A Test-Bed for the Dynamic Adaptive Streaming over HTTP featuring Session Mobility

Christopher Müller and Christian Timmerer
Klagenfurt University, Multimedia Communication
Universitätsstraße 65-67
9020 Klagenfurt am Wörthersee, Austria
+43 (0)463 2700 3600

{*firstname.lastname*}@itec.uni-klu.ac.at

ABSTRACT

In this paper, we present a multimedia test-bed enabling session mobility in the context of the emerging ISO/IEC MPEG standard, Dynamic Adaptive Streaming over HTTP (DASH). In general, session mobility is defined as the transfer of a running streaming session from one device to another device where it may need to be consumed in an adaptive way. The two main challenges are: (1) taking into account the new context of the device (e.g., capabilities) to which the session is transferred and (2) performing the actual transfer in a seamless and interoperable way. Our system addresses both challenges supported by a prototype implementation integrated into VLC. In anticipation of the results we can conclude that interoperability is achieved adopting existing standards while the performance of the system does not depend on these standards. That is, the modules responsible for the performance are usually not defined within such standards and left out for competition. However, our system is designed in an extensible way and is able to accommodate this fact.

Categories and Subject Descriptors

H.5.1 [**Multimedia Information System**]: Video

General Terms

Documentation, Design, Standardization.

Keywords

Dynamic Adaptive Streaming over HTTP, MPEG-21 Digital Item Declaration, MPEG-21 Session Mobility, Test-bed.

1. INTRODUCTION

The streaming of media resources over the Hypertext Transfer Protocol (HTTP) is nowadays omnipresent and has become a de-facto standard on the Internet for two reasons. First, reasonable Internet connectivity (i.e., in terms of bandwidth for media content) is nowadays available anywhere, anytime, and almost on any device. Second, the usage of HTTP does not cause any NAT/firewall issues as it is the case with other media transport protocols like RTP/RTSP. In order to provide interoperability among vendor implementations, standards developing organizations such as MPEG have recognized the need for an international open standard in order to reduce the number of already existing proprietary solutions (cf. Section 5 for details). The MPEG standard which is used in this paper is referred to as ISO/IEC 23001-6 entitled Dynamic Adaptive Streaming over HTTP (DASH) [1]. It is based on 3GPP Adaptive HTTP Streaming (AHS) which has been also adopted by the Open IPTV Forum (OIPF) as a baseline standard.

In this paper we address – in the context of DASH – the need for leveraging existing media repositories (and their representation formats) and enabling session mobility in an interoperable way. Therefore, we have developed a test-bed for DASH which enables session mobility in an interoperable way and may be added on top of existing media repositories. In particular, existing media repositories utilize XML data formats in order to describe the content and the relationship among the assets within the repository. One well-known example is Universal Plug and Play's (UPnP) Content Directory which specifies DIDL-Lite [2] that is derived from a subset of MPEG-21 Digital Item Declaration Language (DIDL) [3]. Both DIDL-Lite and DIDL provides means to describe the relationship among various information assets that may be consumed as such by a user and which are collectively referred to as Digital Item (DI). In this paper we leverage the fact that media resources are described by existing formats such as DIDL(-Lite) and introduce a method that enables its usage within the emerging DASH standard. Additionally, we demonstrate the use case of session mobility which enables the seamless session transfer from one device to another per user request, everything fully interoperable by utilizing existing as well as emerging (MPEG) standards. We have uploaded a demo video to YouTube (http://www.youtube.com/user/timse7) in order to demonstrate our system in action.

The remainder of the paper is organized as follows. Background about the technology standards used in this paper is presented in Section 2. Section 3 defines the system architecture and Section 4 provides the details about our implementation. Section 5 provides an overview of related work and the paper is concluded in Section 6 including future work items.

2. BACKGROUND

2.1 Dynamic Adaptive Streaming of HTTP

During its 93rd meeting, MPEG evaluated 15 submissions from 20 organizations (including companies, research institutions, and

MMSys'11, February 23–25, 2011, San Jose, California, USA.

