Separation of Concerns for Multimedia
Publication in Heterogeneous Environments

Master of Science Thesis

Final Version 1.1

Stijn Decneut
Student No. 837508618

13th August 2004
Open Universiteit Nederland
Technische wetenschappen
Graduation Committee

Chairman: prof. dr. ir. Jack Gerrissen (Open Universiteit)
E-mail: Jack.Gerrissen@ou.nl

Secretary: dr. ir. Frans Mofers (Open Universiteit)
E-mail: Frans.Mofers@ou.nl

Coach: dr. ir. Frans Mofers (Open Universiteit)
E-mail: Frans.Mofers@ou.nl

Coach: dr. ir. Steven Van Assche (VRT)
E-mail: Steven.VanAssche@vrt.be
Contents

Gratuation Committee i

Contents ii

List of Figures vi

Preface vii

Summary viii

Summary (Dutch) ix

1 Introduction 1

2 Research Context 4

2.1 Recent Evolutions in Information Distribution 4

2.1.1 Client-Server Systems evolve to N-Tier Systems 5

2.1.2 Multimedia is Becoming a Mobile Experience 5

2.1.3 Software Frameworks are Designed Evermore Loosely Coupled 5

2.2 Approaches to Multimedia Publication 5

2.2.1 Device-Specific Authoring 6

2.2.2 Multi-Device Authoring 7

2.2.3 Automatic Re-Authoring 8

2.3 Distribution of Media in an On-Line Context 9

2.3.1 A Multimedia Content Provider’s Viewpoint 9

2.3.2 A Multimedia Content Consumer’s Viewpoint 10

2.3.3 A Multimedia Content Distributor’s Viewpoint 11
3 Problem Definition

3.1 Multimedia and Distributed Information Systems: A Clash of Cultures? .......................... 13

3.2 Automatic Re-Authoring with Media Services ......................................................... 15
   3.2.1 Location of the Media Services ................................................................. 15
   3.2.2 Case Study: Adaptation of Player-specific Presentations to Other Players 16
   3.2.3 Case Study: Adapting the Format of Multimedia Exchanged between Mobile Phones. ......................................................... 17

3.3 Key Challenges and Contribution ................................................................. 18
   3.3.1 A Service-Oriented Architecture for Multimedia Adaptation ....................... 19
   3.3.2 Broker Services ......................................................................................... 19
   3.3.3 Web Services that can handle Media Transmoding ........................................ 19
   3.3.4 Transport of Multimedia Data from and to Web Services ............................. 20

4 Related Technologies ................................................................. 21
   4.1 Standards ........................................................................................................ 21
      4.1.1 ISO/IEC JTC1/SC29/WG11 – MPEG ....................................................... 21
      4.1.2 W3C  ........................................................................................................ 24
   4.2 Industry Initiatives ............................................................................................... 27
      4.2.1 Apache Web Services Project ..................................................................... 27
      4.2.2 Microsoft .NET ......................................................................................... 28
      4.2.3 Web Services Interoperability Organization ............................................... 31
   4.3 Related Research ................................................................................................. 33

5 A Service-Oriented Architecture for Multimedia ......................................................... 35
   5.1 Architecture ........................................................................................................ 36
   5.2 Extending Web Services for Streaming Multimedia ............................................. 38
      5.2.1 Transport of Multimedia Data ...................................................................... 38
      5.2.2 Lifetime Management ................................................................................. 40
   5.3 Architecture Components ................................................................................. 41
      5.3.1 Download Manager Service ......................................................................... 41
      5.3.2 Home Gateway ............................................................................................ 41
      5.3.3 Transmoder Service ..................................................................................... 43
# Table of Contents

5.3.4 Broker Service .................................................. 46
5.4 Case Studies ....................................................... 46
  5.4.1 Viewing SMIL presentations with a Web Browser .......... 46
  5.4.2 Generating SMIL presentations ............................... 52

6 Overview of our Contribution and Rationale .................. 58
  6.1 The Concept of Multimedia Adaptation with Web Services .... 58
  6.2 Unique Contributions ........................................... 59
    6.2.1 Streaming SOAP over DIME ............................... 59
    6.2.2 The Use of DIME for ‘Enriched’ Media Transport (Messaging) .... 59
    6.2.3 The Use of DIME for ‘Enriched’ Media Transport (Streaming) .... 60
    6.2.4 Invocation Mechanisms for Web Services .................. 61
    6.2.5 Polymorphism and Overloading with Web Services .......... 61
  6.3 Comparison with Related Research ............................ 61

7 Conclusion ......................................................... 63

8 Future Work ........................................................ 64
  8.1 The Limitations of DIME ....................................... 64
  8.2 Real-Time Streaming ............................................ 64
  8.3 Adaptation of Presentations .................................... 65
  8.4 Integration of Algorithms ...................................... 65

Bibliography .......................................................... 69

A List of Terms and Standards ..................................... 70

B Paper Published at ICWS2004 ..................................... 73
List of Figures

2.1  Device-specific authoring in the broadcast world: media data that is transmitted over a common carrier can be consumed on a limited variety of devices when encoded according to a common-denominator standard.................................................. 6

2.2  When distributing media in an on-line context, an unpredictable path lies ahead................................................................. 10

2.3  When receiving media in an on-line context, three main parties are involved: the Consumers (with various devices), the Media Providers and the Service Providers................................................................. 11

3.1  Problems occur when exchanging multimedia data over the Internet, as numerous different players and PC-configurations are in use................................................................. 16

3.2  Even within a particular media player scenario, different screen resolutions and bandwidths are possible, as illustrated by this screenshot from a web page................................................................. 17

3.3  Problems occur when exchanging multimedia data between heterogeneous devices, even when using recently standardized multimedia-exchange formats like MMS................................................................. 18

4.1  Overview and organization of Digital Item Adaptation tools................................................................. 24

4.2  Dime records form a segmented message................................................................. 29

4.3  The header fields in a DIME record................................................................. 30

4.4  A SOAP message in DIME references to an attachment using its message ID................................................................. 31

4.5  Processing a SOAP message with WS-attachments in .NET................................................................. 32

5.1  The web services at the proxy server adapt the Content Creator’s multimedia and place it on the consumer’s Home Gateway................................................................. 35

5.2  A more detailed look at the services (and the communication in between) that adapt the Content Creator’s multimedia to the capabilities of the Client................................................................. 37

5.3  A SOAP message with a png image attached consists of two DIME records................................................................. 39

5.4  A chunked DIME stream that contains a SOAP message and an MPEG-1 video stream................................................................. 40

5.5  The web service int getDimePort(String uri) creates a thread, streamDigitalItem, that runs in the same context, streaming media with DIME over TCP................................................................. 41
Acknowledgements

By the time this thesis is finalized, many people have had some involvement in its final form.

I could not have completed this work without the confidence and feedback I received from Jack Gerrissen and Frans Mofers at the Open Univeriteit. One of the most important lessons I learnt from them is that the context of information systems is more than just relevant. A thorough understanding of the broader context in which information systems are deployed, is crucial for the development of usable and relevant systems. Frans taught me the pleasure of working with a focused and well delineated work plan, and guided me through the process of its creation.

Steven Van Assche kept his promise, even in difficult and busy times, to help and support me through the entire process of my thesis. I gratefully appreciate the efforts he had to make. As a true colleague, Filip Hendrickx of IMEC helped me out every time I got stuck with something. No matter what practical or conceptual problem I faced, Filip was there to help me find a solution. I wish to thank him for being a such a superb sparring-partner.

This document marks a bigger milestone in my life. I have almost come to the completion of my second academic studies.

It all started out as a wild idea... My former manager, Paul van Kampen, and colleagues at Philips motivated me to take the big step, even if this meant leaving the job that I liked and starting all over at IMEC. I immediately felt supported by my colleagues at IMEC (in particular Paul Six, Johan Cockx and Freddy Potargent) and the Advanced Media project, even in difficult times.

Without doubt, I received the greatest support imaginable from my wife, coach, manager and best buddy: thanks Liesbet, the waiting is almost over ;-).
Summary

The number of mobile and networked multimedia platforms that were introduced into the market has increased dramatically in recent years. They enable consumers to access multimedia productions at diverse places and by diverse means.

Current approaches to multimedia distribution do not scale to the growing set of client configurations and heterogeneous dynamic networks. They are either of a too static –assuming few changes to occur during a multimedia session– or a too homogeneous –targeting a limited set of devices only– nature. New mechanisms are required to enable (multiple) distributed, complex and dynamic adaptation steps. Distributed multimedia adaptation and publication mechanisms assume a certain degree of responsibility and involvement of network nodes that lie inbetween the multimedia producer and consumer. We propose to enable this by means of a distributed architecture that offers a scalable solution to multimedia publication and distribution in heterogeneous environments. It builds upon recent standardization efforts related to web services to provide flexibility and enable widespread deployment.

This document details our multimedia-aware web services that cooperate on a loosely coupled basis to tailor content creators’ multimedia to clients’ environments. We start by providing a global context that is highly relevant to our research. Within this context, we define the exact problems that we wish to address with multimedia web services. The state-of-the-art technologies that are available and could be used to tackle these problems are described next. We provide a detailed description of our architecture that builds upon a selection of these technologies.

The experiments we performed, show that our web service-oriented architecture is applicable to the re-authoring of multimedia presentations. By introducing only a few image-to-image transcoding and XML transformation services, our architecture can already be used to transform multimedia presentations from one format to another, taking into account the client’s capabilities such as screen resolution, available bandwidth and supported player.

A direct result of our approach is that content creators need no longer bother with a multitude of client platform specifications and connecting networks. Their primary concerns are the multimedia applications they publish, the data formats in which those applications are stored and the way in which they are experienced by clients. The actual operations, that ensure a high quality of experience for the client, can be performed by other organisations that operate in the network. Naturally, these operations will be mostly based on guidelines provided by the multimedia creators.
Samenvatting

Het aantal mobiele en genetwerkte multimediaplatformen die gelanceerd worden, is de laatste jaren drastisch gestegen. Ze staan de gebruiker toe multimediaproducties te consumeren op diverse locaties en met diverse middelen.

Huidige multimediadistributietechnieken schalen niet naar zulk een groeiende set van platformconfiguraties en heterogene dynamische netwerken. Ze zijn ofwel te statisch (veronderstellend dat er slechts weinig wijzigingen optreden tijdens de consumptie van multimedia) of te homogeen (gericht op een beperkte set van apparaten). Nieuwe mechanismen zijn vereist om gedistribueerde, complexe en dynamische adaptatie van multimedia mogelijk te maken. Gedistribueerde multimedia-adaptatie en -publicatie vereist dat netwerkbeheerders, die tussen producent en consument opereren, een zekere mate van verantwoordelijkheid opnemen in het bereiken van een mooi eindresultaat. Wij willen dit mogelijk maken door middel van een gedistribueerde architectuur die een schaalbare oplossing moet bieden voor multimediapublicatie en -distributie in heterogene omgevingen. De architectuur kan makkelijk en flexibel in gebruik genomen worden daar ze gebaseerd is op recente standaarden op het vlak van web services.

Dit document beschrijft onze multimedia web services die (los gekoppeld) samenwerken om multimedia van producenten aan te passen aan de omgeving van consumenten. We vertrekken van een globale contextschets waarbinnen we de precieze problemen die we wensen op te lossen definiëren. Daarna beschrijven we de meest recente technologieën die gebruikt zouden kunnen worden om deze problemen op te lossen. We voorzien een gedetailleerde beschrijving van onze architectuur die voortbouwt op een selectie van deze technologieën.

Onze experimenten tonen aan dat onze web service-georiënteerde architectuur toepasbaar is op het herwerken van multimediapresentaties. Beschikkend over slechts enkele beeld-naar-beeld-services en XML transformaties, kan onze architectuur reeds gebruikt worden om multimediapresentaties te transformeren van het ene formaat naar een ander, rekening houdend met de mogelijkheden (schermgrootte en beschikbare bandbreedte) van het apparaat van de consument. Een direct gevolg van onze benadering ligt in het feit dat multimediaproducenten zich geen zorgen hoeven te maken over het enorm aantal diverse apparaten en netwerkomstandigheden. Ze kunnen hun gehele aandacht richten op de vormgeving en de inhoud van hun multimediapresentaties. De eigenlijke adaptatie-operaties, die ervoor zorgen dat de consument de multimedia ‘goed’ kan consumeren, kunnen verricht worden door organisaties die opereren in het netwerk. Vanzelfsprekend zullen deze operaties plaatsvinden binnen de mogelijkheden die door de producenten geschetst worden.
Chapter 1

Introduction

Large-scale production and distribution of media has traditionally been the private playground of large media corporations. The small amount of widespread broadcast and display standards (such as PAL and NTSC) enables these companies to reach large audiences at a very low cost per consumer [24]. Every media company can safely assume that almost everyone is able to receive and consume its media content when distributed according to the standards.

With the unstoppable rise of multimedia on the Internet, a new publication channel appears to settle in the broadcast world. Broadcast companies have always been used to a far-reaching control over the conditions in which their content is distributed (thanks to self-owned terrestrial transmission antennas and strict regulations for cable-distributors) and consumed (thanks to the omnipresent PAL and NTSC standards). However, both the distribution and consumption aspects fall beyond complete control of any organization, as the Internet is a non-hierarchical collection of millions of networks with extremely varied properties, and even more consumer environments (player, network connection, location, preferences, etc) are in use.

Although recent telecommunication laws try to separate the responsibilities of creation and distribution of content as much as possible, broadcasting companies (e.g. BBC and VRT) are turning towards distributors to provide intermediate services in their networks to optimize the experience of their multimedia content on the Internet.

Furthermore, with new multimedia consumption platforms and standards being introduced into the market at a high rate, each one with specific capabilities, massive challenges are arising [25] [58]. Due to the inherent mobile nature of these platforms, their capabilities and connecting network environment may even change dramatically at run-time [6]. Content creators are looking for ways to cost-efficiently publish their content in this heterogeneous environment, where they have to distribute content over multiple dynamic networks to various consumption platforms.

We propose to introduce a web service-oriented architecture with transmoding services in the network, at the proxy server for instance, to offer a scalable approach to publishing multimedia to a heterogeneous environment. With transmoding, we take a broader view to multimedia adaptation than traditional transcoding. A transmoded multimedia item may have
a very different appearance than its original, e.g. a textual transcript of an audio sample containing speech or a bitmap version of a vector image.

Our service-oriented architecture takes the responsibility of tailoring multimedia data to a suitable format for consumer environments. It provides for continuous adaptation\footnote{In the context of our research, the term ‘adaptation’ denotes the adjustment of any multimedia item to match the context in which it is consumed. Please note the difference with adaptive information systems that automatically change their internal structure when faced with changing contexts.} of multimedia presentations and items to the changing environments in which users wish to consume them. Even when the capabilities of clients change dramatically at run-time, our architecture continues to adapt multimedia to well-suited formats.

Ideally, our service-oriented architecture should be able to perform a wide range of complex operations, such as caching, adaptation and optimization of complex multimedia presentations that consist of multiple multimedia items, load balancing of adaptation tasks with a heavy computational load, etc. However, we wish to focus our research on the web service infrastructure (communication mechanisms and encapsulation of transmoding functionality in web services), which is the most fundamental building block of our architecture. We monitor relevant algorithms for other aspects of our architecture (caching, load balancing, etc.) but we do not implement them in the scope of this research.

Since our research goals are made up of basic questions regarding the feasibility of our novel ideas, it is not advisable to adhere strictly to a sequence of design phases like drawing up Use Cases and testing a resulting system according to these Use Cases. Therefore, it is important to note that we do not follow a ‘traditional’ design flow for software architectures. Naturally, once we build the architecture, we do start from requirements, create a functional and technical design and implement and test it. The design phases, however, are only used to substantiate answers to the basic research questions that we pose. As such, our work should be categorized as experimental research, with the following phases:

1. formulate our basic hypothesis: ‘Web services can be used to improve the multimedia experience on the Internet’;
2. build a model (service oriented architecture with media services) and formulate predictions;
3. perform experiments (case studies);
4. analyze results and draw conclusions.

Chapter \[2\] details some context that is highly relevant to our research. In this context, Chapter \[3\] states the basic questions (hypotheses) to which we wish to find an answer. The state-of-the-art technologies that are available and could be used to answer these questions are described in Chapter \[4\]. Our service oriented architecture (model) that builds upon a selection of these technologies, is detailed in Chapter \[5\]. In Chapter \[6\] we emphasize on the differences of our approach to related work, and explain some of the rationale of our architecture. Chapter \[7\] concludes and Chapter \[8\] briefly describes the future work that directly builds upon this research.
CHAPTER 1. INTRODUCTION

After an extended bibliography, we append a list of relevant terms and standards and a paper that is accepted for publication at the IEEE International Conference on Web Services in San Diego, USA. The article is based entirely on the research described in this document.
Chapter 2

Research Context

This chapter provides a global context for the remainder of the document. We start by briefly indicating three important and recent evolutions in the way information is distributed and processed by software systems in Section 2.1. When multimedia needs to be published to a heterogeneous set of devices and players, it needs to be adapted to each set of specific requirements and capabilities. There are currently three popular approaches to this adaptation, as detailed in Section 2.2. We end this chapter with a global picture of the various parties involved in the distribution of multimedia over the Internet and the interests and viewpoints of these parties in Section 2.3.

Chapter 3 builds upon the context provided by this chapter, to provide a specific definition of the problems we address.

2.1 Recent Evolutions in Information Distribution

Three recent technological evolutions illustrate a new era in the way we exchange and distribute multimedia information:

1. client-server systems evolve to N-tier systems (Section 2.1.1);
2. multimedia is becoming a mobile experience (Section 2.1.2);
3. software frameworks are designed evermore loosely coupled (Section 2.1.3).

