

An Approach to Interactive Media System for Mobile Devices

Eun-Seok Ryu

Chuck Yoo

Department of Computer Science and Engineering, Korea University

Anam-dong, Sungbuk-ku, Seoul, Korea

{ esryu, hxy }@os.korea.ac.kr

Abstract

The interactive system which interacts human with computer has been recognized as one direction of computer development for a long time. For example, in cinema, a person gets the information he wants or she plays the media data while moving, by using a mobile device. For the development of this system, we designed and implemented a system that interacts with users in a small terminal. Our study has three categories. The first category is the development of a new markup language (IML) for the writing of interactive media data. The second category is the *IML translator*, which transfers to the mobile terminal and translates the best form to play. And the third category is the *IM player*, which plays the transferred media data and interacts with user. The IML(Interactive media Markup Language) was designed for controlling vector graphics and general media objects in detail and supporting synchronization, is considered as an advantage of SMIL, among each media object. Also, it was designed with many ideas to be operated in small mobile devices as well as a desktop PC or set-top box which has high CPU performance. The player, implemented finally, operates on PDA (HP iPAQ) and plays the multimedia data that consist of vector graphics (OpenGL), H.264 and AAC etc. according to the choice of the user. This system can be used in the ways of interactive cinema and interactive game, and can substitute new interactive web services for existing web services.

Keywords: Interactive Media, IML, Mobile Device.

1. Introduction

Interactive media which is served by user interaction has recently become a hot topic in the areas of human-computer interaction (HCI). Though the information service on a mobile device has become commonplace, most

of the research of the interactive media targeted desktop PC and the TV set-top box platform on wired network. So, it is becoming very attractive to offer an interactive media system on a mobile platform. For these reasons, we designed a whole interactive media system and implemented it. First, we designed a new markup language in detail for supporting the interactive media service which is most suitable for a mobile device. IML (Interactive Multimedia Language), which is designed for user interaction, has sync-information to synchronize various media objects and descript information of those objects. And it can control each media object directly by supporting a raster graphic module and a core OpenGL, which is based on vector graphic, on language level. For example, it can represent video content on a face of a 3-dimensional rotate cube composed of a vector graphic. Second, we designed and implemented the *IML Translator* which takes charge of the midway process for playing IML contents on a mobile device which has low resources. Through this process, we eliminated the XML parser in a mobile device and saved memory and CPU resources. Third, we also designed and implemented a player for viewing these intermediate contents on a PDA. To support this, we ported the codec of OpenGL, H.264 (MPEG-4 Part 10), AAC (Advanced Audio Codec) to PDA. After these all, we made some interactive media contents (e.g. Interactive Cinema, Interactive Game, and Psychological Test) for demonstrating at a ubiquitous area workshop.

In this paper, we will discuss a method for supporting interactivity to media on a mobile device and describing a internal system architecture in detail.

2. Related Works

2.1 MIT Media Lab - Interactive Cinema

The MIT media lab is a well known 'Interactive Cinema'. Interactive Cinema reflects the longing of cinema to become something new, something more complex, and something more personal, as if in a 'conversation with an

audience' [1]. But, they did not consider transmitting on a wireless network and saving the CPU resources for a mobile device and object based on a media data treatment.

2.2 ETRI - Intellectual Multimedia Broadcast

It is expected that an intelligent broadcasting service (IBS) will be able to provide broadcast programs based on user preference and program-associated information (metadata) in order to assist users in easy navigation of the program content being broadcast [2]. It has been researched at ETRI (Electronics and Telecommunications Research Institute) to support interactivity with a MPEG-4 system (ISO/IEC 14496-1) and an MPEG-7 [3]. It manages scenario trees by using a BIFS (Binary Format for Scenes) module in a MPEG-4 system. But, it was designed to work on a TV set-top box and was not considered as a wireless network condition. So, this system is not appropriate for a mobile device and wireless home network condition.

2.3 W3C – SMIL

SMIL (pronounced smile) stands for Synchronized Multimedia Integration Language. It is an XML-based markup language and is designed to be easy to learn and deploy on Web sites. SMIL was created specifically to solve the problems of coordinating the display of a variety of media (multimedia) on Web sites. By using a single time line for all of the media on a page their display can be properly timed, coordinated, and synchronized. Using SMIL, an author can describe the temporal behavior of a multimedia presentation, associate hyperlinks with media objects, and describe the layout of the presentation on a screen [4][5][6].

In our research, we adopt a temporal synchronization idea in SMIL, and added it to our newly invented IML definition.

