Helsinki University of Technology

Publications in Telecommunications Software and Multimedia

Teknillisen korkeakoulun tietoliikenneohjelmistojen ja multimedian julkaisuja

Espoo 2005 TML-A12

A Graphics Software Architecture for High-End Interactive TV Terminals

Pablo Cesar

TEKNILLINEN KORKEAKOULU
TEKNISKA HÖGSKOLAN
HELSINKI UNIVERSITY OF TECHNOLOGY
TECHNISCHE UNIVERSITÄT HELSINKI
UNIVERSITE DE TECHNOLOGIE D'HELSINKI

Helsinki University of Technology

Publications in Telecommunications Software and Multimedia

Teknillisen korkeakoulun tietoliikenneohjelmistojen ja multimedian julkaisuja

Espoo 2005 TML-A12

A Graphics Software Architecture for High-End Interactive TV Terminals

Pablo Cesar

Dissertation for the degree of Doctor of Science in Technology to be presented with due permission of the Department of Computer Science and Engineering, for public examination and debate in Auditorium T2 at Helsinki University of Technology (Espoo, Finland) on the 9th of December, 2005, at 12 noon.

Helsinki University of Technology
Department of Computer Science and Engineering
Telecommunications Software and Multimedia Laboratory

Teknillinen korkeakoulu
Tietotekniikan osasto
Tietoliikenneohjelmistojen ja multimedian laboratorio

Distribution:
Helsinki University of Technology
Telecommunications Software and Multimedia Laboratory
P.O.Box 5400
FIN-02015 HUT
Tel. +358-9-451 2870
Fax. +358-9-451 5014
http://www.tml.hut.fi/

Available in PDF format at http://lib.hut.fi/Diss/2005/isbn951227888X/

ISBN 951-22-7887-1 (printed version)
ISSN 1456-7911
ISBN 951-22-7888-X (electronic version)
ISSN 1455-9722

Otamedia Oy
Espoo 2005

ABSTRACT

This thesis proposes a graphics architecture for next-generation digital television receivers. The starting assumption is that in the future, a number of multimedia terminals will have access through a number of networks to a variety of content and services. One example of such a device is a media station capable of integrating different kinds of multimedia objects such as 2D/3D graphics and video, reacting to user interaction, and supporting the temporal dimension of applications. Some of the services intended for these devices include, for example, games and enhanced information over broadcasted video.

First, this thesis provides an overview of the digital television environment, focusing on the limitations of current receivers and hints at future directions. In addition, this thesis compares different solutions from regional standardisation bodies such as DVB, CableLabs, and ARIB. It proposes the adoption of the most relevant initiative, GEM by DVB. Unfortunately, GEM software middleware only considers Java language as an authoring format, meaning that the declarative environment and advanced functionalities (e.g., 3D graphics support) remain to be standardised.

Because in the future different user groups will have different demands with regard to television, this thesis identifies two major extensions to the GEM standard. First, it proposes a declarative environment for GEM that takes into account W3C standardisation efforts. This environment is divided into two configurations: one capable of rendering limited interactive applications such as information services, and another intended for more demanding applications, for example a distance learning portal that synchronises videos of lecturers and slides. Second, this thesis proposes to extend the procedural environment of GEM with 3D graphics support. The potential services of this new profile, High-End Interactive, include games and commercials.

Then, based on the requirements the proposed profiles should meet, this thesis defines a graphics architecture model composed of five layers. The hardware abstraction layer is in charge of rendering the final graphics output. The graphical context is a cross-platform abstraction of the rendering region and provides graphics primitives (e.g., rectangles and images). The graphical environment provides the means to control different graphical contexts. The GUI toolkit is a set of ready-made user interface widgets and layout schemes. Finally, high-level languages are easy-to-use tools for developing simple services.

The thesis concludes with a report of my experience implementing a digital television receiver based on the proposals described. In addition to testing the application of the proposed graphics architecture to the design and implementation of a next-generation digital television receiver, the implementation permits the analysis of the requirements of such receivers and of the services they can provide.

Keywords: Digital TV, Software Architecture, GEM, MHP, XML, 3D Graphics, OpenGL

PREFACE

This thesis was written at the Telecommunications Software and Multimedia Laboratory (TML), Helsinki University of Technology, between January 2002 and October 2005 under the supervision of Professor Petri Vuorimaa.

First of all, I would like to thank those who paid my bills so I could expend my time and energy at a higher abstraction layer, thinking instead of worrying about my payments. The financial support during the process came from two different sources: grants and research projects. The grants were awarded by the Nokia Foundation (December 2002-December 2004), the Department of Computer Sciences and Engineering at Helsinki University of Technology (June 2003-June 2004), and the Helsinki Graduate School of Computer Science and Engineering (January 2004-December 2005). The research projects I have worked for include ”Future TV” (May 2000-September 2001) and ”Brocom” (September 2001-December 2004), to whose partners I would like to express my gratitude.

Second, I would like to thank my colleagues who made me feel at home in the laboratory (or maybe should I curse them because in the end I spent most of my time working there instead of enjoying life!). First I would like to thank Petri Vuorimaa, my supervisor, for his patience, trust, and constant support. Juha Vierinen played an essential role, not only as a teacher and pupil - depending on the topic - but as a friend and brother. You really helped me all the way through! I want to thank the people from the ”Future TV” project, who helped out this lost Spanish man in Helsinki: Petri Koistila, Pentti Peisa, and Chengyuan Peng (who taught me hard work is the only way to get a PhD done). People from the X-Smiles project were very important to my development as a researcher, especially Mikko Honkala (you have been my companion through all the doubts and questions of being a researcher) and Kari Pihkala. J. Luc Lamadon, Carlos Herrero, SPB Rao, and Jukka Rauhala from the ”Brocom” project taught me how to manage a research group. Finally, Ilpo Lahtinen for all those Saturday and Sunday morning talks in the lab, and Kristian London for proof-reading this thesis.

In addition, I am really grateful to every single open-source project (you can make a difference!), especially Kaffe, SDL, JSDL, DirectFB, LinuxTV, LinuxSTB, X-Smiles, and OpenMHP.

Finally, I want to thank my family and friends, because a PhD is only a small part of one’s life, even though one tends to forget this. My family has been the source of who I am and how I am. I am more than grateful to them for my education, their constant support, and their constancy. It would have been impossible to go through the painful process without stability at home, and Yoko Yamaguchi has been my shrine (Yokosama, you are my home!). And to all those friends that make this world worth living in: Nachote, Gorka, Dani, Bassam, Jorge, Javi, la Ceci, Julio, Shebeen, Quique, Roberto...Thank you all!

Otaniemi, Espoo, August 2005 Pablo Cesar

TABLE OF CONTENTS

LIST OF PUBLICATIONS

This thesis summarises the following articles and publications, referred to as [P1]-[P9]:

[P1] G. Sivaraman, P. Cesar, and P. Vuorimaa. System software for digital television applications. In *Proceedings of the IEEE International Conference on Multimedia and Expo*, Tokyo, Japan, August 22–25, 2001, pages 784–787. IEEE.

[P2] C. Peng, P. Cesar, and P. Vuorimaa. Integration of applications into digital television environment. In *Proceedings of the 7th International Conference on Distributed Multimedia Systems*, Taipei, Taiwan, September 26–28, 2001, pages 266–272. Knowledge Systems Institute.

[P3] P. Cesar and P. Vuorimaa. A graphical user interface framework for digital television. In *Proceedings of the 10th WSCG International Conference in Central Europe on Computer Graphics, Visualization, and Computer Vision*, Plzen, Czech Republic, February 4–8, 2002, pages 1–4. WSCG.

[P4] K. Pihkala, P. Cesar, and P. Vuorimaa. Cross-platform SMIL player. In *Proceedings of the 1st IASTED International Conference on Communications, Internet and Information Systems*, St. Thomas, US Virgin Islands, November 18–20, 2002, pages 48–53. IASTED.

[P5] J.L. Lamadon, P. Cesar, C. Herrero, and P. Vuorimaa. Usages of a SMIL player in digital television. In *Proceedings of the 7th IASTED International Conference on Internet and Multimedia Systems and Applications*, Honolulu, Hawaii, August 13–15, 2003, pages 579–584. IASTED.

[P6] C. Herrero, P. Cesar, and P. Vuorimaa. Delivering MHP applications into a real DVB-T network, Otadigi. In *Proceedings of the 6st IEEE International Conference on Telecommunications in Modern Satellite, Cable, and Broadcasting Services*, Nis, Serbia and Montenegro, October 1–4, 2003, pages 231–234. IEEE.

[P7] M. Honkala, P. Cesar, and P. Vuorimaa. A device independent XML user agent for multimedia terminals. In *Proceedings of the 6st IEEE International Symposium on Multimedia Software Engineering, IEEE-MSE2004*, Florida, Miami, December 13–15, 2004, pages 116-123. IEEE.

[P8] P. Cesar, J. Vierinen, and P. Vuorimaa. Open graphical framework for interactive TV. *International Journal on Multimedia Tools and Applications, 2006.* Springer, formerly Kluwer Academic. (in print)

[P9] P. Cesar, P. Vuorimaa, and J. Vierinen. A graphics architecture for high end interactive television terminals. *ACM Transactions on Computing Multimedia, Communications, and Applications*. ACM. (Submitted)

LIST OF ABBREVIATIONS

ACAP	Advanced Common Application Platform
AIT	Application Signalling Information Table
API	Application Programming Interface
ARIB	Association of Radio Industries and Business (from ISDB)
ATSC	American Television Standards Committee
A/V	Audio/Visual
AWT	Abstract Windowing Toolkit
BIFS	Binary Format for Scenes
BAT	Bouquet Association Table
CA	Conditional Access
CAT	Conditional Access Table
CDC	Connected Device Configuration
CDMA	Code Division Multiple Access
CG	Computer Graphics
CLDC	Connected Limited Device Configuration
CSS	Cascading Style Sheets
DAVIC	Digital Audio Video Council
DASE	DTV Application Software Environment (from ATSC)
DOM	Document Object Model
DMS-CC	Digital Storage Media - Command and Control
DSL	Digital Subscriber Line
DTV	Digital Television
DVB	Digital Video Broadcasting
DVB-C	Digital Video Broadcasting - Cable
DVB-H	Digital Video Broadcasting - Handheld
DVB-HTML	Digital Video Broadcasting - HyperText Markup Language
DVB-J	Digital Video Broadcasting - Java
DVB-S	Digital Video Broadcasting - Satellite
DVB-T	Digital Video Broadcasting - Terrestrial
DVR	Digital Video Recorder
EBU	European Broadcasting Union
EIT	Event Information Table
EPG	Electronic Program Guide
ETSI	European Telecommunications Standards Institute
GEM	Globally Executable Multimedia Home Platform
GUI	Graphical User Interface
HAL	Hardware Abstraction Layer
HAVi	Home Audio Video interoperability
HCI	Human Computer Interaction
HDTV	High Definition Television
HTML	HyperText Markup Language
ISDB	Integrated Service Digital Broadcasting
ISO	International Standardization Organization
ITU	International Telecommunication Union
JDK	Java Development Kit

J2ME	Java 2 Micro Edition
J2SE	Java 2 Standard Edition
JMF	Java Media Framework
JRE	Java Runtime Environment
JSR	Java Specification Request
JVM	Java Virtual Machine
LAN	Local Area Network
MHP	Multimedia Home Platform (from DVB)
MHEG	Multimedia and Hypermedia Information Coding Experts Group
MIDP	Mobile Information Device Profile
MPEG	Moving Pictures Expert Group
MPEG-J	Moving Pictures Expert Group - Java
MSTV	Microsoft Television
MUG	Multimedia Home Platform Umbrella Group
MVC	Model View Controller
NIT	Network Information Table
OCAP	OpenCable Applications Platform
OpenGL ES	OpenGL Embedded Systems
PAN	Personal Area Network
PAT	Program Association Table
PID	Packet Identification
PMT	Program Map Table
PSI	Program Specific Information
PPV	Pay Per View
PSTN	Public Switch Telephone Network
RAM	Random Access Memory
ROM	Read-Only Memory
RST	Running Status Table
SDT	Service Description Table
SI	Service Information
SMIL	Synchronized Multimedia Integration Language
SMS	Short Messaging Service
STB	Set-Top Box
SVG	Scalable Vector Graphics
TS	Transport Stream
UI	User Interface
VRML	Virtual Reality Markup Language
W3C	World Wide Web Consortium
WAN	Wide Area Network
WAP	Wireless Application Protocol
WIMP	Windows, Icons, Menus, and Pointing devices
WWW	World Wide Web
X3D	eXtensible 3D Graphics
XAIT	eXtended Application Information Table
XHTML	eXtensible Hypertext Markup Language
XML	eXtensible Markup Language
XMT	eXtensible MPEG-4 Textual format
XSL	eXtensible Stylesheet Language
XSLT	XSL Transformation
XSL FO	XSL Formatting Objects

1 INTRODUCTION

Chapter 1: Introduction
(Big Picture of multimedia systems)
Multimedia Devices
Devices' Connectivity
Content Authoring Formats
Graphics Architecture

Technological advances such as the convergence of networks, hardware design improvements, and the development of efficient compression techniques are making projections of invisible [125], ubiquitous [162, 163, 164], and pervasive [76] computing a reality. This revolution in interactive multimedia devices has drawn the attention of both researchers and corporations, generating a number of challenges. Olsen, who defined the current situation as chaotic, wrote in his article "Interacting in chaos" [129]:

> *Exponential growth in computing capability puts vast capabilities into very small devices [...] Dealing with the chaotic diversity of information, collaborators, and interactive devices is the principal interactive challenge of the next decade [...] Yet another complication in the future of interactive computing is the variability of future interactive devices.*

The starting assumption of this thesis, depicted in Figure 1.1, can be defined as follows:

> *In the future, a number of multimedia devices will have access through a number of networks to a variety of content and services.*

Such a wide problem requires decomposition to its basic elements: terminals, connectivity, and content. First, the number of multimedia platforms has increased astoundingly in recent years (e.g., digital television receivers, mobile phones, game consoles). Second, these devices can potentially be connected to different networks. For example, digital television decoders can receive the broadcasted stream, use the return channel, and also be Bluetooth enabled. Finally, the number of choices available when implementing services is increasing day after day. For instance, the Java Application Program Interface (API) chosen for development depends on the targeted device; Digital Video Broadcasting - Java (DVB-J) for digital television receivers and Mobile Information Device Profile (MIDP) for mobile phones are just two examples.

The first four sections of this chapter review the different components of multimedia systems in more detail. In Section 1.1, different multimedia devices are described and their major differences are studied. Next, Section 1.2 overviews the devices' connectivity. Then, Section 1.3 introduces

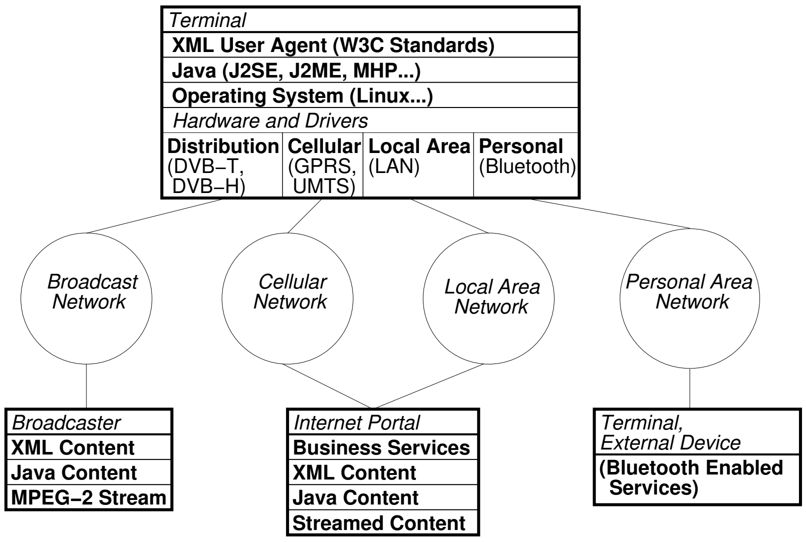


Figure 1.1: Multimedia Device's Situation at the Turn of the Century.

various kinds of content and service formats that can be used to implement multimedia applications. Section 1.4 defines the graphics system of a multimedia device. Sections 1.5 to 1.7 delimit the scope of the study by stating the assumptions and research questions, describing the followed methodology, and highlighting the contributions and implications, respectively. Finally, Section 1.8 outlines this thesis.

1.1 Multimedia Devices

Not so long ago, the only platform developers had in mind when implementing multimedia applications was the personal computer. But, as Myers wrote in the year 2000 [116], "*We are at the dawn of an era where user interfaces are about to break out the desktop box where they have been stuck for the past 15 years*". The corollary is that today's multimedia applications are expected to run on a variety of devices, from personal computers to non-desktop devices.

The intention of this section is to answer two basic questions:

- *What are the differences between a multimedia-enabled personal computer and a non-desktop multimedia device?*
- *What are the differences among these non-desktop devices?*

There are as many ways of classifying multimedia devices as people who have thought about the topic. For example, Want [161] differentiates them depending on their location: office, home, and automobile. Revett [149] classifies them as fixed (e.g., set-top boxes, kiosks) and mobile (e.g., mobile phones). Neuvo's [119] taxonomy is based on platforms and their market penetration: mobile phones, personal computers, game consoles, set-top

Table 1.1: Taxonomies of Multimedia Devices in the Literature.

Author and Publication	**Categories**
Want et al. [161]	Office Home Automobile
Revett and South[149]	Fixed Mobile
Neuvo and Yrjanainen [119]	Mobile phone Personal computer Game console Set-top box Personal digital assistant
Dustdar [45]	Personal Communication Corporate
Lewis et al. [94]	Televisions Phones Personal information managers Miscellaneous
Hansmann et al. [76]	Information access Intelligent appliances Smart controls Entertainment systems

boxes, and personal digital assistants. Dustdar [45] groups devices depending on their usage: personal (e.g., interactive TVs, kiosks), communication (e.g., point-to-point conference systems), and corporate (e.g., multimedia training systems). Lewis [94] categorises information appliances, defined as hardware devices with online connectivity, depending on their dominant features: televisions, phones, personal information managers (e.g., personal digital assistants), and miscellaneous (e.g., digital cameras). Finally, Hansmann [76] identifies four main classes: information access devices (e.g., smart phones), intelligent appliances, smart controls (e.g., smartcards), and entertainment systems (e.g., televisions). Table 1.1 summarises these different taxonomies.

In this thesis, the ideas presented in the previously described taxonomies are combined: multimedia devices are classified according to their purpose (e.g., watching TV) and their particular capabilities (e.g., input device). The main problem is that such a classification might become obsolete in the near future. On the one hand, several companies such as Nokia are exploring the boundaries of the use of mobile phones (e.g., through their N-Gage products, which feature game console capabilities). On the other hand, the configuration of a computing device changes in a short period of time - trials to include a keyboard as an input mechanism for digital television receivers are just one example.

Even though differentiating between platforms based on purpose might seem straightforward, the personal computer as a multipurpose device is in opposition to, for example, mobile phones being primarily intended for voice communication. These boundaries are slowly disappearing. Today mobile phones can access the Internet using Wireless Application Protocol (WAP) or can even receive digital television transmission using Digital Video Broadcasting - Handheld (DVB-H) [56]. Digital television receivers run DVB-J applications and eXtensible Markup Language (XML)-based content: Digital Video Broadcasting - HyperText Markup Language (DVB-HTML) [52].

Still, certain basic differences and similarities related to the platform's purpose, or to user expectations of the platform's purpose, can be observed. First of all, non-desktop devices are rarely used for development, in contrast to personal computers. Second, all the devices fall into the broader definition of information access systems (e.g., for watching the news or accessing the Internet). Nevertheless, depending on how the information is transmitted, one can divide information access systems into open and closed environments. In open environments, the information can be retrieved from diverse sources, e.g., the World Wide Web (WWW). Closed environments, similar to walled gardens, rely on specific service providers, such as broadcasters in the case of digital television or an internal database in the case of a kiosk.

