the mobile factory, through the:
 - AXMEDIS Editor
 - AXCP Editor through rules
 - Mother licenses will be produced in the mobile factory for our contents and on the basis of a master license granted by ANSC, through:
 - AXMEDIS DRM Editor
 - AXCP Editor through rules

DE9.5.4 Integrated Prototype of content Production and Distribution on-demand for Mobile Phones, and new generation of PDAs

- End-user final licenses will be produced at the mobile factory side through:
 - AXMEDIS DRM Editor
 - AXCP Editor through rules
- Registration of user and devices, will be needed for the player that will have to be installed on the user device and for the AXMEDIS tools installed on the Factory side.
- Content distribution for mobile will be based on category selection and strictly connected to user and device profiling.
- Accounting collection and action monitoring, will rely on the fact that the AXOM inside AXMEDIS Tools will communicate to the AXCS the usage data. Both Accounting services and AXCS will be at the Factory side
- Description of the effective installation
 - Distribution infrastructure needed comprises a server (where AXMEDIS Tools will be installed along with the firewall and the other back office components needed) and a few PDAs/Smartphones for experimental content fruition.
 - Servers: at least one
 - Players: one for each device in use
 - Streaming/downloads: TBD
- AXMEDIS tools
 - Major AXMEDIS tools used:
 - The whole set of AX-Tools
 - P2P will be used in the MOBILE-factory mainly for content search and retrieval or for content sharing between partners involved in the demonstrator (ANSC and ILABS)
 - AXCP will be used at the factory side in order to create contents and licenses
 - Workflow will be used to schedule and control periodic updating of categories lists on the basis of usage statistics analysis and monthly local event planning
 - Programme and publication will be used to feed on time distribution servers
 - PMS/AXCS usage:
 - During content creation in order to protect and create mother licenses
 - During content distribution preparation, in order to create end user licenses
 - During content fruition, through AXMEDIS players and tools, since they have to verify the users' rights
 - AXMEDIS database usage:
 - Mobile-factory database instance, for locally storing and retrieving
 - Remote databases, to retrieve partners' contents

- Target Market:
 - Cultural heritage institutions and/or societies providing services for museums and archaeological sites.
- Description of the business model
 - The expected business model is a combination of two typical ones: the subscription and the pay per access/use. Payments will be done in cash and the paying user will be assigned a code that will be used in payment procedures (the code corresponds to one or several tokens that will grant access/fruition of selected content)
- Description of content:
 - On the average each month will be available a set of about 100 AXMEDIS objects ready to be distributed and focused on the defined event schedule. To this has to be added the availability of all the collections which are provided by ANSC and ILABS and that will be accessible in a standardized manner that may differ from the “monthly” offer mainly in terms of aggregation.
 - Digital resources with the needed clearance of rights will come from ANSC and ILABS
 - Content will comprise data related to the collections of the ANSC museums and events organized at the ANSC auditorium plus textual and graphical content related to art and history from ILABS.
 - Available resources will certainly comprise text, images and audio, while videos may be available but only in limited number and dimensions.
 - Typical Content size for each content type may greatly vary especially as far as audio/video is concerned while the typical size textual/graphical content is in the order of 2-10Mb.
- Final Users/Clients:
 - In terms of final users it is expected that a 5% of regular visitors of the ANSC museum and auditorium could be highly probably interested in experimenting the mobile services installation and use its content, in this manner approximately 250 final users per year will be reached.
- Partners involved and roles:
 - ANCS content owner and manager of the local site
 - ILABS content owner, manager of the factory

7 Installation and configuration

From here and for the following, be \$REPOS\$ = <https://cvs.axmedis.org/newrepos/> (the CVS code repository of the project)

Prerequisites:

- Microsoft.NET Framework v1.1.4322 or higher
- MySQL 4.1.16 or higher
- Internet Information Services (IIS) 5.1 or higher

Factory side:

- create and populate a database using the scripts at \$REPOS\$Applications/mobilecomponents/doc/other/
- create a folder, be it axmobile_folder and copy in it:

- \$REPOS\$Applications/mobilecomponents/bin/ (create a bin subfolder)
- \$REPOS\$Applications/mobilecomponents/source (all the files inside, recursively)
- Open Web.config file inside axmobile_folder and check the application parameters, edit them if you need to
- Create from IIS a virtual folder and make it point to the axmobile_folder, name it for example “axmobile”

To test go to:

http://<your_host_name>/axmobile/Default.aspx

you should be able to see the main page of the application.

8 Technical specification and details

Here we present the description of the architecture of the distribution towards Mobile specified in terms of blocks, classes, interfaces, data structures, methods and relations; possible GUIs are depicted too. In the following picture the overall scenario is represented together with involved actors and relations.

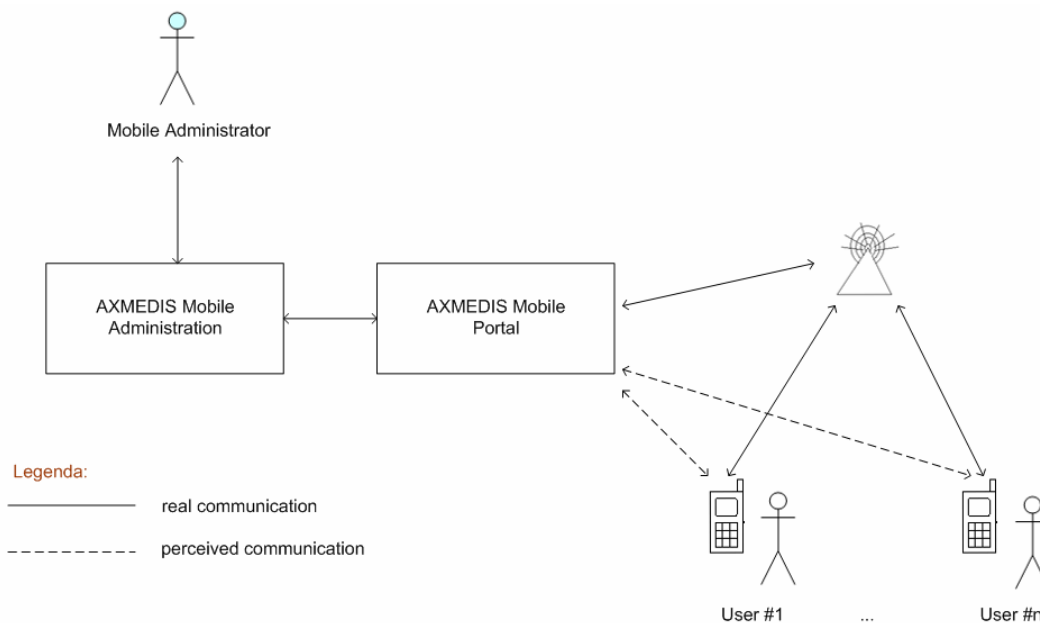


Figure 14. Overall Scenario and involved actors.

Actors

In this scenario the foreseen actors are the following:

- ❑ **Mobile Administrator:** interacts with the Mobile Server Side in order to execute Administrative functions. His tasks include: domain registration; content selection criteria definition; content preparing; supported device profiles management; users management.
- ❑ **User #x:** a generic user that can interact with the Mobile Application via a Mobile device. The user will need to register and certify own device prior to be granted the possibility to use it.

System components

System components can be summed up as follows. This one is a very high level, just to introduce the whole architecture; we will enter in details later.

- ❑ **Mobile Administration:** this is the part of the system that supports administration functionalities previously described.
- ❑ **Mobile Portal:** this is the part of the system that provides access to services to Mobile Devices.

8.1 Overall Use Cases

Before any user can connect to AXMEDIS Mobile Application, an AXMEDIS Mobile Administrator has to register a new domain. The Administrator has also to create some content selection criteria that can be used later by Portal users. A kind of content preparation is also required, in which we have to distinguish an authoring phase and a delivery phase. In the authoring phase, content for mobile is created; AXMEDIS content is packed into an IMS Content Package, where an item refers to a resource that is the AXMEDIS object. In the delivery phase, content is played through a dedicated player. Mobile Administrator has also the task of inserted supported device profiles management. Furthermore, he can manage users data, adding, updating and removing them as needed. He also manages users roles within the Mobile Application. When a user wishes to use AXMEDIS Mobile Application, s/he has to connect to Mobile Portal and register. Then s/he can certificate his own device. Now s/he can login and browse through available contents. A mechanism of query on demand is also supported. When the user finds a content of interest, s/he can ask more information about it, have a short preview and decide to get it or not. If the user decides to acquire some rights over the object, s/he asks for the license containing those rights and has to pay the related amount. Once acquired the requested license/s, he can exercise granted rights through one or more downloaded plug-ins. Users can also find, the list of the objects for which rights have been acquired. The user can also check personal data inserted and modify them. It is now possible to define the use cases depicted in the following diagram for the Mobile Scenario, where foreseen actors are two: the “*Mobile Admin*” and the “*User*”. The Mobile Admin has privileges to perform administrative tasks, such as Domain Registration for the whole Mobile Application, Content Preparing, definition of criteria for content retrieving, supported device profiles management, users data management. The User can certificate, register, search for contents, retrieve information about available contents, acquire some rights and exercise them. In the following diagram light blue has been used to identify actions related to the Mobile Admin, and no color for actions allowed for the User. Language Selection is in green and is not related to other actions only for sake of clarity.

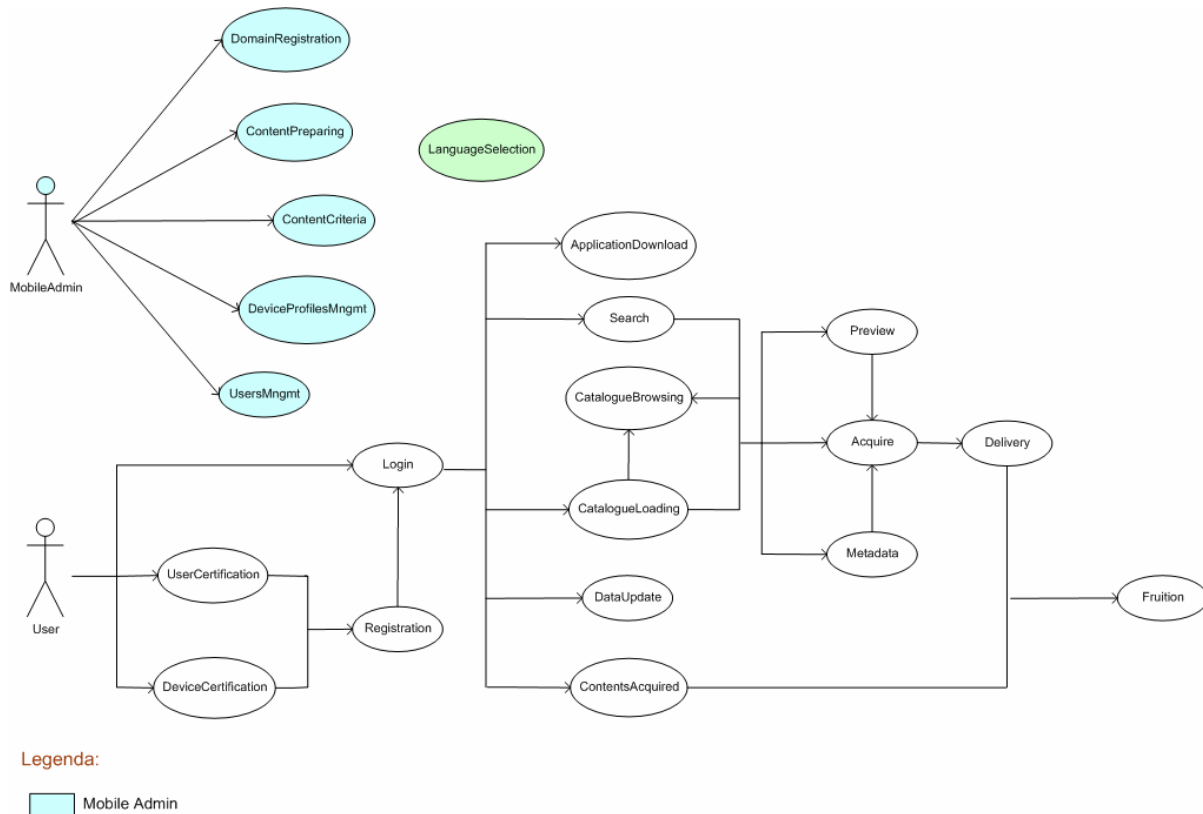


Figure 125. Use Cases.

8.2 Overall Architecture

The AXMEDIS Mobile Application Architecture foresees a number of modules with different functions: some constitute the user interface, others manage contents and data manipulation, others perform a communication function between Mobile Application and the whole AXMEDIS system. Below a sketch of the whole architecture is presented at a high abstraction level in term of packages. Inside a package one or more classes will be used as detailed later.

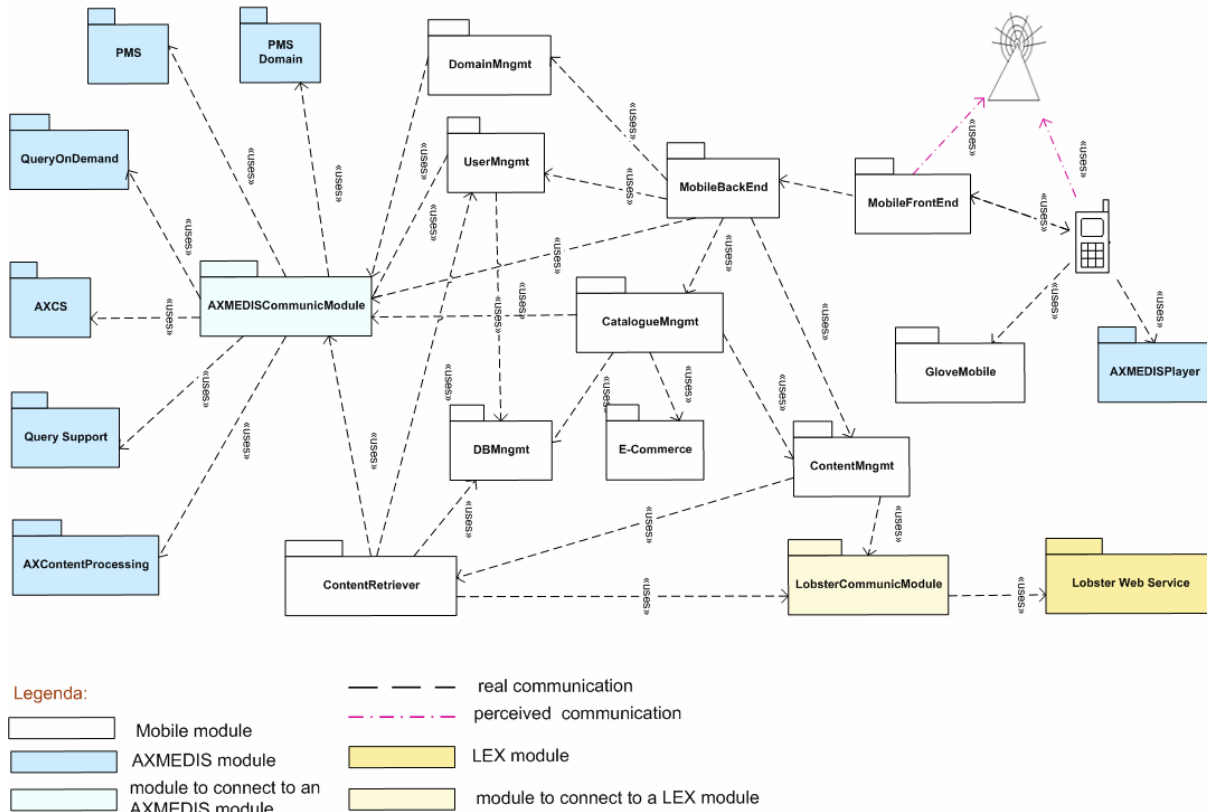


Figure 136. Overall Architecture

8.3 Data Types

Before giving details of modules, we'll shortly illustrate data types used. Apart from simple types (string, boolean and so on) we will use following data types:

- **AXMEDIScontent:** a content as defined within AXMEDIS system; for further details please see DE3.1.2 AXFW Specifications (General and Model) Part-A Section8
- **Category:** this object is defined within Mobile Application. It is meant as an "area of interest" and it consists of:
 - a category key, that has to be unique inside the Mobile Application
 - a sequence of category names, one for each language supported by the Mobile Application; these names are the ones that will be seen by Mobile Application users
 - an associated query to retrieve contents for category
- **DeviceProfile:** device profiles are expressed using DELI, for further details please see <http://sourceforge.net/projects/delicon/>, <http://www.w3.org/Mobile/CCPP/>
- **Domain:** a Domain object as defined in AXMEDIS-DE3-1-2H-AXFW-Spec-(Protection-Account)-Part-H, Section 4.6.4. Here we report just a draft of its xml definition:

```
<xs:complexType name="Domain">
  <xs:sequence>
    <!-- ID of the AXMEDIS Domain -->
    <xs:element name="AXDOM" type="xs:string" nillable="true"/>
  </xs:sequence>
</xs:complexType>
```

```
<!-- Domain Manager ID: it can be an ID of a B2B User or an End User according to the value of typeOfID -->
<xs:element name="AXID" type="xs:string" nillable="true"/>
<!-- Reference to AXID and can have one of the two values>B2BUser/AXUID -->
<xs:element name="typeOfID" type="xs:string" nillable="true"/>
</xs:sequence>
</xs:complexType>
```

- **IMSContentPackage:** IMS Content Packaging focuses on defining interoperability between systems that wish to import, export, aggregate, and disaggregate packages of content. For further details please see <http://www.imsglobal.org/content/packaging/index.html>
- **RegistrationResult:** this object tells the result of an user registration, for further details please see AXMEDIS-DE3-1-2H-AXFW-Spec-(Protection-Account)-Part-H, Section 2.2.3
- **UserProfile:** we are going to use a subset of LIP standard (for more details please see <http://www.imsglobal.org>). In detail we are going to use following elements, here reported with their multiplicity:

Element (Multiplicity)	
learnerinformation (1..1)	
	comment (0..1)
	contenttype (0..1)
	identification (0..*)
	comment (0..1)
	contenttype (0..1)
	formname (0..*)
	name (0..*)
	address (0..*)
	contactinfo (0..*)
	demographics (0..*)
	ext_identification (0..1)
	accessibility (0..*)
	comment (0..1)
	contenttype (0..1)
	language (0..*)
	preference (0..*)
	eligibility (0..*)
	disability (0..*)
	ext_accessibility (0..1)
	interest (0..*)
	typename (0..1)
	comment (0..1)
	contenttype (0..1)
	product (0..1)
	description (0..1)
	ext_interest (0..1)
	ext_learnerinfo (0..*)

We report a sample of a LIP user profile :

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE learnerinformation SYSTEM "ims_lipv1p0.dtd">
<!-- Author: Elisabetta Parodi -->
<!-- Date: 18th January, 2006 -->
<!-- Proposed Profile -->
<learnerinformation>
  <comment>An example of LIP Accessibility information.</comment>
  <contenttype>
    <referential>
      <sourcedid>
        <source>IMS_LIP_V1p0_Example</source>
        <id>basic_1001</id>
      </sourcedid>
    </referential>
  </contenttype>
  <identification>
    <comment>-----Identification</comment>
    <formname>
      <typename>
        <tysource sourcetype="imsdefault"/>
        <tyvalue>Preferred</tyvalue>
      </typename>
    </formname>
  </identification>

```

```

        </typename>
        <comment>-----Formatted Name details</comment>
        <text>Miss Elisabetta Parodi</text>
    </formname>
    <name>
        <typename>
            <tysource sourcetype="imsdefault"/>
            <tyvalue>Preferred</tyvalue>
        </typename>
        <comment>-----Name details</comment>
        <partname>
            <typename>
                <tysource sourcetype="imsdefault"/>
                <tyvalue>First</tyvalue>
            </typename>
            <text>Elisabetta</text>
        </partname>
        <partname>
            <typename>
                <tysource sourcetype="imsdefault"/>
                <tyvalue>Last</tyvalue>
            </typename>
            <text>Parodi</text>
        </partname>
    </name>
    <address>
        <typename>
            <tysource sourcetype="imsdefault"/>
            <tyvalue>Permanent</tyvalue>
        </typename>
        <comment>-----Address details</comment>
        <street>
            <streetname>via XXV Aprile</streetname>
            <streetnumber>34</streetnumber>
            <aptnumber>18</aptnumber>
        </street>
        <city>Sestri Levante</city>
        <postcode>16100</postcode>
        <country>Italy</country>
    </address>
    <contactinfo>
        <typename>
            <tysource sourcetype="imsdefault"/>
            <tyvalue>Private</tyvalue>
        </typename>
        <comment>-----Contact details</comment>
        <telephone>
            <countrycode>+39</countrycode>
            <areacode>010</areacode>
            <indnumber>6655443</indnumber>
        </telephone>
        <mobile>
            <countrycode>+39</countrycode>
            <areacode>347</areacode>
            <indnumber>6655443</indnumber>
        </mobile>
        <email>e.parodi@giuntlabs.com</email>
    </contactinfo>
    <demographics>
        <typename>
            <tysource sourcetype="imsdefault"/>
            <tyvalue>Adult</tyvalue>
        </typename>
        <comment>-----Demographic details</comment>
        <gender gender="F"/>
        <date>
            <typename>
                <tysource sourcetype="imsdefault"/>
                <tyvalue>Birth</tyvalue>
            </typename>
            <datetime>1980:07:20</datetime>
        </date>
    </demographics>
</identification>

```