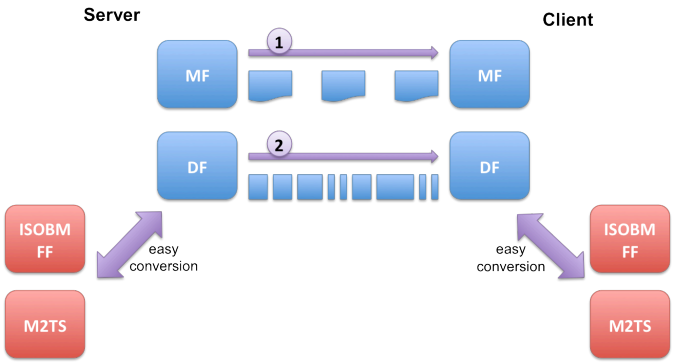

Figure 1. MPEG-DASH System Architecture.

universities). The submissions provided technologies for the HTTP streaming of MPEG media in the following areas:

- Manifest File (MF), i.e., playlist, media presentation description, etc. which is mostly based on XML.
- Delivery Format (DF) as extensions/specializations of ISOBMFF and M2TS.

The system architecture is depicted in Figure 1 and based on that MPEG started a new work item called Dynamic Adaptive Streaming over HTTP (DASH) which will be Part 6 of MPEG-B (i.e., ISO/IEC 23001-6).

On request, the manifest file will be provided to the client in order to initiate the session (cf. step-1 in Figure 1). The client will parse the manifest file and request individual segments compliant to the delivery format using HTTP and according to the information found in the manifest file (cf. step-2 in Figure 1). For the manifest file, DASH adopted the Media Presentation Description (MPD) as defined by 3GPP AHS [4] as a starting point. The MPD follows a data model comprising a sequence of one or more consecutive non-overlapping periods for which one or more representations may be available. A single representation refers to a specific media following certain characteristics such as bit rate, frame rate, resolution, etc. Furthermore, each representation consists of one or more segments that actually describe the media and/or metadata to decode and present the included media content. The actual delivery format is also based on 3GPP files which are derived from the well-known ISO base media file format. Additionally, extensions to the MPEG-2 Transport Stream (TS) will be also defined within DASH in order to support its transport over HTTP.

2.2 MPEG-21 Digital Item Declaration

The aim of the MPEG-21 standard (ISO/IEC 21000), the so-called Multimedia Framework, is to enable transparent and augmented use of multimedia resources across a wide range of networks, devices, user preferences, and communities, notably for trading (of bits). That is, to support the transaction of Digital Items among Users.

A Digital Item is a structured digital object with a standard representation, identification, and metadata. The standard representation of Digital Items is defined by a model which describes a set of abstract terms and concepts and is expressed by the XML Schema based Digital Item Declaration Language (DIDL) [3]. The resulting XML document conformant to DIDL is called Digital Item Declaration (DID). The DID may contain several building blocks as defined in DIDL which defines the structure of the Digital Item. A brief overview of the most important building blocks is given in this paper, for further details the reader is referred to [3][5].

The *Item* comprises a grouping of sub-items or components. In general, an item can be considered as a declarative representation of a Digital Item. Note that an item without sub-items can be considered a logically indivisible work and an item that does contain sub-items can be considered a compilation.

The *Component* defines a binding of a multimedia resource to a set of descriptors which provides information related to all or parts of the resource. These descriptors will typically contain control or structural information about the resource such as bit rate, character set, start points, or encryption information.

A *Descriptor* associates information with the enclosing element, i.e., its parent (e.g., item) or following sibling (e.g., component). The information can itself be a component (e.g., thumbnail of an image) or a textual statement.

A *Resource* is defined as an individually identifiable asset such as a video, audio clip, image, or textual asset. Note that the resource must be locatable via an unambiguous address.

Digital Items are configurable through the so-called choice/selection mechanism. A *Choice* describes a set of related *Selections* which can affect the configuration of an item. As such it provides a generic and flexible way for multimedia content selection based on certain criteria defined by the Digital Item author. Such criteria may include rights expressions and/or usage environment constraints.

Finally, DIDL allows to associate annotations and assertions to its building blocks using the equally named elements, i.e., *Annotation* and *Assertion* respectively.

2.3 MPEG-21 Digital Item Adaptation

Part 7 of MPEG-21, entitled Digital Item Adaptation (DIA) [6], addresses issues that are related to Universal Multimedia Access (UMA) [7] which refers to the ability to seamlessly access multimedia content from anywhere, anytime, and with any device. Due to the heterogeneity of terminals and networks and the existence of various coding formats, the adaptation of the multimedia content may be required in order to support the requirements of the consuming user and his/her environment. Therefore, MPEG-21 DIA specifies description formats (also known as tools) to assist with the adaptation of Digital Items. In the context of this paper, the MPEG-21 DIA Session Mobility (SM) tool is adopted which enables the interoperable transfer of a session from one device to a second device. Therefore, one needs to capture the configuration state of a Digital Item (i.e., the instantiation of the choices and selections) as well as the DASH application state (i.e., the representation and the segment as well as the time position within the segment) and transfer this information to the second device.

