LAMP-TR-039 March 2000
CAR-TR-939
CS-TR-4111

Multimedia Presentation and Transmission Standards and Their Support for Automatic Analysis, Conversion and Scaling: A Survey

A. Koivisto

Language and Media Processing Labratory
Instititue for Advanced Computer Studies
College Park, MD 20742

Abstract

The increasing popularity of WWW-based services, the rapidly increasing penetration of smart phones and PDAs, and the effect of emerging technologies such as WAP, has awakened the service provider and content producer community to the need for providing lightweight versions of their applications. Another recent trend has been user-adaptive services, which most of the popular search engines already provide through customizable front ends as well as a wide range of audio and multimedia search facilities. Services that facilitate streaming media are gaining popularity but are creating even more stress on the overloaded Internet. High-speed subscriber lines, however, would give the user the necessary bandwidth to use multimedia in its most complex form. The adaptation of services to meet users' settings is usually done without considering media or transmission, thus requiring the user to be aware of many technical details and creating a constant need for them to upgrade their hardware and software. In this survey, we study the most common representation standards and protocols used to deliver multimedia over the public Internet and demonstrate how the information we extract from them can be used in automatic, media-wise adaptation of multimedia to improve the quality of service. We conclude with a comprehensive application example that demonstrates how standard multimedia and transmission protocols can be utilized in application-independent adaptation of multimedia.

***The support of the LAMP Technical Report Series and the partial support of this research by the National Science Foundation under grant EIA0130422 and the Department of Defense under contract MDA9049-C6-1250 is gratefully acknowledged.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
MAR 2000		**00-03-2000 to 00-03-2000**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Multimedia Presentation and Transmission Standards and Their Support for Automatic Analysis, Conversion and Scaling: A Survey	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Language and Media Processing Laboratory,Institute for Advanced Computer Studies,University of Maryland,College Park,MD,20742-3275**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**46**	

Table of Contents

LIST OF TABLES

List of Figures

Abbreviations and Concepts

Term	Explanation
ADPCM	Adaptive Pulse Code Modulation
ASF	Advanced Streaming Format. Microsoft s format for streaming media.
ASN	Abstract Syntax Notation, commonly used to define network protocols and their interfaces. Often called ASN.1.
BER	Bit Error Rate
CDF	Channel Definition Format
CGI	Common Gateway Interface, usually a programming interface provided by a Web server for generating HTML documents on demand. Usually written in Perl or C. Comparable to Java Servlets, which can be considered as Java CGI or server-side Java.
CSS	Cascading style sheets: a mechanism proposed by the W3C to define layout in a specific document aside from the actual text. Two levels, CSS1 and CSS2, the first currently having better software support.
COM	Component Object Model. Microsoft s component model for Windows, although limited cross-platform features exist.
DCT	Discrete Cosine Transform
DMIF	Delivery Multimedia Integration Framework. Part of the MPEG-4 specification for controlling the transmission of the presentation. [37]
DOM	Document Object Model. W3C standard for tree presentation of (SGML-related) documents.
DTD	Document Type Definition
DVI	Digital Video Interactive
EPOC	A 32-bit, small-footprint operating system developed by the Symbian consortium, http://www.symbian.com, targeted for small, portable terminals such as PDA and smart phones.
FAP	Facial Animation Parameters
FBA	Face and Body Animation
FDP	Facial Definition Parameters
GIF	Graphics Interchange Format
GPRS	General Packet Radio System
GSM	Global System for Mobile Communications (Not an official definition; acronym originally from French translation)
H/PC	Handheld/PC, Microsoft s definition for a handheld (having no keyboard) portable computer
HDML	Handheld Device Markup Language
HTML	HyperText Markup Language
HTTP	HyperText Transport Protocol
IETF	Internet Engineering Task Force
IMT-2000	International Mobile Telecommunications-2000, yet another attempt to make a global mobile network
IP	Internet Protocol
ISDN	Integrated Services Digital Network: a multi-service digital network. With two B-channels, it offers 128 kbits of (uncompressed) bandwidth.
ISO	International Standardization Organization
IT	Information Technology
ITU	International Telecommunications Union
JPEG	Joint Photographers Expert Group
LAN	Local Area Network
MPEG	Motion Picture Experts Group, http://www.mpeg.org

PDA	Personal Digital Assistant
PICS	Platform for Internet Content Selection, http://www.w3.org/PICS/
PNG	Portable Network Graphics
POTS	Plain Old Telephone Service
RDF	Resource Description Framework
RSVP	Resource ReSerVation Protocol
RTP	Real-Time Transport Protocol
RTCP	Real-Time Transport Control Protocol
RTSP	Real-Time Streaming Protocol
SGML	Standardized General Markup Language
SMIL	Synchronized Media Integration Language
SMS	Short Message Service
SMTPE	Society of Motion Picture and Television Engineers, http://www.smpte.org
SNR	Signal to Noise Ratio. How much meaningful information a signal carries compared to the amount of noise.
TCP	Transmission Control Protocol
TIFF	Tagged Image File Format
UDP	Unreliable Datagram Protocol
UMTS	Universal Mobile Telephone System
VLBV	Very Low Bitrate Video
VR	Virtual Reality
W3C	World Wide Web Consortium, http://www.w3c.org
WAE	Wireless Application Environment
WAP	Wireless Application Protocol, http://www.wapforum.org
WLAN	Wireless Local Area Network
XML	eXtensible Markup Language.

1 Introduction

Nowadays, most of the content provided by popular services such as the World Wide Web is created manually or extracted from a database using CGI scripts, while some services, such as banking or news services, facilitate real-time connections to legacy systems. The content is served on an as-needed basis to a vast variety of clients ranging from office workers with fast connections to on-the-road workers with cellular links. Very little attention has been paid to utilization of application-independent automatic analysis, scaling and/or conversion of the media itself to provide better quality of service (QOS). Most services obtained from the Web simply assume that the content, which is optimized for large displays and fixed connections, is processed by the client's software, totally ignoring the speed of his connection.

In order to be able to present the content found on the WWW, standard Internet browsers must be able to handle dozens of different formats, resulting in complex and heavy program structure, and sometimes slow and awkward response and usability and the need to update the software constantly. Small-scale terminals having little processing power and memory space are not capable of running software to handle even the most popular formats.

At the same time, the vast amount of information received from the standard HTTP request is often ignored or used only to solve some compatibility problems related to client software. The quality of service experienced by a mobile user is often unacceptable.

The emerging mobile multimedia with a new generation of devices enabled by technologies such as WAP [32], Symbian s EPOC small-footprint yet advanced operating system [31], and third-generation mobile networks, are efficient, on-demand adaptation mechanisms of the web content.

Industry efforts such as Oracle's Project Panama [30], and some academic studies such as experiments done with proxy technology by Fox et al. at Berkeley [33], have been addressing this problem but haven t reached wide acceptance or deployment. Fox et al. proposed a mechanism based on standard HTTP proxy technology that performed on-demand scaling on images contained on HTML pages.

Recently, a new generation of services providing live broadcast of audio and video has gained wide popularity, perhaps due to better support from basic installations of web browsers. Both Microsoft s Internet Explorer and Netscape s Communicator suite offer support for streaming audio with plug-ins such as Realnetwork s RealPlayer G2 or Microsoft s Windows Media Player. On installation, the client is asked to specify the type of network he is using, ranging from modem connection to fixed LAN. Neither of the programs actually utilizes this information when negotiating the quality of service with a broadcasting node, but rather relies on the fixed range of bandwidths created at the service. For example, the popular live service provided by General Broadcasting Service of Finland provides live radio at rates of 28.8 kbps and 56 kbps, from which the client must make his selection. The client is supposed to buffer at least a few seconds of the broadcast and smaller or non-constant bandwidths (due to errors or varying jitter, for instance) are supported with usage of UDP/IP unreliable transmission.

Network resources, or more precisely the lack of them, limit the penetration of bandwidth-hungry, true-multimedia applications comprising both live audio and video feed. The multicast mechanism, which would save on expensive bandwidth, is rarely utilized, as it is not always supported by Internet routers. The problem of missing bandwidth is solved for home users with new (or not that new) high-speed connections such as cable modems, ISDN, and other digital subscriber line solutions to some extent. Still, mobile users will have to wait for better solutions to be able to use the services as they should: quickly and conveniently.

Given the vast diversity of formats, protocols, networks and terminals, it is obvious that multiple versions of the same content are needed in order to service clients efficiently. The process of modifying the content with respect to clients' needs and capabilities is referred to as adaptation. The adaptation is performed in two separate yet linked domains: physical, referring to the real files and objects, and semantic, which deals with the semantics of the presentation. Physical adaptation can be further divided into three classes: analysis, scaling, and conversion [22] [23].

The goals of this study are to:

1. Explore multimedia as a concept and define a general framework for its representation
2. Discuss the most commonly used and standardized forms of multimedia and how it is transmitted over the common network
3. Extract the features from standards that can be used to analyze, scale, convert and distill i.e., adapt multimedia, and map them to a content model and adaptation taxonomy

Where appropriate, we distinguish two cases: what can be done *on-line* and what can be done *off-line*, i.e. before anything is even asked about the estimated cost of a given adaptation task. The focus of this study is on understanding the *representation* of media, whereas the transmission of media is considered within this work to find information about the structure, semantics and metrics needed in the adaptation, such as the parameters we can obtain from content type or client capability negotiation procedures.

In this study, we provide brief backgrounds to the most important and widely used multimedia presentation and transmission standards and discuss how they can be used to support automatic adaptation of multimedia. We begin with the introduction of the Multimedia Content Model (MCM) and Multimedia Adaptation Taxonomy (Chapter 2), both of which form the basis of the framework which will be utilized when exploring various multimedia presentation standards. In Chapter 3, we discuss several multimedia presentation standards and their properties. Chapter 4 addresses the transmission of multimedia in the context of standard protocols, such as the HTTP, and points out some of their features that are usable for analysis and adaptation. In Chapter 5, we demonstrate the usage and benefits of MCM and standard-specific adaptation support with an example. Finally, we summarize and conclude with Chapter 6.

2 Multimedia Content Model

The concept multimedia is often defined as a mixture of several types of media, in the simplest form just formatted text and images. More advanced and complicated cases include mixed audio, video, and text where the data is transmitted to the client in separate flows. Furthermore, the presentations may also contain dynamic, interactive components such as scripts or Java applets.

The diversity of different formats used within the enormous and somewhat chaotic World Wide Web is vast, although efforts such as XML [15], MPEG-4-7 [37] [17], and SMIL [1] attempt to introduce some discipline. Even basic versions of the most popular Internet browsers can handle several markup languages and their versions, image, audio and video formats, and numerous proprietary presentation standards such as Macromedia Flash [18] and Apple QuickTime [19].

In most cases, as the concept multimedia would suggest, a presentation consists of several parts, which are usually transmitted to the client as per separate request or transmission channel or stream. To address the challenges emerging from the diversity of formats and to provide a general framework for handling presentations in an integrated and uniform manner, we have proposed a model to represent multimedia within four layers, the presentation, subscript, object and primitive layers, respectively as depicted in Figure 1. [23]

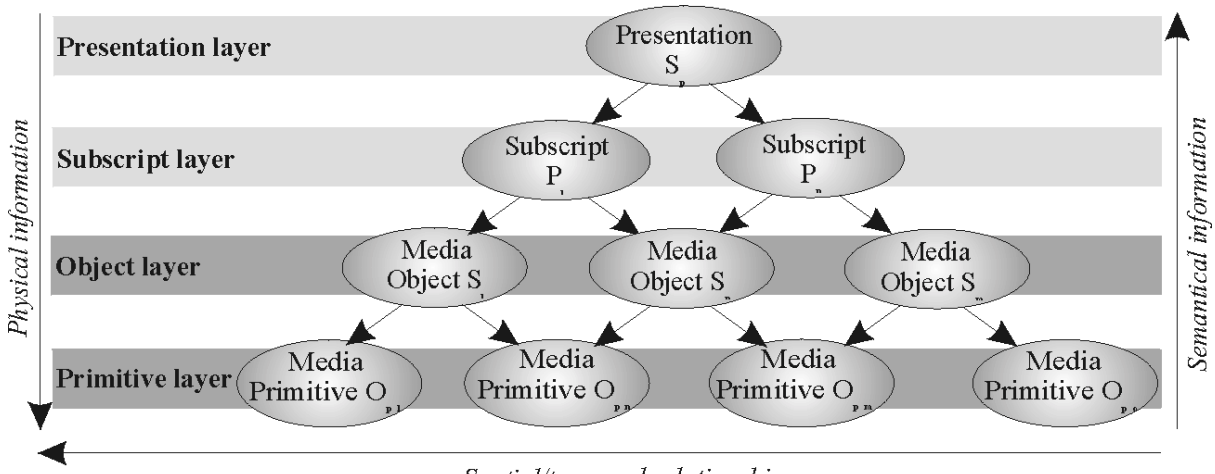

Figure 1. Multimedia Content Model

The connection between the Multimedia Adaptation Taxonomy [22] and the content model is presented at a conceptual level in Figure 2. Whereas *optimization* utilizes the relationships, structure and semantic information of the presentation, which are mostly defined on the presentation and subscript layers, the *scaling* process i.e. physical adaptation is performed on the object and primitive layers. Optimization can be seen as a process, which interprets the client s request and properties, such as his geographical location or current access medium, and combines them with high-level information extracted from the media itself. The scaling process should be seen as a tool to form the final form of the presentation controlled by the constraints and decisions made at the optimization phase.

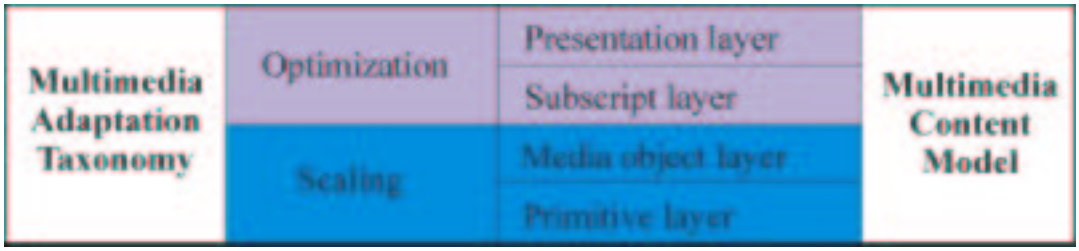

Figure 2. Mapping of Multimedia Content Model and Adaptation Taxonomy

A definition of the model and its layers is provided in Table 1. The *presentation* objects define attributes such as the title and possible textual information related to the presentation and serve as the topmost, root component of the model. The primary function of the *subscript* layer is to represent the content described by the presentation layer objects as singular entities and describe the relationships between them. The *object* layer corresponds to objects that have specific locations on the temporal axis, such as an audio sample, a frame of the video, or an image. Finally, the *primitive layer* contains the encoded forms of the media objects, for instance files broken into two pieces, namely the header and the content.

