Generation, Streaming and Presentation of Electronic Program Guide

Cyril Concolato
Télécom ParisTech; Institut Télécom; CNRS LTCI
46, rue Barrault, 75013 Paris, France
Cyril.Concolato@telecom-paristech.fr

ABSTRACT
With the deployment of IPTV and mobile TV systems on the one hand, and the diversity of devices capable of displaying rich-media content on the other hand, the traditional monolithic way of presenting electronic program guide (EPG) data is becoming inappropriate. In this paper, we describe a system which proposes to separate the generation of presentable EPG data from its actual display. We present a system which achieves the generation and streaming of EPG based on streamable declarative languages. This system allows a reactive distribution and the efficient presentation of dynamic EPG.

Categories and Subject Descriptors
I.3.2 **[Computer Graphics]**: Graphics Systems – *Distributed/network graphics*; I.7.2 **[Document and Text Processing]**: Document Preparation – *Hypertext/hypermedia, Standards*

General Terms
Design, Experimentation, Languages, Performance.

Keywords
Electronic Program Guide, Presentation Language, Streaming.

1. INTRODUCTION
Traditional Electronic Program Guides (EPG), displayed on today's televisions, are the result of the on-the-fly generation of an interactive presentation based on information extracted from the broadcast channel. The generation and presentation are performed in a single place using software embedded either in the TV or in a STB, developed using programmatic languages. In traditional DVB environments, the EPG information is extracted using Event Information Tables (EIT) [1]. With the deployment of IPTV and mobile TV systems, this ecosystem is currently experiencing many changes. First, the number of sources, the type and quantity of information is changing. For example, EPG information can be retrieved from Web feeds (e.g. XMLTV [8]) or from DVB-IPDC [2] channels in the form of TV Anytime data [11]. Second, the number and type of devices capable of displaying rich media content (such as EPG) is also changing: TV, PC, PDA, Smartphones, Portable Media Players (PMP). All these devices have different screen configurations, interaction methods, connectivity features or processing power. The existing method which consists in embedding monolithic software to process and display EPG from a single source is therefore not well suited anymore because of development costs. As a consequence, more and more solutions now rely on dedicated and tailored Web browsers such as [3] to display EPG.

The generation and display of EPG is not a new research topic and many papers have been written on the topic. Most of them focus on the design of efficient interactive paradigms [14] or on recommendation systems [16]. In this work, we show a system which decouples the generation and the display of the EPG. This can be viewed as an example of the secondary screen approach described in [17]. The EPG is generated in a device, different from the rendering device, tuned to the broadcast channel to retrieve the raw EPG data. This data is then transformed into a presentation form which is streamed to the remote rendering device (phone, PMP). In this paper, we use the 3GPP DIMS [4] declarative language, rather than XHTML, to first benefit from its light, declarative rich graphics and animation features but also, to allow streaming of the EPG. The use of streaming follows the push approach of broadcasting technologies used in the delivery of raw EPG data but also allows guarantying a strong synchronization between the presentation and raw data. Additionally, streaming, coupled with the DIMS update mechanism enables light presentation processing in the client.

In the remaining of this paper, we present in Section 2 the possible architectures for such a system. Then, in Section 3, we detail our system. In Section 4, we show and discuss some results of generated EPG. Finally, we conclude this paper and propose future work in Section 5.

2. EPG GENERATION AND PRESENTATION ARCHITECTURES
This section presents existing approaches for the generation, delivery and presentation of EPG. We focused only on approaches where a declarative language is used to describe the presentation. From the literature [15] or from existing products, we propose to classify the approaches into two main extreme approaches, depicted in Figure 1. We note that this classification is actually rather generic and could be applicable to the generation, streaming and presentation of many types of metadata other than EPG. For both approaches, we assume that the metadata comes from either some broadcaster content management system or is dynamically produced, as for live events. In both cases, we also

assume that a metadata filtering mechanism can be used at the server-side or at the client-side either driven by direct user interaction (e.g. show sport programs only) or based on user modeling systems.