```

<accessibility>
  <language>
    <typename>
      <tysource sourcetype="imsdefault"/>
      <tyvalue>English</tyvalue>
    </typename>
    <proficiency profmode="OralSpeak">Good</proficiency>
    <proficiency profmode="OralComp">Excellent</proficiency>
    <proficiency profmode="Read">Good</proficiency>
    <proficiency profmode="Write">Good</proficiency>
  </language>
  <preference>
    <typename>
      <tysource sourcetype="imsdefault"/>
      <tyvalue>InputTech</tyvalue>
    </typename>
    <prefcode>Large Font Display Devices</prefcode>
  </preference>
</accessibility>
<interest>
  <typename>
    <tysource sourcetype="imsdefault"/>
    <tyvalue>Recreational</tyvalue>
  </typename>
  <contenttype>
    <referential>
      <indexid>interest_01</indexid>
    </referential>
  </contenttype>
  <product>
    <typename>
      <tysource sourcetype="imsdefault"/>
      <tyvalue>Portfolio</tyvalue>
    </typename>
    <contenttype>
      <referential>
        <indexid>product_01</indexid>
      </referential>
    </contenttype>
    <date>
      <typename>
        <tysource sourcetype="imsdefault"/>
        <tyvalue>Create</tyvalue>
      </typename>
      <datetime>1928:10:21</datetime>
    </date>
    <description>
      <short>A picture of the violin</short>
      <full>
        <media mediamode="Image" mimetype="image/gif"
          contentrefreftype="uri">sh/violin.gif
        </media>
      </full>
    </description>
  </product>
  <description>
    <short>Music - playing the violin</short>
  </description>
</interest>
</learnerinformation>

```

- **UserRegistrationInfo:** this object encapsulates all information required for an user registration, for further details please see AXMEDIS-DE3-1-2H-AXFW-Spec-(Protection-Account)-Part-H, Section 2.2.3

8.4 Temporal diagrams & related GUI aspects

For a better understanding of the AXMEDIS Mobile Application we report below the main use cases. They are grouped into four sections: Administrative Use Cases, User System Access Use Cases, Content Fruition Use Cases and User Data Management Use Cases. For each Use Case there is a short informal descriptive

presentation and the UML related temporal diagram; for some use cases a sketch of possible user interface is also reported.

8.4.1 Administrative Use Cases

The following Use Cases are the ones that represent administrative functionalities (domain registration, content preparing, definition of content retrieving criteria, supported device profiles management, users data management). These functionalities are reserved to a “special” user, the Mobile Admin, who has more rights and privileges than “normal” users have.

8.4.1.1 Domain registration

One of the first operations the Mobile Admin has to perform for a correct functioning of the AXMEDIS Mobile Application is the Domain Registration. The Mobile Admin accesses to a reserved section of the Portal and gives a new domain identifier. The Mobile Front End forwards the identifier to the Mobile Back End, which in turn forwards it to the Domain Management module. This one prepares a “Domain object” that will be passed to the PMS Domain through the PMS Communication module in order to register the newly created domain within the AXMEDIS system. The following diagram represents the steps needed to create and register a new domain.

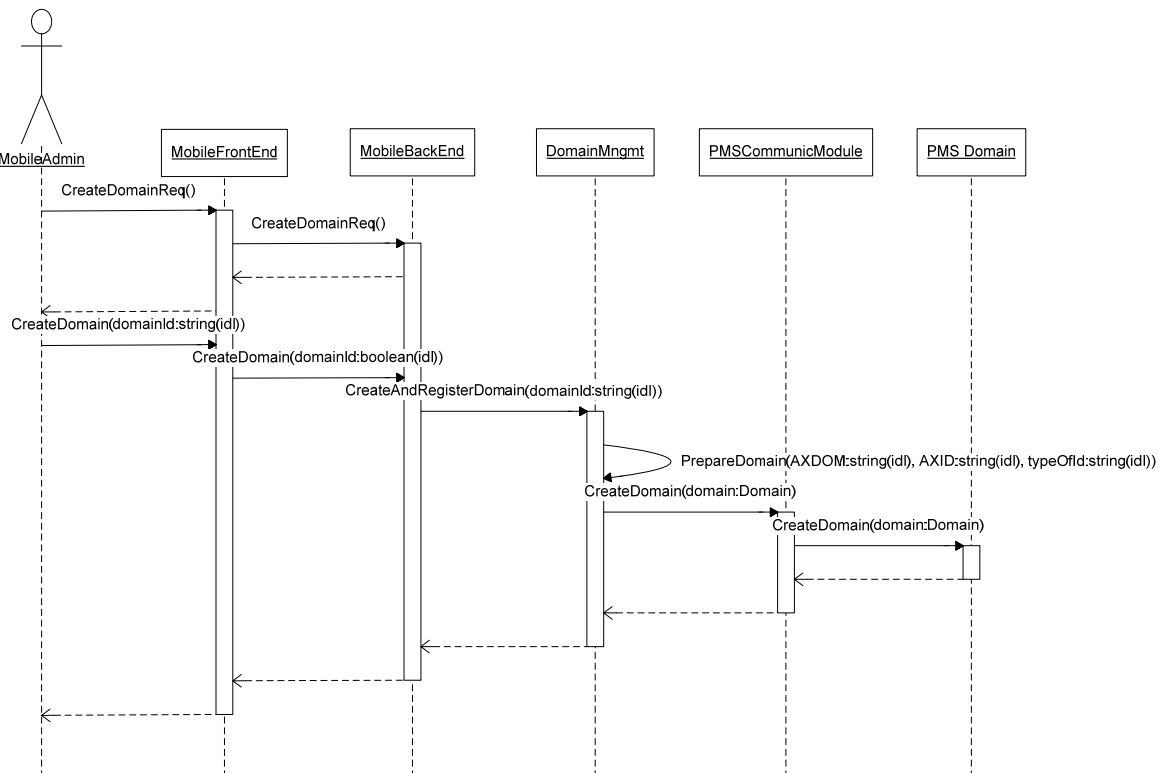


Figure 17. Domain registration Use Case.

8.4.1.2 Content Preparing

In preparing contents for Mobile we have to distinguish an authoring and a delivery phase. In the authoring phase, content for mobile is created: an AXMEDIS object is packed into an IMS Content Package, where an item refers to a resource that is the AXMEDIS object. Conceptually it is a very simple process: like a letter is put into an envelope before sending it, here an AXMEDIS object is encapsulated into an IMS Content Package without opening or modifying it in any way. In an advanced phase of development, some kind of SCORM tracking can also be added to the “envelope” whenever specific need for content tracking content

DE9.5.4 Integrated Prototype of content Production and Distribution on-demand for Mobile Phones, and new generation of PDAs

access may turn of interest (for example in specifically designed or supported eLearning delivery processes). Later, in the delivery phase, contents can be played through a dedicated AXMEDIS player.

Below the sequence diagram for the content Preparing is reported. For enhancing readability, it has been split into two parts: before the user selects some contents and the categories to which those contents will belong. Then he selects some devices for which the contents will be adapted and then “envelopes” (in the sense we saw before) and publishes to Lobster.

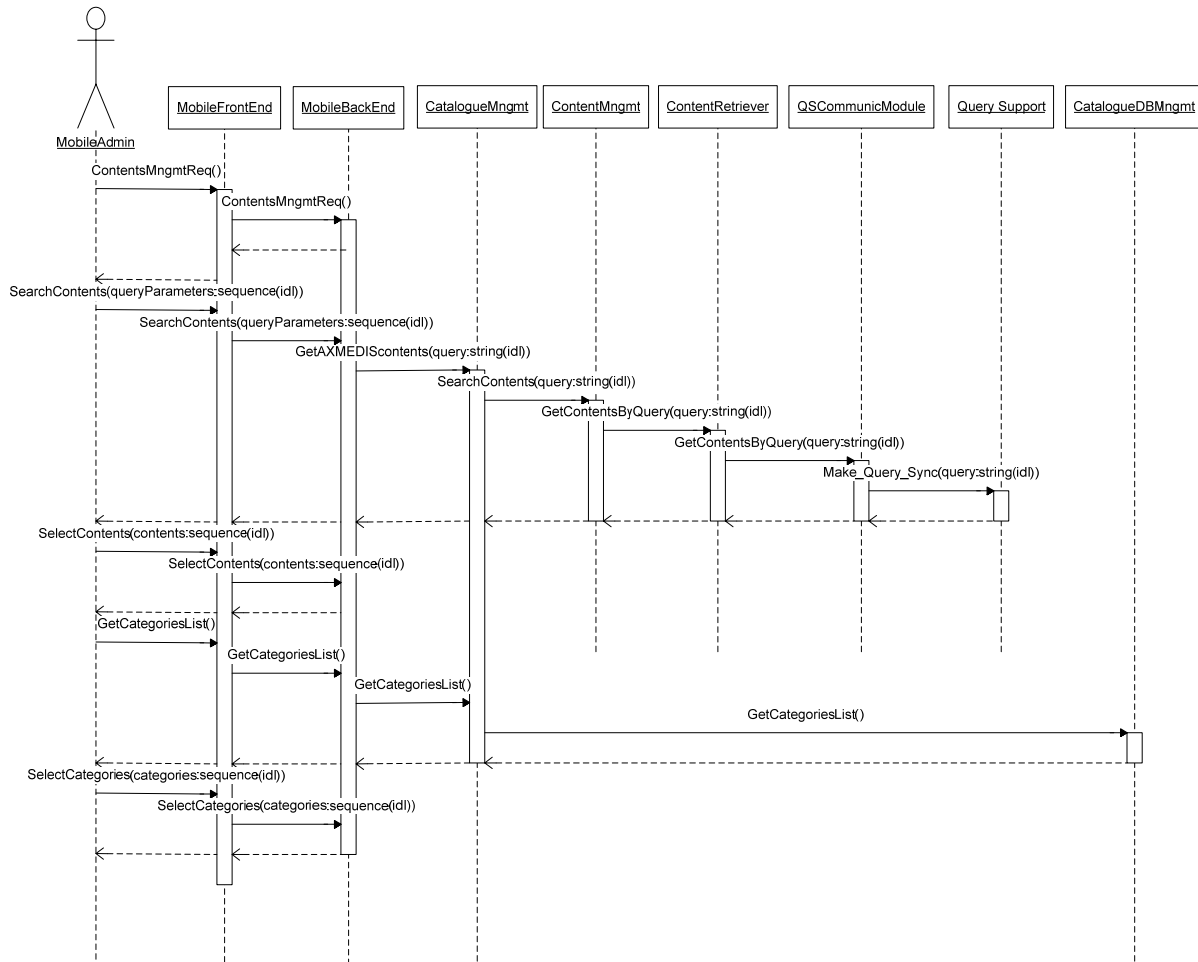


Figure 18a. Content Preparing Use Case - first phase.

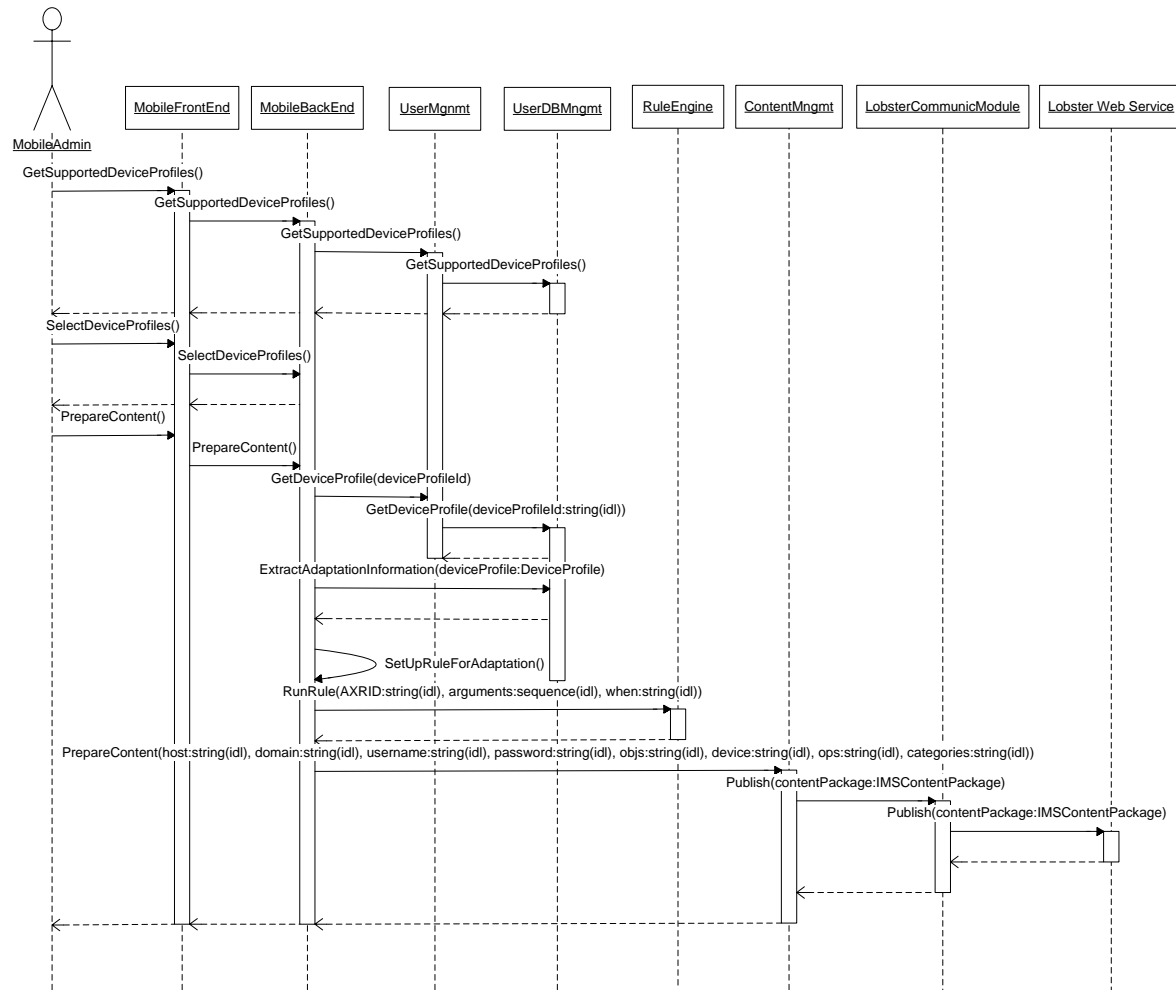


Figure 18b. Content Preparing Use Case - second phase.

8.4.1.3 Content Retrieving Criteria Management

The Mobile Admin is responsible for the definition of content retrieving criteria. S/He can define catalogue categories, which represent possible areas of interest, such as art, science, history and so on. For each category s/he assigns a category name and defines related retrieving query. Later, users will be prompted the categories list: choosing one of these categories the associated query will be activated and the contents will be retrieved through the predefined criteria.

In order to define the categories available to users, the Mobile Admin first defines some categories and then chooses which ones are active (that means visible by users). By default, a new created category is not active; it has to be explicitly activated by the Mobile Admin. Of course, categories can also be deleted from system.

9.3.1.3.1 Content Retrieving Criteria Definition

The Mobile Admin interacts with the Front-End which forwards requests to the Back End. The Criteria definition query form is retrieved and filled, inserted data are checked by the Mobile Back-End and forwarded to the Catalogue Management module, which interacts with a proper Database Management module in order to store queries. A category requires a unique key within the whole categories set since it is used to retrieve corresponding query. The Mobile Admin is also requested to give translation of category

name for each language supported by the system since users won't see the category key but a category name properly translated.

VARIATION: the Catalogue Management module could cache categories queries in order to avoid communication operations with database to retrieve them later.

Temporal diagram for content retrieving criteria definition use case is presented here below, together with a sketch of a possible user interface to insert category query parameters..

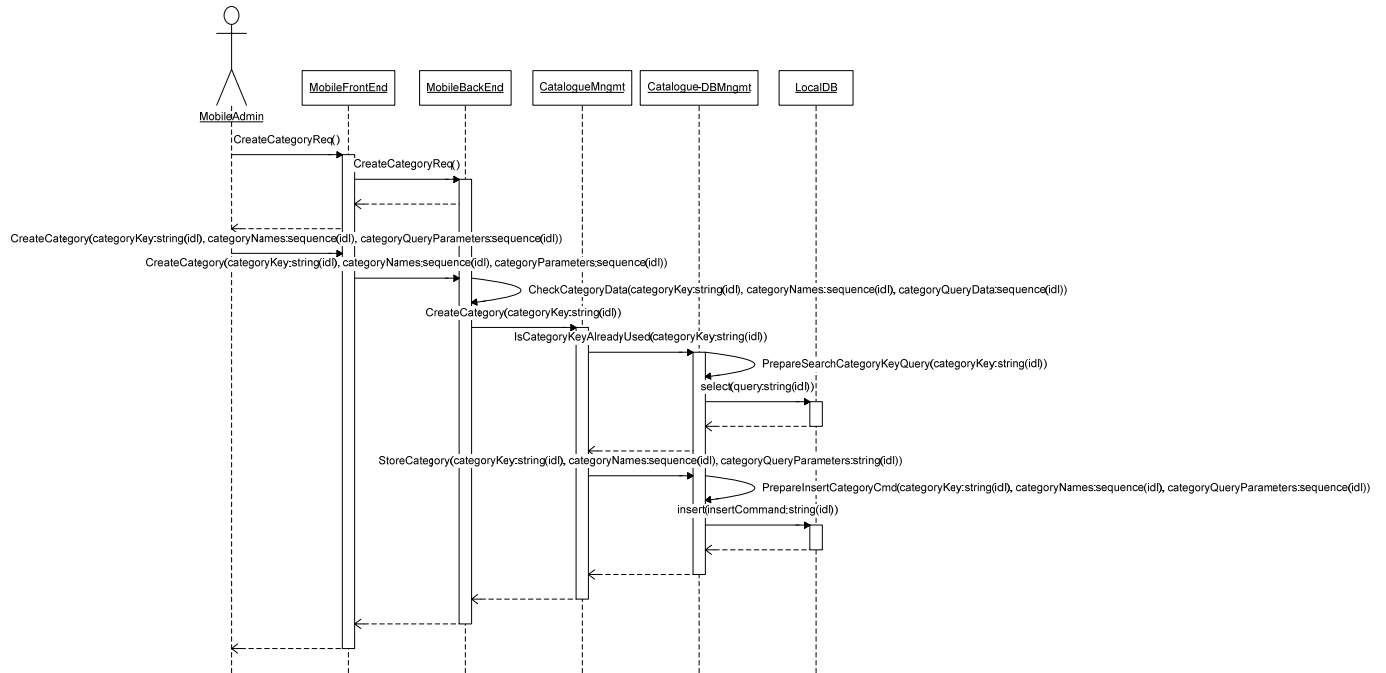
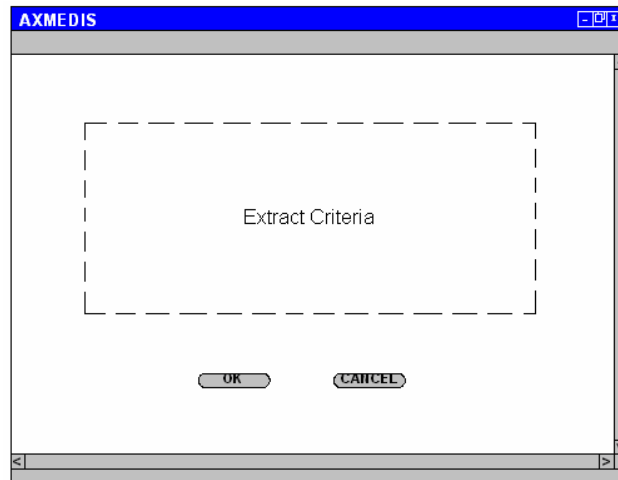


Figure 149. Content Retrieving Criteria Definition Use Case.



9.3.1.3.2 Content Retrieving Criteria Selection

The Mobile Administrator can decide which of the categories previously defined and stored into the system are actually seen by users. The Mobile Administrator asks to the Front End the list of the previously defined categories. The request is forwarded to the Back End and Catalogue Management modules; this latter one retrieves from database the list of categories and their current status (active/ not active). The generated List is shown to the Mobile Admin that chooses whether updating or not categories status and then sends confirmation (whenever needed). The Categories status (active/not active) is then updated in the database. Temporal diagram for this use case is presented hereafter.

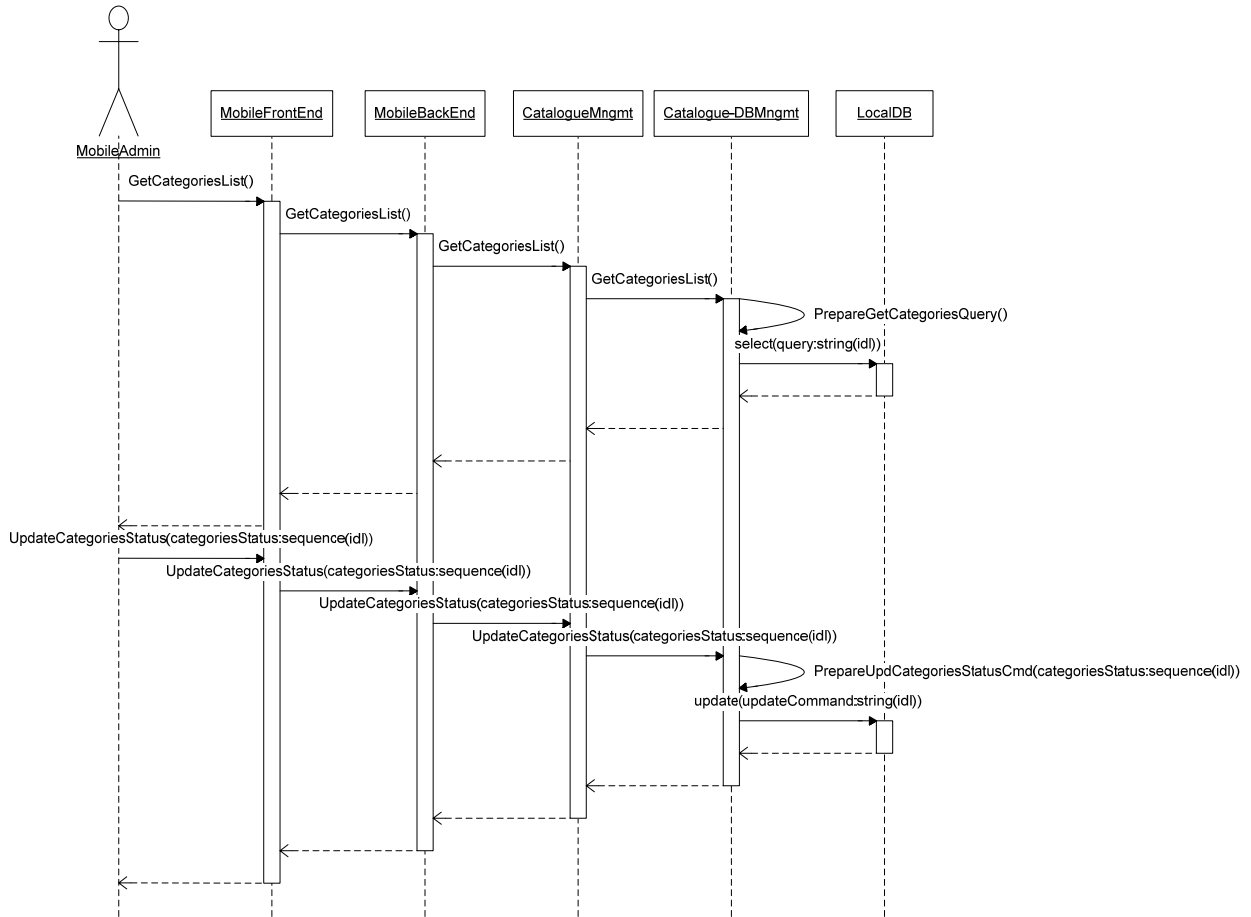


Figure 20. Content Retrieving Criteria Selection Use Case.

9.3.1.3.3 Content Retrieving Criteria Removing

The Mobile Administrator can decide to remove content selection criteria previously stored into the system. Through the Front End and Back End modules s/he can retrieve the list of inserted criteria from the database and select the ones to delete. Temporal diagram for content retrieving criteria removing is reported hereafter

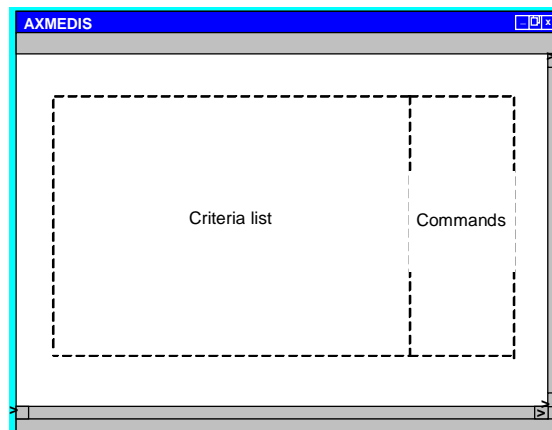


Figure 21. Content Retrieving Criteria Removing possible user interface.

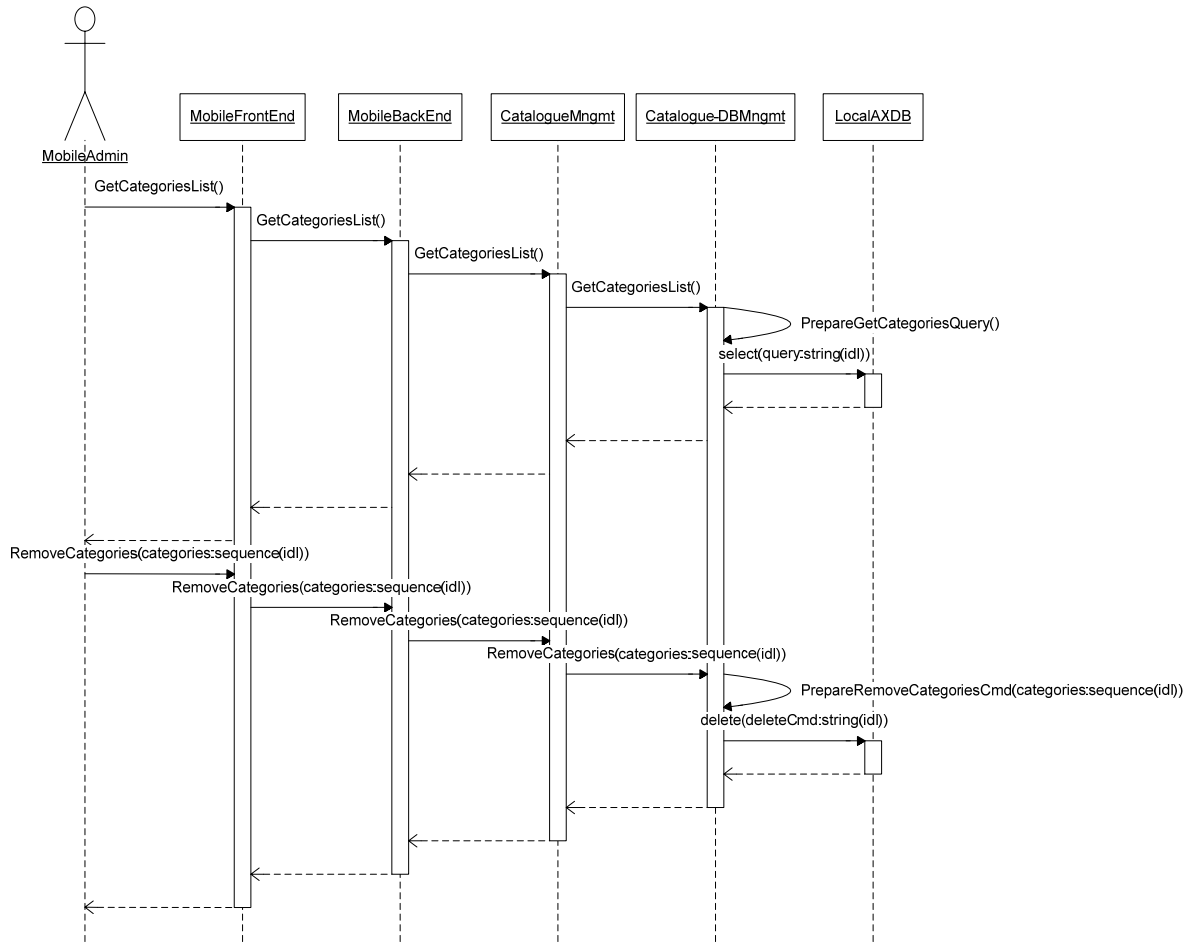


Figure 22. Content Retrieving Criteria Removing Use Case.

8.4.1.4 Supported device profiles management

The Mobile Administrator is also responsible for managing supported device profiles. At any time s/he can insert a new supported device profile within the system or remove profiles previously stored. The related use cases are detailed hereafter.

9.3.1.4.1 Supported device profile adding

In order to add a profile for a newly supported device, the Mobile Administrator requests insertion to the Mobile Front End, which in turn requests it to Mobile Back End. The Mobile Administrator fills all required form fields and submits the insertion request to the Front End. The Request is forwarded to the Back End, which performs basic data checking including one with the User Management. This latter module communicates to the Device Profile Management, which creates a Device Profile object from inserted data and asks for inserting it into local database. Objects are adapted for new device profile also. Temporal diagram for adding a supported device profile is shown hereafter.

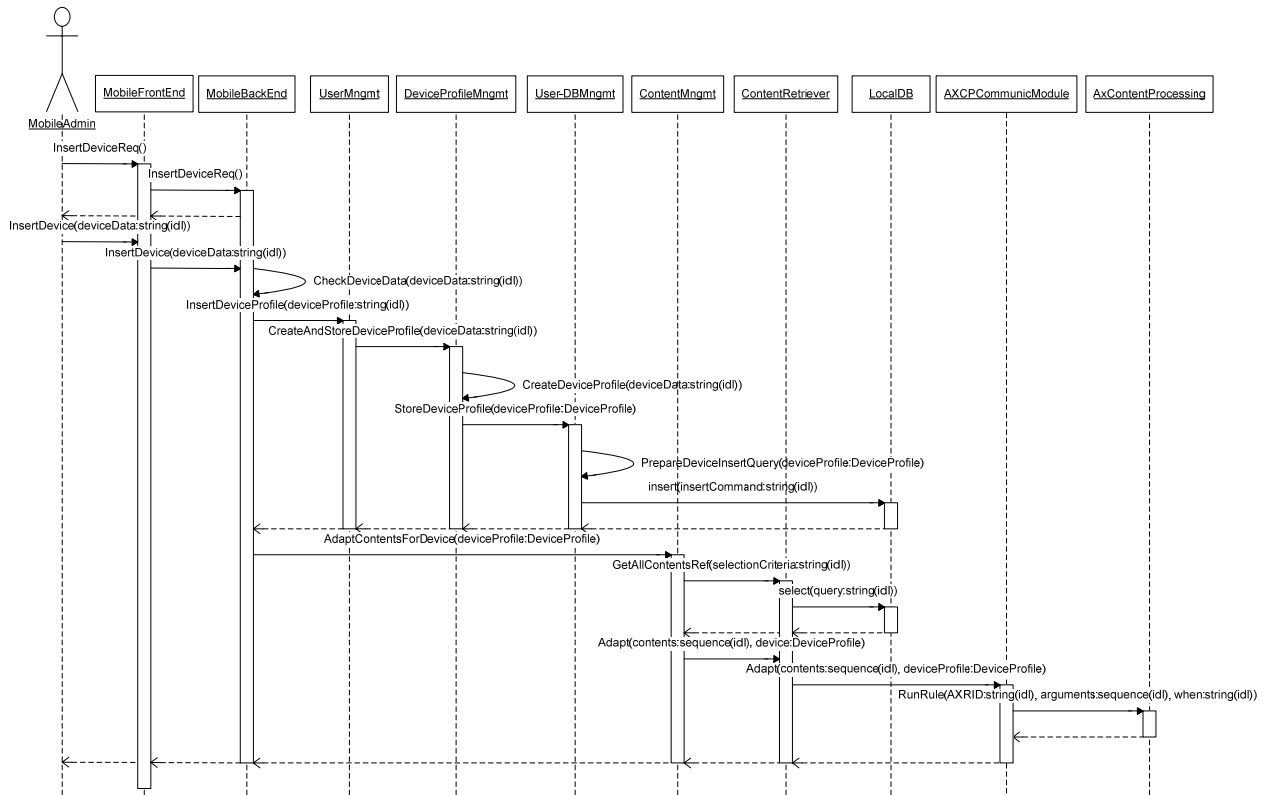


Figure 23. Adding supported device profile Use Case.

9.3.1.4.2 Supported device profile removing