Typically, objects at lower layers are linked to a single upper-layer object (parent), whereas upper-layer objects may have multiple children, thus forming a tree-shaped structure, whose root is the presentation object and whose leaf nodes are the primitives.

Layer	Represents	Relationships	Notes
Presentation	Semantic content used to form a viewable entity.	Includes one or more subscripts. Objects at the presentation level can be used to form multiple subscripts.	Comparable to concept of service.
Subscript	Distinguishable,	Includes one or more media objects. Has	Viewable as itself;

	semantic part of parent object, e.g. a page, paragraph, video clip, audio track	spatial and semantic relationships to other subscripts and defines the spatial relationships of the media objects belonging to these subscripts.	e.g. selecting audio only from the video would result as a presentation with the audio subscript only
Object	Smallest visible entity, e.g. image, frame of video, audio sample, text section, animation object	Belongs to one or more subscripts (can be shared); semantics of the object are defined as temporal associations to other objects; accurate position on the temporal axis within the subscript. Optionally, also layout relationships to other objects on the same subscript.	Synchronization in playback and transmission; timestamps
Primitive	Binary or textual presentation content; defines the content and formatting of a object	Headers may define multiple content blocks; headers may link to the next primitive (streaming).	Divided into content and header.

Table 1. Layers of the Multimedia Content Model

This model will be used throughout the study to find and point out the features of a given media type or transmission protocol that can be utilized to adapt media, i.e. analyze their properties and use them when performing scaling, conversion and distillation.

It should be noted that not all the standards reviewed in this study directly map to this model, although some models such as DOM [3] can be represented in a similar manner with a tree-shaped structure. Therefore, this model should be viewed as a high-level, conceptual approach to handling multimedia. We suggest, however, that having a uniform, hierarchical representation allows not only efficient and general, binary or encoded format independent media adaptation components, but also makes efficient storage and retrieval of presentations less of an effort. We stress the fact that we are not proposing a new presentation standard or format, but rather a new, semantic approach to accessing and manipulating them.

3 Multimedia Presentation Standards

Multimedia comes in several formats. As the name suggests, one presentation usually consists of multiple pieces that are often in different formats. With the exception of headerless formats, such as the MPEG layer 3 audio, multimedia objects that is, a single yet viewable piece of media, such as an image or a video clip are usually constructed of two parts: header and payload, directly mapping to the primitive concept of our content model. [23]

The header usually defines the format-specific details of the payload, such as the resolution, number of colors or total number of bytes. The payload carries the actual content, be it an encoded image, a video frame, or audio. Generally, but not always, adaptation will affect both parts. For example, reducing the resolution of a digital image will quite obviously alter both the header and payload, while modifying the contents of a markup document can leave the header untouched.

The diversity of formats and their versions are the main driving force behind the most recent activities, such as XML and SMIL by W3C, an organization run by the major players in the IT market. Due to corporate policies and healthy competition, the world is not about to accept one uniform standard. Some of the most commonly used multimedia formats are summarized in Table 2.

Standard	Description
HTML	Most widely used markup language used today; many proprietary extensions. Most recent version 4.01 [9]. Can be represented as a DOM tree [3] with a parser. Can be optimized with style sheets CSS1 [11] and CSS2 [12].
SMIL	Synchronized Multimedia Integration Language [1]. Current version 1.0 with upcoming SMIL Boston release which will offer better XML compatibility and some new features.
MHEG5-8	Multimedia and Hypermedia information coding Expert Group or WG12 is an SIO/IEC/JTC1SC 29 working group, which provides standards for the coded representation of multimedia and hypermedia information objects that are interchanged among applications and services using a variety of media. Their objective is to define the structure of a multimedia hypermedia presentation. MHEG-5: Standard, ISO/IEC 13522-5, which is the base standard for broadcast interactive digital television. ASN.1-based specification of laying out components, for example push buttons, sliders, and graphics. MHEG-8: This new part of MHEG will provide the XML encoding of MHEG-5. MHEG-7 (ISO/IEC 13522-7) defines a test suite that can be used to test an MHEG-5 engine s interoperability and conformance to a specific application domain
Macromedia Flash	A presentation format which allows sophisticated animations, sounds, and graphics to be delivered on an HTML page with a browser plug-in. Standalone mode is available as well. Typically, a plug-in is required to view these. A closed and somewhat proprietary yet versatile way to create multimedia. [18]
Adobe PDF	Actually postscript; includes enhanced features such as smart font compression and inter-document linking. No support for streaming media. A popular format for delivering formatted text documents.
Windows Media Technologies 4	Microsoft s counter-attack in the battle for a streaming media market share; embeds codecs, transmission and playback; server software for Windows NT Server, client software for Windows and MacOS. Many proprietary elements from transmission to encoding.
Apple QuickTime 4	Apple s definition of multimedia; video, audio, pictures, animations, and even VR. Server support for MacOS 8 and Linux, playback software for Windows, Mac, Linux, and some UNIX clones.
RealNetworks G2	A full range of applications including a content production suite, an efficient streaming server, and a SMIL-compatible playback engine offering Java and COM support. Currently perhaps a semi-de-facto standard for internet radio and video broadcast services.

Table 2. Summary of the most widely used presentation standards used on the Internet

Picking the major players from Table 2, Microsoft with their ASF Windows Media Technologies package v4, RealNetworks s G2 and Apple s QuickTime are fighting for the same market share and are not likely to merge under a common standard. All of them support variable-bandwidth streams, embedded graphics, text and images and, of course, audio.

In this chapter, we begin with a definition of streaming presentations in Section 3.1. The properties of static presentations are covered in Section 3.2. We map the content model to the standard under consideration in each of the sections, and point out the features, if any, that can be utilized in automated adaptation.

3.1 Streaming Presentations

A stream is defined as a flow of data having temporal structure. When using digital video in (near) real-time, it is usually transmitted in the form of a *video stream*, which is constructed from a sequence of

packets each containing some portion of the video. Depending on the standard being used, the audio signal either is embedded into packets with the video frames or transmitted over a separate channel, but is usually treated as a separate stream when decompressed by the client software. This requires synchronization; the audio track should be played at the same speed as the video. Transmitting video and audio in separate streams allows clients to select either one of the streams. For conferencing applications, components referred to as mixers utilize streams as well; they can combine several audio and video streams or perform a transcoding to provide the client with the type and rate of video they are capable of receiving.

The properties of streaming presentations related to the scope of this study are as follows:

1. Video streams tend to create a number of substreams, i.e. streams that must be synchronized at the receiver
2. They are often transmitted and viewed in real time; this usually requires buffering from both sender and receiver, and if reliable transmission is required, a separate control channel to retransmit lost packets
3. Video sets strict requirements on the performance of the storage system, transfer medium and playback software/hardware
4. Streaming media is often encoded in the frequency domain and transmitted in *incremental* form: sequential packets refine each other in the frequency domain
5. Streaming media is often transmitted over the network divided into packets

We begin our tour with *streaming presentations* with a high-level standard, namely SMIL in Section 3.1.1. Then we move on to a layer below SMIL in our content model with discussions about Windows Media Technologies (Section 3.1.2), Apple QuickTime 4 (Section 3.1.3), and finally Realnetwork s G2 suite (Section 3.1.4). In Section 3.1.5 we discuss various *video codecs* and what they have to offer for adaptation. We conclude the subject of *streaming presentations* in Section 3.1.6.

3.1.1 SMIL Synchronized Multimedia Integration Language

SMIL [1] allows integrating a set of independent multimedia objects into a synchronized multimedia presentation. According to W3C, the SMIL 1.0 standard is able to

1. Describe the temporal behavior of the presentation
2. Describe the layout of the presentation on a screen
3. Associate hyperlinks with media objects

SMIL is becoming a de facto standard due to wide support from software vendors as a wrapper for displaying video, audio, and other media types. The SMIL Boston upgrade [2] will add some features, such as better XML compliance and enhanced DOM support.

Mapping to Media Content Model

SMIL is based on the XML 1.0 markup language and thus can be used with a wide variety of available authoring, validation and storage tools, most of them available free of charge. It can be represented as a DOM level 1 [3] tree when fed to a parser. A SMIL document, representing a single presentation object, typically contains a number or *regions* and *streams*, which are in turn translated into subscripts having both spatial and temporal relationships. As a SMIL document only defines the structure and references the actual binary or other streams from external sources, these components are translated into media objects, which belong to the subscript under which they were referenced. Primitives can be extracted by obtaining referenced objects and parsing them into content model primitives. However, a SMIL stream can also be a textual document, such as an HTML page or something proprietary (as we will see later with the G2 suite) and must be treated individually. Our content model allows subscripts to contain

subscripts through the spatiotemporal-relationships property, and thus we don t restrict ourselves to a strict mapping of a SMIL stream to the MMCM Object Layer.

Support for Adaptation

A SMIL document can be analyzed either in its textual form or parsed to a DOM tree. Analysis of a well-formed SMIL document can provide us with the following parameters that can be used later in scaling and conversion:

1. A <HEAD> structure similar to the one found in HTML and <META> tags to provide summaries. W3C recommends that at least title, author, and copyright be defined; optionally, the PICS rating can also be defined.
2. Direct mapping of streams and elements to specified regions; regions are formed at the <HEAD> section of the document (layout tweaking, conversion to another format, selecting elements of interest)
3. Actual sizes of different regions (scaling the size streams and elements)
4. Definition of timely relationships between objects through <PAR> (parallel) and <SEQ> (sequential) tags
5. A mechanism to select content based on the pre-defined system bitrate attribute (see Code Fragment 2); however, SMIL doesn t define any mechanism for determining this rate; it is done by the client software and in currently available applications this is done manually during the installation of the client software.

A more complete example, which demonstrates these features, is provided at Code Fragment 1 given below.

```
<smil>
   <head>
      <!-- Presentation attributes. -->
      <meta name="title" content="Video and Ad Template"/>
      <meta name="author" content="RealNetworks, Inc."/>
      <meta name="copyright" content="(c) 1998"/>

      <layout>
         <!--The root-layout sets the height and width of the entire presentation
                                 in pixels.  Each region sets specific areas in the presentation
                                 that media will play to -->
         <root-layout width="468" height="204"  />
         <region id="video_region" width="176" height="144" left="146" top="0" />
                 <region id="pix_region" width="468" height="60" left="0" top="144" />
                 <region id="text_region" width="468" height="144" left="0" top="0" />
      </layout>
   </head>
   <body>
        <!-- Each line between the  tags is a media file which will play to a
                          specified region.  The  tags mean that they will play at the same
                          time (in parallel).  Fill="freeze" means that the final frame will stay
                          visible when that media file is done. -->
      <par>
                <seq>
                   <textstream src="text/text.rt" region="text_area" fill="freeze" />
                   <a href="http://www.real.com/" show="new">
                   <img src="pix/ad.gif?bitrate=20000&preroll=10"
                        region="pix_region" fill="freeze" />
                   </a>
                   <video src="video/video.rm" region="video_region" fill="freeze" />
                </seq>
    </par>
  </body>
</smil>
```

Code Fragment 1. A SMIL example with streaming video and text

```
<smil>
  <body>
    <switch>
      <ref src="audio56.rm" system-bitrate="32000"/>
      <ref src="audio28.rm" system-bitrate="20000"/>
    </switch>
  </body>
</smil>
```

Code Fragment 2. A SMIL Example with selection of bandwidth

To provide an example of an SMIL document as a DOM tree, we have fed a SMIL document containing two parallel components, an audio track and a set of images, to an XML viewer, Figure 3. The XML-compliant structure of SMIL provides several, convenient ways to represent, analyze and modify documents ranging from adding and removing nodes to validating the document against the SMIL DTD (provided by the W3C).

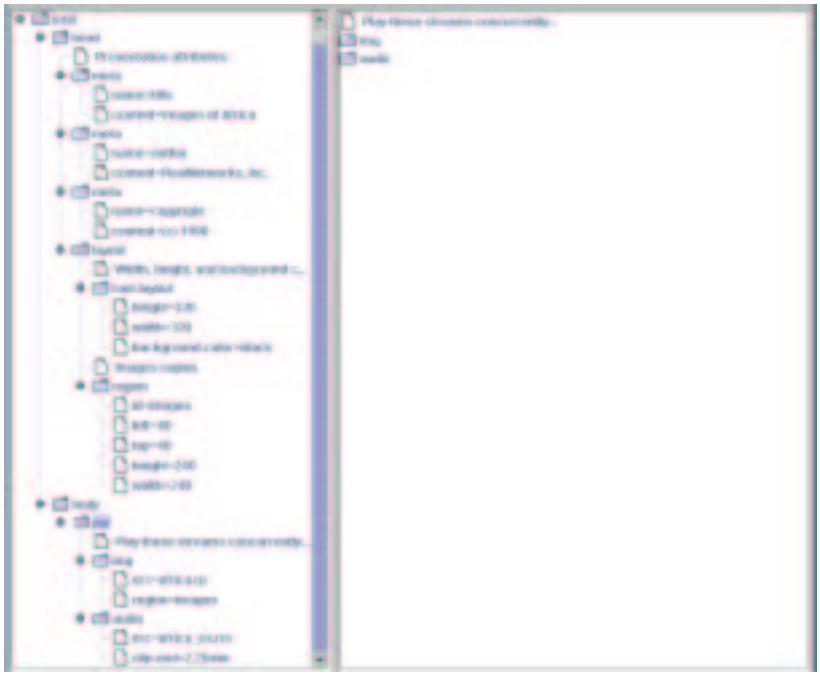

Figure 3. A Sample DOM Tree Generated from a SMIL Document

3.1.2 Windows Media Technologies v4

Windows Media Technologies is a combination of video and audio codecs and a streaming format. The audio codec is based on MPEG1 layer 3 whereas the video is transmitted as low-bandwidth MPEG-4 [37]. Other codecs, such as Intel Indeo and MPEG1-2, are available as well. Unfortunately, as the name suggests, WMT4 is strongly tied to the Windows platform from both client and server sides.

Mapping to the Media Content Model

The core of the WMT is based on the concept of ASF Advanced Stream Format. ASF can embed anything ranging from Office 2000 documents to Microsoft s Video and Audio payload formats (which are built on various video codecs, including MPEG-4 for video and MPEG1 Layer 3 for audio). This breaks the ASF into two layers on the Media Content Model, the upper layer being the ASF definition itself, and the other being the object layer with the actual multimedia payloads.

Unfortunately, decent white papers or specifications were unavailable, whereas excellent and thorough tutorials on how to handle the WMT with ASF or with Microsoft s multimedia suite were easy to find.

Support for Adaptation

As the WMT4 is in fact just one proprietary, closed, and somewhat rapidly evolving standard, its adaptation will most probably, if needed, be done via third-party tools. It should be considered as just one more streaming, transmission and representation format instead of one we would use in adaptation. This holds also for Apple s QT4, which we will discuss next.

