MORE: A MOBILE OPEN RICH MEDIA ENVIRONMENT

Vidya Setlur, Tolga Capin, Suresh Chitturi, Ramakrishna Vedantham, Michael Ingrassia

Nokia Research Center, Dallas, TX, USA

ABSTRACT

'Rich media' is a term that implies the integration of all of the advances we have made in the mobile space delivering music, speech, text, graphics and video. This is true, but it is more than the sum of its parts. Rich media is the ability to deliver these modalities, to interact with these modalities, and to do it in a way that allows for the construction, delivery and use of compelling mobile services in an effective and economic manner. In this paper, we introduce a system called Mobile Open Rich-media Environment ('MORE') that helps realize such mobile rich media services, combining various technologies of W3C, OMA, 3GPP and IETF standards. The different components of the system include formatting, packaging, transporting, rendering and interacting with rich media files and streams.

1. INTRODUCTION

Until recently, applications for mobile devices were text based with limited interactivity. However, with more wireless devices equipped with color displays and advanced graphics rendering platforms, consumers demand a real-time rich media experience from new services and applications. With rich media, it becomes possible to define a common layout within which other media can be embedded and played. The Third Generation Partnership Project (3GPP) and the Open Mobile Alliance (OMA) have begun work on the streaming of mobile rich media content over Packet-Switched Streaming (PSS) and Multimedia Broadcast/Multicast Service (MBMS) [1, 2]. Example applications include real-time traffic and map information, interactive mobile TV and entertainment (Figure 1).

In this paper, we present 'MORE', a system that leverages the respective components of existing W3C, 3GPP, MPEG, OMA solutions such as the SVG Mobile 1.2 Profile [3], MBMS, ISO Media File Format [4] and browsing enablers. Our solution is also compatible with the Mobile Java Environment via the shared DOM definition found in JSR-226 [5]. The MORE solution includes:

- the use of SVG as a presentation format for the support of scenes and dynamic scene updates.
- a solution to embed vector graphics content such as SVG and metadata into the existing ISO Base Media File Format for progressive download and streaming of live rich media content over MMS/PSS/MBMS services [4].
- transport mechanisms for broadcast download and streaming of SVG content and its associated media via FLUTE/ALC and RTP.
- mechanisms for local (client side) and remote interaction (server-client), as well as for real time and non-real time feedback over various broadcast and unicast transport protocols.

2. RELATED WORK

There are a number of solutions proposed to provide rich media services on mobile devices. Macromedia Flash [6] provides a proprietary solution for rich media distribution, primarily over HTTP.
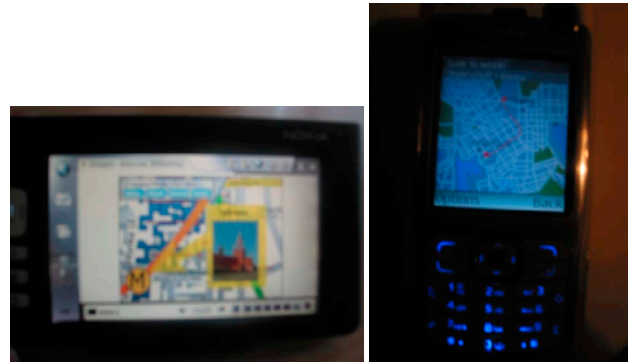

Fig. 1. *Examples of rich media applications running on Nokia 770 and Nokia N70 respectively.*

Flash also includes a profile, called Flash Lite, targeted for resource constrained devices. However, primary differences between Flash and MORE exist because of the underlying animation model. Each Flash SWF presentation is defined on a frame-by-frame basis. On the other hand, our SVG and SMIL based solution uses an interval-based presentation, which is specified using a clock value. Thus, our solution can scale to different devices with varying CPU speed and target display rate. Also, Flash does not address the streaming of graphics content over broadcast and packet-switched networks.

MPEG-4 LASeR [7] is a recent standard targeted for rich media services on mobile devices. As such, there are similarities between our solution and LASeR, given that SVG is used as the main presentation format. However, our solution differs from LASeR in that its architecture is largely motivated by the need for a strong separation of its components and their interfaces. By enforcing such a strong separation, the system is extensible, allowing to incorporate the best solutions within the architecture, such as new compression techniques, new media types in the container format and encoding.