Both producers and consumers of multimedia could gain considerable benefit from these evolutions. On the other hand, they also confront them with significant challenges. How they employ and embrace these new technologies today will determine their success in the multimedia world of tomorrow.
2.1.1 Client-Server Systems evolve to N-Tier Systems

Client-server systems that have been in use for decades at numerous organizations are evolving gradually into N-tier systems. This evolution may very well decimate the amount of large monolithic software systems, running on expensive mainframes, still in use today. As a logical continuation of this decentralization, Nomadic Services\(^1\) are on the rise, allowing a client to make abstraction of the location and executioner of the information services he employs. Also, the networks that route traffic from source to destination are evolving. Increasing levels of processing are being offered in switches. The extension of this concept to active and programmable network nodes provides an ideal platform for (nomadic) services. As such, the network (e.g. the Internet) becomes a source of computation power, storage and services alike.

2.1.2 Multimedia is Becoming a Mobile Experience

Whereas in the past multimedia experiences have been limited primarily to expensive personal computers and home cinema set-ups, there is a clear trend towards low cost and mobile platforms and applications \(^2\). Originally, only voice communication was feasible, but now Internet access and even video telephony are becoming possible. More and more mobile phones and personal digital assistants support reception, processing and transmission of multimedia content (sound, images, video and music) in diverse formats. The availability of open multimedia standards (see Chapter \(^4\)) has had a major impact on this progression. These standards have made the creation and communication of digital multimedia simple, inexpensive and commonplace.

2.1.3 Software Frameworks are Designed Evermore Loosely Coupled

In recent history, ever fewer organizations have been willing to base their entire information infrastructure on (a set of) closed systems. Integrating –at least partially– open systems, based on standardized frameworks, seems to be the primary strategy for many. The rise of Enterprise Application Frameworks, like Sun’s J2EE and Microsoft’s .NET platform, illustrates this clearly. This approach recently has gained a lot of popularity with the advent of web services and the promise of loose coupling they bring about.

2.2 Approaches to Multimedia Publication

There are currently three popular approaches to preparing multimedia for multiple platforms \(^7\). These are:

1. device-specific authoring (Section 2.2.1),
2. multi-device authoring (Section 2.2.2) and
3. automatic re-authoring (Section 2.2.3).

\(^1\)A brief explanation of this and other terms can be found in Appendix A
2.2.1 Device-Specific Authoring

Device-specific authoring is performed by providing a multimedia presentation in a format (layout, format, size, etc.) that is specifically suited and optimized for one particular device. This presentation format should require no further adaptation to be rendered on the target device.

One way to perform device-specific authoring is to enforce a common-denominator format, like FM for radio and PAL or NTSC for TV images. While adhering to these standards, a device may offer even better capabilities than the standards require (e.g. higher screen resolution). Yet, no matter what the capability of a device is, it will only render the received multimedia according to the specified standards. For instance, a High Definition Television set may be capable of displaying images at a much higher resolution and frame rate than those defined by the standard, nevertheless it will only display images according to the standard’s specifications. Even so, (small) differences between devices that are unavoidably present often bring about a less than ideal result on many devices. For instance, when broadcasters decided to completely switch from black-and-white to colour transmission about forty years ago, the image quality on many black-and-white TV receivers degraded considerably.

![Figure 2.1: Device-specific authoring in the broadcast world: media data that is transmitted over a common carrier can be consumed on a limited variety of devices when encoded according to a common-denominator standard.](image)

The differences between client platforms are often more significant in a network context
CHAPTER 2. RESEARCH CONTEXT

than in the traditional broadcasting world. A standard that is developed specifically for one
platform is not likely to scale to very heterogeneous consumer environments where display
resolution, processing power and memory size differ tremendously and multimedia data becomes
ever more complex and diverse. In such environments, multimedia needs to be made available in
specific encoding and formats suitable for the target consumer environments depending on their
available resources. Therefore, the Internet today follows a different approach to the delivery
of multimedia. A multimedia presentation is typically placed on a server in multiple versions,
each one targeting a popular network connection speed and multimedia player configuration.
Some examples of very popular versions are 56kbp/s, 100kbp/s and 300kbp/s versions of Windows
Media [3] and Real Media [4]. Naturally, web pages that are developed for specific browsers
and screen resolutions (e.g. 800 x 600 pixels) can as such also be regarded as device-specific
versions of multimedia presentations.

Device-specific authoring clearly enables multimedia content providers to publish their
presentations to large audiences at a very low cost. The total production and transmission
cost is divided by the number of receivers (in the case of broadcasting, the division is linear),
which usually is very large. However, device-specific authoring doesn’t scale to the very large
set of client configurations and heterogeneous dynamic networks that multimedia distributors
will have to address in the near future. Preparing specific presentations for every popular
configuration is not sustainable when the number of popular configurations grows strongly. Even
worse, in the case of mobile terminals, the device configurations can vary in time, depending
on for instance battery power and available wireless bandwidth.

2.2.2 Multi-Device Authoring

Other interesting approaches for video publication are proposed by MPEG-2 and MPEG-4.
They describe a layered approach to video encoding, allowing one multimedia presentation to
scale to different bandwidths. On top of a base layer, which contains encoded media that every
client should be able to receive and decode, reside several enhancement layers with extra infor-
mation that can be consumed by clients with higher bandwidths and decoding capabilities [39].
Such an approach is very suitable for highly responsive adaptation of fairly simple multimedia
(audio and/or video), within a limited range of capability changes. The scalable approaches
can be described as multi-device authoring.

The World Wide Web Consortium (W3C) employs another example of multi-device au-
thoring. It proposes the use of cascading style sheets (CSS, [41]) in combination with HTML.
In CSS, a style sheet defines a set of display attributes for different structural portions of a
document. Each style sheet can target a particular group of users or web browser configurations.

Multi-device authoring mechanisms provide excellent results when used for a limited
range of consumer environments in specific situations. However, it is impossible to anticipate
all modifications that need to be made when complex multimedia content (multiple audio,
video, text and other items) needs to be published to a very heterogeneous set of consumer
environments.
2.2.3 Automatic Re-Authoring

In complex, distributed and dynamic situations, where single-device and multi-device authoring come short, automatic re-authoring may prove to offer a more elegant solution. Automatic re-authoring is based on a software system that analyses a multimedia presentation together with the characteristics of the target environment and transforms the presentation (and the items therein) so that it can be transported efficiently and rendered appropriately on the target device. The re-authoring software system is often placed on a proxy server, as proposed by [7] [12] [17]. In MPEG-21, the terminology Digital Item Adaptation (DIA) is used to denote the context of automatic re-authoring. In this document, we regard the terms ‘automatic re-authoring’ and ‘adaptation’ as synonyms.

Automatic re-authoring is particularly interesting when consumer environments change within a session. Such run-time changes often occur in mobile environments where sudden drops in bandwidth or processing power may occur at unpredictable moments in time. It is often combined with various caching algorithms to limit the amount of re-authoring and transmoding work that needs to be performed [25].

Re-authoring can occur in a number of ways:

- Encoding: Often used as a term for the process of reducing the amount of data required for the representation of multimedia. For instance, when analogue audio is recorded digitally as 16-bit pulse coded modulation (PCM), it is said to be digitally encoded in PCM. When that PCM audio is compressed with the MPEG-1 layer 3 compression algorithm, it is encoded in mp3.

- Decoding: This term is mostly used to express the inverse process of encoding. Translating mp3 audio in PCM for editing or playback is an example of decoding.

- Transcoding: Used as a more generic term of any combination of decoding and encoding steps. These steps can perform any of the following changes on the media:
  
  - coder-decoder (codec) (e.g. translating a MPEG-2 encoded video stream into a MPEG-4 stream);
  - bit rate (e.g. re-encoding a 192bps mp3 audio stream into a 128bps mp3 audio stream);
  - resolution (e.g. downscaling an image of 640x480 pixels to 320x240 pixels);
  - etc.

- Transmoding: With transmoding, we denote any combination of transcoding operations plus the process of changing the modality of the media. A change in modality of media occurs when it is translated from one occurrence (e.g. text) to another (e.g. audio). This concept offers the most flexibility when combining multiple media items into a single presentation, since drastic reductions in bandwidth become possible per media item. For example, consider a multimedia presentation consisting of a video stream and an audio stream. When transmitted over very low bandwidth connections, one may choose to transmode the video stream into a set of still images that are taken from the video.
and are transmitted at the appropriate time intervals (e.g. every 3 seconds). The large amount of bandwidth that is saved by this transmoding allows for the audio stream to be transmitted in higher quality than otherwise would have been the case.

2.3 Distribution of Media in an On-Line Context

The mechanism of distributing multimedia by broadcast differs drastically from network-based distribution. Broadcasting experts sometimes refer to the Internet-like network distribution as the *on-line* context. In an on-line context, the rules of multimedia distribution and the involved parties are not yet fixed. Whereas the broadcast context offered a fairly stable and controlled distribution environment, with a very limited number of parties involved, the on-line context (i.e. the Internet) is still quite anarchistic and unstable. The Internet is not managed by an organization, it is a non-hierarchic collection of networks. Yet some network managers (e.g. Internet service providers, or ISPs) do try to offer guaranteed bandwidth and services within their network. Both companies and private persons are becoming more and more willing to pay for these services and a guarantee of quality.

The on-line context can be considered from three distinct viewpoints: the multimedia content producer’s viewpoint, the multimedia consumer’s viewpoint and the multimedia content distributor’s viewpoint. Each viewpoint has its particular requirements, preferences and goals, that influence the composition and architecture of multimedia distribution mechanisms on the Internet.

2.3.1 A Multimedia Content Provider’s Viewpoint

A primary goal of any content provider is to reach his (preferably large) audience at a low cost per consumer. When a content provider offers multimedia presentations for publication in an on-line context, however, he is faced with a large set of unpredictable and potentially influencing factors that complicate its publication process. The content provider himself cannot anticipate most of the important factors that influence the way its presentations are experienced by consumers.

Individual packets of data from a presentation may follow different paths from media servers to client devices. This is a direct consequence of the underlying network protocol, in this case the internet protocol (IP). The reliability and speed of the different followed paths can differ, which makes guaranteeing a certain quality level at the client devices hard. This situation is totally different from the broadcast context, where a predictable quality and user experience are commonplace.

Next to the unpredictable nature of the network, other factors influence the consumer experience greatly. Each consumer has a device and network connection with particular properties. For instance, a multimedia presentation that targets a personal digital assistant (PDA) with a universal mobile telecommunications system (UMTS) connection has totally different requirements from one that targets a high-end multimedia PC with a high-speed local area network (LAN) connection.
It is almost impossible to provide an ideal presentation format at the media server, that takes into account all relevant parameters of all the consumers. That is why we believe that intermediate parties, such as ISPs, will play an important role in optimizing the multimedia experience for consumers. The network servers (e.g. HTTP proxy servers) that these parties operate are shown in the middle of Figure 2.2. Such intermediate ‘active’ network nodes can offer services that greatly influence the multimedia experience in the on-line context.

2.3.2 A Multimedia Content Consumer’s Viewpoint

The viewpoint of multimedia content consumers is important to consider. Indeed, the ultimate goal of content providers is to reach large audiences. That goal can only be achieved when consumers experience the multimedia in an attractive manner and their quality wishes are met.

What a consumer ideally wants to obtain in an on-line context is the universal availability of multimedia, continuously adapted to his situation. Pereira captures the consumer’s goals in two concepts: Universal Multimedia Access (UMA) and Universal Multimedia Experience (UME) [28].

UMA The notion (and associated technologies enabling) that any content should be available anytime, anywhere, possibly after adaptation. This may require that content be transcoded from, for, example, one bit rate or format to another or transcoded across modalities: e.g., text to speech (also known as transmoding). UMA concentrates on altering the content to meet the limitations of a user’s terminal or network environment.
UME The notion that a user should have an equivalent, informative experience anytime, anywhere. Typically, such an experience will consist of multiple forms of multimedia content. Each will be adapted as in UMA but rather than to the limits of equipment, to limits that ensure the user has a worthwhile, informative experience. Thus, the user is central and the terminal and network are purely vehicles of the constituent content.

Figure 2.3: When receiving media in an on-line context, three main parties are involved: the Consumers (with various devices), the Media Providers and the Service Providers.

Note that the requirement that multimedia content should be available *anytime* and *anywhere* might be somewhat exaggerated. A consumer indeed wants the idea (or illusion) that he or she has such ubiquitous multimedia availability. Yet, most consumers tend to behave in more or less predictable ways.

The degree in which content providers and distributors are able to provide UMA/E to consumers determines the perceived quality of multimedia on the Internet. It is therefore of capital importance that producers and distributors cooperate in the future to achieve high levels of UMA/E at a low cost per consumer.

### 2.3.3 A Multimedia Content Distributor’s Viewpoint

In order to enable the goals of the content provider and consumer, it is of vital importance that at certain places in the network, someone takes the responsibility of adapting the multimedia to its current network context and the client device’s capabilities.
As Figure 2.3 illustrates, the Internet service providers (ISPs) (e.g. Belgacom and Te- lenet) are in an ideal position, as intermediaries between the provider and the consumer, to provide an infrastructure for adaptations and thus to help enable UMA/E. While the consumers wish to experience the multimedia ‘conceptually’ provided by the content providers (e.g. VRT and VMMa), they can receive an optimized version from their ISPs. The bubbles in the figure illustrate connections as they are perceived by consumers, whereas the full lines represent actual network connections (e.g. UMTS, ADSL, etc.).

Content distributors provide an optimal ‘Internet experience’ for consumers mainly by providing fast and efficient transport of data. Hence, full and automatic adaptation of multimedia presentations, created by content providing organisations, seems to be out of scope for distributors. Also, content providers are not likely to welcome a distribution environment where they would have to give up their control over resulting multimedia presentations (as viewed by consumers) in favour of distribution companies. This situation calls for a mechanism that allows content providers to specify possible adaptations and changes to their productions as metadata. This metadata can be distributed together with the multimedia presentations it describes and interpreted by content distributors in order to adapt the presentations to the context of consumers within the boundaries that are set by the authors.
Chapter 3

Problem Definition

The previous chapter provided a global context for our research. In this chapter, we define precisely which problems we wish to address. We start by illustrating the challenges on a conceptual level in Section 3.1. We explain that these challenges have emerged due to a difference in background between multimedia and internet technologies. Building upon the challenges, we define the three fundamental open questions which form the basis for the research.

In Section 3.2, we focus our research on one particular problem. After we indicate the location in the network where we wish to provide a solution, we define the specific case study that we will examine and enable in the rest of this document (Section 3.2.2). A second use case is given in Section 3.2.3. We do not elaborate on this use case within our research, but we do mention it here to enable the reader to get an impression of related problems that could benefit from our ideas.

When meeting the challenges of Section 3.1 and enabling the case study mentioned in Section 3.2, we wish to differentiate us from related research and industry initiatives. Therefore, we indicate our key challenges and unique contribution in Section 3.3. A detailed discussion of our contribution can be found in Chapter 6.

3.1 Multimedia and Distributed Information Systems: A Clash of Cultures?

The goal of this research is to investigate whether web service technologies and architectures can offer a useful contribution to the distribution and transcoding of media. We consider the distribution of several media (audio, moving video, still images, text, etc.) over (IP based) networks. The problems that occur when distributing media over networks are often related to the fact that a multitude of media formats are in use on the Internet. These media formats can differ in several aspects, such as:

- presentation: the way in which media is presented to the user and how the user can interact with it;
• (display) resolution: the width and height of the display on which the media is rendered, the same applies to the sampling frequency of audio that is played;

• transport medium: the transmission protocol and mode over which the media is transported (e.g. RTP \[33\] over IP);

• player format: the specific application that is used to consume the media (e.g. RealOne \[4\] and Apple Quicktime \[1\]);

• encoding: the algorithm that is used to compress the media to decrease the amount of data that needs to be transmitted over the network;

• numerous other aspects.

In this document, we use the term Media Services for the extended web services that can handle media in a network. Employing media services enables a decentralized approach to sharing and consuming media. At various places in the network (e.g. at an Internet Service Provider’s proxy server), media services can be deployed that perform specific tasks. These media services need to be aware only of the formats in which they are able to handle media data. As such, media can be adapted at several places on its path from the provider to the consumer. For example, a particular media service may decide to increase the error-resilience of a media item (by adding redundant coding data) when it knows the media is about to be transmitted over an unreliable (e.g. wireless) network. Such adaptations can occur independently of the formats in which the media data is processed by other services in the network. The translation between formats may also be performed by media services that are deployed in the network. Media services allow every node in a network to receive and transmit media in the format that is most suited for its current situation and network environment.

Apparently, media services have a lot to offer for multimedia and networks. However, up until now the ‘Internet’ world of web services and the ‘multimedia’ world of MPEG standards have remained fairly separated. It is true that many web sites offer some multimedia experience with animated images and sometimes even embedded audio clips. A real multimedia experience, with movies and sounds around every corner, however, is still painfully absent on the Internet. All too often, multimedia that is consumed from the network is hampered by interruptions or long delays before actual playback can occur. The interruptions in streaming media are caused by the loss of data-packets in an unpredictable network. The delays at startup occur when the media is not consumed directly upon reception by the client, but it is buffered in local memory to prevent interruptions during playback. Such problems seem to be inherent to the unpredictable nature of the Internet. However, they can be alleviated to a great extent by proper quality-of-service (QoS) management at critical places in the network. We are convinced that web services could play a significant role in providing such distributed QoS management.

There are three fundamental questions that need to be answered. We wish to find an answer to those questions in this research.

• To which extent are web services applicable to large-scale, complex and distributed media processing?
• Where do web services offer an important contribution to better media distribution/processing in a network environment?

• How should web services be built in order for them to perform optimally as media processing elements?

