Mathematical Problems of Computer Science 36, 133--139, 2012.

Interactive Multimedia Publication System for Building Web Services

Ruben Vardanyan

IT Educational and Research Center
e-mail: ruben@emagy.com

Abstract:

Interactive Multimedia Publications are of high importance in many application areas including education, training and entertainment. It explains the high demand in web services providing the effective tools of authoring and interacting with this kind of publications. This paper is aimed to present the Interactive Multimedia Publication System, which can be used for building the above mentioned web services and having capabilities significantly exceeding those of the existing ones. The paper gives the high-level architectural view of the systems and outlines the communication protocol used between the client and server components of the system.

Introduction

In recent years the landscape of the web is rapidly changing, absorbing more and more applications.Advances in web technology allow the publishers (universities, book and magazine publishers, etc.)to start transitioning from the printed media to online interactive multimedia publications gaining thus in publishing time, expressiveness of the final publication and overall cost reduction.To simplify the adoption of the modern publication methods by the readers and authors there is a need in having enhanced interactive multimedia publication web servicessupporting the traditional paper format and tightly cooperating with the Print on Demand services, supporting wide range of user devices including smartphones and tablets and providing effective authoring tools for embedding the rich media and interactivity elements into publication. The existing commercial services cover just some part of the above listed requirements and research works focus mainly on specific areas related to reuse of the multimedia resources [1], scheduling multimedia synchronization [2,3] and building the multimedia document models and frameworks [4,5].

This paper is aimed to proposea complete solution meeting the above mentioned requirements - the Interactive Multimedia Publication System (IMPS) to be used for building the advanced interactive multimedia publication web services with capabilities significantly exceeding those of the existing ones.An essential part of IMPS is the Extensible Interactive Multimedia Transfer Protocol (XIMTP) specially developed for communication between the IMPS clients and server.The need for developing a new protocol versus to using an existing one has emerged by the necessity of having a communication protocol behaving well in high-traffic client-server systems (web services) with low performance devices (like smartphones and tablets) located at the client side.

2. Interactive Multimedia Publication System Architecture Overview

The system consists of the IMPS Client (IMPSC) and IMPS Server (IMPSS) components.The client componentsare running on the client devices (PC, smartphone, tablet, etc.) and their interiors may vary with the underlying platform. However, irrelevant to what platform is in use, all clients have a similar layered architecture (see Figure 1) where each layer is responsible for a specific task. Many client components may run at the same time on different client devices and work with the same server and the same interactive multimedia publication (IMP).

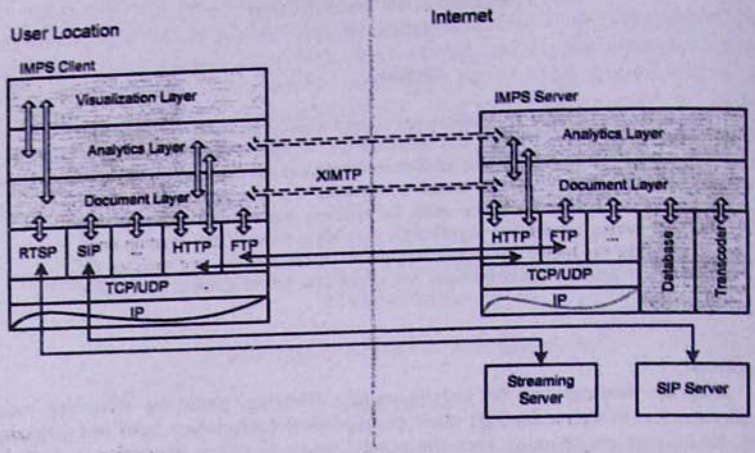

Figure 1

The Visualization layer is responsible for displaying the publication on the client device. It gets the skeleton of the whole IMP (containing the information about its size, title, description, number of pages, etc.) from the Document layer and then loads up the structure and design elements (content) of the page which should be displayed first.After that it startspre-buffering the other IMP elements at the quality and resolution relevant to the Internet bandwidth and screen resolution.The sequence in which these elements will be pre-buffered is determined by the Analytics layer, based on the user behavior analysis described in more details in [6].

The Analytics layer registers the user actions and provides information to Visualization layer (as mentioned above) and IMPS server. The list of the user behavior parameters tracked by this layer includes but not restricted to, mouse clicks on corresponding IMP objects, average page view time, readers geo-locations, unique views, browsers used, etc. The information received by the server is utilized for the user behavior analysis and further system optimization as well as for building reports for the interested parties (authors, publishers, service providers, etc.).

