LASeR: the MPEG Standard for Rich Media Services

Jean-Claude Dufourd, Olivier Avaro (Streamezzo)

Cyril Concolato (Ecole Nationale Supérieure des Télécommunications)

to be published in IEEE Multimedia

1. Abstract

LASeR, formally known as ISO/IEC 14496-20 (MPEG-4 Part 20), is the new Rich Media standard dedicated to the mobile, embedded and consumer electronics industries specified by the MPEG standardization group.

LASeR enables a fresh and active user experience on constrained networks and devices based on enriched content, including Audio, Video, Text, and Graphics. It addresses the requirements of the end-to-end rich media publication chain: ease of content creation, optimized rich media data delivery and enhanced rendering on all devices.

Inspired by the best concepts of state-of-the-art solutions (e.g. W3C/SVG, Macromedia Flash, ISO/IEC MPEG/BIFS), LASeR tunes and optimizes each feature required by Rich Media services to effectively fulfill the need of an efficient open standard.

This paper presents the current situation of the Rich Media services market, analyses the state-of-art technologies, gives a detailed technical overview of LASeR and illustrates the use of this new technology by giving use cases.

Sidebar: Acronyms

ISO/IEC: a joint standardization group made of the International Organization for Standardization, www.iso.org and of the International Electrotechnical Commission, www.iec.ch.

MPEG: Moving Picture Expert Group, a group within ISO/IEC, responsible for many standards including **BIFS** (Binary Format for Scenes), a binary standard 2D/3D vector graphics, **MPEG-J**, standard Java interface to BIFS scenes and the focus of this paper, **LASeR**, Lightweight Application Scene Representation, for 2D vector graphics, and **SAF**, Simple Aggregation Format, for lightweight streaming, both in part 20 of MPEG-4.

W3C: World Wide Web Consortium, www.w3.org, a standards organization responsible for **SVG** (Scalable Vector Graphics) with its **SVGT** mobile profile (SVG Tiny), **SMIL** (Synchronized Multimedia Integration Language) i.e. an XML-based standard for interactive audiovisual presentations, **CSS** (Cascading Style Sheet) i.e. a standard for document styling, **DOM** (Document Object Model), i.e. a standard software interface to access XML documents, and **XHTML** (Extensible Hyper Text Markup Language) a reformulation of HTML in XML.

ITU: International Telecommunication Union, www.itu.int, a standards organization.

3GPP: Third Generation Partnership Project, www.3gpp.org, a standards organization for 3G mobile telecommunication.

OMA: Open Mobile Alliance, www.openmobilealliance.org, a standards organization for mobile applications, formerly the WAP forum, responsible for the **WAP** (Wireless Application Protocol) standard.

IETF: Internet Engineering Task Force, a standards organization responsible for **IP** (Internet Protocol), **RTP** (Real Time Protocol) for streaming, **HTTP** (Hyper Text Transfer Protocol) for download, and many others. IETF standards are called **RFC**s (Request For Comments).

2. Introduction

2.1. A Vision on Rich Media Services

A Rich Media service is a dynamic, interactive collection of multimedia data such as audio, video, graphics and text. It ranges from a movie enriched with vector graphics overlays and interactivity (possibly enhanced with closed captions), to complex multi-step services with fluid interaction and different media types at each step. The demand for such Rich Media service is increasing at a high pace, spurred by the development of the next generation mobile infrastructure and the generalization of TV content to new environments.

Despite long lasting deployments and significant investments that have been made by the various actors of their respective industries, mobile and more generally embedded interactive services (e.g. mobile internet, interactive mobile TV) have failed to reach the masses.

There are certainly a number of reasons for this failure, some of them being conjectural (e.g. economical) or structural (e.g. lack of compelling business models). Still, from a pure user experience point of view, the technologies in place also suffers from major drawbacks:

- XHTML-like approaches (e.g. WAP, I-Mode, XHTML …), although successful on the Internet, have shown their limits to provide a simple, efficient and deterministic user experience on non-PC heterogeneous devices.
- The transposition of PC technologies to constrained devices and networks without the requested adaptation to take into account the characteristics of these environments (e.g. network latency, bandwidth, devices capabilities, screen sizes …) lead to very low browsing capacity and poorly usable services.
- In addition, the current technical approaches did not integrate from their original design the integration of audio and video media. Although they then attempted to integrate multimedia when it was recognized as a key driver of growth of services, only poor results have been achieved so far.