Based on the previous discussion and taxonomies, the following uses for interactive multimedia non-desktop devices are differentiated in this thesis:

- **Information Management:** a small, pocket-size device in which the user carries information, for example, a personal digital assistant.
- **Person-to-Person Communication:** a device used for communication in a synchronous (e.g., voice call) or asynchronous manner (e.g., messages) with other people, for example, a mobile phone.
- **Entertainment:** a device dedicated to free time activities, for example, an MP3 player, DVD player, digital television receiver, or game console.

At the time of writing this thesis, the issue of device specialisation versus multipurpose devices is not yet answered. In fact, it will not be answered until the market decides, and anyway the world is in a continuous evolution. But still, this classification permits us to measure to what extent the inclusion of new capabilities into a device constitutes a risk or becomes a natural extension. Table 1.2 shows different devices and their usages.

Multimedia systems can be described, as well, as those systems capable of handling user interaction, supporting multiple media (i.e., discrete and continuous), and taking into account the temporal dimension of applications. Hence, multimedia devices can be categorised according to their configuration: input and output devices, multimedia objects support, synchronisation mechanism (e.g., synchronisation among multimedia objects), and internal features (e.g., memory, processing power, and network connection) [19]. Table 1.3 shows the different configurations of selected devices. In the table, only relative values regarding the internal features are given, because absolute values would be obsolete next year.

In this section, the current situation regarding multimedia devices has been studied from two different points of view: platform purpose and in-

Table 1.2: Multimedia Devices and their Usage.

		PC	**Non-desktop Device**		
			DTV Receiver	**Mobile Phone**	**PDA**
Development		+++	+	+	++
Information Access	Open	+++	+	++	++
	Closed	+	+++	+	++
Information Management		+++	+	++	+++
Communication		++	+	+++	++
Entertainment		+++	+++	++	+

Table 1.3: Device Configuration (based on the table from Petri Vuorimaa's course Multimedia Techniques, Helsinki University of Technology).

	PC	**Non-desktop Device**		
		DTV Receiver	**Mobile Phone**	**PDA**
Output Device	Monitor	TV Screen	Small Screen	Medium Screen
Input Device	Mouse, keyboard	Remote Control	Key Pad	Keyboard, Stylus
Continuous Media	+++	+++	+	++
Discrete Media	+++	+++	+++	+++
Processor	+++	++	+	++
Memory	+++	++	+	++
Network	+++	+	++	+++

ternal features. The general conclusions are that personal computers are multipurpose and powerful devices, whereas non-desktop devices tend to be specialised, and thus with limited capabilities. In addition, information access is one capability all multimedia devices share, both as closed environments (e.g., kiosks, CD-ROMs, broadcast channels) and as open environments (e.g., devices offering Internet access). Finally, digital television receivers can be defined as computer-mediated entertainment devices [29] supporting multiple media types but in which network capabilities can still be improved.

1.2 Devices' Connectivity

Networks can be differentiated, depending on their range, into Personal Area Networks (PANs), Local Area Networks (LANs), and Wide Area Networks (WANs) [75, 95, 156, 157]. PANs are used to interconnect devices close to one person, usually within a few meters' distance. LANs, on the other hand, are restricted to a small geographical area (e.g., a building or campus) of typically up to a few-kilometer radius, so terminals in that area can easily be interconnected. Because of the small distance between terminals, LANs offer a high data transmission rate. Finally, WANs cover a large geographical

Table 1.4: Examples of Device Connectivity.

	Fixed	**Wireless**
Other Devices	USB, Firewire	Bluetooth
Home Network	Ethernet	Wireless Ethernet
Internet Access	PSTN, xDSL	GSM, CDMA
Broadcast	DVB-T, DAB	DVB-H

area (e.g., a city or country).

In addition to their range, networks can be divided into fixed and mobile. As noted by Black, our society has become ”information dependent” [13], meaning that individuals need to be reachable regardless of location. This shift in society has also instigated an effort to develop wireless network technologies that are comparable to existing fixed networks.

Table 1.4 categorises wired/wireless connectivity for multimedia systems. This connectivity can be divided into:

- **Other Devices:** some examples include Universal Serial Bus (USB) and Firewire, in which the computer is wired with a computer bus to download or upload content. One wireless example is Bluetooth, a PAN that uses a radio frequency to connect devices, creating a network called Piconet in a master-slave relationship.
- **Home Network:** some examples include ethernet (i.e., a LAN connection) and wireless ethernet.
- **Internet Access:** in order to access the Internet, the device can use the analogue telephone network: Public Switched Telephone Network (PSTN). But currently a broadband technology, Digital Subscriber Line (DSL), is becoming the most accepted option. Some wireless technologies to access the Internet are Global System for Mobile Communications (GSM) and Code Division Multiple Access (CDMA).
- **Broadcast:** some examples include Digital Video Broadcasting - Terrestrial (DVB-T) and DVB-H for terrestrial digital television and Digital Audio Broadcasting (DAB) for terrestrial digital radio. Since broadcast is a unidirectional path, the device only receives information.

As an example of and conclusion to this section, the connecitivity of digital television receivers is studied. As depicted in Figure 1.2, their configuration can include the following technologies:

- **Broadcast:** for reception of DVB-T streams, coded using Moving Picture Experts Group (MPEG)-2.
- **Return channel:** analogue modem or xDSL provides two-way connectivity. In addition, a LAN network such as ethernet can be used for creating a home network. The use of these technologies allows the user to enhance the content received through the broadcast network and to submit information (e.g., when purchasing an item).

Table 1.5: Content and Service Technologies for Multimedia Terminals.

Content	**Technology**
Broadcasted A/V	MPEG-2
Streamed Content	MPEG-4
Application	J2SE, J2ME, DVB-J
Structured Content	SMIL, SVG, XHTML, DVB-HTML

- **Personal:** wireless, short-range networks such as Bluetooth [147], which can be used for a variety of purposes: for example, to upload pictures from a digital camera to enjoy a slide show from the sofa.
- **Infrared:** even though not a real network, infrared technology is used to connect peripherals such as remote controls or wireless keyboards.

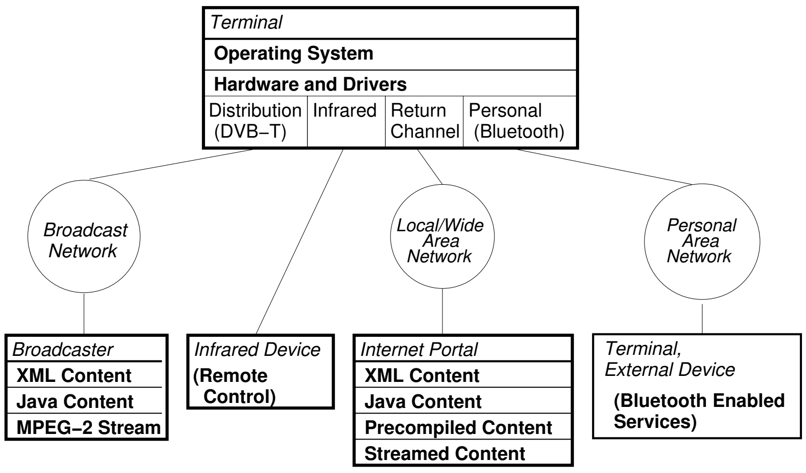


Figure 1.2: Example of Digital Television Receiver Connectivity (A specific case of the model depicted in Figure 1.1)

1.3 Content Authoring Formats

In addition to diversity of terminals and hybrid networks, the third factor, as depicted in Figure 1.1, is the growing number of standards used to digitise content and services. The adoption of such standards simplifies the developer's task because she can target to a group of devices instead of to a specific model. At the same time, third parties (i.e., companies apart from the device manufacturer) can implement services based on the standards, thus increasing the amount of available content. Table 1.5 shows current technologies used as content and application formats.

First, the basic Audio/Visual (A/V) digital television stream uses the MPEG-2 standard, while streamed content will, most probably, use MPEG-4. Java language, because of its platform-independent nature, is widely used for developing applications. For example, Sun has defined Java 2 Standard

Edition (J2SE)[1] and Java 2 Micro Edition (J2ME)[2] for applications targeted to personal computers and non-desktop devices, respectively.

Regarding XML-based languages, the World Wide Web Consortium (W3C)[3] has put a lot of effort on standardising different multimedia issues using XML [18]. Some examples include:

- eXtensible HyperText Markup Language (XHTML) as a reformulation of HyperText Markup Language (HTML) 4 in XML [11, 133].
- Synchronised Multimedia Integration Language (SMIL) for multimedia presentations [80, 81].
- XForms as the new generation of Web forms [43].
- Scalable Vector Graphics (SVG) to describe vector-based graphics for the Web [61].

Moreover, Multimedia Home Platform (MHP), the European standard for digital television middleware, has defined, apart from DVB-J, an XML-based content format for digital television called DVB-HTML.

1.4 Graphics Architecture

The element that makes the two-way communication between the human user and the computer system possible is called the user interface [89]. Hence, the user interface of a system is responsible of what the user experiences (i.e., sees and hears) and determines his potential behaviour: what he can do and how he can do it. Studying the user interface involves a number of issues, and the literature is broad [44, 122, 124, 126]. Some essential topics in Human Computer Interaction (HCI) curricula include requirements analysis and specification, user interface design, development, and evaluation. This thesis focuses in one specific topic from among these: system's graphics architectures.

The graphics architecture of a multimedia platform is one part of the system software in charge of the user interface at runtime. It has to accept and process user inputs and to render the system output on a screen. Regarding interactive multimedia systems, there are three basic issues that the graphics system should provide:

- **Multimedia objects support:** for both continuous (e.g., video and audio) and discrete (e.g., text and graphics) media [77, 78, 143, 166].
- **Interaction:** the user should be able to modify the state of the system as desired. There are different levels of interactivity, for example, it is not the same thing to browse web pages and to purchase a e-ticket.
- **Synchronisation:** apart from the spatial dimension, multimedia presentations include a temporal dimension that has to be considered by the graphics systems (e.g., knowing when to hide a component) [15, 140].

[1]http://java.sun.com/j2se
[2]http://java.sun.com/j2me
[3]http://www.w3c.org

1.5 Aim of the Study

First, this thesis aims to show that the decision about which content authoring format to use when implementing services depends on the development purpose. For example, certain simple applications (e.g., advertisements) can be coded using XML-based languages, but applications demanding more resources (e.g., Doom-like interactive games) should be coded at a different level. Thus, a graphics architecture model for media stations is proposed. The model is divided into different layers, depending on their difficulty of use and expressive power. Second, this thesis is based on the Globally Executable MHP (GEM) standard, a worldwide alternative for digital television receivers. Because the capabilities of devices are evolving constantly, this thesis proposes a group of extensions to GEM. Finally, because manufacturers also need to differentiate their products, this thesis recommends improvements to their graphics architectures, such as convergence with game consoles.

Definitions

- Media Station: a computer-mediated entertainment device. Some examples include digital television receivers, DVD players, and game consoles.
- Procedural Application: "an application which primarily makes use of procedural information to express its behaviour; a Java program is an example of a procedural application" [86].
- Procedural Environment: "an environment that supports the processing of procedural applications" [86].
- Declarative Application: "an application which primarily makes use of declarative information to express its behaviour; an XML document instance is an example of a declarative application" [86].
- Declarative Environment: "an environment that supports the processing of declarative applications" [86].
- Application Environment: composed of both the procedural and the declarative environments.

Assumptions

- The main capabilities of digital television receivers are reception of the A/V stream, its computer-mediated entertainment nature, and its potential access to the WWW.
- GEM is a valid alternative as a worldwide application environment for digital television receivers.
- In order to be interoperable, manufacturers need to follow GEM.
- In order to be competitive, manufacturers need to differentiate their products.

- Currently, GEM only defines the procedural environment. Hence the declarative environment remains to be standardised.
- Currently, GEM's procedural environment only takes into consideration the broadcasted A/V stream and 2D graphics support. Thus, DVB should continue evolving (e.g., towards 3D graphics support and WWW access) if it wants to continue being a principal actor.

Research Questions

1. *What are the limitations of current digital television receiver implementations?*
2. *How can the GEM standard be extended so it defines a declarative environment?*
3. *How can the GEM standard be extended so it includes support for all kinds of multimedia objects (2D, 3D, and video) and WWW access?*
4. *How can manufacturers, following GEM, differentiate their products?*
5. *What graphics architecture is needed to implement GEM and the extensions proposed in this thesis? Is this architecture valid for other media stations?*

1.6 Methodology

The methodology used in this thesis was an analysis of the current state of digital television environments. After a literature review and an examination of existing standards and solutions (e.g., Tivo and WebTV), the missing parts of the puzzle were identified. First, each receiver deals with a specific aspect of user expectations: Digital Video Recorder (DVR) in Tivo, Web access in WebTV, and MHP implementation in Osmosys. Second, the most relevant theoretical model for a digital television metaphor is Chorianpoulos' doctoral thesis [29]. While Chorianpoulos concentrated on a high-level metaphor, digital television as a virtual channel provider, this thesis focuses on the graphics architecture needed to implement next-generation digital television terminals.

This study and analysis was carried out through a "step-by-step" approach. When some ideas, such as SMIL usage in digital television environments, were studied, the concept implementation[4] research method was used and the results were submitted to international conferences. For this reason, it was decided to present this thesis as a group of publications, nine in total, instead of a monograph. By integrating each of the results at the end, a whole system called Ubik was constructed as the final proof of concept of the contributions of this thesis. It was decided, as well, to include in this thesis the first papers published because they highlight the basic starting problems, even though they might not be completely relevant for the final

[4]"[...] papers whose prime research method is to demonstrate proof of a concept by building a prototype system" [146]. This method, according to Glass' findings in [72], is the second-most used in the software engineering discipline.

results of this thesis. Finally, the introductory part of this thesis was written at the end of the process using literature review and conceptual analysis[5].

1.7 Contributions

The contributions of this thesis can be divided into those related to standardisation and those related to commercialisation. First, this thesis analyses digital television standards and proposes extensions such as 3D graphics support. Second, it defines a graphics architecture model that takes into account the special restrictions of media stations. Finally, this thesis provides a reference implementation, following the model, of a next-generation digital television receiver. The implementation can enhance television content by overlaying 2D and 3D graphics, running Java services, and displaying XML-based documents.

The evolution of digital television receivers will result in a diversity of configurations, each targeted to a certain user group. In addition, because of the importance of television in our societies, there is a necessity to standardise the field. Thus, open standards like GEM need to evolve in order to maintain their predominant role. This thesis proposes extensions to current standardisation initiatives, differentiating two main groups of digital television receivers: broadcast and interactive. These groups can be further divided into profiles. First, regarding the procedural environment, a new profile is defined, High-End Interactive, in which 3D graphics support is included. In addition, this thesis recommends implementing it as a thin layer wrapping OpenGL for Embedding Systems (OpenGL ES) functionality. Second, this thesis describes two declarative-environment profiles as replacements for the MHP solution (i.e., DVB-HTML): first, XHTML + Cascading Style Sheets (CSS) for defining the structure and look of limited interactive applications (e.g., Super Teletext) and, second, a XHTML + XForms + SMIL syntax to support more complex applications (e.g., a distance education Web portal).

Based on the requirements identified for the most advanced profile, High-End Interactive, a layered graphics architecture model for digital television receivers is proposed. This model is based on a traditional desktop graphics architecture, but replaces the windowing system by a scene manager and explicitly integrates different multimedia objects (e.g., video and 3D graphics). Because implementing a 3D game is different from implementing a information service, the layers of the model are ordered based on their difficulty of use and expressive power. Hence, the developer can decide which of the layers is most appropriate for her goals. Moreover, this model can be generalised for other media stations, such as game consoles.

Finally, the model is validated by the development of a reference implementation of a digital television receiver called Ubik. The validation criteria was, first, to prove that a prototype system following the model could be constructed. In other words, the purpose was to see if the layers included in the model constituted a valid approach or if another group of layers were

[5]A study of the concepts presented, not using mathematical methods [146, 160]. For example, the validation of a model depending on requirements. This method, according to Glass' findings in [72], is the most used in the software engineering discipline.

needed to design and implement next-generation digital television receivers. In addition, several use cases (e.g., open-source 3D games or XML-based applications) meeting the requirements of High-End Interactive television were tested on the system in order to prove that Ubik met the identified requirements. Finally, with the construction of Ubik, the expressive power of the model's layers were tested in [P8] and the potential services and their performance were analysed in [P9].

1.8 Organisation of This Thesis

This thesis, as depicted in Figure 1.3, is divided into six chapters. The first one has introduced the current situation regarding multimedia interactive devices. Chapter 2 provides an overview of digital television, focusing on current limitations and possible improvements. Next, Chapter 3 reviews different programming and content description languages for interactive multimedia applications, categorising them depending on their expressive power and difficulty of use. After that, it reviews the most relevant digital television standardisation initiatives and describes a worldwide application environment based on the GEM standard. In Chapter 4, extensions to GEM and specific recommendations for receiver manufacturers are outlined, and in Chapter 5, a graphics architecture for next-generation digital television receivers is proposed. Next, Chapter 6 introduces two reference implementations - Otadigi and Ubik - of the proposals and the concepts that were reviewed theoretically. Ubik is a graphics architecture implementation for configurable digital television receivers, while Otadigi is a broadcast terrestrial television broadcast environment. Finally, Chapter 7 concludes this thesis. Table 1.6 indicates the chapter in which each research question is answered.

1.9 Summary

In this chapter, the general assumption of this thesis was proposed: in the future, a number of multimedia devices will have access through diverse networks to a variety of content. For this reason, a general overview of the issues involved, including multimedia systems, device connectivity, and variety of content, was described. Then, the topic of this thesis was defined as a runtime user interface system software for digital television receivers. Finally, the five research questions that motivated this thesis were proposed:

- *What are the limitations of current digital television receiver implementations?*
- *How can the GEM standard be extended so it defines a declarative environment?*
- *How can the GEM standard be extended so it includes support for all kinds of multimedia objects (2D, 3D, and video) and WWW access?*
- *How can manufacturers, following GEM, differentiate their products?*

Table 1.6: Chapters in Which the Research Questions Are Answered.

