Design and Implementation of a Multimedia Content Delivery System for Portable Devices

José D. González Arvelo
Master Student
Technical University of Catalonia. Barcelona, Spain
gonzalez.josedavid@gmail.com

Josep Paradells Aspas
Wireless Networks Group – Telematics Department
Technical University of Catalonia. Barcelona, Spain
Mod. C3 Campus Nord, c/ Jordi Girona 1-3, 08034.
josep.paradells@entel.upc.edu

ABSTRACT
This paper describes the design of a multimedia contents delivery system able of offering TV, radio, podcast and file services to portable devices. These services are distributed using reliable multicast, specifically FLUTE protocol (File Delivery over Unidirectional Transport). An alternative mechanism for transmitting TV contents is also proposed in this paper. This mechanism is based in podcast and seeks to take advantage of the mobile broadcasting networks currently in development. In addition, an Electronic Service Guide (ESG) has been designed to inform the users about the available contents. Such ESG allows client applications to personalize contents according to user's tastes and interests. The paper is structured as follows: first, it provides an introduction about mobile television; second, it explains briefly key aspects for the reliable multicast transport and mobile broadcasting technologies; then, our system design is exposed and finally, the system implementation is described.

Keywords
Broadcasting networks, reliable multicast, personalized contents, mobile devices, multimedia services.

1. INTRODUCTION
The third generation of mobile communications provides voice and data services at very high transmission rates (up to 14.4 Mbps with HSDPA) to users that find attractive the idea of being connected to Internet anytime and anywhere. In this scenario emerges the users' desire for enjoying TV on their mobiles devices. It is envisaged that by 2011 demand of mobile TV will explode with more than half a billion customers [1]. Furthermore, the recent deployment of new portable devices (PDA, UMPC, Tablet PC, etc.) extends the options for enjoying mobile TV.

TV service is by nature a one-to-many service. Therefore, despite of the high transmission rates that 3G networks offer, these are not well suited for mobile TV due to their point-to-point approach. Thus, together with this new service broadcasting networks emerge. These networks are oriented to broadcast services in downlink channels with great capacity. In addition, the idea of one-to-many services leads us to multicast transmissions.

Mobimedia'07, Month 8, 2007, Nafpaktos, Aitolokarnania, Greece

1.1 Mobile Television
Mobile television should not be considered as a substitute for regular television, but as a complement. In fact, mobile TV is very different from the regular TV service that we are used to. Recent studies define interactivity and personalization as key drivers for the success of this new service. People are interested on contents customized to their tastes. Moreover, users are also interested in interacting with motivating contents. This interactivity can be the source for big revenues to broadcasters and mobile operators.

Contents should be also of short duration for two main reasons: first we have the limited battery life of mobile devices and second, users will use mobile TV in short periods of time. Recent pilots show that most of the users will use mobile TV between 5 and 40 minutes a day, being the average program length less than 5 minutes.

1.2 TV Transport on the Internet
There are two transport mechanisms normally used for TV contents distribution: streaming and podcasting. Streaming consists on sending data as a steady stream that allows an application to reproduce audio and video services while data is arriving. This technique is very useful when the user storage capacity is limited, since there is no need to download the content before playing it. Video an audio are encapsulated by codecs into small packets that are sent continuously across the network in some kind of "continuous flow". In addition, with streaming providers keep control over the content distribution given that users cannot longer enjoy such contents after they have been transmitted. Real-time contents must relay on protocols like RTP (Real-time Transport Protocol) and RTCP (Real-time Transport Control Protocol), which are able of offering the synchronization support that IP networks lack.

On the other hand, podcasting consist basically in downloading video, audio or event data from the Internet. The content provider (podcaster) offers its contents that can be downloaded given a Uniform Resource Identifier (URI) in a web server. These URIs are listed in a file known as *feed*, which works as a guide to the user that wants to access the available contents. The podcast receiver known as *aggregator* keeps updated the user's files list with the *feeds* provided. The files can be played several times or

even be stored in other devices. The service can be free or it can work by previous subscription.

These transport mechanisms present their advantages and limitations. With streaming we can deliver live contents to users with a very small delay. Reliability is not as important as data delay; packets must be received in time. Therefore, the transport network must assure some minimal parameters of Quality of Service in order to support time-dependent data. This problem is not present in podcasting because when the content is played all the data has been received already. However, in this case reliability is mandatory given that data is treated as any other file. Moreover, with podcast we cannot transmit live TV or radio programs due to delay considerations.