3. SYSTEM ARCHITECTURE

3.1 Composition of Media Presentation

The Composition of Media Presentation (CMP) description comprises another layer that is added on top of the MPD and provides means for the selection of a specific configuration (e.g.,

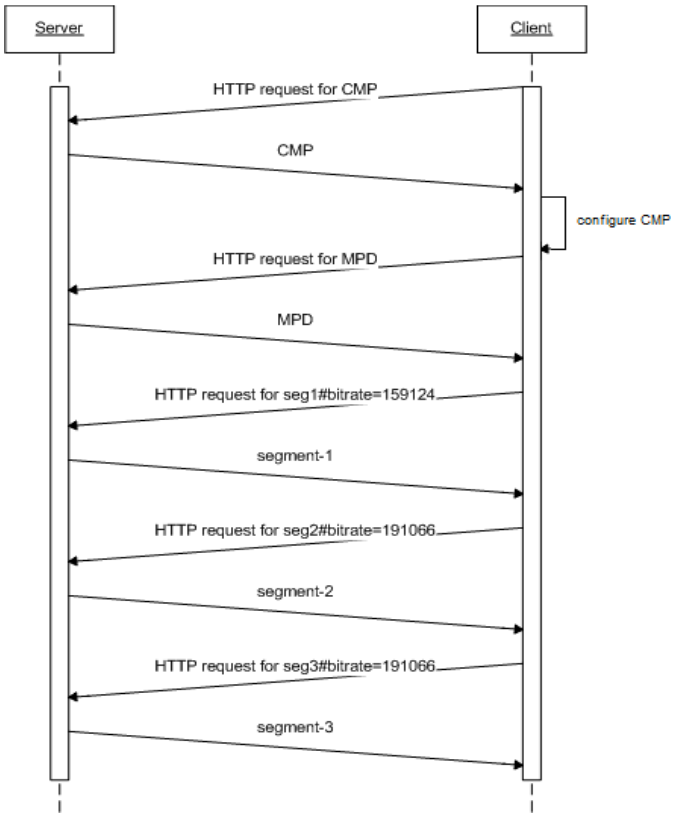

Figure 2. Sequence Diagram for CMP-enabled DASH.

codec selection, subtitles, different views, etc.) prior to the delivery of the actual MPD. In particular, a CMP description is an XML document which specifies initial user and device options like camera angel, audio language, subtitle language, resolution, decoding algorithm, and any other possible choice that the user or the device may decide. The CMP is very flexible and, thus, overlaps with the MPD are possible but not advisable.

The CMP description is compliant to MPEG-21 DID and utilizes its choice/selection mechanism to allow a runtime configuration of the DI which makes the static XML dynamic [5]. A full example of such a CMP can be found at [8]. Upon receipt of a CMP description the DI is configured according to the usage environment which may be accomplished manually by the user, automatically by the device, or semi-automatically (i.e., some choices may be presented to the user whereas some are resolved by the device). A more detailed walkthrough of CMP-enabled DASH is provided in the next section.

3.2 CMP-enabled DASH

The sequence diagram of CMP-enabled DASH is shown in Figure 2. At the beginning the client requests the CMP from the server. In this case the request/response of the CMP is based on HTTP but any other protocol or mechanism could be used. After the successful download of the CMP the client configures the DI based on the CMP in such a way that it fulfills the requirements of the device and the user. This configuration typically includes the elimination of choices and selections that are not supported by the capabilities of hardware or software modules of this device (e.g., decoding or decryption algorithm). Furthermore, the user may select her/his preferred configuration that fulfills his/her needs. For example, the user is able to choose between different movies, different camera angles (e.g., of a football game), different languages, or if subtitles and in which language should be used. In this sense, the configuration of the CMP comes close to a DVD menu where it is also possible to configure several options as mentioned before.