3.1.3 Apple QuickTime 4

QuickTime 4 is Apple s response to the streaming media market, which is dominated by RealNetworks and Microsoft. It comprises the entire end-to-end path from production to playback, mostly with standard protocols and codecs. As Apple has been in the Multimedia market for some time now even compared to Microsoft, the QT4 suite offers a great deal of choice and many features. [19]

Mapping to the Media Content Model

With QT4, one is able to author interactive presentations, consisting of a mixture of audio, video, text, animation, and even 3D virtual reality. A summary of QT4-supported encodings is provided in Table 3. QT4 maps into the MMCM on all its layers, but its main purpose and application is to transmit and play video footage, packed into a single, binary file which is not expected to be tampered with once it is created.

Class	Format(s)
Video	Sorenson Video 1&2 (preferred by Apple for variable-bandwidth encoding), MPEG1, H.263, MPEG-4, Indeo (not standard), Cinepak, IMA, AVI, DV, OpenDML, QuickTime Movie, H.261, MS Video 1, MJPEG A&B
Audio	PureVoice, Qdesign Music2, MPEG 1 Layer (1-2 Mac, 3), MIDI, AIFF/AIFC, WAV, GSM, IMA ADPCM, ALAW 2:1, AU, DV, MS DVI, RTP DVI, PCM (8, 16, 24 and 32 bits), G.723 and G.728 (through extensions)
Images	FlashPix, JPEG, GIF, PNG, PICT, TIFF, SGI, Photoshop, MacPaint, Targa
Text	Multilanguage support, streams with video
Animations	MacroMedia Flash, FLI/FLC, 3DMF, GIF, PICS, VR

Table 3. QuickTime4 Media Formats

Support for Adaptation

Although transmission and representation with QT4 is handled with standard formats, QuickTime itself is a closed standard with no complete specifications publicly available. Some applications are able to input QT4 content to some extent (as PPT 2000), but a real-life application able to adapt QT4 content would most probably be based on Apple s tools.

Therefore, the adaptation support directly extractable from QT4 consists only of physical adaptation, i.e. QT can be used as a final representation format in some cases. The codecs used with QT4 video and audio can be utilized in low-level analysis as well.

QT4 offers smooth video even with modem-class bandwidth, which makes it a feasible solution even for small terminals. However, currently Windows CE and EPOC based devices (for example) do not contain QuickTime decoding software as a default, although some support is available from third party vendors.

3.1.4 RealNetworks G2 Audio and Video (RealAudio/RealVideo), Slide shows, Text streams

In addition to supporting SMIL as well as many other popular video and audio formats, RealNetworks G2 suite offers two more formats to cope with: RealAudio and RealVideo. They are both targeted on the

web market as bandwidth-sensitive formats to broadcast video and audio with a technique called SureStream, which actually stores pre-generated versions with different rates in the same file. Their RealServer is even able to switch between the different versions during the playback should conditions change, for example due to replicated or lost packets. This is discussed further in Section 4.3.

The G2 suite also defines two, more or less proprietary, XML-based text formats for slide shows and textual streams. An example of a RealPix slideshow presentation is shown in Code Fragment 3. A similar format is also defined for RealText, an HTML-like markup language. RealPlayer G2 also offers a complete SMIL 1.0 implementation for defining the presentation structure.

```
<imfl>
       <head
         title="Testing 1-2-3"
         author="anttik"
         copyright="©No one"
         timeformat="dd:hh:mm:ss.xyz"
         duration="1:4.718"
         bitrate="12000"
         width="320"    height="240"
         preroll="10.0"
       />

  <!-- Assign handle numbers to images -->
  <image handle="1" name="slides-1.jpg"/>
  <image handle="2" name="slides-2.jpg"/>
  <image handle="3" name="slides-3.jpg"/>
  <image handle="4" name="slides-4.jpg"/>
  <image handle="5" name="slides-5.jpg"/>

  <!-- These effects define the timeline of the presentation -->
  <wipe start="0.0" duration="1.0" target="1" dstx="0" dsty="0" dstw="320" dsth="240"
    aspect=true />
  <wipe start="10" duration="1.0" target="2" dstx="0" dsty="0" dstw="320" dsth="240"
    aspect=true />
  <wipe start="20" duration="1.0" target="3" dstx="0" dsty="0" dstw="320" dsth="240"
    aspect=true />
</imfl>
```

Code Fragment 3. Simple RealPix Example

Mapping to Media Content Model

RealAudio and RealVideo both map directly to the Media Object layer of the Content Model. The packet-enumerated streams map to both media object and primitive layers. RealText and RealPix markups are located on the subscript layer, referencing media objects such as images and text blocks.

The Payload formats used by RealAudio and RealVideo are specific to RealNetworks and thus are not open standards. RealServer and RealPlayer are able to handle some other formats, but this support is by no means unique and will not be discussed any further here.

Support for Adaptation

RealAudio and RealVideo support different encoding schemes for a variety of (a pre-defined set of) bandwidths. The format is proprietary, and despite the fact that the RealServer suite can also generate variable-bandwidth rate video and audio, the adaptation support directly offered by these two standards is limited to this.

For *analysis*, probably the best way to use RealAudio and RealVideo is on the content-producer side. The bitrated streams are designed to perform best on a preset bandwidth, whereas they can be switched before viewing with SMIL s <switch> element. The RealText and RealPix formats can be parsed with XML parsers for the timeline and structure, but as for video and audio, these formats are RealNetworks proprietary standards and are not generally supported by playback engines other than those provided by

RealNetworks. However, if we access RealNetworks-specific components from a SMIL script we can use all the features provided by SMIL. See Section 3.1.1 for details.

Converting the binary encodings of RealAudio and RealVideo files to other video and audio formats is probably feasible. For more advanced conversions, such as speech-to-text or video-to-key frames, RealVideo and RealAudio are considered to be too proprietary and not to offer anything over standard codecs such as MPEG-1 or 2. Conversion of RealText and RealPix to text-only, summaries or captions requires a great deal of application logic. For example, if a slideshow contains both a RealPix-stream and RealText-captions, they must be compared to see which caption maps to which picture when transforming the slideshow to a single-file HTML. Converting from another format to RealAudio and Video streams is possible with the G2 Producer kit, both on demand and in real time, when it's used with RealServer, although the possible source formats and performance of this feature were not tested in detail.

For binary streams, *scaling* comes as a standard from SureStream-encodings and RealServer features. For text and image streams, scaling could involve stripping unnecessary formatting or further compression of the linked images (audio captions are supported with RealPix as well), but whether this kind of adaptation would offer any value is questionable due to the proprietary nature of these formats.

3.1.5 Video Codecs MPEG4

Due to the excessive amount of information generated by even a slow-motion video, the focus in the development of video encoding formats has remained on the compression of data. MPEG-4 [37] and the upcoming MPEG7 address content-based retrieval, with layers and AVOs allowing separate objects to be encoded in distinct chunks instead of one large piece. [14]

Most recent codecs, in particular RealVideo and Microsoft s ASF, address variable transmission and representation environments with variable bandwidth encoding, which means in practice pretty much the same as simple packet-dropping algorithms used with traditional video formats when used in real time, reduction in frame rate and/or size of the footage. Some codecs perform better, resulting in smoother motion or better sharpness, but the video quality is still largely restricted by the available resources.

Even the most space-efficient encodings fail when used to transmit video over a common network when compared to VCR or TV quality. However, the quality can be enhanced in other ways, such as the layered approach in MPEG-4: instead of trying to deliver the entire clip, the user might select the area or stream of interest, as with images. This can be implemented with adaptive encoding; for example, we can use more bits for moving objects than for the background, although delta-encoding actually works exactly this way.

In the case of plain video, a good overview of the video with an option to skip uninteresting parts would be of significant value (for example, one might want to skip commercials), and has been shown to have many advantages over merely toggling some part of the footage, such as turning the background or audio track on and off. Many techniques for constructing static summaries by analyzing the video on a frame-by-frame basis have been developed. Most rely on comparing subsequent frames for rapid movement or finding peaks from the amplitude of the audio track. For some time the best indexer will be a human being rather than a computer program, as many of these methods tend to be more or less dependent on a specific application, and often lack the capability of providing any textual information on these events. The University of Maryland has done a great deal of work on Adaptive Browsing and related technologies [40].

Some noteworthy, generally used video encodings are briefly summarized in Table 4.

Format	Summary
Apple QuickTime [19]	A versatile, codec-independent media format. Able to carry most audio, video and image payloads. A standard developed by Apple Computer Inc. API for Java provided. Consult Section 3.1.3 for details.
RealVideo [20]	Variable-bitrate (when used with SureStream) codec targeted for low to high bandwidth video broadcast. Uses proprietary codecs of RealNetworks. Discussed in Section 3.1.4.
WMT4	Windows Media Technologies, a direct competitor of RealVideo. Multiple codecs, including MPEG-4 for video and MPEG1 layer 3 for audio. Further details in Section 3.1.2.
MPEG-4 [17]	A video codec suitable for low-bitrate applications. Sophisticated compression techniques (VLBV), layering, face animation, 3D animation. See 3.1.5 for details.
MPEG2 [17]	Enhanced version of MPEG-1, a codec intended for high-quality video and widely used with digital television broadcasts, videodiscs, etc.
MPEG-1 [17]	The ancestor of the MPEG codec family. DCT encoded video. Still the most widely used format for video on the Internet; relatively good compression ratio.
H.261, H.262, H.263	Video codecs by ITU-T. Used primarily for videoconferencing, but other applications exist as well.

Table 4. Summary of Video Formats

MPEG-4

The latest release (version 1) of the MPEG-4 standard was published in March 1999. It is gaining rapid deployment with decoders available on Windows32, Macintosh, and even Windows CE platforms, although current implementations not often implement the most advanced features and treat it much like any other video encoding, thus offering very little value over, for example, MPEG-1.

In MPEG-4, presentations are described in terms of AVOs, which are laid out on a coordinate system in a hierarchical manner. According to the standard document:

More generally, MPEG-4 provides a standardized way to describe a scene, allowing for example to:

- *place media objects anywhere in a given coordinate system;*
- *apply transforms to change the geometrical or acoustical appearance of a media object;*
- *group primitive media objects in order to form compound media objects;*
- *apply streamed data to media objects, in order to modify their attributes (e.g. a sound, a moving texture belonging to an object; animation parameters driving a synthetic face);*
- *change, interactively, the user s viewing and listening points anywhere in the scene.* [37]

The standard addresses the concept of media objects as described in [37]: *Media objects may need streaming data, which is conveyed in one or more elementary streams. An object descriptor identifies all streams associated to one media object. This allows handling hierarchically encoded data as well as the association of meta-information about the content (called object content information) and the intellectual property rights associated with it.*

After transmission and demultiplexing of the interleaved streams, MPEG-4 *elementary streams* are decoded and composed into an audiovisual scene, whose appearance (layout) among other things is determined from the tree-shaped structure of the presentation. The leaves of the tree (always Media

Objects) represent the encoded video, audio, images, etc., objects; intermediate nodes can represent various transformations.

Mapping to Media Content Model

The MPEG-4 system defines its independent, tree-shaped structure for presentations, which incorporates features such as definitions of spatiotemporal locations for all objects, synchronization, and even support for limited interaction between the encoder and decoder with the DMIF protocol. Therefore, MPEG4 maps to not just one but all layers in our model, in fact largely resembling it.

The MPEG-4 interpretation of a media object does not map to the media object layer in our content model but instead to the subscript layer. The structure (links between objects and upper nodes of the tree) of the MPEG-4 presentation represents the spatiotemporal structure of the presentation and thus can be interpreted as associations between presentation-layer object(s) and subscript-layer objects.

Supported payload formats

MPEG-4 version 1 defines encoders for several types of audio, video and even images, 2D meshes, faces and textures. They are all MPEG-4 -specific. The DMIF protocol/framework relates to the Session-layer protocol at the OSI model (somewhere between the transport and application layers in the IP model; see Figure 5). The standard document claims that MPEG-4 payload can be carried over any transport-layer protocol, such as ITU-T Recommendations H.22x, MPEG-2 Transport Stream, or IETF RTP.

Support for Adaptation

The MPEG-4 standard offers several features that can be utilized in both on-demand and real-time adaptation.

In the *analysis* phase, the adaptation process can extract (at least) the following information from the MPEG-4 presentation, of course depending on how much the author has provided:

- Number of layers
- What kind of objects are on the layers; metadata
- QoS hints; definition of the resource requirements for each object
- Where the objects occur in at the scene (coordinates)
- Inter-object synchronization information
- The spatial (layout) and temporal relations of objects

Conversion of an MPEG-4 presentation to a single-stream video is quite straightforward, depending on the complexity of the presentation. For synthetic scenes with 3D meshes the conversion would actually mean rendering and composition of the presentation and capturing of the result. For selective conversion, only selected branches or leaves of the tree, such as audio only, could be selected and re-encoded to the desired format.

Scaling of an MPEG-4 presentation can be done either through exchange of MPEG-4 Media Objects with lower-bitrate versions, or simply through reducing the number of concurrent streams; for example, a face animation (FBA/FPA/FDP) based on 3D description and lip-movement primitives, video with a synthetic background, and HQ audio could be scaled down to LQ audio without the background.

The MPEG-4 standard defines encoding for extremely low bandwidth voice, music, and video (VLBV) and could be used as the final format of a presentation over a bandwidth-limited channel.

3.1.6 Adaptation of Streaming Presentations

In the above sections, we have discussed various ways to represent digital video. We distinguish the concepts of a video codec and video format. Usually, a video format such as QuickTime accepts multiple codecs, often even mixing them, meaning that how content is encoded and compressed is often up to the user, naturally within the bounds set by the video standard. This corresponds directly to our content model. The concept of digital video maps to the *presentation layer*, whereas video codecs operate on the *subscript* and *object layers*.

Compared to static presentations with large and high quality images, the amount of information carried by even a small video stream is of an entirely different magnitude. Although emerging compression algorithms and codecs, like the one in MPEG-4 [17], are reducing the bitrate of digital video, the gap will remain for the near future. Another resource-critical characteristic of digital video is the CPU power needed to encode and decode it, although different codecs differ in this constraint considerably. Generally, the most space-efficient encodings tend to require more power, given that we do not make any compromise in quality, but in practice, and especially when transmitting video over a busy network, the bottleneck is more often in the bandwidth than in the lack of encoding power.

No movement in the camera means a low bitrate, but for example a person appearing in the image causes a burst in the stream. The bitrates and their tendency to vary over time make it difficult to apply economical resource sharing mechanisms to the networks. The RSVP standard [16], which is capable of addressing this problem, has been around for a while now, but hasn t gained much support. IPv6 [38] will offer much better support for routers that work with network-layer PDUs, thus remaining totally ignorant of higher-layer stream control, such as RTP/RTSP, but Ipv6 is still years ahead.

The quality of a digital video of a reasonable frame rate and size distributed over a common network and played back in real time continues to lag far behind the quality we obtain from a moderate analog VCR representing the technology of the 60 s or 70 s.