Research Question	**Chapter**
1. What are the limitations of current digital television receiver implementations?	2: Digital Television Overview
2. How can the GEM standard be extended so it defines a declarative environment? 3. How can the GEM standard be extended so it includes support for all kinds of multimedia objects (2D, 3D, and video) and WWW access? 4. How can manufacturers, following GEM, differentiate their products?	4: Extensions to Multimedia Home Platform and Recommendations to Manufacturers
5. What graphics architecture is needed to implement GEM and the extensions proposed in this thesis? Is this architecture valid for other media stations?	5: Graphics Architecture for Digital Television Receivers 6: Reference Implementations (Otadigi and Ubik)

- *What graphics architecture is needed to implement GEM and the extensions proposed in this thesis? Is this architecture valid for other media stations?*

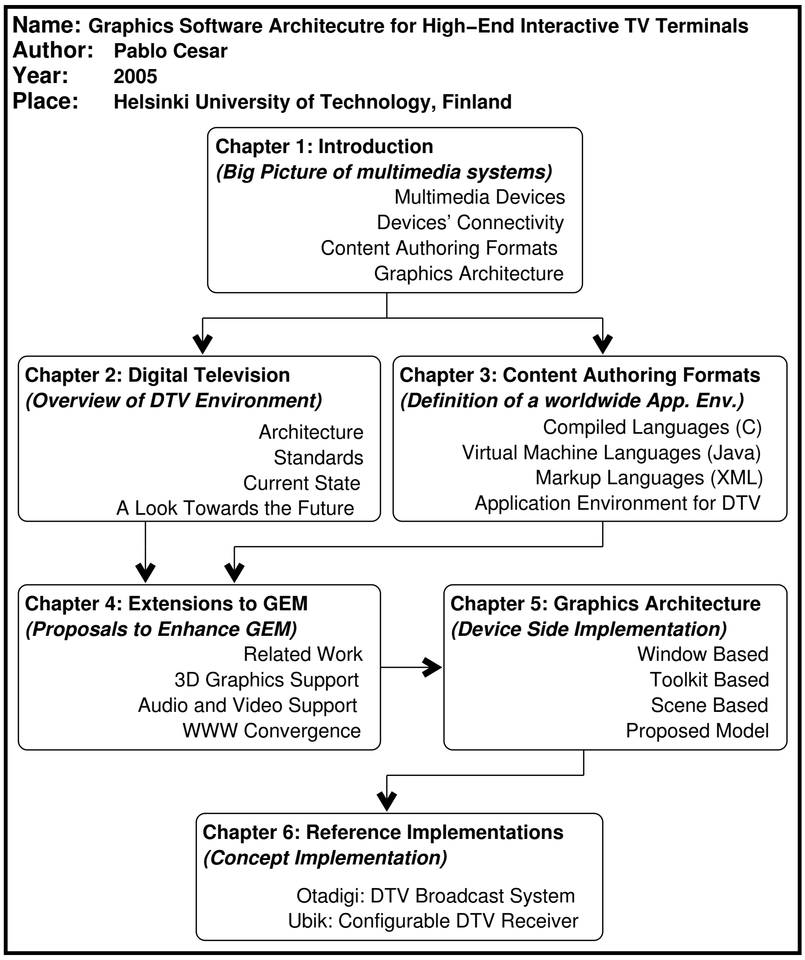

Name: Graphics Software Architecutre for High−End Interactive TV Terminals
Author: Pablo Cesar
Year: 2005
Place: Helsinki University of Technology, Finland
Chapter 1: Introduction
(Big Picture of multimedia systems)
Multimedia Devices
Devices' Connectivity
Content Authoring Formats
Graphics Architecture
Chapter 2: Digital Television
(Overview of DTV Environment)
Architecture
Standards
Current State
A Look Towards the Future
Chapter 3: Content Authoring Formats
(Definition of a worldwide App. Env.)
Compiled Languages (C)
Virtual Machine Languages (Java)
Markup Languages (XML)
Application Environment for DTV
Chapter 4: Extensions to GEM
(Proposals to Enhance GEM)
Related Work
3D Graphics Support
Audio and Video Support
WWW Convergence
Chapter 5: Graphics Architecture
(Device Side Implementation)
Window Based
Toolkit Based
Scene Based
Proposed Model
Chapter 6: Reference Implementations
(Concept Implementation)
Otadigi: DTV Broadcast System
Ubik: Configurable DTV Receiver

Figure 1.3: Overview of This Thesis.

2 OVERVIEW OF DIGITAL TELEVISION

Chapter 2: Digital Television
(Overview of DTV Environment)
Architecture
Standards
Current State
A Look Towards the Future

The importance of television in our society is unquestionable. It has become a special piece of furniture in the home. Not only it is a convenient surface for placing flowers or photos on; once it is switched on, it can receive and display a huge amount of information from outside the living room. Although its use depends on a strict schedule defined by the broadcaster, it informs, teaches, entertains, and helps us to relax. In the last few years, the world of television is facing a critical challenge: its convergence with the digital world.

On August 27th 2001, digital television broadcast started in Finland. It was decided, as well, that by the year 2007 analogue television broadcast will be brought to an end. Similar scenarios can be seen all over the world (e.g., in the United Kingdom, South Korea, and Germany)[1]. Figure 2.1 shows the launch dates of digital terrestrial television in European countries. Because of the intrinsic importance of television in our societies, the adoption of digital television becomes a delicate issue, of which the main concern today is ensuring a smooth transition from analogue.

Even though broadcasters have used digital technology for years (e.g., for storage of material), digital television refers to the transmission of the signal in digital form. This is possible because of the international compression standard MPEG-2 [64, 148, 150, 151]. Its main feature is the compression of audio, video, and data in the same transport stream. The stream can be manipulated using software, opening new opportunities to process the information (e.g., encryption techniques). The major benefit of digital television is its high transport efficiency [111]. Some of the benefits of digital television in comparison to its analogue counterpart include:

- Increase in the number of television programs broadcasted using the same bandwidth.
- Improvement in the quality of the signal.
- Inclusion of additional data in the transport stream (i.e., provision of information and services).
- Elimination of interference, such as ghosting.

[1]Updated information about each European country can be obtained from DigiTag (http://www.digitag.org). Worldwide updated information can be obtained from DVB (http://www.dvb.org).

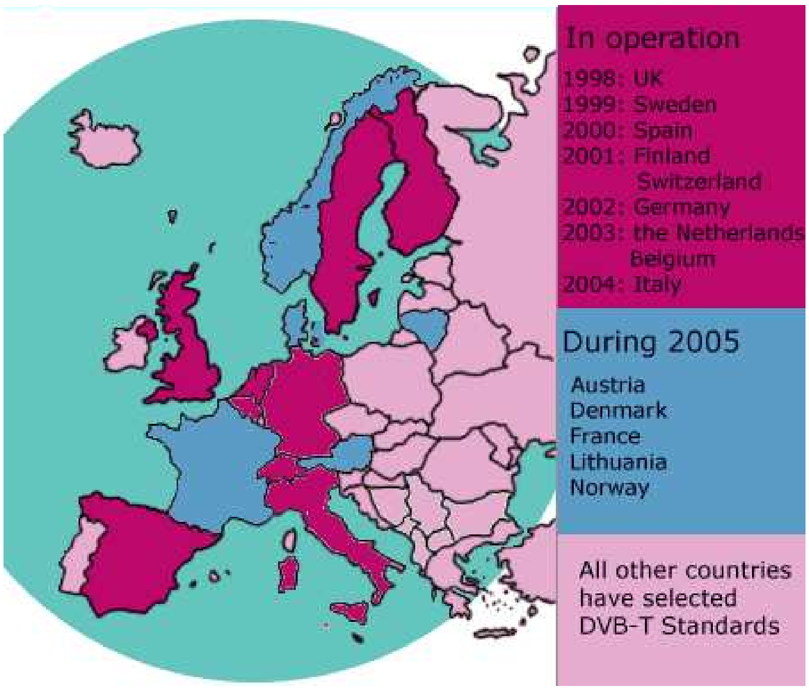


Figure 2.1: DVB-T Launch Dates. Source: DigiTag (http://www.digitag.org, April ’05).

- Provision of return channel mechanisms (i.e., true interactivity).

Television is about bits, not about pictures, wrote Negroponte in ”Being Digital” [117], meaning that the major challenge for television was to become digital. This statement implies that technological improvements will provoke a dramatic change in the way viewers conceive the television receiver: it will transform into a multimedia terminal [130]. This transformation implies two major issues. First, the receiver would have the capability to communicate with other home devices (i.e., through a home network), which is out of the scope of this thesis. Second, the evolution of the terminal most probably will result in a wide variety of receivers differentiated by their capabilities, and thus by their prices. The cheapest ones will only allow the reception of the A/V stream. More expensive ones will include interaction capabilities (i.e., local or interaction channels), support for interactive services (e.g., Java-based applications), and storage capacity. Finally, high-end receivers will provide WWW access or converge with game consoles.

This chapter is structured as follows. First, Section 2.1 proposes a group of requirements that digital television should meet in order to provide an incentive for the shift from analogue. These requirements are divided into several subtopics: how to change the current infrastructure both at the broadcast station and at home, how to assure a smooth transition, and what the specific requirements for the home receiver are. Sections 2.2 - 2.4 study each of these requirements more in detail. Section 2.2 describes the new infrastructure needed to transmit and receive digital television. In Section 2.3, different open digital television standards, which assure a smooth transition for the benefit of users, are presented. Section 2.4 studies the requirements for the home receiver, taking into account different regional standards. Section 2.5 describes the current state and limitations of digital television receivers, focusing on MHP. Finally, Section 2.6 takes a look towards the future, thus answering the first research question.

2.1 Requirements

A number of requirements can be identified related to digital television technology. In this section, the requirements are divided into three different categories: infrastructure, smooth transition, and receiver functionality. The following list explains them in more detail.

- **Infrastructure:** digital television technology implies an equipment upgrade, both for the broadcaster and at home. The broadcaster needs to transmit not only the A/V stream, but data as well within the MPEG-2 stream. At home, the MPEG-2 stream has to be received and processed.

- **Smooth Transition:** today, the most basic and important requirement is to assure a smooth transition from analogue television to its digital counterpart. The four main requirements in this category are:

  - **Horizontal-Market:** the decisions of governments, broadcasters, institutions, independent consortia, and receiver manufacturers towards the creation of open and interoperable standards meet this requirement [23].
  - **Easiness of Use:** digital television must take into consideration the technical skills of its viewers, a heterogeneous group with very different backgrounds (e.g., elderly people versus teenagers) [42].
  - **Price:** taken into account that most households have at least two televisions, the amount of money needed for the transition should be minimal. One alternative followed by some governments (e.g., Italy, Berlin) is to provide financial subsides.
  - **Viewers Expectations:** users will adopt digital television only if they can enjoy its benefits. Basic improvements include:
    - **Better Quality:** ”Picture quality was another important factor in the decision to go digital, particularly in areas where reception quality was bad” [35].
    - **Content:** ”their main motivation to subscribe was for more channels” [35]. In addition, the CENELEC report identifies multiple channels as an important element in attracting consumers [23].
    - **Services:** a Counterpoint study found that interactive services were a ”nice extra” for the users [35]. As confirmed

by the CENELEC report: ”there are no broadcasters or operators generating significant revenues from interactive services” [23]. Nevertheless, it is important to notice that in both documents, Electronic Program Guide (EPG) / Navigator and Super Teletext are considered basic requirements. For example, in the section of the Counterpoint study referring to consumer attitudes towards EPG: ”...they did not see it as a product or service, but rather as simply the new way of finding programmes to watch” [35].

- **Receiver:** the first approach to attract consumers is to provide basic capabilities that meet viewer expectations at the lowest possible price. The most basic requirement is the support of the broadcasted A/V stream and subtitles. Some extra capabilities include conditional access, return channel mechanisms, storage capacity for DVR, and Java application support.

Table 2.1 summarises the identified requirements for digital television systems based on the previous list. In addition, it includes the solutions currently adopted to meet them.

2.2 Infrastructure

Digital television implies a change in infrastructure, both at the broadcast station and at home. First, the broadcast stations need to update their equipment so that they can transmit an MPEG-2 stream including A/V signal and data. Then, viewers need a receiver that supports the digital television stream. In this section, the broadcast system is explained by using Otadigi[2] as a reference. Otadigi is a terrestrial digital television channel operating since December 2002 in the Helsinki University of Technology campus area. The basic architecture of the receiver is described based on the functionality it is expected to perform.

Broadcast System

Figure 2.2 depicts a typical example of a terrestrial digital television broadcast system [P6]. It is composed of the following components: MPEG-2 encoder, DVB Asynchronous Serial Interface (ASI) Internet Protocol (IP) link pair, gateway server, remote control/monitor unit, object carousel, multiplexer, modulator/transmitter, and antenna. First, the A/V stream is encoded with the MPEG-2 encoder. Because people in several locations (e.g., students, researchers) will encode their A/V content, the encoder is stored in a mobile rack and connected to the broadcasting system by using the DVB-ASI IP links. The object carousel[3] contains application code and data. It can be uploaded using the gateway server. Next, the multiplexer generates the final MPEG-2 transport stream by combining the outputs of the object carousel and the DVB-ASI IP link. Finally, the modulator/transmitter feeds

[2]http://www.otadigi.tv

[3]A data carousel is an object carousel, in which the information is not structured.

Table 2.1: Current Basic Requirements of Digital Television.

Requirements	**Current Solutions**
Infrastructure	
Home	
Receive TV	Antenna
Watch TV	TV set
Tune to DTV Stream	Set-top box, Integrated receiver
Broadcast System	
Audio/Visual Stream	MPEG-2 encoder
Data Stream (e.g., applications)	Object / Data carousel
Transmission of MPEG-2 stream	Multiplexer / Modulator / Transmitter / Antenna
Smooth Transition	
Horizontal-Market	Use of standards agreed by major players
Ease of Use	Usability considerations
Price	Lower prices (e.g., governmental subsidies)
Viewer Expectations	
Video and Audio	Better quality
Content	Increased number of channels
Services	Navigator / Super Teletext
Receiver	
Audio/Visual Support	Visual stream (MPEG-2)
Subtitle Support	Text over video
Application Support	Procedural / Declarative
Storage Capacity	Hard drive
Conditional Access	Security mechanism
Return Channel	Analogue modem
World Wide Web Access	XML user agent

the multiplexed MPEG-2 transport stream to the antenna. The remote control/monitor can be used to monitor the whole system.

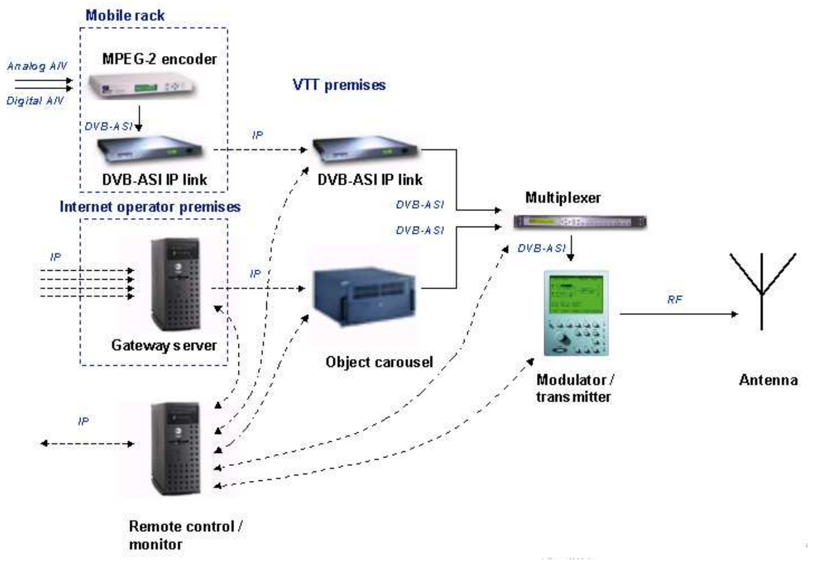


Figure 2.2: Broadcast System Architecture.

Home

The digital television receiver is designed to receive and decode broadcast television services. It tunes to the required channel, extracts and decodes the selected data, checks the access rights of the user, and outputs picture, sound, and services. Figure 2.3 depicts the basic architecture of a digital television receiver. It can be divided into five main blocks: tuner / demodulator, demultiplexer, conditional access system, decoders, microcontrollers, and I/O devices [12, 30, 40, 82, 127].

First, the transport stream coming from the antenna (i.e., RF input) is tuned and recovered as an MPEG-2 stream by the tuner/demodulator block. Then, the MPEG-2 stream is demultiplexed into audio, video, subtitles, and data streams. The demultiplexing is controlled by the conditional access subsystem, which controls the user rights to the broadcasted material. After decoding each stream separately (i.e., video, audio, subtitle, and data), the resulting information is fed to the microcontroller. The microcontroller embeds the software architecture, controls the I/O ports, and manages the whole functionality of the receiver.

2.3 Smooth Transition: Open Standards

Four major requirements needed to ensure a smooth transition to digital television have been defined: horizontal-market, ease of use, price, and user expectation. Because a thesis should be kept within the boundaries of the

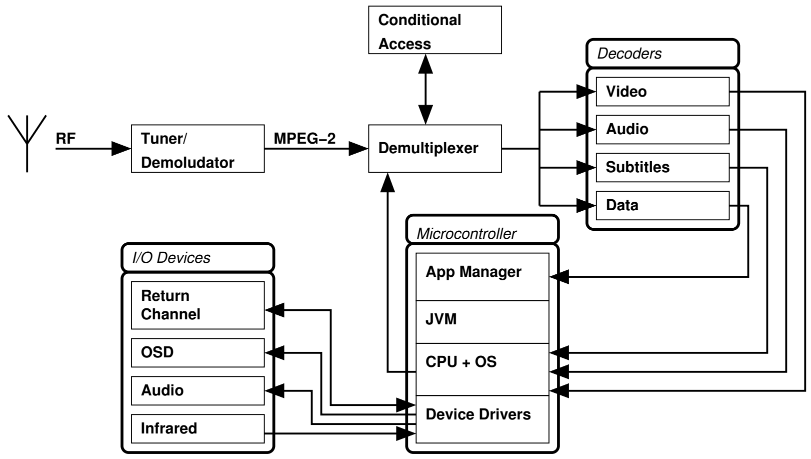


Figure 2.3: Receiver Architecture [36].

research questions, this section only studies the different open standards developed around the world in order to achieve a horizontal-market model for digital television. This thesis pays special attention to the DVB standardisation procedure because of its guiding principles:

- **Market-led:** It is not worth developing standards if the market is not ready to accept them.
- **Interoperability:** Assuring viewers that all systems on the market behave the same way.
- **Open:** The standard is agreed, approved, and published by a recognised standards body so they are available at minimal cost worldwide.

Digital Television Transmission

Regarding the transmission of digital television, there are three major regional standards: Advanced Television Systems Committee Advanced (ATSC) in North America, Integrated Services Digital Broadcasting (ISDB) in Japan, and DVB in Europe. In addition, at the time of writing this thesis, China is developing its own standard, Digital Multimedia Broadcasting - Terrestrial (DMB-T), for terrestrial transmission. Even though all the standards rely on MPEG-2 streams to deliver digital television content, the audio format and video resolution allowed are not the same, as shown in Table 2.2. Other major differences are related to modulation, bandwidth, and video rate. For example, the modulation used in ATSC terrestrial is 8 Vestigial Side Band (8-VSB), the poor performance of which made Taiwan change their decision and adopt the DVB-T transmission standard instead.

DVB is a consortium of around 300 broadcasters, manufacturers, network operators, and regulatory bodies officially inaugurated in 1993. They have come together to establish a common European standard for digital television, defining DVB-S [47] for satellite, DVB-T [53] for terrestrial, and DVB-C [49] for cable transmission. During 2004, DVB also published

Table 2.2: Overview of Broadcast Technologies [7, 10, 57, 65, 103, 113, 134].

	DVB	**ISDB**	**ATSC**
Audio Coding	MPEG-2 Audio	MPEG-2 AAC	Dolby AC-3
Video Config	MPEG-2 Video		
Multiplexing	MPEG-2 Transport Stream		

Video Format (i) = interlaced (p) = progressive	1920x1080
	
	25Hz:
	
	720x576
	544x576
	480x576,288
	
	30Hz:
	720,640,544,
	448,325x480
	352x240
		1920x1080(i)
		1280x730(p)
		
		720x480(p)
		
		720x480(i)
			1920x1080(p)
			1980x1080(i)
			
			1280x720(p)
			
			704x408(p)
			640x480(p)

DVB-H, which extends concepts from DVB-T to handheld terminal reception. In addition, DVB has standardised other transmission-related issues such as:

- **Service Information System (DVB-SI):** to comprehensively identify transport communication and service content [55].
- **Access Common Interface (DVB-CI):** to restrict unauthorised access to programs.
- **Teletext transmission transport system (DVB-Text):** to carry a fixed-format teletext system in the MPEG-2 transport stream [46].
- **DVB Subtitling System:** to allow the user to see different subtitles [50].
- **DVB Interaction Channel Public Networks:** to include networks such as PSTN [48].

Software Middleware

Today there are a number of alternatives to receiver software middleware. Some of them are proprietary systems (e.g., MediaHighway[4], OpenTV[5]), but this thesis would rather concentrate on open standards to meet the horizontal-market requirement. Nevertheless, as with transmission mechanisms, open standards vary according to region, as shown in Figure 2.4:

- **MHP:** defined by DVB for terrestrial, cable, and satellite environments [51, 52].

[4]http://nds.com/middleware
[5]http://www.opentv.com

- **DTV Application Software Environment (DASE):** defined by ATSC for North American terrestrial transmission [8].

- **OpenCable Application Platform (OCAP):** defined by CableLabs for North American cable transmission [21, 20].

- **Advanced Common Application Platform (ACAP):** the latest North American initiative for the harmonisation between OCAP and DASE standards [9].

- **STD-B23/STD-B24:** defined by the Association of Radio Industries and Businesses (ARIB) for Japanese digital television receivers [3, 4].