Such a configuration of a DI may contain several iterations as some selections may affect other/subsequent choices (and vice versa). However, this kind of functionality is fully supported by the CMP based on MPEG-21 DID. After the client has successfully configured the CMP according to its needs, the corresponding MPD is requested in the same way as the CMP. The MPD is used to perform the dynamic adaptive streaming (over HTTP) of the media resources which is associated to the CMP configuration. The MPD is an XML document compliant to DASH and may describe different bitrate representations of the media resource enabling dynamic adaptive streaming over HTTP. Each representation is divided into small parts called segments. These segments are located on a regular Web server and accessible per HTTP.

The MPD contains fully qualified URLs to these segments and it expresses the relationship between the segments and the corresponding representation. Some examples for MPDs can be found at [9]. After the download of the MPD the client is able to start with the download of the first segment at a lower bitrate as shown in Figure 2, e.g., to reduce the start-up delay. At segment boundaries, the client is able to switch to segment of different representations, e.g., to higher or lower bitrates depending on the actual download rate of the segments. In this way dynamic adaptive streaming over HTTP is achieved.

3.3 Session Transfer during DASH

3.3.1 General Considerations

At some point in time during a DASH session the user may decide to transfer the session from one device to another device. Obviously, the session should be started with the same configuration and the same point in time where the user has initiated it on the first device. Additionally, the user should also be able to reconfigure the initial setup from the first device at the start of the session on the second device where the session is transferred. An example case for such a user reconfiguration is that the user decides that it is not necessary to display the subtitles on a mobile device because they are not readable on such a small screen. On the other hand, an automatic reconfiguration may become necessary in cases where the second device does not fully support the requirements of the configuration on the first device (e.g., different decoding or decryption algorithm). One common reconfiguration may concern the spatial resolution when transferring the session from a TV screen to a mobile device and vice versa.

3.3.2 Session Transfer with MPEG-21

Our approach is based on the session mobility tool specified in MPEG-21 DIA as introduced earlier and now described in more detail for DASH.

The session mobility architecture based on MPEG-21 DIA is depicted in Figure 3. In this scenario, the user decides during the DASH session to switch to another device (i.e., from the laptop to the mobile device). Therefore the client on the laptop has to save the current state of the CMP, the media play time and the state of the video player. For this purpose MPEG-21 has introduced the Context Digital Item (CDI). It is based on XML and it describes

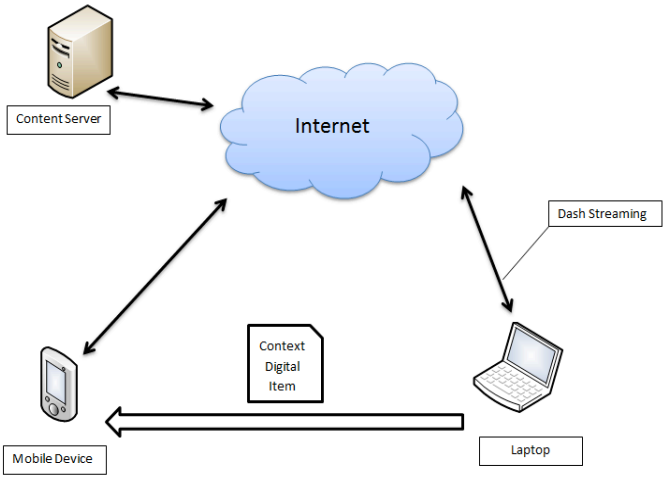

Figure 3. Session Mobility Architecture.

the user's current state of interaction with a Digital Item (DI) and it is also possible to describe application-specific information [5] (i.e., the DASH context with respect to the segment which is currently decoded/displayed).

In our system the CDI describes the state of the CMP which is based on a DI. The play time and play state is stored in the CDI as application-specific information as mentioned before. Also the path to the CMP which is a fully qualified URL is stored as application-specific information in the CDI. Thus, a CDI contains all information that is required to reconstruct a session at another device. Like shown in Figure 3 the laptop generates the CDI upon user request and transfers it over the network to the mobile device. The next step is that the mobile device has to extract the information from the CDI that is important to reconstruct the session. At this point the most important information of the CDI is the path to the CMP, because the CDI contains only the selected configuration of the CMP which is useless without the actual CMP.