2. RELIABLE MULTICAST TRANSPORT

Services like TV and radio broadcasting, videoconferencing or interactive gaming involve a large amount of receivers to whom the same content is distributed across a given network. A point-to-point solution for providing these services is not efficient given that it increases signaling and media-load overhead linearly with group size, consuming many resources and bandwidth over the network. Multicast reduces bandwidth consumption and signaling becoming a more scalable solution. Nevertheless, multicast by itself does not guarantee that packets are received correctly by all receivers, so reliability must be provided by the upper layers.

In unicast transmissions ARQ (Automatic Retransmission Request) is a widely used mechanism for providing reliability. Protocols like TCP use ACK messages to keep the sender updated of all correctly received datagrams. However, these methods do not scale well when many users receive data from the same flow. In a multicast transmission with thousands of receivers, the server may receive thousands of requests for retransmission, which may result in the server overload, causing what is called the *feedback implosion problem*. Moreover, these mechanisms require a return channel from the receiver to the sender, not always available in satellite or broadcast networks. Therefore, other mechanisms are needed to provide reliability in multicast transmissions.

2.1 FEC Codes

Forward Error Correction (FEC) is a key component for reliability in multicast. In block FEC codes the input file is divided into source blocks with k source symbols each. The encoder adds n-k redundant symbols, generating a total of n encoding symbols [2]. A block FEC decoder can reconstruct the original data if it receives correctly any k of the n encoding symbols. There are different types of FEC codes described in [2]: small block FEC codes (e.g. Reed-Solomon codes), which use blocks of maximum 255 encoding symbols; large block FEC codes (e.g. Tornado codes), which use larger blocks (thousands of symbols) keeping coding and decoding efficiency; and expandable or rateless FEC codes (e.g. Raptor codes) that can generate an unlimited amount of redundant symbols keeping the coding and decoding performance.

2.2 RMT Protocols

Nowadays, there are protocols oriented to the massively scalable distribution of files called RMT (Reliable Multicast Transport) Protocols. Two examples of these protocols are FLUTE (File Delivery over Unidirectional Transport) and NORM (NACK-Oriented Reliable Multicast).

FLUTE is a reliable file delivery protocol over UDP (User Datagram Protocol) that can be used as in unicast as in multicast transmissions, although is particularly suited to multicast [3]. It is based on ALC (Asynchronous Layered Coding) protocol, which is composed by three Building Blocks (BBs): LCT (Layered Coding Transport) BB, FEC BB and congestion control BB; the last two blocks are optional in FLUTE. In a FLUTE session, uniquely identified by its TSI (Transport Session Identifier), each object file is identified by a TOI (Transport Object Identifier) and an object called File Delivery Table (FDT) is continuously sent to the multicast group. The FDT is an XML file with TOI equal to zero that lists all the file objects transmitted in that session mapped with their respective TOI. The group members receive the FDT, look for the TOI of the desired file and then wait in the session until the desired object is transmitted or retransmitted if a file carrousel is implemented. The FDT can change dynamically during the session, keeping the users updated about the available contents. It also carries important information like the file size, content type, FEC type (if used), encoding type (if used) and an MD5 digest for data integrity.

On the other hand, NORM is a protocol designed for the reliable delivery of files to large groups via multicast using NACK (Negative ACK) policies for retransmissions [4]. As FLUTE, it is composed by several Building Blocks, being the most important NORM Sender Transmission, NORM Repair Process and NORM Receiver Join Policies. Basically, the sender multiplexes new data content with retransmission content. This data is normally protected using a FEC scheme and transmitted at a controlled rate according to a congestion control mechanism. When the receiver detects that data cannot be recovered with FEC protection, it will wait a backoff initial timeout period randomly picked. After the initial backoff timeout expires, the receivers will send their NACKs only if their retransmission needs have not been already requested by other receivers (NACK suppression). Therefore, either the receivers must multicast their NACK messages to the group or the sender must transmit some kind of NACK data to the receivers to inform the pending retransmissions. When the sender receives the first NACK message, it will wait a given period to receive other retransmission requests. This period will allow the source to send a more complete repair data.