In 1997, DVB started a subproject, called MHP, in order to define a platform-independent interface between applications and receivers (i.e., a common middleware software). The guiding principles (i.e., market-led, interoperability, and open) ensure that the user can purchase any MHP-compliant receiver on the market and it will support all the broadcasted services. The major components of MHP are the following [139]:

- **DVB-J platform:** consists of a Java Virtual Machine (JVM) and a set of digital television-specific APIs.

- **Content formats:** includes image, video subtitles, and resident and downloadable fonts.

- **DVB-HTML:** allows the visualisation of XHTML-based content.

- **Network:** defines both broadcast access and interaction channel usage.

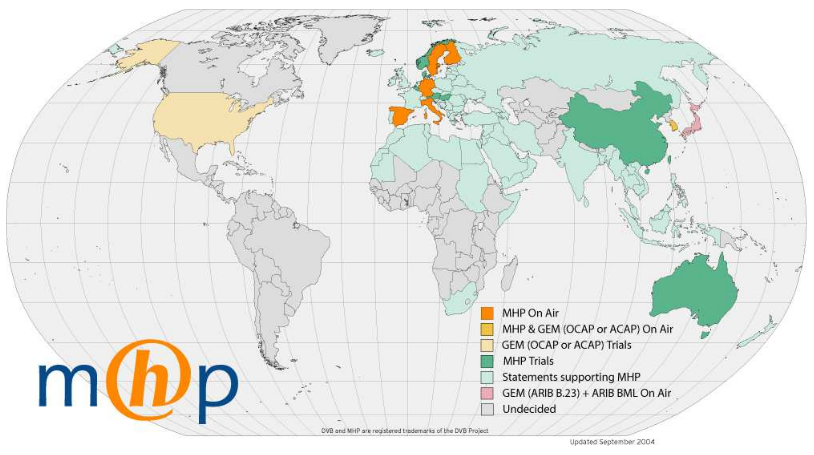

MHP On Air
MHP & GEM (OCAP or ACAP) On Air
GEM (OCAP or ACAP) Trials
MHP Trials
Statements supporting MHP
GEM (ARIB B.23) + ARIB BML On Air
Undecided
mhp
DVB and MHP are registered trademarks of the DVB Project
Updated September 2004

Figure 2.4: Worldwide View of Standards for Receivers. Source: MHP (http://www.mhp.org, April '05).

MHP middleware defines two different languages for implementing broadcasted applications: DVB-J and DVB-HTML. The former is based on Java and is intended for interactive applications, while the latter is a subset of XHTML intended for information services (e.g., Super Teletext).

Currently, the major challenge is to harmonise all the different regional open software middleware solutions into a common core framework. With that intention in mind, DVB formed the MHP Umbrella Group (MUG), whose task was the definition of such a framework, i.e., GEM [54]. The principles guiding the conception of GEM are:

- To maximise interoperability between different organisations.
- To maximise the presence of MHP components.
- To take into account local business and technical constraints.

GEM was defined by removing from MHP 1.0.2 these-region specific parts (i.e., DVB-dependent parts), thus introducing a worldwide core software middleware definition. As such, GEM cannot be considered a standalone specification, but each region should incorporate the functional equivalents of these removed parts (i.e., service information and conditional access). Figure 2.5 depicts GEM as described in the following list:

- **STD-B23 (ARIB):** describing its own service information description and network protocols.
- **OCAP (CableLabs):** apart from its own service information and conditional access, OCAP provides support for native, unbound, and monitor applications. Native applications are those written for a specific host, while unbound applications are those not dependent on a specific channel (e.g., e-mail). Finally, the monitor application is responsible for managing the lifecycle of the unbound services, thus making the network operator responsible for the state of the receiver. For example, it can upgrade an application stored in the receiver.
- **ACAP (ATSC and CableLabs):** it will include features from DASE and OCAP. Basically, it combines Java from OCAP and HTML from DASE. Still, ”the formal relationship between OCAP and ACAP remains to be decided” [109].

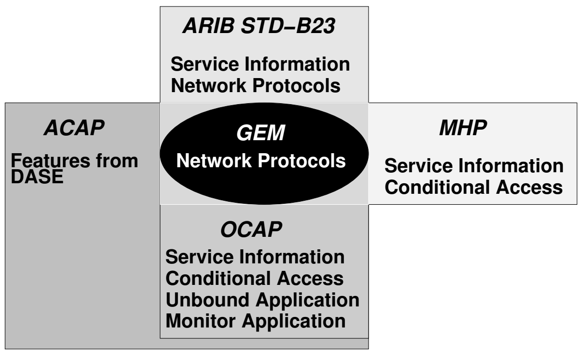


Figure 2.5: GEM Specifications [109].

2.4 Evolution of Digital Television Receivers

This section studies the evolution of the receivers from the standardisation point of view. Thus, it studies how the different regional initiatives have considered the requirements for digital television receivers presented in Table 2.1. These requirements include A/V, subtitle, and application support, storage capacity, conditional access, the return channel, and WWW access.

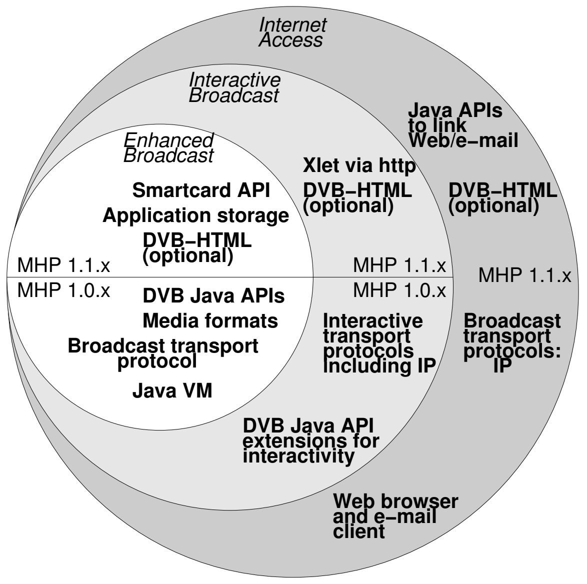


Figure 2.6: MHP Profiles [108].

MHP defines three different profiles so that the transition from analogue television follows gradual steps. These profiles, depicted in Figure 2.6, are:

- **Enhanced Broadcast:** intended for accessing broadcasted services with local interactivity.
- **Interactive Broadcast:** for real interactive applications supporting the interaction channel.
- **Internet Access [120]:** includes Java APIs for Web browsers and e-mail clients and IP broadcast transport protocol.

MHP decided to publish two standards: MHP 1.0 [51] and MHP 1.1 [52]. Because of their similarities, both the Enhanced and Interactive Broadcast profiles are included in MHP 1.0. On the other hand, MHP 1.1 introduces, in the Internet Access profile, a number of extensions (e.g., persistent storage of applications, application download over the interaction channel, smartcard API) [121]. Currently, the MHP test suite used to evaluate the compliance of commercial set-top boxes is based on MHP 1.0.2 (i.e., MHP 1.0 version 1.2.1). Hence, manufacturers have been so far concentrating on MHP 1.0 and waiting for a future market for MHP 1.1.

Similarly to MHP, OCAP decided to publish two profiles: 1.0 [21] and 2.0 [20]. The first one is based on MHP 1.0.2, but takes into account cable

Table 2.3: Receiver Capabilities in Different Regional Standards.

	MHP		**OCAP**		**DASE Level**			**ARIB**
	1.0	1.1	1.0	2.0	1	2	3	
Audio/Visual	X	X	X	X	X	X	X	B24
Subtitles	X	X	X	X	X	X	X	B24
Applications								
Procedural	X	X	X	X	X	X	X	B23
Declarative	-	X	-	X	X	X	X	B24
Return Channel	X[6]	X	X	X	-	X	X	B21/ B24
Conditional Access	X	X	X	X	-[7]	-	-	B21/ B38[8]
Storage Capacity	-	X	X[9]	X	X[10]	X	X	B38
Internet Access	-	X	X[11]	X	-	-	X	B21/ B24

transmission specifics. Hence, it includes return channel mechanisms and storage capacity. The second one, based on the MHP 1.1 series, introduces a declarative application environment. DASE, on the other hand, decided to produce three standards: DASE Level 1 (DASE-1) [8], DASE Level 2 (DASE-2), and DASE Level 3 (DASE-3). The first defines both a declarative and a procedural environment, but it is only intended for broadcasted content. The second will consider return channel capabilities, while the last will include Internet access. In Japan, ARIB standardises both procedural and declarative environments in STD-B23 [3] and STD-B24 [4], respectively. In addition, STD-B21 [5] describes the configuration of the receivers and STD-B38 [6] specifies the home servers.

In conclusion, all the regional standards take into consideration the basic requirements for digital television receivers such as video, audio, application, and subtitle support, storage capacity, bi-directional channels, and conditional access mechanisms. Table 2.3 shows a comparison of the profile of each standard that includes these requirements.

2.5 Current State - Multimedia Home Platform

Currently, a maturity period regarding terrestrial digital television and MHP has started. Some regions like Finland, Berlin, Italy, Korea, and Catalonia already broadcast applications together with television content. At the same time, MHP authoring tools are spreading; some examples include Evolu-

[6]Interactive Broadcast Profile.

[7]"This specification does not define or rely upon the use of conditional access facilities" [8]. Yet it is included in ATSC Doc A/70: "Conditional Access System for Terrestrial Broadcast".

[8]Defines Super Conditional Access for copyright protection. In addition, it includes a payment control system.

[9]For example, the Monitor application.

[10]Local file system (e.g., org.atsc.registry).

[11]An unbound application can be, for example, a WWW browser.

tion Console from Osmosys[12], Sofia Tools from Sofia Digital[13], Cardinal Studio from Cardinal System[14], and AltiComposer from Alticast[15]. Moreover, open-source MHP implementations can be downloaded from the Web (e.g., OpenMHP[16], mhp4free[17], XletTView[18]) for individuals that want to learn about the subject. Universities are including DVB and MHP in their curricula, and books specialising in the topic are being published [38, 112, 127, 152]. This section reviews the status of MHP at the time of the writing of this thesis. First, it studies MHP-compliant receivers and describes broadcasted applications. Then current limitations are outlined.

Receivers

In the last years, the number of MHP-compliant receivers has increased. These receivers implement the Interactive Broadcast profile of MHP 1.0.2. Hence, they support DVB-J, an interaction channel via analogue modem, and an application manager. Some of the receivers include I-can from Advanced Digital Broadcast (ADB)[19], Mediamaster 310T from Nokia[20], DTR 4600 from Philips[21], and DTB-9500F from Samsung[22]. New models include better return channel connections (e.g., LAN) and digital video recorder capabilities (e.g., 7100TX model from I-can). There are, as well, several MHP stacks developed for commercial use, such as those from Osmosys, Sony, Motorola, and Zentech. Finally, regarding the hardware configuration, receivers include Flash memory from 8MB to 32MB for storing system software, 16MB to 64MB of RAM for running applications, and chip processors with MPEG-2 decoders and CPUs of around 250MHz or less[23].

Applications

The development of MHP applications has become a lucrative business, as can be seen from the number of graphical authoring tools available on the market (e.g., Cardinal Studio). The range of available services in digital terrestrial broadcast networks is ample; some examples include games, commercials, weather, home shopping and banking, EPG, and Super Teletext services. Please refer to the report "Analysis of the Current Situation" [92] from The MHP Knowledge Project for a complete study on the issue. As pointed out in the requirements table, Table 2.1, at this moment the two basic applications for digital television are EPG and Super Teletext. The former shows information related to the scheduled A/V content. The latter is a new version of traditional teletext in which multimedia content is broadcasted within the MPEG-2 stream as an MHP application, instead of using

[12]http://www.osmosys.com
[13]http://www.sofiadigital.com
[14]http://www.cardinal.fi
[15]http://www.alticast.com
[16]http://www.openmhp.org
[17]http://www.mhp4free.net
[18]http://xletview.sourceforge.net
[19]http://www.adbglobal.com/
[20]http://www.nokia.com/
[21]http://www.philips.com/
[22]http://www.samsung.com/index.htm
[23]For example, ADB's 7100TX uses STi5100 (http://www.us.st.com/stonline/books/pdf/docs/10421.pdf).

the vertical blanking interval.

Figure 2.7 depicts an application broadcasted in the Otadigi network. The application is a television portal for Red Cross[24], but it follows basic digital television design guidelines that can be generalised as essential concepts. The application layout is divided into different frames for visibility purposes. First, on the left, different sections can be navigated using the remote control up and down buttons. As the user navigates through the sections, an icon at the bottom left indicates his current location within the portal. In addition to the icon, the title at the top of the current page changes as the user navigates. The content of each section is displayed to the user in the middle of the screen. When the full text and images do not fit, the content is divided into different pages. Finally, at the bottom of the screen, the colour buttons provide specific navigational functionality: go to the home view (red), go to previous page (green), go to next page (yellow), and exit the application (blue). Following usability principles, only those colour buttons that can be used at a specific moment are fully visible to the user; the rest are disabled.

There are several important lessons to be learnt from this example. First, because the number of colours defined in DVB-J is limited[25], the static information, such as the frame structure, the background, the colour buttons, and the static icons, is developed using an MPEG-2 I-frame. In addition, the rest of the images used in the application are dithered to the DVB-J palette. The smallest font used in the application is 18 points high, otherwise the viewer cannot see the text from the sofa. Also, Tiresias type is used. Finally, the safe area defined by MHP is taken into consideration, which is why the amount of information that can be displayed is limited.

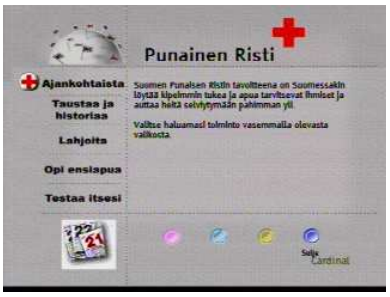
Limitations

The limitations of current commercial MHP-compliant digital television receivers are the following:

- **Start-up time:** even though manufacturers have improved the current models in relation to their first ones, the start-up time of the applications is not optimal yet.
- **Interaction channel capabilities:** currently most of the commercial receivers use analogue modem, which makes accessing other content than the broadcasted content extremely slow.
- **Usability issues:** in my opinion manufacturers have not yet carried out enough usability studies concerning the receivers, meaning that it is difficult for people with no technical skills to use the receivers.
- **MHP 1.0.2 compatibility:** since the MHP suite in use is based on MHP 1.0, MHP 1.1 features are not included in current set-top boxes. These include storage capacity, Web and e-mail clients, smartcard APIs, and XML parsers.

[24]The original developers of this application were Jami Vilpponen, Lin Quan, Piia Pulkinen, and me. The application was developed during the course T-111.077 ”Special Project on Content Production: Digital Television” (2002-2003).

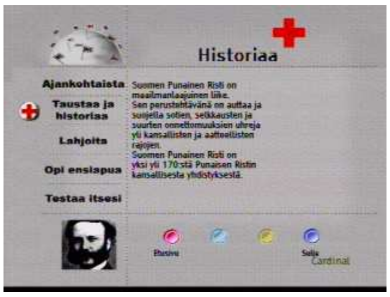
[25]April 2005: MHP 1.1.2 [58] includes RGB 4444 as minimum.



Suomen Punaisen Ristin tavoitteena on Suomessakin löytää kipeimmin tukea ja apua tarvitsevat ihmiset ja auttaa heitä selviytymään pahimman yli.

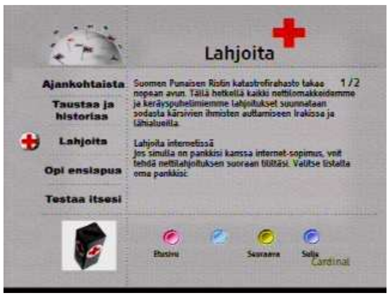
Valitse haluamasi toiminto vasemmalla olevasta valikosta.

(a) Current News Page.



Suomen Punainen Risti on maailmanlaajuinen liike.
Sen perustehtävänä on auttaa ja suojella sotien, selkkausten ja suurten onnettomuuksien uhreja yli kansallisten ja aatteellisten rajojen.
Suomen Punainen Risti on yksi yli 170:stä Punaisen Ristin kansallisesta yhdistyksestä.

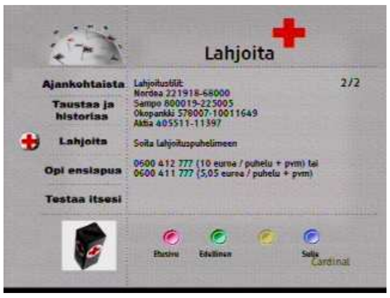
(b) History of Red Cross.



Suomen Punaisen Ristin katastrofirahasto takaa nopean avun. Tällä hetkellä kaikki nettilomakkeidemme ja keräyspuhelimiemme lahjoitukset suunnataan sodasta kärsivien ihmisten auttamiseen Irakissa ja lähialueilla. 1 / 2

Lahjoita internetissä
Jos sinulla on pankkisi kanssa internet-sopimus, voit tehdä nettilahjoituksen suoraan tililtäsi. Valitse listalta oma pankkisi.

(c) Contribute to Red Cross (Page 1).

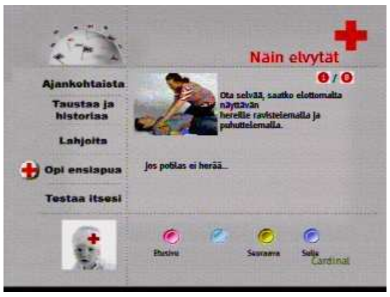


Lahjoitustilit: 2/2
Nordea 221918-68000
Sampo 800019-225005
Okopankki 578007-10011649
Aktia 405511-11397

Soita lahjoituspuhelimeen

0600 412 777 (10 euroa / puhelu + pvm) tai
0600 411 777 (5,05 euroa / puhelu + pvm)

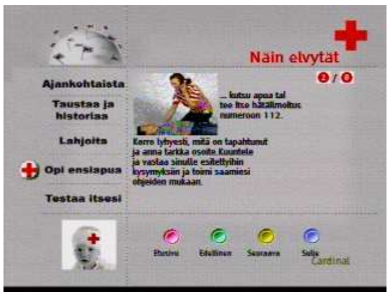
(d) Contribute to Red Cross (Page 2).



Ota selvää, saatko elottomalta näyttävän herellle ravistelemalla ja puhuttelemalla.

Jos potilas ei herää...

(e) First Aid Online Course (Page1).



... kutsu apua tai tee itse hätäilmoitus numeroon 112.

Kerro lyhyesti, mitä on tapahtunut ja anna tarkka osoite. Kuuntele ja vastaa sinulle esitettyihin kysymyksiin ja toimi saamiesi ohjeiden mukaan.

(f) First Aid Online Course (Page2).

Figure 2.7: Red Cross Application Running in Otadigi.

- **DVB-HTML:** its adoption in the short term is quite improbable[26]. Some products such as Pontegra from Nionex[27], ACE Browser from Ortikon[28], Escape from Espial[29], and Osmosys browser claim to be DVB-HTML compliant. The problem is that because there is no test suite for MHP 1.1, true complete compliance cannot be assured.

2.6 A Look Towards the Future

In the future, when both digital television and MHP 1.0 have reached the majority of the population, there will be a group of people demanding high-end receivers. In this situation, the receiver will truly become another multimedia platform at home. Thus, its convergence with other devices will be a major challenge. Some of the topics that should be taken into account are:

- **Storage Capacity (MHP 1.1 compatible):** including storage capacity in the receivers will permit new services such as downloadable games and DVR. One important related issue is how to manage the digital rights of the distributed content.

- **Conditional Access (MHP 1.1 compatible):** in order to receive pay-per-view channels and events.

- **WWW Access (MHP 1.1 compatible):** there are a number of requirements for making the receiver WWW capable. First, the interaction channel mechanism should be upgraded (e.g., to LAN). Second, XML parsers should be included locally. Finally, the browser client specified in the Internet Access profile should be implemented.

- **Streamed Media (MHP 1.1 extension):** apart from the broadcasted A/V content, media could be streamed to receivers via any network (e.g., broadcast, interaction channel). In order to support these media, apart from upgrading the interaction channel, a video player capable of showing formats other than MPEG-2 is required[30].