The Document layer retrieves the publication information from the server on request from Visualization layer. Based on the hardware and software parameters of the user device, it builds and returns to Visualization layer the document with the required settings. For example, if the end user views the publication on a tablet device thenthe Document layer will build the document with predefined templates designed for such device.

The underlying Data layer includes the standard protocol stacks used for transporting the messages and other data between the client and server and their description is out of scope of this paper. The protocol stacks are provided usually by the client device basic software and are not part of the IMPSC.

The IMPS Server component runs on one or many server machines located in data center (at the hosted service provider). Similar to IMPS Client it has a layered architecture with the client's Analytics and Document layer counterparts in it. The main feature of the server component is to provide the client with IMP related data necessary for visualizing the IMP on the client device. This feature is supported by the server's Document Layer communicating with its counterpart on the client machine via XIMTP (Extensible Interactive Multimedia Transfer Protocol) protocol discussed below.

Some part of the IMP data is stored in IMPSS database. It is mainly the publication metadata describing the IMPformat as well as the basic content elements like texts, geometric shapes and possibly the images. The other part is the data which is external to IMPSS like, for example, YouTube video or video streamed from IP cameras. In case the client retrieves the external data, the server resolves the external address (URL) and refers the client to that address. From that point the client component connects directly to that external address via protocol specified in URL. The example on Figure 1 shows that the client component is connected directly to the streaming server via RTSP [7] protocol.

The IMP elements are stored in the database with some redundancy. The same element might be stored in many different formats;for example, the image can be stored in low, medium and high resolutions. Depending on the type of the client device and available Internet bandwidth, the server makes decision on the image type to be sent on the client's request. For example, if the client is accessing the server from a smartphone with a low resolution screen or the internet bandwidth is shared then the server will send the low resolution image otherwise, it may opt for a high resolution image. In case the stored dataformat is not supported on the client device, the Document Layer may ask the Transcoder to change the data format to one of the supported on the client device. For example, if the client device supports the video in FLV [8] format only then all video files will be converted to that format on the server before sending to client.

3. Extensible Interactive Multimedia Transfer Protocol

The Extensible Interactive Multimedia Transfer Protocol (XIMTP) is used in IMPS for communication between the clients and server. XIMTP is a stateful, application level protocol based on XML for its message format and using HTTP as a transport.

The XIMTP messages consist of the three main sections: *Header*, *Request* and *Response*. Any of those sections can be empty, but not omitted. The skeleton of an XIMTP message is presented in Example 1.

```
<envelope>
<header>HEADER DATA</header>
<request>REQUEST DATA</request>
<response>RESPONSE DATA</response>
</envelope>
```

Example 1.

The *Header*includes the protocol metadata like user identification string (GUID), protocol version,version compliance information, etc. The content of this section is used for the initial handshake, session establishment between the client and server and further session support. GUID is the main element of the *Header* that allows the client and server to bind the messages in one session. Even though the session support adds complexity to software at both the client and server ends (compared to stateless protocols used for building the web services [9]), it brings

tosignificant data traffic reduction between the client and server resulting in better response times and interactivity improvement. Example 2 below presents an XIMTP client message requesting the publication's next page from the server.Due to session support there is no need in passing the publication identity or page number as that information can be extracted from the user session information which is identified by GUID.

```
<envelope>
<header>
<guid>b0e7fea59e8e297ae3daed7fbe9e0eac</guid>
<header>
<request>
<getNextPage></getNextPage>
</request>
</response></response>
</envelope>
```

Example 2.

The *Request* section of the XIMTP message is used for Remote Method Invocation (RMI). It contains the tags for each method to be invoked and nested tags for method arguments. The client side message in Example 3 requests forthe page number 4 of publication with identity number 105.

```
<envelope>
<header><header>
<request>
<getPage>
<publicationID>105<publicationID>
<pageNumber>4</pageNumber>
</getPage>
</request>
</response></response>
</envelope>
```

Example 3.