3. System Architecture

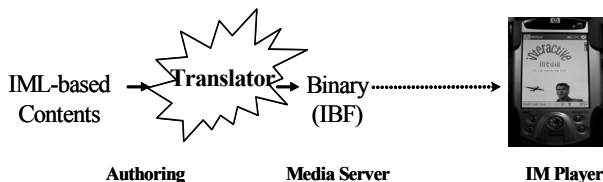


Figure 1. Interactive Media System

As shown figure 1, our proposed interactive media system is composed of a server part and a client part. The server part translates an XML-based content to a intermediate binary format. After all, this IBF (Intermediate

Binary Format) file is played on a player of a client part. Each part is discussed below.

3.1 Interactive Media Language (IML)

The characteristics of IML are as bellows.

- Serviced as byte-aligned binary format.
- Easy to handle for it's similarity with SMIL.
- Embed vector graphic (OpenGL) functionality.
- Can control media object directly.

The proposed IML is represented in a byte-aligned binary format. The binary format reduces the size of interactive multimedia contents, and the byte-aligned format reduces the processing cost of terminals. Another feature of the IML is vector graphics, which assist the terminals to obtain user interaction easily and gives more flexibility to the contents. Vector graphic, which has strength in zooming, can be used on various screen sizes. For example, small buttons (e.g. labeled 'previous', 'next' and 'stop') made in vector graphics are useful to take interaction from users, because they can be maintained in clear shapes according to sizes [7]. Another merit of IML is that it can control each media object directly, because it has a video control module and an OpenGL control module.

The proposed IML enables us to create interactive multimedia contents which not only have the same features as written by SMIL, but also provides much more interactivity than SMIL does. We found that our IML gives authors rich functionalities to create an interactive multimedia suitable for a ubiquitous computing environment [8].

IML has several functional modules for supporting temporal-spatial synchronization and user interaction as bellow.

- Structural and Media Object Declare Module
- Animate Module
- Content Control Module
- Prefetch Control Module
- Layout Module
- Linking Module
- Media Object Module
- Meta-information Module
- Time Manipulation Module & Transition Effects Module

Each module has detail descriptions. And an example of Layout Module is as bellows.

Module ID	Element ID	Description
0x4	0x0	<layout>

	0x1	<region> backgroundColor: (24 bits) leftTop: (32 bits) width: (16 bits) height: (16 bits) z-index: (8 bits)
	0x2	<regFlag> leftTop: (32 bits) z-index: (8 bits)

Table 1. An example of Layout Module in IML

Another merit of an IML is that it supports some functions of an OpenGL. For that reason, IML can combine a raster object and a vector graphic object. For example, it can represent video content on 3-dimensional vector graphic rotate cube. Functional modules of an OpenGL supported by IML are as below [9].

- Primitive function
- Viewing function
- Transform function
- Input function
- Control function

As mentioned above, IML was designed as an interactive media language on a mobile device and provides much more interactivity than SMIL. More detailed information about the IML modules (table 1) was treated in our previous papers.

3.2 IML Translator

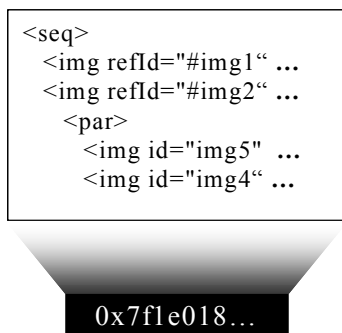


Figure 2. Translating of IML Translator

An *IML Translator* removes the redundant XML parsing work on a mobile device by translating a content file from an XML-based text type into a binary type, in advance. So, the advantage of this *IML Translator* is that it does not need an XML parser, saves CPU power, and reduces content size (about 70% of XML-based content) [6]. And it uses a byte-aligned format for reducing bit operations,

because mobile devices do not have enough calculating power.

3.3 Interactive Media Player

The structure of an interactive media player is as shown below. And it is composed of *Object Scheduler*, *IBF translator*, and *Media Codec Decoder*.