- **3D Graphics Support (MHP 1.1 extension):** 3D graphics are becoming more important, and their support in receivers is probably a matter of time. One can think of three different approaches: first, convergence with game consoles, in which special video cards with hardware acceleration and use of more efficient languages than Java are required. Second, 3D-based widgets are already provided by some MHP authoring tools, since only basic graphics functions (e.g., polygons) are required. Finally, 3D graphics concepts based on Open GL (e.g., lighting, texture) can be provided at the Java level (e.g., Java 3D) and at the declarative level (e.g., X3D).

[26]April 2005: MHP 1.1.2 [58] says that it relevance in today's market is unclear.
[27]http://www.nionex.com
[28]http://www.ortikon.com
[29]http://www.espial.com
[30]April 2005: MHP 1.1.2 [58] includes support for multiple video decoders.

2.7 Summary

This chapter has explained the fundamentals of digital television. First, it presented the results of the regional organisations involved in the standardisation of digital television: ATSC and OpenCables in the USA, DVB in Europe, and ARIB in Japan. In addition, because the existence of open standards will assure a smooth transition from analogue television, the worldwide initiative started by GEM has been introduced. Then, the current state of digital television systems was surveyed and a look towards the future was proposed, identifying future extensions to the MHP standard. These extensions include streamed media and 3D graphics support and WWW access, which will be analysed in more detail in the rest of this thesis.

3 CONTENT AUTHORING FORMATS AND APPLICATION ENVIRONMENT FOR DIGITAL TELEVISION

Chapter 3: Content Authoring Formats
(Definition of a worldwide App. Env.)
Compiled Languages (C)
Virtual Machine Languages (Java)
Markup Languages (XML)
Application Environment for DTV

The previous chapter started the discussion about the digital television environment. It covered the following topics: transmission mechanisms, receiver configurations, current limitations, and possible improvements. In addition, the major regional standardisation bodies and their software middleware proposals were introduced.

This chapter focuses on the authoring content formats that digital television service providers can use for implementing applications. First, Section 3.1 defines interactive multimedia. Based on this definition, a set of metrics for multimedia authoring formats are presented in Section 3.2. Section 3.3 proposes a taxonomy of content authoring formats, dividing them into programming and markup languages. Then, examples of each category are reviewed and compared, resulting in an study of their potential use for digital television services in Section 3.4. Finally, common procedural, declarative, and application environments for digital television receivers are proposed in sections 3.5, 3.6, and 3.7, respectively.

3.1 Multimedia and Interaction

Developing a new multimedia taxonomy is a task beyond the scope of this thesis. Hence, this section, based on a literature review, aims to define multimedia and interaction. In the literature, multimedia is defined as [158]: "*an interactive computer-mediated presentation that includes at least two of the following elements: text, sound, still graphic images, motion graphics, and animation*". This widely accepted definition sounds convincing, but what does media actually mean? What are the real differences between text and graphics if both are received visually?

A deeper study of multimedia foundations defines media as a combination of signs, and takes into account their modality, nature, and syntax [143]. First, the basic purpose of multimedia is to transmit information from a source to a recipient [158]. This information is later perceived by the recipient through her senses (i.e., modality). Second, the information is composed of a number of signs. These signs can be categorised, depending on their abstraction level (i.e., nature of the sign), into concrete, iconic, and symbolic. Thus, any common media type (e.g., image) can be described based on modality and nature of the sign. For example, a photorealistic im-

Table 3.1: Proposed Media Taxonomy, Based on [78, 143, 166].

	Moda-lity	**Nature of the Sign**			**Syntax**	
		Concrete Iconic	**Abstract Iconic**	**Symbolic**	**Spatial**	**Tem-poral**
Audio	aural	yes	yes	yes	yes	yes
Graphic / Image	visual	yes	yes	no	yes	no
Text	visual	no	no	yes	yes	no
Animation	visual	yes	yes	yes	yes	yes
Video	visual	yes	yes	no	yes	yes

age is a visual concrete sign and a written word is a visual symbol. Finally, in order to acquire a meaning, these signs have to be organised in a valid manner (i.e., syntax).

As a conclusion, different media can be categorised based on the following concepts, as shown in Table 3.1: modality, nature of the sign, and syntax. For example, a show featuring 3D talking heads will fall into animation + audio media, while a movie with subtitles will be animated text + audio + video.

Another essential characteristic of multimedia is its interaction. The level of interaction provided by a service can be seen on a continuum ranging from zero to one [158], as depicted in Figure 3.1. Zero interaction stands for non-interactive presentations, such as films shown in a movie theatre, in which the user just pays for a ticket, sits down in his seat, and receives information. At the other end of the spectrum, interaction level one stands for highly interactive communications such as face-to-face communication. In between levels one and zero, one can differentiate activities such as scrolling a Web page, following a hyperlink, or playing a video game.

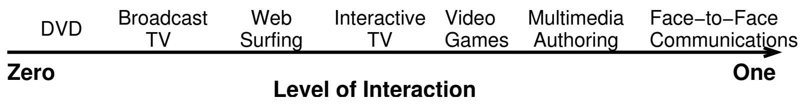


Figure 3.1: Levels of Multimedia Interaction.

According to Aleem, the level of interactivity can be divided into the following categories [1]: passive, reactive, proactive, and directive. Table 3.2 shows an example based on text media.

- **Passive:** the multimedia presentation is visualised to the user, but she does not have any control over it.
- **Reactive:** the user has limited interaction (e.g., turning pages or using scroll pane functionality).
- **Proactive:** the user can, for example, choose a path or make selections (e.g., clicking a button).
- **Directive:** corresponds to user authoring of information.

Table 3.2: Example of Levels of Interaction [1].

	Passive	Reactive	Proactive	Directive
Text	Delivers information	Page turner, change language	Browse, hypertext, fixed anchors	Prompt user for information
Audio	Delivers audio	Change volume	Change radio station	Create new sounds

As a conclusion, interactive multimedia services must include four main elements:

- **Nature of the signs:** which kinds of signs are presented to the user.
- **Modality:** the sense through which the information is acquired.
- **Syntax/Arrangement:** how the signs/symbols are spatially and temporally arranged.
- **Interaction:** level of interaction of the service.

3.2 Metrics

After defining interactive multimedia, the next challenge is to identify a group of measurements that can later be used to categorise content authoring formats. In this thesis, the following measures are taken into account:

1. Difficulty of Use: measures the programming skills needed for learning how to use the authoring format.
2. Expressive Power: measures how much a service developer can do with the authoring format.
3. Interoperability: measures how dependent a format is on the running environment.
4. Safety of Distribution: the user must completely trust consumer products.
5. Supported Media Types: as defined in Table 3.1, media types include: audio, text, graphic/image, animation, and video.
6. Arrangement of the signs:
   (a) Temporal layout: when different media types are visible or hidden.
   (b) Spatial layout: where different media types are placed.
7. Interaction:
   (a) Passive: zero interaction.
   (b) Reactive: limited interaction, such as turning pages.

(c) Proactive: advanced interaction, for example, selecting a path or following hyperlinks.

(d) Reciprocal: full interaction. The authorial control over the information shifts to the recipient.

3.3 Terminology

This thesis divides authoring content formats into programming languages and markup languages. Programming languages are those in which the developer programs how to solve a problem (e.g., Java and C languages), while in markup languages the developer states what is to be solved. Because markup languages have a higher-level of abstraction than programming languages, they require little programming experience, are highly interoperable, and their expressive power is lower [138]. In addition, this thesis divides programming languages into compiled and Virtual Machine (VM) languages. The source code of compiled languages has to be compiled for the settings of the device where they are going to run. On the other hand, a VM is a software component that emulates its host computer. Hence, compiled languages lack the benefits of a VM, such as interoperability and safety of running the code, but they do not require the same overhead.

In addition to the authoring content formats, this chapter studies the application environment for digital television. The application environment is a software middleware that is capable of running services and is composed of a procedural part and a declarative part. The procedural environment supports the processing of applications that primarily use procedural information to express their behaviour (i.e., the programming languages domain), while the declarative environment supports the processing of applications that primarily use declarative information to express their behaviour (e.g., the markup languages domain). In this thesis, the terms ”procedural environment” and ”execution engine” are equivalent, as are the terms ”declarative environment” and ”presentation engine”.

3.4 Content Authoring Formats

After a discussion of the terminology used in this thesis, this section compares the most relevant content authoring formats for digital television applications. First, compiled (e.g., C) and VM (e.g., Java) programming languages and markup (XML-based) languages are reviewed. In addition, two other multimedia solutions, MPEG-4 and Multimedia and Hypermedia Experts Group (MHEG), are introduced. Finally, this chapter compares them based on the metrics identified in Section 3.2.

Compiled Programming Languages: C

One major example of compiled languages is C. Unfortunately, the number of special-purpose libraries written in C make it impossible, within the scope of this thesis, to cover all of them. However, some interesting libraries for the digital television environment include linuxstb[1], linuxtv[2], and libsofmpeg[3]. Both linuxtv and linuxstb provide software intended for tuning television channels (e.g., *tzap* or *dvbtune*) and downloading content (e.g., *dvb-data*), while libsoftmpeg software decodes the MPEG-2 A/V stream. Regarding multimedia capabilities, two other interesting initiatives are Simple DirectMedia Layer (SDL)[4] and DirectFB[5]. SDL, which is widely used for game development, is a cross-platform library providing multimedia primitives and an abstract canvas. DirectFB, which started as an alternative to X-Window for digital television receivers, has currently evolved into a complete windowing system rendering directly in the framebuffer memory.

Another multimedia API considered in this thesis is OpenGL[6]. Currently, OpenGL has become the de facto standard for 3D graphics, providing a comprehensive set of primitives (e.g., camera position, textures), which permit software developers to implement 2D and 3D graphics applications. Most recently, Khronos Group has defined OpenGL ES [14]. This standard is based on OpenGL 1.5 and is intended for mobile and embedded devices (e.g., digital television receivers).

Virtual Machine Programming Languages: Java

The most significant example of a VM language is Java. Java was conceived as a tool to produce software for networked consumer devices [62], so portability and reliability were the main goals. Java programs run on top of a VM [97], an abstraction of the computing environment. The use of a VM provides platform independence and restricts potential security attacks. In order to support Java, a device requires the JVM and must include general- (e.g., *java.lang*) and specific-purpose (e.g., *java.awt* and *javax.media*) APIs. This subsection concentrates on APIs related to the digital television environment, which include Java TV, Abstract Windowing Toolkit (AWT), Java Media Framework (JMF), and APIs for 3D graphics.

Sun Microsystems defines the application lifecycle of digital television services in JavaTV[7] API [22]. When the user launches a JavaTV service, called Xlet, the receiver's application manager downloads and controls its lifecycle. As part of the general API set, Java includes an AWT package for 2D graphics and user interface development. AWT is in charge of handling user interaction, so developers can provide reactive, proactive, and even directive levels. Second, AWT supports graphics and text media types by providing basic graphics primitives, and even allows one to program animations by using threads. Finally, JMF [73] is a Java package intended for controlling video and audio media types, providing a temporal dimension in Java. Basically, JMF allows the developer to create A/V players in Java and control their behaviour.

In addition to supporting A/V content, 2D graphics, and user interac-

[1]http://cvs.sourceforge.net/viewcvs.py/dvbtools
[2]http://www.linuxtv.org
[3]http://www.linuxtv.org/cgi-bin/viewcvs.cgi/libsoftmpeg
[4]http://www.libsdl.org
[5]http://www.directfb.org
[6]http://www.opengl.org
[7]http://java.sun.com/products/javatv

Table 3.3: Java Support for Multimedia.

	Java API
Media Types	
Audio	JMF
Text	AWT
Images	AWT
Animation	AWT + Threads
Video	JMF
Arrangement of the Sign	
Spatial	AWT Layout
Temporal	JMF / Threads
Interaction	AWT Events

tion, Java defines APIs for 3D graphics support. Mainly, there are two alternatives for developers: Java3D and Java bindings for OpenGL. The first option, Java3D[8] is a complete API for developing stand-alone applications. Java 3D specifies a high-level API, but depends on lower-level implementations (e.g., OpenGL or Direct3D) for the rendering. The second option, Java bindings for OpenGL, consists of a thin Java layer that implements wrappers for OpenGL functionality. Currently, two Java Specification Requests (JSRs) are active: JSR 231[9] for OpenGL and JSR 239[10] for OpenGL ES. There are three main advantages in binding native functionalities. First, programmers do not need to learn a complete new API. Second, the virtual machine overhead is reduced because the resulting API is a thinner layer. Finally, the development of the API becomes easier to implement. One major problem, though, is the number of Java-native communication calls during runtime.

In conclusion, the Java APIs presented in this section provide the following support, as described in Table 3.3. AWT includes media types, such as images and text, and provides spatial arrangement of the signs (i.e., different kinds of layout classes). In addition, AWT events provides the mechanisms for user interaction. Finally, the thread class or the JMF API include temporal dimension.

Markup Languages: XML-based

Because the Web has evolved, during recent years, from a information service provider into an application framework, W3C is updating Web technologies for current demands. Some recommendations W3C is responsible for are compared in Table 3.4. These include:

- **XML [18]:** meta-language used for describing other languages.
- **Document Object Model (DOM) [79]:** defines the XML tree.
- **XML Events [104]:** defines how user interaction can be integrated into DOM.

[8]http://java.sun.com/products/java-media/3D
[9]http://www.jcp.org/en/jsr/detail?id=231
[10]http://www.jcp.org/en/jsr/detail?id=239

- **HTML [145]:** Markup language used for developing Web pages.

- **CSS:** defines the layout and visual qualities of a document. CSS1 [96] defines properties for, for example, fonts, background, and text, while CSS2 [16] extends it with, for example, relative and absolute positioning of the elements, new pseudo-classes (e.g., hover), and new box types (e.g., compact). In addition, CSS2 introduces the concept of media types, taking into consideration the special presentation qualities of different media (e.g., television, aural, and handheld media types).

- **XHTML 1.0 [133]:** reformulation of HTML into XML. XHTML 1.0 is specified in three different Document Type Definitions (DTDs): Strict forces the uses of style sheets for the look of the document, Transitional allows the use of style sheets, and Frameset includes frame support.

- **Modularisation of XHTML [2]:** the logical division of XHTML elements into abstract modules, each for a different purpose. These modules include, as depicted in Figure 3.2 (a), core, other XHTML, W3C, and private modules. This way, depending on the purpose, different XHTML-compliant languages can be defined, as long as they include the XHTML core module. One example of such a language is XHTML Basic, intended for resource-constrained devices [131].

- **XHTML 1.1 [132]:** a modularised version of XHTML 1.0 Strict, as depicted in Figure 3.2 (b). Frames and related modules (e.g., target), legacy modules, and name identification modules are not included. In addition, no applet module is included, but developers can use the object module instead.

- **XForms [43]:** for advanced Web forms description.

- **XHTML 2.0 [11]:** a truly revolutionary language for Web page development, in which other W3C languages such as XForms and XML Events are included [141].

- **SVG [61]:** for vector graphics images.

- **SMIL [80, 81]:** a language that defines the spatial and temporal relationships of a multimedia presentation.

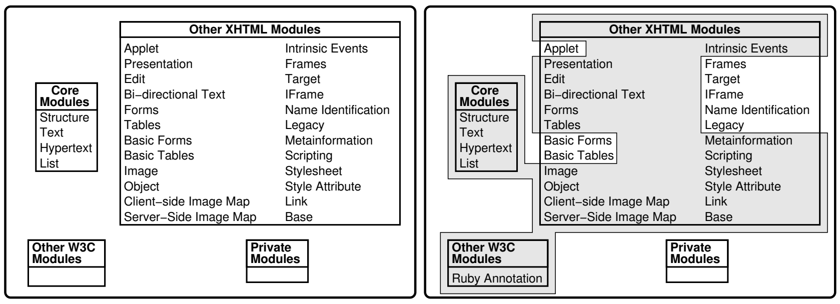
- **Timesheets [91]:** based on SMIL syntax, this is like CSS but defines the timing of a document instead of its look. In contrast to SMIL, it can be applied to any XML document (not a WC3 recommendation).

- **XML Compound Documents**[11]**:** integrate multiple XML recommendations as different namespaces, each one intended for a different purpose.

[11] http://www.w3c.org/2004/CDF

Table 3.4: Comparison between XML-based Languages.

	HTML-Based[12]				
	XHTML 1.1+CSS	**XHTML 2.0+CSS**	**SVG**	**SMIL**	**XForms**[13]
Media Types					
Audio	No	No	Yes	Yes	–
Text	Yes	Yes	Yes	Yes	–
Images					–
Bitmap	Yes	Yes	Yes	Yes	–
Vector	No	No	Yes	No	–
Animation	No	No	Yes	Yes	–
Video	No	No	No	Yes	–
Arrangement of the Signs					
Spatial	flow and absolute		absolute		–
Temporal	No	No	No	Yes	–
Interaction					
Passive	Yes	Yes	Yes	Yes	Yes
Reactive	Provided by User Agent (e.g., Scrolling)				
Proactive	Yes[14]	XForms	Links		Yes
Directive	Yes[15]	XForms	No	No	Yes



(a) Modularisation of XHTML. (b) XHTML 1.1.

Figure 3.2: Evolution of XHTML Modules.

[12]Scripting can provide, for example, higher interactivity, but in this thesis it is not considered a markup language. HTML-based languages can include objects such as video or audio, but their behaviour depends on the user agent's implementation. Moreover, synchronisation between media objects is only possible by using scripts or other XML languages such as Timesheets.

[13]The media types and arrangement of the signs are determined by the root document.

[14]HTML Forms (e.g., Buttons), but their logic still needs scripting.

[15]HTML Forms include widgets such as Text Input, but their logic still needs scripting.

MPEG-4

MPEG has evolved from a set of video and audio compression standards to a ”complex toolkit capable of providing solutions to the environment” [28]. The MPEG-4 standard defines a scene as a composition of different multimedia objects (e.g., video, 3D, and 2D graphics), including their spatial and temporal relationships [135]. MPEG-4 is a large and complex standard, hence it is divided into a considerable number of profiles that permit implementations of the standard. For example, the Simple 2-D Graphics profile provides the means to place one or more objects in a scene, while the Simple 2-D Graphics+Text profile supports, as well, coloured and transparent text [60]. MPEG-4 is intended for all kinds of transmission networks (i.e., it is transmission agnostic) and can therefore be used, for example, for streamed content (i.e., Internet portals) and interactive television (i.e., broadcasts).

In addition, MPEG-4 specifies different programming levels, such as binary format, XML, and Java. First, Binary Format for Scene (BIFS) defines a hierarchical structure, called a scene graph, which forms a multimedia presentation. Each node of the graph is an object with properties such as colour, size, position, and timing [136]. In addition, the author of the presentation can specify the behaviour of the objects by using BIFS Animations and BIFS Commands [32]. The first describes the behaviour of an object (e.g., make an object spin), while the latter describes its conditional behaviour (i.e., its reaction to user interaction) [63]. Second, the eXtensible MPEG-4 Textual (XMT) format is declarative, XML-based language, to describe the scenes. Finally, MPEG-4 defines MPEG-Java (MPEG-J), a set of Java APIs to permit running of Java programs embedded in MPEG-4 presentations.

MHEG

”Although an interchange format, MHEG is more than just a binary format. It also possesses features that suit real-time interchange in a networked environment” [107] said Meyer-Boudnik about MHEG-1 in the year 1995 [84]. Since then, the International Organization for Standardization (ISO) has published several MHEG-related standards. These standards include, for example[16]:

- **MHEG-5:** a simplified version of MHEG-1 intended for set-top box-like terminals.
- **MHEG-7:** conformace testing of MHEG-5 [85].
- **MHEG-8:** Web technologies and MHEG-5 integration.

In the scope of this thesis, MHEG-5 [41] is the most interesting standard in the family, because it has been chosen for digital terrestrial television receivers in the UK. Basically, MHEG-5 defines an application as a composition of several scenes. Each scene integrates different multimedia objects such as graphics, bitmaps, text, and streams (i.e., audio and video), the content of which can be accessed from a URL or from the object carousel. Moreover, MHEG-5 authors can include interactive components

[16]http://www.jtc1.org/news/MHEGpress.htm

such as buttons, hyperlinks, and sliders in a scene. Apart from these interactive components, the behaviour of a scene can be defined by objects called Links. A Link class can specify, for example, to go to the next scene when the user presses ’right’. In conclusion, MHEG-5 is a platform-independent authoring format that supports different multimedia objects, temporal and spatial dimensions, and user interaction.

