Automating Production of Cross Media Content for Multi-channel Distribution

www.AXMEDIS.org

DE9.3.3
First prototype of content production and distribution in push and on-demand for I-TV

Version: 1.2
Date: 31/08/2006
Responsible: EUTELSAT (revised and approved by the coordinator)

Project Number: IST-2-511299 Project Title: AXMEDIS Deliverable Type: Public Visible to User Groups: YES only the document and the demonstrations Visible to Affiliated: YES only the document and the demonstrations Visible to Public: YES only the document and the demonstrations

Deliverable Number: DE9.3.3 Contractual Date of Delivery: M24 Actual Date of Delivery: 31/08/2006 Work-Package contributing to the Deliverable: WP9.3.1, WP9.3.2 Task contributing to the Deliverable: WP9.3.2.1 Nature of the Deliverable: Documentation for promoting and presenting Prototype Author(s): EUTELSAT

Abstract:
This is a report on the current status of the prototype for demonstrating the content distribution via push.
Keyword List:
Distribution channel, content, satellite, prototype

Table of Contents

1 EXECUTIVE SUMMARY AND REPORT SCOPE 3

2 INTRODUCTION 3

3 PROTOTYPE FOR CONTENT DISTRIBUTION IN PUSH AND ON-DEMAND FOR I-TV 3

3.1 SENDER SIDE 4
3.1.1 API for content packaging and scheduling 5
3.2 RECEIVER SIDE 6
3.2.1 The ActionManager Module 7
3.3 DEMONSTRATION 9
3.3.1 Demonstration Video 9
3.3.2 Notes 9

4 BIBLIOGRAPHY 10

1 Executive Summary and Report Scope

This document represents an accompanying documentation to the Prototype for content production and distribution in push, as defined in the WP9.3. It contains a description of the first prototype for content distribution via satellite data broadcast, delivered as DE9.3.3.
The prototype is based on the specifications resulting from WP4 and WP5, in particular WP4.8.

2 Introduction

Satellite Data Broadcast is a content distribution mechanism that enables the distribution of the AXMEDIS content in a very efficient manner. This technology, provided by EUTELSAT OPENSKY platform, allows large quantities of data to be pushed via satellite directly on the user's PC without congesting local networks. The PUSH mechanism can be used, again, to renovate the catalogue of the Distributors periodically at low cost.
The prototype for the Satellite Data Broadcast inside the AXMEDIS environment focuses on the content delivery towards i-TVs, which, at this stage, are represented by a PC equipped with a DVB/IP satellite card.
This platform supports the complete cycle performed by a new content to be distributed by the Content Provider to a large number of Content Distributors. The distribution process uses the PUSH transmissions of the content according to the protocols that have been previously defined.
The Distributors acts as proxies for the content, making it available for the clients without the need to contact the Content Provider directly. On the client side, the demonstrator focuses on the B2C distribution, with a typical scenario of transmission of contents from a Distributor to some i-TV Clients. Client stations are able to store the received data in a cache area, where additional applications can access the data and handle it, likely for filtering, statistics, etc.
The prototype validates the distribution in multicast via PUSH technologies of a new multimedia content. The key elements for this system are:

- Content Provider station,
- Adaptation server,
- Satellite network infrastructure,
- Distributor station,
- i-TV client (PC equipped with a DVB/IP Satellite Card).