The *Request* section may contain multiple method invocation requests. They are executed in the order they are listed in the message. The execution results are returned in the *Response* section described below.The *Request* section can be filled in not only by the client but also by server. In the latter case the methods are invoked on the clientside. However,the use of the client-server approach sets a limitation on the server side requests:a) the server can send its request only in XIMTP response message to client;b) the set of methods that can be invoked by the server are limited to a few methods controlling the media objects on client device (like start and stop the video, etc.).The basic set of methods allowed for invocation on the client side includes the methods that allow the client to request the user authentication from the server, get publicationgeneral data, its pages,embedded objects, etc.

Response section is intended for returning the output of the RMI execution, which includes the status return value as well as an XIMTP described IMP data discussed below.

4. Interactive Multimedia PublicationFormat

When designing IMP format, the following objectives were pursued:

- Simplicity of splittingthe IMP into parts for hosting on different servers
- Abstraction of the document layout for easiness ofporting from one layout to another
- Rights management support, preventing the unauthorized users from reading or downloading the IMP
- Interactivity support with ability to integrate the rich media elements in publications, including video and audio streams, animations and user-interaction elements

IMP is rendered on the user device in the form of a regular printed document with one of two pages opened on the screen at any time. The way how IMP is presented on the screen may vary depending on the type of device but the underlying document format is static across all devices.IMPhas hierarchical structure with the *Document* object on the top of hierarchy (see Figure 2). This object contains information pertaining to IMP as a whole (including title, size, etc.) as well as the *Page* objects.

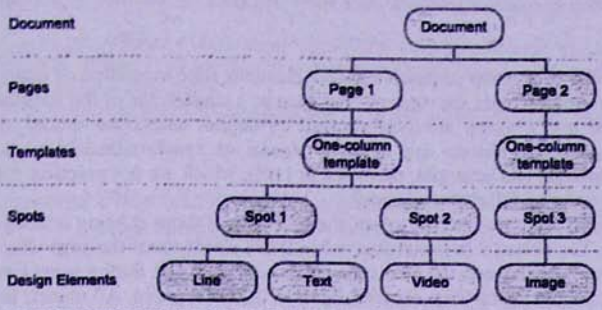

Figure 2

The *Page* object holds information about the IMP page as a whole (like page number, background image, pre-indexed texts for fast searching, fonts used in it, etc.) as well as the collection of *Spot* objects.

Spot is an editable section of the page containing the actual design elements, which can be based on a pre-designed template.It is also in charge of the user access control and dynamic data synchronization (like video and audio).

Design Element holds information about the object data, its placement, dimensions and transformations. There are two types of design elements: passive and active. The examples of passive objects are static texts, shapes (like line, oval, rectangle, etc.), images, etc. The set of the active objectsincludes video, audio, animations,effects, etc.Example 4 presents an on-demand video object, which has a solid black border and reads the video from the file.

```
<video>
<id>54978521</id>
<name>my video</name>
<version>1</version>
<points>-132.5,132.5,-132.5,-132.5,132.5,-132.5,132.5,132.5</points>
<transform>1,1,0.0,153.5,187.5</transform>
<frame>
<thickness>2</thickness>
<lineStyle>solid</lineStyle>
<color>black</color>
</frame>
<opacity>1</opacity>
<source>
<stream>
<type>ondemand</type>
<url>file.a44d84e78764cc034064b62ccdb865cb</url>
</stream>
</source>
</video>
```

Example 4.

The next example (Example 5) specifies the animation object moving the image along the path during a given time frame.

```
<animation>
<object>
<image>
<points>0,0,100,0,100,100,0,100</points>
<transform>1,1,0,0,100,100</transform>
<file>a44d84e78764ce034064b62eedb865cb</file>
</image>
</object>
<type>motion</type>
<path>0,0,400,0</path>
<duration>5</duration>
<fill>freeze</fill>
</animation>
```

To apply an action to a group of passive design elements (like animation of many images at the same time), the protocol uses the <group> tag used as a constructor of the complex object. The objects included into a group are synchronized by default unless the special synchronization rules for the group of objects are defined. Versus to synchronization methods using the scheduling algorithms (for example, proposed in [10]), which are not practical for using in web services, IMPS uses the following approach.

For each object included into the group, the sequence of states is being defined when creating the publication. The object is in initial state when IMPS client opens the page where that object is located. Object may move to the next state either automatically, during some timeframe, or upon getting a specific signal from another object in the same group. All objects included into group "emit" signals every time when moving to a new state. Arriving at the final state the object either stays in that state until the IMP page closes or gets back to the initial state. To illustrate the idea of the proposed synchronization method on example, let's assume that we have the following objects in the group:

- "Play" button B with some initial state B_0 and final state B_1 when the button is pressed by the user
- Video object V with the initial state V_0, state V_1 somewhere in the middle of the video and the final state V_2 at the end of video
- Another video (let's say video advertisement) object A with the initial state A_0 and final state A_1.