Naturally, the answers to these questions will not be specific, nor final, due to their abstract nature. This research will not answer these questions to a full extent, since they entail much more than can be investigated in this scope. The answers we do find to these questions, however, will act as guidelines for future research to be performed.

3.2 Automatic Re-Authoring with Media Services

Section 3.1 stipulated the basic questions we want to see answered. In this section, we start to focus our research specifically on a particular case study. It is this problem we will tackle with media services, knowing that there are other related problems that could be solved in a similar way.

3.2.1 Location of the Media Services

As illustrated in Section 2.3, problems occur when exchanging multimedia data over the Internet, as numerous different players/configurations and network/transport conditions are present. The required adaptation or automatic re-authoring of multimedia can take place at any combination of three places in the delivery path of multimedia:

• at the multimedia server;

• in one or more network nodes;

• at the client;

One of the most important advantages of web services technologies is that they allow software systems to employ services without having to take into account the actual location and host platform of the services. In other words, changing the actual location (i.e. server, network node or client device) of a particular service has no impact on the functionality provided. However, this location can have a major impact on the performance of the entire system.

As illustrated in Section 2.3, the content provider does not want to – and cannot – provide computational power for all the adaptations that are required for every single consumer who consumes its multimedia presentations. However, he can anticipate possible adaptations to its presentations and describe how these adaptations could take place. Also, the consumer devices should not be required to adapt any incoming multimedia presentation to a format that is well suited for its context. Such client-side adaptation may occur, but only in a limited number of well defined scenario’s. For instance, when only small still images are available on a server, they can be enlarged by the client device in order to use its full display size optimally. Rescaling such
images prior to transmission to the device would cause an unnecessary overhead in bandwidth. Therefore, the most suited location to perform automatic re-authoring is in the network nodes. This re-authoring in the network should be based on the client capabilities/context and the guidelines provided by the content provider.

3.2.2 Case Study: Adaptation of Player-specific Presentations to Other Players

A consumer who selects, from within a web browser, a particular multimedia presentation like a movie clip, is often required to indicate the player software that is installed on his PC. This is a direct consequence of the device-specific authoring approach that is embraced by the Internet community today. Figure 3.1 illustrates this situation, with a web page offering a choice between Realmedia’s RealPlayer, Microsoft’s Media Player and Apple’s Quicktime. Note that in this example the page appears to offer support for Quicktime version 4 only, so one might expect problems when a different version of this player is installed. It is up to the consumer to identify the player software he has installed on his PC, verify the correct version number, and select the correct version of the multimedia item in the web page. Needless to say, this causes all kinds of problems on a regular basis.

Figure 3.1: Problems occur when exchanging multimedia data over the Internet, as numerous different players and PC-configurations are in use.

Even within a particular media player choice, a screen resolution and bandwidth needs to be selected, as illustrated by Figure 3.2.

The specific case study that we wish to demonstrate is elaborated in the remainder of this section.

The World Wide Web Consortium (W3C) has defined the synchronized multimedia in-
CHAPTER 3. PROBLEM DEFINITION

Figure 3.2: Even within a particular media player scenario, different screen resolutions and bandwidths are possible, as illustrated by this screenshot from a web page.

integration language (SMIL) [46]. SMIL enables simple authoring of interactive audiovisual presentations. It is typically used for multimedia presentations which integrate streaming audio and video with images, text or any other media type. The language is entirely based on XML. An example SMIL document is listed in Figure 5.10. SMIL documents can only be consumed with specific player software. Some popular media players, like Apple’s Quicktime, offer various degrees of support for SMIL.

We consider the case where consumers wish to view SMIL presentations from within their web browser, without installed plug-ins that offer specific support for SMIL. Since the size of web browser windows on PC screens can vary considerably, we want to adapt the generated web pages to the window size in which they will be rendered. The SMIL presentations we wish to adapt consist of still images and pieces of text that are updated after given time intervals. With this functionality, nice slideshow-like presentations can be created, with images and accompanying textual descriptions.

The adaptations (SMIL-to-HTML translation and image resizing) need to be performed ‘in the network’ with as little interaction as possible of the multimedia content providers and consumers. In Section 5.4.1 we show how we solve this problem with the web service oriented architecture we introduce in Section 5.

3.2.3 Case Study: Adapting the Format of Multimedia Exchanged between Mobile Phones.

A related problem that also could benefit from our technology is shown in Figure 3.3. It is given here as an extra illustration, even though we will not discuss it further in this document. Currently, many people are buying a mobile phone with the intention of using its (built-in) camera and display-capabilities to send others small messages with pictures. This mechanism, known as the Multimedia Messaging Service (MMS), provides users with the ability to send messages comprising of multimedia items (text, sounds, images and even video) to other MMS
capable mobile phones.

In practice, the exchange of multimedia with MMS suffers important problems, though. Already today, there are literally dozens of different mobile devices with MMS capability on the market. Yet, there is no provision for mechanisms to adapt multimedia items to particular capabilities, except for some basic receiver-side rescaling of incoming items, which is far from ideal (see Section 3.2.1). Images that are exchanged between devices with different screen and camera resolutions, for instance, are not adapted to suit the capabilities of receiving devices. Proper adaptation (i.e. in the network) of multimedia items for particular mobile phones is not foreseen in the near future, so we expect problems like these to continue to occur for a while.

![Figure 3.3: Problems occur when exchanging multimedia data between heterogeneous devices, even when using recently standardized multimedia-exchange formats like MMS.](image)

### 3.3 Key Challenges and Contribution

To allow web services to handle multimedia, some key software technologies need to be identified and developed. We will evaluate existing technologies and standards and extend them where useful.

Current standard web service technologies do not provide the necessary mechanisms to perform multimedia adaptation. The original contribution of this research lies in the topics which we identified as the most fundamental obstructions to automatic multimedia re-authoring with web services:

- Design of a web service oriented architecture for automatic re-authoring (Section 3.3.1).
- Inventarization of required brokerage functionality to enable automatic re-authoring (Section 3.3.2).
- Design and implementation of mechanisms that enable multimedia re-authoring with web services (Section 3.3.3).
- Exchange of multimedia data between web services (Section 3.3.4).
CHAPTER 3. PROBLEM DEFINITION

The topics we describe here often build upon related work which is described in Chapter 4. In Chapter 5, we design and implement our solutions. We distinguish all the aspects of our approach from related initiatives (i.e. the current state-of-the-art) and elaborate on our rationale in Chapter 6.

3.3.1 A Service-Oriented Architecture for Multimedia Adaptation

Some architectures for distributed automatic re-authoring have been developed already, as indicated in Section 4.3. Yet none of these architectures employ standard web service protocols and mechanisms. We apply the standards whenever possible and thus build a true web service oriented architecture for complex multimedia adaptation.

3.3.2 Broker Services

The development and deployment of multimedia transmoding web services on itself is not sufficient to enable automatic re-authoring. The procedure of finding a suitable media processing service (e.g. transmoder) is a common operation in numerous media frameworks. Quite often, such a processing block is selected only by the particular input and output formats it supports.

Automatic re-authoring of multimedia requires registration and identification of the transmoding web services, and brokerage services that make sure the transmoding web services are invoked in the right order to tailor multimedia to consumers’ environments. We will define the requirements this places on both the transmoding and broker services.

3.3.3 Web Services that can handle Media Transmoding

A lot of research currently is performed on the application of web services in diverse domains. Numerous large corporations are already starting to integrate web services to organize logistics, production planning and (financial) transactions, both internally and across company boundaries.

Yet, we have encountered very little initiatives that try to handle complex media using web services. Today’s web services are stateless pieces of software that perform operations within a relatively short time span. This means that transmoding operations that have a long duration (e.g. transcoding of a 2 hour video easily takes up several hours) can not be performed, since they would need to be performed in a single operation. Calling a web service to perform a transmoding operation and waiting for several hours to find out if it was successful obviously is not an elegant option. Furthermore, as consecutive calls to a single web service have no notion of each other, splitting up a large transmoding operation in multiple related partial tasks, is not feasible.

The development of stateful web services that support efficient transcoding of media items is a key contribution of this research to the growing convergence of multimedia and the Internet.
3.3.4 Transport of Multimedia Data from and to Web Services

Currently, the only available communication mechanism for web services uses XML documents, packaged in textual messages with a mechanism named ‘SOAP’ (see Section 4.1.2 for a discussion on SOAP). With this mechanism, one-way textual messages can be exchanged. It offers no support for streaming (binary) data, since messages are processed only when they have been received completely. When we wish to send binary data (i.e. multimedia items), we require a mechanism to include or attach of binary multimedia data with the textual messages. We will investigate the applicability of a recent industry initiative, DIME (see Section 4.2.2 for more information on DIME), for this purpose.
Chapter 4

Related Technologies

In this chapter we briefly discuss of the technologies that are most relevant in our context. The discussion includes an evaluation of the strengths, weaknesses and limitations of the various technologies when faced with our problem statement of Chapter 3. How we use these technologies in our architecture is specified in the following chapter.

In Section 4.1 we describe highly relevant standards of ISO/IEC and the World Wide Web Consortium (W3C). Related industry initiatives in the field of web services development are given in detail in Section 4.2. Finally, the research projects that aim at solving (parts of) our problem are given in Section 4.3.

4.1 Standards

4.1.1 ISO/IEC JTC1/SC29/WG11 – MPEG

Within the international organization for standardization (ISO), the joint technical committee 1, sub committee 29, working group 11 (ISO/IEC JTC1/SC29/WG11) standardizes algorithms and frameworks for the coding of moving pictures and audio. This group is more commonly known as the ‘Motion Picture Expert Group’ (MPEG). Its omnipresent mp3 audio files and DVD MPEG-2 video coding are some of the most widely adopted standards in recent history. The MPEG standards relevant for our technology are MPEG-2, MPEG-4 and MPEG-21.

MPEG-2

MPEG-2 (formally ISO 13818) is developed as the standard for digital television and DVD. It consists of 10 parts, of which 13818-1 (Systems) and 13818-2 (Video) are most relevant to us.

MPEG-2 video had to be forward and backward compatible with the older MPEG-1 video coding standard. Forward compatibility, meaning that an MPEG-2 video decoder is able
to decode MPEG-1 video bitstream, was achieved effortlessly, since MPEG-2 video is developed as a superset of MPEG-1 video. The backward compatibility was met thanks to the definition of MPEG-2 video as a scalable bitstream. Scalability is that property which allows decoders of various complexities to be able to decode video at a resolution or quality suited for their capabilities, starting from a single bitstream.

MPEG-2's generalized codec supports two layers of scalability: a base layer and a higher layer that provides enhancement over the base layer \[30\]. Conceptually, MPEG-2 offers support for various types of scalability, but usually only spatial (image resolution) and temporal (frames per second) scalability are supported.

While the scalability of MPEG-2 may provide a mechanism for the delivery of video to more than one platform, it is much too limited to target the diverse client devices that we consider in our research. Furthermore, MPEG-2 is entirely unaware of distributed networks with processing nodes. It simply describes a video bitstream syntax.

**MPEG-4**

MPEG-4 (formally ISO 14496) is proposed as the open standard for multimedia applications. The original target of MPEG-4 was very low bit rate video coding, but it was extended to generic coding of audiovisual objects for various multimedia applications (Internet video, interactive home shopping, virtual reality games, etc.). The MPEG-4 standard consists of 6 parts:

- 14496-1: Systems
- 14496-2: Visual
- 14496-3: Audio
- 14496-4: Conformance
- 14496-5: Software
- 14496-6: Delivery Multimedia Integration Framework

Again, the Systems and Visual parts are the most relevant to our research.

The Visual part offers a generalized scalability framework that supports both temporal and spatial scalability \[35\]. This framework offers means of scaling the decoder complexity if processor and/or memory resources are limited and vary in time. Scalability also allows for graceful degradation of quality (i.e. degradation of the perceived quality in a predictable and convenient way) when resources are limited or change continually.

The main difference with MPEG-2 scalability lies in a more advanced preprocessing stage and the broader range of encoders and decoders allowed in both the base and enhancement layers. Furthermore, MPEG-4 allows multiple enhancement layers, enabling a finer grain scalability than MPEG-2.

While MPEG-4 already goes a lot further than MPEG-2 in its scalable coding and delivery of multimedia, the standard does not define where and how the actual adaptation steps
should take place. MPEG-4 does not provide a (distributed) architecture for the adaptation of multimedia to particular client platforms and contexts. However, it is a good candidate to be used in our solution for describing scalability of simple streaming multimedia (i.e. video or audio).

MPEG-21

The MPEG-21 multimedia framework (formally ISO 21000) is an open framework for multimedia delivery and consumption for use by all the players in the delivery and consumption chain \[21\]. It aims to support the augmented use of multimedia resources across a wide range of networks and devices used by different communities \[9\].

An MPEG-21 Digital Item is a structured digital object with a standard representation, identification and meta-data within the MPEG-21 framework. Examples of such digital items are: a movie clip, a picture or even an audio Compact Disc. Manipulation can consist of any combination of transmoding operations on any combination of digital items \[14\].

The first seven parts in MPEG-21 are:

- Part 1: Vision, Technologies and Strategy
- Part 2: Digital Item Declaration
- Part 3: Digital Item Identification
- Part 4: Intellectual Property Management and Protection (IPMP)
- Part 5: Rights Expression Language
- Part 6: Rights Data Dictionary
- Part 7: Digital Item Adaptation

Part 7, Digital Item Adaptation (DIA) \[10\], \[22\] describes the manipulation of multimedia content in a networked context, to tailor for the needs of end-user terminals. One of the most important applications of DIA lies in the support it offers to enable Universal Media Access (described in Section \[2.3.2\]). As shown in Figure \[4.1\] it offers an extensive description of various related technologies \[9\]. The characteristics that are most relevant to our research describe the usage environment in terms of terminal, network, delivery, user and natural environment capabilities. They are related to CC/PP \[50\], as detailed in Section \[4.1.2\]. These are the described capabilities and characteristics:

- user characteristics, including user preferences and demographic information;
- terminal capabilities, including acquisition properties, device type (e.g. encoder, decoder, gateway, router, camera) and profile, output properties, hardware properties (e.g. processor speed, power consumption, memory architecture), software properties, system properties and IPMP related capabilities;
• network capabilities, including delay characteristics, error characteristics and bandwidth characteristics;
• natural environment characteristics, including location, type of location (e.g. indoor, outdoor, public place, home, office), available access networks in a given area, moving speed of terminal.

<table>
<thead>
<tr>
<th>Usage Environment Description Tools</th>
<th>User Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Terminal Capabilities</td>
</tr>
<tr>
<td></td>
<td>Network Characteristics</td>
</tr>
<tr>
<td></td>
<td>Natural Environment Characteristics</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Digital Item Resource Adaptation Tools</th>
<th>Bitstream Syntax Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Terminal and Network QoS</td>
</tr>
<tr>
<td></td>
<td>Bitstream Syntax Description Link</td>
</tr>
<tr>
<td></td>
<td>Metadata Adaptability</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Digital Item Declaration Adaptation Tools</th>
<th>Session Mobility</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DIA Configuration</td>
</tr>
</tbody>
</table>

Figure 4.1: Overview and organization of Digital Item Adaptation tools

MPEG-21 is a very large set of standards that describe almost any aspect of multimedia delivery and consumption. However, many questions remain unanswered in the standards. Before an actual system can be built with MPEG-21, many missing parts need to be filled in. For instance, DIA provides characteristics, tools and mechanisms for multimedia adaptation, but it does not specify how and where the adaptation should occur. Only today, in the sixth Framework Program of the European Union, people are taking the initiative of trying to build actual operational systems according to (parts of) the MPEG-21 standard.

4.1.2 W3C

Web Services Activity

The World Wide Web Consortium (W3C) works on the infrastructure of Web services, defining the architecture and the core technologies for Web services [47].

In September 2000, W3C started the XML Protocol Activity in order to address the need of an XML-based protocol for application-to-application messaging. From this activity emerged the Simple Object Access Protocol (SOAP) [48]. In January 2002, the Web Services Activity was launched, subsuming the XML Protocol Activity by extending its scope to all the different aspects of Web services.

The goal of the Web Services Activity is to design a set of technologies fitting in the Web architecture in order to lead Web services to their full potential.

The activity is currently composed of three Working Groups and one Interest Group, coordinated by one Coordination Group:
• XML Protocol Working Group

The initial focus of the XML Protocol Working Group is to create simple protocols that can be ubiquitously deployed and easily programmed through various tools. The goal is a layered system which directly meets the needs of applications with simple interfaces (e.g. getStockQuote, validateCreditCard), and which can be incrementally extended to provide the security, scalability, and robustness required for more complex application interfaces. XML-based messaging and remote procedure call (RPC) systems, layered on standard Web transports such as HTTP and SMTP, can effectively meet these requirements. Specifically, the XML Protocol Working Group is chartered to design the following four components:

- An envelope for encapsulating XML data to be transferred in an interoperable manner that allows for distributed extensibility and evolvability as well as intermediaries.
- A convention for the content of the envelope when used for RPC (Remote Procedure Call) applications. The protocol aspects of this should be coordinated closely with the IETF and make an effort to leverage any work they are doing, see below for details.
- A mechanism for serializing data representing non-syntactic data models such as object graphs and directed labeled graphs, based on the datatypes of XML Schema.
- A mechanism for using HTTP transport in the context of an XML Protocol. This does not mean that HTTP is the only transport mechanism that can be used for the technologies developed, nor that support for HTTP transport is mandatory. This component merely addresses the fact that HTTP transport is expected to be widely used, and so should be addressed by this Working Group. For coordination with the IETF, see below.

SOAP is a simple one-way protocol that provides a flexible and extensible way to send structured and typed XML data over any transport protocol. Sending various types and large loads of multimedia data with SOAP, however, quickly becomes complex and inelegant. A lot of extra work is involved when encoding binary data (to fit the character encoding of the SOAP envelope) and splitting it into smaller chunks (to limit the effects of packet-loss in transport) \[60\].