Even though technology is definitely not a sufficient condition of economical success, we think it is a pre-condition to the pervasive development of Rich Media services on non-PC devices:

- Using Rich Media services on embedded devices is more challenging than on a PC where various interfaces are available and homogeneously implemented (e.g. mouse, keyboard, …), and for which ergonomic concepts have been tested and validated for years. On the move, in situation where it is not always easy to interact, where time is limited, users expect to be one click away from the information they need.
- End-users have been accustomed to quality interfaces on the Web. To be successful on the embedded domain, service interfaces need to leverage the “on-line experience”.
- Finally, since these services are expected to generate revenues, the users also expect a decent level of quality, efficiency and readability as a pre-condition to pay.

2.2. State of the Art

Several technologies are competing to achieve the vision described in the previous section, among which Flash [16] is the first to analyze. Indeed, Flash is very successful on the PC. It is the current de-facto standard for distributing Rich Media content on the Internet but suffers from serious drawbacks to address efficiently other industries requirements:

- Flash is not an open standard. This is a critical issue to get massive industry support, especially on mobiles. Content creators, services operators and device manufacturers would be tied to Macromedia for the creation, distribution and playback of Rich Media content. Knowing that once deployed, a media infrastructure is hard to make evolve, the trend is to avoid proprietary solutions and promote open standards.
- Flash is a technology designed for the PC. As such, it is not suitable for the mobile environment as demonstrated by the current development of the mobile version of Flash, called Flash Lite. To match the double constraint of being compatible with its existing PC format and to fit constrained devices requirements, Macromedia had to compromise a lot on technology. The first version of Flash Lite is a downgraded version of what is available on PC. In addition, the problem of having a single vendor and a proprietary format remains.

Following this analysis, an important requirement for a Rich Media solution for mobiles is to be open and allow easy conversion from the many existing types of Flash and other proprietary content into this new standard.

Two standardization groups have tried to specify standards which would satisfy this requirement: MPEG and W3C.

MPEG-4 BIFS [03] is the first attempt of MPEG in the field of composition coding. It features innovative tools that allow the creation of multimedia content mixing 2D and 3D graphics, introduces the notion of incremental updates of the scene, enabling streaming of long running scenes, and insures a tight synchronization between the different audiovisual elements of the scene. We attempted at profiling MPEG-4 BIFS to create a small enough subset to be used on mobile phones, to no avail. The inherent content and binary encoding structure makes it inappropriate for the mobile. Instead of compromising on the technology performances, MPEG reached the conclusion that an optimum between feature richness/compression efficiency and device constraints needed to be found and decided to create a new standard for Rich Media for constrained devices.

As an alternative to the Flash proprietary solution, the W3C also made attempts to define languages for creating Rich Media content. The SMIL [10] and SVG [06],[08],[09] standards are among these languages. Both of them are getting traction in the mobile industry, where SMIL and SVG mobile profiles have been adopted by the 3GPP and OMA consortia. However, both are XML languages, relying on the HTML model for content consumption: download-and-play, or progressive download and rendering. However, the streaming of SMIL or SVG content is not specified, making these models inappropriate for fast, dynamic and interactive and interoperable content. A strong requirement for the new standard for Rich Media content for mobiles is to leverage the adoption of SVG by providing means to extend its consumption scenarios to streaming and broadcast.

Creating Rich Media content services for the mobile is not only about composition coding. An important aspect in the success of a mobile service is the reactivity and fluidity of the user experience. Such characteristic is achieved through efficient delivery mechanisms. However, efficiency is difficult to achieve when distributing Rich Media content made of individual audio, video, and image content. Separate delivery of all these media streams to a mobile

incurs a high latency unless an efficient aggregation mechanism is used: high-latency networks attach a specific penalty to multi-media content when consumed in download-and-play mode, because the waiting time of the content is the sum of the waiting time of each media requested separately. Instead, getting all streams in one single package would reduce waiting time to one single request delay.