Comparison of Authoring Formats

C language is the most efficient and powerful (e.g., the developer has direct control over pointers to memory) alternative. But at the same time it is more difficult to use, less safe to distribute, and less interoperable (i.e., each service has to be compiled for the target device). Java, on the other hand, is less powerful than C. But, because Java is safer to distribute and developers do not have to worry about the settings of the target device[17], digital television standardisation bodies have selected it for their procedural environment. Still, in situations in which efficiency is more important than interoperability, the the use of a compiled language might be the best option (e.g., a device manufacturer providing a 3D game to their own receivers).

XML languages have been already taken into account for the declarative environment of digital television receivers. Their major benefits, as commented before, are their ease of use, their interoperability, and safety of distribution. Because their expressive power is small (i.e., they are not programming languages), authors tend to abuse scripting. The main problem of scripting is that its heavy use leads to low maintainability, accessibility, and a high learning curve [141]. In my opinion, if an application needs too much scripting, it probably means that a programming language, like Java, would better suit its implementation. Hence, markup languages can be used for less resource-demanding services, such as Super Teletext. Finally, another benefit of adopting XML-based languages is that they are used in the Web, opening up the opportunity of Web browsing from digital television receivers.

The MPEG-4 audio and video codecs are a natural evolution from current MPEG-2 formats. In fact, the Japanese standard and the latest DVB specification on Video and Audio Coding for Broadcast Applications [57] already includes support for them. Regarding the application environment, there are two major opinions. On the one hand, some people consider MPEG-4 as a companion of Java in the form of a plug-in, providing complementary functionalities [63]. On the other hand, because of the expressive power of MPEG-4, it can be considered more of a competitor than a companion. Probably the main reason why the digital television standardisation bodies have not relied more heavily on MPEG-4 is its immaturity for the conservative broadcast environment. In addition, even though MPEG-4 follows a modular approach, its main drawback is its complexity [27]. Hence, the development of an MPEG-4 player becomes expensive. In fact, players supporting 3D graphics are still under development [159]. In addition, there are only a few authoring tools available, forcing developers to learn a new language [27], and a script is needed to provide the application logic. Fi-

[17]But the device manufacturer has to include a JVM + APIs.

nally, integration between the BIFS node layer and Web browsers does not explicitly exist [159].

Finally, MHEG has lately been widely used in UK digital broadcasts. Some of the problems of MHEG are its lack of 3D graphics support, as well as the need for scripting for the application logic. In the CENELEC technical report [24], the 6th objective states: ”Enabling coexistence between MHEG-5 and MHP and facilitating migration from MHEG-5 towards MHP ...”. But, in my opinion, because of the high number of MHEG-5-enabled receivers on the market, MHP will include MHEG-5 support using an interoperable plug-in mechanism[18].

3.5 Procedural Environment for Digital Television

This section first reviews the different regional alternatives for a procedural environment, which, as explained in the previous chapter, have defined Java as programming language. Then it compares them and defines a worldwide common architecture for digital television receivers.

DVB-J

The idea behind MHP was to provide a cross-platform middleware for digital television receivers. DVB-J is the Java stack defined by MHP and is based on the PersonalJava Specification, upgraded and modified for television-specific needs. Even though with the advent of J2ME Sun has respecified PersonalJava as Foundation and Personal Basis Profiles, at the time of writting this thesis, DVB-J has not taken any decision regarding the matter[19].

DVB-J defines a common set of Java APIs for the development of digital television applications. The packages included, as depicted in Figure 3.3, are [90, 98, 99]:

- **Sun Java:** includes the following packages:
  - **Core Java:** basic APIs to implement Java applications, such as *java.lang* or *java.util*. In addition, in order to provide a Java graphical framework, some parts of AWT are incorporated (e.g., Component, Container, Graphics).
  - **Java TV:** manages the lifecycle of the applications.
  - **JMF:** includes JMF 1.0 as a compulsory packet with some clarifications, restrictions, and extensions in order to control the broadcasted video and audio, and play back audio files.
- **Home Audio/Video interoperability (HAVi) Level 2 UI[20]:** specific widgets extending AWT components for the digital television environment. In addition, HAVi includes a television-specific means to present and interact with interactive applications.

[18]April 2005: MHP 1.1.2 [58] includes modifications to MHP for its co-existence with MHEG-5.

[19]April 2005: MHP 1.1.2 [58] is based on J2ME Personal Basis Profile.

[20]http://www.havi.org

- **Digital Audio Video Council (DAVIC):** addresses issues related to the MPEG stream.

- **DVB:** a set of miscellaneous functionalities to glue rest of the packages together (e.g., access to SI tables, definition of *org.dvb.ui.DVBColor* including alpha channel).

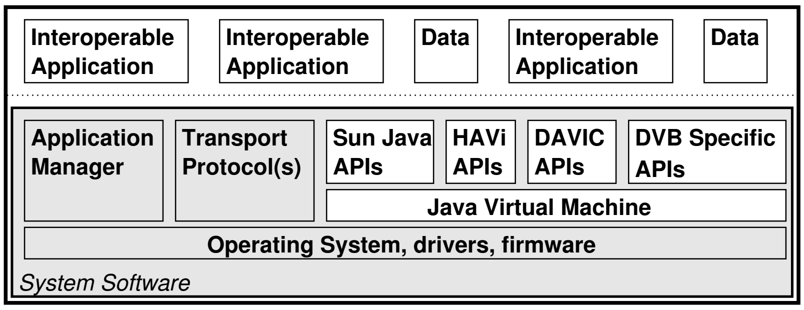


Figure 3.3: DVB-J Middleware Stack [98].

ACAP-J

ACAP-J is the procedural environment for digital television receivers defined in USA. ACAP is an initiative by OpenCable and ATSC (i.e., DASE) to comply with the GEM standard. ACAP-J is based on the same Java API and application model as OCAP, and thus DVB-J, with minor modifications. Topics such as security and application and graphics models are different than in DVB-J. For example, the ACAP-J graphics architecture does not assure the presence of a background plane and takes into account the National Television System Committee (NTSC) resolution. Moreover, ACAP-J defines, apart from the monitor application, two kinds of services: bound and unbound. Bound applications are those dependent on the currently tuned channel, while unbound are independent (e.g., e-mail). In order to signal these unbound applications, ACAP defines, in addition to the Application Information Table (AIT), the eXtended Application Information Table (XAIT).

STD-B23

In its conception, ARIB did not include any procedural environment, but only a declarative one, STD-B24 [4]. In order to harmonise with the GEM specification, ARIB produced the STD-B23 [3] standard. STD-B23, named ”Application Execution Engine Platform for Digital Broadcasting”, is a combination between GEM and ARIB STD-B24. Some of the specific extensions to GEM, include, for example, Japanese language fonts and H.264 video format. In addition, applications are transmitted using a data carousel instead of the object carousel.

Table 3.5: Comparison Between DVB-J, ACAP-J, and ARIB-AE [3, 9, 39, 51, 52].

	DVB-J	**ACAP-J**	**ARIB-AE**
Java Profile	Personal Java		
Java APIs	Java TV, JMF, DAVIC, HAVi		
Service Information	Different mechanisms[21]		
Bridge to Declarative Content	Yes		No
Other Application	Plug-ins	Bound, Unbound, Monitor	–
Graphics	Background+Video+Graphics		
	–	NTSC	Japanese fonts

Harmonisation

Table 3.5 compares the different procedural approaches decided by the regional bodies: DVB, ACAP, and ARIB.

The International Telecommunication Union (ITU)-T has published a document, J.202 [87], for the worldwide harmonisation of the procedural environment. This document provides the GEM specification as the mandatory common core of the execution engine, while region-specific standards such as ARIB, DASE, OCAP, and MHP are considered as optional and informative. The packages included in J.202, as depicted in Figure 3.4, are the following [86]:

- Personal Java Application Environment: *java.** package.
- Java TV: *javax.tv.** package.
- JMF: *javax.media.** package.
- DAVIC 1.41 Specification part 9: *org.davic.** package.
- HAVi Level 2 User Interface: *org.havi.ui.** and subsets of *java.awt* package.

3.6 Declarative Environment for Digital Television

This section first reviews the different regional alternatives for a declarative environment. Then it compares them and defines a worldwide common architecture for digital television receivers.

DVB-HTML

DVB-HTML is a new optional application type specified in MHP 1.1. It was introduced as a plug-in in the Enhanced Broadcast profile and as optional in the Interactive Broadcast and Internet Access profiles. DVB-HTML is

[21] For example, OCAP defines the XAIT for unbound applications.

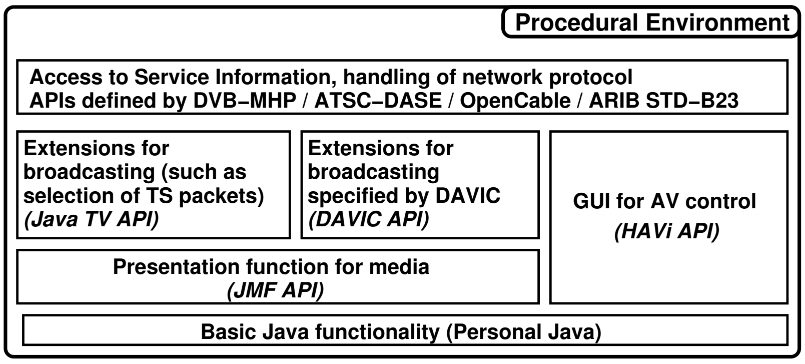

Procedural Environment
Access to Service Information, handling of network protocol APIs defined by DVB−MHP / ATSC−DASE / OpenCable / ARIB STD−B23
Extensions for broadcasting (such as selection of TS packets) (Java TV API)
Extensions for broadcasting specified by DAVIC (DAVIC API)
GUI for AV control (HAVi API)
Presentation function for media (JMF API)
Basic Java functionality (Personal Java)

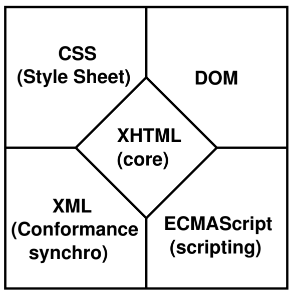
Figure 3.4: Worldwide Procedural Environment for Digital Television [86, 87].

intended for broadcasters that wish to deliver XML-based content. It is important to notice that DVB-HTML and the Browser Client defined in the Internet Access profile are different topics. The former is specified in MHP in great detail as an XML-based format, ensuring interoperability between broadcasters. In contrast, the Internet Access profile does not specify which XML languages the browser must support. Hence, it would be possible to implement the Internet Access profile and not support DVB-HTML. [26, 69, 70, 71, 120, 121, 137, 167]

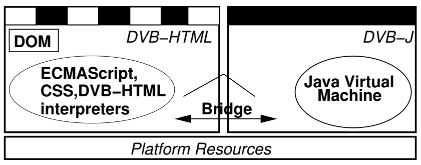
DVB-HTML, as depicted in Figure 3.5 (a) includes a number of W3C standards such as XHTML, CSS 2.0, DOM 2.0, and ECMAScript [59]. First, the core of DVB-HTML is a superset of XHTML Basic and includes a new module, DVB Intrinsic Events, scoped within a new XML namespace, dvbhtml. Second, DVB-HTML is fully compliant with CSS 2.0, adding a new media type (dvb-tv) and including some extensions (e.g., MHP graphics model integration). Third, DVB-HTML includes a number of modules from DOM 2.0 (e.g., Level 2 core and Level 2 views) and new modules: DVB-HTML DOM replacing DOM Level 2 HTML because support for HTML 4.01 is not needed; DVB Events DOM to integrate DVB-HTML within the MHP platform; Key Events DOM to bind remote control events; DVB CSS DOM for the CSS extensions for the television environment; and DVB Environment modules to access environment variables (e.g., the containing window). Finally, DVB-HTML specifies issues such as an application model, a bridge with DVB-J, a security model, and A/V synchronisation. The bridge is an interface that allows the presentation engine to access procedural methods (e.g., to resize the broadcasted A/V content). Figure 3.5 (b) depicts the relation of DVB-HTML and DVB-J.

ACAP-X

ACAP selected DASE-1 specifications for their declarative environment. In terms of XHTML modularisation, ACAP-X is a superset of XHTML Basic comparable to XHTML 1.1. The only difference is that ACAP does not support server-side image maps, but it includes support for frames and name identification. ACAP also includes a number of popular media types such as



(a) DVB-HTML Languages [26].



(b) DVB-HTML Architecture [137].

Figure 3.5: Relation Between DVB-HTML and DVB-J.

video (e.g., MPEG), animated graphics (e.g., MNG), audio (e.g., MPEG), and text (e.g., ECMAScript). But most ambitious is the number of supported applications types, including ACAP-X, ACAP-J, security mechanisms, and even compressed (i.e., ZIP). The style sheet support is a subset of CSS2, similar to the approach used in DVB-HTML, but without extensions outside the W3C specification. Finally, the defined DOM tree specifies the following modules: Core Fundamental, Core Extensions, View, Stylesheets, CSS, Event, VirtualKeys, and Mouse, User Interface, and Mutation Events.

BML

In October 1999, ARIB published the document ”Data Coding and Transmission Specification for Digital Broadcasting”, STD-B24, which specified an application authoring format for digital television receivers: Broadcast Markup Language (BML). The BML specification is really ambitious, being a superset of XHTML 1.1. In fact, it includes all the modules of modularised XHTML. Probably the fact that Japanese consumers have a high-level of technology adoption that cannot be compared to European or American markets is an important factor here. In addition, the number of supported media types is extensive. For example, apart from MPEG-2, it includes support for MPEG-4 video and audio, Japanese characters, streamed text (i.e., subtitles). It is fully CSS2 compliant and includes some properties from CSS3 User Interface Module such as nav-index. Finally, it includes DOM Level 1 core, extension, and HTML, and some BML extensions.

Harmonisation

At the time of writting this thesis, one of the biggest challenges ahead is the definition of the worldwide common declarative environment for digital television. In this case, contrary to the procedural environment, the final solution does not seem to be DVB-HTML on a worldwide basis. The major problems of DVB-HTML are that it is tightly linked with DVB-J and DVB transmission mechanisms, it includes a complex event model, and it is complicated and expensive to implement. An extract from the MHP Web page gives a better idea of the situation:

> **”Is it true that DVB-HTML may be replaced in the future within the MHP 1.1 specification by another TV-centric HTML-based language?** In the spirit of global harmonisation, it has been suggested that DVB-HTML should be modified or replaced by the resulting work of the CableLabs-OpenCable and ATSC-DASE harmonisation. The discussions are still ongoing.[22]”

Figure 3.6 compares the different regional standards according to their included XHTML modules. Because of the differences between the standards, ITU has published ITU-T J.201 [88], a document aiming to specify a worldwide declarative environment. This document describes the presentation engine, as depicted in Figure 3.7, including the following elements [86]:

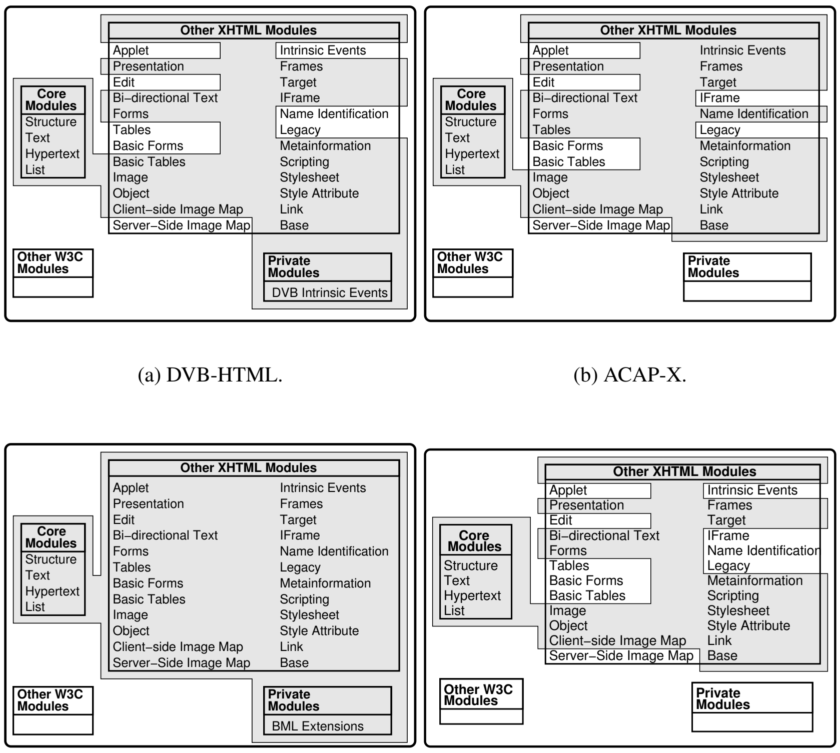
- Common Modules defined in XHTML modularization and/or HTML.
- CSS Style Sheets to control the style.
- TV-specific extension APIs and additional APIs and DOM objects (e.g., remote control input).
- Media types such as JPG images and MPEG-2 video.
- XML parsers.

In spite of the fact that the three regional standards provide similar functionality, their final solutions and viewpoints differ greatly. In addition to the technological differences, political interests make the harmonisation of the presentation engine a difficult task. The CENELEC technical report [24], referring to the integration of presentation and execution engines into a single standard, clearly expresses this issue:

> ”The most likely candidate [...] is ACAP. ACAP [...] will be made available to DVB in order to for it to evaluate the possible replacement of its DVB-HTML presentation engine in MHP 1.1 specification. It should however be noted that ARIB has also proposed BML specification for consideration in this harmonization process. Therefore, there is insufficient certainty at this point in time to determine whether harmonization efforts with respect presentation engines will actually lead to a positive and useful result within a reasonable amount of time” [24].

In conclusion, a concrete solution worth mentioning is the latest standard proposed by the Society of Motion Picture and Television Engineerings (SMPTE), called ”Declarative Data Essence – Transitional” [155]. This proposal takes into account the specifics of each regional standard such

[22]http://www.mhp.org/index.php?id=202



(a) DVB-HTML. (b) ACAP-X.

(c) BML. (d) Common Modules.

Figure 3.6: XHTML Modules in Digital Television Regional Standards.

as Poter-Duff composition and navigation properties, includes basic media types (e.g., MPEG-2 video, jpg, and png), and describes simple locators (e.g., tv: for tuning and stream selection). Unfortunately, it fails to describe a proper bridge between the declarative and procedural environments. As a final conclusion, I would like to propose the following questions: What organisation should be in charge of defining a standard for the declarative environment: DVB, OpenCable, ARIB, W3C, or SMPTE? Would it not be better if all these groups join and harmonise, so the number of tags defined for the same purpose stop increasing unnecessarily?

3.7 Harmonisation of Application Environment for Digital Television

After a detailed comparison of different alternatives for content authoring formats, this section completes the picture by studying the application environment for digital television receivers. First, as a summary of the previous sections, Figure 3.8 shows the main differences between the regional software middlewares: ACAP, ARIB, and MHP.

ITU has published a document entitled: ”Worldwide Common Core -

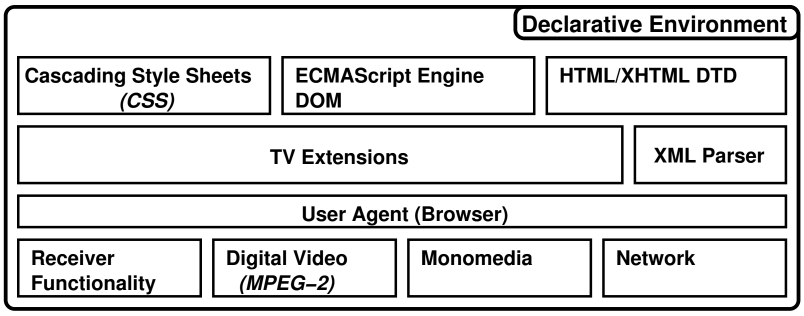


Figure 3.7: Worldwide Declarative Environment for Digital Television [86, 88].