At some time the Mobile Administrator can decide/need to remove some of the supported device profiles previously defined. So s/he asks for the list of supported device profiles, chooses among them and requires deletion. Some kind of marking is required for contents associated to the removed device profiles. This is specifically needed to avoid loss of content and yet prevent useless computational effort during search and retrieval phases related to content offering preparation. Method calls for device profiles removing are reported in diagram after the possible aspect of the user interface.

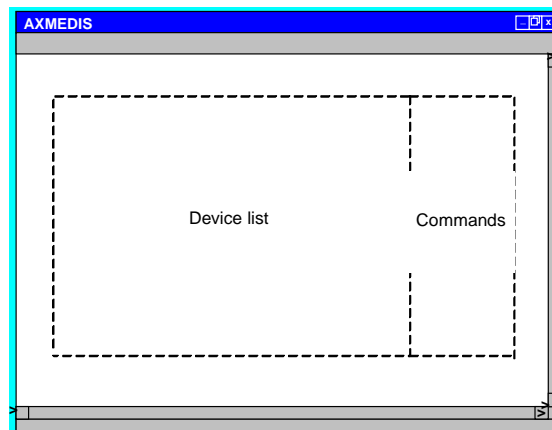


Figure 24. Sketch of possible Device Management Operation Page.

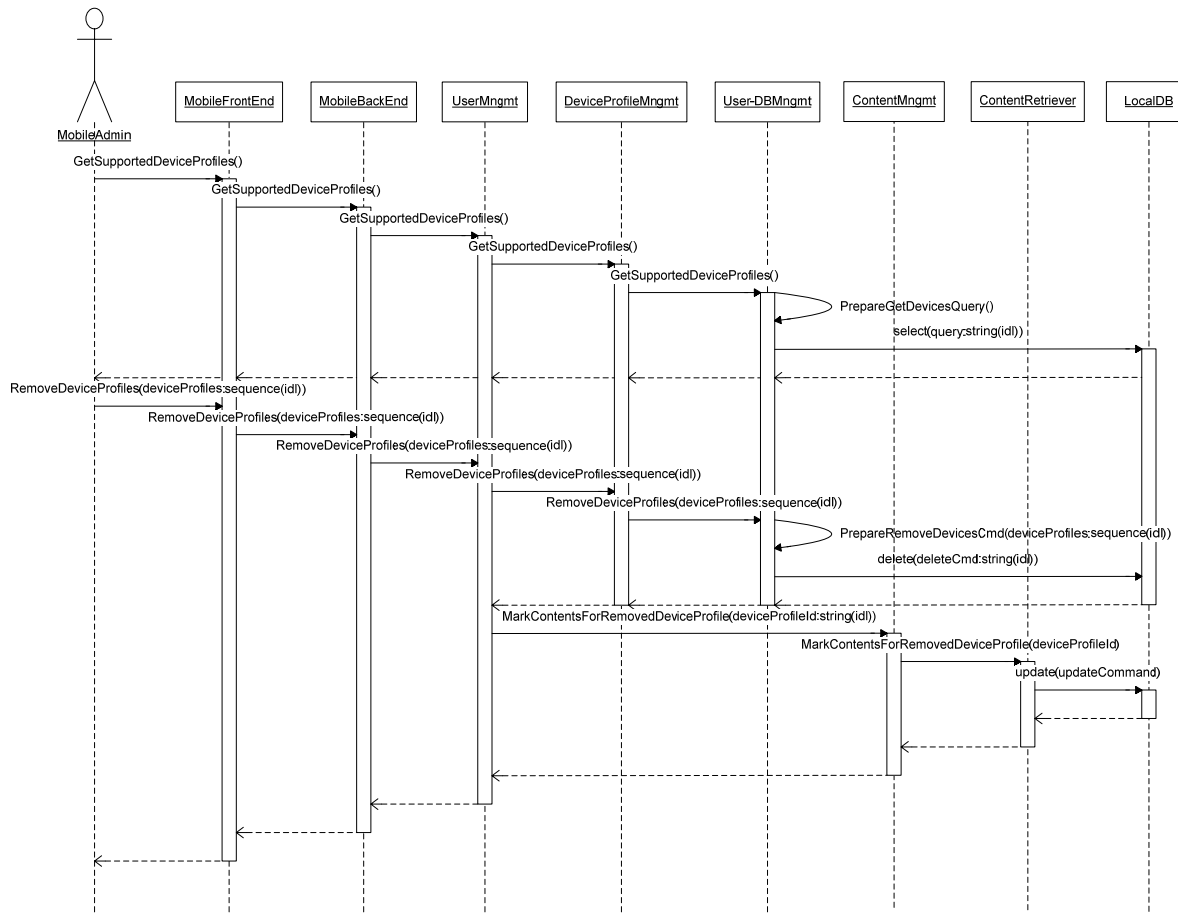


Figure 25. Removing supported device profile Use Case.

8.4.1.5 User data management

One of the Mobile Administrator several tasks is the user data management. Thanks to his rights the Mobile Administrator can register a new user, update or delete an existing user or change users' roles. All these use cases are reported and detailed below. Next Figure shows a sketch of a possible starting page for user data management.

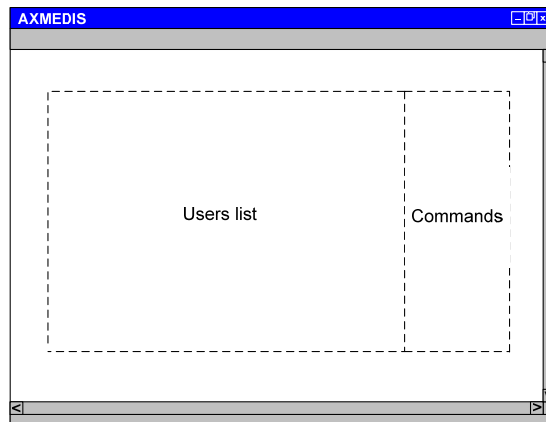


Figure 26. Sketch of Users Data Management Starting Page.

9.3.1.5.1 User registration by administrator

As already briefly mentioned the Mobile Administrator can register a new user into the AXMEDIS system thanks to his grants. Related use case is essentially the same as the one for user self-made registration, later explained into section 9.3.2.1, except for the fact that the new user to register is not the one performing operation. For this reason temporal diagram is not reported here. Briefly we can anticipate that the Mobile Admin has to retrieve the registration form, fill-in at least all mandatory data and forward them to AXCS module. User registration by Mobile Admin could be activated, for example, via e-mail request coming from the user being registered.

9.3.1.5.2 User update by administrator

The Mobile Administrator can update users' data. S/He has to provide the system with the identifier of the user to update, modify and confirm provided data in order to make the update operative. Temporal diagram for this use case is essentially the same as the one for "data update" performed by the user himself later detailed, so it is not reported here. Please see Section 9.4.1 for further information.

9.3.1.5.3 User remove by administrator

To remove a user from the system, the Mobile Admin has to request the AXCS to update "finalRegDeadline" parameter into user's registration data, exploiting an update registration request. Furthermore User's data have also to be deleted from local database. An email is sent to the user to notify deletion. Temporal diagram for this use case is reported hereafter.

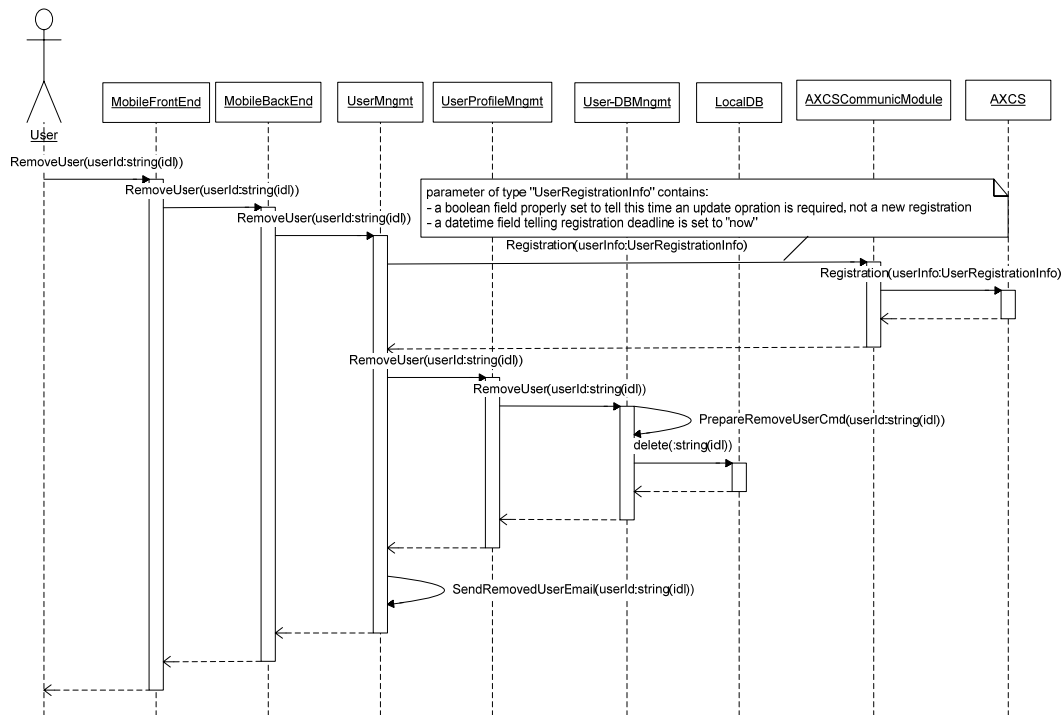


Figure 27. User remove by Administrator Use Case.

9.3.1.5.4 User roles management

Within the Mobile Application, users have roles that correspond to different privileges within the system, that is, users are authorized to different operations according to their role. Basically two roles are foreseen in this case: the "user" and the "admin" role. First role allows to access application and contents, while the second role allows performing administrative functions. In future new roles could be added in order to manage a finer granularity of allowed actions. This latter point may depend on the results of the initial

experimentation phase. One of the Mobile Administrator tasks is the “user roles management”. By default any new registered user is a common user. The Mobile Administrator can change a user’s role, for example assigning also an administrator role, in fact users can have more than one role at the time. Temporal diagram for user roles management is reported hereafter.

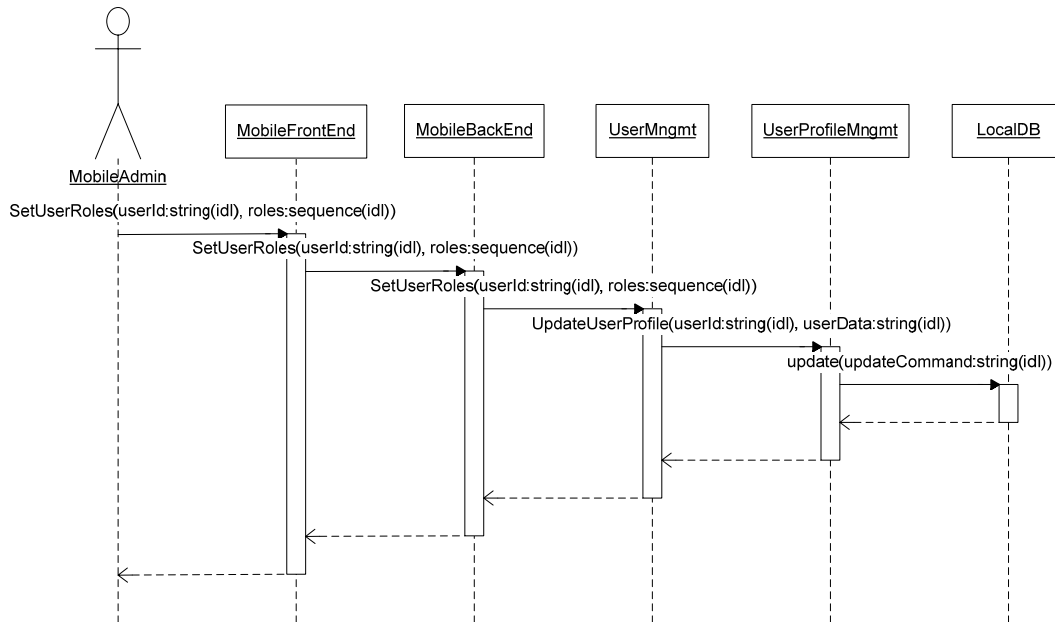


Figure 15. User Roles Management Use Case.

8.4.2 User System Access Use Cases

When a user wishes to use AXMEDIS Mobile Application, s/he has to register first, to certificate own mobile device, and to download one or more plug-ins (according to device and needs). When a user registers, s/he also automatically logs in, but next time if s/he wants to access Mobile Application, has to explicitly login. At any time, s/he can change current visualization language choosing from the list of supported languages.

8.4.2.1 User registration

The User has to register before being able to use AXMEDIS Mobile Application. S/He has to provide some personal information. Once registered into the system s/he will receive an e-mail containing own unique AXMEDIS user identifier. In more details, s/he has to interact with the Application Front End in order to retrieve a registration form. There are two types of data to insert: the elements that should be filled by every user wishing to register (mandatory elements) and the elements that would be very useful whenever the user decides to fill in (recommended elements). If provided data are formally corrected they are passed to User Management module, which makes a first check to verify if the user is already present into the local DB. If not, this module takes the charge to register user into AXMEDIS system by communicating registration request to the AXCS Communication module, which function is to properly format and forward requests to the AXCS. If a user registration is successful then user data are stored locally into local database. At the end of the registration process, an email is sent to the user as feedback to communicate the attributed AXMEDIS user identifier.

VARIATION: the user has already registered himself into AXMEDIS through another distributor so s/he has to register only for the Mobile Application now. Therefore the Registration form should allow inserting also an AXUID if the user already owns one. In this case communication with AXCS is limited to the notification of the new subscribed distributor if not needed at all.

Temporal diagram for registration is depicted in the following picture. A sketch of a possible user interface is also presented.

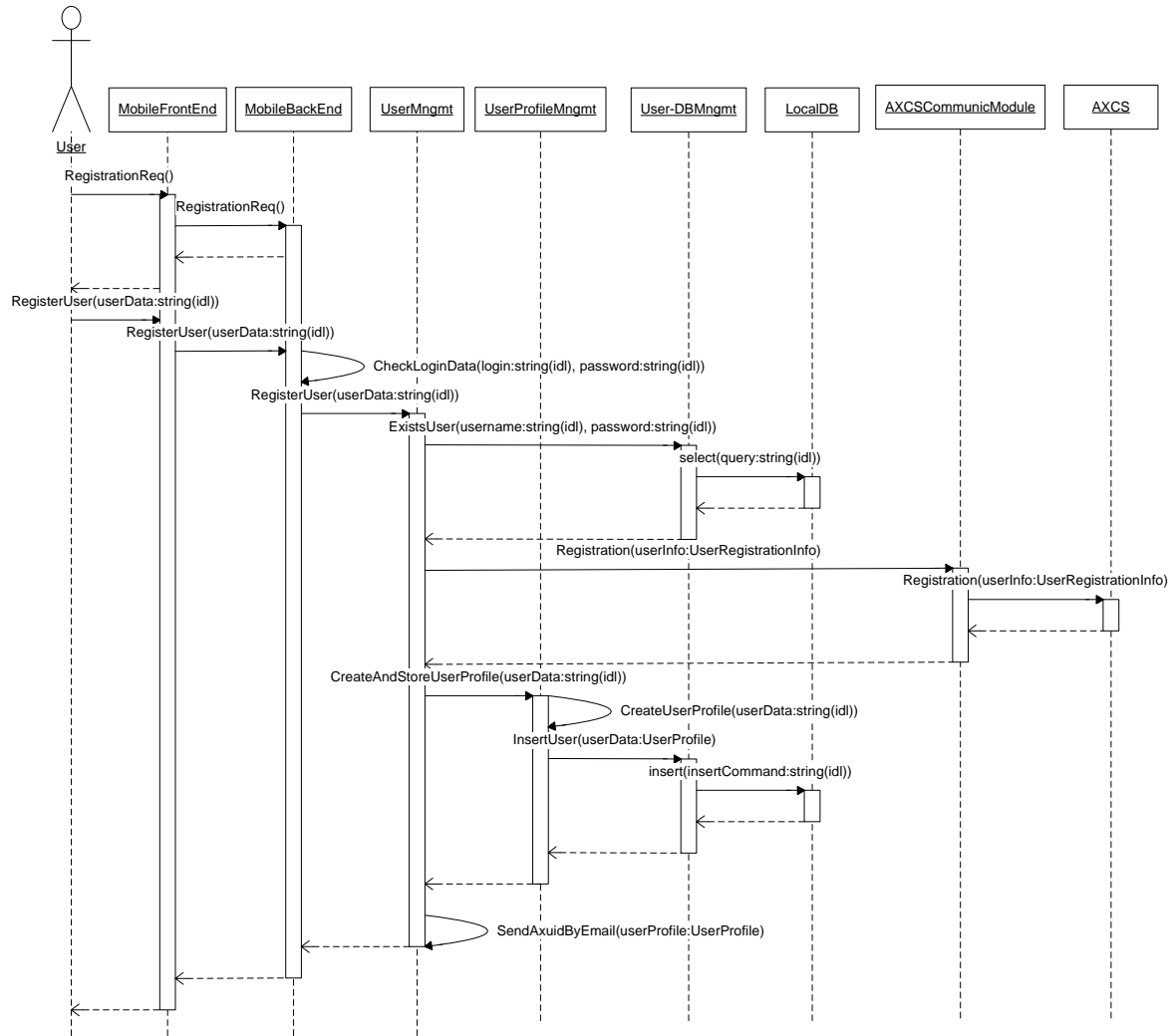


Figure 29. User Registration Use Case.

AXMEDIS
⏏ ⏏ ⏏

User Registration

Label

Input fields

OK
CANCEL

8.4.2.2 Certification of users

When a user wishes to be certified (with own device) in order to access to AXMEDIS objects, s/he has to provide (directly or indirectly – that is mediated by some mechanism) the needed information to the application that forwards user and tool information to the PMS, this latter performs all necessary checks and operations. Then, in case of positive result, the PMS returns an acknowledge to the requesting application. A proper feedback is the given to the user. It is worth recalling that an AXMEDIS tool is certified for an AXMEDIS user on an AXMEDIS device. Tool-user-device become linked once certified. Certification diagrams are not reported here since they have already been presented in Part H, Section 2.3.2 where the Certification process is detailed.

8.4.2.3 Client application download

Users have to download an AXMEDIS Client Player on their devices in order to be able to use AXMEDIS contents. In order to get an AXMEDIS Client Player, a user has to be already registered and logged in the system. Then s/he requires download instructions to the Mobile Application Front End, which in turn requires instructions to the Mobile Application Back End. Once instructions are returned to the user, s/he can connect to “the network” (here this has to be regarded in the wider sense of the term) using his personal device and download the AXMEDIS client application. Temporal diagram for Client Application download is reported here below.

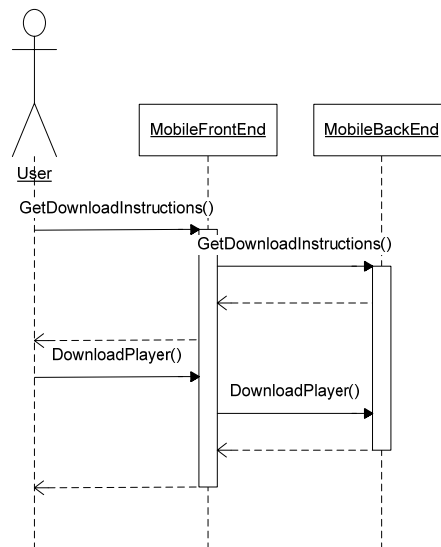


Figure 30. Client Application Download Use Case.

8.4.2.4 User login

When a user wishes to use AXMEDIS Mobile functionalities, s/he has to login first. S/He accesses the login form and provides login and password chosen at registration time. The Mobile Front End will pass user data to the Mobile Back End, which performs some basic checks and then further forwards received data to the User Management module that will request the User Database Management module to verify user credentials: if someone associated to given login and password already exists, then access to the Mobile Portal is granted. Temporal diagram for this use case is reported hereafter along with a sketch of a possible user interface.

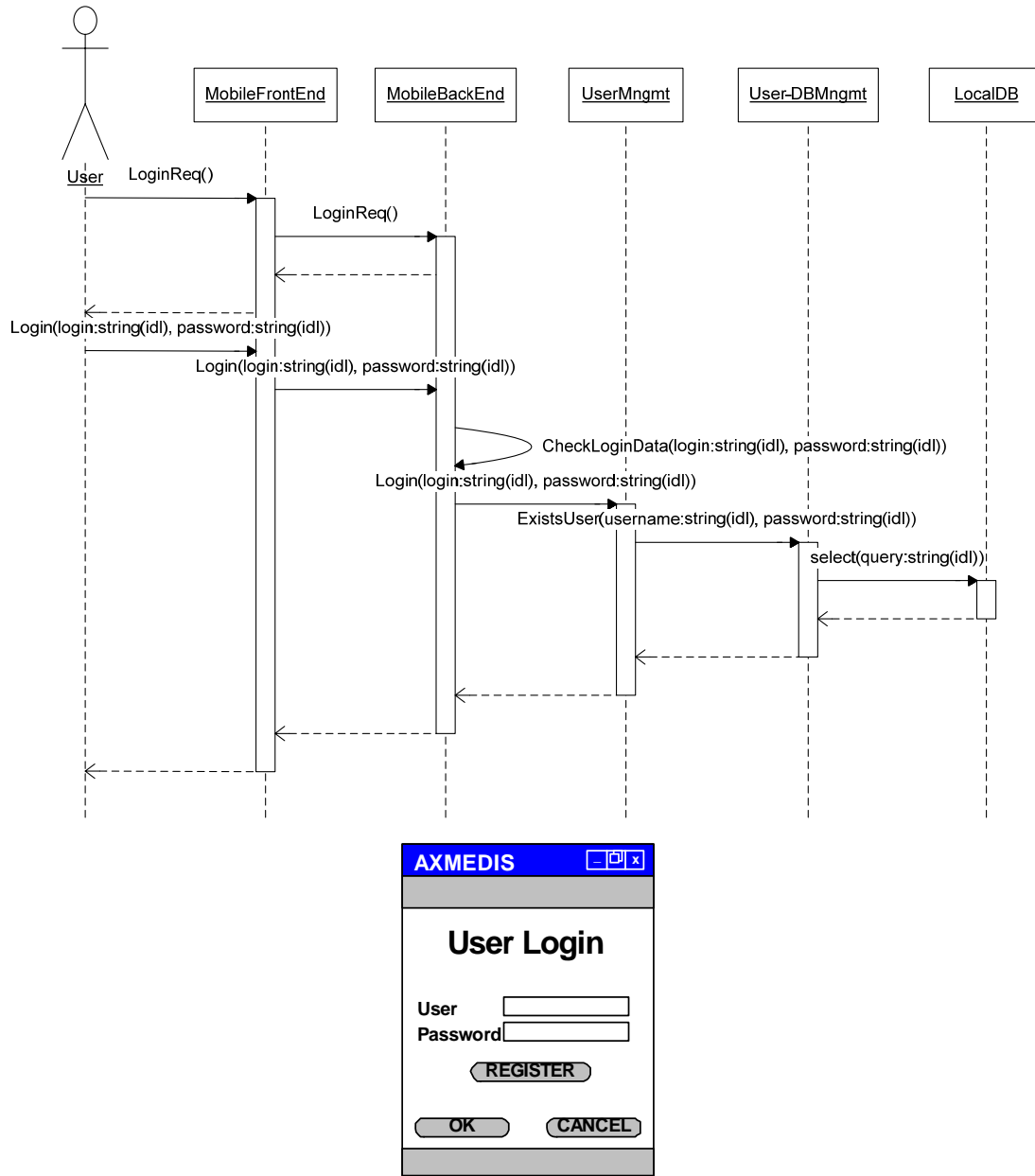


Figure 16. User Login Use Case.

8.4.2.5 User interface language selection

At any time a user can change the current language used by the application. In order to do so, s/he has to interact with the Front End, which forwards the requested change to the Back End. In reply the supported languages list is displayed, at this point the user chooses a language and from this point onwards all pages will be visualized using the selected language. If a user is already logged then the currently selected language is updated also into user's profile, elsewhere language changes only for visualization. The following diagram represents the temporal use case for language selection; we assume user is already logged when performing language selection so that actually a user's data update is performed also on the local database. A sketch of a possible user interface is also presented.

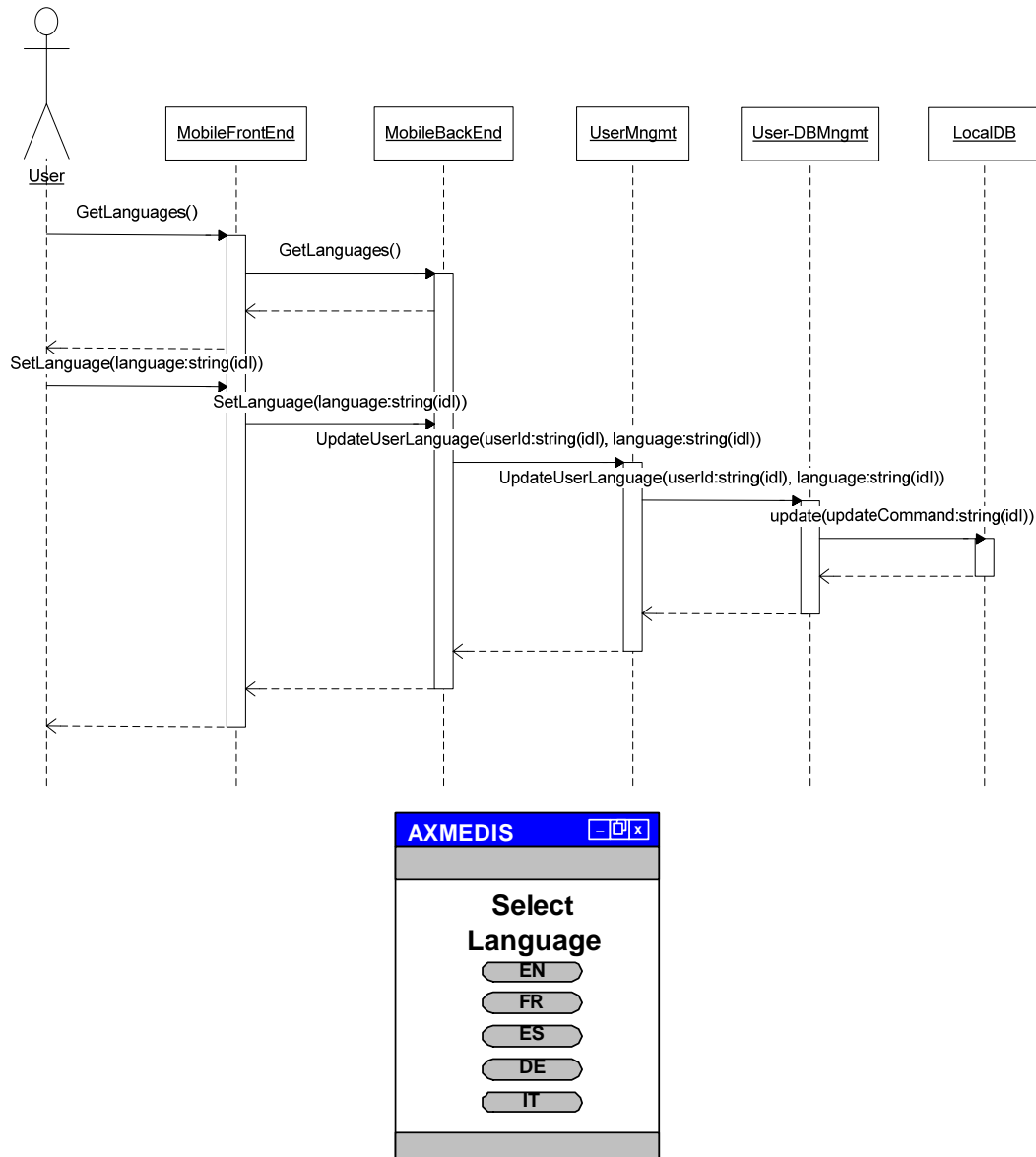


Figure 172. Language Selection Use Case for logged user.

8.4.3 Content Fruition Use Cases