3. MOBILE BROADCASTING NETWORKS

After the deployment of DTTB (Digital Television Terrestrial Broadcasting), the new target for broadcasters is to reach mobile devices. Standardization organisations have begun the struggle for defining the best mobile broadcasting technology. These technologies are based on DTTB standards, often adding new features in order to provide mobility to the user, make data more robust against harsh wireless conditions and cope with the problem of the limited battery life in mobile devices. The main mobile broadcasting technologies currently in development are: Digital Video Broadcasting-Handheld (DVB-H), Multimedia Broadcast/Multicast Service (MBMS), MediaFLO (Media Forward Link Only), Digital Multimedia Broadcasting (DMB) and *1seg* and *MobaHo!* from ISDB.

These technologies have in common the use of OFDM, the use of different FEC schemes for transmission reliability and the use of time multiplexing in order to save battery power in mobile devices, which lead us to data transmission in bursts. In addition, all broadcasting technologies, with the exception of MBMS, have standardized only the broadcasting link toward the user, leaving unspecified which will be the network for the uplink or return channel. This is the reason why mobile data networks like UMTS, GPRS or HSDPA are an important complement for broadcasting networks, forming with these what is known as hybrid networks.

Broadcasting technologies are ideal for the delivery of one-to-many services, typically involve wide area, high throughput and "push" networks as we can see in Figure 1. On the other hand, cellular networks typically offer low power transmitters covering smaller areas (cells). The result of such topology is that the network cost per user is significantly higher than in the case of broadcast networks, making broadcasting very expensive. However, mobile network operators offer excellent unicast communication channels (Figure 1) and an extensive and complex billing systems that broadcasting technologies lack. Hence, hybrid networks highlight the strong points of the two networks implicated in the delivery of advanced services to consumers.

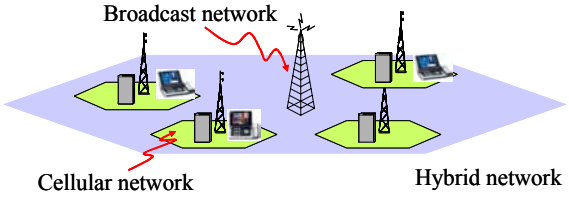

Figure 1: Hybrid Networks diagram

4. SYSTEM DESIGN

Our system has been designed to offer not only TV services, but also radio, podcast and download services to mobile devices. With this approach the user no longer has to follow the broadcaster schedule. The content arrives to the user and he plays it when preferred. The main elements of the system are described bellow.

4.1 Electronic Service Guide

An Electronic Service Guide (ESG) works as an interface that allows users to access efficiently to several services. It normally provides useful information to the user about the current available contents. The ESG is a key element in broadcasting and hybrid networks, since it can contribute for the development of mobile TV personalization and interactivity key features.

Our ESG is versatile enough to describe a wide range of contents like TV programs, radio, songs, ringtones, images, games, books and even advertisements. In addition, it provides to the user information about the acquisition of those contents according to the OMA (Open Mobile Alliance) specifications for DRM (Digital Rights Management) systems [5].

The ESG must be transmitted in a light format in order to consume the least bandwidth possible. A light format will also involve a faster ESG acquisition by the user, which allows the saving of battery power. For this reason, we have implemented our ESG in XML. One advantage of XML, as its name suggests, is its extensibility; even though our ESG design provides a complete data model, it is also possible to add new elements and attributes. Moreover, XML does not specify any representation rules for displaying the data, so this gives more flexibility to applications.

Some ESG current implementations like the IPDC (IP Datacast) ESG system contemplates the possibility of more than one ESG, so the end user must first perform an ESG bootstrap to find the list of available guides and then acquire the desired one. On the other hand, most of the possible business models for hybrid networks described in [6] envisage the offering of an integrated services packet (Triple-play service) to the end user. Given this scenario, it is totally feasible to implement a unique ESG where all the contents are offered, controlled by the mobile telecom operator or by the broadcaster. In this way, it is easier for the client application to personalize the contents because it only has to search them in one ESG instance. Acquisition and processing is faster, given that the user does not have to perform very long procedures to receive a requested ESG.