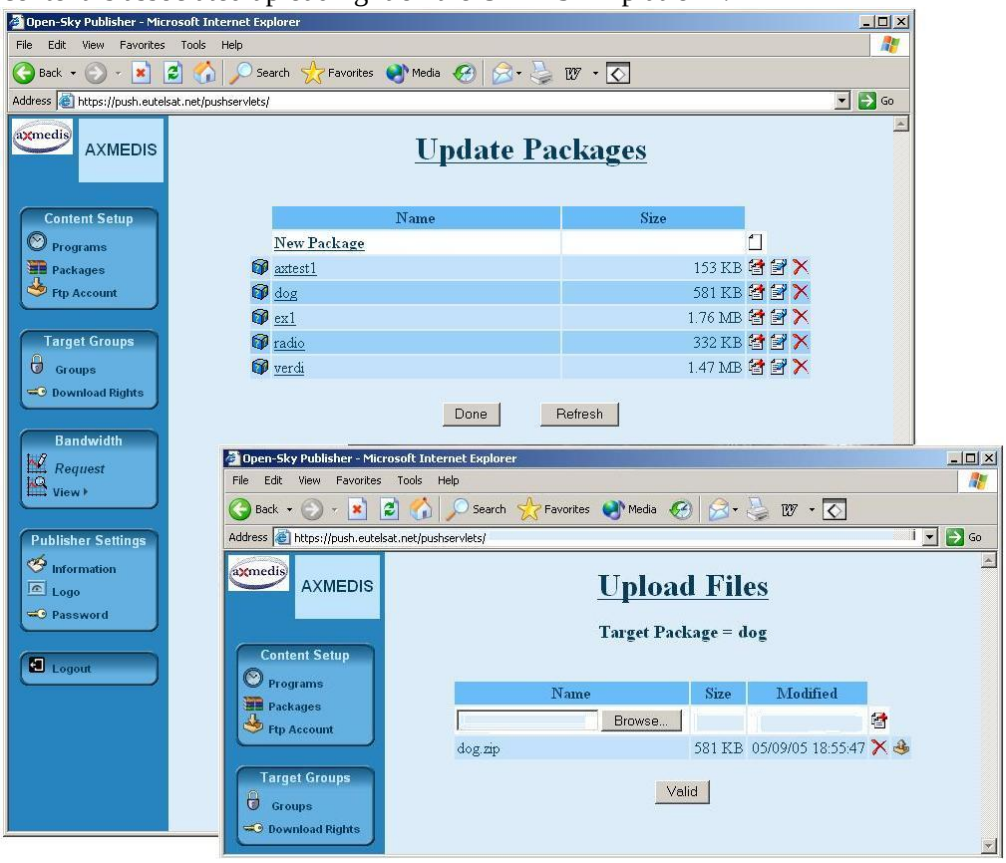
3 Prototype for Content distribution in push and on-demand for i-TV

The prototype focuses on the two sides of the transmission: the sender and the receiver; for the first side of transmission, the packaging of AXMEDIS Objects into OPENSKY Packages before, and the program creation and scheduling then.
In parallel, on the receiving station, the demonstrator validates how content transmissions are presented on the Electronic Program Guide, then selected for download or even automatically downloaded, according to the transmission parameters, and stocked on a predefined Cache Area for fruition or post-reception handling.
The prototype of Satellite Data Broadcast for content distribution in push is a stand-alone demonstrator, and it makes use of graphical interfaces to show the complete process.
On the sender side, a Web Interface is used to format the content in OPENSKY packages and programs, create and manage groups and security parameters.
At the same time, a set of API are available for the automatic creation of OPENSKY packages from AXMEDIS Objects, and they are especially conceived and employed for the integration with the other AXMEDIS distribution tools.

3.1 Sender Side

Content Packaging

The content has to be transformed in an OPENSKY package before being inserted into a program. The OPENSKY package contains a set of metadata used by the satellite distribution protocol. Once the package created, the content is associated uploading it on the OPENSKY platform.



OPENSKY Content Packaging

Program Scheduling

A program is a collection of content with a transmission schedule. Packages are grouped together inside a program, and then the program is scheduled for transmission. The schedule defines the speed (bandwidth), the start, the duration and possible repetitions of the distribution.