The requirements for such aggregation mechanisms are simplicity of implementation and efficiency. A natural candidate for this task is the ISO Base Media File Format [04] because it is very popular and already adopted by the mobile industry. However, this file format was designed for storage of large amount of media data, easy editing or streaming operations. It is not efficient for the storage of small amounts of timed media data. A simple to implement yet efficient aggregation format for mobile is needed to complement the ISO Base File Format.

2.3. Key Features that Make a Difference

The LASeR standard has been designed to satisfy the end-users expectation listed above. Four key features can be highlighted that makes a real difference with existing technology:

1. Graphic Animations, Audio, Video and Text are packaged and streamed altogether. Contrary to existing technologies on mobile that are mostly aggregation of various components, not necessarily well integrated together (e.g. XHTML + SMIL + SVG + CSS + Ecmascript + …), LASeR grounds its design from what made the success of Macromedia Flash on the Web: a single, well defined and deterministic component that integrates all the media. This integration ensures both the richness and quality of the end-user experience.
2. Full screen and interactivity with all streams. With the use of vector graphic technology, content can easily be made to fit the screen size. This feature enables to provide an optimal content display although the screen resolution is highly varying. In addition, virtually all pixels can be used as elements of the user interface. This allows the design of rich and user-friendly interfaces, similar to what people used to have when interacting with their devices.
3. Real-time content delivery. LASeR has been designed for an efficient delivery over constrained networks. More specifically, Rich Media LASeR content can be delivered into packaged pieces, allowing display as soon one piece is received (as opposed to a download an play mechanism). This concept of “streaming”, already into place for audio and video data, has been generalized to scene description and Rich Media. As such, services can be designed such has there is always some information of interest on the screen.
4. Last but not least, LASeR has been designed to deliver Rich Media Service starting from 10 Kb/s. The key technology used here is vector graphics compression and dynamic updates of the scene. This feature enable to drastically limit the waiting time of the end-users as opposed to a standard Web-like approach where the complete page is re-sent even though only small changes had been made. Needed for low bitrate networks such as GPRS, this functionality is also useful on higher bit rate network where Rich Media services can be sent at low rate, therefore preserving bandwidth to improve audio and video quality.

The rest of this paper focuses on the LASeR standard and is organized as follows: Section 3 describes the intended use for this standard by giving example use cases; Section 4 details the technical merits of MPEG-4 Part 20 and its relationship with other parts of MPEG-4. Finally, Section 5 draws the conclusion of this paper and analyses the possible evolution of the standard.

3. Use cases for MPEG-4 Part 20

This section describes three use cases which demonstrate the benefit of MPEG-4 Part 20 and which exercise the key features described in 2.3. These use case are the following: Rich Media Portal; Interactive Mobile TV and Interactive Screen Saver.

3.1. Rich Media Portal

This application (illustrated in Figure 1-a) demonstrates how a LASeR engine can enhance an existing WAP or XHTML service with Rich Media in a very similar way Flash enhances Web sites. The user first accesses the WAP portal. A hyperlink gives him access to the Rich Media part of the site. He then accesses to a complete, deterministic and consistent Rich Media experience. Navigating the site is simpler with an intuitive interface with pop-up menu, making the size of the screen feel "bigger". In the example, rich text is demonstrated with quality, small-sized but still readable Arabic fonts. At any time, a hyperlink can bring him back to the usual portal.

3.2. Interactive Mobile TV

Interactive Mobile TV (illustrated in Figure 1-b) is the aggregation of multiple different Rich Media use cases spanning from interactive mosaic, electronic program guide, voting to personalized newscast. All of these require the ability to provide a deterministic rendering and behavior of rich-media content including audio-visual content, text, graphics, images, along with streamed TV and/or radio channels, all together in the end-user interface. Fluid navigation through content in a single application or service must be provided, as well as local or remote synchronized interaction for voting and personalization (e.g. related menu or sub-menu, advertising and content in function of the end-user profile or service subscription).