3.3.1 The Architecture of IM Player

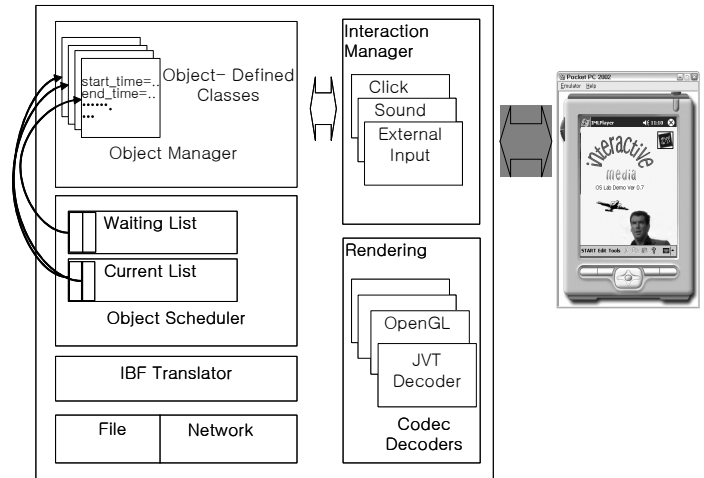


Figure 3. Internal architecture of an IM player

In detail, after content is transferred to the *IBF Translator* by a network or a file system, it is translated and analyzed. Then, the timestamp information which indicates when the object has to be started and ended and the event information which is related to a user interaction are registered to an internal list. Each *Waiting List* and *Current List* contains a pointer value which indicates objects will be played and the object playing now. To manage these internal lists (*Waiting List* and *Current List*), an *Object Scheduler* was designed and implemented [10].

3.3.2 Intermediate Binary Format Translator

IBF Translator means Intermediate Binary Format Translator, and parses a binary content which was converted by an *IML Translator* and assigns an object ID and timestamps to the *Waiting List* of the *Object Scheduler*. By this processing, an *IM Player* knows when it treats what object. An *IBF Translator* is faster and lighter than an XML parser, because it doesn't treat a text-form data but a binary-form data.

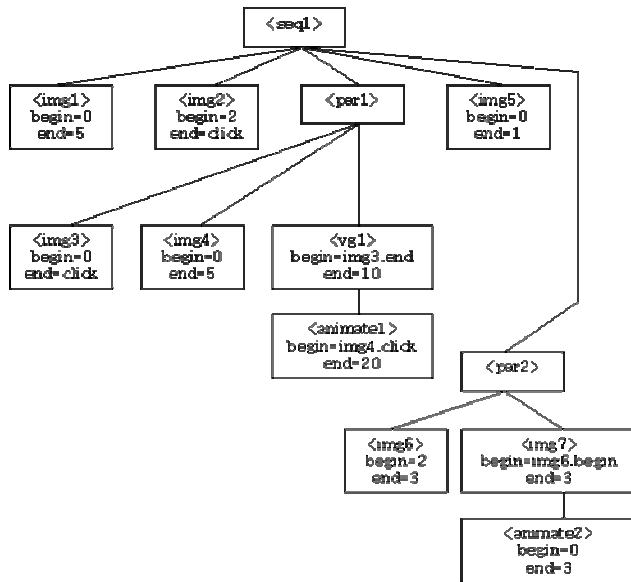


Figure 4. Object tree for *Waiting List*

3.3.3 Object Scheduler

Multimedia Object	Begin	End	Parent
<seq1>	0	last	none
<img1>	seq1.begin	5	<seq1>
<img2>	img1.end+2s	img2.click	<seq1>
<par1>	img2.end	last	<seq1>
<img3>	par1.begin	img3.click	<par1>
<img4>	par1.begin	5	<par1>
<vg1>	img3.end	10	<par1>
<animate1>	img4.click	20	<vg1>
<img5>	par1.end	1	<seq1>
<par2>	img5.end	last	<seq1>
<img6>	par2.begin+2s	3	<par2>
<img7>	img6.begin	3	<par2>
<animate2>	img7.begin	3	<par2>

Table 2. Example of *Waiting List* by figure 4

The Player serves synchronization by managing a *Waiting List* and a *Current List* strictly. At this time, the *Object scheduler* decreases the remaining time value in *Current List* by a timer event and checks if there is any object which has to be moved to a *Current List* in *Waiting List*. As shown figure 3, this moved media object in *Current List* is decoded by a *Codec Decoder* on every timer event.

3.3.4 Codec Decoder

The *IM Player* supports H.264 (JVT, MPEG-4 Part 10) a video codec and an AAC (Advanced Audio Codec) audio codec used in MPEG-2 and MPEG-4 [11]. And it also supports a vector graphic module (it supports some function of OpenGL), and a BITMAP, WAV codec [12].

Now, we are trying to support an MPEG-4 part 2 and a MPEG-1 Layer III (MP3) codec to an IM player. The codec were supported by an IM player which existed in the form of a library and were ported to an *IM player* after being optimized.