From AXMEDIS Mobile Application a user can load and browse the set of available contents. These are grouped by “areas of interest” hereby called “categories”. The whole set of available contents is referred to as “catalogue”. A User can select contents of interests and get more information about them; here is also assumed that s/he can also have some kind of “preview”, view a list of available rights (where available here stands for “available for purchase”) with related prices and on such basis decide if to buy or not. If the user decides to buy, then is assumed that s/he has to pay before getting actually any real access grant to the selected content. At this point is needed to get in contact with the PMS to purchase the wanted license/s (grants), possibly based on the available PARs and distribution licenses. Contents for which a user has acquired some grants are listed into a specific section that the user can access anytime he connects.

8.4.3.1 Catalogue loading and browsing

When a user wants to access contents, s/he has to reach the Portal Catalogue section. Here s/he gets the list of foreseen content categories defined by the Mobile Admin; once s/he chooses a category, related contents

are retrieved. User requests are processed via the Front End, which in turn communicate with the Back End. Requests for content categories are dispatched to the Catalogue Management module and categories are retrieved from the local database. When a user chooses a category then contents for this category are retrieved both in the Lobster and via Query Support (and related communication modules).

VARIATION1: the Catalogue Management module could cache locally queries associated to categories in order to avoid retrieving them from database every time they are needed.

VARIATION2: in order to achieve better performances, the Catalogue Management could cache contents data in order to be able to answer directly to further requests and to avoid calling the Database Management and the Content Retriever modules functionalities again later. This certainly applies to non AXMEDIS data, on the other hand for AXMEDIS data references used for retrieval in local AXDB will be cashed in order to speed up the retrieval process as described previously.

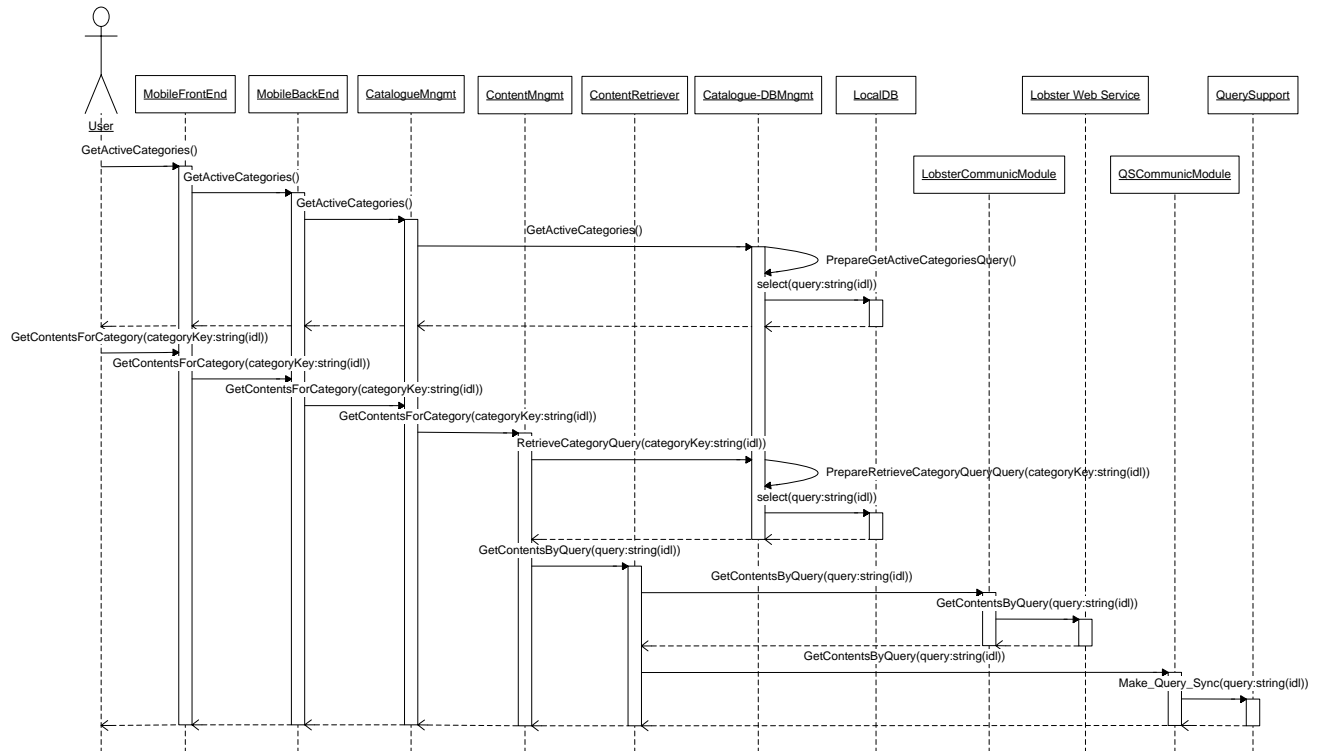
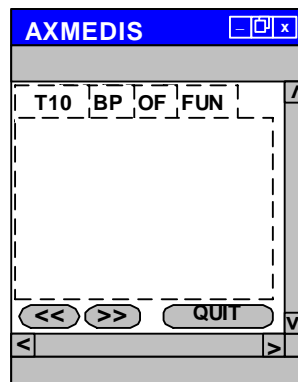


Figure 33. Catalogue Loading and Browsing Use Case.



8.4.3.2 Contents Search

An user may wish to get more contents than the ones available and listed on the Mobile Portal by categories selection. To retrieve other contents the user can exploit the Search Contents page and fill-in query form parameters. The query is then dispatched locally and remotely through the Query on Demand module. Temporal diagram for this use case is reported below.

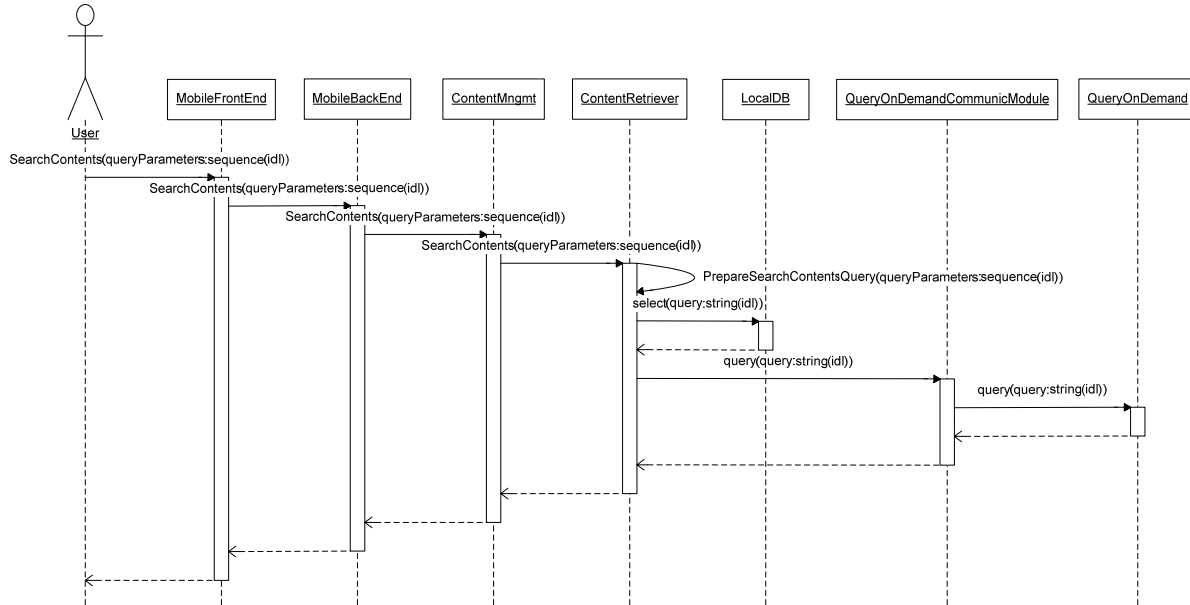


Figure 34. Contents Search Use Case.

8.4.3.3 Getting content information

When a user is interested into a content found on any Portal pages, s/he can ask for more information about that content. Through the Front and Back End the user request arrives to the Catalogue Management, which retrieves content metadata from the local database or remotely through the Query Support.

VARIATION: the Catalogue Management module already cached contents on catalogue loading. No forwarding of requests is needed towards the Content Retriever and Query Support modules.

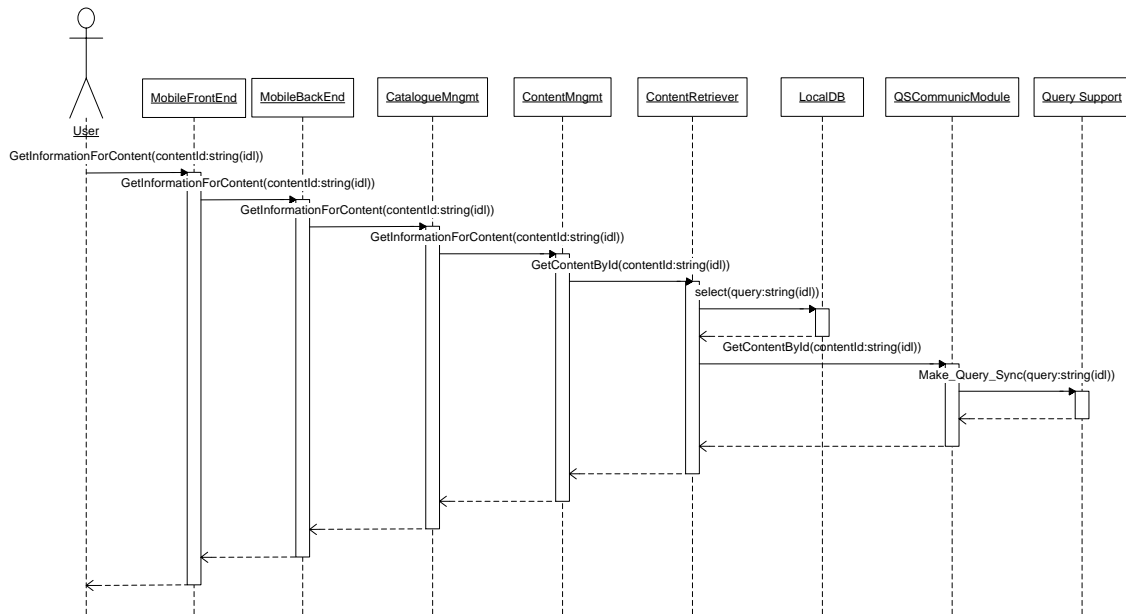


Figure 18. Getting Content Information Use Case.

8.4.3.4 Content Preview

User wants to have a good idea on what a specific content is before spending money to buy some kind of license. So s/he may want a kind of “preview” of the content, this preview will basically be a “piece” of the content to view/play/other depending on the type of the content and the possibility granted by the copyright regulations managed via the DRM. In order to get content preview the user requires it via the Front End, which invokes the AXMEDIS Client Player on the user device.

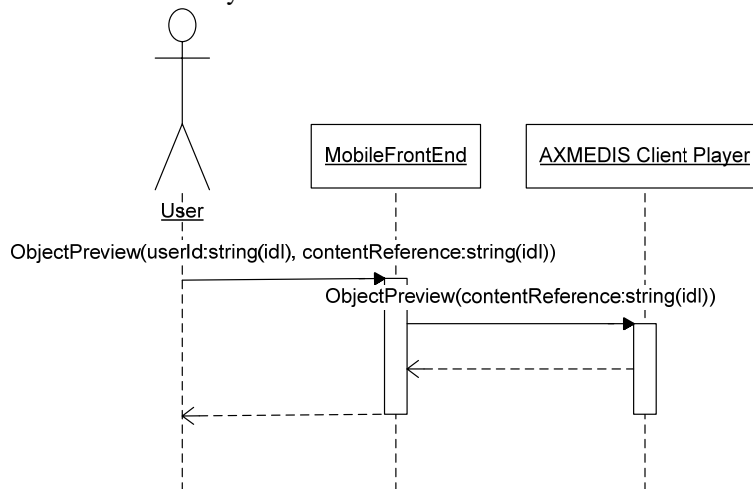


Figure 36. Content Preview Use Case.

8.4.3.5 Content Acquire

If a user wants to acquire rights to execute some actions over an object (install, uninstall, play, modify, etc...) s/he has to pay before. The set of rights that will be available to the user for purchase is a subset of the original Potential Available Rights (PAR) that depends on the following factors:

- Original PAR;
- Rights acquired by the distributor (eventually already restricted either by the rights owner at selling time or by any other distributor that is preceding the present one in the sale value chain);
- Rights that the distributor decides to present the user as available (and that have to be less or equal to the acquired ones).

Potential Available Rights (PAR) are defined by right owner who can then grant some rights combination to right purchaser. In case among acquired PAR there is the right to issue licenses or grant some rights (always a subset of the acquired ones), the new issuer can define the subset of PAR that will be available to the next actor in the chain. This brings to a nested set of restrictions that are supported by a chain of licenses each enforcing the correct set of rights and ensuring the possibility for the actor to properly use the object respecting the DRM chain. In the following example is sketched a simple sequence with a main owner, two chained distributors (for example EU and National level) and the user.

Owner

Has rights to: **{Use \wedge Distribute \wedge Modify \wedge License \wedge Grant rights}** In detail can perform the following set of actions divided by category:

Use: **{Play | Print | Store | Replicate | ...}**

Distribute: **{For free | Sell | Resell | Re-license | ...}**

Modify: **{Format | Aspect | Adapt | Translate | Aggregate | Incorporate | Split | ...}**

License: **{Use | Distribute | Modify | ...}**

Grant rights for: **{Use | Distribute | Modify | ...}**

Dist.1

Has rights to: **{Use \wedge Distribute \wedge Modify \wedge License \wedge Grant rights}** In detail can perform the following set of actions divided by category:

Acquire Use: **{Play | Print | Store | Replicate | Modify | Distribute | ...}**

Distribute: **{Sell | Resell | Re-license | ...}**

Modify: **{Format | Aspect | Adapt | Translate | Aggregate | Incorporate | Split | ...}**

Sale: **{Use | Distribute | Modify | ...}**

Dist.2

Has rights to: **{Use \wedge Distribute \wedge Modify \wedge License}** In detail can perform the following set of actions divided by category:

Acquire Use: **{Play | Print | Store | Replicate | Modify | Distribute | ...}**

Distribute: **{Sell | Resell | ...}**

Modify: **{Format | Aspect | Adapt | Translate | Aggregate | Incorporate | ...}**

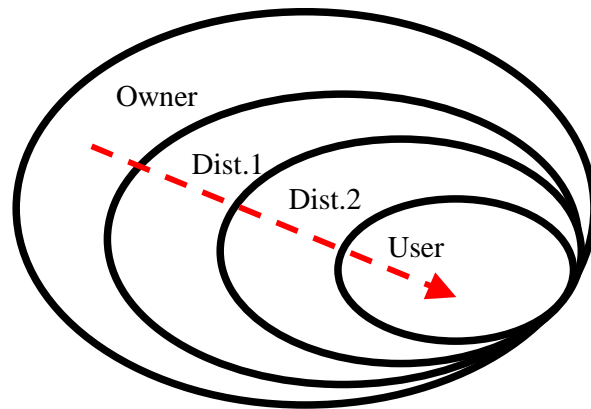
Sale: **{Use}**

User

Has rights to: **{Use}** In detail can perform the following set of actions divided by category:

Acquire Use: **{Play | Print | {Play \wedge Print} | {Play \wedge Store} | {Play \wedge Print \wedge Store}}**

What just stated is more clearly explained by the following graph:



User devoted/oriented set of potential available right could be profitably referred to as User Potential Available Rights (U-PAR), nevertheless as this has not been agreed elsewhere we will keep un using just the usual reference form (namely PAR) leaving to the reader to understand from the context to which one we are referring. Therefore in this scenario, a User chooses which rights he wants to acquire, then he has to give payment data and confirm (of course a distributor license has been requested and created before). If payment is successful, user is granted required rights for the selected content. First of all, the user has to request the Front End the PARs list for a given content through the Back End and the Catalogue Management modules. The User chooses the set of rights to acquire from the provided list then s/he is requested to provide needed payment data. Temporal diagram for content acquisition and also a sketch of a possible user interface are reported below.

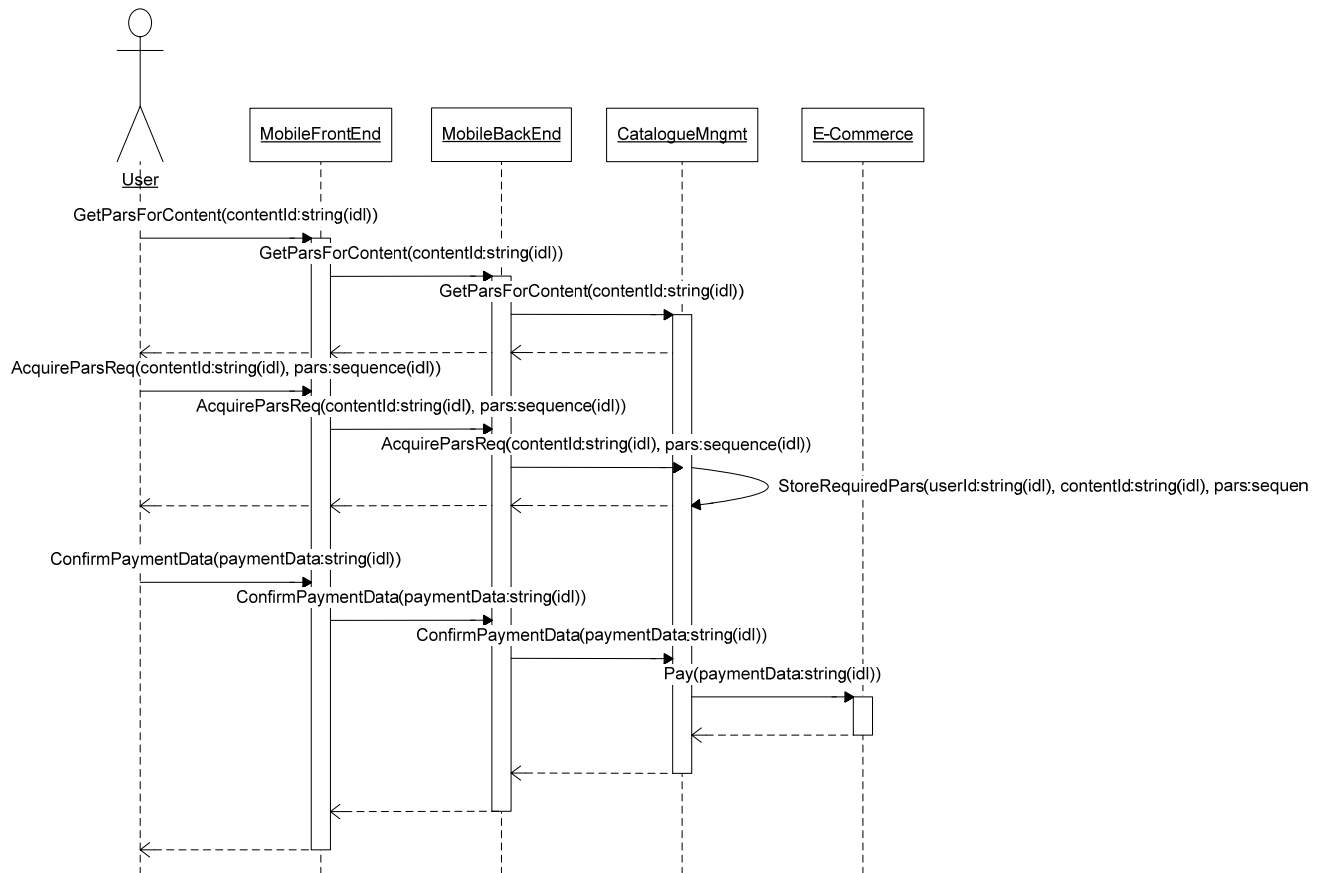
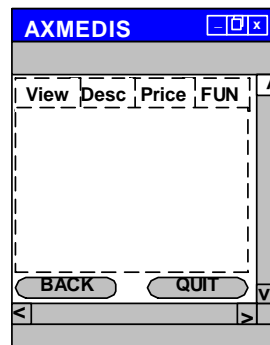


Figure 37. Content Acquire Use Case.



8.4.3.6 Content Delivery

Content is delivered to the user according to acquired license rights. The Front End interacts with the Back End which forwards requests to the Content Management and Retriever modules. Content can be retrieved either from the Lobster or the AXDB. Also User Profile and Device Profile are needed since content has to have a proper format for user device and preferences. Temporal diagram for this use case is reported below.

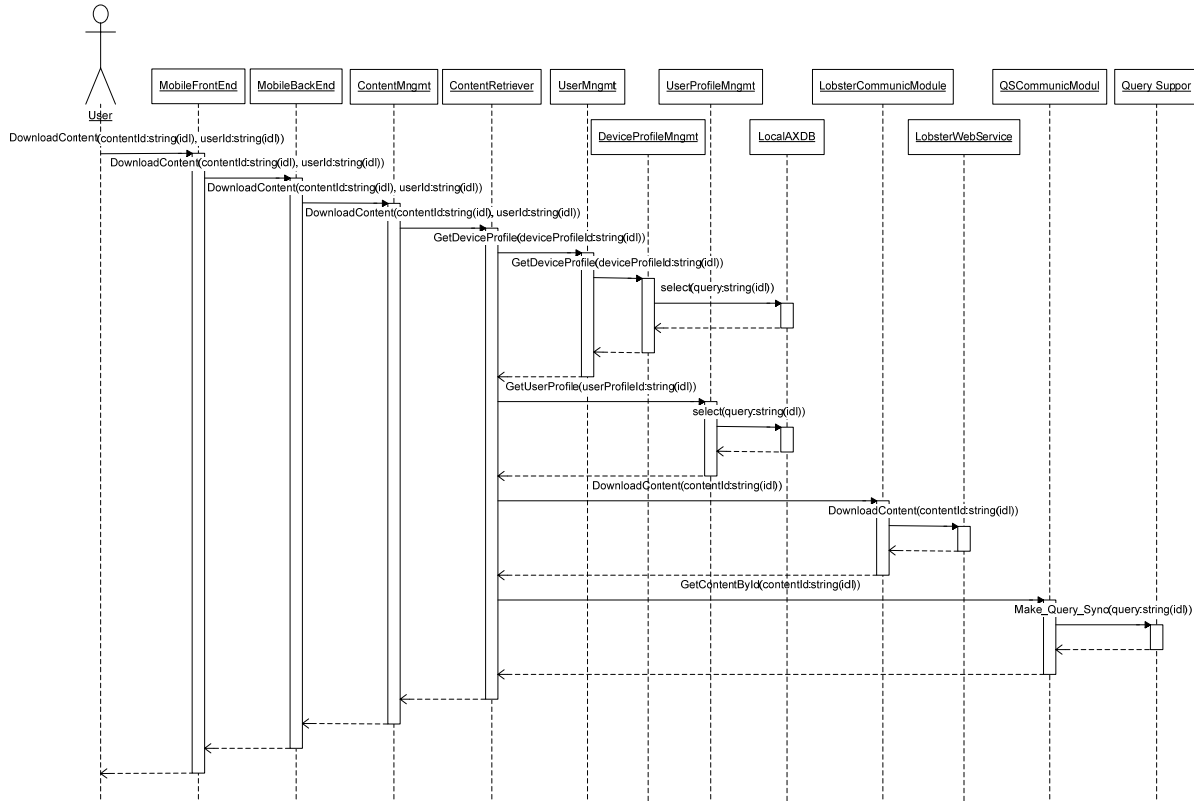


Figure 19. Content Delivery Use Case.

8.4.3.7 Content fruition

The Mobile Portal offers to users a special section where they can find the list of contents over which they have acquired a license with some rights. Once entered this section, users will simply need to click on a content title in order to play it: the AXMEDIS Player(s) present on the user device will perform all needed rights checking before playing. The PMS client embedded into the player contacts AXCS to check the user status in the system and verify the tool integrity. These operations are performed before authorising the user or the first time the tool is used. Temporal diagram for this use case is reported below.

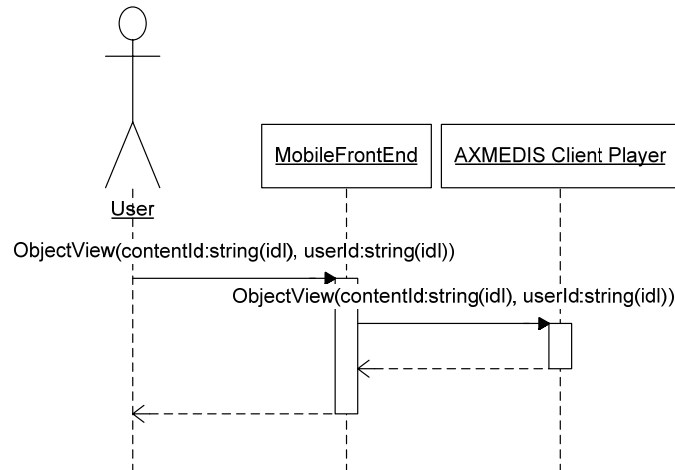


Figure 39. Content Fruition Use Case.

8.5 User Data Use Cases

At any time an user registered into the system can login and change personal data previously inserted.

8.5.1 User Data Update