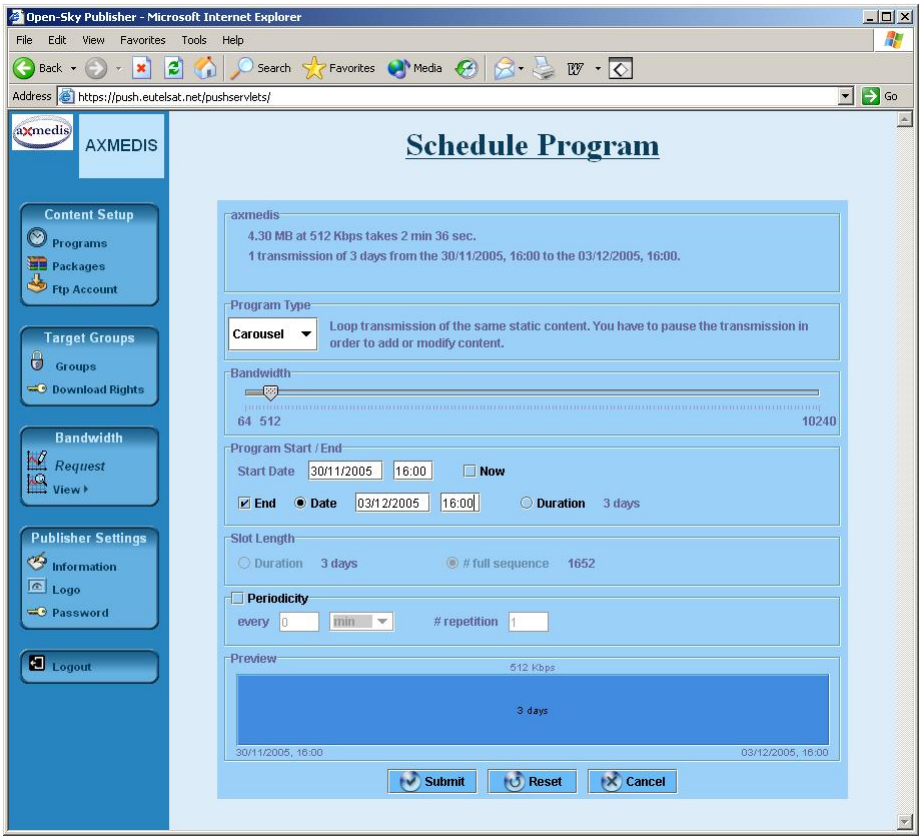


Figure: OPENSKY Program Scheduling

Program activation / stop

When the program has associated a schedule, the program enters an “active” state. When is such a state, the program can receive commands like play / pause / stop.

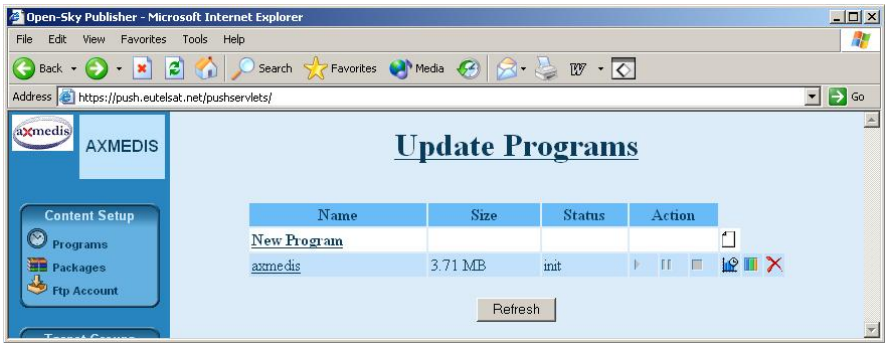


Figure: OPENSKY Program Updating

3.1.1 API for content packaging and scheduling

The communication between the Database access server and the server of a given provider can also do use of a set of Application Program Interface functions. These APIs provide the same functionalities illustrated using the Web Interface.

The APIs use an XML protocol encapsulated in HTTPS protocol, and their data format is XML.

The Eutelsat Database clients send request to a server provider that asks for data the Database access server. Both the Database access server and Eutelsat Database are owned and administrated by Eutelsat S.A.

Example:

The following lines provide an example of how calling a method of Push Server API, and how to parse the response received from the API Server.
A complete set of public functions and their possible responses (successful/unsuccessful) are contained in the document *API Push Server public functions* [APS].

Definition: create_package
Function to create a package.