(a) (b) (c)

Figure 1 - LASeR-Based Rich Media Services

3.3. Interactive Screen Saver

This application (illustrated in Figure 1-c) is one instance of a larger class of applications that get content updates in background (e.g. fix/mobile convergent services). The screen saver is the variant that has mostly static data, i.e. text, graphics and images, arranged with transitions similar to a generalized slide show, with an element of randomness in the presentation of successive elements. The server adds new elements to or removes expired ones from the current application as stored on the device. Although very attractive, this application is to be designed with care to avoid a too heavy usage of the device power resource.

4. Technical aspects of MPEG-4 Part 20

Following the requirements expressed in Section 2.2, MPEG decided to create Part 20 of MPEG-4. This new standard is divided into two specifications: LASeR and SAF. This section describes the technical aspects of these specifications in section 4.1 and 4.2 as well as their relationships with the other parts of MPEG-4 in Section 4.3.

4.1. LASeR

The LASeR standard specifies the coded representation of multimedia presentations for rich-media services. In the LASeR specification, a multimedia presentation is a collection of a scene description and media (zero, one or more). A media is an individual audiovisual content of the following type: image (still picture), video (moving pictures), audio and by extension, font data. A scene description is composed of text, graphics, animation, interactivity and spatial and temporal layout.

A LASeR scene description specifies four aspects of a presentation:

- how the scene elements (media or graphics) are organized spatially, e.g. the spatial layout of the visual elements;
- how the scene elements (media or graphics) are organized temporally, i.e. if and how they are synchronized, when they start or end;
- how to interact with the elements in the scene (media or graphics), e.g. when a user clicks on an image;
- and if the scene is changing, how these changes happen.

A LASeR scene description may change by means of animations. The different states of the scene during the whole animation may be deterministic (i.e. known when the animation starts) or not, e.g. when a server sends modifications to the scene. The sequence of a scene description and its timed modifications is called a LASeR stream. This notion of LASeR AU is the key to streaming LASeR content while guaranteeing tight synchronization between the scene and the media assets composing the rich media presentation.

The specification defines a LASeR engine as the viewer for LASeR presentations. Such an engine has rich media composition capabilities on top of those of a classic multimedia player with audio, video, images and text capabilities. These composition capabilities are, as a result of the technology selection process, based on SVG Tiny 1.1 [06]. The composition capabilities rely on the usage of an SVG scene tree. They are enhanced with key features for mobile services, such as a binary encoding, dynamic updates, state-of-the-art font representation and stable features of the up-coming SVG Tiny 1.2 specification [09], including audio and video support. These features are presented in Figure 2 and explained in the next sections.

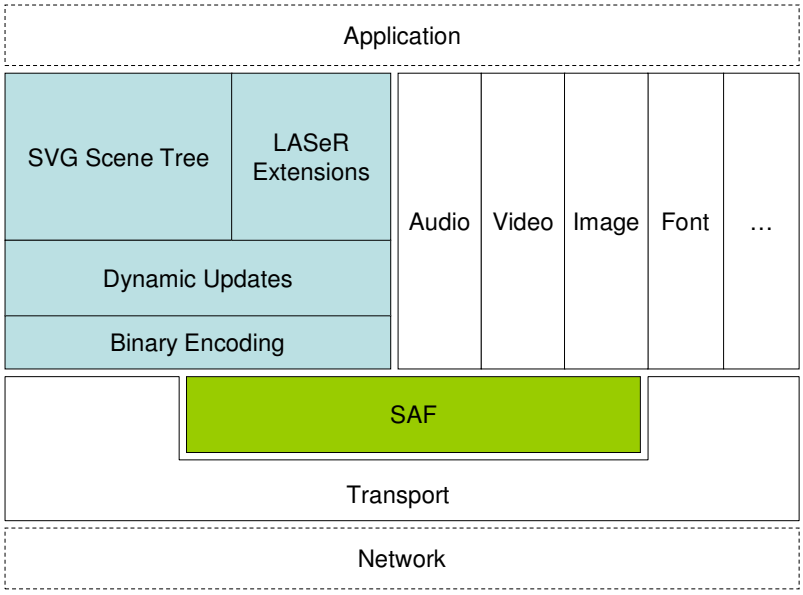

Figure 2 – Architecture of a LASeR Engine

4.1.1. SVG Scene Tree

LASeR uses an SVG scene tree at its core. It imports composition primitives from the different W3C specifications (all of SVG Tiny 1.1 [08], some of SVG 1.1 [06] and SMIL 2 [10]) and uses the SVG rendering model to present the scene tree. Among all its composition primitives, LASeR specifies hyperlinking capabilities, audio and video media embedding, vector graphics representations, animation and interactivity features.