Therefore, the mobile device needs to download the CMP and configure it with the information of the CDI. Please note that the user and also the device are able to reconfigure the CMP as discussed in the previous section. This is not really a problem because the same will be done during the startup of DASH. The only thing that we had to change was the initial configuration of the CMP. Another important thing is the play time and play state. This information will be used after the download of the MPD. As also mentioned in the previous sections the MPD contains the bit rate representations of the media and the corresponding segments. The MPD describes also the length of the whole media and the length of each individual segment. With that information and the play time of the CDI it is possible to calculate the offset and to select the appropriate segment, so that the session could start at the same position where the user has stopped on the other device.

4. IMPLEMENTATION

This section describes our implementation of CMP-enabled DASH and the session transfer based on the MPEG-21 DIA. As indicated before our implementation consists of two applications: (a) playback of CMP-enabled DASH content and (b) session mobility.

Our first application, the playback of CMP-enabled DASH content is based on the well-known VideoLan Client (VLC) [10]. We have used a common application like VLC as basis because it should be easier for the user to accept DASH within a known environment. However, VLC does not support DASH out of the box and, thus, we had to include this functionality into VLC. At the beginning of writing this paper we were not aware of any other player supporting DASH like specified by MPEG (cf. Section 5 for more details).

4.1 VLC Architecture

This section describes the VLC architecture in general. VLC is built in a highly modular way around the VLC core library called LibVLC. At each layer it is very easy to build an own module, because modules always follow the same structure. In general one could say that a module consists of two major parts.

The first part of a module consists of a module description that is made out of macros which describe the behavior and the capabilities of a module. Also the priority of the module will be specified in this part including some other functionality like variables that are specific to the module. These variables are also important for our system because we use them to add a command line parameter named *contextDI*. This command line parameter will be used to start the application with a CDI, for the pre-configuration of the CMP.

The second part of a module is the callback part. A module always has to implement two callback functions named *open(vlc_object_t *)* and *close(vlc_object_t *)* that are specified in the module description part. The *open(vlc_object_t *)* function will be called by the VLC core to initialize the module. The *close(vlc_object_t *)* function should basically uninitialize the module and free all memory that has been allocated by the *open(vlc_object_t *)* function.

The main layers of VLC:

- *Interface*: Contains all modules that have something to do with the user interaction like stop, play, forward and backward key presses or mouse events and playlist management.
- *Access*: Modules that are able to open files or streams are located at this layer. The access modules supply the demux modules with data.
- *Demux*: At this layer different formats will be demultiplexed such as MP4, AVI, MKV, etc.
- *Decoder*: This layer provides modules that are able to decode the demultiplexed data from the demux layer.
- *Output*: This layer simply displays the decoded data on the screen and it is also possible to stream to another device from this layer.

4.2 CMP-enabled DASH support for VLC

In this subsection we will describe the modifications of VLC in order to support our architecture described in the previous section.

4.2.1 Interface layer

At the interface layer we have added a new tab called DASH. The tab is fully written with the common graphical library QT [11]. The main class of the tab is located at the *open_panles.cpp* file which also contains the other tabs. The class manages the whole user interaction with the DASH tab. After the user has entered the URL of the CMP into the textfield on the top of the tab, the main

class of the DASH tab initializes a new *CMPManager* class that is able to parse the CMP. The DASH tab provides the user the information about the choices that he or she can make. After the user has selected a configuration that fulfills his/her needs the DASH tab generates a CDI and passes this with the selected path of the MPD to the access plugin. Another modification that we have made at this layer is the session transfer menu. If the user right clicks on the video a menu appears that provides the user some options that relate to the media that is currently playing. We have added a new submenu called session transfer to this menu. The session transfer submenu presents the user the devices to which VLC could transfer the session.

4.2.2 Access Layer

At the access layer we have written our own plugin called MPD plugin that is able to handle DASH streams. The plugin gets the information where the MPD is located and the CDI from the DASH tab. Thus, at the beginning the plugin starts to download the MPD and after the parsing it begins with the download of the segments. It is also responsible for the representation change, i.e., when the plugin detects that more bandwidth is available it selects the next higher representation of the media and downloads the segments that relate to this representation, and the other way around if the bandwidth decreases.

4.2.3 Demux Layer

It was also necessary to make some small changes at the demux layer because VLC did not support the DASH delivery format out of the box. In our approach we have used the avformat library which is able to demultiplex MP4 and the DASH delivery format is also similar to MP4. But we had to change some things inside of the library because the offset of the second track would be false interpreted by the library. However, after this little modification it was no problem to demultiplex the DASH delivery format.