Params	Notes
Name	Package name
Description	Package description

Request

```
<dbrequest>
      <administrator login="my_login" password="my_password"/>
      <operation name="create_package">
            <param name="name">packagename</param>
            <param name="description">the package description</param>
      </operation>
</dbrequest>
```

Response

```
<?xml version="1.0" enconding="UTF-8" ?>
<response>
  <operation name="create_package">
  <row>
    <code>0</code>
    <message>Package created</message>
  </row>
  </operation>
</response>
```

The API includes functions for content packaging and program scheduling and activation. They will be used in the integration of the AXMEDIS P&P tool with the Satellite Distribution Channel. With the integration, once a Distributor define a Programme and chooses the satellite channel for the delivering, the calls to the API will prepare the corresponding packages and programs on the Eutelsat platform, and finally activate the distribution.

3.2 Receiver Side

The reception is then validated on a client station previously installed with needed material, i.e. DVB card connected to a dish. The AXObjects sent with the programme are received on the enabled station, and stocked in the AXMEDIS cache area for further filtering. The pictures below show the GUI of the client application from which is possible to know which content is on air, what is currently download, and the content already received.

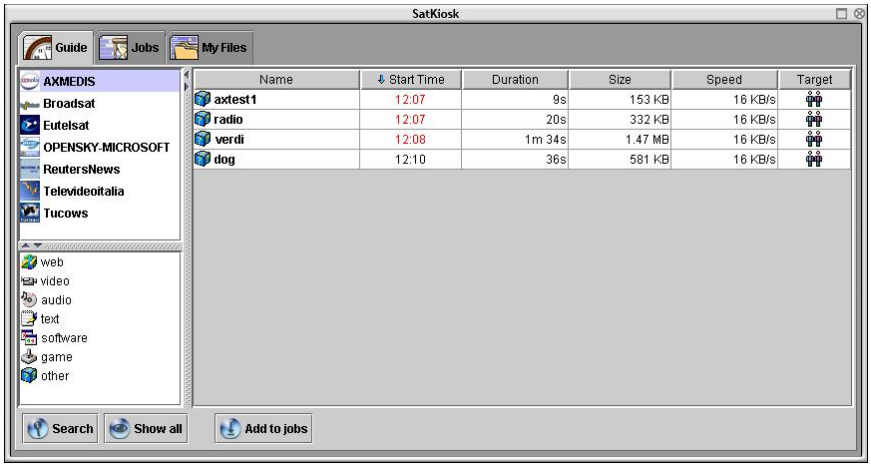


Figure: Program Guide with on-air content

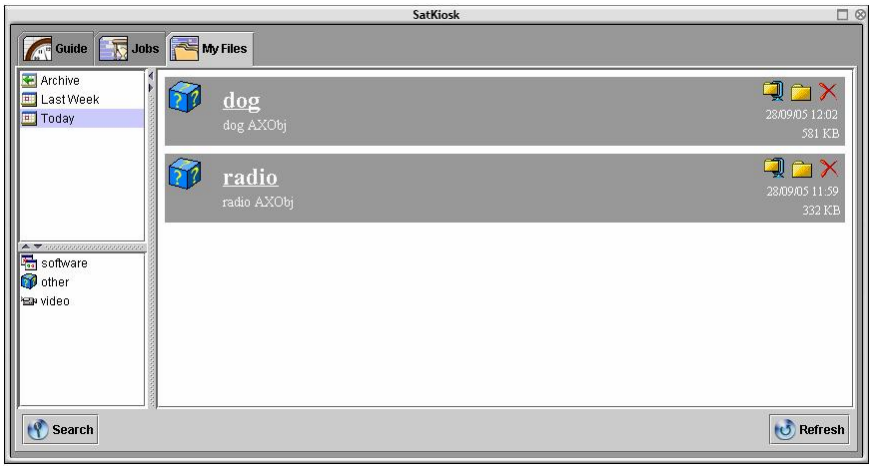
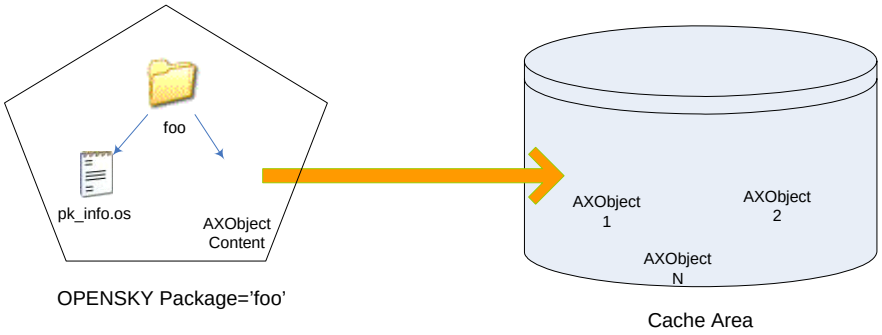