4.1.2. LASeR Scene Tree Extensions

After selecting SVG as core technology for LASeR, MPEG has identified a few areas where extensions are needed to allow the development of efficient services with LASeR:

- SVG Tiny does not have clipping, so MPEG added simple axis-aligned rectangular clipping: this easy-to-implement feature is a must to be able to create such common user interface widgets as ticker tapes and simple transitions.
- SVG lacks, for video and images, a restricted, non-resampling rotation and a full screen mode, which MPEG also added.
- SMIL and SVG only allow the signaling of one single synchronization master, whereas MPEG allows the specification of multiple synchronization references.
- MPEG defines new events: longkeypress, pause and resume, the last two used to pause and resume video, audio and other timed elements.
- Finally, SVG Tiny does not have simple underlining of text, which MPEG allows by adding a set of additional font-styles imported from digital television captions.

4.1.3. Dynamic Updates

SVG currently supports the following content consumption use cases:

- The first option is the classical “download and play” mode. The user waits until the end of the download to start viewing the content.

- The second option is the progressive rendering mode. This mode is an improved version of the previous one enabling visualization while downloading the content. But the downloaded content only adds new content to the existing one making it difficult to manage long-running documents.
- The third option is based on the use of scripting and DOM Network software interface and an ad-hoc protocol to communicate scene modification from the server to the client.

However, the following use cases, currently permitted by Flash or MPEG-4 BIFS, cannot be satisfied in an efficient and interoperable way in SVG:

- The efficient representation of streamable cartoons,
- The partitioning of scenes into small packets that fit in size-limited delivery mechanisms (such as cell broadcast),
- The dynamic creation of answers to a user request, and their integration in the current scene,
- Or the dynamic push of content into an existing scene.

In order to enable these use cases, LASeR complements SVG by a so-called dynamic updating mechanism which uses LASeR commands. Using these commands, a server can for instance insert or delete graphical elements or modify the visual properties of an object. These commands can also be used to enable a web cookie-like mechanism.

4.1.4. Binary Encoding

As specified by the W3C, SVG content is created, stored and transmitted in XML form. Although XML is well suited for the Web as browsed on powerful PCs with high-bandwidth Internet connections, XML incurs a severe penalty in performance, code size and memory requirements for small, pre-determined vocabularies. While debate rages in many circles about the generality of the above statement, this is the choice made by MPEG for LASeR: decoding a LASeR binary stream is much more efficient, as much of the complexity of the XML parsing is removed.

The binary format specified in LASeR allows the encoding of SVG content. It uses a compact representation for the structure of the SVG elements and uses specific coding algorithms to encode the attribute values of the SVG elements. Because the mobile platforms usually lack hardware float processing, the compression of these attribute values has to be simpler than on PC. Complex computations that would improve the compression ratio by a small amount at the cost of doubling the decoding time have been rejected during the standardization process. Thus, the binary encoding of LASeR is straightforward, and its quality resides in the complexity/efficiency balance. Special care was taken for the encoding of values for some attribute types, like list of float coordinates, vector graphics paths or transformation matrices

The LASeR binary syntax is extensible, so that private extensions can be mixed among normal LASeR elements and attributes, to be ignored by decoders that do not know how to process them.

The LASeR binary syntax can be decoded by a generic BiM decoder, since it is compatible with a predefined configuration of BiM. One of the purposes of the BiM compatibility is to be able to encode other XML syntaxes together with LASeR. One interesting example of this would be XHTML. However, the implementation of the LASeR standard does not mandate the use of BiM: The LASeR binary syntax is specified in such a way that a specific BiM-agnostic decoder can process it.