A W3C Note that never made it to a recommendation is the SOAP Messages with Attachments (SwA) proposal \[54\]. The note was published by Microsoft and Hewlett Packard Labs in December 2000, but the work was discontinued later on when Microsoft developed WS-Attachments with DIME (see further) as an alternative. SwA uses the MIME multipart mechanism \[18\] for encapsulating SOAP envelopes together with (binary) attachments. MIME is well known thanks to its widespread use in email applications for attaching any arbitrary data to textual messages. In Section 4.2.2, we will elaborate on a very similar, yet more recent initiative and compare it to the MIME-based approach.

• Web Services Description Working Group

The Web Services Description Working Group develops the Web Services Description Language (WSDL) \[53\]. WSDL provides a model and an XML format for describing Web
services. It enables one to separate the description of the abstract functionality offered by a service from concrete details of a service description such as how and where that functionality is offered.

WSDL describes a Web service in two fundamental stages: one abstract and one concrete. Within each stage, the description uses a number of constructs to promote reusability of the description and separate independent design concerns. At the abstract level, WSDL describes a Web service in terms of the messages it sends and receives. Messages are described independent of a specific wire format using a type system, typically XML Schema. At the concrete level, a binding specifies transport and wire format details for one or more interfaces. An endpoint associates a network address with a binding. And finally, a service groups together endpoints that implement a common interface.

- **Web Services Choreography Working Group**

  The Web Services Choreography Working Group specifies mechanisms for a common definition of the sequence and conditions in which web service messages are exchanged. It claims that activities that involve multiple different organizations or independent processes that use web service technology to exchange information can only be successful if they are properly coordinated. This means that the sender and receiver of a message know and agree in advance on:

  - the format and structure of the (SOAP) messages that are exchanged and
  - the sequence and conditions in which the messages are exchanged (e.g. a Pay-HotelRoom service may require a ReserveHotelRoom service to be called prior to its invocation).

WSDL and its extensions provide a mechanism by which the first objective is realized. However, it does not define the sequence and conditions, or choreography, in which messages are exchanged. That is the specific focus of this working group.

- **Semantic Web Services Interest Group**

  The purpose of the Semantic Web Services Interest Group is to provide an open forum for W3C Members and non-Members to discuss Web Services topics essentially oriented towards integration of semantic web technology into the ongoing Web Services work at W3C. Using semantic web technology, users and software should be able to discover, invoke, compose and monitor web services in a more intelligent and automated fashion.

**Device Independence Activity**

The mission of the Device Independence Activity is to avoid fragmentation of the Web into spaces that are accessible only from subsets of devices. Its aim is to enable access to a unified web from any device in any context by anyone.

In particular the Device Independence Working Group

- collects requirements for Web access via various kinds of presentation devices;
• reviews related specifications within and outside of W3C;
• provides use cases and requirements to related activities within W3C;
• describes techniques which allow authors to improve management of device dependencies;
• in some specific areas not covered by other groups, it proposes recommendations that will lead to enhanced device independence.

In March 2003 the Device Independence Activity took over the work of the CC/PP Working Group. CC/PP stands for Client Capability/Preference Profile. A CC/PP profile is a description of device capabilities and user preferences. This is often referred to as a device’s delivery context and can be used to guide the adaptation of content presented to that device. CC/PP is based on the Resource Description Framework, a general purpose metadata description language also developed by W3C [52]. CC/PP is designed to be broadly compatible with the UAProf specification [57] from the WAP Forum.

4.2 Industry Initiatives

4.2.1 Apache Web Services Project

The Apache Software Foundation provides support for open-source software projects, involved in the apache web server technology. It provides software under an open source license, building on efforts of a large community of developers and users. Adherence to standards is crucial for the Apache community.

One of the major Apache projects today is the Web Services Project [5]. It groups more than ten development initiatives in the field of web services and enhancements. The initiatives that are the most relevant to our research are:

• AXIS: AXIS offers an implementation of standard SOAP. SOAP version 1.1 is fully supported and an implementation for SOAP 1.2 is in beta stages of development.

• WSFX: While Apache’s other web services initiatives currently target the lower end of web services stack (with SOAP, UDDI etc.), more functionality is required in many application domains. The Web Service FX (Functionality Extensions) subproject aims to accelerate and guide this process. WSFX currently consists of three sub projects:

1. Addressing - An implementation of the WS-Addressing specification
2. Sandesha - An implementation of the WS-Reliable Messaging specification
3. WSS4J - An implementation of the WS-Security specification

WS-Addressing, WS-Reliable Messaging and WS-Security are part of a growing set of enhancements to the standard web service mechanisms (WS-* or WSE). The WS-* specifications are currently under development by Microsoft, IBM and BEA. The following section provides more information on these specifications.
From our study of the Apache initiatives, it quickly became clear that they do not offer solutions for the problems that we pose in Section 3. The most important technology that is lacking is the transport of (multimedia) data between web services.

4.2.2 Microsoft .NET

Next to many, many other things, Microsoft .NET is a framework used for building and running all kinds of software, including Web-based applications, smart client applications, and XML Web services.

Microsoft has defined the Web Services Enhancements (WSE, or WS-*) for .NET to provide various enhancements to web service mechanisms in the field of security, scalability and performance. The Web Services Enhancements (WSE) 1.0 Service Pack 1 provides support for security features such as digital signature and encryption, message routing capabilities, and the ability to include message attachments that are not serialized into XML, using DIME.

Direct Internet Message Encapsulation (DIME)

The Direct Internet Message Encapsulation (DIME) specification defines a mechanism for packaging binary data with SOAP messages. It offers a way to send (binary or text) attachments along with SOAP messages, regardless of their format and encoding. The similarities to MIME are apparent, but it can be parsed more efficiently and it is conceived specifically for use with SOAP and web services.

In its bare essence, DIME is a mechanism for including multiple binary blocks of data within a single package. The blocks, or records as they are called in DIME, could contain any kind of data, including media items and SOAP messages. There is no restriction on the size or format of any of the data. As opposed to MIME, it is not necessary to know the length of the total data being sent when preparing a DIME package.

Figure 4.2 shows the record organization within a DIME message. A DIME message consists of one or more records with no restriction on the number of records in the entire message. Each record has a header (designated by the light sections at the top) and data (designated by the dark sections). Among other things, the record header includes various flags. These include a flag to indicate that a record is the first in the DIME message, and another flag to indicate that a record is the last in the DIME message.

The size of the data in each record can vary in length. The sequence of the data records is significant, and must be maintained over whatever channel is being used to transmit the DIME message. By using the begin and end message flags, DIME eliminates the need for an application to know the precise length of the entire DIME message before it starts to send it. When an application has completed transferring a DIME message, it simply sets the end message flag on the last data record.

The data record format is shown in Figure 4.3 in two parts, again with the headers in light and the data in dark. The portion of the headers above the dotted line is a fixed length of 64 bits. The first three bits shown in the first line is a bitmask that represents three different
Figure 4.2: Dime records form a segmented message

flags that describe the record. The first two bits are used to indicate the two flags that we saw in Figure 4.2. MB is the Message Begin Flag and ME is the Message End Flag. The third bit is the Chunked Flag (CF), which indicates that this record is part of a chunked data representation. Chunking allows us to split large DIME messages in an unlimited number of parts that can be transported (or streamed) independently.

The rest of the first 16 bits of the header is used to indicate the length of the ID field in the header. The ID field is a variable length field that provides a mechanism for identifying a particular record within a DIME message. For instance, a SOAP message in one data record of a DIME package may need to refer to an image file that is in a different data record of the DIME package. The SOAP message can refer to the image file by indicating the ID in the image file's data record. We will look at an example of this shortly.

The second 16 bits of the data record header describes the variable length Type field that follows the ID field. The Type field is used to associate the data in the data record with
some kind of type specification. The three-bit Type Name Format field indicates what kind of mechanism is being used to describe the data type. For instance we may want to specify a type like we do with the HTTP Content-Type header with a string like "text/html". The remaining 13 bits represented on the second line of Figure 4.3 indicate the length of the type field.

The third line in Figure 4.3 is simply the length in bytes of the data in the data record. The data length is a 32-bit field, so it specifies a maximum data size of 4 gigabytes. This is a potentially limiting restriction on the size of the data that may need to be packed into a DIME data record. Fortunately, DIME has an excellent solution for avoiding the data size limitation, in the form of chunking (as mentioned earlier).

WS-Attachments

DIME by itself is simply a mechanism for packaging a collection of arbitrarily formatted records of data. It sets no requirements on the contents of record payloads, what is contained in the ID fields or how a SOAP message might be encapsulated in a DIME message. WS-Attachments defines how multiple documents can be combined, how they reference one another, and how DIME packaging can be used to provide the attachment capability that can be used by Web services.

WS-Attachments specifies various constraints on the use of DIME, such as the indication that the main SOAP message must be contained in the first record of a DIME message. WS-Attachments also defines the use of the href attribute for referring to attachments from within the SOAP message. The href attribute is a URI that can be used to point to an HTTP URL if desired. However, WS-Attachments also defines the ability to refer to a specific DIME record using the ID field of the DIME record header. So if a secondary part of a DIME message has an ID of uuid:6FF57C24-74A1-426F-92D9-98861E105B4F, then the primary SOAP message part that
references such an attachment might look like the one given in Figure 4.4.

<?xml version='1.0' ?>
<s:Envelope
  xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:mes="http://example.org/message/response">
  <s:Body>
    <mes:responseMesssage>
      <messageText>Here is the data</messageText>
    </mes:responseMessage>
  </s:Body>
</s:Envelope>

Figure 4.4: A SOAP message in DIME references to an attachment using its message ID.

WS-Attachments also defines the details of how a compound SOAP message can be sent in an HTTP request. For the most part it is similar to simply sending the primary SOAP message part on its own, except that the HTTP Content-Type header must be set to "application/dime" and the body of the HTTP request is the DIME message instead of the SOAP message. This mechanism requires extra basic processing functionality at both the sender and receiver of DIME messages. A simple SOAP parser is no longer sufficient for this purpose. Figure 4.5 shows how this extra functionality is provided in .NET, to extract the main SOAP message from DIME streams.

4.2.3 Web Services Interoperability Organization

The Web Services Interoperability Organization (WS-I) is an industry founded initiative dedicated to enabling and promoting interoperability between web services. To achieve this, the WS-I produces four types of deliverables [56]:

- Profiles: profiles contain a list of named and versioned specifications together with a set of implementation and interoperability guidelines recommending how the specifications should be used to develop interoperable web services.

- Testing Tools: testing tools are used to monitor and analyze interactions with web services to determine whether or not the messages exchanged conform to WS-I Profile guidelines.

- Use Cases and Usage Scenarios: use cases and usage scenarios capture business and technical requirements for the use of web services. They reflect real-world requirements for web services solutions and provide a framework to demonstrate the guidelines described in WS-I profiles.

- Sample Applications: sample applications demonstrate the implementation of applications that are built from web services use cases and usage scenarios, that conform to a given set of profiles. Implementations of the same sample application on multiple platforms (using
different languages and development tools) allows WS-I to demonstrate interoperability in action and to provide readily usable resources for the web services developer.

The WS-I deliverables that are most relevant to our research are:

- WS-I Basic Profile (WS-Basic): The four fundamental specifications of web services (XML Schema 1.0, SOAP 1.1, WSDL 1.1 and UDDI 2.0) are brought together in the WS-I Basic Profile.

- WS-I Attachments Profile [55], which builds upon :
  - WS-Basic;
  - W3C’s SOAP Messages with Attachments (SwA, see Section 4.1.2) and defines some constraints (e.g. character encoding) on its use;
  - WSDL 1.1 Section 5.0 (which defines MIME bindings) but limits it to the SwA protocol.
WS-I did not yet provide the WS-I Attachments Profile and accompanying testing tools/usage scenarios at the time when we designed our architecture. A version 1.0 Working Group Draft of the profile became available in December 2003, when our architecture and technology was already in final stages of development. Until today, no other deliverables have been made available in the WS-I Attachments field.

4.3 Related Research

Recently a few research initiatives have emerged that are related to service oriented architectures for multimedia re-authoring.

Jia Zhang proposes a SOAP-oriented framework to support device-independent multimedia web services [60]. The framework introduces a mechanism for transporting large multimedia streams from and to web services, which offers an alternative to the SOAP Messages with Attachments and WS-Attachments proposals. While Zhang offers interesting ideas, the proposed research is unaware of multimedia-related standardization efforts, like MPEG-21. The authors have expressed that they have no intention of changing this unawareness. They consider the ISO multimedia standards to be ‘out of their league’ (yet another example of the clash of the ‘Internet’ world versus the ‘multimedia’ world). Also, it focuses on efficient streaming of media through web services, assuming that the required services have already been discovered and assigned. The framework meets the requirements of our problem statement with regard to enabling streaming of multimedia from and to Web Services, albeit by means of a proprietary packaging mechanism. It falls short, however, on most other aspects, such as the adaptation of complex multimedia (consisting of multiple media objects) and the presence of Broker services.

A high-level distributed system architecture is proposed by Roy [32]. It focuses on load balancing and resource distribution related to transcoding media in networks. A number of resource monitoring schemes for transcoding services are introduced. However, the authors do not provide information on the standards and mechanisms that are used to build the infrastructure. We have learnt that they do not respond to requests for more information on the topic. From their publications we can derive that they probably only developed resource management and load balancing algorithms. Presumably, the architecture they claim to have developed is not worth a detailed publishing.

Digestor [7] is a software system that provides automatic re-authoring of web pages. It implements techniques for transforming various elements in web pages to suit particular client browser window sizes. A very simple example of the transformations they propose is the removal of white spaces in text by placing numerations and lists on a single line. The authors place the Digestor system at the http-proxy server, without specifying communication mechanisms or providing an architecture. Their results are interesting, and could benefit greatly from a service oriented architecture that offers more flexibility than their current implementation offers.

Bellavista [6] describes an active middleware to manage quality-of-service for streaming Video-On-Demand systems. The algorithms he proposes could very well help in building an operational service oriented architecture for video-processing. He focuses on the negotiation phase of video delivery at the level of video streaming servers. The architecture he proposes,
however, is very tightly coupled and fixed at design time. We consider it therefore to be unsuitable for large scale deployment in a distributed manner over the Internet.

TranSquid [25] provides caching algorithms at transmoding services, specifically for e-commerce environments. As the caching algorithms are the main focus of research, they use basic HTTP mechanisms and TCP-socket communication to build the infrastructure. Hence, their architecture lacks many of the great features provided by web services. The developers of TranSquid have not taken streaming multimedia into account, yet focus on a limited set of possible adaptations on simple media.

Xiaolan Zhang [59] describes a translation proxy for connecting different proprietary players and servers. The client, server and proxy communicate using proprietary extensions to the Real-Time Streaming Protocol (RTSP). The proposed proxy layer is applicable to the adaptation of RTP-streams with video and or audio data only. Its location is determined and fixed at design-time. As we will indicate in Section 8, we consider an RTSP-based approach to be insufficient for our problem statement.

The multimedia personalization framework of Boll probably offers the most complete and elegant solution to our problem statement [8]. It employs recent web service technologies to provide services in the network that tailor multimedia presentations to specific user preferences. The user preferences it can take into account are: available bandwidth, player, personal preferences of the viewer, location, etc. It is built specifically for the adaptation of SMIL presentations that contain still images, text and streaming videos. The adaptation it performs, however, consists only of transformations of SMIL and selection of particular multimedia items on (streaming)servers. For instance, when targeting a PDA, small pictures and brief text fragments, that already reside somewhere on a server, are referred to from the generated SMIL presentation. As such, no ‘real’ run-time adaptation and transport of multimedia occurs. It is assumed that the required multimedia items are available before the personalization occurs. One of the most important drawbacks of this system is the fact that it can not take into account changing preferences and/or circumstances while a presentation is being consumed. Moreover, only the precise scenarios that were anticipated when the media was prepared can be handled, since no re-adaptation of multimedia items can occur.

In July 2004, Stephane Coulombe and Guido Grassel of Nokia Research Center have published an article in IEEE’s Communication Magazine, titled ‘Multimedia Adaptation for the Multimedia Messaging Service’ [15]. The authors provide an adaptation architecture for the Multimedia Messaging Service (MMS) on mobile phones. It is based entirely on the WAP and UAProf standards, and thus not directly applicable to our problem statement. Still, the authors have provided an elegant architecture which is highly relevant and related to our work.
Chapter 5

A Service-Oriented Architecture for Multimedia

The previous chapters provided a context (Chapter 2) and background information (Chapter 4) for our research. In this chapter, we propose a service oriented architecture to target the problems we have introduced in Chapter 3. An evaluation of the advantages and challenges that are brought about by this architecture is provided in the next chapter.

Figure 5.1: The web services at the proxy server adapt the Content Creator’s multimedia and place it on the consumer’s Home Gateway.

Figure 5.1 shows an abstract overview of the different aspects that are detailed in this chapter. We introduce an architecture (Section 5.1) that employs several multimedia web services (Section 5.3), cooperating to tailor content creators’ multimedia to clients’ environments:
the Download Manager, Transmoder and Broker services. Another multimedia web service runs on the Home Gateway, providing the Transmoder services with a web service interface to provide their results to the client.

As standard SOAP-based web services are not sufficient to solve the problems stated in Chapter 3, we specify multimedia-specific extensions in Section 5.2.

We end this chapter by providing two case studies in Section 5.4. Whereas the first case study is described in detail, showing the mechanisms involved, we have provided a more abstract description of the second case study.

5.1 Architecture

Figure 5.2 illustrates our service oriented architecture in more detail. We start this chapter by providing a walkthrough of the architecture.