Figure: List of received content

Note that this graphical application is used to better explain how the reception works. It is a merely visual aid on the demonstration. The client application for the satellite data broadcast proposed by EUTELSAT includes this GUI, but, of course, it is not essential. The core of the client application is composed by other components that run in background and automatically let the receiving station download and store the content, without the necessity of a human interaction (except for the initial configuration phase).

3.2.1 The ActionManager Module

The prototype answers to the necessity of caching together the received AXMEDIS Objects. On the server side, AXMEDIS Objects are wrapped for transmission inside an OPENSKY Package, and with this format they are received on the client side. As the following picture shows, the received OPENSKY package, stored on the content area of the client, contains the pk_info.os with transmission metadata and the real AXMEDIS content (one or more files indeed).
The goal is to unwrap this content from OPENSKY package, and move it to a common specific *cache area*, where all AXMEDIS Objects are stored.



The location of the cache area is specified in the configuration files used by the client application. Once the AXMEDIS Objects are stored in a common area, they are available for direct fruition using the Axmedis Player, or can be accessed there for further content filtering and other post-reception handling.

The module in charge of this operation is the Action Manager. It is a module of the OPENSKY Listener, that is responsible for handling post-reception actions at the reception of new contents.
The agent entrusted of the content download, the Job, is also responsible to fire the corresponding actions.
Once the content reception is completed without errors, the Job class checks if any action is associated to the received package, and, if it the case, it instantiates the Action Manager classes responsible of these actions and invokes the execution of the actions.
The Job class implements this part of post-reception as follows:

1. Parse the package information field from the guide
2. Try to instantiate the appropriate Action Manager for each action parsed (the post-reception action may not always have an appropriate Action Manager)
3. Retrieve the action properties and store them within the Action Manager (destination folder, email address, …)
4. Try to execute the action on the content
5. If action succeeds, execute following actions, otherwise just stop processing anymore actions.

One of the actions associated to the content is the one that move the received content on a specified folder (MoveAction), as in the case of the AXMEDIS Objects, where the ActionManager module is automatically called when an OPENSKY package identified as an *axmedis* type is downloaded.