4.2 Contents Distribution

FLUTE has been chosen as the RMT protocol for our distribution system. It has been selected due to its simplicity and scalability compared with NORM, which is not that scalable due to its NACK based mechanism that can cause feedback implosion problems. NORM is more suitable to small or medium size groups, which is not the case for TV and radio services. Moreover, in order to allow NACK suppression, this mechanism requires a return channel from the receivers to the sender, which is not always available in broadcasting networks. The idea is to develop a system where the return channel is an additional service, but not a necessity. Moreover, the FDT of FLUTE allows receivers to be aware of all the contents transmitted within the same session. Figure 2 shows the protocol stack of the system.

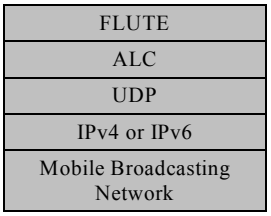

Figure 2: System protocol stack

The distribution system must be able of offering real-time and non-real-time contents. Therefore, it is not feasible to use the same sessions to send these services. For instance, FEC carrousel and retransmission techniques are useful when sending file contents, but they might not work well for real time services (TV and radio services), since in those cases data must arrive on time. Hence, different types of sessions have been specified in order to adapt the session characteristics to each service requirements.

4.2.1 ESG Session

It has been defined a specific FLUTE session for the transmission of the ESG. This session is permanent and follows an on-demand delivery model in order to have always available the ESG service to users. By sending solely the ESG instance on a FEC carrousel, the user does not have to wait for missing packets on the next cycles. Instead, it can join the session anytime and receive any k of n correct symbols to decode completely the ESG without waiting for retransmissions, since every symbol is equally useful

to decode the guide. Moreover, a client does not have to be synchronized with the sender; it can join the session whenever he wants, receive correctly *k* packets and then leave the session with the complete ESG. In this way, we achieve a very fast ESG reception.

4.2.2 SDP Session

This session carries the session description files that the users need in order to join a multicast group to receive a given content. This session is always available and it is implemented using a dynamic FEC carrousel like the example shown in Figure 3 .

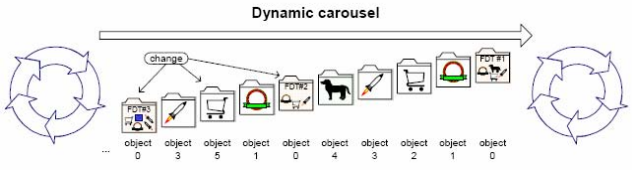

Figure 3: Example of dynamic file delivery carousel [7]

In order to send the session description in advance to all receivers, the Session Description Protocol (SDP) defined in RFC 2327 has been used complemented with the extensions defined in [8]. By separating session's description from the content description (ESG), we obtain a more efficient system because many contents may be sent in the same session and therefore, will have the same SDP file.

4.2.3 Podcast Session

This type of session transmits a large variety of contents of short duration and advertisements, which can be displayed while the application is in the initial waiting time (buffering time) or just between programs as a general rule. It is possible that advertisements need to be downloaded quickly, so it is not recommended to send them with large files on the same session. The idea is that the user application is continuously checking this session in background looking for new advertisements and short clips to offer or to download automatically. This is also an on-demand session with dynamic FEC Carrousel.

4.2.4 File Session

File sessions are designed to send any non-real-time content to the user. These contents have no limitation in size and can be sent solely or together on the same session. The file sessions can follow an on-demand or a push model. However, FEC Carrousel is mandatory on these sessions.

4.2.5 Podstream Session

In networks like DVB-H data transmission on the physical layer is in bursts in order to save battery power at the receiver. Even though the transport scheme may be streaming, the user is not receiving a constant data flow at all. This fact leads us to the intuitive idea that the transport mechanism can also be in bursts. We can transmit TV contents as a sequence of files with small duration, representing each file a data burst in a *podcast-based streaming* or *podstream*. As we can see in Figure 4 when a new user joins the group it waits an initial time similar to buffering time in streaming, in order to receive the first bites; then while he reproduces those bites the next bites arrive and so on.

For our implementation, the contents were partitioned into 10 second files (bites) at 250 Kbps with QVGA (Quarter Video Graphics Array) format. Since our system is designed for wireless environments, these small files were protected with FEC and were sent more than once in order to obtain reliability. Nevertheless, a FEC carrousel cannot be implemented because files have a time constraint. Hence, in these sessions a file is retransmitted consecutively for 10 seconds (the length of each file) and then the next file is transmitted. In this way, we emulate a TV transmission in real-time. Logically, the transition between bites must be as transparent as possible to the end user, so each bite has a redirection script that informs the application which file should reproduce next.