Imagine a consumer that wishes to consume a particular multimedia presentation with his player, in a form that is optimized for its capabilities and context. To achieve this, he contacts a Broker service and provides the URI of the presentation. It may also provide parameters describing its current capabilities. As a capability description language, MPEG-21 DIA Usage Environment Description Tools, CC/PP or basic textual descriptions (as in our experiments) can be used. This set of parameters should contain at least one of the MIME-types the client player accepts. Naturally, the amount of parameters that a client provides influences the way in which a Broker service can provide optimal presentations and improve the experience for the client. The URI that the client provides, points to a particular multimedia presentation on a content provider server. A content provider server may be anything ranging from a small web-enabled digital camera to a broadcaster’s web farm.

Acting upon a request, the Broker service contacts Download Manager services to provide the required multimedia items in a predictable and standardized way. These Download Manager services provide abstraction over the physical location, transport protocol (e.g. FTP) and (to a lesser extent) format of multimedia items. To achieve this abstraction, they may need to prefetch items from their original location, using the required protocol or retrieve them from local cache when available. Download Manager services need to provide two web service-based methods that enable other services to access multimedia items:

- **Message-based**: the requested items are attached to a single message that is sent in response to a web service call (DIME over HTTP).
- **Stream-based**: the requested items are prepared for streaming on a particular TCP/IP or UDP/IP socket. The handle of this socket is returned in a response message of a web service call (DIME over TCP).

For a detailed description and comparison of these mechanisms, please refer to Section 5.2.1. If Download Manager services offer multimedia items in a predictable way, using a popular, widespread, generic and high quality format, the re-authoring process may be greatly facilitated later on. However, this is not a hard requirement for the success of our architecture.
While the Download Manager services may be in the process of retrieving the multimedia items from content creators or from local cache, the Broker service selects and allocates various Transmoder services. Item Transmoder services receive from the Broker service a URL to the
Download Manager service where they can obtain multimedia items using the message-based mechanism. Item Transmoder services are fairly simple to build and offer an ideal mechanism for transmoding operations on multimedia items that do not require streaming, such as still images, small sound clips, etc. Streaming Transmoder services, on the other hand, receive a reference to a TCP or UDP socket on which they can contact a Download Manager service to receive multimedia items with stream-based mechanisms. Such services are particularly useful for large multimedia items of which parts can be transmoded even before the entire item has been received. Typical examples of such items are large video streams and audio streams.

The Broker provides Transmoder services with a reference to a Home Gateway service, so they know where they have to publish their results. Home Gateway services host (adapted) multimedia items. They provide methods to place multimedia items either in streaming or message-based mode.

When the Broker service has received notification from a Transmoder service that (a first version of) the multimedia presentation is uploaded to a Home Gateway (using a message-based or stream-based mechanism), it returns the location of this presentation to the client. Finally, the client uses its preferred protocol (e.g. HTTP or RTP) to retrieve the presentation from the Home Gateway.

Note that the actual location of the media services is not specified in our architecture. We deliberately choose not to do this because an important merit of the application of service-oriented architectures is the fact that the deployment location of services need not be determined and fixed at design time.

This means that each service in our architecture can be deployed at various nodes in the network involved. For example, a set-top box in a home network may provide a hardware accelerated MPEG-2 decoding service. When the service on this box becomes unavailable (e.g. because the box needs to perform MPEG-2 decoding for its own application), a similar service that runs on an ISP server, or even a non-accelerated service that runs on a PC in the network, may be contacted to take over the processing. The Universal Description, Discovery and Integration (UDDI) protocol is a IBM/Microsoft initiative that aims to facilitate abstraction of service location [36]. UDDI and run-time selection of web services are currently under high debate at W3C. Both technologies are very likely to undergo massive changes in the near future, to incorporate semantic information of web services in the discovery and selection procedures. We have chosen not to include UDDI technology in our architecture, since the static (compile-time) discovery it currently provides, has little added value in our context. For a brief discussion on where to deploy media services, please refer to Section 3.2.1.

5.2 Extending Web Services for Streaming Multimedia

5.2.1 Transport of Multimedia Data

As indicated in Section 4.1.2, SOAP, the preferred standard for exchanging messages with web services, suffers from a few serious drawbacks when applied for transport of multimedia data.

That is why we employ the Direct Internet Message Encapsulation (DIME) (see Sec-
CHAPTER 5. A SERVICE-ORIENTED ARCHITECTURE FOR MULTIMEDIA

4.2.2 Using DIME, our multimedia data can be sent together with its metadata in a SOAP message. The metadata in a SOAP message can contain various information on the multimedia data. This can consist of information about encoding, description, resolution, preferred relative location, a space of possible adaptation strategies as described by the multimedia provider, etc. Providing metadata is commonly regarded as a key requirement for the successful processing and authoring of multimedia [23]. The ability to send metadata, in a separate description, along with multimedia in an open and standardized way is much more elegant and safe than sending that information on a custom basis.

We use DIME for two purposes: to send SOAP messages (that contain metadata) with multimedia attachments over HTTP and to stream large sets of multimedia data, together with their SOAP-packaged metadata, over TCP and UDP. Figure 5.3 shows a sample of the first: a SOAP message that is sent to the PlaceDigitalItem web service of the Home Gateway (namespaces were removed for clarity). It contains a DIME record with the SOAP message, and a second DIME record with the attached data. These records are sent in their entirety using HTTP-POST in a single web service call.

In Figure 5.4 we show what happens in the second scenario: a SOAP message is sent as a DIME stream to the socket that was provided by the StreamDigitalItem web service of the Home Gateway (namespaces were removed for clarity). Next to the first DIME record with the SOAP message, a second DIME record containing the attached data, is split in multiple chunks (only first and last are shown). These DIME chunks are sent individually over a TCP socket connection.

Figure 5.3: A SOAP message with a png image attached consists of two DIME records.
5.2.2 Lifetime Management

When streaming large amounts of multimedia data with DIME, the standard web service mechanisms are not sufficient. The life-time of a web service is limited to the duration of its web method call, while the streaming itself may take a long time. It may even take several hours in the case of real-time video streaming. Therefore, we introduce threads that run in the memory space of web services and handle the streaming and transmoding of the multimedia. Figure 5.5 shows how we create a thread for that purpose in a web service using C# for Microsoft .NET [2]. Such a thread typically waits for another component to connect to its specific socket, and communicates using SOAP messages over a DIME stream on that port.
[WebMethod]
public int getDimePort(String uri)
{
    // Prepare a DIME streaming-server over TCP
    server = new TcpListener(0);
    server.Start();

    // Start a new thread to handle the client, runs in the same context
    ThreadPool.QueueUserWorkItem(new WaitCallback(streamDigitalItem));

    // Return the port on which the DIME-server is waiting.
    IPEndPoint ipend = (IPEndPoint) server.LocalEndpoint;
    return ipend.Port;
}

void streamDigitalItem(Object stateInfo)
{
    TcpClient client = server.AcceptTcpClient();
    NetworkStream myNetworkStream = client.GetStream();
    DimeWriter dimeWriter = new DimeWriter(myNetworkStream);
    // Send data to 'dimeWriter' in (chunked) records
    ...
}

Figure 5.5: The web service int getDimePort(String uri) creates a thread, streamDigitalItem, that runs in the same context, streaming media with DIME over TCP.

5.3 Architecture Components

5.3.1 Download Manager Service

Our Download Manager service may reside at the proxy-server and provides for an abstraction of the protocols used by content creators. It offers the multimedia, which it has downloaded over any protocol it is aware of, in a uniform way through web services attachments or a continuous DIME-stream, as explained in the architecture walkthrough of Section 5.1. Figure 5.6 illustrates the functionality of a Download Manager service. Upon request of a Broker service, it may start pre-fetching multimedia data from a content provider even before a Transmoder service requests it, caching the data in its local memory.

5.3.2 Home Gateway

The Home Gateway can be a device that is placed by an Internet Service Provider (ISP) in the consumer’s home. It acts as a gateway between the consumer’s Local Area Network (LAN) and the ISP. As such, it plays a key role in offering a guaranteed quality of service when the consumer accesses information and multimedia on the Internet. Multimedia items form a
significant portion of the data that is placed on the Home Gateway, which caches frequently used data and offers a maximum quality-of-service for multimedia consumption.

In our architecture, the Home Gateway is used to store and cache transmoded multimedia presentations so they can be consumed locally. The mechanisms it employs to achieve this functionality are illustrated in Figure 5.7. Similar to the Download Manager service, it offers two types of transport mechanisms: message-based and stream-based. The multimedia items that are placed on the Home Gateway, by means of messages or streams, are stored in local memory for consumption. The local memory, where items are stored, can be either volatile or non-volatile (disc) memory.

A Transmoder service that places an item on the Home Gateway, using one of the web service interfaces, may indicate a relative location for that item. This provides Transmoder services with the ability to prepare complex multimedia documents that aggregate (and refer to) multiple multimedia items. The case study in Section 5.4.1 illustrates this mechanism.

When an item is placed on the Home Gateway without indicating a relative location, the Home Gateway decides upon this location (and naming) autonomously. The chosen location
and name of the item is returned to the calling service in the response message of the web service call.

After receiving the location of a multimedia item, which is a finished presentation, from the Broker (Home Gateway URI + relative location of the item), a client can retrieve this item from the Home Gateway using its preferred standard protocol (e.g. HTTP or RTP).

When the consumer’s environment characteristics (e.g. available bandwidth or processing power) changes during a presentation session, the Home Gateway selects from a number of strategies to adapt the properties of the multimedia presentation:

1. If the changes remain within a given preset range, scalable multimedia formats are likely to be able to handle the changes. This is the easiest strategy, only applicable to scalable formats.

2. A (non-scalable) multimedia item that was delivered by an Item Transmoder can be re-requested in a different form from the Item Transmoder that delivered it, using the reference that the Item Transmoder gave when delivering the item, as will be explained in Section 5.3.3. This strategy is also employed when a multimedia presentation is delivered in multiple steps that require iterations. An example is given in the case study of Section 5.4.1.

3. When a Streaming Transmoder service is providing (a part of) the multimedia presentation, the bi-directional DIME channel that exists between the Home Gateway and the Streaming Transmoder service can be used to indicate the changes that occurred, enabling the Streaming Transmoder service to adapt the multimedia stream it delivers. This bidirectional channel can also be employed for presentations that consist of multiple steps.

4. When the previous strategies are not applicable, the Home Gateway needs to contact a new Broker service specifically for the new requirements of this multimedia item.

5.3.3 Transmoder Service

Transmoder services form the distributed engine of our architecture. They are responsible for a range of transmoding tasks, yet each Transmoder service may support only a limited transmoding functionality. Such tasks can be fairly simple, e.g. rescaling a PNG image from one resolution to another. They can also be quite complex, e.g. extracting key frames from a video stream and sending them as a series of PNG images for consumer devices that do not support streaming video. Transmoder services can be very specialized, focusing on the tasks that they perform best without having to offer other transmoding functionality.

The services place the information they produce directly on a Home Gateway, a reference to which was given by the instructing Broker service. Apart from the multimedia item and metadata, the information placed on the Home Gateway may also contain a reference to the generating Transmoder service itself, so the Home Gateway can access it at a later moment, e.g. when a slightly altered version of the multimedia item is required.

We introduce two families of Transmoder services:
• Item Transmoders receive requests of a Broker service to transmode single multimedia items. They retrieve and provide the items they handle using DIME over HTTP, as SOAP messages with attached multimedia data.

• Streaming Transmoders connect to given Download Manager services and Home Gateways using DIME over a streaming protocol. They continue to run as a thread for as long as the multimedia data continues to stream to the consumer. Streaming Transmoders may need to employ quite complex buffering and scheduling schemes, since multimedia streams have to be retrieved, transmoded and provided at an appropriate pace.

Figure 5.8: Simplified UML class diagram of Transmoder services.

Figure 5.8 shows the UML class diagram of the message-based Transmoder services infrastructure. At the top of this picture we have placed the fundamental interface, MessageBasedTxM, which any message-based Transmoder service has to implement. It declares three abstract methods:

1. `String getFileExtension(String mimeType)`: Every instance of a class that implements this interface needs to be aware of the MIME-types and file extensions of the multimedia items
it can handle. Naturally, many Transmoder services are able to process/generate multiple MIME-types (e.g. the Image-to-Image Transmoder service supports seven distinct MIME-types). However, each Transmoder service is instantiated to target a particular MIME-type.

2. String transModeDigitalItem(String dlManager_uri, String hg_uri, String digitalItem_uri, String outputMimeType): This is an abstract method declaration which is exposed as a web service declaration. This method gets called by Broker services to perform the actual transmoding. No adaptation options (such as Size) are given, so a default adaptation is applied.

3. String transModeDigitalItemWithSize(String dlManager_uri, String hg_uri, String digitalItem_uri, String outputMimeType, Size size): This is an abstract method declaration which is exposed as a web service declaration. This method also gets called by Broker services to perform the actual transmoding, with a 2-dimensional Size as adaptation parameter.

The DimeMessageBasedTxM class extends the MessageBasedTxM interface with methods that perform DIME-specific operations, such as retrieving/placing items from/in DIME records. It also implements the transModeDigitalItem and transModeDigitalItemWithSize methods, in accordance to the Template Method design pattern \[19\]. This method provides an algorithm that retrieves a multimedia item in a stream using the fetchDigitalItem method, calls the transMode method to adapt the multimedia item, and sends it to the given Home Gateway location using placeDigitalItem. The protected helper method transMode is implemented in deriving classes, the others are implemented in the DimeMessageBasedTxM class directly.

The SessionTxM and ItemTxM classes derive from the DimeMessageBasedTxM and partially implement the string transMode(System.IO.Stream inputStream, String inputMimeType, String parameters, Size maxSize) method. The difference between the SessionTxM and the ItemTxM class lies in the fact that an ItemTxM simply places its results on the given Home Gateway service whereas a SessionTxM provides extra state information and call-back functionality to the Home Gateway service. A SessionTxM service can be called back to perform subsequent iterations on the media presentation it generates. A Home Gateway can execute the call-back by calling the web service method getNext on the TransModer service. Since the SessionTxM sends its state information along with the adapted item to the Home Gateway, both Transmoding service implementations are stateless and can be invoked as standard web services.

At the bottom of the figure, we show some Transmoder services which we have implemented:

XMLtoSMIL employs a XSLT processor to generate a SMIL presentation from a custom XML-description.
SMILtoHTML also uses a XSLT processor, generates individual web pages from a SMIL presentation.
ImgtoImg wraps the Microsoft .NET image processing library to enable transcoding between multiple image formats.
SVGtoImg executes the Java-based BATIK code to generate PNG or JPG images from SVG.
These services are used in the case studies of Section 5.4.

5.3.4 Broker Service

A Broker service instantiates and selects Download Manager and Transmoder services upon a client’s request. The selection it makes may be derived from a track history it has stored in an internal database. A Broker service decides upon the selection of specific Transmoder services by analyzing a client’s capabilities and network context. This analysis results in a working space for Transmoder services, determined by the capabilities as indicated by a client. In our experiments, clients use a simple textual description to advertise their capabilities. Using this description, a Broker service determines which formats (encoding, resolution, etc.) can be consumed by a client. It instantiates the appropriate Transmoder services to provide the multimedia in the most suitable descriptions and formats.

Figure 5.9 shows a simplified extract of the Broker service source code in C# for Microsoft .NET. It illustrates the selection process of a suitable Home Gateway, Download Manager and Transmoder service for the re-authoring of a multimedia item (denoted Digital Item for compliance with MPEG-21). Only the selection algorithm for the SVG-to-Image service is shown. When the source media item can be processed and the adaptation to the requested output MIME-type can be performed by the SVG-to-Image service, the service URI is set to a relevant (i.e. nearby) location of the SVG-to-Image service. This kind of polymorphism with web services is made possible thanks to the inheritance structure of Figure 5.8. Indeed, the web service /emphfacade that every Transmoder service exposes to the Broker service is identical to the /emphMessageBasedTxM. Next to this uniform interface, however, some Transmoder services may expose a more complex interface to other Transmoder or Home Gateway services to enable more complex interactions.

Once the services are started, the client can start consuming the multimedia and the Broker service becomes available for other client requests.

5.4 Case Studies

To illustrate the added value of our architecture, we have applied it to two distinct case studies:

- Viewing SMIL presentations with a standard Web browser (Section 5.4.1).
- Generating SMIL presentations from an XML-based description (Section 5.4.2).

5.4.1 Viewing SMIL presentations with a Web Browser

The problem statement of this case study is described in Section 3.2.2. We consider the case where consumers wish to view SMIL presentations from within their web browser, without having installed plug-ins that offer support for SMIL. Since the size of web browser windows can vary considerably, we also want to adapt the generated web pages to the window size in which they will be rendered.
private DimMessageBasedTxMwse service = new DimMessageBasedTxMwse();
...

[WebMethod (Description="Requests a DI to be adapted to the client platform. Returns the URI of the 'first' result")]
public string getDigitalItem(string uri, string targetMime, Size size)
{
    string result = default_result;
    // Find the Home Gateway of the client, based on IP address.
    string homeGateway = ...;
    // Find a suitable Download Manager, based on URI of requested item.
    string downloadManager = ...;
...
    // Determine whether the SVG-to-Image service is applicable,
    // based on target MIME-type and the source URI’s extension.
    {
        // output limited to "image/jpeg", "image/png", "image/tiff"
        string[] svg2image = {"image/jpeg", "image/png", "image/tiff"};
        foreach (string type in svg2image)
        {
            if (type == targetMime)
            {
                service.Url = "http://mpeg03/TransModer/SVG2Image.asmx";
                result = service.transModeDigitalItem(downloadManager,
                                                        homeGateway,
                                                        uri,
                                                        type,
                                                        size);

                return result;
            }
        }
    }
...