3. OVERVIEW

The MORE system can be perceived as a client-server architecture, comprising 3 main components: The rich media server, transport mechanisms and the rich media client. Figure 2 illustrates the architecture. The server takes as input, rich media content containing SVG, discrete (e.g images) and continuous (e.g audio, video) media. SVG content is represented as scenes and can be dynamically updated by scene updates (Section 4). This raw rich media content can be encapsulated into a container format, with additional information such as media synchronization, metadata, and hint tracks for packetization (Section 5). The system then utilizes various transport mechanisms for 1-to-1 and 1-to-many protocols for download, progressive download and streaming scenarios as described in Section 6. The content is played on the client, allowing for local and remote interactivity of feedback and data requests (Section 7).
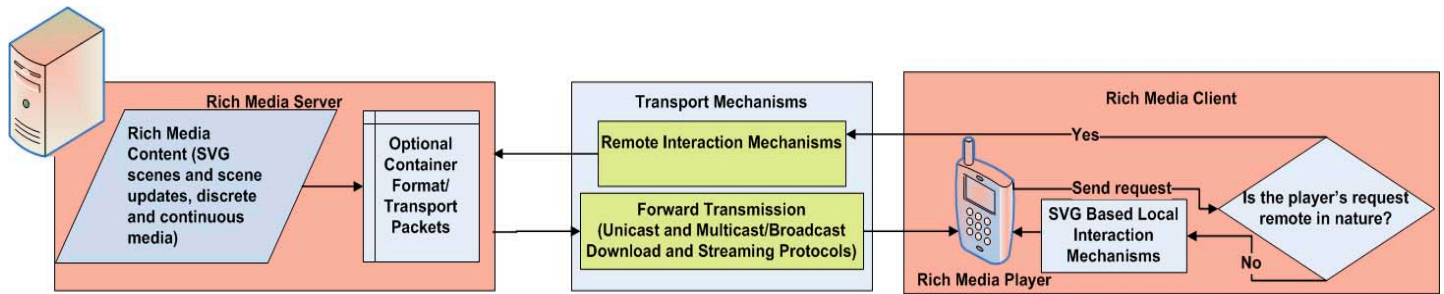
Fig. 2. *Client-Server Architecture of the MORE system.*

4. SCENE AND SCENE UPDATES

One of the motivations for rich media services is the ability to receive rich media content with minimal latency. In order to do so, the content or 'scene' on the client is dynamically updated with small changes rather than a completely new document being re-sent every time.

The scene, based on the SVG format, describes the spatial and temporal organizations of scene elements, synchronization information, and interaction among the elements. The scene is either a complete SVG document or content enclosed within group ('g') elements, first sent to initialize the client's presentation layout. Scene updates are DOM-compliant incremental updates to the scene. These updates include one or more element addition, element deletion, element replacement and element attribute update operations. The client then updates the SVG document without destroying and recreating the SVG uDOM for every streamed packet of information.

5. CONTAINER FORMAT

SVG supports media elements similar to that of Synchronized Multimedia Integration Language (SMIL) [8]. The continuous media elements in particular, contain their own pre-defined frame-based timing. The server is responsible for generating and transmitting packets containing rich media data to the clients in a temporally compliant manner with low delay request. For this, we utilize a container format to efficiently package the different media, providing timing synchronization, and enabling clients to realize, play, or render rich media content.

The ISO Base Media File Format, defined by 3GPP is a standard for the creation, delivery and playback of multimedia over 3rd generation, high-speed wireless networks [4]. In addition, the file format already contains media tracks and associated information for audio and video. MORE extends the file format's box hierarchy to incorporate SVG as a new media track. By adding an additional media track, leveraging the use of time synchronization along with existing audio and video track information, the solution is relatively simple and is extensible to other media formats if needed.

SVG media data, being an XML based language, can be stored in individual files, different media data boxes ('mdat') within the same file, or in the XML boxes ('xml' and 'bxml'), or a combination of them. We introduce a new SVG handler with handler type 'svxm' and a name 'image/svg+xml'. Under the Sample Description Box (stsd) an SVGSampleEntry is defined as follows:

```
class SVGSampleEntry() extends SampleEntry {
    unsigned int(16) pre_defined = 0;
    const unsigned int(16) reserved = 0;
    unsigned int(8) type;
```

```
    string content_encoding;
    string text_encoding;
    string content_script_type;
    unsigned int(16) format_list[];
}
```

where, *type* specifies the type of payload format; *content_encoding* has possible values of 'none', 'gzip', 'bin_xml', 'compress', 'deflate'; *text_encoding* has possible values taken from the 'name' or 'alias' field in the IANA specification [9]; *content_script_type* identifies the default scripting language for the given sample (E.g. EcmaScript); *format_list* indicates the format numbers of the internally linked embedded media within the corresponding SVG sample. We also extend the functionality of random access to SVG stream by allowing one or more SVG scenes to serve as random access points, with the Sync Sample Box providing a compact marking of the random access points within the stream.