4. IM System Demo

The result of this research was published and exhibited at the 'International Ubiquitous Computing Symposium (Dec. 2003)' and 'HCI2004 Ubiquitous Workshop (Feb. 2004)'. During those exhibitions, many people enjoyed and were satisfied with our interactive media system.

As mentioned above, this system supports a 'JVT (H.264, MPEG-4 Part 10) Simple Profile' for video codec. The H.264 codec has high compression power (about twice MPEG-4 Part 2), it appropriates to a PDA which has low resources (e.g. low memory, low bandwidth wireless network) [13]. Though it needs high CPU power to decode content, it can decode over 15 frames per second in our implemented system. Because this system was designed to use very low resources, an H.264 codec which is in IM system was newly optimized by us.



Figure 5. Implemented IM Player (HP iPAQ)

The Above figure shows a demonstration of scenes implemented by our interactive media player. The left side figure introduces the scene of an interactive cinema written in IML, and it shows a moving airplane by controlling the vector and raster graphic object. After these intro scenes, the cinema moves on to various stories as user interaction. On the right side, the two figures are scenes of playing the IML game content whose name is 'Find differences'. This interactive game is going on the next stage when a user finds all of the differences between the two pictures.

5. Conclusion

In this paper, we have presented an interactive media system in a mobile environment. It is an interactive, lightweight, and resource-efficient system. The first contribution of this research is that it is not a one-way system but an interactive system, and the second one is that it is designed for a mobile device like a PDA in the first beginning time unlike the other systems which are designed for a high powered desktop PC or home server. Therefore, a user can enjoy an interactive media service as his opinion while he is moving. To put it more concretely, since a content written in IML is converted to a binary format, it has a lower data size. Moreover, it also needs a lower network bandwidth and does not need an XML parser, which is a burden on the memory and CPU resources. Another merit of the IML is that it can control each media object directly, because it has a video control module and an OpenGL control module in its internal library. Consequently, objects written in internal vector graphic interface can be reused and reduce network connection cost.



Figure 6. Demo scenes of interactive cinema content

In a point of wide view, we defined a novel interactive markup language IML, implemented an IML translator and an IML player in this research. The examples of practical applications of our developed system are an interactive cinema player, an interactive game player, and interactive web browser on a mobile device. Moreover, it can be applied to many media fields without complicated programming if a sensor or a voice is used for an input event, and it can substitute new interactive web services for existing web services.

6. Acknowledgement

The research reported herein was supported in part by Digital Media Lab.

7. References

1. MIT Media Lab Interactive Cinema, <http://ic.media.mit.edu/>
2. Munchurl Kim, Jeongyeon Lim, Kyeongok Kang, and Jinwoong Kim, ETRI Journal, vol.26, no.2, Apr. 2004, pp.136-148.
3. ISO/IEC 14496-1 Information Technology Generic Coding of Audio Visual Objects Part 1 : Systems
4. Libwww The W3C Protocol Library, <http://www.w3.org/Library>
5. Synchronized Multimedia, <http://www.w3.org/AudioVideo>
6. Chia-Yuan Teng, Compression of SMIL Documents", Data Compression Conference 2000. Proceedings DCC 2000, 2000.
7. Scalable Vector Graphics (SVG), <http://www.w3.org/Graphics/SVG>
8. Mi-Ha Kim, Eun-Seok Ryu, Jin-Hwan Jung, Hyuck Yoo, Interactive Multimedia Language for Mobile Media Player, KIPS Fall Conference, 2003.
9. OpenGL ES Overview, <http://www.khronos.org/opengles/>
10. Eun-Seok Ryu, Mi-Ha Kim, Hyuck Yoo, Design and Implementation of Interactive Media System Using Mobile Device, KIPS Spring Conference, 2004.
11. Joint Video Team(JVT), <http://www.itu.int/ITU-T/studygroups/com16/jvt/>
12. 3GPP Specifications, <http://www.3gpp.org/specs/specs.htm>
13. ISO/IEC 14496-2 Information Technology Generic Coding of Audio Visual Objects Part 2 : Visual
14. Perlin, K., Goldberg, A. Improv: A System for Scripting Interactive Actors in Virtual Worlds, SIGGRAPH'96 Conference Proceedings, pp. 205-16, 1996.
15. Hari Kalva et al, Implementing Multiplexing, Streaming, and Server Interaction for MPEG-4, IEEE Transaction on Circuits and Systems for Video Technology, vol. 9, no. 8, pp. 1299-1312, 1999.
16. Hjelsvold.R, Vdaygiri.S., Web-based Personalization and Management of Interactive Video, 10th WWW Conference, Hong Kong, 2001.