Figure 5.9: Simplified extract from a basic Broker service implementation.

The SMIL presentations we wish to adapt consist of still images and pieces of text that are updated after given time intervals. An example of such a presentation (with mp3 audio added) is shown in Figure 5.10. Using this functionality, nice slideshow-like presentations with images and accompanying textual descriptions can be created.

The images that are referred to in the XML-based SMIL presentations are PNG and Scalable Vector Graphics (SVG). SVG images are vector graphics described with XML documents [45]. SVG descriptions may contain references to other SVG descriptions, thus enabling nesting of external images when rendering the images. The textual descriptions in our SMIL presentations are stored in small text files, which are also referred to from within the SMIL documents.
We use a popular web browser with support for HTML and JavaScript as a client. The only way our web browser can display a SMIL presentation directly is by showing the source XML document, since at this moment no popular web browser is able to interpret and render SMIL presentations. Our approach is based on the transformation of the SMIL presentation to a presentation that consists of HTML pages with a similar layout.

In the remainder of this section, we provide a detailed walkthrough of the case study. The walkthrough is illustrated by the sequence diagrams in Figure 5.11 and Figure 5.15.

When we wish to consume a particular SMIL presentation with a web browser, we start by contacting a Broker service. We can do this conveniently with our web browser, since most web service platforms generate a simple HTML page from a WSDL description, when requested by a browser. Figure 5.12 shows this web page for our Browser’s getDigitalItem service.

When we invoke the service with the requested arguments (using HTTP-POST), the Broker selects appropriate Download Manager and Home Gateway services (in this case, both run
on the local machine. It then matches the presentation’s and client’s MIME-types (text/smil and text/html respectively) to select an appropriate Transmoder service. The Transmoder service it selects in this scenario is our SMIL-to-HTML service.

Since the SMIL-to-HTML service did not yet receive a client capability description (i.e. screen size), it provides an HTML page with JavaScript code that requests a new page, sending along the capability description in HTTP-POST (Figure 5.13). This page is thus the first resulting HTML page of the SMIL-to-HTML Transmoder service. It is uploaded to the Home Gateway service, together with the indication (by setting a flag in the web service call) that a call-back mechanism is required.

The SMIL-to-HTML service returns the relative path of the generated web page to the Broker service. The Broker service then returns an XML document with the location on the Home Gateway (Home Gateway’s absolute URI + relative path of the results) where the adapted presentation can be found, as Figure 5.14 shows.

When the user wishes to view the presentation, he selects the location which he obtained from the Broker service in his browser. The browser immediately receives the first generated HTML page (Figure 5.14), which causes it to request the first actual slide of the presentation, adapted to its current width and height.

As mentioned earlier, the Home Gateway service provides a mechanism to translate a request for a new page to a web service call-back to the SMIL-to-HTML service. This server-side mechanism, shown in Figure 5.16, is written specifically for the ASP.NET platform. However, the basic mechanism is also valid for other server-side systems (e.g. PHP and Java Server Pages). In our implementations, this mechanism resides on the Home Gateway. It is inserted in the hosted web page that is uploaded by a Transmoder service, when the call-back flag is set.
When called, the code retrieves the next HTML page (again with the JavaScript mechanism of Figure 5.13) from the SMIL-to-HTML service and forwards it directly to the client.

Since our SMIL presentation refers to other media items such as SVG and PNG images, other Transmoder services need to be called in order to adapt these items to the generated presentation. For this purpose, the SMIL-to-HTML service requests the transmoding operation from a Broker service, indicating the required MIME-type and size and source media item. The same happens when SVG images contain embedded images, where SVG-to-Image services call upon other services to perform the transmoding of the embedded images.

Experimental Results

A screenshot of the resulting presentation in a web browser is shown in Figure 5.17. As a reference, Figure 5.18 shows the original SMIL presentation in Apple’s QuickTime Player.

The only overhead that our architecture poses on transmitted information, is caused by the DIME and SOAP headers. Especially when dealing with multimedia data, this overhead turns out to be fairly small. It is not possible, however, to indicate the average overall transmission overhead, since that is highly case-specific.
CHAPTER 5. A SERVICE-ORIENTED ARCHITECTURE FOR MULTIMEDIA

Figure 5.13: The first HTML page generated by the SMIL-to-HTML service. It requests a new page and conveys a basic capability description (using the width and height parameters).

Figure 5.14: Web page view on the response of the Broker service.

In the specific scenario that we implemented, with a transmission of PNG-images of 1280x1024 pixels, the overhead proved to be less than 3%. However, when the source SMIL-presentation consists entirely of SVG-images, the amount of transmitted data (including PNG-versions of the images) turns out to be orders of magnitude higher than the source.

This clearly supports our statement of Section 3.2.1 where we indicate that the ideal location for many adaptations to occur is in the network, eventhough this can not be generalized to all scenarios.

Network packet loss has a limited impact on our architecture, as the entire communication runs on top of HTTP. If retransmission of a message is required due to a lost packet, that portion of a single (DIME’d) SOAP message needs to be re-transmitted. A similar retransmission of
a portion of one item (e.g., PNG-image) would have been required if our architecture was not used.

5.4.2 Generating SMIL presentations

To further illustrate its added value, we have applied our architecture to scenario in which a content creator places a small XML document describing a simple presentation on a server. The