6. TRANSPORT MECHANISMS

The transport mechanisms support rich media delivery in the following modes: unicast download (HTTP/TCP or MMS protocol), broadcast/multicast download (FLUTE/UDP or ALC/UDP), unicast streaming and broadcast/multicast streaming (RTP/UDP). SVG is traditionally considered to be a discrete media and hence no RTP payload format has been defined. It has been transported only in download and progressive download mode. With increasing richness and dynamism in the SVG presentations, it can now be considered as a continuous media. Consequently, we define an RTP payload format for SVG. Rich media is a combination of continuous media and discrete media, so rich media streaming should uses relevant transport mechanisms for these two media types. Rich media streaming is thus naturally realized by (a) streaming continuous media like SVG, video and audio (b) downloading the discrete media like images. As SVG is often the driving engine for the rich media presentations, the RTP payload format for SVG enables synchronous playback of SVG with other media, tools for packet loss detection and graceful recovery from packet losses.

Based on Figure 3 of the payload format, the fields are as follows: The marker bit 'M' is set for the last unit in a group of scenes and scene updates, where this last unit corresponds to the scene update immediately preceding a new scene. This is similar to the use of the marker bit in video coding and enables efficient buffering. The 'timestamp' indicates the sampling instant of the SVG sample. The Type field (3 bits) specifies which specific header fields follow. A TYPE1 packet contains one or more SVG sample descriptions. A TYPE2 packet contains a complete SVG scene sample or one of its fragments. A TYPE3 packet contains a complete SVG scene update sample or one of its fragments. A TYPE4 packet contains a
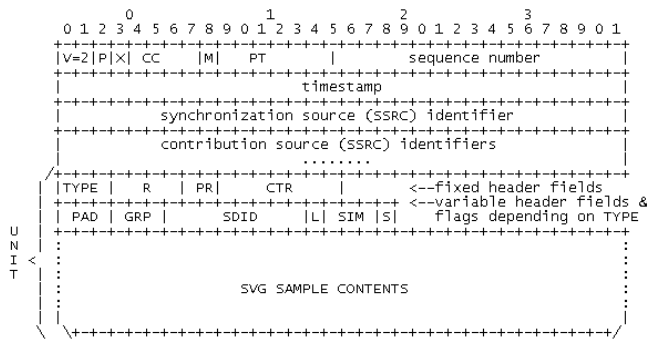
```
           0                   1                   2                   3
           0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
           +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
           |V=2|P|X| CC    |M|   PT      |        sequence number         |
           +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
           |                           timestamp                         |
           +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
           |           synchronization source (SSRC) identifier          |
           +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
           |           contribution source (SSRC) identifiers            |
           |                           ........                          |
          /+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
        | |TYPE |  R    | PR|    CTR      |      <--fixed header fields
        | +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  <--variable header fields &
        | | PAD | GRP   |     SDID      |L| SIM |S|     flags depending on TYPE
    U   | +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    N   | :                                                             :
    I  <  :                                                             :
    T   | :                                                             :
        | :               SVG SAMPLE CONTENTS                           :
        | :                                                             :
        | :                                                             :
         \ \+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+/
```

Fig. 3. *RTP payload format for SVG.*

list of SVG elements that are currently active to enable SVG DOM repair in the case of packet loss. A TYPE5 packet contains Sample Dissimilarity Information (SDI). SDI provides information as to how a sample is different from the SVG scene graph currently on the client. This information helps in error recovery and concealment if required.

The Reserved Bits Field R (4 bits) is used for future extensions. The Priority Field PR (2 bits) indicates the level of importance for a particular sample. A higher value indicates a higher priority. The Counters Field CTR (7 bits) is tied with the PR flag. Each of the counters is incremented by one for each new packet of the corresponding priority. While a discontinuation in the sequence number indicates a lost packet, a discontinuation in the counter value (of a certain priority) indicates the priority of a lost packet. The Number of Padding Bits Field PAD (3 bits) indicates how many padding bits are in the final octet of the real payload. The GRP field (4 bits) indicates to which group the scene belongs. Generally, one scene followed by one or more scene updates constitutes a group. The Sample Description Index Field SDID contains a reference to the sample description that must be used to decode the scene description contained in the current packet. The Link flag L (1 bit) indicates whether the current SVG sample references external audio, video, SVG or image content. The SIM field (3 bits) is used to denote if Sample Dissimilarity Information (SDI) is present or not. The S flag (1 bit) indicates whether the current packet contains the starting point of the current sample. Multiple fragments belonging to the same sample are identified by both the timestamp and TYPE.