How can we utilize raw video in adaptation? Several approaches exist, ranging from analysis of independent frames to utilization of frequency-domain coefficients obtained from the encoded form. We begin to address this problem by proposing a generalized mapping of digital video is to the multimedia content model, Figure 4. On the presentation layer, a video object references one or many videos and audio streams, which are constructed from a number of frames and audio blocks, which are in turn further divided into primitives.

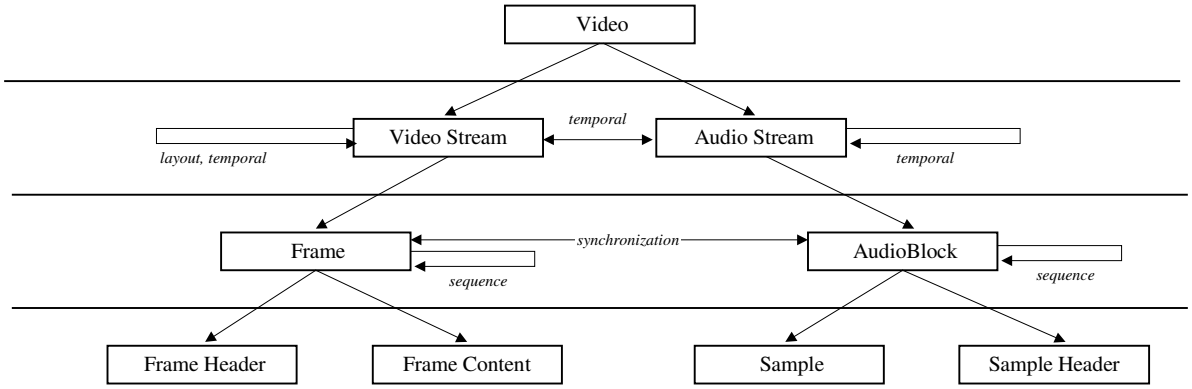

Figure 4. Generalized Mapping of video and Media Content Model.

When mapping to the content model, a video presentation is defined to contain one or more subscripts, e.g. an image stream and an audio stream as illustrated above. The mapping to a specific codec is specified at this layer; some video standards support multiple types. The subscripts relate to each other in

the layout (spatial) and temporal sense; for example, if a presentation is constructed from a multiple video and audio streams, which may start and end at specified times and appear at specific locations in the playback window.

The horizontal axis, defined in the content model as the spatiotemporal axis, represents relationships in time between objects at the same layer. A mechanism often referred to as timestamping is used to synchronize two streams during playback (the rate of the audio should match the frame rate of the video) and are in this case expected to be defined at the object layer. Synchronization refers to matching two media objects on the time axis, whereas the temporal relationships defined at the subscript layer refer to controlling the starting and stopping of streams in relation to others.

Looking at the last layer, the primitive layer, the video objects are divided into header and content blocks. A video frame consists of two inter-related components: a frame header, which specifies constraints such as the length and type of the content block, which is in turn used to carry the actual pixel or frequency domain data of a single frame. One header block can be used to specify multiple content blocks. The audio block is divided in the same way into two segments, sample and header, respectively.

Analysis

We recall from Chapter 2 that the goal of analysis was to support subsequent operations (i.e. scaling and conversion) with a set of constraints and parameters. The SMIL standard, provided it is properly used, offers several mechanisms for selecting between different versions of media prior to transmission. In addition, it supports on-demand adaptation, given that different versions of the media object in question are available or can be created in real time.

Analyzing presentations encoded in other standards, namely QT4, WMT, G2 and MPEG-4 discussed above, is more difficult due to their single, big file approach. These standards are more closely related to the encoding of media primitives and their features that are directly extractable by general analysis are very much dependent on the standard.

Features usable in further adaptation include parameters ranging from basic frame rate and bandwidth requirements (both peak and average) to more advanced algorithms, such as object or scene change detection. The latter class of analysis modules operate very close to the binary encoding of video and are very much dependent on it. Thus for a practical application we would probably use one common format. Which format/encoding is best for this is beyond the scope of this work. Determining the size-related parameters is relatively easy for all standards and selecting a best candidate is quite subjective.

On-demand analysis of video, especially if we need the more advanced, semantic and structural parameters, tends to be very expensive in terms of CPU cycles and memory depending on the algorithm used. The analysis of a SMIL document, either in textual or DOM format, can probably be done in real time, whereas analysis of encoded video often (though not always) requires decompression of video, a factor which makes this solution infeasible for any practical use. From this point of view, it would be easy to claim that space-hungry encodings such as AVI would require much less CPU power to analyze. This, however, is not necessarily always the case, as many algorithms for analysis of video compressed in the frequency domain instead of the time domain have been developed, some of which require only modest processing power.

Scaling

Scaling of the presentation is defined by the adaptation taxonomy to be performed on the object or primitive layer. A SMIL document only references the scaled objects, so for the formats under

consideration, the scaling applies only to encoded video. Several options for scaling digital video exist. In this study, we limit this set to frame-rate reduction, color space quantization, and frame-size reduction.

Frame-rate reduction is the easiest way to limit the width of a video stream, and possibly the worst if quality is central. In the most primitive case, it can be implemented via dropping transmission packets. Color space quantization and frame size reduction trade single frame quality for smoother motion but require decoding/encoding the video and are more difficult to implement. No major difference can be found in the standards covered in this work as regards in their efficiency, as they all provide good tool support for adjusting frame rate versus frame size and quality.

On-demand scaling of video with a method other than dropping the frame rate via merely dropping packets is probably very CPU-intensive. To be performed in real time, it must be restricted only to small-scale (and low- bitrate) applications. Techniques such as stream thinning (discussed later in this document) may do the job, but for optimal quality in terms of frame rate and size, the best solution is probably to create a fixed set of streams optimized for a specific bandwidth rather than attempting to serve every request separately.

Conversion

The motivation for converting video can be either the fact that the client is unable to decode the video format, or that some other format could improve the overall QoS. Once again, the SMIL document only specifies the source of the video and doesn t really care in what format it is, but on the other hand converting a SMIL document into some other format is difficult and is dependent on the encoding of the streams.

Converting video from one encoding format to another often requires complete decoding of the original and therefore, given that hardware capable of doing this in real time is not available, must be done off-line on a limited set of the most popular formats. For the QT, ASF and G2 formats, the conversion is done with tools provided by the vendors of the formats.

If seen in a wider context, conversion of video could also mean transforming it into a static view, using the presentation structure, title, and sophisticated analysis algorithms. Prime candidates for this kind of conversion would be SMIL and MPEG-4, as they already define a good skeleton for a static summary.

In conclusion, all of the streaming presentation standards discussed here provide advanced features and space-efficient features to be used in resource-limited environments. They all offer good quality of service in terms of video quality and fast access, and converting video presentations from one format to another is trivial, as good tools for doing this are already out there.

Thus, adaptation of streaming media should be seen in a wider context, as improvement of the quality of service, with an entirely different meaning from plain frame rate or size, but rather by offering good navigation tools, a descriptive index, and a possibility of selecting elements of interest from the video.

3.2 Static Presentations with Embedded Media

Most content found on the Internet is presented in the more or less standard HTML and is typically static. It is written manually and remains the same for all clients that request it until the author or webmaster alters it, again by hand. The static nature remains when viewed with a browser: once retrieved and rendered, the document remains the same. Usually, these documents and associated elements are retrieved using the HTTP protocol, which will be discussed in Section 4.1 in more detail.

In most cases, static documents are interlinked using the linking facility provided by the HTML language, or in the future, with the more advanced pointers provided by XML [15] [6], which allow defining the link values outside the original presentation.

This section describes the properties of static documents, lists some of the most common formats with examples, and finally concludes with some remarks about automatic analysis, scaling, and conversion of static documents.

Mapping to model
A typical static document is distributed over all layers of the content model. The content of the service index is distributed on the presentation layer as multiple objects, each linked semantically. One presentation maps to one instance of the document. Documents typically create one or more subscripts, for example, one for title, one for scripting element, and one to several for content. Depending on the application, the presentation, and its total size, we can create subscripts, for instance for each page, paragraph, or table. If represented as objects, the content is parsed to multiple objects presented in hierarchic format. Document layout can be expressed as linking between the subscripts (for example, pages) and objects (text sections, paragraphs, columns, or graphics). Finally, the primitives are defined as header-data pairs, separating the formatting from the payload. Links are represented as semantic relationships in the content model.

We begin the discussion of static document formats with document markup languages, Section 3.2.1. Some aspects of digital images and their utilization in adaptation are covered in Section 3.2.2. In Section 3.2.3, we point out some aspects of scripting of the documents. Adaptation of static documents is addressed in general form in Section 3.2.4. Finally, some issues and the status of mobile access to multimedia are outlined in Section 3.2.5.

3.2.1 Document Markup Languages and Style Sheets

As a subset of SGML, HTML has dominated the Web as the de facto markup language and format for hypertext content. Due to competition in the browser market and its shortcomings, it has been used in a way not originally intended, by defining the layout instead of sticking to the content. In addition to proprietary extensions, the use of tables as layout elements has made most of the HTML found on Internet WWW sites very hard to process and index.

In a response to requirements raised by the WWW provider and user community, the W3C has been working on new versions of HTML and stylesheets, CSS (Cascading Style Sheets), for many years now. The stylesheet can be used to define the format outside of the document. This has at least three benefits: First, the document content can be written without defining its format, which makes it easier to produce and update. Second, the bandwidth required for viewing multiple documents having the same stylesheet is reduced since the stylesheet need be transmitted only once. The third benefit is the uniformity of the service which, as with original content, can more easily be modified than would be the case if we defined the format directly in the document. CSS stylesheets can be cascaded, as their name implies, to refine the original layout as necessary.

Currently, the latest recommendation is HTML 4.0. At the same time, the XHTML 1.0 language, actually XML-reformatted HTML, has reached the proposed standard status in the W3C with the accompanying XML-based stylesheet language XSL (standard still under construction), XSLT, and others. [6]

3.2.1.1 XML, XHTML

Given the vast interest in XML and its already wide range of applications, from medical data management to configuration of software components (XML JavaBeans, for instance) the world seems to be heading towards XML also for Web content. Other remarkable XML applications include the RDF initiative, which aims to define a common framework for metadata exchange, and CDF, Microsoft s Channel Definition Format.

XML documents are formatted according to a Document Type Definition (DTD), which defines the schematics of the document. The validity of a document can be checked against its DTD at any time to ensure that it doesn t contain non-defined elements or attributes, that elements appear in the correct order, and so on. The well-formedness (all tags are closed, all attribute values are quoted, etc.) can be checked even without the DTD.

XHTML (XML formatted XHTML) can be viewed on any browser capable of displaying HTML if some simple guidelines are respected.

The XSL stylesheet standard is still on the drawing board, although some partial implementations already exist. With XSL and XSLT, XML documents can be easily converted from one namespace (DTD) to another.

Support for adaptation

For *analysis*, XML (XHTML) documents can be analyzed through a tree-shaped structure with the DOM (Document Object Model) interface. Methods such as search-by attribute or search-by type (tag) can be applied in addition to various tree-traversal methods offered by third-party tools, for example the preorder algorithm from Sun s Java Project X tool. The results of search can be used to create necessary MMCM objects, for example by finding all image objects of a given subscript. With DOM, it is also easy to represent the document in question as MMCM.

The W3C will modularize the XHTML into several namespaces, which can be used in the analysis to determine the features used in the document and to estimate its relative complexity in terms of formatting elements. The proposed DTDs are strict, transitional and frameset.

XML documents can be *converted* with XSLT processors from one DTD to another with relatively simple methods, e.g. from XHTML to WML, but these methods are still application-dependent and require relatively great effort from the implementer. XML document namespaces can be extended to provide assistance for media adaptation algorithms which process XML documents by providing QoS hints, such as explicit definition of importance or relevance of any given element.

Scaling of XML documents to XHTML documents, could be implemented by selectively removing elements, either by removing them altogether or by reducing the number of bounding formatting elements, such as boldface or even a table. Reducing the number of non-printing characters, that is, tags and attributes, would probably lead only to a small saving in space, but additional speedup can be achieved when these reduced documents are viewed on a low-capability terminal, whose HTML rendering speed may not match the heavy (and often malicious) formatting produced by popular GUI HTML editors, such as Microsoft s Outlook or Netscape s Composer.

Potentially greater savings in document size can be accomplished by scaling images by small to medium reduction in document quality, especially if they are viewed on a small, grayscale screen. This is discussed in detail in Section 3.2.2.

3.2.1.2 HTML Dialects

HTML comes in various shapes and forms, and despite standardization efforts by W3C, HTML has been modified with numerous proprietary extensions. Like XML, HTML is a dialect (subset) of SGML. The latest version is 4.01 [9], whereas the most widely deployed version is probably ~3.2.

The motivation for extending HTML came from the content provider community, which has (with help from inferior HTML export filters found in popular page composition tools such as M$ Outlook) forced web site editors to actually create multiple versions of their content and on the other hand, to optimize their content to a subset of popular browsers. Even now, many sites present a logo somewhere on their welcome page stating the preferred browser or optimized for x , although these statements don t necessarily have anything to do with the actual content, as with AOL, but rather arise from industry policy and the browser war.

HTML, originally designed for defining *structure* and leaving the rendering of documents to the client software, has been misused to the ultimate extent; for example, tables are being used as layout grids, and text sections are being replaced with images just to present the content in a non-standard typeface. This results in significant problems when using this content. First, the processing of a document, once created, requires heavy logic which complicates (and sometimes makes impossible) the work of search engines and directories. Second, a document containing merely a few lines of text may require many kilobytes of space, not an issue for fast-LAN workers but certainly for those behind low-bandwidth links, or even overseas from the original site. Third, authors are forced to create multiple versions of their content, often manually. Finally, automatic analysis, conversion, and scaling of a modern HTML document are often very hard, often practically impossible.

Well-formatted HTML can be processed through a DOM tree in the same way as SMIL or XML; actually, DOM level 1 is targeted both for XML 1.0 and HTML 4.0. The formatting of HTML content can be defined with a stylesheet mechanism, namely CSS1 or CSS2, although the quality of current stylesheet-rendering implementations of stylesheets is not even close to perfect. The benefits of using style sheets are:

- As the name CSS (Cascading Style Sheets) indicates, stylesheets can be used in a layered manner; lower layers inherit the formatting defined on upper layers
- Expensive bandwidth can be saved if the site uses the same sheets in several documents; stylesheets need to be transmitted only once
- The content can be separated entirely from the format; this reduces the effort needed from content providers significantly; another benefit would be uniformity of the application
- With software, cleaned up HTML structure can be rendered for different views, such as a DOM tree instead of a flat document structure
- Conversion to other formats is much easier, as with XML; conversion to other formats is more straightforward, although conversion of HTML to another (document) format can often be compromised due to the ubiquituousness of HTML browsers.