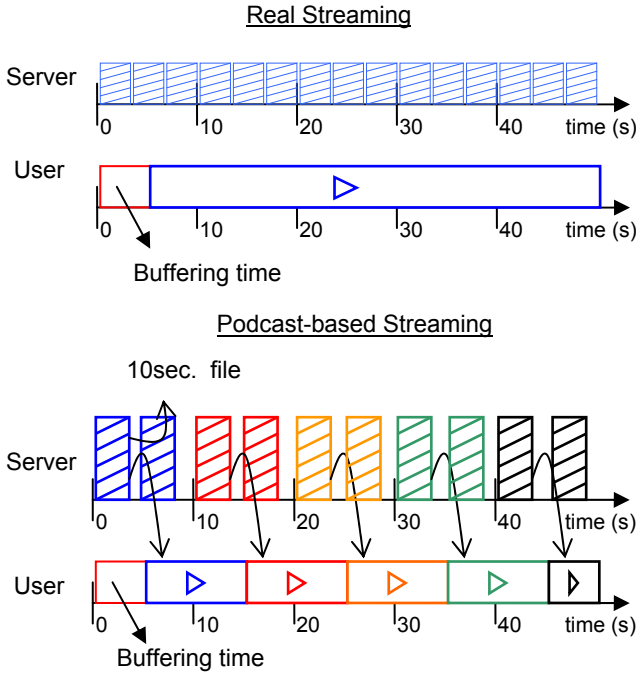

Figure 4: Streaming types comparison

4.3 Sessions Configuration

Event though there will be different types of session to distribute contents, there are some common parameters to all of them. For instance, the system MTU (Maximum Transmission Unit) will be 1424 Bytes in order that IP packets will not exceed Ethernet MTU (1500 Bytes); moreover, according to [9] a packet size ranging from 1024 bytes to 2048 bytes provide optimal efficiency in terms of data bandwidth and error resiliency.

Other important parameter is FEC ratio. In this system application layer FEC complements any other protection mechanism on the link layer. A study done in [10] evaluates the performance of FEC carrousel using RS codes with FLUTE. These simulations show that using a FEC ratio of 10% the user can receive data correctly in less than 2 carrousel loops. Moreover, in section 4.2.5 we explained that in podstream sessions files should be sent more than once with a time limit of 10 seconds. Given these parameters, we can calculate the transmission rate required for transmitting a podstream file of 360 KB twice, using FLUTE and RS codes with 10% ratio in 10 seconds, obtaining as result 763 Kbps. Transmission rate in the other sessions can vary depending

on the service requirements. However, it is recommended to leave the available bandwidth to video and radio services, with the exception of podcast sessions where files should be downloaded quickly. Table 1 shows the major components of the system designed.

Table 1: System main characteristics

ESG Format	XML
Sessions Description	SDP with FLUTE extensions
Video Format	QVGA at 250 Kbps
Application Layer FEC coding	RS codes (255,231)
FEC protection overhead	10%
Encoding symbol length (MTU)	1424 Bytes
IP/UDP/FLUTE header size	76 Bytes
ALC channels per session	1
Transmission rate	50-1500 Kbps

5. SYSTEM IMPLEMENTATION

5.1 Server Application

We have developed a FLUTE server based on a multithreaded scheme, which provides a graphic user interface that allows the service administrator to specify all the session parameters, from the transmission rate to the FEC configuration, in an easy way, storing such parameters on a SDP file for future sessions and displaying the active sessions that the administrator can cancel individually (Figure 5).