During a rich media application, a client can often report the quality of transmission and presentation to the server. Such information is quite useful for the server to make optimal decisions about adjusting the transport scheme and synchronization mechanisms. Currently, higher-layer frameworks such as PSS, MBMS are widely used in multimedia applications to address this issue and several such QoS parameters are defined. However, these solutions mainly concern continuous media and do not cater specifically to rich media applications. We therefore define some extensions to the present framework, allowing clients to send quality-related feedback concerning reception of rich media content. These newly defined QoS metrics are: The number of RTP packets lost for each packet priority during a specific period; List of elements in the most recent list of active SVG elements that have not been correctly received and decoded; SVG elements correctly received, decoded and active in current group; Corruption duration; Corrupted groups.

7. INTERACTION

During a rich media presentation, a user can often interact with the client to request more information, update the content, or even send some information back to the server. Feedback mechanisms associated with audio, video media typically comprise a report of packet loss, quality measures, or controlling information (e.g. play, pause, record, etc.). SVG provides local interaction through declarative animation and scripting. Functionality for remote interaction is fairly limited with the use of hyperlinks for invoking HTTP GET/POST commands.

We represent the SVG based feedback information in the form of a generic text payload to serve as a back channel over other protocols as well (E.g. MMS, RTSP). This payload has two parts. The first part contains the MSG_ID, ELEMENT_ID and the EVENT, where the MSG_ID is a unique identifier to identify the feedback message from the client, ELEMENT_ID is the ID of the source element in the SVG DOM that triggers the event, and EVENT is an SVG event or a user defined event. The actual feedback data is stored after the first part as a series of octets. This data may contain attributes of the SVG event itself An example of such a payload is as follows:

```
<metadata>; MSG_ID=1;ELEMENT_ID="my-button1";
EVENT="click";[OCTET1.OCTET2. ... OCTETN];
```

The above example pertains to a SVG scene with a set of buttons to select a movie. On clicking one of the buttons, the client stores the X and Y positions where the button was clicked. This information is formulated into a remotely processed feedback message to the server. Octets may store information such as clickX and clickY in this example. However, the actual feedback data can also contain information like the user selected movie. In this case, the octets contain information such as "movieSelected=Lord of the Rings." Therefore, on clicking one of the buttons in the scene, a script basically stores this value in a field called 'movieSelected.' There is no particular restriction on the values of the octets in the feedback, but should follow a convention known to the service, i.e. the server and the clients.
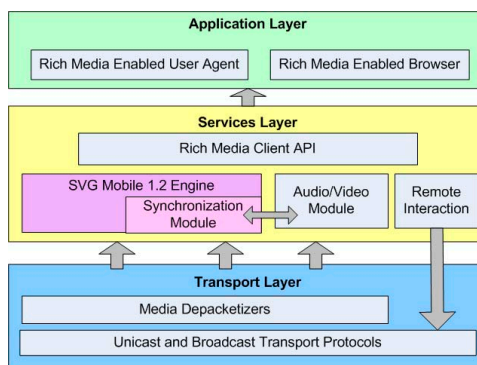
8. MORE CLIENT

Fig. 4. *MORE client architecture.*

The MORE client is a lightweight entity present on the mobile terminal (Figure 4). This is substantiated due to the fact that it builds on top of existing application enablers such as SVG Mobile 1.2, ESMP and XHTML-basic, thereby reusing their associated underlying components such as the XML parser, rendering libraries and

media decoders. The client uses media packet depacketizers to obtain the different media that constitute the scene and scene updates. The synchronization module helps synchronize the frame rate and timing of continuous media with that of the non-frame based SVG content. The SVG engine in turn takes the different media and timing information as input to compose the dynamically rich multimedia presentation. The client is also responsible for transmitting any feedback occurring during interaction.

The MORE system has been used to realize several specific use cases of rich media services. These applications include a karaoke service to allow the display of animated SVG characters along with a video clip; live chat allowing end-users to dynamically exchange rich media content; interactive mobile TV service to provide access to TV content and to additional services along with this TV program, such as voting in a live show. Figure 5 shows one such interactive rich media map application.
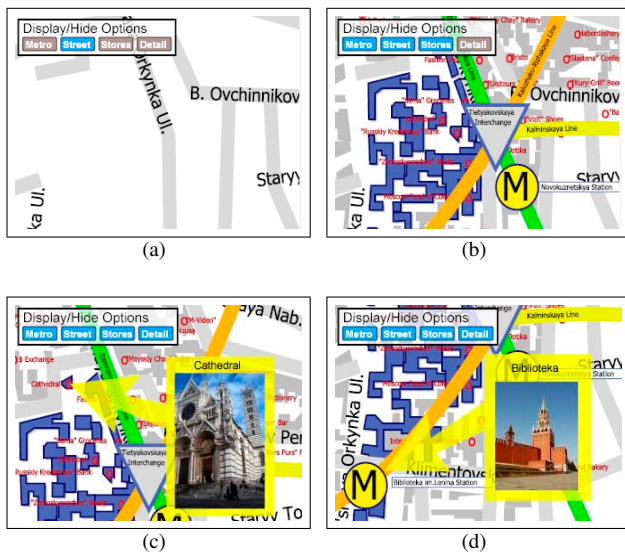