An user registered into the system can decide to modify his data previously inserted at registration time. The Mobile Portal offers a specific section where the user is prompted with inserted data. The User can update and confirm changes. The User interacts with the Front End, which forwards request to the Back End, this one retrieves user data from the database through User Management and User Profile modules. Updated information is then forwarded and stored in the database. Temporal diagram for this use case is reported below.

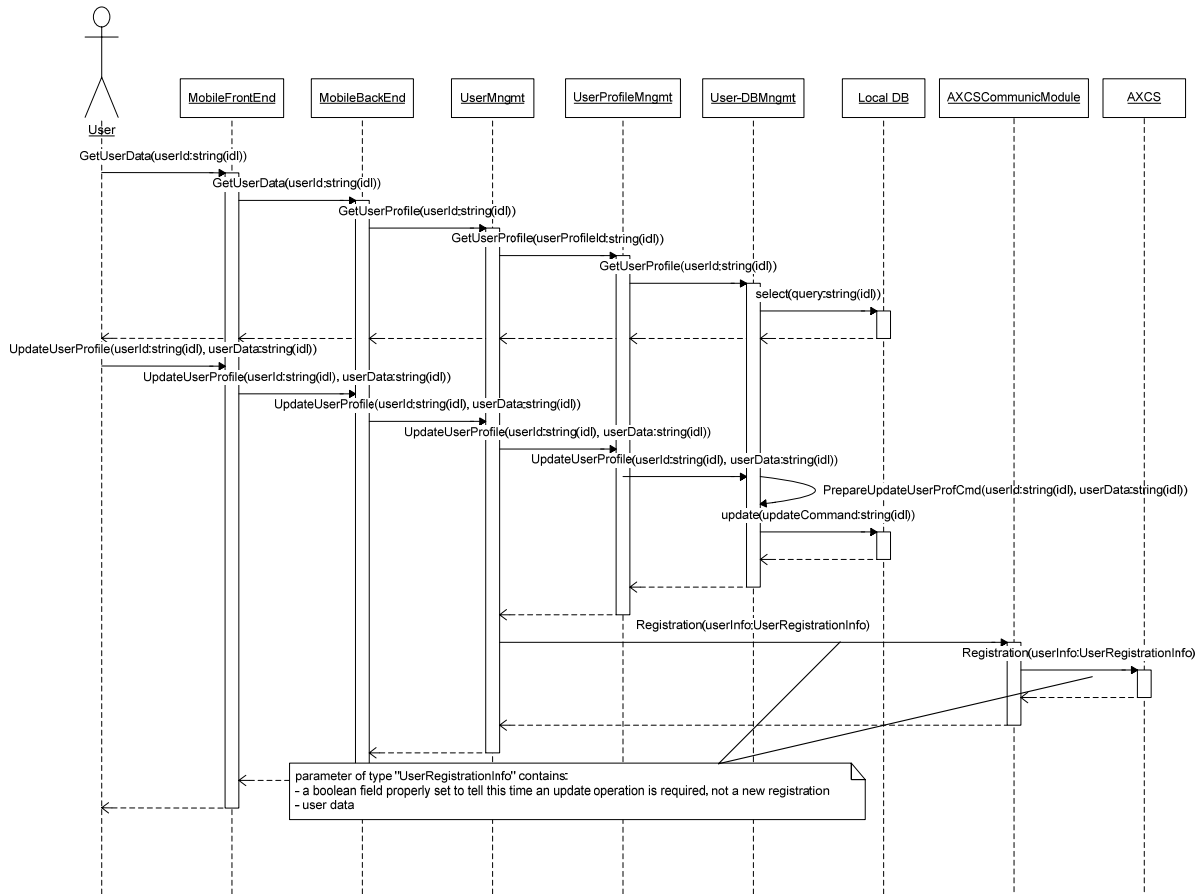


Figure 20. User Data Update Use Case.

8.6 Mobile Architecture

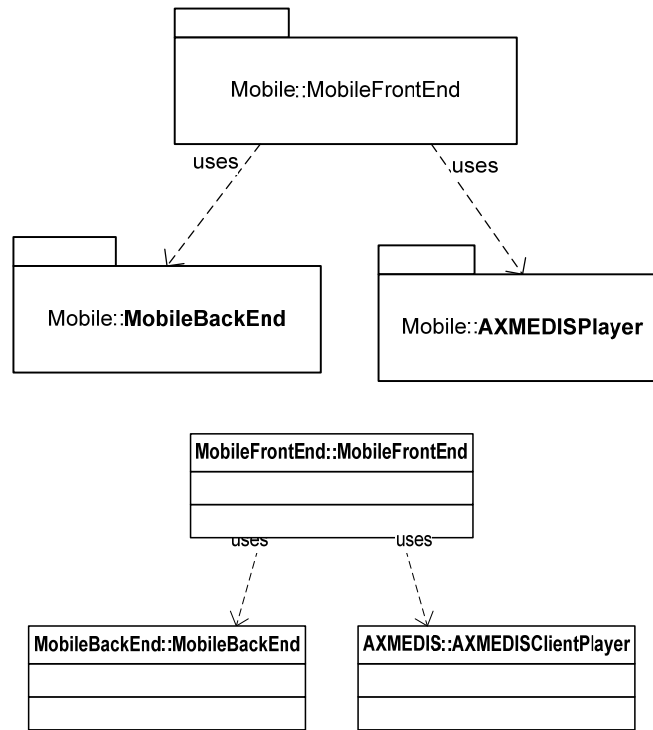
The whole Mobile Architecture schema has already been presented. Next we'll see more in detail modules composing such architecture grouped by functionalities: top level, AXMEDIS Communication, Domain Management, User Management, Content Management, Catalogue Management, Database Management, E-Commerce, Lobster Communication. For each module, first are presented related package and class diagrams, a tabular description and a detailed list of methods. From class diagrams the methods list has been deleted in order to keep diagram size reasonable.

8.6.1 Top Level Modules

Top level modules are Mobile Front End and Mobile Back End. While the former is basically a set of user interfaces, the latter is a kind of dispatcher which has the task to forwards user's requests to proper module.

8.6.1.1 Mobile Front End

This module is basically a set of user interfaces, without computational functions. There's not a single class that realizes the user front end interface, but a set of pages.



Module Profile MobileFrontEnd		
Executable or Library(Support)	Static Library	
Single Thread or Multithread	Multithread	
Language of Development	.NET Framework (ASP.NET)	
Responsible Name	Andrea Lorenzon	
Responsible Partner	ILABS	
Status (proposed/approved)	Proposed	
Platforms supported	Microsoft Windows	
Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)
AXMEDISClientPlayer	AXMEDISClientPlayer	ActiveX, ocx
File Formats Used	Shared with	File format name or reference to a section
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET,c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

Mobile Front End API

DE9.5.4 Integrated Prototype of content Production and Distribution on-demand for Mobile Phones, and new generation of PDAs

MobileFrontEnd:MobileFrontEnd				
+AcquireParsReq(in contentId : string(idl), in pars : sequence(idl)) : boolean(idl) +ConfirmPaymentData(in paymentData : string(idl)) : boolean(idl) +ContentsMngmtReq() : void +CreateCategory(in categoryKey : string(idl), in categoryNames : sequence(idl), in categoryQueryParameters : sequence(idl)) : boolean(idl) +CreateCategoryReq() : object(idl) +CreateDomain(in domainId : string(idl)) : boolean(idl) +CreateDomainReq() : object(idl) +DownloadContent(in contentId : string(idl), in userId : string(idl)) : object(idl) +DownloadPlayer() : boolean(idl) +GetActiveCategories() : sequence(idl) +GetCategoriesList() : object(idl) +GetContentsForCategory(in categoryKey : string(idl)) : sequence(idl) +GetContents(in query : string(idl)) : sequence(idl) +GetDownloadInstructions() : string(idl) +GetInformationForContent(in contentId : string(idl)) : string(idl) +GetLanguages() : sequence(idl) +GetParsForContent(in contentId : string(idl)) : sequence(idl) +GetSupportedDeviceProfiles() : sequence(idl) +GetUserData(in userId : string(idl)) : object(idl) +InsertDevice(in deviceData : string(idl)) : boolean(idl) +InsertDeviceReq() : object(idl) +Login(in login : string(idl), in password : string(idl)) : boolean(idl) +LoginReq() : object(idl) +ObjectPreview(in userId : string(idl), in contentReference : string(idl)) : boolean(idl) +ObjectView(in contentId : string(idl), in userId : string(idl)) : boolean(idl) +PrepareContent() +RegisterUser(in userData : string(idl)) : boolean(idl) +RegistrationReq() : object(idl) +RemoveCategories(in categories : sequence(idl)) : boolean(idl) +RemoveDeviceProfiles(in deviceProfiles : sequence(idl)) : boolean(idl) +RemoveUser(in userId : string(idl)) : boolean(idl) +SearchContents(in queryParameters : sequence(idl)) : sequence(idl) +SelectCategories(in categories : sequence(idl)) : sequence(idl) +SelectContents(in contents : sequence(idl)) : sequence(idl) +SelectDeviceProfiles(in deviceProfiles : sequence(idl)) : sequence(idl) +SetLanguage(in language : string(idl)) : boolean(idl) +SetUserRoles(in userId : string(idl), in roles : sequence(idl)) : boolean(idl) +UpdateCategoriesStatus(in categoriesStatus : sequence(idl)) : boolean(idl) +UpdateUserProfile(in userId : string(idl), in userData : string(idl)) : boolean(idl)				

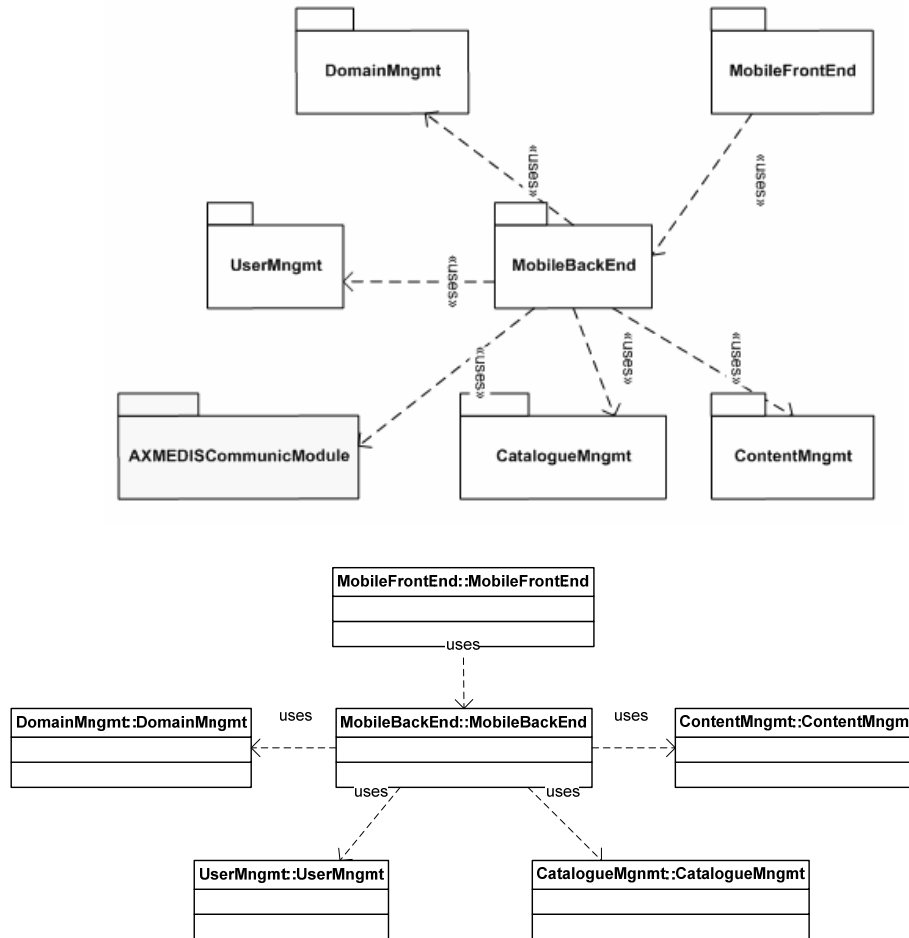
Function	Description	Method	Parameters	Reply
Content acquire	Requests PARs to acquire for a given content	AcquireParsReq	String contentId ArrayList pars	boolean acquiredPars
Content acquire	Confirms payment data in order to start rights payment and acquisition	ConfirmPaymentData	String paymentData	boolean dataOk
Content management	Asks to start the adaptation of contents	ContentsMngmtReq	-	-
Category definition	Requestes to create a new category with given parameters	CreateCategory	String categoryKey ArrayList categoryNames ArrayList categoryQueryParameters	Boolean createdCategory
Category definition	Requests form to insert new category parameters	CreateCategoryReq	-	Object createCategoryForm
Domain registration	Requests to create a new domain with given identifier	CreateDomain	String domainId	Boolean createdDomain
Domain registration	Requests form to insert new domain parameters	CreateDomainReq	-	Object createDomainForm
Content delivery	Gets content after rights acquisition	DownloadContent	String contentId String userId	Object content
Player download	Downloads an AXMEDIS Player	DownloadPlayer	-	Boolean downloaded
Catalogue loading	Shows to user categories he can select	GetActiveCategories		Array activeCategories
Category selection	Gest all categories stored into the system	GetCategoriesList	-	Object categoriesList

DE9.5.4 Integrated Prototype of content Production and Distribution on-demand for Mobile Phones, and new generation of PDAs

Function	Description	Method	Parameters	Reply
	and their status			
Catalogue loading	Gets contents for a given category	GetContentsForCategory	String categoryId	Array contents
Content retrieving	Get contents through a given query	GetContents	String query	Array contents
Player download	Gets instruction to download an AXMEDIS Player	GetDownloadInstructions	-	String instructions
Content information	Gets information about a given content	GetInformationForContent	String contentId	String contentInformation
Language selection	Gets all languages supported by the system	GetLanguages	-	ArrayList languages
ContentAcquire	Requests list of PARs for a given content	GetParsForContent	String contentId	ArrayList pars
Device Profile Removing	Gets the list of all supported device profiles	GetSupportedDeviceProfiles	-	ArrayList deviceProfiles
User data update	Retrieve a form populated with previously inserted user data	GetUserData	String userId	Object userDataForm
Device profile adding	Adds a new device profile with given data	InsertDevice	String deviceData	Boolean insertedDevice
Device profile adding	Requests form to add a new device profile	InsertDeviceReq	-	Object insertDeviceForm
Login	Logins user with given login and password	Login	String login String password	Boolean logged
Login	Requests form for login	LoginReq	-	Object loginForm
Content preview	Requests content preview	ObjectPreview	String contentReference String userId	Boolean oPreview
Content fruition	Requests content play	ObjectView	String contentReference String userId	Boolean okView
Content preparing	Requests to envelope content into an IMS Content Package	PrepareContent	String contentId	IMSContentPackage content
User registration	Requests to register user inside AXMEDIS system	RegisterUser	String userData	Boolean registeredUser
User registration	Requests form to register user	RegistrationReq	-	Object registrationForm
Category removing	Requests to remove given categories from the system	RemoveCategories	ArrayList categories	Boolean removedCategories
Device profiles removing	Requests to remove given device profiles from the system	RemoveDeviceProfiles	ArrayList devices	Boolean removedDevices
User remove	Requests to remove given user from the system	RemoveUser	String userId	Boolean removedUser
Content search	Requests to search for contents with given query parameters	SearchContents	ArrayList queryParameters	ArrayList contents
Language selection	Requests to set given language as the current one	SetLanguage	String language	Boolean settedLanguage
Roles management	Requests to set given roles for given user	SetUserRoles	String userId ArrayList roles	Boolean settedRoles
Category selection	Requests to update categories status	UpdateCategoriesStatus	ArrayList categoriesStatus	Boolean updatedCategories
User data update	Requests to update data for given user	UpdateUserProfile	String userId String userData	Boolean updatedUser

8.6.1.2 Mobile Back End

This module is a kind of dispatcher which has the task to forwards user's requests to proper module. It also performs some basic data checking in order to avoid dispatching requests with uncorrect data.



Module Profile MobileBackEnd		
Executable or Library(Support)	Static Library	
Single Thread or Multithread	Multithread	
Language of Development	.NET Framework (ASP.NET)	
Responsible Name	Andrea Lorenzon	
Responsible Partner	ILABS	
Status (proposed/approved)	Proposed	
Platforms supported	Microsoft Windows	
Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)
File Formats Used	Shared with	File format name or reference to a section
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET, c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

Mobile Back End API