Support for Adaptation

HTML documents can be *analyzed* in several ways, for example by building keyword maps, counting links between documents, or indexing on a selected subset of tags. For adaptation, perhaps the most useful ways to analyze HTML documents are as follows:

1. Weight the relative importance of sections, coupled with well-defined stylesheets or explicit tag usage, assuming that, for example H1>H2>H3>SPAN, or adding proprietary values to selected sections. This would require at least some degree of a priori knowledge about document structure and the semantic meaning of each element.

2. Determine the length of the document with and without downloading all embedded components, such as images in bytes to estimate transmission time – a rather straightforward task
3. Determine the length of the document in terms of required display size or printing characters, lines or pages; this probably requires some kind of pseudo-rendering of the document.

HTML offers (at least) the following methods of *scaling* and *conversion:*

1. Strip unnecessary tags, such as excessive formatting, images, and/or links. This, however, many not result in a significant decrease in document length.
2. Fetch the embedded objects and referenced pages a priori for faster access (caching).
3. Create summaries (only selected sections) based on explicit tag priorities or application logic; either present text enclosed in a specified tag set, or limit the number of characters in the document by replacing lengthy sections with summaries/headlines.

3.2.1.3 Mobile HTML Access

W3C has tackled Mobile hypertext addressing with several recommendations, most notably the notes on Compact HTML for Small Information Appliances [36] and HTML 4.0 Mobile Access Guidelines [35]. Being only guidelines instead of specifications, these notes relate only to the production of content and will not be translated into a markup standard in the near future. The equipment manufacturers are heading towards WAP access instead, and the web community hasn't yet woken up to this. Anyway, these notes suggest multiple ways to make HTML presentations more compact. They can thus be considered as conversions of full HTML, and also offer upward compliance for viewing on non-mobile platforms as well.

Using only a limited set of the complete HTML specification often results in better backward compatibility with older or even text-based browsers and in improvements in transmission time and rendering latency.

3.2.2 Digital Images

Transmitting digital images over the Internet is estimated to take up most of the bandwidth, though audio and video come close behind. Images come in various formats, the most popular being JPEG and GIF [34]. Most images are distributed in compressed format, and decompressing tends to be a relatively CPU-intensive operation, although for some formats, such as hierarchically encoded JPEG, a primitive method of reducing resolution by of powers of two can be performed very efficiently. A non-standard extension of HTML defines the *lowsrc*-attribute to image tags to download smaller versions of an image, but it is not commonly used and does not work on all browsers.

Analyzing Images

Before we can scale or convert an image, we must first analyze it and decide, how (if at all) the image should be scaled. We need to identify the type of image, come up with some estimate of how much benefit can be obtained from scaling, and make sure that the quality of the image remains acceptable.

Content-based image analysis and indexing are out of the scope of this survey, as this is a survey of standards. However, we will point out one area that would have an application when reducing the image size is one of our goals: to preserve the quality of the image even when reducing its pixel or color depth domain complexity, as different classes of images tolerate different amounts of scaling in this context.

Roughly, digital images can be divided into three groups when determining the minimum size:

1. Photographs
2. Graphics
3. Mixed

Photographs usually contain much noise and gradients and are often compressed using lossy algorithms such as JPEG. Image size can be reduced by spatial scaling (reducing the resolution of the image) or clipping the unnecessary parts out of it while leaving the area of interest untouched , or simply by increasing the amount of data the encoder is allowed to lose, or in other words, allowing it to degrade the SNR. Photographs can generally be reduced to something in the range of 1/5 to 1/10 of the original size while still preserving (some of their) semantic meaning.

Graphics tend to contain sharp edges, such as text and other shapes, and compressing them with a lossy algorithm usually degrades their quality. Also, reducing the spatial size of a digital image containing graphics often leads to unacceptable quality, as clipping the area of interest might, for example, leave the title of the graph outside the image. Images containing graphics, say an advertisement on a web page, are usually encoded in GIF format, which is generally speaking LZW-compressed raw data.

When processing graphics, we might want to employ OCR software to extract the textual data. In the case of bitmaps distributed over the web, the performance of most OCR systems is not as good as with a black-and-white, high-resolution document, but they could be used together with the original image to at least provide a caption for the image.

Ideally, computer graphics can be transmitted in their original format, which is often wrapped around a vectorized, Bezier-curve representation, but this leads to the same obstacle as with many multimedia standards: the client is probably not able to render the object. A link to the original graphics file could be useful in many cases.

Mixed images contain both computer-generated graphics, such as text, and photographic data. A reliable mechanism to distinguish this class of images from photographs or pure graphics would not be very straightforward to implement. Using OCR or image segmentation might help us provide a caption, but this topic is out of the scope of this paper and could be used as a good title for another survey.

The assumption can be made that graphics contain more high-frequency components than photographs do, as they often contain large, one-color surfaces and only a limited number of colors compared to photographs. This could be a start for an image classification algorithm that analyzes images in the frequency domain or even in the compressed frequency domain, but it would work only for JPEG images. Images encoded in lossless GIF or PNG format could either be transformed to the frequency domain first, or our classification will be based only on the number of colors and their relative frequencies (essentially the same thing). [26]

Athitsos et al. suggested that images can be categorized using the following properties [26]:

1. Graphics are often encoded in GIF format, photographs in JPEG
2. Graphics tend to contain a smaller number of colors than photographs
3. Graphics tend to have multiple peaks in their frequency spectra while photographs have continuous spectra
4. The typical image aspect ratio (width divided by height) tends to be close to 1 with photographs, whereas graphics are often either of fixed sizes (banners) or have one dimension much greater than the other.

If done offline, a combination of these techniques might be adequate, but for on-demand scaling we should select the least CPU-intensive methods that don t require decompressing. Still, as image quality depends on the judgment of the viewer, the option of getting the full version must be included in practical applications. Mixed-type images call for a manual definition of the class, for example adding a parameter to the mark-up tag for image class.

Scaling

To scale a digital image into a smaller size on demand, the following must be ensured:
1. Scaling should not take more time than the expected benefit in transmission time
2. Scaling should not render the quality of image useless
3. The client should be able to view the original image if needed
4. The amount of scaling must not depend on size only; the quality (*semantic* message carried by the image) must be preserved

For the first requirement, formats differ in the relative computational load they create when scaled on demand. Images encoded in the frequency domain tend to be more expensive due to FFT/IFT or DCT/IDCT transformations (e.g. JPEG) than formats that use plain lossless compression algorithms, such as LZW (e.g. PNG, GIF).

For the second requirement, image encoding standards do not differ, but as with requirement 1, lossy compression algorithms add noise to the image, resulting in jagged edges and similar effects, and therefore result in inferior quality especially when applied to an already small amount of data.

To satisfy the third requirement, obviously no encoding has an advantage over any other. Finally, for the fourth requirement, the case is much the same as with requirements 1 and 2, as none of the standards offers any semantic knowledge of the content. When determining an area of interest of the image, such as a face or other shape, no encoding format can be seen to be superior, as most algorithms work with decoded, raw image data.

In most cases, when looking for the smallest image size that is still of acceptable quality, the selection of the encoding ought to depend on the *nature* of the image, be it a photograph, computer-generated graphic, or both. Several methods exist for limiting the encoded size of the image; see Table 5.

Finally, a good, straightforward method of reducing the size of image files referenced from an HTML file could be as simple as re-compressing them with a higher loss ratio and transforming the GIF images into PNG format with little or no effect on quality, [27].

Method	Notes
Reducing spatial resolution (size)	Depending to some extent on the encoding algorithm, reduction in the spatial domain is directly proportional to reduction in encoded size. Performs better on photographs than graphics, as it tends to blur edges and render the overlaid text unreadable
Reducing number of colors	Reduces the encoded size of the image dramatically, e.g. 320 × 200 × 24=1,536,000 bits vs. 320 × 200 × 8=512,000 bits, thus converting from 24bits (16 million colors) to 8bits (256 colors); results in roughly a 67% saving in the bits needed to represent the image data and still offer relatively good quality, especially when using smart color reduction algorithms and dithering. More dramatic reductions, such as to 4 bits or even 2 bits, work better on graphics than on photographs, especially if used in conjunction with dithering and some high-pass filtering to remove background noise.
Selecting an area of interest	Application- and image-dependent. The number of bytes saved in scaling depends on the size of the area of interest compared to the

	original. Extensively used with map-type images; instead of the total image, supplies a high-level view or area map with navigation links to refine/move the map.
Interlacing	Preserves quality quite well, but results in a 50% saving in size. Useful with GIF/PNG images due to in-format methods to do this; less useful with JPEG images. If the even rows are not transmitted, this method also reduces the total latency; otherwise, gives only quicker draft-level transmission.
Progressive encoding	Related to interlaced encoding of GIF and PNG, this method is often used with JPEG images; the image is transmitted in a layered manner where each subsequent layer refines the preceding layer. Every layer carries one bit plane of the image, so an image encoded in 16 layers can be seen with overview quality after 1-2 layers, i.e. roughly 1/8 of the transmission time. If all layers are transmitted and the document contains multiple images, the total latency is not affected.

Table 5. Summary of Image Scaling Methods

3.2.3 Scripting Active Documents

Recently, we have seen a debate about a concept often referred to as Dynamic HTML, although no formal standard for DHTML actually exists, as it has been invented more by advertising agencies than engineers. Originating from simple form-checking applications, current versions of JavaScript (which, despite its name, is not related to the Java programming language), and ECMAScript among others, have evolved to the point where interactive games and applications can easily be implemented and run through a www browser. In this section, we explore how various scripting and programming techniques can be used to enhance the quality of service through adaptation, but stress the point that our focus is on on-demand, thin-client architecture.

The main motivation of moving application logic is to minimize server transactions, for example from an incorrectly filled order form form parsing, error messages, and pre-filled forms can be created at the client s browser. Software solutions such as Microsoft s ActiveX and Java are providing web developers with full-blown sandboxes to create full-scale applications on the browser, but they are often not supported by small-capability devices and often suffer from high startup latencies, security flaws and version conflicts. The pros and cons of ActiveX and Java are, however, beyond the scope of this study.

The benefits of scripting, or more generally utilization of programmatic components embedded in or referenced from a document, are mainly in providing some enhancement to a standard user interface of the browser, checking the user's input or simply generating part of the document, for example with a hit counter, title-bar scroller, or the like. In the analysis phase of the adaptation, scripted (we refer to all documents which embed either scripts or program components as scripted) documents can be used to deliver information about the client to the adaptation engine, presumably located behind a network. This information can range from authentication to capability negotiation and setting runtime preferences, which in turn can be utilized when serving media. After receiving the adapted document, the scripting can be utilized if we want to provide different *views* of data, for example, plotting a graph of the transmitted data, potentially reducing the number of subsequent requests.

The main problems in the utilization of scripts relate to security and version control flaws, for example severe security holes in ActiveX components (everybody remembers the ActiveX component, which could retrieve the PIII hardware ID even if it was disabled from the PC s BIOS) or various Java virtual machine versions.

Mapping to Media Content Model

We locate the scripting elements on the subscript layer and consider them mainly as immutable, external components that are potentially removed. Some standard, general-use components, however, might have applications as alternative user interfaces for presentations even when filtered through adaptation. For example, RealNetworks s G2 player offers a good Java API to create customizable video and audio players comprising methods of controlling the playback and appearance of the presentation.

Support for Adaptation

As scripting is assumed to support client-side interactivity, its application in adaptation is just this: scripting can be utilized to negotiate capabilities and preferences between client and service provider and to define alternative views of presentations. Client-side scripts can be used for navigating large images without a need to transmit the entire document and to minimize client-server interactions using techniques such as form validation, simple calculations, etc.

3.2.4 Adapting Static Documents

We face two challenges when adapting static documents:

1. The quality of service, experienced by the client as document quality ought to be improved and transmission latency ought to be shortened
2. The semantic content of the document must be preserved

Most of the bandwidth used when transmitting text (hypertext) over the network is occupied by the linked binary data, mostly images or simple animations. Obviously, the greatest savings are also there. Tweaking the text data doesn t offer as great benefits in terms of bytes, although improving the signal to noise ratio (which, for static documents, refers to the ratio between amount of formatting and number of characters) can yield some improvement in some applications by reducing the transmission latency, and potentially to an even greater degree, the rendering latency.

If we are not able to compress the textual data directly without losing information and format, we can attempt to provide *alternative views* of the original documents. For example, predictive caching of links to the recently accessed documents can yield substantial reduction in total latency. Another option could be merging multiple documents into a single file to prevent unnecessary client-server interactions, resulting in faster access and less server load. In particular, as will be seen in more detail later in this survey, combining multiple documents can reduce the amount of bandwidth consumed in the transmission of totally redundant data carried in HTTP headers, and the latencies created by TCP/IP socket creation.

For limited capability terminals, careful analysis and optimization of format can make rendering of the document much faster while also saving a little bandwidth; sometimes this is necessary even to make the document viewable, as some proprietary or advanced features of documents are not even supported by all browsers. Another method of reducing transmission and rendering latency is dividing the original document into multiple fragments instead of a single, potentially large file of which only a portion is interesting to the user, but the use of this method will require extensive knowledge about users interests and document semantics in order for it to be of any benefit.

3.2.5 Mobile Multimedia

The properties of digital mobile networks, i.e. low bandwidth, high latency and high (and highly variable, especially when used from a moving vehicle) error rates, have limited the penetration of their use as the primary last-mile connection; instead, they are used only when no other options are available.

Emerging networks, such as the GPRS and especially the UMTS, will provide mobile users with enough bandwidth, at least in highly populated urban areas, for real-time video and real multimedia.

The Mobicom industry has been developing the WAP standard for a while, and the first WAP-capable phones are finally entering the market. WAP has been developed for smart phones with limited displays, and offers basic hypertext and scripting features through the Wireless Markup Language (WML, an XML-compliant mark-up language) and WMLS, Wireless Markup Language Scripting, actually a dialect of JavaScript. Only monochrome images are supported, and the current version (1.1) of the standard lacks a definition of the push feature. The WAP standard also defines WAE, a standard runtime environment, and several protocols suitable for secure and efficient communication over a low-bandwidth network. Possible applications of WAP include, for example, enhancing the interfaces to awkward SMS-based services, directories, and banking services.

Despite the marketing hype around WAP, it has many limitations, not to mention the inferiority of the first client implementations. First, WAP doesn t address streaming data, even audio, in any way. It lacks a good push mechanism and will probably also lack one in the future, as it is by and large constrained by the diversity of underlying wireless carriers for which a universally usable push mechanism would be only a optional feature. In the first version, the push and notification features are used in carrier-dependent, non-WAP methods such as GSM short text messaging (SMS) technology, which in fact was among the proposed wireless carriers as well but will not probably be deployed or even usable due to its extremely low bandwidth (160 characters/message), very high latency (potentially in the range of tens of seconds per message), and typical fixed pay-per-message billing principle.