(a) (b)

(c) (d)

Fig. 5. *Representation of content update and user interaction with a rich media application. (a) Initial rich media scene received by the MORE client. (b) User requesting for more topographical information via a button menu. The content is updated after the client transmits this user request and subsequently receives the scene update. (c) and (d) Clicking on different regions of the map, a corresponding information video is streamed to the client and played.*

9. REFERENCE MODEL ANALYSIS

Rogge et al. have recently introduced a reference model based on several heuristics for a systematic comparison of different existing multimedia formats [10]. We apply this model to our solution, and provide a reference model analysis. SVG Mobile 1.2, being MORE's primary presentation format, serves as a host language for SMIL and derives several features from it:

Granularity of Temporal Model: The clock values for MORE have the same syntax as in SMIL animation, with the temporal relationship expressed as 'h:min:s:ms.' Therefore the fine-grainedness is unlimited and dependent on the precision of the synchronization mechanisms. The presentation time indicates the position in the timeline relative to the document begin of a given SVG scene. Animations as well as embedded continuous media can have start and end times, and duration length in presentation time.

Interaction: MORE provides additional functionality than just simple navigational interaction between documents and local media control interaction as shown in Section 7.

Extensibility: MORE is based on open standards, and is consistent with the semantics of different component technologies from W3C, 3GPP, OMA and IETF. Further, the architecture is not tightly bound to a particular solution, for example compression, allowing the flexibility to accommodate more optimal methods if available.

Reusability: MORE supports reusability of media elements, content fragments and documents by inheriting this functionality from SMIL.

Adaptability: MORE supports dynamic adaptation to preferences by using the <switch> tag borrowed from SMIL. Also, packet size and error concealment schemes can be adapted based on the network conditions, priority of content and the application.

Presentation-Neutral Representation: Using SVG as the primary presentation format in MORE, a complete presentation description can be provided, independent of the rich media player.

Real-time Application Support: MORE provides scenes and scene updates that are streamed in real time along with associated media.

QoS Parameters: In addition to standard QoS parameters defined for multimedia applications [10], we define a new set of QoS parameters to cater specifically to rich media solutions as described in Section 6.

10. CONCLUSION

We present solutions that address various technology components needed for providing mobile real-time, interactive and streaming services. The solutions include dynamically delivering and updating scene content, a storage format for SVG content based on the ISO Base Media File Format including media synchronization, transport mechanisms and packetization for SVG and its discrete/continuous embedded media, and user interaction. The MORE standard could potentially have positive implications in W3C, 3GPP and OMA standards.

11. REFERENCES

[1] 3GPP: Dynamic and Interactive Rich Multimedia Scenes, http://www.3gpp.org/ftp/Specs/html-info/WiSpec–34032.htm.

[2] OMA-RD-RichMediaEnvironment-V1_0-20050825-D, August 25, 2005.

[3] Scalable Vector Graphics (SVG) Mobile 1.2 Specification, http://www.w3.org/TR/SVGMobile12/.

[4] ISO/IEC 15444-12:2005, "Jpeg 2000 image coding system - part 12: Iso base media file format," 2005.

[5] JSR226 Scalable 2D Vector Graphics API for J2ME™, http://www.jcp.org/aboutJava/communityprocess/review/jsr226/.

[6] Macromedia Flash, http://www.macromedia.com/flash.

[7] Jean-Claude Dufourd, Olivier Avaro, and Cyril Concolato, "An MPEG Standard for Rich Media Services," 2005, vol. 12, pp. 60–68, IEEE Computer Society Press.

[8] Synchronized Multimedia Integration Language (SMIL) 1.0 Specification, http://www.w3.org/TR/REC-smil/.

[9] Character Sets, http://www.iana.org/assignments/character-sets.

[10] Boris Rogge, Jeroen Bekaert, and Rik Van de Walle, "Timing issues in multimedia formats: review of the principles and comparison of existing formats," 2004, vol. 6, pp. 910–924, IEEE Computer Society Press.