```
+AcquireParsReq(in contentId : string(idl), in pars : sequence(idl)) : boolean(idl)
-CheckCategoryData(in categoryKey : string(idl), in categoryNames : sequence(idl), in categoryQueryData : sequence(idl)) : boolean(idl)
-CheckDeviceData(in deviceData : string(idl)) : boolean(idl)
+CheckLoginData(in login : string(idl), in password : string(idl)) : boolean(idl)
-CheckRegistrationData(in userData : string(idl)) : boolean(idl)
+ConfirmPaymentData(in paymentData : string(idl)) : boolean(idl)
+ContentsMngmtReq() : void
+CreateCategory(in categoryKey : string(idl), in categoryNames : sequence(idl), in categoryParameters : sequence(idl)) : boolean(idl)
+CreateCategoryReq() : object(idl)
+CreateDomain(in domainId : boolean(idl)) : boolean(idl)
+CreateDomainReq() : object(idl)
+DownloadContent(in contentId : string(idl), in userId : string(idl)) : object(idl)
+DownloadPlayer() : boolean(idl)
+GetActiveCategories() : sequence(idl)
+GetCategoriesList() : object(idl)
+GetContentsForCategory(in categoryKey : string(idl)) : sequence(idl)
+GetDownloadInstructions() : string(idl)
+GetInformationForContent(in contentId : string(idl)) : string(idl)
+GetLanguages() : sequence(idl)
+GetParsForContent(in contentId : string(idl)) : sequence(idl)
+GetSupportedDeviceProfiles() : sequence(idl)
+GetUserData(in userId : string(idl)) : object(idl)
+InsertDevice(in deviceData : string(idl)) : boolean(idl)
+InsertDeviceReq() : object(idl)
+Login(in login : string(idl), in password : string(idl)) : boolean(idl)
+LoginReq() : object(idl)
+GetContents(in query : string(idl)) : sequence(idl)
+PrepareContent()
+RegisterUser(in userData : string(idl)) : boolean(idl)
+RegistrationReq() : object(idl)
+RemoveCategories(in categories : sequence(idl)) : boolean(idl)
+RemoveDeviceProfiles(in deviceProfiles : sequence(idl)) : boolean(idl)
+RemoveUser(in userId : string(idl)) : boolean(idl)
+SearchContents(in queryParameters : sequence(idl)) : sequence(idl)
+SelectCategories(in categories : sequence(idl))
+SelectContents(in contents : sequence(idl))
+SelectDeviceProfiles(in deviceProfiles : sequence(idl))
+SetLanguage(in language : string(idl)) : boolean(idl)
+SetUpRuleForAdaptation()
+SetUserRoles(in userId : string(idl), in roles : sequence(idl)) : boolean(idl)
+UpdateCategoriesStatus(in categoriesStatus : sequence(idl)) : boolean(idl)
+UpdateUserProfile(in userId : string(idl), in userData : string(idl)) : boolean(idl)
```

Function	Description	Method	Parameters	Reply
Content acquire	Requests PARs to acquire for a given content	AcquireParsReq	String contentId ArrayList pars	boolean acquiredPars
Category definition	Checks inserted data for category definition	CheckCategoryData	String categoryKey ArrayList categoryNames ArrayList categoryQueryData	Boolean categoryOK
Device profile adding	Checks inserted data for new device profile	CheckDeviceData	String deviceData	Boolean deviceDataOK
Login	Checks inserted data for login	CheckLoginData	String login String password	Boolean loginDataOK
User registration	Checks inserted data for registration	CheckRegistrationData	String userData	Boolean registrationDataOK
Content acquire	Confirms payment data in order to start rights payment and acquisition	ConfirmPaymentData	String paymentData	boolean dataOk
Content preparing	Asks to start content preparing	ContentsMngmtReq	-	-
Category definition	Requestes to create a new category with given parameters	CreateCategory	String categoryKey ArrayList categoryNames ArrayList categoryQueryParameters	Boolean createdCategory

DE9.5.4 Integrated Prototype of content Production and Distribution on-demand for Mobile Phones, and new generation of PDAs

Function	Description	Method	Parameters	Reply
Category definition	Requests form to insert new category parameters	CreateCategoryReq	-	Object createCategoryForm
Domain registration	Requests to create a new domain with given identifier	CreateDomain	String domainId	Boolean createdDomain
Domain registration	Requests form to insert new domain parameters	CreateDomainReq	-	Object createDomainForm
Content delivery	Gets content after rights acquisition	DownloadContent	String contentId String userId	Object content
Player download	Downloads an AXMEDIS Player	DownloadPlayer	-	Boolean downloaded
Catalogue loading	Shows to user categories he can select	GetActiveCategories		ArrayList activeCategories
Category selection	Gest all categories stored into the system and their status	GetCategoriesList	-	Object categoriesList
Catalogue loading	Gets contents for a given category	GetContentsForCategory	String categoryId	ArrayList contents
Player download	Gets instruction to download an AXMEDIS Player	GetDownloadInstructions	-	String instructions
Content information	Gets information about a given content	GetInformationForContent	String contentId	String contentInformation
Language selection	Gets all languages supported by the system	GetLanguages	-	ArrayList languages
ContentAcquire	Requests list of PARs for a given content	GetParsForContent	String contentId	ArrayList pars
Device Profile Removing	Gets the list of all supported device profiles	GetSupportedDeviceProfiles	-	ArrayList deviceProfiles
User data update	Retrieve a form populated with previously inserted user data	GetUserData	String userId	Object userDataForm
Device profile adding	Adds a new device profile with given data	InsertDevice	String deviceData	Boolean insertedDevice
Device profile adding	Requests form to add a new device profile	InsertDeviceReq	-	Object insertDeviceForm
Login	Logins user with given login and password	Login	String login String password	Boolean logged
Login	Requests form for login	LoginReq	-	Object loginForm
Content preparing	Lools for contents using a given query	GetContents	String query	Array contents
Content preparing	Requests to envelope content into an IMS Content Package	PrepareContent	String contentId	IMSContentPackage content
Content preparing	Requests form to envelope content into an IMS Content Package	PrepareContentReq	-	Object prepareContentForm
User registration	Requests to register user inside AXMEDIS system	RegisterUser	String userData	Boolean registeredUser
User registration	Requests form to register user	RegistrationReq	-	Object registrationForm
Category removing	Requests to remove given categories from the system	RemoveCategories	ArrayList categories	Boolean removedCategories
Device profiles removing	Requests to remove given device profiles from the system	RemoveDeviceProfiles	ArrayList devices	Boolean removedDevices
User remove	Requests to remove given user from the system	RemoveUser	String userId	Boolean removedUser
Content search	Requests to search for contents with given query parameters	SearchContents	Array queryParameters	Array contents

Function	Description	Method	Parameters	Reply
Language selection	Requests to set given language as the current one	SetLanguage	String language	Boolean settedLanguage
Content preparing	Set up adaptation rule parameters	SetUpRuleForAdaptation	-	-
Roles management	Requests to set given roles for given user	SetUserRoles	String userID ArrayList roles	Boolean settedRoles
Category selection	Requests to update categories status	UpdateCategoriesStatus	ArrayList categoriesStatus	Boolean updatedCategories
User data update	Requests to update data for given user	UpdateUserProfile	String userID String userData	Boolean updatedUser

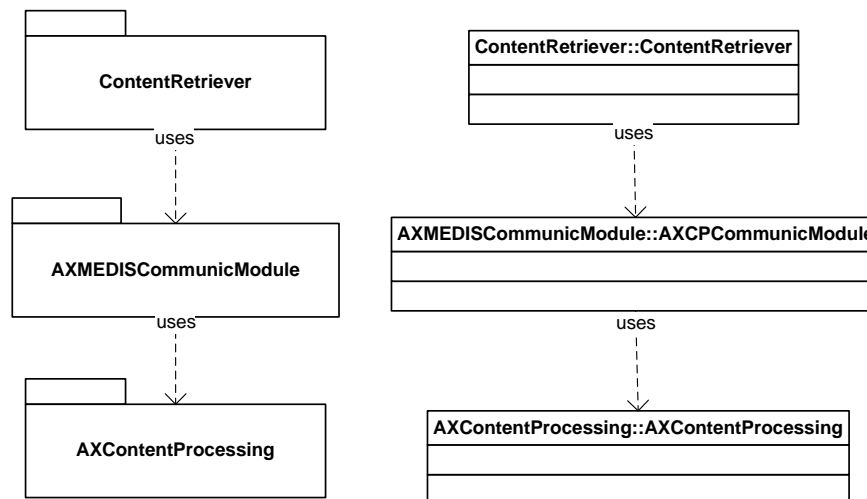
8.6.2 AXMEDIS Communication Modules

AXMEDIS Communication modules are the ones responsible of calling methods and receive results from AXMEDIS modules.

This section comprises: AXCPCommunicModule, AXCSCommunicModule, PMSCommunicModule, QSCCommunicModule and QueryOnDemandCommunicModule.

8.6.2.1 AXCPCommunicModule

AXCPCommunicModule wraps communication functionalities towards AXCP.



Module Profile AXCPCommunicModule		
Executable or Library(Support)	Static Library	
Single Thread or Multithread	Multithread	
Language of Development	.NET Framework (ASP.NET)	
Responsible Name	Andrea Lorenzon	
Responsible Partner	ILABS	
Status (proposed/approved)	Proposed	
Platforms supported	Microsoft Windows	
Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)
	AXCP	
File Formats Used	Shared with	File format name or reference to a section

User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET, c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

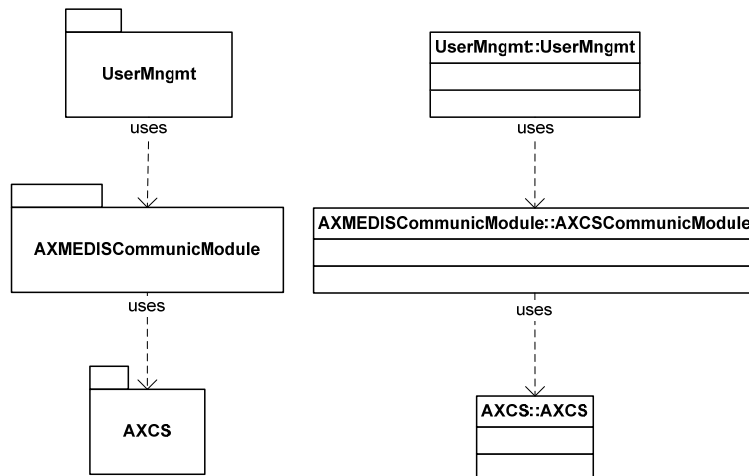
AXCPCommunicModule API

AXMEDISCommunicModule:AXCPCommunicModule
Adapt(in contents : sequence(idl), in deviceProfile : DeviceProfile) : boolean(idl)

Function	Description	Method	Parameters	Reply
Device profile adding	Requests to adapt contents for a given device profile	Adapt	ArrayList contents DeviceProfile deviceProfile	boolean adapted

8.6.2.2 AXCSCommunicModule

AXCSCommunicModule manages communication towards AXCS module.



Module Profile AXCSCommunicModule		
Executable or Library(Support)	Static Library	
Single Thread or Multithread	Multithread	
Language of Development	.NET Framework (ASP.NET)	
Responsible Name	Andrea Lorenzon	
Responsible Partner	ILABS	
Status (proposed/approved)	Proposed	
Platforms supported	Microsoft Windows	
Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)
AXCS Registration Web Service	AXCS	Web Services
File Formats Used	Shared with	File format name or reference to a section

User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET, c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

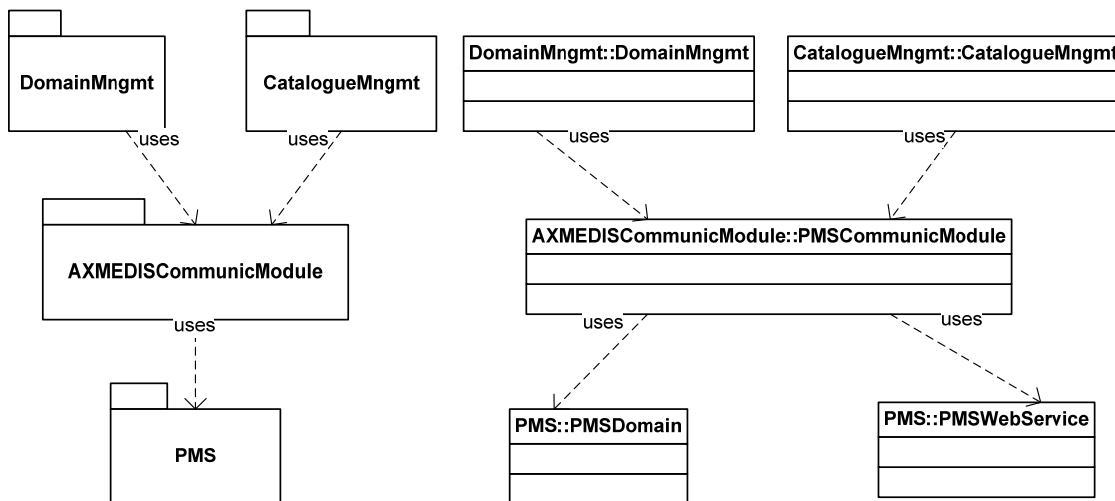
AXCSCommunicModule API

AXMEDISCommunicModule::AXCSCommunicModule
Registration(in userInfo : UserRegistrationInfo) : RegistrationResult

Function	Description	Method	Parameters	Reply
User registration User remove User data update	Requests to register user within AXMEDIS system	Registration	UserRegistrationInfo userInfo	RegistrationResult regResult

8.6.2.3 PMSCommunicModule

This module supports communication towards AXMEDIS PMS. It does not mean to reflect one by one PMS methods but only exposes some functionalities to rest of Mobile Application, asking for PMS to execute necessary operations. PMS will be also contacted by the AXMEDIS PDA player in order to check user's permissions over contents.



Module Profile PMSCommunicModule		
Executable or Library(Support)	Static Library	
Single Thread or Multithread	Multithread	
Language of Development	.NET Framework (ASP.NET)	
Responsible Name	Andrea Lorenzon	
Responsible Partner	ILABS	
Status (proposed/approved)	Proposed	
Platforms supported	Microsoft Windows	
Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)

PMS Web Service	PMS Domain PMS Web Service	Web Services
File Formats Used	Shared with	File format name or reference to a section
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET, c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

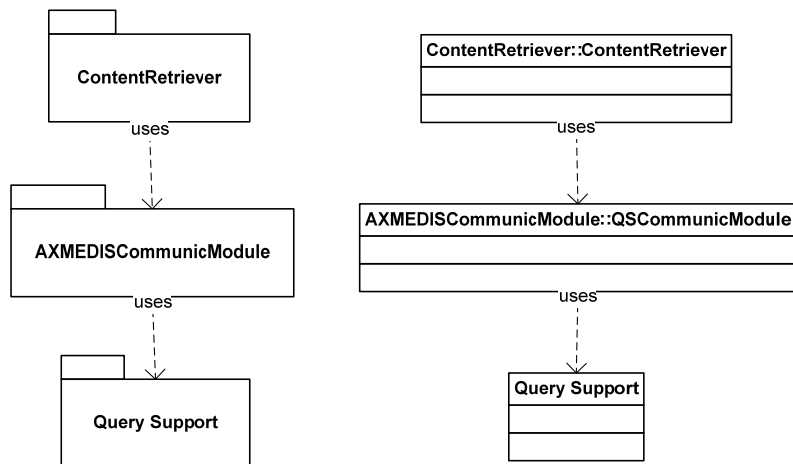
PMSCommunicModule API

AXMEDISCommunicModule::PMSCommunicModule
CreateDomain(in domain : Domain) : boolean(idl)
GenerateLicense(in licenseData : string(idl)) : string(idl)

Function	Description	Method	Parameters	Reply
Domain registration	Requests to create a new domain	CreateDomain	Domain domain	Boolean createdDomain
Content acquire	Requests to generate a license	GenerateLicense	String licenseData	String licenseld

8.6.2.4 QSCommunicModule

QSCommunicModule wraps communication functions towards AXMEDIS Query Support.



Module Profile QSCommunicModule	
Executable or Library(Support)	Static Library
Single Thread or Multithread	Multithread
Language of Development	.NET Framework (ASP.NET)
Responsible Name	Andrea Lorenzon
Responsible Partner	ILABS
Status (proposed/approved)	Proposed
Platforms supported	Microsoft Windows

Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)
Query Support	Query Support Web Service	Web Services
File Formats Used	Shared with	File format name or reference to a section
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET, c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

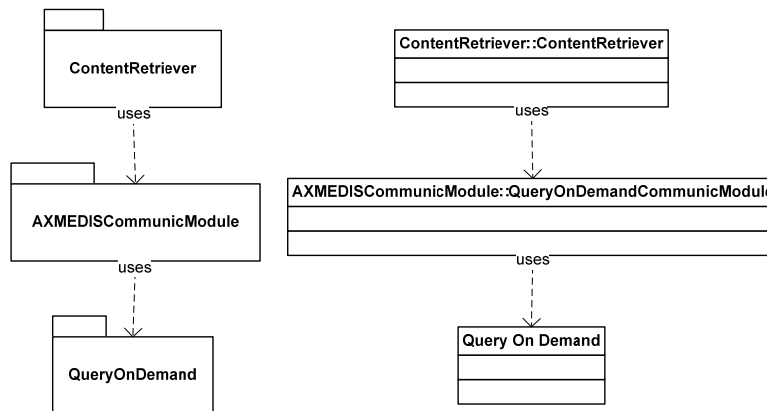
QSCommunicModule API

AXMEDISCommunicModule::QSCommunicModule
GetContentById(in contentId : string(idl)) : AXMEDIScontent
GetContentsByQuery(in query : string(idl)) : sequence(idl)

Function	Description	Method	Parameters	Reply
Content preparing Content information Content delivery	Retrieves content with a given identifier	GetContentById	String contentId	AXMEDIScontent content
Content acquire	Retrieves contents using given query	GetContentsByQuery	String query	ArrayList contents

8.6.2.5 QueryOnDemandCommunicModule

QueryOnDemandCommunicModule wraps communication functions towards AXMEDIS Query On Demand.



Module Profile QueryOnDemandCommunicModule	
Executable or Library(Support)	Static Library
Single Thread or Multithread	Multithread
Language of Development	.NET Framework (ASP.NET)
Responsible Name	Andrea Lorenzon
Responsible Partner	ILABS
Status (proposed/approved)	Proposed
Platforms supported	Microsoft Windows

Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)
	Query On Demand	
File Formats Used	Shared with	File format name or reference to a section
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET, c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

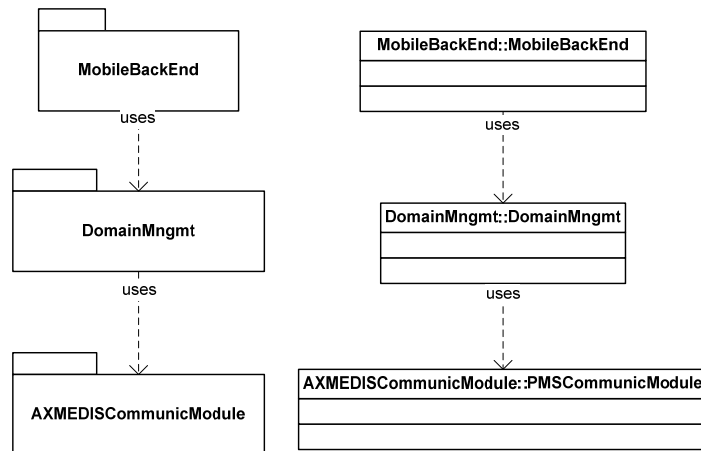
QueryOnDemandCommunicModule API

AXMEDISCommunicModule:QueryOnDemandCommunicModule
query(in query : string(idl)) : sequence(idl)

Function	Description	Method	Parameters	Reply
Content search	Retrieves contents using given query	query	String query	ArrayList contents

8.6.3 Domain Management Module

Domain Management module is the one responsible of the handling of the request for the creation and registration of a new domain .



Module Profile DomainMngmt	
Executable or Library(Support)	Static Library
Single Thread or Multithread	Multithread
Language of Development	.NET Framework (ASP.NET)
Responsible Name	Andrea Lorenzon
Responsible Partner	ILABS
Status (proposed/approved)	Proposed
Platforms supported	Microsoft Windows

Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)
No interaction with AXMEDIS tools		
File Formats Used	Shared with	File format name or reference to a section
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET, c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

DomainMngmt Module API

DomainMngmt:DomainMngmt
CreateAndRegisterDomain(in domainId : string(idl) : boolean(idl)
PrepareDomain(in AXDOM : string(idl), in AXID : string(idl), in typeOfId : string(idl)) : Domain

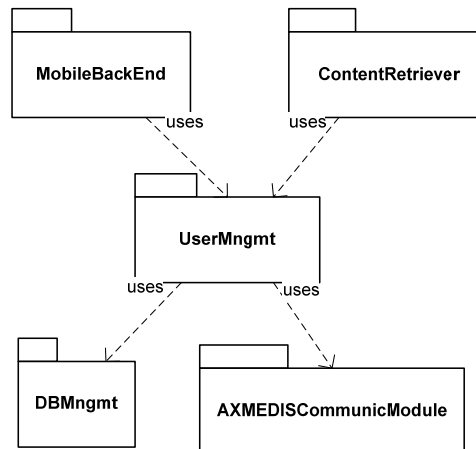
Function	Description	Method	Parameters	Reply
Domain registration	Requests to create and register a new domain	CreateAndRegisterDomain	String domainId	boolean domainOK
Domain registration	Requests to create a new domain	PrepareDomain	String AXDOM String AXID String typeOfId	Domain domain

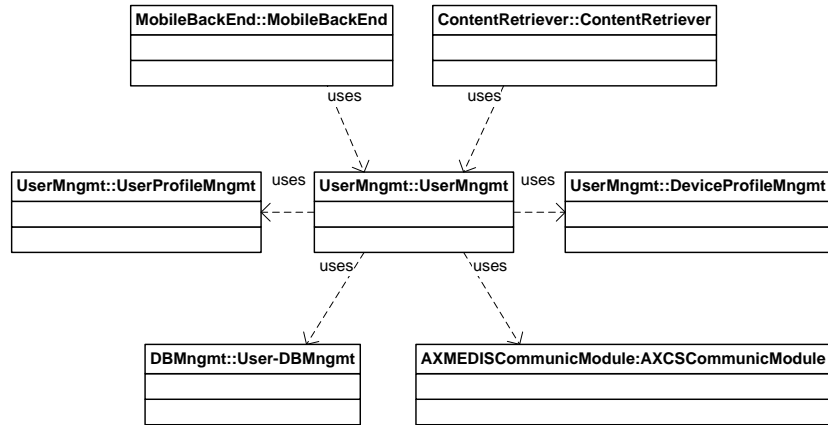
8.6.4 User Management Modules

User Management modules are the ones responsible of users data management: registration of new users, users data updating and removing, users roles management. This section comprises: UserMngmt, UserProfileMngmt and DeviceProfileMngmt modules.

8.6.4.1 UserMngmt Module

UserMngmt Module receives all requests for users data management and dispatches them to the specific module (UserProfileMngmt, DeviceProfileMngmt or DBMngmt module). It also uses AXCSCommunicModule in order to communicate to AXCS.





Module Profile UserMngmt		
Executable or Library(Support)	Static Library	
Single Thread or Multithread	Multithread	
Language of Development	.NET Framework (ASP.NET)	
Responsible Name	Andrea Lorenzon	
Responsible Partner	ILABS	
Status (proposed/approved)	Proposed	
Platforms supported	Microsoft Windows	
Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)
No interaction with AXMEDIS tools		
File Formats Used	Shared with	File format name or reference to a section
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET, c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

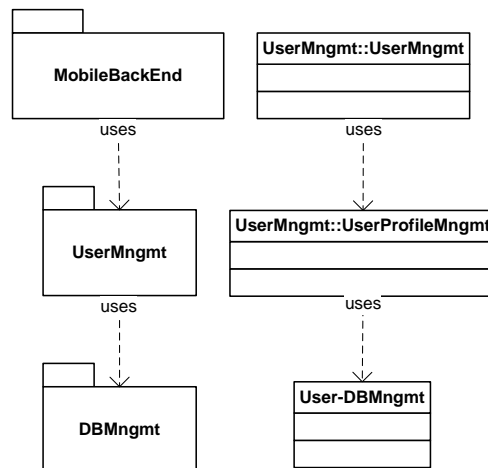
UserMngmt Module API

UserMngmt::UserMngmt
GetDeviceProfile(in deviceProfileId : string(idl)) : DeviceProfile GetSupportedDeviceProfiles() : sequence(idl) GetUserProfile(in userId : string(idl)) : UserProfile InsertDeviceProfile(in deviceProfile : string(idl)) : boolean(idl) Login(in login : string(idl), in password : string(idl)) : UserProfile RegisterUser(in userData : string(idl)) : boolean(idl) RemoveDeviceProfiles(in deviceProfiles : sequence(idl)) : boolean(idl) RemoveUser(in userId : string(idl)) : boolean(idl) SendAxuidByEmail(in userProfile : UserProfile) : boolean(idl) SendRemovedUserEmail(in userId : string(idl)) : boolean(idl) SetUserRoles(in userId : string(idl), in roles : sequence(idl)) : boolean(idl) UpdateUserLanguage(in userId : string(idl), in language : string(idl)) : boolean(idl) UpdateUserProfile(in userId : string(idl), in userData : string(idl)) : boolean(idl)

Function	Description	Method	Parameters	Reply
Content delivery	Requests to retrieve a device profile	GetDeviceProfile	String deviceProfileId	DeviceProfile deviceProfile
Device profile removing	Requests all supported device profiles	GetSupportedDeviceProfiles	.	ArrayList deviceProfiles
Content delivery User data update	Requests to retrieve an user profile	GetUserProfile	String userId	UserProfile userProfile