Figure 5.15: Sequence Diagram of a page retrieval in the SMIL2HTML adaptation.
CHAPTER 5. A SERVICE-ORIENTED ARCHITECTURE FOR MULTIMEDIA

```csharp
<%@ Page Language="C#" %>
<script runat="server">
...
void nextSlide(object sender, EventArgs e)
{
    // Load NameValueCollection object
    NameValueCollection coll = Request.QueryString;
    // Get values of the keys that interest us
    int slideNr = Int32.Parse(coll.GetValues("slideNr"));
    int width = Int32.Parse(coll.GetValues("width"));
    int height = Int32.Parse(coll.GetValues("height"));

    String place;
    // Instantiate a proxy class for the SMIL2html service
    Demo.SMIL2HTML service = new Demo.SMIL2HTML();

    contents = service.getSlide(contextInfo, slideNr, width, height);

    using (System.IO.StreamReader sr = new System.IO.StreamReader(contents))
    {
        String line;
        // Read and send lines from the service's contents until the end is reached.
        while ((line = sr.ReadLine()) != null)
        {
            Response.Write(line + "\n");
        }
    }
}</script>

Figure 5.16: The ASP.NET code that calls a SMIL-to-HTML service when a new slide is requested.

A consumer wishes to view this presentation on a personal computer using the x-smiles media-player [29]. Through its Home Gateway, it contacts a Broker service and provides the url of the presentation and a basic CC/PP description of its environment (player type and version, supported formats and screen resolution). The Broker decides upon the application of specific Transmoder services. In this scenario an XML-to-SMIL2 Transmoder (performing XSL transformations [43]) and a tiff-to-png Transmoder will be applied, since x-smiles requires a SMIL description of the presentation and it does not support tiff bitmap images [46]. Both the sound clips and textual descriptions can be placed on the Home Gateway without transmoding if their required bandwidth and encoding are suitable for the consumer’s environment. This action can be performed by a simple Transport service, which is a message-based Transmoder service that performs no actual transformation on the items.

The Broker returns the web service call and indicates at what location on the Home
Gateway the appropriate presentation, shown in Figure 5.10 can be found. This presentation has been adapted as well as possible to the consumer’s environment. Since a Broker service is only responsible for the presentation at initialization time, it may indicate that a presentation is ready for consumption even before all multimedia items have been placed on the Home Gateway, assuming that these items will be present when needed. The Home Gateway is responsible for re-requesting multimedia items when they are not present in the correct format during a presentation session.

We have also applied this scenario to an Apple QuickTime player, which offers no support for SMIL 2 and ogg vorbis [1]. Therefore, the Broker service applies an XML-to-SMIL Transmoder and ogg-to-mp3 Transmoder to the presentation and sound clips respectively, since these formats are supported by the player [12].

**Experimental Results**

Figure 5.20 shows a screenshot of the x-smiles player, playing a SMIL presentation that is generated by our architecture. In this version, the selected tiff-to-png Transmoder service does
not support image rescaling. Images that are too large for the player’s display region are shown only partly. The textual description of each image is printed in the white text-box underneath the image-region.

The same presentation is shown in Figure 5.21, but here we selected a tiff-to-png Transmoder service that does support rescaling of images. The black rectangle was drawn on the figure after taking the screenshot, to indicate the portion of the image that was visible in Figure 5.20.

Naturally, rescaling of the images can occur at the client, but that results in a significant overhead in the amount of data that has to be sent to the client. In our experiments, the image of Figure 5.21 has a size of 13 kb, while the image of Figure 5.20 is about 64kb. If the image would be rescaled at the client, this would result in a bandwidth overhead of $\frac{64-13}{13} \approx 400\%$. 

Figure 5.18: Screenshot of a slide from the original presentation, played in QuickTime.
<?xml version="1.0" encoding="utf-8"?>
<content>
  <item>
    <image src="media/1.tif"/>
    <text src="media/1.txt"/>
    <audio src="media/1.ogg"/>
    <duration>5s</duration>
  </item>
  <item>
    <image src="media/2.tif"/>
    <text src="media/2.txt"/>
    <audio src="media/2.ogg"/>
    <duration>7s</duration>
  </item>
  ...
</content>

Figure 5.19: An XML description of a presentation consisting of two tiff-images and accompanying text and ogg vorbis audio.

Figure 5.20: The generated SMIL presentation without adaptation of the images' resolution. Large images do not fit in the image region and are shown only partly.
Figure 5.21: The generated SMIL presentation with adapted resolution of the images. The black rectangle shows the area that would have been visible without resolution adaptation.
Chapter 6

Overview of our Contribution and Rationale

In this chapter, we illustrate the uniqueness of some of our results and discuss the rationale behind important decisions. Section 6.1 briefly describes the basic idea that this research is based upon. We believe it is a new, viable and good idea that will continue to gain popularity. In Section 6.2, we continue with an overview of some unique technologies that we developed in the context of this research, together with the rationale behind them. Finally, Section 6.3 concludes with a comparison of our approach to the related research initiatives of Section 4.3.

6.1 The Concept of Multimedia Adaptation with Web Services

The exchange of multimedia on the Internet raises the demand for adaptation and management at various places in the network. Web services are inherently well suited to operate in such a distributed and heterogeneous environment. We have devised techniques for web services to perform multimedia adaptation at key locations in the network. Especially when multiple nodes in the network are involved in the adaptation or management process, web services can provide a significant added value. The main advantages of using web services for this purpose lies in their supporting techniques for security, transaction management and universal discovery, availability and flexible invocation. Web services that adapt multimedia can be deployed and employed using standard mechanisms anywhere, at anytime.
6.2 Unique Contributions

6.2.1 Streaming SOAP over DIME

In Section 4.2.2 we introduced DIME as a proposed standard for sending arbitrary data along with SOAP messages. Next to the ‘standard’ DIME over HTTP messaging mechanism, we have introduced the notion of streaming multimedia with DIME records over TCP in Section 5.2.1. Using our streaming mechanism, web services can be used to start possibly long-lasting processing threads that communicate over DIME, as explained in Section 5.2.2.

One might argue that streaming DIME over TCP obstructs loose coupling, one of the fundamental promises of web services. However, the fact that this type of communication in our architecture is not loosely coupled is not simply a consequence of the mechanism itself. The tight coupling that we see in this kind of applications is inherent to multimedia-streaming systems. Trying to build a distributed architecture with services that cooperate constantly while sending large amounts of data across using loose coupling does not seem to make sense.

We propose to use the original web service mechanisms (SOAP or DIME over HTTP) for their specific purpose – the invocation of web services; and we provide a mechanism to stream multimedia (with DIME over TCP/UDP) between (long-running) processes. Please note that since multiple definitions of the concept ‘web service’ are in use, depending on the definition used, the processes that stream DIME may or may not be considered web services. One thing is certain, they are not loosely coupled modules that operate on a single ‘method call’ with SOAP messages.

6.2.2 The Use of DIME for ‘Enriched’ Media Transport (Messaging)

An important contribution of our research lies in the enriched transport of multimedia. Both our Download Manager (Section 5.3.1) and Home Gateway (Section 5.3.2) provide web service enhanced mechanisms for uploading/downloading multimedia items.

Currently, the available protocols for file transfer (e.g. FTP and HTTP) provide the only standard mechanisms to upload/download multimedia items such as still images. The notion that the management of (semantic) meta-information of multimedia and other presentations is crucial is becoming hugely popular with the advent of Semantic Web technologies and MPEG-7. The presence of semantic information of multimedia (e.g. encoding complexity, recording location, contributors, contents of the media, etc.) can greatly

- enhance the multimedia experience: when viewing a video clip with an interview of musicians, information such as the name of the musicians, their recordings and the location of their website can be used to provide some interactivity to the user.
- alleviate and optimize multimedia processing: the availability of information on the complexity of an audio clip (e.g. silent passages) can help an encoder to nicely balance the amount of bits for each part in the clip.
Yet, the current file-transfer protocols offer no support for the joint transport of meta-information and the multimedia presentations/items it refers to. As a consequence, the distribution of semantic meta-information of multimedia items is a separate operation. As such, the meta-information runs a high risk of becoming unreachable when the multimedia is transported multiple times. After all, there is no guarantee that the meta-information is placed at the correct location and within the permitted time when a multimedia item is uploaded with a simple file-transfer protocol, such as FTP.

The DIME-based messaging mechanism we have developed in Chapter 5 may provide an elegant solution for the joint transport of multimedia and related meta-data. It allows to send any XML-based information along with the multimedia data in a SOAP envelope. The resulting transport mechanism ensures that the metadata is kept together with the multimedia data it describes.

### 6.2.3 The Use of DIME for ‘Enriched’ Media Transport (Streaming)

At first sight, a logical continuation of Section 6.2.1 and Section 6.2.2 appears to be to transport both SOAP and multimedia data in a single DIME stream. Such a mechanism might seem to extend the advantages of Section 6.2.2 to streaming media, such as large audio and video clips. We have experimented with such a mechanism, adding time-stamps to the DIME records to enable real-time streaming and rendering of the data.

We examined our application of DIME and compared it to related literature. The overhead that DIME causes when used stream packaged multimedia data proves to be intolerable.

The most widely supported standard for streaming of multimedia is rfc3550, RTP [33]. RTP is supported by many major streaming media providers and players, such as RealNetworks and Apple (Quicktime). It is a fairly simple standard that builds upon the UDP, by adding a few time-stamps to the header. Multimedia data has the unique property that data may be discarded in the network in the case where it would arrive too late for rendering at the client. This requirement differs dramatically from more traditional data transport, where reliability is crucial, even at the cost of transmission speed. RTP supports the multimedia transport requirements nicely by using the datagram-oriented UDP in combination with time-stamps that can be processed at any node in the network. The overhead on transported multimedia data caused by RTP is highly dependent on the usage scenario (underlying transport protocol, characterization and format of the multimedia, etc.). In practice, however, it rarely exceeds 5 percent.

We devised a mechanism to stream multimedia data, packaged together with its SOAP-ed metadata in DIME records, over TCP and UDP. As with RTP, calculating an average overhead of DIME’d multimedia streaming is not possible, as this overhead is also highly usage scenario dependent. Since our approach is based on DIME, the overhead of XML encoding of the binary data itself ought to remain limited. This distastates us from more traditional mechanisms where binary data is often base-64 encoded prior to inclusion in XML (and SOAP) documents, causing an overhead of minimum 33% to 50%. However, we have calculated that in a particular realistic scenario (4 kByte records, basic SOAP header, 5 lines of 40 characters XML metadata), the total overhead (metadata + DIME packaging) can amount to approximately 50% (2 kByte).
Because of the huge popularity and the relative simplicity and efficiency of RTP, we have concluded that the use of DIME for the streaming of multimedia data is not advisable.

6.2.4 Invocation Mechanisms for Web Services

In Section 5.4.1, we have implemented a mechanism with which a web service can indicate to its client that the service needs to be called back at a later moment in time. While our approach is effective and elegant in that particular use case, it can not easily be extended to other scenario’s and implementations. Furthermore, some of our Media Services are able to operate both in a synchronous (client waits for response) and asynchronous (client proceeds immediately after invoking the service) way. However, there is no standard way of describing such behaviour and integrating such services in an architecture. We have experienced that no less than three essential technologies are still lacking in order to enable such rich collaboration between Web Services:

1. Semantic information on the functionality provided and context required by a Web Service. How do we know, for instance, what the 'Size' field in our client capability description means? How do we ensure that a web service interprets such fields correctly? Numerous Semantic Web Services initiatives try to tackle exactly these issues.

2. Formal and logical (XML-based) descriptions of the composition and cooperation of multiple Web Services. Various business-oriented initiatives (BPEL4WS, XLANG, WSFL, XPDL, etc.) are on the rise to accommodate such descriptions.

3. Richer and stricter communication models for Web Services. Currently, web services are mainly invoked as remote procedure calls (RPC). There is no formal description of this communication, which results in several variants (like ours). Both WSDL and SOAP should formally allow a finer description of the communication model in use (RPC, simple messaging, statefull invocation, etc.).

6.2.5 Polymorphism and Overloading with Web Services

As shown in Section 5.3.3, we have devised a simple mechanism for invoking and substituting different web services that implement the same interface. The services guarantee that they provide the exact same interface, since the classes to which they belong all inherit from the same base class that defines the web service interface.

In our architecture, this mechanism allows Broker services to invoke various Transmoder services using a single interface, as provided by the MessageBasedTxM base class.

6.3 Comparison with Related Research

Section 4.3 discusses the most relevant research initiatives that aim to solve related problems. These initiatives can be grouped in two categories:
• algorithms for multimedia adaptation/caching in networks and
• (service oriented) architectures that provide an infrastructure for such algorithms.

As we have focused our research to the second category (architectures), it is interesting to evaluate how the initiatives in the first category (algorithms) fit in our results.

Roy’s load balancing and resource distribution algorithms have been implemented with older technologies that existed prior to the emergence of web services. However, the algorithms contain nothing that hints a possible obstruction to their implementation in our architecture.

The Digestor system provides algorithms for the transformation of web pages. Its components exchange information and items that are similar to ours. We expect little or no problems when implementing the Digestor algorithms on our architecture. Even more, the Digestor system could benefit from the standard message-based communication mechanisms of web services, since that standardized communication would allow it to become adapted by a broader audience.

Bellavista’s negotiation algorithms and Zhang’s translation proxy should also be implementable on our architecture with very little effort. The various caching algorithms of TranSquid could improve the performance and efficiency of our architecture significantly.

As for the second category of research initiatives, we briefly compare their strengths and weaknesses to ours.

The service oriented framework of Jia Zhang has many similarities to our architecture. It has most of the same advantages and disadvantages. However, the transport mechanism they have proposed for multimedia data do not follow the recent proposals for standardization by W3C and WS-I. Our architecture builds upon DIME and WS-Attachments, at the time of writing the most mature proposals for transport of binary data between web services.

The framework of Boll is the most complete service oriented architecture for multimedia adaptation we have seen. It is very similar to our architecture, yet it suffers from two important drawbacks. First of all, it supports adaptation of multimedia only at the start-up phase of multimedia consumption. This means that changing capabilities of the client device (e.g. battery runs low or wireless connection degrades) are not anticipated by the architecture. Our architecture is developed specifically to tackle also changing circumstances while a multimedia presentation is being consumed. Secondly, Boll’s framework does not perform any multimedia adaptation itself. It only provides for a mechanism to choose from multiple versions of multimedia items that have been prepared beforehand. Therefore, when a situation occurs that has not been anticipated by the supplier of one of the multimedia items, the adaptation framework fails to provide a suited presentation. Our architecture, on the other hand, is intended to continuously adapt multimedia items and presentations to the preferences, configuration and context of the consumer. When a combination of these parameters occurs that was not anticipated by the Broker service that prepared a presentation, it suffices to contact a Broker service with the new parameters to obtain a new suited version.
Chapter 7

Conclusion

Using our service-oriented architecture, content creators may provide multimedia items and/or presentations using the formats and protocols they prefer. We assume that several formats and protocols will remain in use in the future. A multimedia format may be selected because it’s available for free, an open standard, or it may offer support for intellectual property protection. Even though we do not require the use thereof, we advise to use open standard formats for multimedia items, like SVG [45] for vector graphics and SMIL for presentations, since they have the advantage of widespread tool-support. Furthermore, content creators are encouraged to provide their multimedia presentation in a generic and open format, like XiMPF [37, 38] or XHTML [44] and CSS [41], to enable maximum flexibility for the Transmoder services. Apart from formats and descriptions, a content creator may provide access points to its multimedia over any combination of transport protocols like HTTP [40] and RTP [33].

The experiments we performed, show that our web service-oriented architecture is applicable to the re-authoring of multimedia presentations. By introducing only a few image-to-image transcoding and XSLT services, our architecture is already able to transform a multimedia presentation from one format to another, taking into account the client’s capabilities such as screen resolution, available bandwidth and supported SMIL version.

A direct result of our approach is that content creators need no longer bother with a multitude of client platform specifications and connecting networks. Their only concerns are the multimedia applications they wish to publish, the data formats in which those applications are stored internally and the way in which the applications are perceived by clients. Naturally, the format in which the multimedia is offered to the network has a great influence on the multimedia experience by the consumer. However, a content creator’s server is no longer aware of every single scenario in which clients may want to consume its multimedia data. As such, the server will be less vulnerable to obsolescence when new client platforms arise, since multimedia applications can be adopted to new emerging consumer environments by the service in the network.

Our architecture is based entirely on recent open standards, making it open and flexible. It supports complex multimedia presentations, composed of multimedia items, like images, text, video, audio, etc.
Chapter 8

Future Work

This chapter provides a brief overview of the planned and ongoing research activities that build upon the results of our research.

8.1 The Limitations of DIME

WS-I and individual companies currently pay very little attention to media-specific aspects when defining web service enhancement specifications. This is mainly due to the fact that other aspects of web services (security, transactions, etc.) have a higher short-term commercial value and priority. Nonetheless, we believe that a ‘WS-Media’ standard for transport and processing of media with web services could have a significant impact on distributed multimedia systems in the near future. As the W3C’s requirement documents indicate, and as we experienced in our research, building such a media-oriented enhancement for the transport of multimedia data between web services is not a straightforward task.

As the BEA Systems and Microsoft paper on XML, SOAP and binary data illustrates, MIME and DIME based approaches may not make it to broadly accepted standards. We will continue to adapt the most recent (proposals for) standards of W3C and WS-I to multimedia transport.

Our proposal for multimedia-specific enhancements to the transport of information between web services will be developed within the Advanced Media project of the Interuniversity BroadBand Technology centre (IBBT).

8.2 Real-Time Streaming

The DIME-based streaming mechanism that we devised is not ideal for real-time streaming of multimedia. One of the most important drawbacks of DIME is the fact that, in its bare form, it assumes that the records are received in exactly the same order as they have been transmitted. That assumption conflicts with the requirements placed on real-time streaming protocols, where
the time at which a particular record is to be consumed is by far the most important criterion, and the order in which the records arrive is (within boundaries) irrelevant. We have tried to tackle this problem by adding timestamps to the DIME records, thus effectively building an equivalent to RTP with DIME.

The (processing and data) overhead that DIME brings about, in comparison with RTP is significant. We will investigate mechanisms to combine the advantages of our service oriented architecture with the efficiency and elegance of RTP.

RTP is often used in combination with RTSP [34] and RTCP [33]. RTSP and RTCP provide mechanisms to exchange and manage control and quality information on the streams that are transported over RTP. In this document we have illustrated that the need for diverse nodes in the network (at the proxy server, media gateway, home gateway, etc.) that perform caching and adaptation operations on passing (RTP-ed) multimedia is becoming very real. However, RTSP and RTCP in their current form are too limited and semantically poor to support the management and coordination of such nodes.

We will provide a superset of RTSP and RTCP functionality using XML and SOAP. The XML schemas that we will develop, will provide a formal definition of the protocols. This research will be performed within the IST-DANAE project of the sixth European Framework Programme and the Advanced Media project of IBBT.

8.3 Adaptation of Presentations

Complex multimedia presentations consist of a combination of several media objects and one or more descriptions defining the layout, synchronization and behavior (interactivity). When adapting such a presentation, the required changes for layout, synchronization and behavior will be complex and may influence each other. Adaptation of media resources may also influence the presentational descriptions. For example, for a low end device, we may want to replace the video (e.g. MPEG-2) with a sequence of key frames (e.g. JPEG images) and a synchronization description (e.g. XMT-Ω) which must be included in the containing presentational description.

We will investigate algorithms for the adaptation of complex multimedia presentations within the IST-MediaNet project of the sixth European Framework Programme and the Advanced Media project of IBBT.

8.4 Integration of Algorithms

We have intentionally chosen to focus our case studies on the application of message-based item transmoder services. This allowed us to focus on the architecture and interfacing mechanisms of the media services, without getting lost in complex load balancing, caching and buffering algorithms. Nonetheless, these algorithms are important (and under investigation at numerous institutions, as indicated in Section 4.3) and will be incorporated in our architecture.
Bibliography


Appendix A

List of Terms and Standards

CC/PP W3C’s Client Capability/Preference Profile. A profile describes device capabilities and user preferences [50].


DIA MPEG-21 Digital Item Adaptation provides several tools for the manipulation of multimedia content in a networked context, to tailor for the needs of end-user terminals [22].

DIME Direct Internet Message Encapsulation. A Microsoft proposal that defines a mechanism for packaging binary data with SOAP messages [26].

FM Frequency Modulation. The most popular analogue scheme for audio transmission.

HDTV High Definition Television. Standard for high quality television at (interlaced) resolutions of 1920 x 1080 pixels.

ISP Internet Service Provider. The organisation that provides infrastructure through which clients can connect to the Internet.

LAN Local Area Network. Based on the popular ethernet and WiFi standards (IEEE 802.11).

MIME Multipart Internet Message Encapsulation. Encoding scheme for including binary data in textual messages (e.g. E-Mail) [18].


MP3 MPEG-1 layer 3 audio compression.

MPEG Motion Pictures Expert Group. Short name for the international organization for standardization (ISO), joint technical committee 1, sub committee 29, working group 11 (ISO/IEC JTC1/SC29/WG11)

MPEG-2 Formally ISO 13818, developed as the audio and video coding standard for digital television and DVD [20].
APPENDIX A. LIST OF TERMS AND STANDARDS

MPEG-4 Formally ISO 14496, proposed as the coordinating standard for multimedia applications [23].

MPEG-7 The content metadata representation standard for multimedia information search, filtering, management and processing [23].

MPEG-21 Formally ISO 21000, an open framework for multimedia delivery and consumption for use by all the players in the delivery and consumption chain [21].

Nomadic Services As explained in pp. 24–25 of [13], nomadic, service-oriented solutions are built up of autonomous modules of software that perform a well delineated functionality and can be invoked in various contexts.

NTSC National Television System Committee. The NTSC standard has a fixed vertical resolution of 525 horizontal lines, with varying amounts of columns making up the horizontal resolution, depending on the electronics and formats involved. The refresh rate is (interlaced) 59.94 Hz. NTSC is used in about 40 countries, mainly in the American continent [24].

PAL Phase Alternating Lines. The PAL standard has a fixed vertical resolution of 625 lines. The refresh rate is lower than NTSC: (interlaced) 50 Hz, but it has a higher resolution and improved modulation for more consistency in colours [24].

PCM Pulse Coded Modulation. A modulation scheme used to digitize analogue (audio) signals.

QoS Quality of Service. A term used for many diverse initiatives that aim to ensure a certain predictable level of quality for consumers that use (mobile) electronic devices.

RDF Resource Description Framework. A general purpose metadata description language developed by W3C [52].

RTCP Formally part of RFC 1889, RTP Control Protocol. RTCP builds upon RTP to provide a mechanism to exchange information on the quality and reliability of connections [33].

RTP Formally RFC 1889, Real Time Transport Protocol. RTP provides end-to-end network transport functions suitable for applications transmitting real-time data, such as audio or video, over multicast or unicast networks [33].

RTSP Formally RFC 2326, Real Time Streaming Protocol. RTSP is an application-level protocol for control over the delivery of data with real-time properties. It acts as a ‘remote control’ for streaming media, offering features like ‘Play’ and ‘Pause’ [34].

SOA Service-Oriented Architecture. A service-oriented architecture is essentially a collection of (web) services. These services communicate with each other. The communication can involve either simple data passing or it could involve two or more services coordinating some activity. Some means of discovery (UDDI), description (WSDL) and connection (SOAP) of services to each other are needed.

SMIL The Synchronized Multimedia Integration Language. W3C’s SMIL enables simple authoring of interactive audiovisual presentations, using an XML-based textual description. It is typically used for "rich media"/multimedia presentations which integrate streaming audio and video with images, text or any other media type [46].
APPENDIX A. LIST OF TERMS AND STANDARDS

SOAP  Simple Object Access Protocol. SOAP is the W3C’s simple one-way protocol that provides a flexible and extensible way to send structured and typed XML data over any transport protocol. It is most commonly used together with HTTP [48].

SVG  Scalable Vector Graphics. The W3C’s modular language for describing two-dimensional vector and mixed vector/raster graphics in XML [45]. It supports embedding of simple scripts and media objects.

SwA  SOAP Messages with Attachments. Proposal by W3C for the attachment of data to SOAP messages [54].