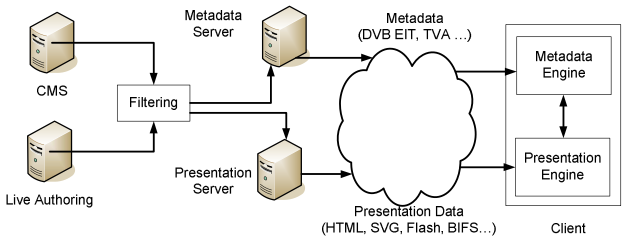

Figure 1 – Alternative approaches for the generation, delivery and presentation of EPG

2.1 The Metadata-Driven Approach

In this first approach, the EPG information is delivered in the form of metadata, and no information is sent about how this metadata should be presented. The receiver is in charge of presenting the metadata in a suitable form. This process can be driven by some specific generation software which creates a declarative presentation (e.g. using XML Transformations). This presentation is then passed to the presentation engine for display, typically a browser. Alternatively, this process can be driven directly by the presentation engine. In that case, the presentation engine loads some presentation templates, and then fills the template with data from the metadata engine. Such metadata, e.g. the name of the current program, can be pulled using some specific API (such as the Joost Widget API [7]); or if the metadata engine is implemented as a web server, by using the XMLHttpRequest standard (XHR) [6]. In an other alternative, the metadata can be pushed to the presentation engine.

The advantages of the metadata-driven approach are the following. First, since the presentation data is generated at the receiver side, it can be easily adapted to the terminal characteristics (e.g. screen size, input methods) and to the user preferences (metadata of interest, user model) without privacy concerns. Second, since this approach delivers ‘raw’ metadata without presentation data, it is efficient in terms of bandwidth. Third, from a server point-of-view, the implementation is simple since it does not have to deal with presentation data.

The short-comings of this approach on the other hand are the following. First, in order to display the presentation, the receiver must implement a metadata engine and the associated interface to the presentation engine. In the context of the growing number of sources and formats of data, it means that the receiver must implement a metadata engine capable of handling several types of metadata. An alternative would be to aggregate all the metadata in one unique form. In some sense, this alternative would transfer some intelligence from the client to the server. Second, in this extreme approach, the presentation look and feel is driven by hardcoded presentation parameters stored on the client. It means that the presentation style cannot change over time (changing colors, fonts or even navigation scheme). Third, since each receiving device can use presentation different parameters, the metadata broadcaster has no control over how the metadata will be presented on the receiving terminal. We believe this is an important problem which leads towards using an approach where presentation data is sent to the receiver.

2.2 The Presentation-Driven Approach

In this second extreme approach, the broadcasting server aggregates the different sources of metadata, transforms them into presentation data, and finally delivers the presentation data to the presentation engine of the client. Oppositely to the previous approach, no ‘raw’ metadata is sent to the client. In this approach, the presentation data can be queried from or pushed by the server.

This approach has the following advantages. First, the client does not implement a metadata engine. Its footprint is therefore smaller. Second, with this approach, it is possible to update the presentation of the EPG for example to change the look of the EPG during the Christmas time or the Olympics, or to change the navigation method. Third, in this approach, the presentation can be displayed in the receivers as the author decided.

However, this approach also has limitations. First, it requires more processing at the server side. Second, the diversity of rendering devices hardens the task of adapting the presentation to the terminal characteristics (presentation formats, screen size, input methods). Third, since no metadata is sent, there are cases where the loss of semantics in the presentation data prevents the client from performing semantic filtering (e.g. showing only Sports) or makes it more complex. Finally, since the metadata is transformed into presentation data, a higher bandwidth is required to transmit the presentation style and navigation paradigm.

2.3 Summary

Table 1 summarizes the comparison between the previous approaches. Obviously between those approaches, many hybrid approaches can exist. One interesting approach could consist in sending some metadata along with some presentation information to control the presentation (at the cost of bandwidth occupancy).