Device profile adding	Requests to store a new device profile	InsertDeviceProfile	String deviceProfile	Boolean insertedDeviceProfile
Login	Requests to login user with given login and password	Login	String login String password	UserProfile up
User registration	Requests to register user	RegisterUser	String userData	Boolean registeredUser
Device profile removing	Requests to remove given device profiles	RemoveDeviceProfiles	ArrayList deviceProfiles	Boolean removedProfiles
User remove	Requests to delete given user data	RemoveUser	String userId	Boolean removedUser
User registration	Sends an email to new registered user telling her/him AXUID	SendAxuidByEmail	UserProfile userProfile	Boolean sentEmail
User remove	Sends user an email to notify deletion	SendRemovedUserEmail	String userId	Boolean sentEmail
Roles management	Updates given user's roles as specified	SetUserRoles	String userId ArrayList roles	Boolean updatedRoles
Language selection	Requests to set user preferred language as specified	UpdateUserLanguage	String userId String language	Boolean updatedLanguage
User data update	Requests to update user profile as specified	UpdateUserProfile	String userId String userData	Boolean updatedUserProfile

8.6.4.2 UserProfileMngmt Module

UserProfileMngmt Module receives from UserMngmt module all requests specific for users profiles.



Module Profile UserProfileMngmt	
Executable or Library(Support)	Static Library
Single Thread or Multithread	Multithread
Language of Development	.NET Framework (ASP.NET)
Responsible Name	Andrea Lorenzon
Responsible Partner	ILABS
Status (proposed/approved)	Proposed
Platforms supported	Microsoft Windows

Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)
No interaction with AXMEDIS tools		
File Formats Used	Shared with	File format name or reference to a section
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET, c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

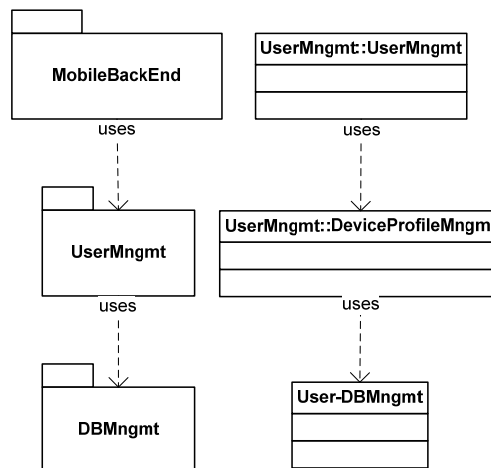
UserProfileMngmt Module API

UserMngmt::UserProfileMngmt
CreateAndStoreUserProfile(in userData : string(idl)) : UserProfile CreateUserProfile(in userData : string(idl)) : UserProfile GetUserProfile(in userProfileId : string(idl)) : UserProfile RemoveUser(in userId : string(idl)) : boolean(idl) UpdateUserProfile(in userId : string(idl), in userData : string(idl)) : boolean(idl)

Function	Description	Method	Parameters	Reply
User registration	Requests to create and store a new user profile	CreateAndStoreUserProfile	String userData	UserProfile userProfile
User registration	Requests to create a new user profile	CreateUserProfile	String userData	UserProfile userProfile
Content delivery User data update	Requests to retrieve an user profile	GetUserProfile	String userId	UserProfile userProfile
User remove	Requests to delete given user data	RemoveUser	String userId	Boolean insertedDeviceProfile
User data update	Requests to update user profile as specified	UpdateUserProfile	String userId String userData	Boolean updated

8.6.4.3 DeviceProfileMngmt Module

DeviceProfileMngmt Module receives from UserMngmt module all requests specific for users device profiles.



Module Profile DeviceProfileMngmt		
Executable or Library(Support)	Static Library	
Single Thread or Multithread	Multithread	
Language of Development	.NET Framework (ASP.NET)	
Responsible Name	Andrea Lorenzon	
Responsible Partner	ILABS	
Status (proposed/approved)	Proposed	
Platforms supported	Microsoft Windows	
Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)
No interaction with AXMEDIS tools		
File Formats Used	Shared with	File format name or reference to a section
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET, c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

DeviceProfileMngmt Module API

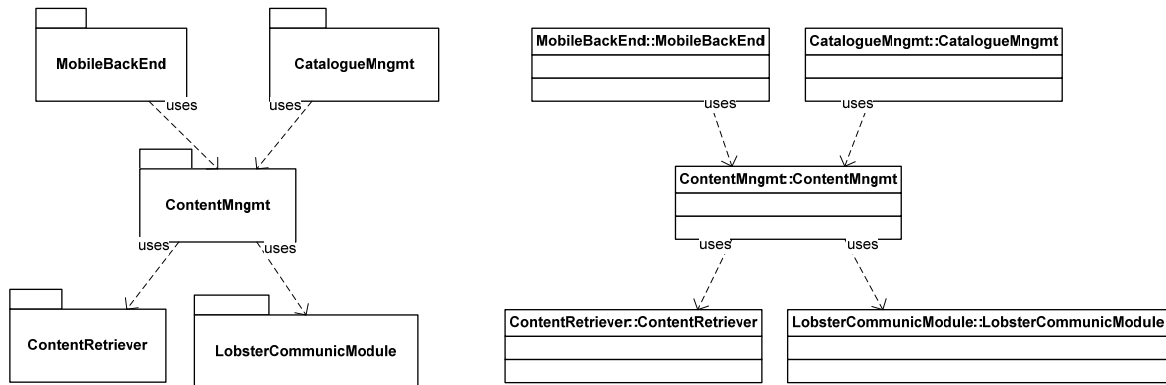
UserMngmt:DeviceProfileMngmt
CreateAndStoreDeviceProfile(in deviceData : string(idl)) : boolean(idl) CreateDeviceProfile(in deviceData : string(idl)) : DeviceProfile GetSupportedDeviceProfiles() : sequence(idl) GetDeviceProfile(in deviceProfileId : string(idl)) : DeviceProfile RemoveDeviceProfiles(in deviceProfiles : sequence(idl)) : boolean(idl)

Function	Description	Method	Parameters	Reply
Device profile adding	Requests to create and store a new device profile	CreateAndStoreDeviceProfile	String deviceData	Boolean deviceOK
Device profile adding	Requests to create a new device profile	CreateDeviceProfile	String deviceData	DeviceProfile deviceProfile
Device profile removing	Requests to retrieve all supported device profiles	GetSupportedDeviceProfiles	-	ArrayList deviceProfiles
Content delivery	Requests to retrieve a device profile	GetDeviceProfile	String deviceProfileId	DeviceProfile deviceProfile
Device profile removing	Requests to remove given device profiles	RemoveDeviceProfiles	ArrayList deviceProfiles	Boolean removedProfiles

8.6.5 Content Management Module

Content Management module is essentially responsible of content retrieving towards other Mobile Application modules: MobileBackEnd and CatalogueMngmt modules. In order to retrieve contents it uses ContentRetriever and LobsterCommunicModule.

DE9.5.4 Integrated Prototype of content Production and Distribution on-demand for Mobile Phones, and new generation of PDAs



Module Profile ContentMngmt		
Executable or Library(Support)	Static Library	
Single Thread or Multithread	Multithread	
Language of Development	.NET Framework (ASP.NET)	
Responsible Name	Andrea Lorenzon	
Responsible Partner	ILABS	
Status (proposed/approved)	Proposed	
Platforms supported	Microsoft Windows	
Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)
No interaction with AXMEDIS tools		
File Formats Used	Shared with	File format name or reference to a section
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET, c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

ContentMngmt Module API

ContentMngmt::ContentMngmt
<pre> +PrepareContent(in host : string(idl), in domain : string(idl), in username : string(idl), in password : string(idl), in objs : string(idl) in device : string(idl), in ops : string(idl), in categories : string(idl)) : boolean(idl) +DownloadContent(in contentId : string(idl), in userId : string(idl)) : object(idl) +EnvelopeContent(in content : AXMEDIScontent) : IMSContentPackage +GetContentsForCategory(in categoryKey : string(idl)) : sequence(idl) +GetInformationForContent(in contentId : string(idl)) : string(idl) +MarkContentsForRemovedDeviceProfile(in deviceProfileId : string(idl)) : boolean(idl) +SearchContents(in query : string(idl)) : sequence(idl) </pre>

Function	Description	Method	Parameters	Reply
Content preparing	Adapts selected contents for given device profiles and categories	PrepareContent	string host String domain String username String password String objs	boolean adapted

Module Profile ContentRetriever		
Executable or Library(Support)	Static Library	
Single Thread or Multithread	Multithread	
Language of Development	.NET Framework (ASP.NET)	
Responsible Name	Andrea Lorenzon	
Responsible Partner	ILABS	
Status (proposed/approved)	Proposed	
Platforms supported	Microsoft Windows	
Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)
No interaction with AXMEDIS tools		
File Formats Used	Shared with	File format name or reference to a section
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET, c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

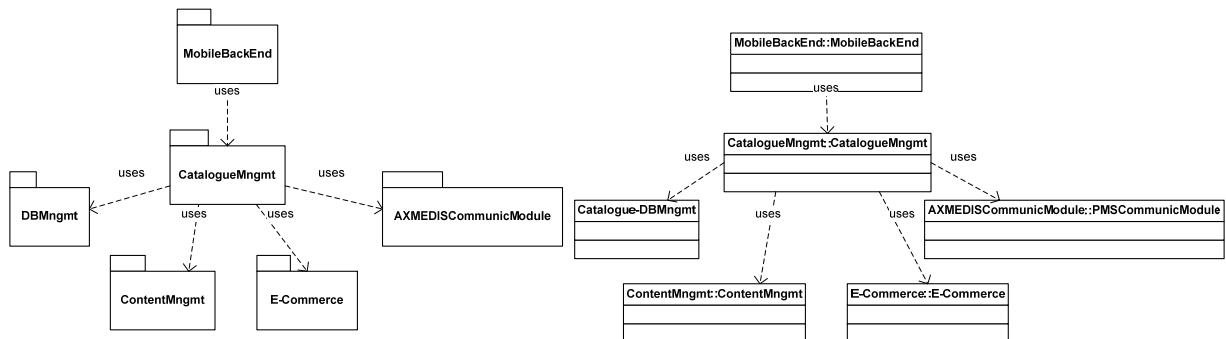
ContentRetriever Module API

ContentRetriever::ContentRetriever
Adapt(in contents : sequence(idl), in device : DeviceProfile) : boolean(idl) DownloadContent(in contentId : string(idl), in userId : string(idl)) : object(idl) GetAllContentsRef(in selectionCriteria : string(idl)) : sequence(idl) GetContentById(in contentId : string(idl)) : AXMEDIScontent GetContentsByQuery(in query : string(idl)) : sequence(idl) MarkContentsForRemovedDeviceProfile(in deviceProfileId : string(idl)) : boolean(idl) PrepareSearchContentsQuery(in queryParameters : sequence(idl)) : string(idl) RetrieveCategoryQuery(in categoryKey : string(idl)) : string(idl) SearchContents(in queryParameters : sequence(idl)) : sequence(idl)

Function	Description	Method	Parameters	Reply
Device profile adding	Requests to adapt contents for a given device profile	Adapt	ArrayList contents DeviceProfile deviceProfile	boolean adapted
Content delivery	Requests to download a content by given user	DownloadContent	String contentId String userId	Object content
Device profile adding	Requests references of all contents	GetAllContentsRef	String selectionCriteria	ArrayList contents
Content preparing Content information	Requests content with given identifier	GetContentById	String contentId	AXMEDIScontent content
Catalogue loading	Requests contents retrieved by given query	GetContentsByQuery	String query	ArrayList contents
Device profile removing	Requests to mark contents that have been adapted for a removed device profile	MarkContentsForRemovedDeviceProfile	String deviceProfileId	Boolean marked
Content search	Create query using given query parameters	PrepareSearchContentsQuery	ArrayList queryParameters	String query
Catalogue loading	Requests to retrieve query associated to given category	RetrieveCategoryQuery	String categoryKey	String category
Content search	Requests to search contents using given query parameters	SearchContents	ArrayList queryParameters	ArrayList contents

8.6.7 Catalogue Management Module

CatalogueMngmt Module receives from MobileBackEnd all the requests related to the catalogue, that is, the whole set of created and stored categories. In order to complete its tasks it uses AXMEDIS Communication, Database Management, E-Commerce and Content Management modules.



Module Profile CatalogueMngmt		
Executable or Library(Support)	Static Library	
Single Thread or Multithread	Multithread	
Language of Development	.NET Framework (ASP.NET)	
Responsible Name	Andrea Lorenzon	
Responsible Partner	ILABS	
Status (proposed/approved)	Proposed	
Platforms supported	Microsoft Windows	
Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)
No interaction with AXMEDIS tools		
File Formats Used	Shared with	File format name or reference to a section
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET, c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

CatalogueMngmt Module API

CatalogueMngmt::CatalogueMngmt
+AcquireParsReq(in contentId : string(idl), in pars : sequence(idl)) : boolean(idl) +ConfirmPaymentData(in paymentData : string(idl)) : boolean(idl) +CreateCategory(in categoryKey : string(idl), in categoryNames : sequence(idl), in queryParameters : sequence(idl)) : boolean(idl) +GetAXMEDISContents(in query : string(idl)) : string(idl) +GetCategoriesList() : sequence(idl) +GetInformationForContent(in contentId : string(idl)) : string(idl) +GetActiveCategories() : sequence(idl) +GetContentsForCategory(in categoryKey : string(idl)) : sequence(idl) +GetParsForContent(in contentId : string(idl)) : sequence(idl) +RemoveCategories(in categories : sequence(idl)) : boolean(idl) +StoreRequiredPars(in userId : string(idl), in contentId : string(idl), in pars : sequence(idl)) : boolean(idl) +UpdateCategoriesStatus(in categoriesStatus : sequence(idl)) : boolean(idl)

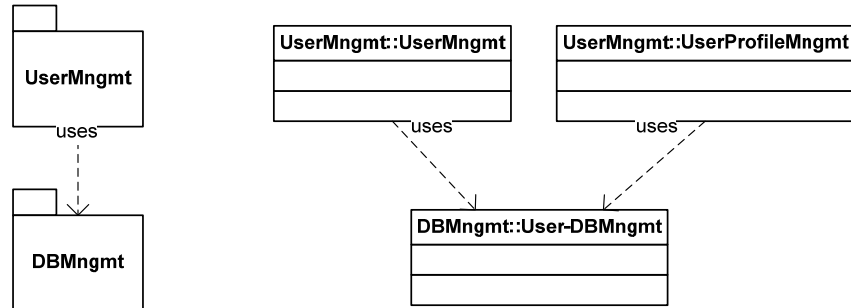
Function	Description	Method	Parameters	Reply
Content acquire	Requests to acquire given pars for given content	AcquireParsReq	String contentId ArrayList pars	Boolean acquired
Content acquire	The user confirms payment data	ConfirmPaymentData	String paymentData	Boolean dataOK
Category definition	Requests to create a new category with given parameters	CreateCategory	String categoryId Array categoryNames Array queryParameters	Boolean createdCat
Content preparing	Retireves contents through a given query	GetAXMEDISContents	String query	Array contents
Category selection	Requests list of all defined categories	GetCategoriesList	-	Array categories
Category removing	Requests to retrieve information about given content	GetInformationForContent	String contentId	String information
Content information	Requests list of all categories visible to users	GetActiveCategories	-	Array categories
Catalogue loading	Requests to retrieve contents for slected category	GetContentsForCategory	String categoryKey	Array contents
Catalogue loading	Requests rights on given content available for users	GetParsForContent	String contentId	Array pars
Content acquire	Reuquests to delete given categories from the system	RemoveCategories	ArrayList categories	Boolean removedCategories
Category removing	Store locally acquired pars until payment completion	StoreRequiredPars	String userId String contentId ArrayList pars	Boolean storedPars
Content acquire	Sets categories status as specified	UpdateCategoriesStatus	ArrayList categoriesStatus	Boolean updated

8.6.8 DB Management Modules

DBManagement modules are the ones responsible of storing data that are local to the Mobile Application. This section comprises User-DBMngmt and Catalogue-DBMngmt modules. These modules receive the retrieve requests and properly elaborate query and commands to be performed on database, providing an abstraction level that is very useful in case of migrating kind of database: other modules using DBMngmt package are not aware of database technical details.

8.6.8.1 User-DBMngmt Module

User-DBMngmt Module receives all requests for storing and retrieving locally users data from UserMngmt and UserProfileMngmt modules.



Module Profile User-DBMngmt		
Executable or Library(Support)	Static Library	
Single Thread or Multithread	Multithread	
Language of Development	.NET Framework (ASP.NET)	
Responsible Name	Andrea Lorenzon	
Responsible Partner	ILABS	
Status (proposed/approved)	Proposed	
Platforms supported	Microsoft Windows	
Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)
No interaction with AXMEDIS tools		
File Formats Used	Shared with	File format name or reference to a section
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET, c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

User-DBMngmt Module API

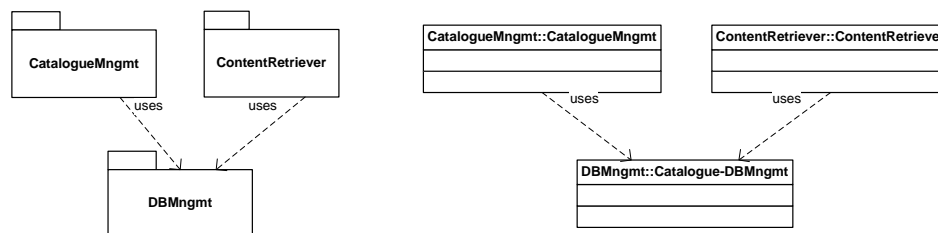
DBMngmt:User-DBMngmt				
ExistsUser(in username : string(idl), in password : string(idl)) : boolean(idl) GetDeviceProfile(in deviceProfileId : string(idl)) : DeviceProfile GetUserProfile(in userId : string(idl)) : UserProfile GetSupportedDeviceProfiles() : sequence(idl) InsertUser(in userData : UserProfile) : boolean(idl) PrepareDeviceInsertQuery(in deviceProfile : DeviceProfile) : string(idl) PrepareGetDeviceProfileQuery(in deviceProfileId : string(idl)) : string(idl) PrepareGetDevicesQuery() : string(idl) PrepareRemoveDevicesCmd(in deviceProfiles : sequence(idl)) : string(idl) PrepareRemoveUserCmd(in userId : string(idl)) : string(idl) PrepareUpdateUserProfCmd(in userId : string(idl), in userData : string(idl)) : string(idl) RemoveDeviceProfiles(in deviceProfiles : sequence(idl)) : boolean(idl) RemoveUser(in userId : string(idl)) : boolean(idl) StoreDeviceProfile(in deviceProfile : DeviceProfile) : boolean(idl) UpdateUserLanguage(in userId : string(idl), in language : string(idl)) : boolean(idl) UpdateUserProfile(in userId : string(idl), in userData : string(idl)) : boolean(idl)				
Function	Description	Method	Parameters	Reply
User registration Login	Tells if an user with given username and password	ExistsUser	String username String password	UserProfile up

DE9.5.4 Integrated Prototype of content Production and Distribution on-demand for Mobile Phones, and new generation of PDAs

	already exists; returns UserProfile or null			
Content delivery	Requests to retrieve a device profile	GetDeviceProfile	String deviceProfileId	DeviceProfile deviceProfile
Content delivery User data update	Requests to retrieve an user profile	GetUserProfile	String userId	UserProfile userProfile
Device profile removing	Retrieves all supported device profiles	GetSupportedDeviceProfile	-	ArrayList profs
User registration	Register a new user data	InsertUser	UserProfile userData	Boolean insertedUser
Device profile adding	Prepares query to store a new device profile	PrepareDeviceInsertQuery	DeviceProfile deviceProfile	String insertDev
Content delivery	Gets device profile	PrepareGetDeviceProfileQuery	String deviceProfileId	String devQuery
Device profile removing	Prepares query to retrieves all supported device profiles	PrepareGetDevicesQuery	-	String getDevicesQuery
Device profile removing	Prepares command to removes given device profiles from system	PrepareRemoveDevicesCmd	ArrayList devices	String removeCmd
User remove	Prepare command to deletes given user data	PrepareRemoveUserCmd	String userId	String removeCmd
User data update	Prepare command to update user profile	PrepareUpdateUserProfCmd	String userId String userData	String updateUsrCmd
Device profile removing	Removes given device profiles from system	RemoveDeviceProfiles	ArrayList profs	Boolean removedProfs
User remove	Deletes given user data	RemoveUser	String userId	Boolean removedUser
Device profile adding	Stores a new device profile	StoreDeviceProfile	DeviceProfile deviceProfile	Boolean storedDevProfile
Language selection	Sets given language as the user's preferred one	UpdateUserLanguage	String userId String language	Boolean updatedLang
User data update	Updates user profile	UpdateUserProfile	String userId String userData	Boolean updatedUserProf