UAProf  Mechanism to describe client device capabilities. Developed by the WAP Forum [57].

UDDI  The Universal Description, Discovery and Integration protocol. UDDI is an IBM/Microsoft initiative that aims to facilitate abstraction of service location [36].

UMTS  Universal Mobile Telecommunications System. The ITU’s standard for third generation mobile telephony. Data speeds will range from 114 to 2000 kbps, allowing a wide range of high bandwidth (data) applications as well as voice telephony.

WS-Attachments  Microsoft and IBM have devised this model for SOAP attachments. Based on this model, they define a mechanism for encapsulating a SOAP message and zero or more attachments in a DIME message. SOAP attachments are described using a compound document structure consisting of a primary SOAP message and zero or more related documents known as attachments.

WSDL  The Web Service Description Language. WSDL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint. WSDL is extensible to allow description of endpoints and their messages regardless of what message formats or network protocols are used to communicate, however, only the binding of WSDL in conjunction with SOAP and HTTP is currently used.
Appendix B

Paper Published at ICWS2004

The paper that is included in this appendix is accepted for publication at the 2nd IEEE International Conference on Web Services.

The 2004 IEEE International Conference on Web Services (ICWS'2004) is the third international conference focusing on Web Services. ICWS is a forum for researchers and industry practitioner to exchange information regarding advancements in the state of art and practice of Web Services, as well as to identify emerging research topics and define the future directions of Web Services computing. ICWS 2004 has special interest in papers that contribute to the convergence of Web Services, Grid Computing, e-Business and Autonomic Computing, or those that apply techniques from one area to another. ICWS 2004 is sponsored by IEEE Computer Society Technical Community for Services Computing (TCSC).

As the first academic conference in the field of Web services, the 2003 First International Conference on Web Services (ICWS’03) was held at the Monte Carlo Resort in Las Vegas, Nevada, June 23 - 26, 2003, attracting hundreds of participants from 25 countries (USA, India, France, China, Hong Kong, Taiwan, Singapore, Australia, Canada, UK, Sweden, Switzerland, The Netherlands, Germany, Japan, Italy, Korea, Thailand, Finland, Austria, New Zealand, Poland, and Turkey). The second one is 2003 International Conference on Web Services - Europe (ICWS-Europe03), held in Erfurt, Germany, 2003-09-23 and 2003-09-24. ICWS’03 and ICWS-Europe’03 have proven to be an excellent catalyst for research and collaboration, and we fully expect that this ICWS 2004 conference will continue this trend.

The program of ICWS’04 will continue to feature a variety of papers, focusing on topics ranging from Web Services and Dynamic Business Process Composition, Web Services and Process Management, Web Services Discovery, Web Services Security, Web Services Based Applications for e-Commerce, Web Services based Grid Computing, Web Services Standards and Technologies, Web Services Solutions, Web Services Industrial, and other emerging technologies or solutions.

Additional information is available at: http://conferences.computer.org/icws/2004/
Targeting Heterogeneous Multimedia Environments with Web Services

Stijn Decneut  
IMEC  
Kapeldreef 75  
Leuven, Belgium  
stijn.decneut@imec.be

Filip Hendrickx  
IMEC  
Kapeldreef 75  
Leuven, Belgium  
filip.hendrickx@imec.be

Steven Van Assche  
VRT  
Auguste Reyerslaan 52  
Brussels, Belgium  
steven.vanassche@VRT.be

Lode Nachtergaele  
IMEC  
Kapeldreef 75  
Leuven, Belgium  
lode.nachtergaele@imec.be

Abstract

The number of networked multimedia platforms that are introduced into the market has increased dramatically in recent years. Current approaches to multimedia distribution do not scale to this growing set of client configurations and heterogeneous dynamic networks. We propose a distributed architecture that offers a scalable solution to multimedia publication and distribution in such heterogeneous environments. It builds upon recent standardization efforts related to web services. This paper details the multimedia web services at the proxy server, that cooperate on a loosely coupled basis to tailor content creators’ multimedia presentations to clients’ environments. The experiments show that our web service-oriented architecture offers a significant added value in heterogeneous multimedia environments.

1. Introduction

Large-scale production and distribution of media has traditionally been the private playground of large media corporations. The limited amount of widespread broadcast and display standards (such as PAL and NTSC) enables these companies to reach large audiences at a very low cost per consumer [16]. Every media company can safely assume that almost everyone is able to receive and consume its media content when distributed according to the standards.

However, with new multimedia consumption platforms and standards being introduced into the market at a high rate, each one with specific capabilities, massive challenges are arising [17] [37]. Due to the inherent mobile nature of these platforms, their capabilities and connecting network environment may even change dramatically at runtime [4]. Content creators are looking for ways to cost-efficiently publish their content in this heterogeneous environment, where they have to distribute content over multiple dynamic networks to various consumption platforms. The simplest way for content creators to target such an environment is to place their multimedia content on a server in a number of formats that suit popular network and platform scenarios. This approach, often referred to as ‘simulcast’, has been adopted widely on the Internet for audio and video presentations. It suffers from serious drawbacks, however, as it doesn’t scale to large sets of scenarios, and is unable to cope with dynamically changing consumer environments.

A number of scalable multimedia formats have been introduced to alleviate these problems [3] [13] [14]. Still, despite their complexity, they cannot anticipate the heterogeneous set of environments and all the (proprietary) multimedia formats that will be used in the near future [38].

We propose to introduce a web service-oriented architecture with transmoding services in the network, at the proxy server, to offer a scalable approach to publishing multimedia to a heterogeneous environment. With transmoding, we take a broader view to multimedia adaptation than traditional transcoding. A transmoded multimedia item may have a very different appearance than its original, e.g. a textual transcript of an audio sample containing speech or a bitmap version of a vector image. Our service-oriented architecture takes the responsibility of tailoring multimedia data to a suitable format for consumer environments. It provides for continuous adaptation of multimedia presentations and items to the changing environments in which users...
wish to consume them. Even when the capabilities of clients change dramatically at run-time, our architecture continues to adapt multimedia to well-suited formats.

The key added value of our architecture over similar initiatives (see Section 5) is that we use recent standardization efforts like SOAP [34], WSDL [32] and DIME [18] extensively to build a loosely coupled, open, flexible and scalable architecture.

In Section 2, we describe three families of strategies for run-time adaptive multimedia publication. The section provides a background for situating our research. We describe our architecture in detail in Section 3. To illustrate the applicability of our approach, we present a case study in Section 4. Related research activities and architecture frameworks are discussed in Section 5.

2. Approaches to Run-Time Adaptive Multimedia Publication

There are currently three popular approaches to delivering multimedia to multiple platforms.

One way is to enforce a ‘common-denominator’ standard, like FM for radio and PAL or NTSC for TV images. While adhering to these standards, a platform may offer even better capabilities than the standards require (e.g. higher screen resolution). Yet, no matter what the capability of a platform is, it will only render the received multimedia according to the specified standards. For instance, a High Definition Television set may be capable of displaying images at a much higher resolution and frame rate than those defined by the standard, nevertheless it will only display images according to the standard’s specifications.

The differences between client platforms are often more significant in a network context than in the traditional broadcasting world. A standard that is developed specifically for one platform clearly doesn’t scale to very heterogeneous consumer environments where display resolution, processing power and memory size differ tremendously and multimedia data becomes ever more complex and diverse. In such environments, multimedia needs to be made available in specific encoding and formats suitable for the target consumer environments depending on their available resources. Therefore, the Internet today follows a different approach to the delivery of multimedia. A multimedia presentation is typically placed on a server in multiple versions, each one targeting a popular network connection speed and multimedia player configuration. Some examples of very popular versions are 56kpbs, 100kpbs and 300kpbs versions of Windows Media [2] and Real Media [3]. Naturally, web pages that are developed for specific browsers and screen resolutions (e.g. 800 x 600 pixels) can also be regarded as such versions of multimedia presentations.

Other interesting approaches are proposed by MPEG-2 and MPEG-4. They describe a layered approach to video encoding, allowing one multimedia presentation to scale to different bandwidths. On top of a base layer, which contains encoded media that every client should be able to receive and decode, reside several enhancement layers with extra information that can be consumed by clients with higher bandwidths and decoding capabilities [25]. Such an approach is very suitable for highly responsive adaptation of fairly simple multimedia (audio and/or video), within a limited range of capability changes.

The first two approaches are examples of device-specific authoring. Device-specific authoring doesn’t scale to the very large set of client configurations and heterogeneous dynamic networks that multimedia distributors will have to address in the near future [5].

The scalable approaches can be described as multi-device authoring. Even though they provide excellent results when used for a limited range of consumer environments in specific situations, it is impossible to anticipate all modifications that need to be made when complex multimedia content (multiple audio, video, text and other items) needs to be published to a very heterogeneous set of consumer environments.

In those situations automatic re-authoring may prove to offer a more elegant solution. Automatic re-authoring is based on a software system that analyses a multimedia presentation together with the characteristics of the target environment and transforms the presentation (and the items therein) so that it can be transported efficiently and rendered appropriately on the target device. The re-authoring software system is often placed on a proxy server, as proposed by [5] [8] [10].

Automatic re-authoring is particularly interesting when consumer environments change within a session. Such runtime changes often occur in mobile environments where sudden drops in bandwidth or processing power may occur at unpredictable moments in time. Whereas device-specific authoring can not handle any run-time changes (a massive amount of versions would be required), multi-device authoring is only suited for a particular limited range of changes. Automatic re-authoring allows full tailoring of multimedia to specific client preferences, device capabilities and available network at run-time. It is often combined with various caching algorithms to limit the amount of re-authoring and transmoding work that needs to be performed [17]. Naturally, automatic re-authoring can easily be combined with the other two approaches, as our architecture illustrates.
3. A Web Service-Oriented Architecture

Figure 1 illustrates our service-oriented architecture. We introduce several multimedia web services at the proxy server, that cooperate on a loosely coupled basis to tailor content creators’ multimedia to clients’ environments: the Download Manager (Section 3.4), Transmoder (Section 3.5) and Broker (Section 3.2) services. Another multimedia web service runs on the Home Gateway (Section 3.3), offering the Transmoder services easy procedures to provide their results to the client.

As stateless SOAP-based web services do not provide all required functionalities for our multimedia web services, we specify some elementary extensions in Section 3.1.

Using our service-oriented architecture, content creators may provide multimedia items and/or presentations using the formats and protocols they prefer. We assume that several formats and protocols will remain in use in the future. Even though we do not require the use thereof, we advise to use open standard formats for multimedia items, like SVG [30] for vector graphics and SMIL [31] for presentations, since they have the advantage of widespread tool-support. Furthermore, content creators are encouraged to provide their multimedia presentation in a generic and open format, like XIMPF [23] [24] or XHTML [29] and CSS [26], to enable maximum flexibility for the Transmoder services. Apart from formats and descriptions, a content creator may provide access points to its multimedia over any combination of transport protocols like HTTP and RTP [22].

Figure 2. A more detailed look at the services (and the communication in between) that adapt the Content Creator’s multimedia to the capabilities of the Client.

A more detailed view on the architecture is depicted in Figure 2. The client indicates to a Broker service which multimedia presentation it wishes to consume and what its current capabilities are (using CC/PP). Upon this request, the Broker service contacts Download Manager services...
to provide the required multimedia items. The Download Manager services preferably offer these items in a popular, widespread, generic and high quality format, to facilitate the re-authoring process later on. While the Download Manager services are retrieving (from the content creator or a local cache) the multimedia items, the Broker service selects and allocates various Transmoder services. Item Transmoder services receive from the Broker service a URI to the Download Manager service where they can obtain multimedia items using a simple web service call. Streaming Transmoder services, on the other hand, receive a reference to a TCP or UDP socket on which they can contact a Download Manager service to receive multimedia items in a streaming fashion. The Broker service also provides a reference to a Home Gateway service to all Transmoder services, so they know where they have to publish their results.

3.1. Extending Web Services for Streaming Multimedia

SOAP [34] has become the preferred standard for exchanging messages with web services. It is a simple one-way protocol that provides a flexible and extensible way to send structured and typed XML [35] data over a transport protocol. Sending various types and large loads of multimedia data with SOAP, however, can become complex and inelegant. A lot of extra work is involved when encoding the data (to fit the character encoding of the SOAP envelope) and splitting it into smaller chunks (to limit the effects of packet-loss in transport) [39].

That’s where the Direct Internet Message Encapsulation (DIME) [18] comes into play. DIME offers a way to send (binary or text) attachments along with SOAP messages, regardless of their format and encoding. It is similar to MIME [11] but it can be parsed even more efficiently and it is conceived specifically for use with SOAP and web services. Using DIME, (multimedia) data can be sent together with its metadata in a SOAP message. Such a SOAP message can contain various information on the multimedia data (e.g. encoding, description and resolution). Providing metadata is commonly regarded as a key requirement for the successful processing and authoring of multimedia [15]. The ability to send metadata, in a separate description, along with multimedia in an open and standardized way is much more elegant and safe than sending that information on a custom basis.

We use DIME for two purposes: to send SOAP messages with multimedia attachments over HTTP and to stream large sets of multimedia data, together with their SOAP-packaged metadata, over TCP and UDP.

When streaming large amounts of multimedia data with DIME, the standard web service mechanisms are not sufficient. The life-time of a web service is limited to the duration of its web method call, while the streaming itself may take a long time. It may even take several hours in the case of real-time video streaming. Therefore, we introduce threads that run in the memory space of web services and handle the streaming and transmoding of the multimedia. Such a thread typically waits for another component to connect to its specific socket, and communicates using SOAP messages over a DIME stream on that port.

3.2. Broker Service

A Broker service selects and contacts Download Manager and Transmoder services upon a client’s request. The selection it makes is derived from a track history it has stored in an internal database. A Broker decides upon the selection of specific services by analyzing a client’s capabilities. This analysis results in a working space for Transmoder services, determined by the capabilities as indicated by a client. In our architecture, clients use a CC/PP-based description to advertise their capabilities [33]. Using this description, a Broker determines which descriptions and formats can be consumed by the client. It instantiates the appropriate services to provide the multimedia in these descriptions and formats.

Once the services are started and the client can start consuming the multimedia, the Broker service becomes available for other requests.

3.3. Home Gateway

The Home Gateway is a device that is placed, often by an Internet Service Provider (ISP), in the consumer’s home. It acts as a gateway between the consumer’s (wireless) Local Area Network (LAN) and the ISP. As such, it may play a key role in offering a guaranteed service when the consumer accesses information and multimedia on the Internet. Multimedia items form a significant portion of the data that’s placed on the Home Gateway, which caches frequently used data and offers a maximum quality-of-service for multimedia consumption.

In our architecture, the Home Gateway is used to store and cache transmoded multimedia presentations so they can be consumed locally. When the consumer’s environment (e.g. available bandwidth or processing power) changes during a presentation session, the Home Gateway may choose from a number of strategies to adapt the properties of the multimedia presentation:

1. If the changes remain within a given predictable range, scalable multimedia formats are likely to be able to handle the changes.
2. A multimedia item that was delivered by an Item Transmoder can be re-requested in a different form from the Item Transmoder that delivered it, using the
reference that the Item Transmoder gave when delivering the item, as explained in Section 3.5.

3a. When a Streaming Transmoder service is providing (a part of) the multimedia presentation, the bi-directional DIME channel that exists between the Home Gateway and the Streaming Transmoder service can be used to indicate the changes that occurred, enabling the Streaming Transmoder service to adapt the multimedia stream it delivers.

3b. The Broker service anticipates the range of changes that the Streaming Transmoder can handle. When this scope is exceeded, the Home Gateway may contact a new Broker service specifically for the new specification requirements of this multimedia item.

3.4. Download Manager Service

Our Download Manager services reside at the proxy-server level and provide for an abstraction of the protocols used by content creators. They offer the multimedia, which they downloaded over any known protocol, in a uniform way through web services attachments or a continuous DIME-stream. Upon request of a Broker service, they may start pre-fetching multimedia data from a content provider even before a Transmoder service requests it, caching the data in local memory. A Download Manager service may cache several versions of a multimedia item (as in simulcast), to facilitate the tailoring by Broker and Transmoder services.

3.5. Transmoder Service

Transmoder services form the distributed engine of our architecture. They are responsible for a range of transmoding tasks, yet each Transmoder service may support a limited transmoding functionality. Such tasks can be fairly simple, e.g. re-scaling a PNG image from one resolution to another. They can also be quite complex, e.g. extracting key frames from a video stream and sending them as a series of PNG images for consumer devices that do not support streaming video. Transmoder services can be very specialized, focusing on the tasks that they perform best without having to offer other transmoding functionality.

The services place the information they produce directly on a Home Gateway, which was referenced by the instructing Broker service. Apart from the multimedia item and metadata, this information also contains a reference to the generating Transmoder service itself, so the Home Gateway can access it at a later moment when a slightly different version of the multimedia item is required.

We introduce two families of Transmoder services:

- Item Transmoders receive requests of a Broker service to transmode single multimedia items. They retrieve and provide the items they handle using DIME over HTTP, as SOAP messages with attached multimedia data.
- Streaming Transmoders connect to given Download Manager services and Home Gateways using DIME over a streaming protocol. They continue to run as a thread for as long as the multimedia data continues to stream to the consumer. Streaming Transmoders can become quite complex, as multimedia streams have to be retrieved, transmoded and provided at an appropriate pace. It is very important to select good load balancing and buffering schemes, as they greatly influence the resulting multimedia presentation [21].

4. Case Study: Generating SMIL Presentations

To illustrate the added value of our architecture, we applied it to the following scenario. A Content creator has placed a small XML document that describes a simple presentation on a server. The presentation consists of some TIFF bitmap images that have to be displayed for a given duration, together with a textual description and an Ogg Vorbis sound clip [19].

A consumer wishes to view this presentation on a personal computer using the media-player X-Smiles [20]. Through its Home Gateway, it contacts a Broker service and provides the URI of the presentation and a CC/PP description of it’s environment (player type and version, supported formats, screen resolution, etc.). The Broker decides upon the application of specific Transmoder Services. In this scenario a XML-to-SMIL2 transmoder (performing XSL transformations [28]) and a TIFF-to-PNG transmoder will be applied, since X-Smiles requires a SMIL-description of the presentation and it does not support TIFF bitmap images [31]. Both the sound clips and textual descriptions can be placed on the Home Gateway without transmoding if their required bandwidth and encoding are suitable for the consumer’s environment.

The Broker returns the web service call and indicates at what location on the Home Gateway the appropriate presentation can be found. This presentation has been adapted as well as possible to the consumer’s environment. Since a Broker service is only responsible for the presentation at initialization time, it may indicate that a presentation is ready for consumption even before all multimedia items have been placed on the Home Gateway, assuming that these items will be present when needed. The Home Gateway is responsible for re-requesting multimedia items when
they are not present in the correct format during a presenta-
tion session.

We have also applied this scenario to an Apple Quick-
Time player, which offers no support for SMIL 2 and Ogg 
Vorbis [1]. Therefore, the Broker service applies a XML-
to-SMIL1 transmoder and OGG-to-MP3 transmoder to the 
presentation and sound clips respectively, since these for-
mats are supported by the player [27].

4.1. Experimental Results

The experiments we performed, show that our web 
service-oriented architecture is applicable for re-authoring 
multimedia presentations. Under ideal network condi-
tions and server availability, our entire presentation was 
ready for consumption on the Home Gateway in a frac-
tion of a second. The minor overhead caused forms a small 
inconvenience compared to the important merits of our ar-
chitecture. Less than ideal network conditions may in-
fluence this time considerably, but such an extra delay is 
inherent to the transport of multimedia data and has no di-
rect relationship to our architecture. The transport from 
content creator to consumer naturally would also have oc-
curred without our architecture at the proxy server. Since 
our Download Manager service maintains several multime-
dia items in its cache, consecutive runs of the experiments 
show a significant decrease (up to 50%) of this total prepa-
rarison time.

By introducing only a few image-to-image transcoding 
and XSLT services, our architecture is already able to trans-
form a multimedia presentation from one format to another, 
taking into account the client’s capabilities such as screen 
resolution, available bandwidth and supported SMIL ver-

dion.

Figure 3 shows a screenshot of the X-Smiles player, 
playing a SMIL presentation that is generated by our ar-
chitecture. In this version, the selected TIFF-to-PNG Trans-
moder service does not support image rescaling. Images that 
are too large for the player’s display region are shown par-
tially only. The textual description of each image is printed 
in the white text-box underneath the image-region.

The same presentation is shown in Figure 4, but here we 
selected a TIFF-to-PNG Transmoder service that does sup-
port rescaling of images. The black rectangle was drawn on 
the figure after taking the screenshot, to indicate the por-
tion of the image that was visible in Figure 3.

Naturally, rescaling of the images can occur at the client, 
but that results in a significant overhead in the amount of 
data that has to be sent to the client. In our experiments, 
the image of Figure 4 has a size of 13 kB, while the image 
of Figure 3 is about 64kB. If the image would be rescaled 
at the client, this would result in a bandwidth overhead of 
\[
\frac{64 - 13}{13} = 392\%.
\]
MIME’d XML documents can be a difficult and error-prone task. Even though our approach could employ any of the described techniques, in these experiments we have chosen to use WS-Attachments and DIME.

Several architectures have been proposed for multimedia re-authoring. Digestor [5] focuses on re-authoring of WWW pages. Bellavista [4] describes an active middleware to control quality-of-service, specifically for streaming Video-On-Demand. Jia Zhang proposes a SOAP-oriented framework to support device-independent multimedia web services [39]. The framework introduces a mechanism for transporting large multimedia streams from and to web services, which offers an alternative to DIME. A high-level system architecture is proposed by Roy [21]. It focuses on load balancing and resource distribution without providing details on the standards and mechanisms that are used to build the infrastructure. TranSquid [17] focuses on caching in the transmoding services, specifically for e-commerce environments. Ellen Zhang [38] describes a translation proxy for connecting different proprietary players and servers.

The MPEG-21 multimedia framework is an open framework for multimedia delivery and consumption for use by all the players in the delivery and consumption chain [12]. It supports the augmented use of multimedia resources across a wide range of networks and devices used by different communities [6]. Within MPEG-21, Digital Item Adaptation (DIA) [7] describes the manipulation of multimedia content in a networked context, to tailor for the needs of end-user terminals. Such manipulation can consist of the transcoding of a video clip, the translation of a text, etc. [9].

6. Conclusions and Future Work

A direct result of our approach is that content creators need no longer bother with a multitude of client platform specifications and connecting networks. Their only concerns are the multimedia applications they wish to publish, and the data formats in which those applications are stored internally. Naturally, the format in which the multimedia is offered to the network has a great influence on the multimedia experience by the consumer. However, a content creator’s server is no longer aware of every single scenario in which clients may want to consume its multimedia data. As such, the server will be less vulnerable to obsolescence when new client platforms arise, since multimedia applications can be adopted to new emerging consumer environments by the service in the network.

Our architecture is based entirely on recent open standards, making it open and flexible. It supports complex multimedia presentations, composed of any combination of multimedia items, like images, text, video, audio, etc.

Recent extensions to web services, like encryption and routing, have opened a whole new world of service-oriented applications. We will investigate their applicability to and added-value for multimedia re-authoring architectures.

Furthermore, we will investigate caching algorithms that can be applied in order to enable efficient support of large numbers of clients.

7. Acknowledgments

This work is a result of a joint collaboration between Vlaamse Radio en Televisie (VRT, public broadcaster of Flanders, Belgium), the Interuniversity MicroElectronics Center (IMEC), Universiteit Gent (UGent) and Vrije Universiteit Brussel (VUB). This is one of the E-VRT projects funded by the Flemish government.

References