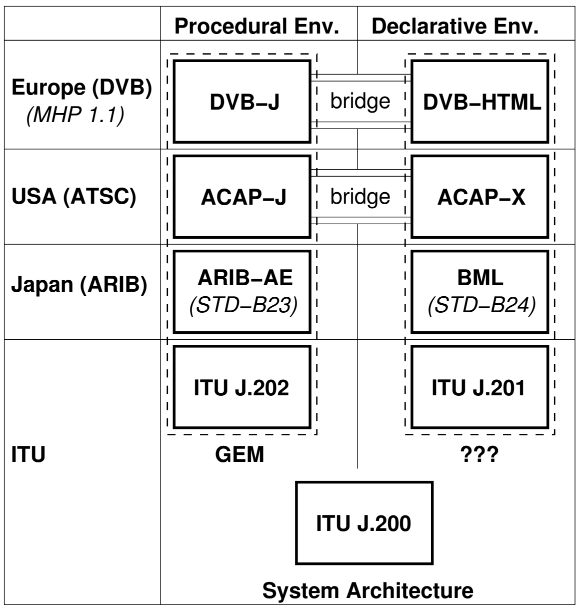


Figure 3.8: Standardisation Situation of the Application Environment [123].

Applications Environment for digital interactive television services," catalogued as ITU-T J.200 [86]. This document describes the worldwide application environment, depicted in Figure 3.9, based on two environments: procedural and declarative. The procedural environment, as depicted in Figure 3.4, is Java-based and was described in Subsection 3.5. The declarative environment, depicted in Figure 3.7, is XML-based and was studied in Subsection 3.6. According to J.200, the two environments do not have to be separated, as a bridge functionality can link them. In addition to the environments, native applications, proprietary formats, and service-specific software and content (e.g., MHEG) can be supported, as well as plug-ins (i.e., Native Software layer).

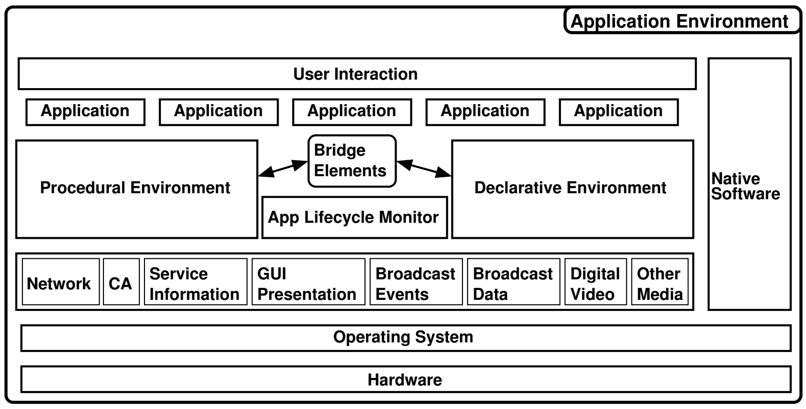


Figure 3.9: Application Environment System Architecture [86].

3.8 Summary

This chapter has studied content authoring formats for the digital television environment. First, interactive multimedia was defined, creating a group of metrics to evaluate different kinds of content formats. These metrics included difficulty of use, expressive power, and safety of distribution. Based on the metrics, a taxonomy of formats was proposed differentiating programming languages and markup languages. After that, some representative examples of content formats, such as C, Java, XML-based, MPEG-4, and MHEG, were compared and their potential use for implementing digital television services was analysed. Then an application environment composed of a procedural part and a declarative part was proposed for digital television receivers. The procedural part is based on Java, as defined in GEM (i.e., DVB-J), and the declarative part is based on XML, using XHTML, CSS, different media types (e.g., video), and extensions such as remote control events.

4 EXTENSIONS TO GLOBALLY EXECUTABLE MULTIMEDIA HOME PLATFORM AND RECOMMENDATIONS FOR IMPLEMENTERS

Chapter 4: Extensions to GEM
(Proposals to Enhance GEM)
Related Work
3D Graphics Support
Audio and Video Support
WWW Convergence

Note: This chapter is an extended version of [P9].

The focus of this chapter is the future evolution of digital television. The specific topics, identified in Chapter 2, for such evolution included:

- Personal video recorder
- WWW access
- Streamed media
- 3D graphics support

First, solutions presented by other researchers and companies are introduced in Section 4.1. These solutions include, for example, commercial products such as Microsoft Networks (MSN) TV and Tivo, and research ideas such as MPEG-4 broadcast and MPEG-7 support. Because one of the intentions of this thesis is to propose a common middleware for digital television receivers, the relation between each solution and the GEM standard is analysed. Finally, Section 4.2 proposes some extensions to the GEM standard.

4.1 Related Work

In this section, the most relevant contributions are analysed. First, from the commercial point of view, Microsoft solutions for future television receivers and the Tivo personal video recorder are described. Then, research about the MPEG family of standards as applied to digital television are overviewed in detail.

Microsoft

MSN TV[1] 2 Internet and media player is a receiver that includes a TV browser and other services such as e-mail and EPG. In order to navigate the Internet, the MSN TV browser supports:

[1]http://webtv.com

- HTML 4.0.
- CSS1 and CSS2 absolute positioning.
- A number of media types, such as video and audio.
- JavaScript 1.3 [118].
- A subset of DOM with some variations.

The receiver includes the Microsoft CE[2] operating system and Windows Media Player[3] for visualising media files. In addition, a Flash[4] player has been integrated, but external plug-ins or programming languages such as Java are not supported. Some requirements taken into consideration during browser implementation were the lack of horizontal scrolling and windows. Thus, the device only displays one window at a time and the user cannot open a new browser window.

The Microsoft (MS) TV[5] platform is an API intended for digital television receivers. It is the middleware included in, for example, MSN TV and Ultimate TV[6] (a personal video recorder). Microsoft has already published a Foundation Edition and an IPTV Edition. The first is intended for cable operators, while the second is for broadband providers. These solutions try to exploit the massive acceptance of Microsoft as a personal computer operating system and to gain in the entertainment market as well. Thus, Microsoft has created a new set of prodcuts, such as Microsoft Media Center[7] and XBox[8], targeting home entertainment.

Tivo

The approach of Tivo[9] towards television is similar to that of Apple towards desktop computers. They have managed to get wide acceptance through openness and good performance. Tivo is a Linux-based personal video recorder for digital television. It allows the user to record programs based on time, program name, genre, and other parameters. In addition, it includes a recommender system based on the user's preferences, so Tivo will record programs that are not explicitly requested by the viewer but can sometimes provide her with a nice surprise. Finally, it records the entire incoming stream with an buffer of around 15 minutes, allowing the viewer to replay or forward content. The main advantage of Tivo is that, being a Linux machine, it can be expanded as desired[10].

[2]http://msdn.microsoft.com/embedded/windowsce
[3]http://www.microsoft.com/windows/windowsmedia
[4]http://www.macromedia.com
[5]http://www.microsoft.com/tv
[6]http://www.ultimate.tv
[7]http://www.microsoft.com/windowsxp/mediacenter
[8]http://www.microsoft.com/xbox
[9]http://www.tivo.com
[10]http://www.tivofaq/hack

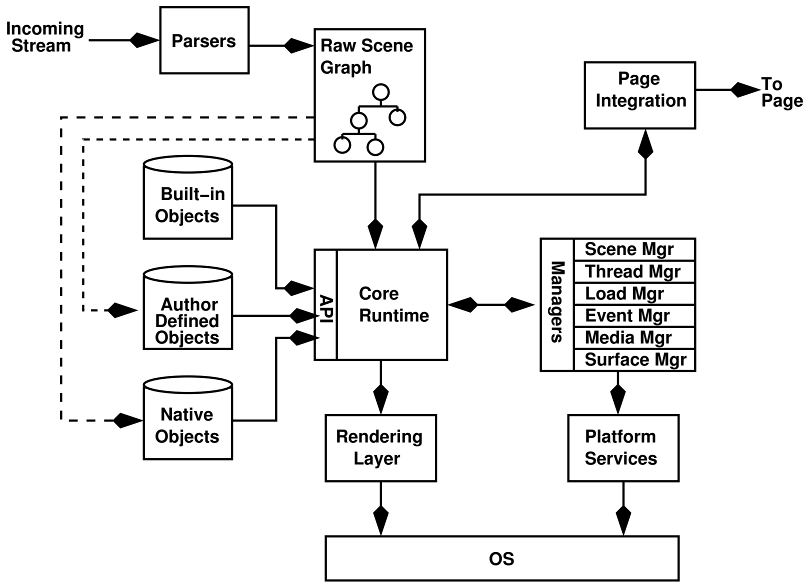


Figure 4.1: The Blendo Language [102].

3D Television

Sony has defined Blendo[11], a declarative markup language derived from Virtual Reality Markup Language (VRML) and intended to move the control of broadcast enhancements from the studio into the living room [102]. The basic two concepts behind Blendo are surface and time-based. First, a surface is a rendering region that can contain a video frame, still image, 2D/3D graphics, or even the output of a rendering application (e.g., HTML, Java). These surfaces can be combined, integrating different objects into the same scene, while matte (i.e., scene composition) can be used for the final rendering. Second, Blendo includes a way of defining the timing of the service, which can be used, for example, to skip watching undesired content. As depicted in Figure 4.1, the input to a Blendo engine is a text file, which is parsed and contains the description of the raw scene (i.e., objects and spatial and temporal information). The objects can be built-in, author-defined, or native objects implemented, for example, in Java. Hence, Blendo offers an extensible object model. During runtime, these objects can use certain managers to access system services such as threads. One implemented prototype is an interactive sports enhancements for car racing. It includes polling questions, as well as overlay information (e.g., real time telemetry data)[144].

Another approach towards 3D TV is the Personalised, Immersive Sports TV Experience (PISTE) project [17, 101], which studied the possibilities of MPEG-4 for integrating animated 3D content and television broadcasts. The PISTE architecture, depicted in Figure 4.2, is composed of the following elements: video capture, database repository, preparation unit, authoring unit, scheduler, and broadcast station. First, the live content is captured with

[11] http://www.blendomedia.com

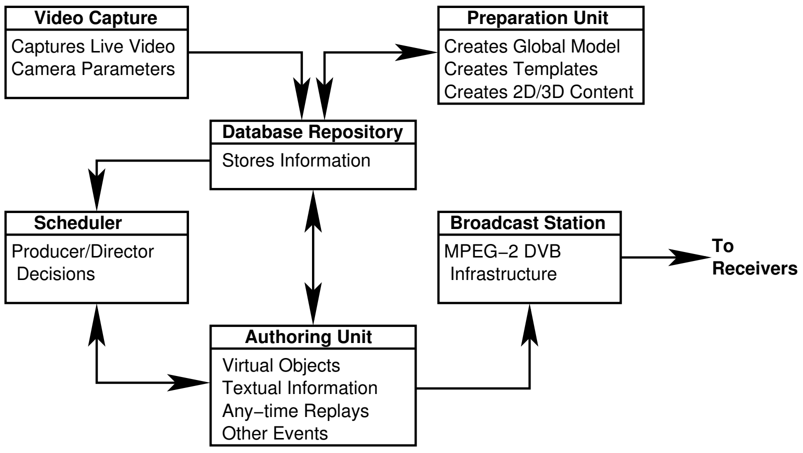


Figure 4.2: 3D Television Configuration [17].

conventional cameras, and specific parameters are taken so that the global environment can be reproduced. After that, the data is fed to the database repository, so it can be accessed by the other units. The preparation unit is in charge of creating the global model (i.e., the viewer can change the viewpoint) and it uses BIFS to create 2D/3D animated content. The authoring unit is responsible for producing the enhanced content, which includes virtual objects (e.g., the trajectory of a ball), additional textual information (e.g., statistics), anytime replays (e.g., 3D animation that can be viewed from any angle), and events notifying of other sports happenings the user can watch. The final content is scheduled based on director/producer comments and transmitted using the MPEG-2 standard.

These solutions towards a real revolution in television, making it a 3D environment, sound interesting for future television systems. But probably, as Magnor wrote: ”they are not coming this year. The TV market exhibits enormous momentum and has already defined a number of previous attempts at technological advances e.g., High Definition TV (HDTV) and digital broadcast” [100].

MPEG-4 Enabled Receiver

One of the use cases examined when defining MPEG-4 was interactive television. Because of the benefits of MPEG-4 in transmitting small-size files in which video and 2D/3D graphics are integrated, several projects have been studying how to provide a MHP/MPEG-4-enabled receiver. Some projects studied in this subsection include CustomTV, Sambits, and Philips research. Another intersting project is the ITEA EUROPA project [68], which in addition to MPEG-4 studies MPEG-21 (i.e., digital rights management) for high-end set-top boxes.

There are a number of projects dealing with MPEG-4 applicability in the digital television environment. The trajectory of John Cosmas' research is representative of how the topic has been studied. First, the CustomTV[12]

[12]http://www.irt/customtv (July 1998 - December 1999)

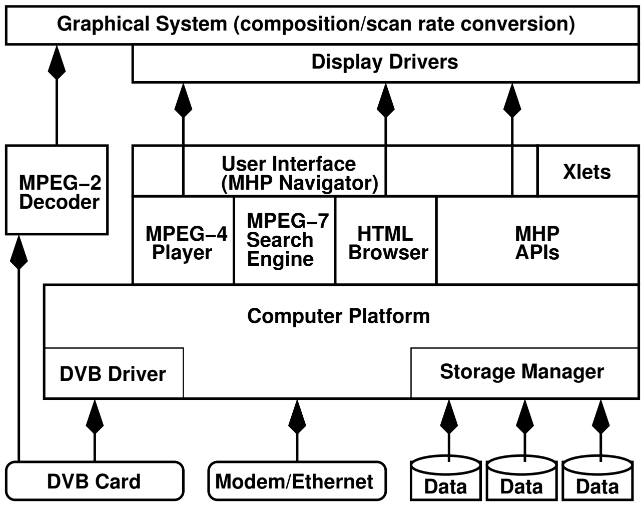


Figure 4.3: Sambits Architecture [83].

project focused on the suitability of the MPEG family of standards for advanced multimedia content (i.e., MPEG-4), customisation (i.e., MPEG-7), and broadcast (i.e., MPEG-2) [33, 34]. After CustomTV, the Sambits[13] project proposed a platform that includes MHP, MPEG-4, MPEG-7, and HTML support [83], as depicted in Figure 4.3. Currently, the Instinct[14] project studies the co-operation between broadcasting and telecommunication services for people on the move.

Another demonstration platform combining MHP and MPEG-4 was presented by Philips at the EUSAI 2004 Symposium [142]. As depicted in Figure 4.4, such a platform includes the best of both worlds. From MHP, it includes data storage, application management, security and certification, storage of user profiles, object carousel support, and remote control support. From MPEG-4, it includes enhanced graphics, 2D/3D animations, interactivity inside the scene, the possibility of presenting several videos on the screen, and enhanced quality of compressing.

As a conclusion, all of these alternatives for expanding the MHP stack with MPEG-4 functionality are interesting. Still, I have not been able to find the reasons why the MHP standard has not taken MPEG-4 into consideration. The main reason is probably its relative immaturity, a drawback in the conservative broadcast environment, as discussed in Subsection 3.4. It is also quite a complex standard, so the development of players and authoring tools for BIFS is expensive. In addition, the integration between the BIFS node layer and Web browsers does not explicitly exist.

Game Consoles

Game consoles have opened new markets in the past years. The newest models include communication mechanisms to play online and offer DVD

[13]http://www.irt/sambits (January 2000 - December 2001)
[14]http://dea.brunel.ac.uk/instinct (January 2004 - December 2005)

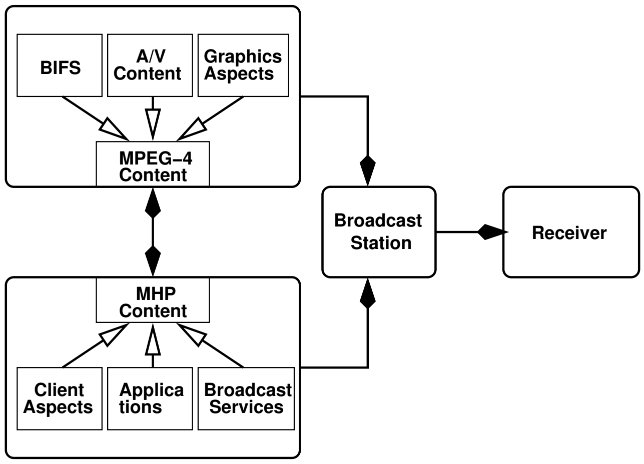


Figure 4.4: Possible Configuration of MHP and MPEG-4 [142].

playback. The next logical step, after including DVD support (i.e., MPEG-2) is to provide digital television visualisation. In fact, Microsoft has truly seen the potential of television, and XBox includes the option of television reception. Apart from these new features, the basic characteristics of game consoles include 2D/3D graphics acceleration with a powerful 3D processor, a game pad input mechanism, and 32 to 64 MB of RAM[15]. Table 4.1 compares the most relevant models of video game consoles.

The software architecture of current game consoles includes an operating system or software kernel that provides functionalities such as memory card management (e.g., DVD) and peripheral device (e.g., game pad) control. In addition, it can use a network card that provides an Internet connection for online gaming. Microsoft's XBox runs a stripped-down version of the Windows 2000[16] Kernel and uses APIs based on DirectX[17]. Playstation 2[18] uses its own operating system, but a version of Linux[19] can be installed, making it an all-purpose computer system. Futhermore, the hard disk of the next version, PlayStation 3, will be able to support a variety of operating systems (e.g., Linux and Mac Tiger), according to Sony's Web page[20]. Some other advances will include, for example, WWW access while playing games and digital photograph display.

Some conclusions from this subsection are the similarities between the evolution and ideas behind digital television receivers and game consoles. With regard to input/output mechanisms, both kinds of devices use a remote control-like device and the television set. In both cases, the return channel or communication mechanisms have been included recently, and both of them are going mobile. The major difference is the processing power of each machine and the level of complexity. Game consoles include complex

[15]http://review.cnet.com and http://www.dealtime.com
[16]http://www.microsoft.com/windows2000
[17]http://www.microsoft.com/windows/directx
[18]http://playstation.com
[19]http://playstation2-linux.com
[20]http://www.sony.co.uk/view/ShowArticle.action?article=1121156666920

Table 4.1: Comparison of Game Consoles.

	Game Console		
	XBox	**PlayStation 2**	**GameCube**[21]
Processor	733MHz	294.9MHz	485MHz
RAM	64MB	32MB	40MB
Communication Type	Ethernet	Ethernet	56 Kbps modem
2D/3D Graphics Acceleration	Yes		
DVD Playback	Yes	Yes	No
Broadcast Television	Yes	No	No
Input Device	Game Pad		
Output Device	Television Set		

Table 4.2: Related Work Topics.

Project	**Topic**	**GEM Based**
MSNTV	WWW Access	No
Tivo	Digital Video Recorder	No
Blendo	3D TV	No
PISTE	3D TV	No (but DVB)
Sambits	MPEG-4 Engine (3D Graphics / WWW Access)	Yes
ITEA/EUROPA	MPEG-7 (Digital Rights Management)	Yes
Game Consoles	3D Graphics (additional DVD, Television)	No

hardware but do not provide a proper set of APIs, while digital television receivers are quite simple in their hardware but very complex at the software level (i.e., APIs).

Summary

A summary of all the projects presented in this section is provided in Table 4.2. This table describes each project, taking into account its main contributions and its relation to the GEM standard. As a conclusion, all these projects study the evolution of digital television, and they share three objectives: 3D graphics support, video streaming, and WWW access.

4.2 Extensions to Globally Executable MHP

In the previous section, different solutions for future digital television receivers were studied. Apart from MPEG-4-enabled receivers, these solutions do not aim at extending the GEM standard. In fact, only the research presented at the EUSAI 2004 Symposium [142] proposed a real integra-

[21] http://www.nintendo.com

Table 4.3: Receiver Configurations (please note that each category extends the previous one) [P9].