4.3 Session Transfer

This section describes the session transfer in the context of DASH with VLC.

As described before the user has to select a device from the session transfer submenu. After that the VLC variable will be changed to the hostname or address of the selected device. The change of this variable triggers the callback function that is located in the MPD plugin. The callback function gathers now the required information like play state and play time. After that it inserts this information to the CDI that the MPD plugin has received during the startup from the DASH tab. Thereafter the CDI will be send through a TCP connection to the selected device. On the selected device a service is running which waits for incoming connections. If it receives a CDI it starts VLC with the parameter *contextDI* and the path that points to the received CDI. The DASH tab perceives that VLC has been started with the parameter *contextDI*. After that it parses the CDI and downloads the CMP that is described by the CDI. Now the user could change some of the available choices within the CMP like the language or the subtitles. Probably also the device makes some changes because it needs a different decoding algorithm or resolution. Afterwards DASH starts as usual with the only difference that the play state and the play time of the received CDI will be passed to the MPD plugin. With that information the MPD plugin could than start the stream at the point where the user has stopped the stream on the other device.

5. RELATED WORK

To start with we provide an overview of related work in the area of HTTP streaming.

Recently, 3GPP already specified Adaptive HTTP Streaming (AHS) [4] which defines a Media Presentation Description (MDP) and extensions to the well-known ISO Base Media File Format (ISOBMFF) [12]. The former is an XML document providing a manifest/session description which enables the client to request individual media segments via HTTP. The media segments are compliant to a delivery format that has been derived from the ISOBMFF.

Adobe's Dynamic HTTP Streaming [13] is based on their own Flash media manifest and F4F file format. The former is an XML document similar to 3GPPs' MPD and the latter are MP4 fragment files, i.e., also based on ISOBMFF. However, the solution is proprietary and not compliant to 3GPP AHS.

Apple's HTTP live streaming [14] is well known for quite some time and implemented on the iPhone and similar devices. It makes use of a M3U playlist file which serves as the manifest and each media file must be formatted as an MPEG-2 Transport Stream (M2TS) [8].

Finally, Microsoft's Smooth Streaming [16] is also around for a while which utilizes a server manifest file (i.e., SMIL document) and a client manifest file (i.e., proprietary XML document). Furthermore, this approach defines a smooth streaming format (ISMV) as an extension of the ISOBMFF. Additionally, they have also provided a comparison with the solutions provided by Apple and Adobe [17].

Recently, another implementation has been provided within the GPAC project on advanced content [18][19]. It ships with a similar functionality to our implementation with respect to DASH. Furthermore, it is open source and comes with its own player, i.e., no integration with VLC or any other existing player and no support for CMP. There is also an early version of reference software – not yet publicly available – which comprises command line tools for creating segments and requesting segments based on the MPD.

We respect to pure research results, a few papers are worth to mention in this context. Riiser et al. defined a low overhead container format for adaptive streaming [20] that proposes an alternative to the MPEG family of delivery formats (i.e., M2TS, ISOBMFF, and derivations thereof) for the streaming over HTTP. Kuschnig et.al. has performed an evaluation of rate-control algorithms in the context of dynamic adaptive streaming over HTTP, specifically when using scalable media resources. Finally, Rong et al. [22] and De Keukelaere et al. [23] (and related publications) provide first publications in the area of MPEG-21 session mobility and have been used as an inspiration for starting this work item in the context of DASH.

6. CONCLUSIONS AND FUTURE WORK

In this paper we have described our test-bed for DASH featuring session mobility. It is based on international open standards and fully integrated into the well-known VLC. In particular, it facilitates MPEG-21 Digital Item in order to leverage existing media repositories such as UPnP and provides a testing framework for actual dynamic adaptive streaming algorithms including the possibility for transferring a running session from one device to another.

Future work includes the integration and comparison of various algorithms for the dynamic adaptive streaming such as [20] and [21] as well as those that will pop up at MMSys'11. Additionally, we plan to implement the actual connector to the UPnP repositories in order to be used in a home entertainment environment. Finally, once our code is stable enough and DASH is technically frozen we will consider submitting our implementation to the VLC developers in order to become open source.

7. ACKNOWLEDGMENTS

This work was supported in part by the EC in the context of the ALICANTE project (FP7-ICT-248652).