Table 1 – Comparison between the metadata and presentation-driven approaches for the generation and presentation of EPG

	Metadata-driven	**Presentation-driven**
Server Complexity	less	more
Client Complexity	more	less
Bandwidth required	less	more
Adaptation handling	simple	complex
Client-side semantic filtering	simple	complex
Presentation control	no	yes

There are several options to do that. One could leverage the ability of the languages like XML to mix data. In that case, a server could deliver a mixed document with presentation data, EPG metadata and the logic to build the final presentation. Another option would be to use separate the data into files or streams and to link the presentation data to the metadata. Files could be acquired using XHR or using file delivery mechanisms such as the DVB Object Carousel [9] or the FLUTE protocol [10]. Streams could use the MPEG-7 BiM [12] standard for metadata and for the presentation data, 3GPP DIMS (MPEG-4 LASeR) or MPEG-4 BIFS [5].

3. THE PINGO SYSTEM

Our system, called PINGO, builds upon the previous analysis. The use case for the PINGO System is the in-house redistribution of television signals, including the EPG information. Indeed, the deployment of mobile TV and terrestrial TV do not allow, for now, for correct in-door reception of the signal by devices like mobile phones or Portable Media Players. In the PINGO system, a centralized device, named PINGO Box, standalone or integrated in a Set Top Box or in an Internet Service Provider Box, receives the TV signal. The raw EPG data is extracted from this broadcast TV signal (mobile or fix) and possibly from other internet sources. The PINGO box redistributes the EPG indoor for other mobile devices (phones, PMP) connected via WiFi.

In this scenario, there is no strong bandwidth constraint for the delivery of EPG data. Additionally, given the requirements that we want to minimize the mobile client developments and processing requirements, and we want to keep fine-grain synchronization between raw the EPG data and the presentation form, we decided to use a streaming delivery of EPG data in a presentation form. Given its mobile standard status, we selected the 3GPP DIMS format.

The EPG raw data is therefore transformed, based on templates, into a stream of presentation information which is updated regularly as new EPG data arrives. In this system, since the source of EPG data is mainly the broadcast channel, the EPG data is delivered in a push mode, using scene updates, as opposed to traditional AJAX pull mode.

The system functions as depicted in Figure 3. First, a designer creates two templates. The first template, called the Main Template, defines the general look and feel of the EPG. This template also handles the navigation between channels and display of information for the current channel. This template is initially empty of programs, but it identifies an entry point for the programs to be inserted, called the Event Dictionary. The second template, called the Event Template, defines the structure of a TV Event or TV Program. It is void of data but contains several entry points for the Event Information to be inserted (e.g. the name of the program, start time, duration, description …). These two templates are written in the target presentation language. Once written, these templates are provided to a server.

The server (in the PINGO Box) is in charge of three tasks: first, sending the initial empty presentation; second, cloning the Event template, filling it in based on the EPG data it receives and sending the result in the form of a presentation update; and third, aggregating the presentation updates to provide a complete presentation for clients who have not joined the streaming session from the beginning or for those (e.g. Web browsers) who just support SVG (not DIMS).

The client in this system only implements a 3GPP DIMS player, it receives an initial scene which displays an empty EPG and then receives updates that progressively add new programs, replace the current time, or delete old programs. Figure 2 shows one possible scene structure. The scene is made of some Javascript code to create, at initialization, the layout of the EPG based on the terminal characteristics. It also contains the event dictionary on which updates are applied. When updates are received, some Javascript code is executed to determine if the program should be displayed based on the presentation time and the program time, on previous user interactions and on the current EPG view. In this example, the whole EPG navigation or channel switching is realized using Javascript code.