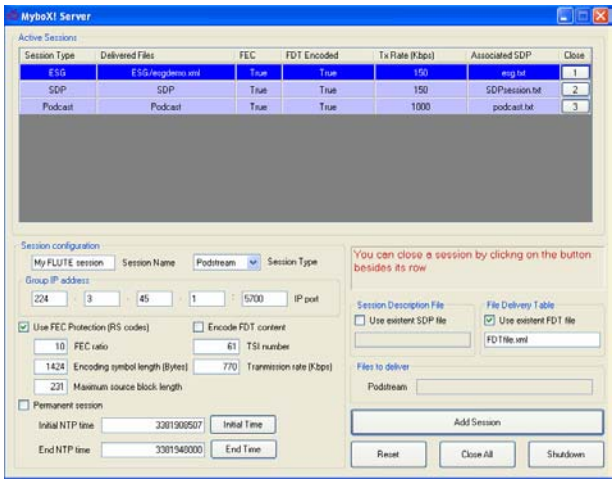

Figure 5: Server main form

This application has been developed in Visual C++ and it is built over MAD-FLUTE, a multicast file transfer tool based on FLUTE specifications developed by the MAD project of Tampere University of Technology in Finland [11]. In Figure 5 we can see the main form of our server.

5.2 Client Application

The server provides contents to a client application deployed also in Visual C++ named MyboX!, designed specially for UMPC devices. This application offers the following features.

5.2.1 Contents Personalization

MyboX! is able of personalizing received contents on the ESG according to user's profile. Such profile is provided by the user by using the form depicted in Figure 6.

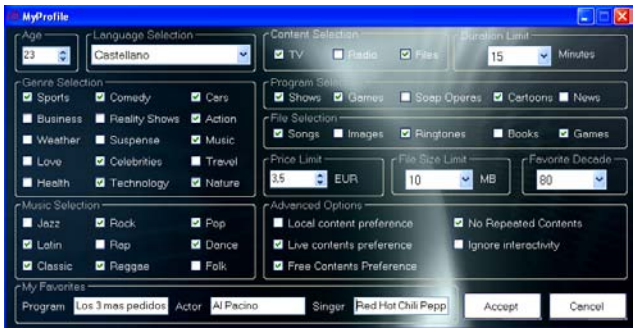

Figure 6: MyProfile form of MyboX!

Once the application has the user's profile, it can personalize the contents according to the information provided on the ESG. This personalization consists on grading each content in a scale from 0 to 10. By qualifying each content with a given score, we can have a measure of the affinity level between content and user, who selects a minimum score that MyboX! uses to personalize the ESG; contents that have a score above or equal to the reference will be displayed and contents bellow the score will be omitted. Furthermore, this method allows the user to adjust the desired personalization level, since with a reference score of zero all contents will be displayed but with a score of six (Figure 7) few contents will be offered.

Figure 7: File downloading while watching a TV program with interactivity

When the application parses successfully a received ESG, it proceeds to grade each of the contents using a pondered matching algorithm between user's profile and the content description. Such algorithm takes into account the type of content, its duration, size, language, genre(s) and type, if the content is free, local, interactive, repeated, live or even if it is from the user's favorite decade. In addition, the application verifies if it has the SDP file of each accepted content. After parsing the ESG the application starts a FLUTE SDP session in background in order to download the missing SDP files.

Finally, the application examines the ads list of the received ESG. Like the other contents, advertisements are graded according to user's profile, but then are compared with a different score for advertisements (4 by default). This is done to avoid that users can eliminate advertisements by putting a very high reference score. As in the case of missing SDP files, the application will begin a FLUTE podcast session in background to look for the required advertisements to show them to the user at the middle of a TV program or at the beginning while waiting for the first files to display.

5.2.2 Program Selection and Content Downloading

With MyboX! the user can enjoy contents in two different ways: watching and hearing contents transmitted by podstream (podcast-based streaming) or downloading a given content (video, song, etc.). If the content is a file to be downloaded, the application will start a file session in background with the configuration of the SDP file that is associated to that content. The path of this file is on the content description carried on the ESG. The user can have up to two simultaneous downloads while he is still able of enjoying a TV or radio program. When the downloading process starts, the content title appears on the data grid bellow the ESG display (Figure 7) together with its size and the initial time of the download. When the selected content is a TV program, the application starts a podstream session based on the content associated SDP file. While the first bite arrives, MyboX! displays advertisements to the user. These advertisements were previously selected and downloaded (if necessary) on the ESG downloading process. The personalized ads are cyclic so that all of them are showed to the user before the first is showed for the second time. Once the first bite arrives, the user begins to enjoy the TV program. In addition, users can "pause" the TV program and come back a few minutes later; also, they can repeat their favorite scenes.