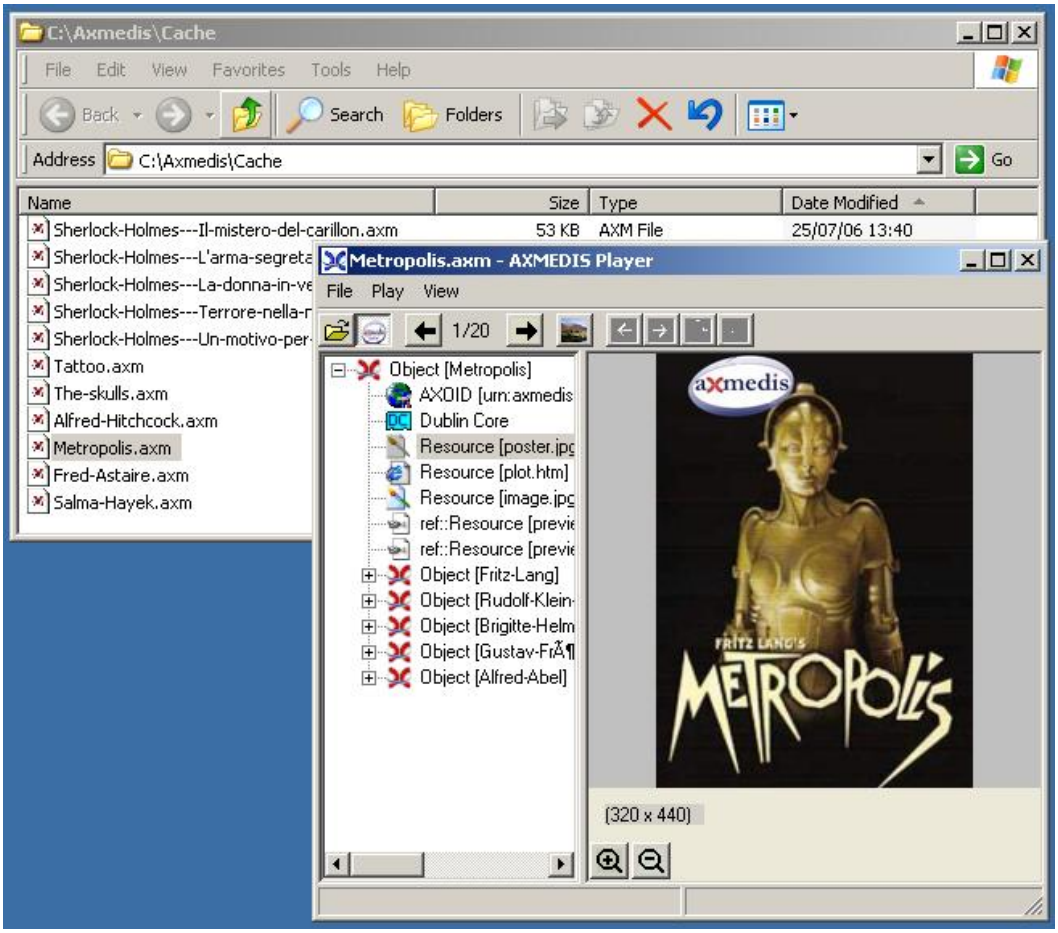


Figure: Fruition of cached AXMEDIS objects

3.3 Demonstration

3.3.1 Demonstration Video

For a multimedial overview of the satellite distribution, please refer to the video presentation of the demonstrator that has been released on July 2006.

The video presents the Content Distribution via Satellite data broadcast process, on both server/distributor side and client/user side.

The Publisher logs on the web interface and prepare the packages corresponding to the content to be delivered; then he/she defines the program, that means a catalogue of packages distributed, and its transmission parameters.

The video shows also in real time what happens when the program is on air. On the client side, the guide with the available content is live updated; the user can select the packages he/she's interested in from the list, and the download automatically starts at the scheduled time.

The received content is then available, thanks to the Action Manager, on the folder designed as cache area, and ready for fruition.

3.3.2 Notes

In the early phases of the project, the Consortium had planned to produce a simulator for the push/pull balance of data broadcasting delivering. Nowadays, the activity has been abandoned since it assumed less relevance for the market and for EUTELSAT;

On the other end, the demonstration could follow in the next months the direction of the WP4.8,

4 Bibliography

DE9.3.1 Specification of content production and distribution in push and on-demand for i-TV
axmedis-de9-3-1-spec-content-distribution-via-satellite-data-broadcast-v1-1.pdf
(http://www.axmedis.org/documenti/view_documenti.php?doc_id=1292)

DE9.3.2 Mock up of content production and distribution in push and on-demand for i-TV
axmedis-de9-3-2-mockup-content-distribution-via-satellite-data-broadcast-v1-0.pdf
(http://www.axmedis.org/documenti/view_documenti.php?doc_id=1919)

API Push Server public functions [APS], Eutelsat SA

Video on Satellite data broadcast and push (July 2006)
(http://www.axmedis.org/documenti/view_documenti.php?doc_id=2175)