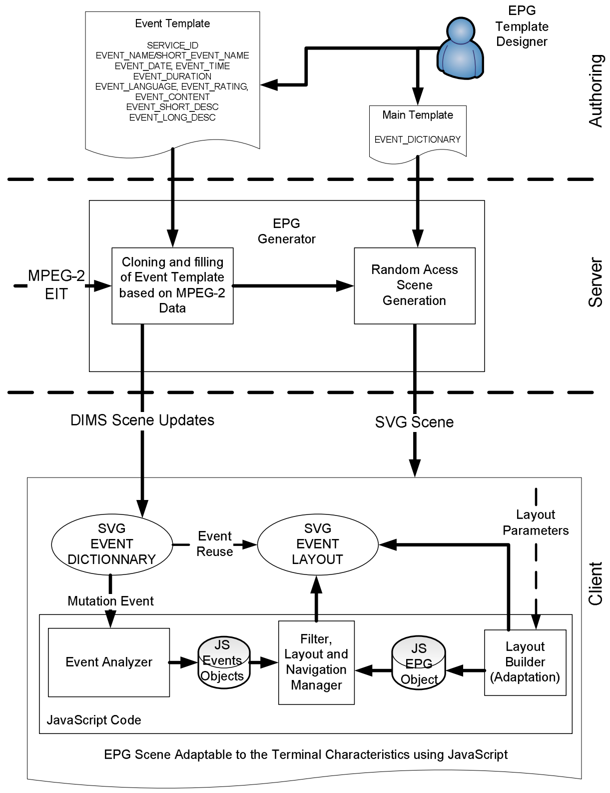

Figure 2 – Description of the end-to-end chain for the creation, delivery and presentation of EPG data in the PINGO System

4. DISCUSSIONS AND RESULTS

In this approach, a few points should be noticed. First, the description of a program is transmitted in the target presentation language (not in a dedicated metadata language), and then analyzed to produce the Javascript equivalent in order to be able to do client-side filtering and navigation. Programs are not described using some generic EPG XML format. This choice has been made to limit the client-side processing because otherwise if the program were sent using a generic XML format, the client would have had to create the SVG objects that represent the program using Javascript. Previous experiments showed us that the creation of objects with Javascript is slower than the creation of objects upon reception of scene updates. Similarly, in our system, we have tried to minimize the read/write access from the Javascript to the scene and we kept in Javascript only algorithmic operations (search, sort …). Only minor attribute changes are done to the SVG scene using Javascript.

This system has been implemented and validated using the GPAC DIMS Player [13]. Figure 3 shows snapshots of the SVG/DIMS EPG produced by our system. Figure 4 demonstrates its usage of three mobile devices (SPV C500, Samsung i780 and Glofiish V900) running Windows Mobile and the GPAC player. Similar experiments have been also made using the MPEG-4 BIFS language.

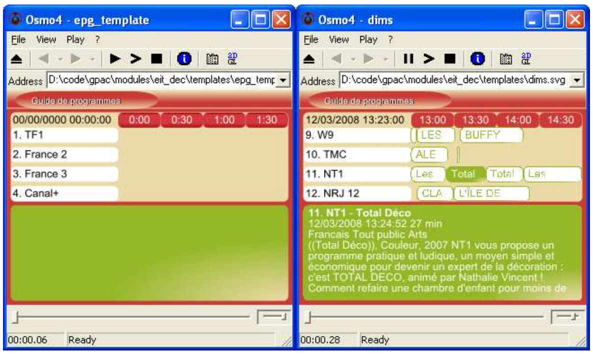

Figure 3 – Empty (left) and filled (right) PINGO Electronic Program Guide

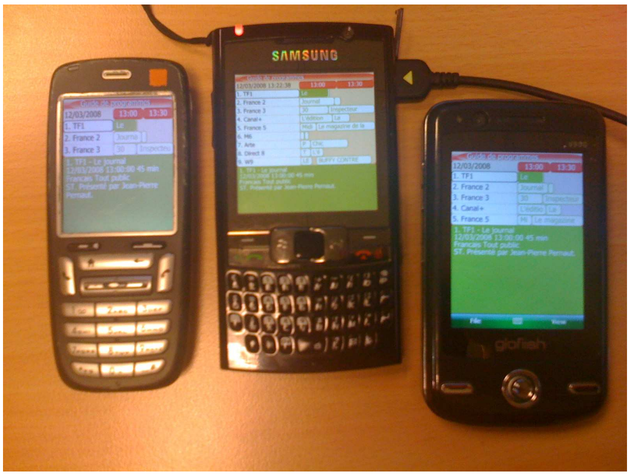

Figure 4 – Adapted display of the PINGO Electronic Program Guide on three different mobile devices