5.2.3 Interactivity Features

Podstream files have encoded scripts that the application detects while it is playing the content. These scripts are XML files and can be of three different types: advertisement, redirection and trivia. With these scripts MyboX! offers interactivity features to the user without return channel. More than a client-server interaction, these features offer an application-client interaction, which is an added value in comparison with regular TV where the interaction with the user is very limited. In Figure 7 we can see an example of interactivity by means of a trivia script inserted on the podstream transmission. When a return channel is used interactivity may be richer and may provide even more possibilities.

6. CONCLUSIONS

We have designed a content delivery system able of providing TV, radio, podcast and file services to portable devices using FLUTE protocol in order to assure scalability with multicast and reliability by means of application layer FEC. Such system is based on an Electronic Service Guide that describes the available contents to users providing DRM support. The system contemplates different types of FLUTE sessions adapted to each service requirements. Furthermore, we propose an alternative method for transmitting TV contents to mobile devices. Podcast-based streaming or podstream consists on sending the contents in small files (360KB approx) that the client application can receive and show to the user like a uniform transmission. This model seeks to take the bursts transmission approach of mobile broadcasting networks to the application layer. In addition, we have concluded that in the possible business models for broadcasting and hybrid networks the idea of a unique ESG that gathers all the services offered by broadcasters is totally feasible.

The system has been implemented by means of a server application that provides a graphic interface for configuring all FLUTE sessions parameters, displaying the active sessions and allowing to cancel them independently. This server delivers the contents to a client application where users can enjoy TV contents as podcast-based streaming while they are downloading other contents in background. This application, based on user's profile, can personalize contents and advertisements that are described on the ESG through a score-based sorting where the user can select the personalization level at any moment, giving him more control on the contents his is interested in. The application has also interactivity support without return channel, offering to users trivia questions and advertisements that are inserted as scripts inside the video files.

7. ACKNOWLEDGMENTS

Our thanks to MAD Project of TUT in Finland. This work was supported in part by the Spanish Government through CICYT project TIC2006-04504

8. REFERENCES

[1] Orgad, Shani. *This Box was Made for Walking*. London School of Economics and Political Science, Department of Media and Communications. November, 2006.

[2] Luby, M., Vicisano, L., Gemmell, J., Rizzo, L., Handley, M. and Crowcroft, J. *The Use of Forward Error Correction (FEC) in Reliable Multicast*. Internet Engineering Task Force (IETF). Request for Comments: 3453, December 2002.

[3] Paila, T., Luby, M., Lehtonen, R., Roca, V. and R. Walsh. *FLUTE - File Delivery over Unidirectional Transport*. Internet Engineering Task Force (IETF). Request for Comments: 3926, October 2004.

[4] Adamson, B., Bormann, C., Handley, M. and Macker, J. *Negative-Acknowledgment (NACK)-Oriented Reliable Multicast (NORM) Building Blocks*. Internet Engineering Task Force (IETF). Request for Comments: 3941, November 2004.

[5] OMA-TS-DRM-REL-V2_0-20060303-A*: DRM Rights Expression Language*. Open Mobile Alliance. Approved Version 2.0, March 2006.

[6] *DigiTAG DVB-H Handbook. Television on a handheld receiver - broadcasting with DVB-H*. Digital Terrestrial Action Group. Version 1.2, 2005.

[7] ETSI TS 102 472: *IP Datacast over DVB-H: Content Delivery Protocols (CDP)*. European Telecommunications Standards Institute. June, 2006.

[8] Walsh, R., Peltotalo, J., Peltotalo, S., Mehta, H. and I. Curcio. *SDP Descriptors for FLUTE. Internet Engineering Task Force (IETF), draft-mehta-rmt-flute-sdp-05.txt*. January 2006.

[9] Kumar, V., Vadakitallt, M., Hannuksela, M., Razaei, M. and Gabbouj, M. *Optimal IP Packet Size for Efficient Data Transmission in DVB-H*. IEEE NORSIG, 2006. pp. 82-85.

[10] Peltotalo, J., Peltotalo, S. and Harju, J. *Analysis of the FLUTE Data Carousel*. Tampere University of Technology, Institute of Communications Engineering. Finland, 2006.

[11] *MAD Project's Home Page*. Available Online: http://www.atm.tut.fi/mad.