4 Multimedia Transmission Standards

Along with formatting and description of content, multimedia imposes some special constraints on transmission when compared to traditional data communications. First, the amount of data is enormous for even a short video clip. Second, no one wants to wait for 10 minutes to see a minute of footage; transmission ought to be done in (near) real-time when possible. Third, controlling the flow of the video stream in a public packet network with no guaranteed bandwidth and/or latency requires a separate control channel which, depending on the nature of the application (on-demand or live broadcast), suffers from the same effects as the payload channel, and of course requires some bandwidth in order to operate.

Unreliable transmission methods, such as UDP, used with on-demand broadcasts are subject to packet losses, reordering, duplication, and jitter. Reliable methods such as TCP (used widely in on-demand applications) suffer from the same effects, causing buffer overruns, retransmissions and even slow socket creation procedures due to errors or congestions in public networks.

In most cases, data transmission in a shared network is based on packets; computers send small bursts of information to communicate with each other. Much effort has been placed on developing protocols that allow multiplexing the same medium in an efficient and secure way. A model often referred to in basic data communications books as the OSI (Open Systems Interconnection) model divides the program handling transmission into seven layers, of which five have been used in the well-known Internet Protocol, version 4.

Originally adopted from the Arpanet effort and with some influence from the OSI model, the most widely deployed protocol model is the Internet protocol. It is actually a family of protocols, divided into five layers; see Figure 5.

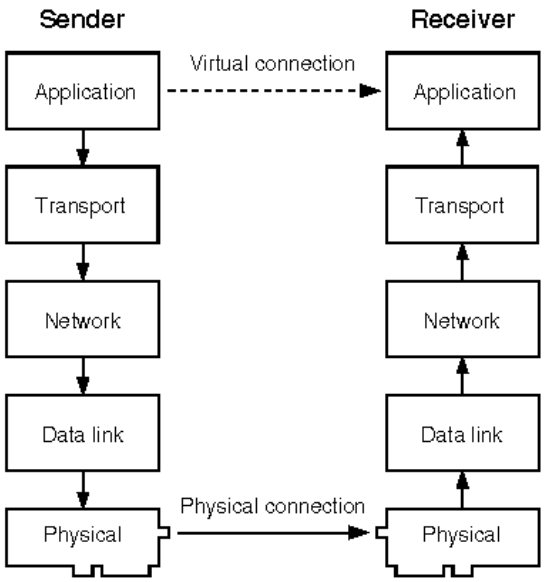

Figure 5. The Internet Protocol v4

The applications enjoy an abstracted view of the network; encoding, reliable transmission, and session handling, among other features, are handled by the lower layers. Layers communicate with their counterparts on the other computer with packets called PDUs or Protocol Data Units. Most, if not all, of the protocols discussed in the following sections are located at the application layer of the Internet model.

In this chapter we cover the most common transmission protocols, and how they can be used to adapt media before and during transmission, in Sections 4.1 and 4.2. The RealNetworks G2 suite is discussed as an example of a commercial, already deployed system in Section 4.3. Finally, Section 4.4 concludes this chapter by discussing the constraints of mobile access to multimedia and how the properties of a mobile network affect the transmission of multimedia.

4.1 HTTP

HTTP, the HyperText Transport Protocol, is the de facto standard application-layer protocol to access the web. Its most recent version, described in RFC 2616 (originally RFC 2068) [7], is 1.1.

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems. It is a generic, stateless, protocol which can be used for many tasks beyond its use for hypertext, such as name servers and distributed object management systems, through extension of its request methods, error codes and headers. A feature of HTTP is the typing and negotiation of data representation, allowing systems to be built independently of the data being transferred. [7]

As this extract from the document suggests, HTTP can be used in a wider context as well to transmit video, or even to let objects (using, for example, Java RMI) communicate with each other. The fact that HTTP is stateless often cripples its application, at least for mobile clients; every time something is accessed through HTTP, a connection is opened, a request sent, a response received, and finally the connection closed, all of which takes time, especially through a high-latency mobile network.

HTTP headers can have sizes of hundreds of bytes, creating (unnecessary) latencies for low-bandwidth channels and affecting the overall (from request to presentation) latency, as much of this data is more or less redundant if using the same server. A technique named HTTP pipelining, introduced with protocol version 1.1, removes the necessity of opening and closing a TCP socket for each request to the same host,

resulting in dramatic performance improvement, yet still transmitting a great amount of redundancy in the headers. However, the support for this feature of HTTP/1.1 still varies, and it will not work if the subsequent requests are not sent to the same server [27].

HTTP offers the following advantages when used to transmit streaming media:

1. HTTP uses the TCP/IP protocol to ensure that all packets are delivered, retransmitting lost or corrupted ones if necessary. [21]
2. HTTP does not attempt to stream in real time. To stream in real time, the bandwidth of the network must be greater than the data rate of the movie (this is true for all the other non-compressing protocols as well). [21]
3. Most firewalls and network configuration schemes will pass HTTP without modification. [21]
4. Most of the major vendors supplying streaming transmission and playback software, including MS WMT4, QuickTime 4 and G2, support HTTP streaming of video, audio, text, and MIDI. [21] [20]

Support for adaptation

1. In the HTTP/1.1 request header, the requester should identify itself with the user-agent request header field, as specified in RFC2616 [7]. This can be used to identify the client s software, and sometimes even the resolution of his display (among other parameters), but this is completely dependent on the client's software implementation. Possible values for the user-agent field are not defined in any RFC and are assigned by software vendors.
2. Utilization of the HTTP Proxy mechanism allows transparent, middle-tier adaptation with no modifications to origin servers or clients
3. The HTTP extension framework allows customization of the protocol. This method is used, for example, with the CC/PP content type negotiation protocol, although it is not on the market yet.

4.2 Real-time Transmission Protocols

This section covers the properties of the family of RTP, RTSP, and RTCP protocols and how their features can be utilized in adaptation. We discuss the RTP protocol in Section 4.2.1 and then continue with RTSP in Section 4.2.2.

4.2.1 RTP

RTP, the Transport Protocol for Real-time applications, is described in RFC1889 [4] by the IETF.

RTP provides end-to-end network transport functions suitable for applications transmitting real-time data, such as audio, video or simulation data, over multicast or unicast network services. RTP does not address resource reservation and does not guarantee quality-of-service for real-time services. The data transport is augmented by a control protocol (RTCP) to allow monitoring of the data delivery in a manner scalable to large multicast networks, and to provide minimal control and identification functionality. RTP and RTCP are designed to be independent of the underlying transport and network layers. The protocol supports the use of RTP-level translators and mixers. [4]

RTP offers the following advantages when used to transmit streaming media [21]:

1. RTP can be used for live transmission and multicast.
2. Real-time streaming allows the user to view long movies or continuous transmissions without having to store more than a few seconds of data locally.
3. Using RTP transmission under RTSP [5] control, a user can skip to any point in a movie on a server without downloading the intervening material.
4. A user can stream a single track over RTP, whereas HTTP streams only whole movies. RTP streams can be incorporated in to a movie using streaming tracks. A streaming track is a track in a QuickTime movie that contains the URL of the streaming content.

5. A QuickTime movie that contains streaming tracks can also include non-streaming tracks whose media exist on the client's computer. This allows a live transmission, or data stored on the Internet, to be incorporated into a movie along with material stored on the client's hard drive or distributed on a CD-ROM.
6. RTP uses the UDP/IP protocol, which doesn't attempt to retransmit lost packets. This allows multicasts as well as live streams, which are both cases where retransmission would not be practical.

In most cases, this will hold for payload types other than QuickTime as well. A number of payload formats have been defined in various RFCs. A complete list of those available as of August 1998 is presented in Table 6. [4]

Number	Title	Date
RFC1889	RTP	January 1996
RFC1890	RTP Profile for Audio and Video Conferences with Minimal Control	January 1996
RFC2029	RTP Payload Format of Sun's CellB Video Encoding	October 1996
RFC2032	RTP Payload Format for H.261 Video Streams	October 1996
RFC2035	RTP Payload Format for JPEG-compressed Video	October 1996
RFC2038	RTP Payload Format for MPEG1/MPEG2 Video	October 1996
RFC2190	RTP Payload Format for H.263 Video Streams	September 1997
RFC2198	RTP Payload for Redundant Audio Data	September 1997
RFC2250	RTP Payload Format for MPEG1/MPEG2 Video	January 1998
RFC2343	RTP Payload Format for Bundled MPEG	May 1998
RFC2429	RTP Payload Format for the 1998 Version of ITU-T Rec. H.263 Video (H.263+)	October 1998
RFC2431	RTP Payload Format for BT.656 Video Encoding	October 1998
RFC2435	RTP Payload Format for JPEG-compressed Video	October 1998
RFC2508	Compressing IP/UDP/RTP Headers for Low-Speed Serial Links	February 1999
RFC2658	RTP Payload Format for PureVoice(tm) Audio	August 1999

Table 6. RTP-Related IETF RFCs

Defined in the same RFC as the RTP, the IETF proposes a control protocol called RTCP to be used with RTP. RTCP's key features are listed below [4]:

1. The primary function is to provide feedback on the quality of the data distribution. This is an integral part of RTP's role as a transport protocol and is related to the flow and congestion control functions of other transport protocols.
2. RTCP carries a persistent transport-level identifier for an RTP source called the canonical name or CNAME. Since the SSRC identifier may change if a conflict is discovered or a program is restarted, receivers require the CNAME to keep track of each participant. Receivers also require the CNAME to associate multiple data streams from a given participant in a set of related RTP sessions, for example to synchronize audio and video.
3. The first two functions require that all participants send RTCP packets; therefore the transmission rate must be controlled in order for RTP to scale up to a large number of participants. By having each participant send its control packets to all the others, each can independently observe the number of participants. This number is used to calculate the rate at which the packets are sent.
4. A fourth, optional function is to convey minimal session control information, for example participant identification to be displayed in the user interface.

Support for on-demand adaptation

1. In conjunction with a separate control channel for RTCP or RTCP (see Section 4.2.2 for details), it is possible to control Quality of Service during transmission
2. RTP is unaware of the lower-layer transport protocol and can be used over a variety of transmission, datalink, or physical layer protocols. RTP can also be used to carry a variety of payload formats potentially, any format. See Table 6 for a list of those defined by the IETF.

4.2.2 RTSP

The acronym RSTP is interpreted as Real-Time Streaming Protocol and is defined in RFC 2326.

The Real-Time Streaming Protocol (RTSP) establishes and controls either a single or several time-synchronized streams of continuous media such as audio and video. It does not typically deliver the continuous streams itself, although interleaving of the continuous media stream with the control stream is possible. In other words, RTSP acts as a "network remote control for multimedia servers. [5]

RSTP offers frame-level accuracy though SMTPE timestamps, which can be used for editing purposes in addition to playback control.

Support for on-demand adaptation

RTSP itself is not used to carry actual multimedia payload but merely controls the transmission over another channel. With RTSP, it is possible to

1. Skip parts of the video with SMTPE time stamps
2. Control the transmission of the stream; in addition to standard play, pause, forward and backward (annotation) features, it can be used to control the bandwidth of the stream
3. Although features such as hand-off, etc., are outside the scope of RTSP, the quality of service can be altered dynamically with RTSP messages. RTSP does not define any mechanisms to determine the performance of the transmission channel.

4.3 RealNetworks G2

The RealSystem G2 supports two types of multicast on an IP network: back-channel and scalable. The back-channel method uses a separate control channel in addition to the actual video feed to every client using either RTSP [5] or PNA (Phoneline Networking Association). The video feed is multicasted using the RealNetworks RDT packet format and protocol. The other method, scalable multicast, uses the data channel using RTP [4] without a separate control channel. [24]

The back-channel method is recommended for a small-scale service where good performance and quality of service is required, whereas scalable multicast is preferred for a large number of users. Back-channel multicast has better reliability and security than scalable multicast; content can be authenticated and lost packets can be recovered by the use of a control channel. Scalable multicast should be used in large-scale real-time applications, whereas back-channel multicast can be used when transmitting stored video at high quality (to a small number of clients). [24]

RealNetworks G2 offers a complete suite of tools for developing multimedia applications, ranging from playback engines to a server application and an on-demand content production suite. Apart from being an active member of W3C and one of the strongest supporters of SMIL, RealNetworks defines a set of proprietary multimedia formats.

RealPlayer G2, at this time available for UNIX, Windows, and Macintosh platforms, is capable of real-time playback of SMIL presentations, video, and audio. The player engine can be accessed from a third party application that is COM-compliant (although, according to RealNetworks, the implementation

doesn t require any of the Microsoft-specific libraries and will be portable to other platforms as well) and from Java Media Framework applications.

RealServer G2 is a real-time media server capable of transmitting media over RTSP, using UDP packets to transmit media packets and a TCP channel to control the transmission. [39]

Support for adaptation
Using SureStream, the RealServer software is capable of adjusting the bitrate of the transmission stream if network condinitions change. Actual testing of this feature requires more tools and resources than were available at the time of writing. The actual performance of the logic that switches between different bit rates during transmission could not be measured either.

RealNetworks claims that SureStream transmissions can also adjust pre-defined bitrate clips with a technique called stream thinning, which seemed actually to be (selective) dropping of payload packets. Measurements of the true performance were also unavailable at the time.

The RealServer G2 suite in general offers a wide range of features for *on-line* (or real-time) adaptation, most notably [39] [24]:
1. The possibility of adding real-time sources, which are encoded in real time to RealAudio and/or RealVideo
2. The possibility of encoding the content of multiple pre-defined bit rates and transmitting it (based on a client-defined rate) at the available bandwidth
3. Partial implementation of the SMIL 1.0 standard and support for other media types, in addition to RealNetwork s proprietary RealVideo, Audio, RealPix, and RealText encodings.

4.4 Mobility Issues in Transmission of Multimedia

The use of streaming media over wireless carriers faces many problems. First, the wireless medium suffers from high BER and is limited in bandwidth. The mobility of the user is always limited to some geographic area due to the lack of a true global standard for mobile data communications. Many efforts, such as Mobile-IP and mobility issues integrated into IPv6 in the protocol layer, and IMT-2000, UMTS and GRPS on the carrier side, are addressing these problems, but solutions are still somewhere in the future.

For the moment, mobile users will have to survive for some time with error-prone narrow bandwidth and limited mobility. The WAP initiative by WAP Forum solves some, mostly user-interface domain issues, such as hypertext and scripting, but doesn t include true multimedia features. Small or medium footprint operating systems, such as EPOC and Windows CE, have already gained some ground and are both equipped with basic multimedia presentation software and some network connectivity. As Microsoft has announced a major rewrite of CE due to complaints from users and manufacturers, mainly concerning the user interface and scalability, we will probably see a stiffening battle between these two players, hopefully resulting in improved quality.