Now, if we want to start the video V when button B is pressed, stop it in the middle and play advertisement A and then continue V when A finishes, we have to define the following rules for this group of objects.

- Start moving V from V_0 to V_1 when signal $S(B_1)$ arrives; where $S(B_1)$ is the signal emitted by button B when it is pressed.
- Start moving A from A_0 to A_1 when signal $S(V_1)$ arrives; where $S(V_1)$ is the signal emitted by V when it gets to state V_1 in the middle of video.
- Start moving V from V_1 to V_2 when signal $S(A_1)$ arrives; where $S(A_1)$ is the signal emitted by A when it finishes.

References

1. T. Beckers and N. Oorts, F. Hendrickx and R. Van De Walle, "Multichannel publication of interactive media documents in a news environment", *14th International WWW Conference*, 2005.
2. J. P. Jarmasz and N. D. Georganas, "Designing a distributed multimedia synchronization scheduler", *International Conference on Multimedia Computing and Systems*, 1997.
3. S. Wirag. *Adaptive scheduling of multimedia documents*. Fakultätsbericht 1997/12, Universität Stuttgart, 1997.

4. S. Wirag. "Modeling of adaptable multimedia documents", *In Proc. Interactive Distributed Multimedia Systems and Telecommunication Services; International Workshop, IDMS'97*, Darmstadt, Germany, 1997.
5. K. Selguk Candan, B. Prabhakaran, and V. S. Subrahmanian, "CHIMP: A framework for supporting distributed multimedia document authoring and presentation", ICDE, 1998.
6. R. Vardanyan, "An adaptive method for visualization of the interactive multimedia publications", CSIT, 2011.
7. H. Schulzrinne, A. Rao and R. Lanphier, "Real time streaming protocol (RTSP)", *RFC 2326*, 1998.
8. Adobe Systems Incorporated, /Adobe Flash Video File Format Specification. Version 10.1/, 2010
9. L. Shklar and R. Rosen, *Web application architecture: principles, protocols and practices. 2nd Edition/, John Wiley & Sons*, 2009.
10. J. Hauser, *Realization of an extensible multimedia document model*, University of Stuttgart, 1999.

Ինտերակտիվ Մուլտիմեդիա Հրապարակումների համակարգ՝ վեբ ծառայություններ ստեղծելու համար

Ռ. Վարդանյան

Ամփոփում

Ինտերակտիվ Մուլտիմեդիա Հրապարակումները մեծ նշանակություն ունեն արբեր ոլորտներում, այդ թվում, կրթության, վերապատրաստման եւ ժամանցի ոլորտներում: Դա բացատրում է այս տիպի հրապարակումների հետ աշխատող վեբ ծառայությունների մեծ պահանջարկ: Այս հոդվածի նպատակն է ներկայացնել Ինտերակտիվ Մուլտիմեդիա Հրապարակման Համակարգ (ԻՄՀՀ), որը կարող է օգտագործվել վերը նշված վեբ ծառայությունները կառուցելու համար, որի հնարավորությունները էապես կգերազանցեն ներկայիս գոյություն ունեցողներին: Հոդվածում ներկայացված է համակարգի բարձր մակարդակի կառուցվածքը եւ մուլտիմեդիայի փոխանցման պրոտոկոլը, որը հատուկ մշակվել է ԻՄՀՀ-

Система Интерактивных Мультимедийных Публикации для создания веб-служб

Р. Варданян

Аннотация

Интерактивные Мультимедийные Публикации играют важную роль во многих сферах, включая сферы образования, обучения и развлечения. Этим обоснован большой спрос на веб-сервисы, предоставляющие эффективные инструменты для создания и взаимодействия с публикациями данного типа. Целью данной статьи является представление Системы Опубликования Интерактивной Мультимедиа, которая может быть использована для создания вышеупомянутых веб-сервисов и иметь возможности намного превосходящие возможности ныне существующих систем. Данная статья предлагает архитектурный вид системы и описывает протокол используемый для коммуникации между клиентскими и серверными частями системы.