5. CONCLUSION

In this paper, we have exposed the existing approaches for the generation and display of electronic program guides for interactive TV. We have discussed the pros and cons of these approaches. We have presented our scenario and argued for the need for an approach based on streamable presentation languages. This approach allows separating the decoding or interpretation of raw EPG data from their presentation, thus allowing the display of EPG on constrained devices. In our system, the use of declarative presentation languages enables the designer control over the presentation and its adaptation to the device characteristics. We believe that this approach, applied here to EPG, is actually very generic and could be applied to the presentation of any type of metadata, through the use of templates. In future work, we will investigate several improvements to this system: the ability to have more input formats (web feeds), to support more output formats (XHTML). We will also work on improvement of the adaptation features of our prototype.

6. ACKNOWLEDGMENTS

This work has been partly financed by the French General Directorate for enterprises and the French Business Cluster for Digital Content.

7. REFERENCES

[1] "Digital Video Broadcasting (DVB); Specification for Service Information (SI) in DVB systems", ETSI EN 300 468 V1.7.1

[2] "Digital Video Broadcasting (DVB); IP Datacast over DVB-H: Electronic Service Guide", ETSI TS 102 471 V1.2.1

[3] ANT Fresco Browser, http://www.antlimited.com

[4] "Dynamic and Interactive Multimedia Scenes (DIMS)", 3GPP TS 26.142, http://www.3gpp.org/ftp/Specs/html-info/26142.htm

[5] "Information technology - Coding of audio-visual objects - Part 11:Scene Description and Application Engine", ISO/IEC IS 14496-11

[6] "The XMLHttpRequest Object", W3C Working Draft 15 April 2008, http://www.w3.org/TR/XMLHttpRequest/

[7] Joost Widget API, Runtime Engine Information, http://dev.joost.com/widgets/api/engine.html

[8] The XML TV Project, http://wiki.xmltv.org/index.php/XMLTVProject

[9] "DVB Data Broadcasting Implementation Guidelines", ETSI TR 101 202 V1.2.1

[10] "File Delivery over Unidirectional Transport", RFC 3926, http://www.faqs.org/ftp/rfc/rfc3926.txt

[11] "Broadcast and On-line Services: Search, select, and rightful use of content on personal storage systems (TV-anytime)", ETSI TS 102 822

[12] "Information technology - Multimedia content description interface - Part 1: Systems", ISO/IEC IS 15938-1

[13] Le Feuvre, J., Concolato, C., and Moissinac, J. 2007. GPAC: open source multimedia framework. In Proceedings of the 15th international Conference on Multimedia (Augsburg, Germany, September 25 - 29, 2007). MULTIMEDIA '07. ACM, New York, NY, 1009-1012.

[14] Cruickshank, L., Tsekleves, E., Whitham, R. and Hill, A. 2007. Making Interactive TV easier to use: Interface design for a second screen approach. The Design Journal, 10, 3.

[15] Lee, H., Yang, S.-J., Kim, J.-G, and Hong, J. 2005. Personalized TV services based on TV-anytime for PDR. In Proceedings of the International Conference on Consumer Electronics (January 8-12, 2005). ICCE 05, 115-116.

[16] Harrison, C., Amento, B., and Stead, L. 2008. iEPG: an ego-centric electronic program guide and recommendation interface. In Proceeding of the 1st international Conference on Designing interactive User Experiences For TV and Video (Silicon Valley, California, USA, October 22 - 24, 2008). UXTV '08, vol. 291. ACM, New York, NY, 23-26.

[17] Cesar, P., Bulterman, D. C., and Jansen, A. J. 2008. Usages of the Secondary Screen in an Interactive Television Environment: Control, Enrich, Share, and Transfer Television Content. In Proc. of the 6th European Conference on Changing Television Environments (Salzburg, Austria, July 03 - 04, 2008). M. Tscheligi, M. Obrist, and A. Lugmayr, Eds. Lecture Notes In Computer Science, vol. 5066. Springer-Verlag, Berlin, Heidelberg, 168-177.