EPOC devices, ranging from smart phones to handheld computers, and Windows-CE based computers, are connected to the fixed network usually with a POTS or GSM modem, or possibly with a Wireless LAN. A common characteristic of these small devices is that they have small screens, slow connections, relatively slow processors, and limited memory, in addition to limited input capabilities.

To summarize the requirements of the mobile domain, they can be compressed into two statements: minimize the number and size of client-server transactions. To meet these requirements, we point out the following aspects that should be considered when adapting media to a mobile environment:
1. Optimize your content; cut off everything that isn t needed

2. Attempt to forecast what the client will do next; attempt to minimize the number of client-server interactions through careful HCI design and structure of service
3. Relieve the client from unnecessary processing; provide media in a format as close as possible to the specific environment, so they can be rendered quickly

5 Application Example

In this chapter, we summarize our discussion about the multimedia content model, presentation and transmission standard with an application example, which involves mixed text, video, audio and graphics which are distributed to the client in real time. We demonstrate the use of the multimedia content model and adaptation taxonomy while using the features from the media and transmission protocol to analyze, scale and convert the multimedia in order to improve the quality of service to all clients in the user pool.

Consider a service incorporating both off-line (on-demand) and live sources of content, ranging from textual content to realtime video. This content is saved to a database and is served to the client via a public network. Our service contains multiple presentations, which are listed in Table 7.

Presentation	Content	Format
Main index	The table of contents for the service	Formatted text, images, and links
Video index	List of videos	Formatted text, images, and links to video
Audio index	List of audio tracks	Formatted text, images, and links to audio
Video broadcast	Broadcasts selected video	Video on-demand or real-time
Audio broadcast	Broadcasts selected audio	Audio real-time or on-demand
Document index	List of documents	Formatted text, images, and links
Document retrieval	Retrieves selected document.	Formatted text, images, and links
Preferences	Personalization of settings	Formatted text, images, and links

Table 7. Presentations for the Application Example

Our user pool constitutes of two mobile workers, a home user and a office clerk, each having a different connection and equipment. In Table 8 below, we define properties of both the terminal and network, along with personal preferences for each worker.

User	Terminal	Network	Preferences
Mobile Worker 1	Laptop	GSM 9.6k /WLAN 2M	Max. size of static presentation 10 kB
Mobile Worker 2	EPOC PDA	GSM 9.6k	Max. size of static presentation 5 kB, no images
Home User	Old desktop	POTS 56 k	Max. size of static presentation 100 kB
Office Clerk	Fast desktop	LAN 10 M	No audio

Table 8. User pool for the Application Example

In this example, we propose a system that is capable of adapting the original presentations according to the preferences and properties of these users. The adaptation method is layer- and object type- as well as user-dependent. We begin the discussion by defining mappings between object type and scaling, conversion and analysis, i.e. adaptation modules; see Table 9.

Object Type	Analysis	Scaling	Conversion
SMIL Presentation	Document type (DTD) Location	Select streams Change starting point	Static summary

	Transmission size		
HTML Presentation	Document type (DTD) Location Transmission size	Fragmenting Scale embedded objects Reduce formatting	Text only
SMIL Header	Version, author, title, keywords, duration, version, layout regions	-	Convert to HTML header
SMIL Body	Number and types of <par> and <seq> elements Layout areas and sizes	Remove/alter <par>, <seq> and <stream> elements	Convert to HTML body
HTML Header	Version Title Metadata (author, etc.) Stylesheet usage		
HTML Body	Length (bytes, printing characters) Number of linked images Number of blocks (div/span elements) Formatting complexity	Reduced tagset-HTML Text only Divide into multiple pieces	Summary
SMIL Video Streams	Duration Required bandwidth Encoding Frame rate, size *	Select another bitrate Reduce frame rate Reduce frame size	Keyframes Slideshow Title frame
SMIL Audio Streams	Duration Required bandwidth Encoding	Select another bitrate	Speech to text*
SMIL Text Streams	Duration	-	Translate* Convert to single text
Images	Pixel size Size Encoding Layout position Location Caption	Reduce pixel size Reduce color depth Select area of interest*	Exchange encoding type Caption only
HTML Text	Length	Strip formatting	Translate to another language
Encoded Video Stream Frame	Size, position*	Reduce frame size	Provide as still shot
Encoded Audio Stream Sample	Position, frequency spectrum peak*	Reduce sampling frequency, requantisize *	A marker to video (GOL!)
HTML Text Node	Formatting	Remove formatting	Plain text

** Option, not reality*

Table 9. Mapping of Object Types and Adaptation Modules

In the next sections, we will walk through selected presentations listed in Table 8 and serve them to the users using both unadapted and adapted forms of the presentations, constructed from the methods listed above.

5.1 Adapting Static Document Main Index

From the discussion in Chapter 2, a static document can be represented in the multimedia content model by analyzing its structure of external, linked elements. Furthermore, when applied together with an adaptation taxonomy, or more precisely in this example with a combination of those listed in Table 8, we define the method by which the proposed system should adapt multimedia according to the client s network and terminal environment and preferences.

First, we map the Main Index Document to the content model as depicted in Figure 6.

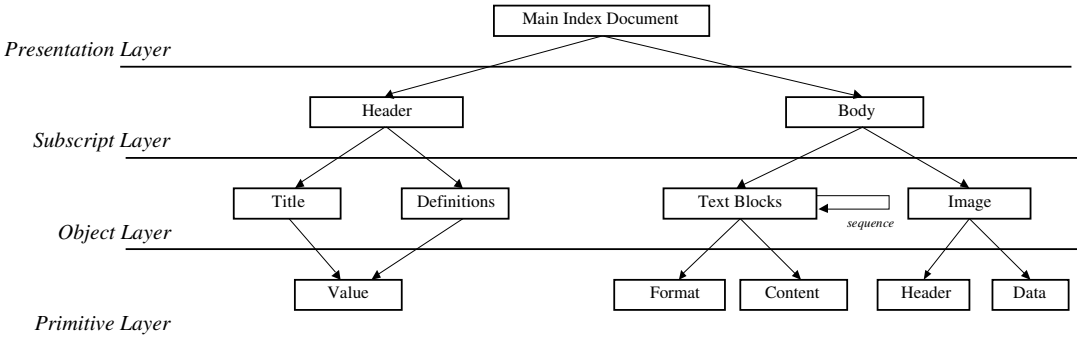

Figure 6. Content Model for Main Index Presentation

Basically, the main index is constructed of two parts, body and header, the first of which mainly describes the common properties of the document and the latter containing and referencing the actual content. The body subscript is in turn broken into interlinked text block and image objects, which are constructed of format, content, image header, and image data primitives. A more specific definition of the content model presentation of the *Main Index Document* is provided in Table 10.

Layer	Object	Defines	Relationships
Presentation	Document	Document type	Semantic: Other documents Children: Body and header subscripts
Subscript	Header	Document header	Child: Title Child: Definitions
	Body	The structure and properties of document content	Object: Text Blocks (2) Object: Image
Object	Title	The document s title	Child: Value string1
	Definitions	Keywords and other metadata	Child: Value string2
	Text Block 1	First block	Child: Body formatting attributes Child: Formatting1 Layout: Next object (Image) Layout: Next text block (Text Block 2)
	Image	The location, size and other attributes of the title image	Children: Header: location, format, parameter Primitive: Image file
	Text Block 2		Primitive: Formatting2 Primitive: Text String2

Primitive	Value String 1	The text of text block 1	
	Value String 2	The text of text block 2	
	Formatting1	The formatting of text block 1	
	Formatting2	The text of text block 1	
	Text String2	The text of text block 1	

Table 10. Object Definition Table for the Main Index Document Presentation

The Main Index document is located on the presentation layer whereas the two subscripts, the header and body respectively, are located on the subscript layer. On the object layer, we have four classes of objects: title, definitions (such as meta-tags for keywords, author, copyright, document DTD, language, etc.), text blocks and images. Finally, the primitive layer carries the actual content; for header-subscript objects this means simply the values of tags/attributes, such as the text for the title. In the body subscript, the objects are composed of two primitives, format and content. The í mage object is divided into two parts as well: the header defines the format while the data block contains the encoded image data. The analysis of the document actually iterates to the object level; it is used to determine the total length of the document in bytes and in (formatted) characters.

For comparison, a DOM tree created from the same document is presented in Figure 7. The DOM model can be used as underlying interface to data while avoiding the tedious task of dealing with various character encodings and painful string parsing. The DOM model can also be used for creating the content model representation, and using it as a uniform interface for the adaptation components for accessing the content and structure of the document.

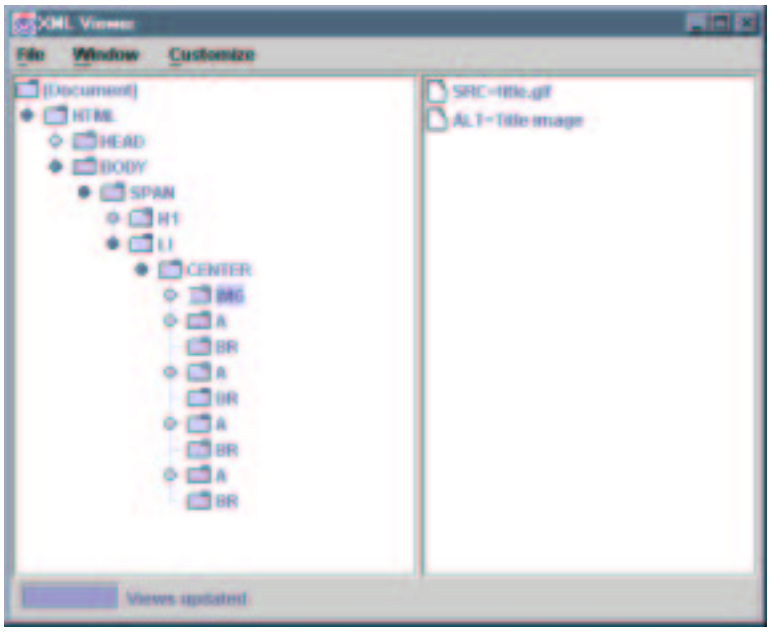

Figure 7. DOM Tree for the Main Index Presentation

Adaptation for our user pool is, in fact, quite straightforward, as the main index is small in size and detail; the document can be transmitted in its original form to the Home and Desktop users.

We have a 20 kB image linked to the body subscript of the document. The user with an EPOC device has specified in his preferences that he doesn t want any images (initially), so we simply replace the <img>

tag with Title Image-tag to replace the image text but still leave the option of viewing the image if required.

Recalling that an EPOC device usually has a small-size display, we still want to limit the number of characters or lines in the document. For this we have to choose from (at least) the following options: a) divide the document into multiple parts, b) attempt to shrink the document by removing most of the markup, or c) convert the document into a summary presentation and provide a link to the original document. In the case of the Main Index, containing only a few lines of text, none of these options seems reasonable, but for longer documents we would certainly have to use some of them.

The choice of what document size reduction method to use depends heavily on the type of the document, whether it is an index, article, list, etc. Removing unnecessary or excessive formatting (option b) and fragmenting the document (a) would result in faster rendering once transmitted to the client and probably would suit most of the different document types. Option c) should be used for lengthy, text-intensive documents, and preferably for building summaries of multiple documents. Such summaries are widely used with search engines.

In order to preserve semantic content, we have to prioritize the elements in some way before we can start removing them. One option could be, for example, weighting headlines and links over body text and pure format elements such as divisors.

Going back to the Multimedia Library application, we recall that the mobile worker s preferences were that static document size should not exceed 10 kB (i.e. some 10-20 seconds in transmission time). The image object is fed into an image scaler which reduces the size of the image to half, limits the number of colors and saves it in PNG format, resulting in a 70% reduction in encoded size. Then, at the object layer, the value of the <IMG src> attribute is replaced with reference to the scaled image and for user convenience, a link to the original image is added to the reduced-resolution image. In this case, this was just enough to meet the 10kB requirement. For a longer document with multiple images we might have been forced to remove at least some of them, or alternatively to break the document into multiple fragments, as discussed earlier in Section 3.2.4.

Despite the relative simplicity of this example, several benefits obtained from the layered approach were demonstrated: First, the client was able to define how long he wants to wait for a presentation. Second, the software performing the adaptation (image scaling, tag exchange) remained totally independent of the actual content of the requested presentation. Third, the quality of service for the mobile worker was increased due to reduced latency in getting the presentation.

5.2 Adapting Streaming Document Video Broadcast

In this example, we define the mapping of a presentation containing streaming media elements and the multimedia content model, and demonstrate how this relationship can be utilized when adapting these presentations to the constraints of our user pool.

Consider having a multimedia document containing real-time video from a live source, where both video and audio are supplied with textual overlay. We depict this presentation on our content model in Figure 8.

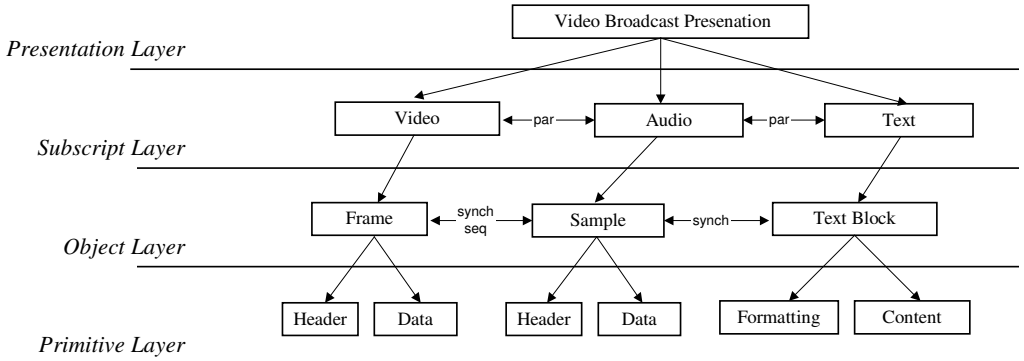

Figure 8. Content Model for Video Broadcast Presentation

As we did for the Main Index Document presentation, we define a similar table defining the objects and their associations to other objects in Table 11 for the Video Broadcast presentation, which is originally defined in a Synchronized Multimedia Integration Language (SMIL) document, consisting of a video feed, an audio track and a text stream providing textual, time-sensitive captions for the video.

Layer	Object	Defines	Relationships
Presenta-tion	Video Broadcast	Document type, title, author, copyrights, layout regions	Subscript: Video Subscript: Audio Subscript: Text Semantic: Multimedia Library index page
Subscript	Video	Location of the source, encoding format, duration, start and stop timestamps	Layout:Region id Temporal: Parallel with audio and text Children: A sequence of frames
	Audio		Temporal: Parallel with video and text Children: A sequence of audio samples
	Text		Layout:Region id Temporal: Parallel with video and audio Children: A sequence of text blocks
Object	Frame n	A single frame of the video feed	Sequence: Next frame
	Sample n	A single sample of the audio feed	Sequence: Next sample
	Text Block n	A single text block visible at the time	Primitive: Body formatting string Primitive: Format1 Primitive: Text String1 Temporal: Start and end times
Primitive	Content n	Content string of Text Block n	-
	Format n	Formatting element of Text Block n	-

Table 11. Object Definition Table for Video Broadcast Presentation

In the diagram, we have divided the presentation into four layers. The video broadcast presentation is defined at the presentation layer, to be more precise, defining the title, format, author and other common properties of the presentation. The presentation breaks into three parallel subscripts; video, audio and (streaming) text. The parallelism of these three subscripts refers to their temporal relationships; they are merely represented at the same time. Actual synchronization of these elements is performed on the object

layer, where we have objects for individual video frames, audio samples and text blocks. Finally, the primitive layer contains the primitives, as in the previous example.