8. REFERENCES

[1] ISO/IEC CD 23001-6. 2010. Information technology -- MPEG systems technologies -- Part 6: Dynamic adaptive streaming over HTTP (DASH) (Guangzhou, China, Oct. 2010)

[2] UPnP Forum. 2006, ContentDirectory:2 Service Template Version 1.01 (May 2006) Available: http://www.upnp.org/specs/av/UPnP-av-ContentDirectory-v2-Service-20060531.pdf (last access: Dec. 2010)

[3] Burnett, I.S., Davis, S.J., Drury, G.M. 2005. MPEG-21 Digital Item Declaration and Identification – Principles and Compression. IEEE Transactions on Multimedia. 7, 3 (Jun. 2005), pp. 400-407.

[4] 3GPP TS 26.234. 2010. Transparent end-to-end packet switched streaming service (PSS); Protocols and codecs.

[5] Burnett, I.S., Pereira, F., Van de Walle, R., Koenen, R. 2006, The MPEG-21 Book, Wiley & Sons.

[6] Vetro, A, Timmerer, C. 2005. Digital Item Adaptation: Overview of Standardization and Research Activities. IEEE Transactions on Multimedia. 7, 3 (Jun. 2005), pp. 418–426.

[7] Vetro, A., Christopoulos, C., Ebrahami, T., Eds. 2003. Special Issue on Universal Multimedia Access. IEEE Signal Processing Magazine. 20, 2 (March 2003)

[8] Composition of Media Presentation (CMP) examples: http://www-itec.uni-klu.ac.at/~cmueller/adaptivestreaming/cmp/ (last access: Dec. 2010).

[9] Media Presentation Description (MPD) examples: http://www-itec.uni-klu.ac.at/~cmueller/adaptivestreaming/mpd/ (last access: Dec. 2010).

[10] VLC: open-source multimedia framework, player and server, http://www.videolan.org/vlc/ (last access: Dec. 2010).

[11] QT: cross-platform application and UI framework, http://qt.nokia.com/products/ (last access: Dec. 2010).

[12] ISO/IEC 14496-12:2008. Information technology -- Coding of audio-visual objects -- Part 12: ISO base media file format.

[13] Adobe HTTP Dynamic Streaming, http://www.adobe.com/products/httpdynamicstreaming/ (last access: Dec. 2010).

[14] Pantos, R., May, W. 2010. HTTP Live Streaming, IETF draft (Jun. 2010) http://tools.ietf.org/html/draft-pantos-http-live-streaming-04 (last access: Dec. 2010).

[15] ISO/IEC 13818-1:2007. Information technology -- Generic coding of moving pictures and associated audio information: Systems.

[16] Microsoft Smooth Streaming, http://www.iis.net/download/smoothstreaming (last access: Dec. 2010).

[17] Adaptive Streaming Comparison, http://learn.iis.net/page.aspx/792/adaptive-streaming-comparison (last access: Oct. 2010).

[18] Le Feuvre, J., Concolato, C., Moissinac, J.-C. 2007. GPAC: Open Source Multimedia Framework. In Proceedings of the ACM Multimedia 2007 (Augsburg, Germany, Sep. 2007)

[19] GPAC Project on Advanced Content, http://gpac.sourceforge.net/ (last access: Oct. 2010).

[20] Riiser, H., Halvorsen, P., Griwodz, C., Johansen, D. 2010. Low overhead container format for adaptive streaming, In Proceedings of the First Annual ACM SIGMM Conference on Multimedia Systems (Scottsdale, Arizona, USA, Feb. 2010), pp. 193–198.

[21] Kuschnig, R., Kofler, I., Hellwagner, H. 2010. An Evaluation of TCP-based Rate-Control Algorithms for Adaptive Internet Streaming of H.264/SVC. In Proceedings of the First Annual ACM SIGMM Conference on Multimedia Systems (Scottsdale, Arizona, USA, Feb. 2010), pp. 157–167.

[22] Rong, L, Burnett, I. S. 2004. Dynamic multimedia adaptation and updating of media streams with MPEG-21, In Proceedings of the First IEEE Conference on Consumer Communications and Networking (Jan. 2004) pp. 436-441.

[23] De Keukelaere, F., De Sutter, R., Van de Walle, R. 2005. MPEG-21 session mobility on mobile devices. In Proceedings of the 2005 International Conference on Internet Computing. (Las Vegas, NV, USA, May 2000)