8.6.8.2 Catalogue-DBMngmt Module

Catalogue-DBMngmt Module receives all requests for storing and retrieving categories data coming from CatalogueMngmt and ContentRetriever modules. Categories are meant as “areas of interest”: they are defined by Mobile Administrator and are later used by “normal” users to retrieved contents.



Module Profile Catalogue-DBMngmt		
Executable or Library(Support)	Static Library	
Single Thread or Multithread	Multithread	
Language of Development	.NET Framework (ASP.NET)	
Responsible Name	Andrea Lorenzon	
Responsible Partner	ILABS	
Status (proposed/approved)	Proposed	
Platforms supported	Microsoft Windows	
Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)
No interaction with AXMEDIS tools		
File Formats Used	Shared with	File format name or reference to a section

User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET, c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

Catalogue-DBMngmt Module API

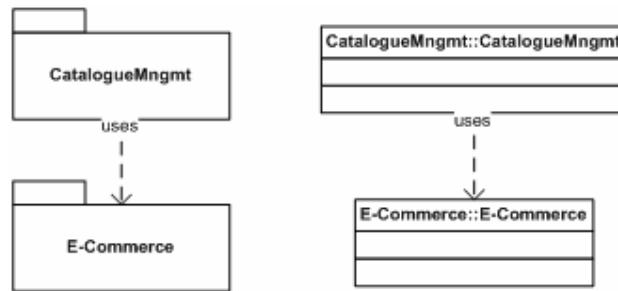
DBMngmt:Catalogue-DBMngmt
GetActiveCategories() : sequence(idl) GetCategoriesList() : sequence(idl) IsCategoryKeyAlreadyUsed(in categoryKey : string(idl)) : boolean(idl) PrepareGetActiveCategoriesQuery() : string(idl) PrepareGetCategoriesQuery() : string(idl) PrepareInsertCategoryCmd(in categoryKey : string(idl), in categoryNames : sequence(idl), in categoryQueryParameters : sequence(idl)) : string(idl) PrepareRemoveCategoriesCmd(in categories : sequence(idl)) : string(idl) PrepareRetrieveCategoryQueryQuery(in categoryKey : string(idl)) : string(idl) PrepareSearchCategoryKeyQuery(in categoryKey : string(idl)) : string(idl) PrepareUpdCategoriesStatusCmd(in categoriesStatus : sequence(idl)) : string(idl) RemoveCategories(in categories : sequence(idl)) : boolean(idl) RetrieveCategoryQuery(in categoryKey : string(idl)) : string(idl) StoreCategory(in categoryKey : string(idl), in categoryNames : sequence(idl), in categoryQueryParameters : string(idl)) : boolean(idl) UpdateCategoriesStatus(in categoriesStatus : sequence(idl)) : boolean(idl)

Function	Description	Method	Parameters	Reply
Catalogue loading	Gets list of active categories	GetActiveCategories	-	ArrayList categories
Category selection Category removing	Get categories list	GetCategoriesList	-	ArrayList categories
Category definition	Tells if given category key is already used within the system	IsCategoryKeyAlreadyUsed	String categoryKey	Boolean usedKey
Catalogue loading	Prepare query to get list of active categories	Prepare GetActiveCategoriesQuery	-	String query
Category selection Category removing	Prepare query to get categories list	Prepare GetCategoriesQuery	-	String query
Category definition	Preapre command to store a newly created category	PrepareInsertCategoryCmd	String categoryKey ArrayList categoryNames ArrayList categoryQueryParameters	String insertCmd
Category removing	Prepare command to remove given categories from the system	PrepareRemoveCategoriesCmd	ArrayList categories	String delCmd
Catalogue loading	Prepare query to retrieve query for given category	PrepareRetrieveCategoryQueryQuery	String categoryKey	String query
Category definition	Prepares query to tell if given category key is already used within the system	PrepareSearchCategoryKeyQuery	String categoryKey	String query
Category selection	Prepare command to update categories status	PrepareUpdCategoriesStatusCmd	ArrayList categoriesStatus	String updCmd
Category removing	Removes given categories from the system	RemoveCategories	ArrayList categories	Boolean removed
Catalogue loading	Retrieves query for given category	RetrieveCategoryQuery	String categoryKey	String query
Category definition	Stores a newly created category	StoreCategory	String categoryKey ArrayList categoryNames	Boolean stored

Function	Description	Method	Parameters	Reply
			ArrayList categoryQueryParameters	
Category selection	Updates categories status	UpdateCategoriesStatus	ArrayList categoriesStatus	Boolean updated

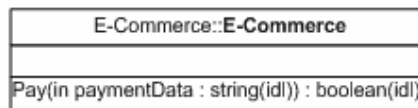
8.6.9 E-Commerce Module

E-Commerce module manages user payment for license rights. The module receives a request from CatalogueMngmt module and is the only component responsible to manage payment operations.



Module Profile E-Commerce		
Executable or Library(Support)	Static Library	
Single Thread or Multithread	Multithread	
Language of Development	.NET Framework (ASP.NET)	
Responsible Name	Andrea Lorenzon	
Responsible Partner	ILABS	
Status (proposed/approved)	Proposed	
Platforms supported	Microsoft Windows	
Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)
No interaction with AXMEDIS tools		
File Formats Used	Shared with	File format name or reference to a section
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET, c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

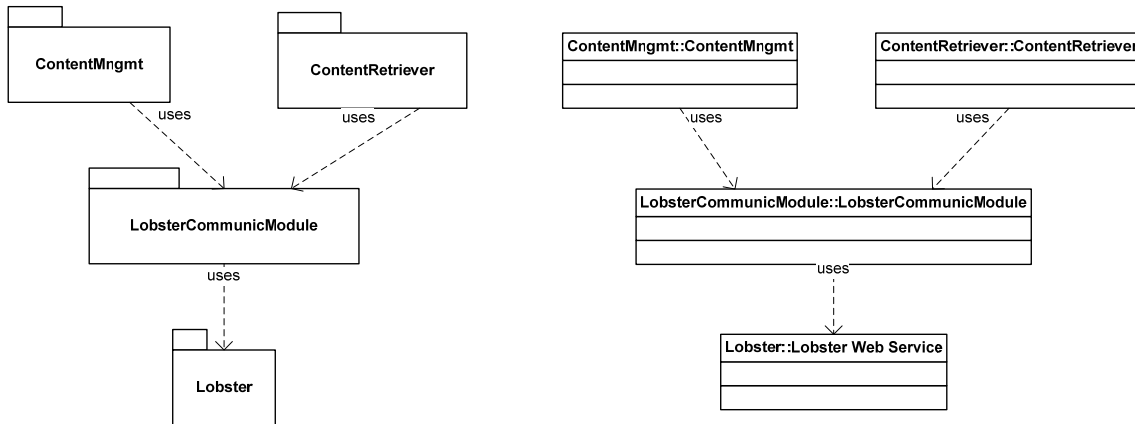
E-Commerce Module API



Function	Description	Method	Parameters	Reply
Content acquire	Payment method	Pay	String paymentData	Boolean payed

8.6.10 LobsterCommunicModule Module

LobsterCommunicModule manages communication with Lobster. ContentMngmt and ContentRetriever modules do not call directly Lobster Web Service but make their requests to LobsterCommunicModule. This one provides the only interface towards Lobster to the whole Mobile Application.



Module Profile LobsterCommunicModule		
Executable or Library(Support)	Static Library	
Single Thread or Multithread	Multithread	
Language of Development	.NET Framework (ASP.NET)	
Responsible Name	Andrea Lorenzon	
Responsible Partner	ILABS	
Status (proposed/approved)	Proposed	
Platforms supported	Microsoft Windows	
Interfaces with other tools:	Name of the communicating tools	Communication model and format (protected or not, etc.)
No interaction with AXMEDIS tools		
File Formats Used	Shared with	File format name or reference to a section
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web User Interface	ASP.NET, c#	.NET framework libraries
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

LobsterCommunicModule API

LobsterCommunicModule::LobsterCommunicModule
DownloadContent(in contentId : string(idl)) : AXMEDIScontent
GetContentsByQuery(in query : string(idl)) : sequence(idl)
Publish(in contentPackage : IMSContentPackage) : boolean(idl)

Function	Description	Method	Parameters	Reply
Content delivery	Gets content	DownloadContent	String contentId	AXMEDIScontent content
Catalogue loading	Gets contents	GetContentsByQuery	String query	ArrayList contents
Content preparing	Publishes content on Lobster	Publish	IMSContentPackage	Boolean published

8.7 Database Entity Relationship

Hereafter is reported the ER diagram for database local to Mobile Application followed by a short explanation of reported tables.

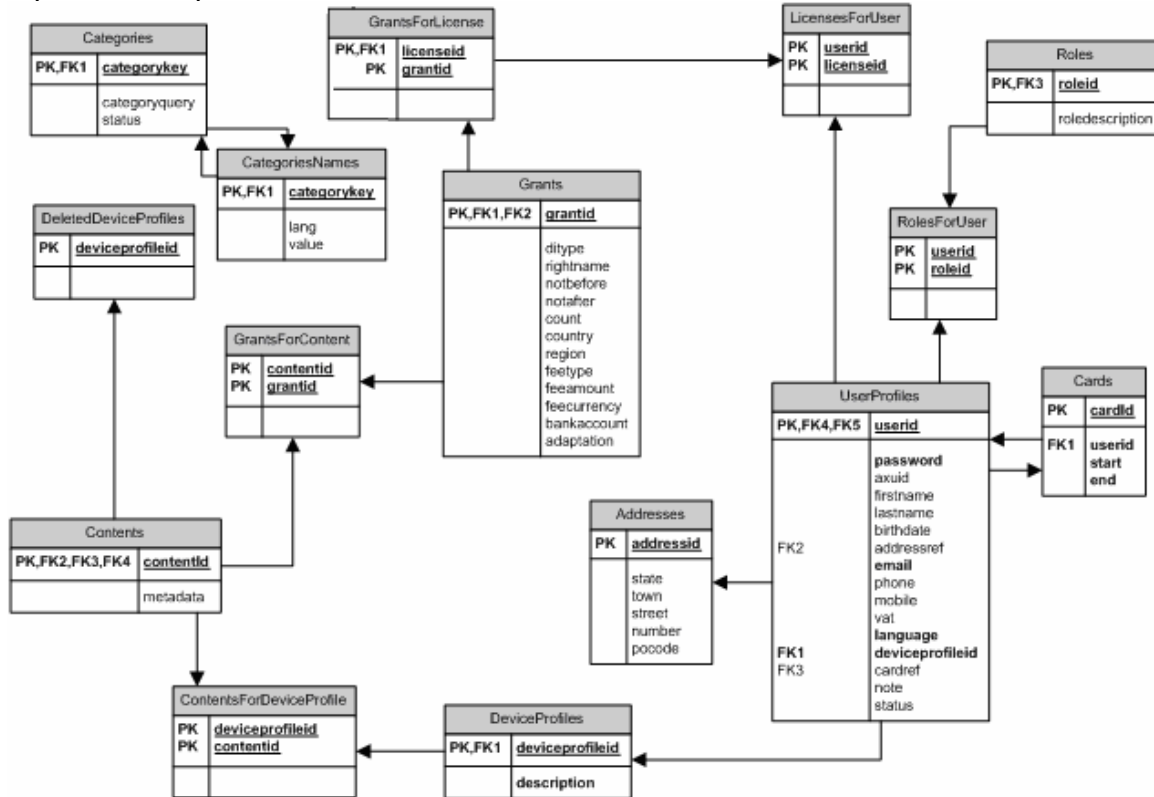


Figure 41. Entity Relationship diagram for local database.

1. UserProfiles: stores information about users

Colum	Data Type	Description
userid	VARCHAR(255)	User identifier that corresponds to user nick name
password	VARCHAR(255)	User password
axuid	VARCHAR(255)	User AXMEDIS identifier
firstname	VARCHAR(255)	User first name
lastname	VARCHAR(255)	User last name
birthdate	DATETIME	User birth date
addressref	VARCHAR(255)	References to an addressid in Addresses table, containing user address data
email	VARCHAR(255)	User e-mail
phone	VARCHAR(255)	User telephone
mobile	VARCHAR(255)	User mobile phone
vat	VARCHAR(255)	User vat code
language	VARCHAR(10)	User preferred language
deviceprofileid	VARCHAR(255)	References to a device profile id in DeviceProfiles table
cardref	VARCHAR(255)	References to a cardid in Cards table, containing user card data

Colum	Data Type	Description
note	VARCHAR(255)	Generic notes and comments written by he user
status	CHAR(1)	Possible values are B(blocked)/U(unblocked). For further details please see AXMEDIS-DE3-1-2H-AXFW-Spec-(Protection-Account)-Part-H, Section 2.9

2. Cards: stores information about users registration validity

Colum	Data Type	Description
cardid	VARCHAR(255)	Identifier of the card
userid	VARCHAR(255)	Owner user identifier
start	DATETIME	Date of registration
end	DATETIME	Date of end of registration validity

3. Addresses: stores information about users address

Colum	Data Type	Description
addressid	VARCHAR(255)	Identifier of the address
state	VARCHAR(255)	User state
town	VARCHAR(255)	User town
street	VARCHAR(255)	User street
number	VARCHAR(10)	User number
pocode	VARCHAR(10)	User PO code

4. Roles: stores information about Mobile Application users roles

Colum	Data Type	Description
roleid	VARCHAR(255)	Identifier of the role
roledescription	VARCHAR(255)	Description of the role

5. DeviceProfiles: stores information about device profiles supported by Mobile Application

Colum	Data Type	Description
deviceprofileid	VARCHAR(255)	Identifier of the device profile
description	VARCHAR(255)	Description of the device profile

6. DeletedDeviceProfiles: stores device profiles identifiers for profiles no more supported by Mobile Application

Colum	Data Type	Description
deviceprofileid	VARCHAR(255)	Identifier of the device profile

7. Grants: stores information about grants acquired by user of Mobile Application

Colum	Data Type	Description
grantid	VARCHAR(255)	Grant identifier
ditype	VARCHAR(255)	Grant ditype
rightname	VARCHAR(255)	Right name
notbefore	TIMESTAMP	Start validity of right

notafter	TIMESTAMP	End validity of right
count	INTEGER	Number of allowed uses
country	VARCHAR(255)	Country in which right exercise is allowed
region	VARCHAR(255)	Region in which right exercise is allowed
feetype	INTEGER	Fee type
feamount	DECIMAL	Fee amount
feecurrency	VARCHAR(255)	Fee currency
bankaccount	VARCHAR(255)	Bank account for payment
adaptation	VARCHAR(1000)	Adaptation

8. **Contents:** stores information about contents

Colum	Data Type	Description
contentid	VARCHAR(255)	Content identifier
metadata	VARCHAR(1000)	Content metadata

9. **Categories:** stores information about categories

Colum	Data Type	Description
categorykey	VARCHAR(255)	Unique category identifier
categoryquery	VARCHAR(1000)	Query associated to category
status	CHAR(1)	Status of category. Allowed values are: - A -> active so visible by users - U -> not active: stored into the system but not visible by users

10. **RolesForUser:** stores information about relationships between users and roles: an user can have one or more roles, a role can be associated to none, one or more users

Colum	Data Type	Description
userid	VARCHAR(255)	User identifier
roleid	VARCHAR(255)	Role identifier

11. **LicensesForUser:** stores information about licenses acquired by users: an user can have acquired zero, one or more licenses, a license can be associated to only one user

Colum	Data Type	Description
userid	VARCHAR(255)	User identifier
licenseid	VARCHAR(255)	License identifier

12. **GrantsForLicense:** stores information about grants inside licenses

Colum	Data Type	Description
licenseid	VARCHAR(255)	License identifier
grantid	VARCHAR(255)	Grant identifier

13. **GrantsForContent:** stores information about grants associated to contents

Colum	Data Type	Description
contentid	VARCHAR(255)	Content identifier
grantid	VARCHAR(255)	Grant identifier

14. **ContentsforDeviceProfile:** stores information about contents and device profiles

Colum	Data Type	Description
deviceprofileid	VARCHAR(255)	Device profile identifier
contentid	VARCHAR(255)	Content identifier

15. **CategoriesNames:** stores localized names for categories

Colum	Data Type	Description
categorykey	VARCHAR(255)	Category identifier
lang	VARCHAR(255)	Language expressed as ISO code
value	VARCHAR(255)	Localized category name

9 Used content samples description

The present section has to be considered as a simple reminder to be used to quickly locate places where the content to be used for the demonstrator is described and referenced. The reference sources for content identification and description may be found in DE3.1.3, DE8.1.1, DE8.4.1, DE8.2.1 and DE2.2.1 where content is introduced and presented. The starting point in relation to content to be used for the demonstrator can be summarised as follows:

- Content is focused on art and tourism
- Basic content samples to be used will have no special look and feel
- Used content will be in Italian and English

Apart from this we have divided info about content in the other deliverables as follows:

- In DE2.2.1 can be found a brief description of content needed for testing purposes,
- In DE8.1.1 can be found the available content and related mapping onto test cases,
- In DE8.4.1 can be found a guideline on editorial format definition,
- In DE3.1.3 can be found a guideline on content aspect and production (with examples and references),
- In DE8.2.1 can be found a guideline on content selection and usage (with examples and references).

10 References

In this first part of the section are reported the relevant project deliverable that are to be used as a reference for the present document:

- DE2.1.1 User Requirements and use cases
- DE2.2.1 Test cases and content description
- DE3.1.1 Guidelines and Specification of research enabling technologies
- DE3.1.2 Framework and tools Specification
- DE4.7.1 Analysis of Distribution Towards Mobiles
- DE3.1.3 AXMEDIS Content Aspects Specification
- DE8.1.1 Content for Test Cases and Validation
- DE8.2.1 Content Selection Guidelines
- DE8.4.1 AXMEDIS Editorial Format Guidelines and basic examples
- <http://www.imagemagick.org/>.
- <http://www.mega-nerd.com/libsndfile/>
- <http://sky.prohosting.com/oparviai/soundtouch/>.
- <http://www.mobile-phones-uk.org.uk/>
- http://www.pdatoday.com/comments/2934_0_1_0_C/
- <http://www.w3.org/Mobile/CCPP/>
- <http://www.hpl.hp.com/personal/marbut/DeliUserGuideWEB.htm>.
- http://arbor.ee.ntu.edu.tw/~jlhuang/publications/IMMCN03_ICC.pdf

- <http://wurfl.sourceforge.net/>
- <http://sourceforge.net/projects/delicon/>
- <http://cocoon.apache.org/2.1/developing/deli.html>
- <http://www.w3.org/TR/NOTE-CCPP/>
- <http://www.hpl.hp.com/personal/marbut/DeliUserGuideWEB.htm>

In this second part of the section are reported reference text, papers, articles, site and other relevant information sources that have been used to prepare the document and are other than project deliverables.

- [1] The Publishers Association – THE GLOSSARY of BOOK TRADE TERMINOLOGY from GLOSSARY OF PUBLISHING TERMINOLOGY 1997 - www.osi.hu/cpd/resources/paglossary.htm
- [2] Henry Budgett, J K Johnstone – Glossary of terms associated with the typesetting and printing industries – Based on a series of articles in a newsletter called "Desktop Publisher" published between 1986 and 1989 - The material contained in this glossary is originally the copyright of The Desktop Publishing Company Ltd and must be acknowledged as such if the material is re-used in any other form. However, permission for re-use is freely granted - <http://members.aol.com/richardw51/typeglossary.htm>
- [3] Harold Underdown – The Complete Idiot's Guide to Publishing Children's Books - Appendix A: Glossary of publishing terms and jargon 2004 - <http://www.underdown.org/cigglossary.htm>
- [4] Book Zone Pro - <http://www.bookzonepro.com/glossary.html> based on the Go Ahead Self-Publish! Glossary, compiled and edited by Eileen Birin - © 2004 Wheatmark, Inc
- [5] Mantex - DeskTop Publishing, a glossary of terms 2000 - www.mantex.co.uk/samples/dtp.htm
- [6] The Rainwater Press Publishing Primer - A glossary of terms for the electronic publishing, graphic arts, and printing industries 2000 - <http://www.rainwater.com/glossary.html>
- [7] Brandon Hall – Glossary 2004 - <http://www.brandonhall.com/public/glossary/>
- [8] Movie Glossary <http://www.factmonster.com/ipka/A0775970.html>
- [9] Movie Terminology Glossary <http://www.imdb.com/Glossary/>
- [10] The FilmLand Web Glossary by TriJet Productions © 1996 <http://www.fimland.com/glossary/Dictionary.html>
- [11] PBS – Public Broadcasting Service - Digital TV Glossary <http://www.pbs.org/digitaltv/glossary.html>
- [12] Digital Audio Guide – Glossary, D&F Services <http://www.digitalaudioguide.com/glossary.htm>
- [13] Digital TV Glossary - WOSU - <http://www.wosu.org/digital/glossary.php>
- [14] Satellite TV Glossary – Dish Network <http://www.dish-network-directv-specials.com/satellite-tv-glossary.aspx>
- [15] VCR Glossary – Dish Network <http://www.dish-network-directv-specials.com/satellite-tv-glossary.aspx>
- [16] Common Industry Format for Usability Test Reports Glossary <http://zing.ncsl.nist.gov/iusr/documents/glossary.htm>
- [17] Christian Salvaneschi, LOBSTER WEB SERVICE V2.1, GIUNTI Interactive Labs Srl, 2004, Sestri Levante

11 Glossary of acronyms & terms

The present section provides a reference to glossaries referenced / provided in other project documents.

11.1 Glossary of terms in the publishing environment

(Sources: please see References & Sources items [1]-[6])

DE9.5.4 Integrated Prototype of content Production and Distribution on-demand for Mobile Phones, and new generation of PDAs

11.2 Glossary of terms in the e-learning environment

(Sources: please see References & Sources items [7])

11.3 Glossary of terms in the movie, TV, Media... environment

(Sources: please see References & Sources items [8]-[15])

11.4 Other glossaries

(Sources: please see References & Sources items [16])