4.1.5. Audiovisual support

One of the key features of Rich Media is the support of audio and visual infomation. SVG 1.1 does not feature audio or video support, while SMIL2 does. It is foreseen that such support will eventually be included into SVG Tiny 1.2. The LASeR specification includes SMIL2 audio and video elements as well, including the additional MediaClipping module for VCR-like media control. Audio and video streams are carried beside the LASeR stream and referred to by binary identifiers.

4.1.6. Usage of Font Information

Both SVG and LASeR allow content creators to embed font information within the content to insure that the content will be presented as it was designed and that no font substitution will happen at viewing run time. However, the SVG Fonts solution was deemed too limited by MPEG for the following reasons:

- It does not leverage the support of state-of-the-art font rendering engines by the devices,
- It does not support OpenType fonts with hinting information, that are more readable at small screen sizes and better equipped for internationalization than SVG fonts,
- It does not allow sharing of fonts with other applications present on the device, such as an XHTML browser.

MPEG also chose to carry the font information beside the scene information as a media stream. The exact format is optional. One option is the usage of MPEG-4 Part 18 [05], which provides the definition of a font data stream, able to carry OpenType fonts, possibly compressed.

SVG Fonts being a functional subset of OpenType fonts, it is possible to transcode without loss SVG Font data into OpenType font data for carriage with LASeR and possible reconstruction of SVG Font data for playback in an SVG player. This is shown in Figure 3.

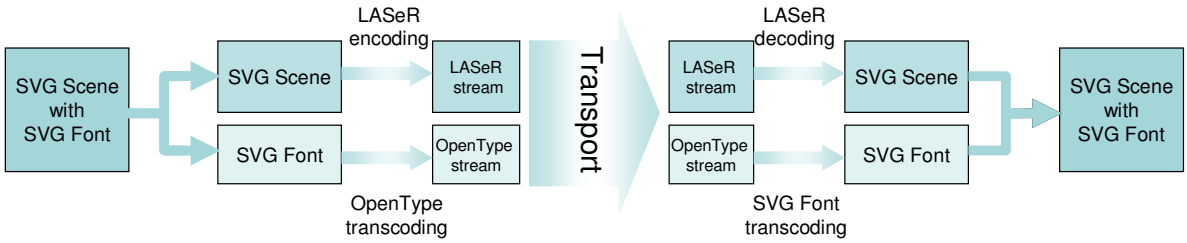

Figure 3 - Transmission of Font beside LASeR Content

4.1.7. Services as Incremental Scenes

Many Rich Media services rely on a key feature of LASeR: incremental scenes, made possible by the LASeR append mode. The append mode is the possibility to create a LASeR stream containing not an independent scene, but an addition to another existing scene.

There are two typical use cases of incremental scenes:

- Streaming style: the scene is designed as a sequence of frames, and there is a continuous stream of updates to change the current frame into the next frame. Bandwidth usage is varying but never drops to 0. The incremental scenes of this kind are usually best transported over streaming protocols like RTP. A typical use case is a cartoon-like animation.

- Interactive style: the scene is interactive and user requests are processed by the server. The response to user request is a change to the existing scene, not a new scene. Such scenario also requires continuous updates to the scene, but the statistics of the transmission are totally different from the previous style: bandwidth is heavily used for a short time after a user request, and then drops to 0 until the next user request. Given the variety of usages of mobiles, the next user request could come a few seconds or a few hours later.

From a server-side point of view, the interactive transmissions can be considered as a series of separate connections, as opposed to the continuous connection of the streaming style. It is typically implemented using separate HTTP connections, since each data burst results from a user request. However, from a LASeR viewer point of view, it is the same scene/service that is modified. Hence the requirement for the server to be capable of signaling an append mode: "this stream does not contain a totally new scene, but an improvement to the scene the viewer is currently processing".

The append mode also allows the creation in advance of multiple responses to possible user requests. If the service is modeled as a state machine, each transition of the state machine represents a change to the current scene and may be implemented as an append component. Careful authoring and scope management is required, in particular to avoid clashes of id between elements added by different append components. Still, this functionality opens the way to servers caching most of the responses to users, therefore dramatically improving the service's performance.