First, we discuss the basic retrieval of this document by the office clerk. As the clerk didn t want audio (perhaps for the sake of his co-workers), the adaptation process first removes the audio track from the presentation by merely discarding it from the model and accordingly from the SMIL script. As the audio track is defined as plain speech, and provided it contains mostly speech, we can execute speech-to-text conversion of the subscript and add the extracted captions to the text substream. Another use for the audio stream would be detecting the peaks in its amplitude and adding navigation links to the corresponding locations in the video.

As speech recognition is language-specific (and often quite unreliable), we would do this only if the audio subscript defined the quality and language of the speech track. The rest of the presentation is provided in its original form.

The home user hasn t specified any preferences for streaming presentations, so we proceed in an automatic way, guided by the knowledge we have obtained from the user: he has a maximum bandwidth of 56 kbps (protocol overhead reduces this even further) and is running a old desktop computer. Furthermore, from the request we discover that he doesn t have the necessary software to play mixed audio, video and text. Thus, we must select if we present the text stream as a serialized chunk, or we can overlay the text onto the video frames by rendering the text to a bitmap, or ultimately discard it totally. As the 56 kbps maximum limits the video frame size to 160 × 120 or so anyway, we decide to discard the text totally and stick with the video. The pre-generated videostream optimized for a 56 k channel is transmitted upon the client's request.

Mobile Worker 1 was equipped with a Laptop computer capable of playing back all the content contained in the Video Broadcast presentation, and thus we need to adapt it based on the properties of the user s network, which actually divides this case into two. A WLAN connection is considered to be able to carry all the data, and the presentation is transmitted in its original form. In the other case, when Mobile Worker 1 is on the road, we are serving the content over a GSM connection, having a maximum throughput of 9.6 kbps, the effective bandwidth available for payload being somewhat less. This restriction renders the transmission of video infeasible, as the frame rate and quality would be far below the acceptable level. Given that we have an on-demand source, i.e., pre-generated content instead of live footage, we analyze the footage with a feature extractor which is capable of measuring the amount of movement between frames and finding amplitude peaks from the audio track. By adjusting the extractor s parameters, we can extract the frames that have the most transformations and/or are near an audio track amplitude peak, and mark them as keyframes.

Mapping to our content model, we actually convert the presentation at the subscript layer; streaming subscripts are replaced with static ones, which are composed of keyframe objects, captions and extracts from the original text stream, and combine them in a flat HTML file, describing the timeline and potentially significant events in the clip. This presentation is, finally, scaled to fit into the 10 kB limit specified by the user by making the keyframes small enough, and in the case of a lengthy original, dividing the summary into multiple pieces or by adjusting the number of keyframes.

For Mobile User 2, we follow the same path as for User 1, but deal with the static presentation as we did in the previous example, resulting in a presentation containing merely a textual summary of the video, in this case the title and captions. If needed (and provided the user has the encoding software and sufficient storage base to buffer the entire clip), he can access the plain video with a link inserted into the summary document.

6 Summary and Conclusions

In this study we have reviewed several multimedia presentation and transmission standards and techniques, related them to the multimedia content model, and discussed how they can be used for multimedia adaptation.

Of the four major streaming formats covered in this study, SMIL seems to be the leader due to its best abstraction of media objects, and of course to the openness and wide acceptance of the W3C-authored standard compared to binary formats, such as QT4, MS WMT4 and G2. Although they are not directly comparable due their different approaches, we are eager to promote a textual, abstract presentation of media instead of embedded, binary coding. In terms of performance, the binary-related formats didn t differ greatly and the selection among them is quite subjective. Their role in multimedia adaptation, if viewed from analysis to scaling and conversion, will probably be as a final payload.

Section 3.2 described standards and formats related to static documents instead of streaming media, although this boundary is somewhat vague. The XML effort by W3C is rapidly gaining ground and is now supported by major browser vendors with a reformulation of HTML XHTML 1.0. Being backward-compatible and enjoying ever-increasing tool and client support, not to mention the hype and drive towards it, XHTML will probably get a large market share of static web content; it also offers excellent adaptation support through DOM and XML features.

Scripting from the point of view of adaptation was discussed in Section 3.2.3 in brief. Adding functionality or interactivity can be accomplished through server-side logic with well-proven techniques such as CGI or Servlets, or on the client side with JavaScript, Applets or other scripting languages. This concept refers mostly to HCI techniques instead of media, and thus no direct conclusion is drawn as to whether some scripting/programming language would offer the best support for adaptation, though Sun s JMF effort and excellent Java support from RealNetwork s G2 and Apple s QT4 would suggest Java as a solution for extending/customizing clients user interfaces.

Selection of video and audio codecs along with a superior image encoding format depends heavily on the application in question; some codecs/formats perform better than others with given resources. This is true for variable-bandwidth encodings, such as MPEG-4 and RealVideo, although MPEG-4 offers many more options for implementing the variation in the required bandwidth, in addition to the frame rate and size options offered by RealVideo.

Transmission protocols offer support for adaptation mainly for on-line applications. This study covered the most common multimedia transmission protocols, namely HTTP, RTP/RTCP, and RTSP with the RealNetworks G2 transmission solution.

In the last chapter we outlined an example system that serves multimedia documents to users with different equipment and personal preferences. Based on this, we suggest that dramatic savings in space can be achieved with only limited information about the semantics of the presentation, for example by relying solely on commercial solutions such as G2 for stream switching and simple image transcoding. These methods are still very fixed in nature and haven t yet matured to a level that can keep bandwidth- and device capability-limited users satisfied.

If more advanced, customizable media-aware methods such as automatic video summary generation or selective scaling of textual content are desired, we still have much work to do on analyzing the content of both static and streaming presentations in order to create a system capable of doing the tasks described in the examples.

This is a study of standards rather than of a specific research problem, so reaching a single, concrete conclusion was not among the original goals of this work. Instead, we conclude with a list of four key findings and future trends that were ranked as most relevant to the scope and aim of this document.

1. Current multimedia presentation standards are centered around specific platforms and binary encodings rather than being self-describing in terms of content semantics. Building an automated system around them will probably require a great degree of application logic, some restrictions on how data must be submitted, and explicit definitions of user-specific constraints.
2. Adaptation of text documents, if taken beyond image scaling, may prove to be technically very complex due to the diversity of the formats used, and due to increasing support for stylesheets, efficient image encodings and better browsers. Proposed methods such as the removal of formatting elements and document fragmentation will require explicit ranking of the importance of content elements in order to preserve semantic content and justify the effort. Also, the implementation of the system must be such that the benefit gained from reduction in size is not lost by poor content quality and additional initial latency.
3. Using streaming components over wireless links will not be of much practical value until their bandwidth is increased by an order of magnitude. Meanwhile, the most feasible way to provide access to video is by creating summaries, which describe the timeline and significant events discovered by low-level analysis of the video and audio data.
4. Adaptation of video on the stream level will probably be addressed by providing several streams of different size from which the user selects, rather than adjusting the streams for every user. Creation of near-real-time summaries will require a set of reliable, fast and general video and audio analysis tools, some of which already exist in academic laboratories, but are yet to be merged into a single, working end-user application.

Acknowledgements

Many thanks to Dr. David Doermann of the University of Maryland for his support and comments on this work. Thanks also to the Princess project consortium for financing my visit to the beautiful campus of the UMD.

References

[1] Synchronized Multimedia Integration Language (SMIL) 1.0 Specification, World Wide Web Consortium (W3C) 1998. http://www.w3.org/TR/REC-smil/. Last visited on 31.8.1999.

[2] Synchronized Multimedia Integration Language (SMIL) Boston Specification , W3C Working Draft 3-August-1999. http://www.w3.org/1999/08/WD-smil-boston-19990803/. Last visited on 9.9.1999.

[3] Document Object Model (DOM) Level 1 Specification, Version 1.0. W3C Recommendation, 1998. http://www.w3.org/TR/REC-DOM-Level-1/. Last visited on 31.8.1999.

[4] RTP:A Transport Protocol for Real-Time Applications, RFC 1889, IETF 1996. http://www.ietf.org/rfc/rfc1889.txt. Last visited on 31.8.1999.

[5] Real Time Streaming Protocol (RTSP), RFC2326, IETF 1998. http://www.ietf.org/rfc/rfc2326.txt.

[6] XML Study, J. Korva and M.Metso, University of Oulu and VTT Electronics, 1999. Available for download at http://ee.oulu.fi:4443.

[7] Hypertext Tranfer Protocol -- HTTP/1.1, RFC2616, W3C 1999. http://www.ietf.org/rfc/rfc2616.txt. Last visited on 1.8.1999.

[8] Hypertext Transfer Protocol -- HTTP/1.0, RFC 1945, May 1996. http://www.ietf.org/rfc/rfc1945.txt.

[9] HTML 4.01 Specification, W3C, http://www.w3.org/TR/html40/. Last visited on 7.9.1999.

[10] XHTML 1.0: The Extensible HyperText Markup Language, Reformulation of HTML 4.0 in XML 1.0, W3C Proposed Recommendation 24 August 1999. http://www.w3.org/TR/xhtml1/. Last visited on 7.9.1999.

[11] Cascading Style Sheets, level 1, W3C Recommendation 17 Dec 1996, revised 11 Jan 1999. http://www.w3.org/TR/REC-CSS1. Last visited on 8.9.1999.

[12] Cascading Style Sheets, level 2 CSS2 Specification, W3C Recommendation 12-May-1998. http://www.w3.org/TR/REC-CSS2/. Last visited on 8.9.1999.

[13] Short- and Long-Term Goals for the HTTP-NG Project, W3C Working Draft 27-March-1998. http://www.w3.org/TR/1998/WD-HTTP-NG-goals. Last visited on 8.9.1999.

[14] Data and Content Based Adaptation, D.Doermann, 1998. Available for download at http://ee.oulu.fi:4443.

[15] Extensible Markup Language (XML) 1.0 Recommendation, World Wide Web Consortium, 1998. http://www.w3.org/TR/REC-xml. Last visited on 1.8.1999.

[16] Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification, RFC2205, IETF 1997. http://www.ietf.org/rfc/rfc2205.txt.

[17] Motion Picture Experts Group, www.mpeg.org. Last visited on 1.8.1999.

[18] Macromedia Web Site, www.macromedia.com. Last visited on 1.8.1999.

[19] Apple QuickTime specifications, http://www.apple.com/quicktime/specifications.html. Last visited on 1.8.1999.

[20] RealVideo Technical White Paper, Real Networks, Inc., 1997. http://www.real.com/devzone/library/whitepapers/overview.html. Last visited on 1.8.1999.

[21] Streaming QuickTime Over RTP or HTTP, Apple Computer, 1998. http://developer.apple.com/techpubs/quicktime/qtdevdocs/REF/Streaming.5.htm. Last visited on 1.8.1999.

[22] Metso M., Koivisto A. and Sauvola J. (1998) Multimedia Adaptation for Dynamic Environments. Proc. Multimedia Signal Processing (MMSP), Los Angeles, CA, USA.

[23] Metso M, Koivisto A. and Sauvola J (1999). Content Model for Mobile Adaptation of Multimedia Information. Proc. 3rd IEEE Workshop on Multimedia Signal Processing, Copenhagen, Denmark, accepted.

[24] White Paper: IP Multicast in RealSystem G2, V.Thomas, RealNetworks Inc., 1998.

[25] An Introduction to MPEG Video Compression, John Viseman, 1997. http://members.aol.com/symbandgrl/. Last visited on 3.9.1999.

[26] Distinguishing Photographs and Graphics on the World Wide Web, V.Athitsos & M. Swain, IEEE Workshop on Content-Based Access of Image and Video Libraries, June 1997. (Can be downloaded from http://www.cs.bu.edu/students/grads/athitsos/.)

[27] Network Performance Effects of HTTP/1.1, CSS1, and PNG, H. Nielsen et al., W3C 1997. http://www.w3.org/Protocols/HTTP/Performance/Pipeline. Last visited on 4.8.1999.

[28] http://webreview.com/wr/pub/guides/style/mastergrid.html.

[29] Cascading Style Sheets, level 1, W3C Recommendation 17 Dec. 1996, revised 11 Jan. 1999. http://www.w3.org/TR/REC-CSS1. Last visited on 3.9.1999.

[30] Oracle Project Panama Project Whitepaper, http://www.oracle.com/mobile/panama/panamawp.htm. Last visited on 3.9.1999.

[31] Symbian Ltd. Homepage, http://www.symbian.com.

[32] WAP Forum Homepage, http://www.wapforum.org.

[33] Adapting to Network and Client Variabilty via On-Demand Dynamic Distillation, A. Fox et al., Proc. Seventh Intl. Conf. on Arch. Support for Prog. Lang. and Oper. Sys. (ASPLOS-VII), 1996, Cambridge, MA, USA.

[34] Reducing WWW Latency and Bandwidth Requirements by Real-Time Distillation, A.Fox et al., Fifth International World Wide Web Conference, 1996, Paris, France.

[35] HTML 4.0 Guidelines for Mobile Access, W3C Note, 15 March 1999. http://www.w3.org/TR/NOTE-html40-mobile/. Last visited on 21.9.1999.

[36] Compact HTML for Small Information Appliances, W3C NOTE, 09-Feb. 1998. http://www.w3.org/TR/1998/NOTE-compactHTML-19980209/. Last visited on 21.9.1999.

[37] Overview of the MPEG-4 Standard, R. Koenen. ISO/IEC JTC1/SC29/WG11 N2725 March 1999, Seoul, South Korea. http://drogo.cselt.it/mpeg/standards/mpeg-4/mpeg-4.htm. Last visited on 23.9.1999.

[38] Internet Protocol, version 6, Specification, S.Deering et al., 1998. http://www.ietf.org/rfc/rfc2460.txt.

[39] RealNetworks, Inc. http://www.real.com.

[40] D. DeMenthon, D.S. Doermann, and V. Kobla. Video Summarization by Curve Simplification. Proc. ACM - Multimedia 98, Bristol, England, pages 211 218, 1998 (PS).

[41] V. Kobla, D.S. Doermann, and C. Faloutsos. Developing High-Level Representations of Video Clips using VideoTrails. Proc. SPIE Conf. on Storage and Retrieval for Image and Video Databases VI, pages 81 92, 1998.