4.2. SAF

4.2.1. Overview

The SAF specification defines tools to fulfill all the requirements of rich-media service design at the interface between scene representation and transport mechanisms. SAF features the following functionality:

- simple aggregation of any type of media streams (MPEG or non-MPEG streams), resulting in a SAF stream with a low overhead multiplexing schema for low bandwidth networks,
- and possibility to cache SAF streams, as described in 4.2.3.

The result of the multiplexing of media streams is a SAF stream which can be delivered any delivery mechanism: download-and-play, progressive download, streaming or broadcasting,

4.2.2. Aggregation Mechanism

The SAF specification defines the binary representation of a compound data stream composed of different data elementary streams such as LASeR scenes, video, audio, image, font, metadata streams. Data from these various elementary streams results in one SAF stream by multiplexing them for simple, efficient and synchronous delivery. A SAF stream is made of SAF Access Units (AU) of the following classes:

- AUs carrying configuration information for the media or LASeR decoder to be initialized.
- AUs carrying configuration information for elementary streams not carried inside this SAF stream. Streams that need to be carried separately include streams which are started interactively, or are delivered through another protocol.

- AUs carrying media or LASeR AU.
- AUs carrying an end of stream signal, indicating that no more data will be received in an elementary stream.
- AUs carrying an indication that no more data will be received in this SAF session.
- AUs carrying cache units as explained in section 4.2.3.

LASeR and SAF can be used independently, but MPEG currently only specifies the usage of SAF stream carrying a LASeR stream. In this case, the first SAF AU must carry the configuration information for the LASeR engine.

4.2.3. Caching capabilities

An approach to reduce server response time in high-latency networks is to anticipate what the user is going to need next and send it together with the previous request. A mechanism for doing this is provided in SAF: the CacheUnit. It is a package of information attached to a URL. Upon receiving this package, the LASeR viewer stores it in its cache, associated with the provided URL. If the user requests that URL, the viewer does not need to send that request to the server, it just uses the cached version. Cache Units have an expiration date. Together with the LASeR append mode, this feature is key to a significant improvement of Rich Media service fluidity.

4.3. Relationship with Relevant Parts of MPEG-4

4.3.1. Overview

The LASeR standard relates mainly to two important existing specifications of MPEG-4: Part 1 – Systems, including the Synchronization Layer (SL) and the Object Descriptor Framework (ODF); and Part 11 – BIFS and MPEG-J.

There is currently no specification to define the usage of LASeR in a complete MPEG-4 Systems environment. But, LASeR can be seen as a replacement of BIFS for applications on mobiles or for applications not requiring integration of 2D and 3D content, nor MPEG-J. Theoretically, it is possible to build an MPEG-4 Systems application using a LASeR stream instead of a BIFS, with one major difference: a LASeR stream does not make use of the ODF.

SAF has been designed to maintain compatibility with the Synchronization Layer although not mandating its use. The syntax of SAF packets follows the syntax of SL packets, with a particular configuration. The SAF specification can be seen as a multiplexing scheme at the SL level.

SAF and LASeR have been designed to be independent, but aware, of the rest of the MPEG-4 specification. Therefore, it is possible to carry and use MPEG-4 video or audio streams as defined in MPEG-4 Part 2, 3 or 10.

4.3.2. Storage in ISO Base Media File Format

LASeR content can be stored within files compatible with the ISO Base Media file format [04]. As a LASeR stream is a timed stream, made of AUs, the storage of LASeR streams in ISO files is straightforward and similar to the storage of audio or video streams. Each LASeR AU is stored as a sample. All these samples form a LASeR track identified by a four character code. The configuration for the LASeR decoder is stored as an entry the sample description box. In case of a LASeR stream comprising only one AU, it is also possible to store this AU,

as it is done in the 3GPP specification for SMIL presentation, i.e. as a primary item of the file, using the Metadata box structure.

A SAF stream is a timed stream but it is an aggregation of several elementary streams like LASeR, audio, video. It would therefore not be appropriate to store it as is in an ISO file. However, SAF could be considered as a delivery protocol and as such, hint tracks could be defined to generate one SAF stream per ISO file, aggregating the LASeR, audio, video tracks of that file.

4.3.3. Streaming

MPEG and IETF have jointly specified, in MPEG-4 Part 8 [02] and in RFC 3640 [14], the transport over IP of any kind of MPEG-4 elementary stream, as long as it is packaged using the Synchronisation Layer syntax. Hence, MPEG-4 Part 20 does not specify any new elements in this domain. LASeR streams, being elementary streams, can be wrapped into Synchronization Layer Packets and transported into RTP packets as specified in RFC 3640. There is currently no specification for the carriage of SAF packets over RTP.

5. Conclusion

In this paper we presented LASeR, the Rich Media standard dedicated to the mobile, embedded and consumer electronics industries, and SAF, its companion aggregation format within MPEG-4 Part 20.

We have described the LASeR and SAF technical details, explained how these standards interact with the rest of MPEG-4 and finally shown how LASeR will impact usages, particularly on mobile devices.

It is expected that the LASeR specification will be extended: support for new features, available on mobile devices and extension to support SVGT1.2 features. One other direction for future work relates to the link between the LASeR specification and the work done in the W3C CDF (Compound Document Format) group. This group is working on the interactions among the W3C standard like SVG, SMIL, XHTML. It is therefore a natural extension for LASeR to study how this work will affect LASeR.

Finally, the development of the LASeR standard may impact other MPEG activities such as M3W (MPEG Multimedia Middleware) activity. M3W aims to improve the portability of applications and services by defining a series of APIs than can evolve as middleware technology itself evolves. Dedicated APIs to access a LASeR engine could be an interesting extension of M3W.

6. Acknowledgments

The work done by one of the authors of this paper has been partially funded by the European Commission in the course of the IST Project DANAE (http://danae.rd.francetelecom.com).

7. References

[01] ISO/IEC 14496-1:2004 - "Information technology - Coding of audio-visual objects - Part 1: Systems".

[02] ISO/IEC 14496-8:2003 - "Information technology – Coding of audio-visual objects – Part 8:Carriage of ISO/IEC 14496 contents over IP networks".

[03] ISO/IEC 14496-11:2005 - "Information technology – Coding of audio-visual objects – Part 11: Scene description and application engine".

[04] ISO/IEC 14496-12:2004 - "Information technology - Coding of audio-visual objects - Part 12: ISO Base Media File Format".
[05] ISO/IEC 14496-18:2004 - "Information technology - Coding of audio-visual objects - Part 18: Font Compression and Streaming".
[06] ISO/IEC 15938-1:2002 – "Information technology -- Multimedia content description interface -- Part 1: Systems"
[07] W3C, Scalable Vector Graphics (SVG) 1.1 Specification [Recommendation], http://www.w3.org/TR/2003/REC-SVG11-20030114/.
[08] W3C, Mobile SVG Profiles: SVG Tiny and SVG Basic, http://www.w3.org/TR/2003/REC-SVGMobile-20030114/.
[09] W3C, Scalable Vector Graphics (SVG) Tiny 1.2 Specification [Last Call], http://www.w3.org/TR/2005/WD-SVGMobile12-20050413/.
[10] W3C, Synchronized Multimedia Integration Language (SMIL 2.0) - [Second Edition], http://www.w3.org/TR/2005/REC-SMIL2-20050107/.
[11] W3C, SMIL Animation, http://www.w3.org/TR/2001/REC-smil-animation-20010904/.
[12] W3C, Cascading Style Sheets, level 2 [Recommendation], http://www.w3.org/TR/1998/REC-CSS2-19980512/.
[13] Real Time Streaming Protocol, RFC 2326, http://www.ietf.org/rfc/rfc2326.txt.
[14] RTP Payload Format for Transport of MPEG-4 Elementary Streams, RFC 3640, http://www.ietf.org/rfc/rfc3640.txt.
[15] LASeR web site, http://www.mpeg-laser.org.
[16] Macromedia Flash, http://www.macromedia.com/flash