		Presentation	**Interaction**	**Network**	**Interoperability**
Broadcast					
	Basic	DVB A/V (MPEG-2)	Remote Control	Broadcast Infrared	DVB-T
	Enhanced	2D Graphics			MHP[22]
Interactive					
	Basic			PSTN	MHP[23]
	Internet Access		Keyboard	LAN xDSL	MHP[24] W3C[25] W3C[26]
	High-End	A/V (Any format), 3D Graphics	Joystick	Bluetooth	Extended MHP[27]

tion between MHP and MPEG-4. Because of the problems encountered with MPEG-4 (e.g., threshold, lack and price of authoring tools), this thesis proposes a revision of the GEM standard that takes MHP efforts into account. This thesis tries to complete the picture regarding the evolution of digital television receivers, keeping in mind the MHP approach. Thus, it proposes the division of the possible configurations of receivers into several categories, each one a superset of the previous. The configurations, shown in Table 4.3 and Figure 4.5, are:

- Broadcast
  - Basic: digitalisation of audiovisual content.
  - Enhanced: support for DVB-J applications.
- Interactive
  - Basic: support for a limited XML user agent.
  - Internet Access: complex XML-based support, and thus convergence with WWW content.
  - High-End: support for all kinds of multimedia objects, such as video and 2D/3D graphics.

[22]Enhanced Broadcast profile.
[23]Interactive Broadcast profile (without DVB-HTML).
[24]Internet Access profile (without DVB-HTML).
[25]Basic XHTML + CSS, no scripting, no forms.
[26]XML Compound Documents (e.g., SMIL+XForms, XHTML 2.0 + Timesheets.)
[27]DVB-J stack extended with 3D graphics support.

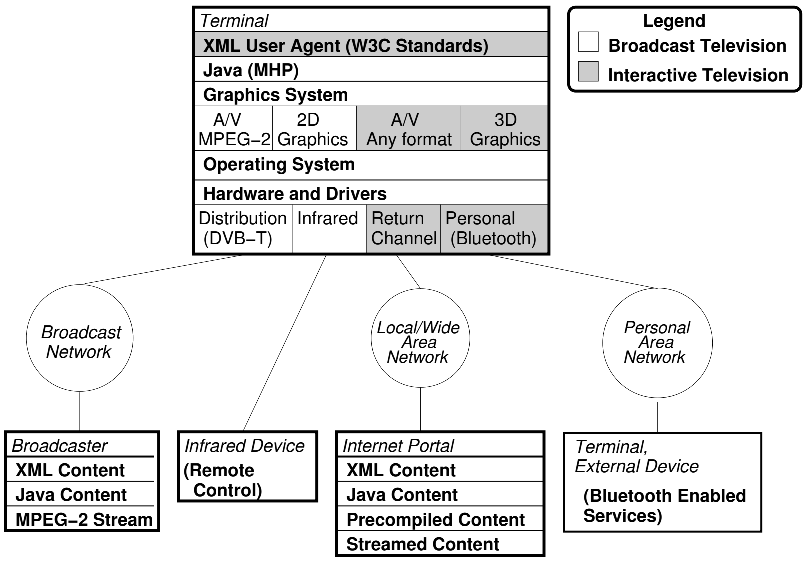


Figure 4.5: Configurations of Digital Television Receivers [P9].

Basic Broadcast

The Basic Broadcast profile is intended for people who only want to sit down on the sofa and watch their favourite show at a given time. This profile represents the first step towards digital television, in which the audiovisual content is digitised. The only services needed are those provided by the receiver manufacturer, such as the native Navigator. This profile takes into account the majority of the viewers, permitting them to have an initial digital experience and, possibly, making them realise some of the potential of a digital television environment.

Enhanced Broadcast

Because of the harmonisation between regional standardisation bodies and its basis on ITU recommendation J.202 [87], this profile requires the support of the GEM standard. In this case, in addition to the audiovisual content, the user can access Java services from the object/data carousel, as explained in Section 3.5. The services include, for example, DVB-J-based Super Teletext, as depicted in Figure 4.6.

Basic Interactive

In this profile, in addition to the broadcast network, the user can access content through the interaction channel. Currently, as discussed in Section 3.6, the major challenge ahead is the definition of the supported declarative environment. Some options include DVB-HTML, BML, and ACAP-X. In any case, such a format should include XHTML, CSS, media types, and specific TV inputs (e.g., remote control input). This thesis proposes the support of basic XML-based documents intended only for visualising content (i.e., information services) and with limited interaction. Hence, the solution is

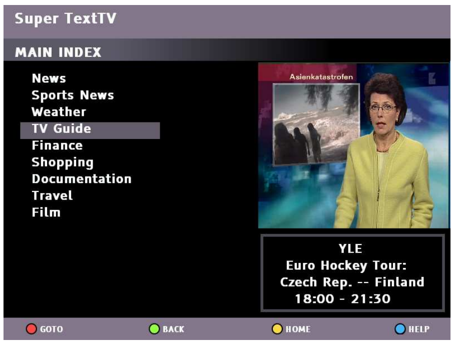
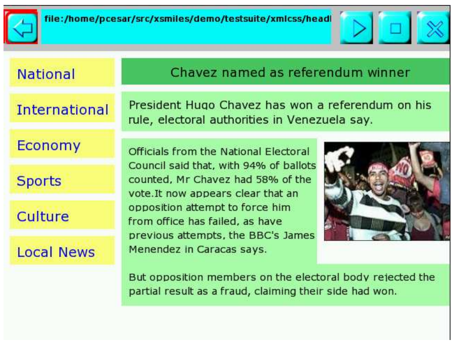


Figure 4.6: Super Teletext Service (DVB-J) [P9].



Chavez named as referendum winner

President Hugo Chavez has won a referendum on his rule, electoral authorities in Venezuela say.

Officials from the National Electoral Council said that, with 94% of ballots counted, Mr Chavez had 58% of the vote.It now appears clear that an opposition attempt to force him from office has failed, as have previous attempts, the BBC's James Menendez in Caracas says.

But opposition members on the electoral body rejected the partial result as a fraud, claiming their side had won.

Figure 4.7: Super Teletext Page (XHTML+CSS) [P9].

Basic XHTML without forms, CSS2 including opacity and navigation keys, DOM 2.0 with support for remote control events, and basic media types such as images. The services include, for example, Super Teletext pages, as depicted in Figure 4.7.

Internet Access Interactive

This profile extends the Basic Interactive with support of XML compound documents. These kind of documents integrate multiple XML recommendations as different namespaces, each one intended for a different purpose. Internet Access Interactive profile is intended for complex services that can provide interactivity and a temporal dimension. Based on the analysis presented in Table 3.4, the host language can be either SMIL or XHTML 2.0. In the first case, because SMIL supports spatial and temporal layout but only restricted interactivity, XForms can be used for advanced user interaction. In the second case, XHTML 2.0 defines the structure of the document and includes XForms for user interaction, CSS is used to control the layout and look of the document, and Timesheets [91] can be used for its temporal dimension. Timesheets, which is based on SMIL syntax, is like CSS but defines the timing of a document instead of its look. In contrast to SMIL, it can be applied to any XML document. One service that can be provided under this profile is a e-Learning portal as depicted in Figure 4.8.

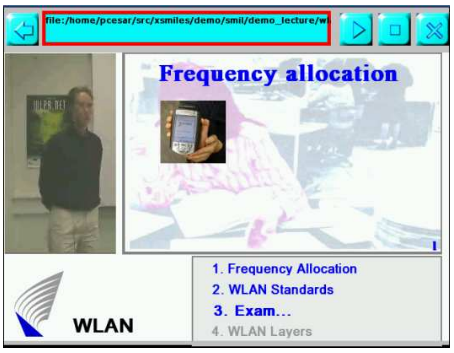


Figure 4.8: E-Learning Portal (SMIL + XForms or XHTML 2.0 + Timesheets) [P7 and P9].

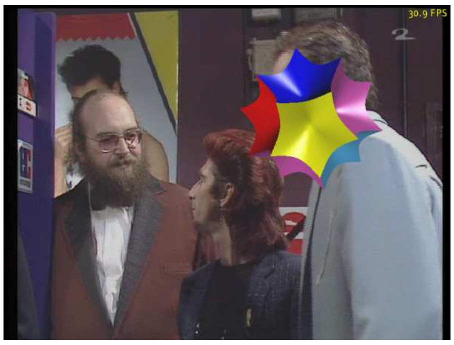
High-End Interactive

With this profile, the receiver becomes a powerful multimedia platform, and thus support for 3D graphics and any audiovisual format is needed. First, the receiver needs to support multiple decoders (i.e., various video streams at the same time) and different formats (e.g., MPEG-2 and MPEG-4). Second, because OpenGL has become the de facto standard of 3D graphics, this thesis proposes the extension of DVB-J with a thin Java layer wrapping OpenGL ES functionality. This feature allows new services such as games, broadcasted 3D commercials, and even game console convergence. For example, Figure 4.9 shows two examples using the underlying OpenGL support. Taking into account the different programming languages presented in Chapter 3, 3D graphics can be supported at the following levels:

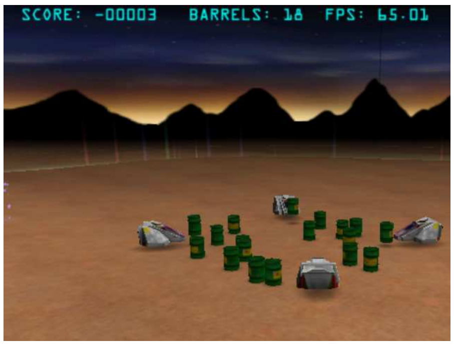
- **Procedural-Intermediate (OpenGL ES)**: this is a recommendation to manufacturers to make digital television receivers a more attractive environment. Two possible scenarios include native games or games downloaded from an Internet portal.
- **Procedural-Interpreted (Java bindings to OpenGL ES):** the DVB-J stack is extended with 3D graphics support. Thus, 3D advertisements can be broadcasted along the audiovisual stream.
- **Declarative (X3D):** the XML user agent can be further extended to support documents containing 3D graphics.

Comparison to Commercial Solutions

The contributions presented in this chapter take into account the most important requirements for a smooth transition presented in Section 2.1: horizontal-market and price. Regarding the horizontal-market, the proposed profiles are based on GEM, the procedural environment of which has already been ratified by ITU, as described in Section 3.7. The price issue was taken into account by proposing a set of profiles, meaning that different users can choose what they really want (e.g., only watching TV or also enjoying the Web).



(a) Simple 3D Object Overlying Audiovisual Content.



(b) Barrel Patrol 3D Game (Downloaded from http://www.fathomgames.com/)

Figure 4.9: 3D Graphics Services in Digital Television Receivers [P9].

The advantage of Microsoft's solutions, from a smooth transition perspective, is that Microsoft has always targeted a wide spectrum of the population (i.e., from kids to grandmothers). In addition, consumers are used to Microsoft's user interfaces, so it would be an easy transition. The main drawback, on the other hand, is that it contradicts the basic requirement of a horizontal-market, according to which proprietary solutions, such as OpenTV or MediaHighway in the European Union boundaries, should follow the GEM standard. In addition, their Web solution does take specific digital television requirements, such as lack of horizontal scrolling or windows, into account, but does not account for current W3C standards for the Web (e.g., XHTML, SMIL).

In the case of Tivo, the problem of standardisation is found once again. As described in Chapter 2, different regional initiatives have already defined personal video recorder capabilities (e.g., STD-B38 in Japan or MHP 1.1). At the same time, with the introduction of GEM, all these issues should be reconsidered, and Tivo seems as a good learning source. In the proposed profiles, digital video recording was not considered, because this is issue is more closely related to the file system than the graphics system. However, this functionality should be integrated into receivers in the Basic Broadcast profile so users can record their favourite shows.

4.3 Summary

This chapter has studied different alternatives for future digital television receivers. Commercial solutions and research have already focused on seamless integration of multimedia objects, user interaction, and WWW access. Unfortunately, these solutions are based on the MPEG-4 standard and, except in a few cases, do not take GEM into consideration. Thus, this thesis proposes the revision of MHP profiles using GEM as the starting point (i.e., MHP 1.0). These profiles include: Basic and Enhanced Broadcast, and Basic, Internet Access, and High-End Interactive. In the Basic Broadcast profile, the receiver is capable of visualising the audiovisual content, while the Enhanced profile extends it with support for DVB-J applications. The Basic Interactive profile includes support for an interaction channel and a limited user agent, meaning that simple XML documents can be browsed. The Internet Access contemplates WWW access and supports better XML compound documents. Finally, the High-End profile supports video streaming and all kinds of multimedia objects, such as video and 2D/3D graphics. These profiles transform the receiver into a powerful multimedia terminal, even including game console capabilities.

5 SOFTWARE GRAPHICS ARCHITECTURE FOR DIGITAL TELEVISION RECEIVERS

Chapter 5: Graphics Architecture
(Device Side Implementation)
Window Based
Toolkit Based
Scene Based
Proposed Model

Note: This chapter is an extended version of [P8 and P9].

In the previous chapter, possible extensions to the GEM standard for next-generation digital television receivers were described. This chapter focuses on the graphics architecture needed to implement such extensions. First, Section 5.1 reviews current research in software architecture in order to place my contribution in context. Second, Section 5.2 defines the requirements the proposed graphics architecture should meet. Third, Section 5.3 presents different models of graphics architecture, studying their benefits and drawbacks for digital television. Fourth, Section 5.4 proposes a graphics architecture based on the reviewed models. Finally, Section 5.5 summarises this chapter.

5.1 Software Architecture Research

According to Garlan and Shaw [67], the architecture of a system describes its computational components and the interactions between these components (i.e., connectors). Modeling the software architecture of a system brings some benefits, such as facilitates understanding and construction of the system, and it allows the use of component libraries to reuse certain components [66]. Garlan identifies three major challenges faced in software architecture research [66]: whether developers should build their own software or use a third party's, how to incorporate external components in a system, and how to define architectures for heterogeneous devices (e.g., toasters and televisions). Some solutions include Lee's [93] actor-oriented design, which takes into account, for example, the mixture of hardware and software design in embedded systems. Another solution was proposed by Wijnstra [165], in which he uses component frameworks to support diversity. He defines the system as a group of self-contained and independent units. Theses units compose the system software, which is divided into application, technical (i.e., abstraction of the underlying hardware), and infrastructure layers. Finally, current research related to this thesis focuses on ”inexpensive, small, heterogeneous, resource constrained” [105] devices or, as it is called, programming-in-the-many [106, 110].

This chapter characterises the system by defining a group of requirements for the graphics architecture of next-generation digital television re-

Table 5.1: Description of the Components of the Graphics Architecture for High-End Interactive Television Terminals.

Component		**Description**
XML User Agent		As described in the standard
Extended GEM Env.		As described in the standard
Application Manager		Controls the lifecycle of the applications
Libraries		
	Graphics	Creates a surface composed of 2D, 3D graphics, and video
	Sound	Plays the audio
	Tuning	Accesses the TV stream
	Audio/Video decoding	Decodes the MPEG-2 stream

ceivers. Then, the next chapter describes a concept implementation [146] of such a system, called Ubik. Ubik is a prototype of a system that meets the identified requirements, and thus follows the feasibility research setting defined by Shaw [153]. Finally, Ubik was implemented by reusing a number of open-source projects, following two strategies described by Clements [31]:

1. ”Systems can be built in a rapid, cost effective manner by importing large externally developed components”.
2. ”The functionality of a component can be separated from its component interconnection”.

5.2 Requirements

This section identifies the requirements for the graphics architecture of next-generation digital television receivers. The requirements can be divided into three groups: software architecture, television characteristics, and entertainment capabilities. The following subsection defines each of these groups.

Software Architecture

Based on the profiles proposed in Section 4.2, the software architecture for High-End Interactive television terminals can be divided into the following layered components: XML user agent, extended GEM environment, and device-dependent libraries. In addition, an application manager is needed as an interface between the applications and the system. Table 5.1 describes each of the components, while Figure 5.1 depicts the system architecture.

Game Capabilities

Televisions are entertainment devices sharing a number of requirements with game environments. Some of these requirements include the use of joysticks or other input devices for interaction. In addition, game authoring requires 2D and 3D graphics primitives and a full colour palette. Other media types that should be supported are images, videos, and animations (e.g.,

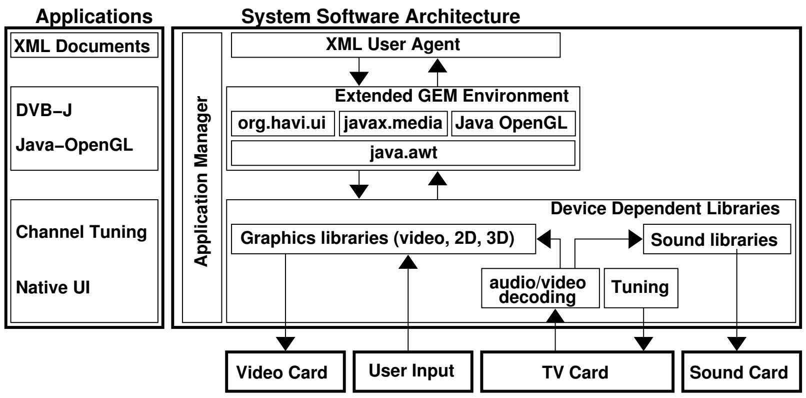


Figure 5.1: Graphics Architecture for High-End Interactive Television Terminals [P9].

moving objects). Moreover, because a game world is composed of different scenes, a scene manager is needed to control user navigation or even overlapping scenes. Finally, the narrative of the games is normally based on time, and thus support for the temporal dimension is required.

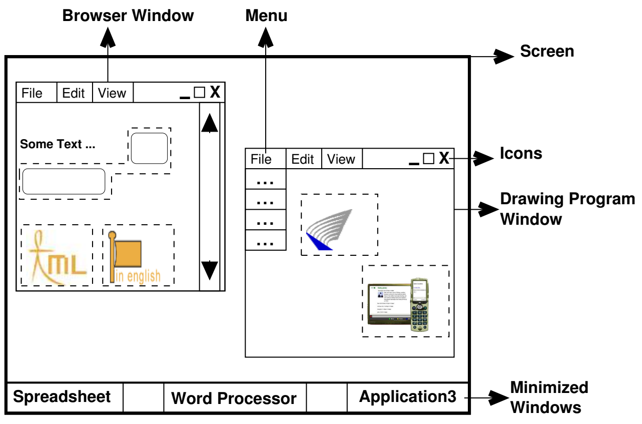
Digital Television Metaphor

Chorianpoulos, in his doctoral dissertation, argues that traditional metaphors cannot be applied to digital television [29]. He starts from the findings of the SIGGRAPH'90 workshop on Software Architectures and Metaphors for Non-WIMP User Interfaces [74], which identifies the following systems that need of a different approach than Windows, Icons, Menus, and Pointing Devices (WIMP):

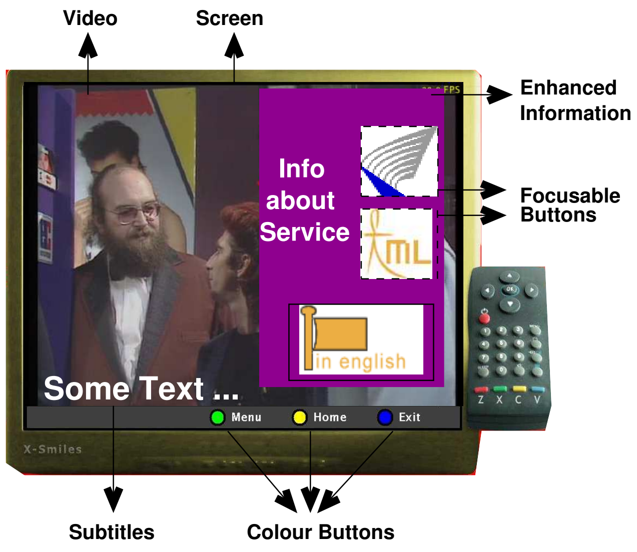
- Virtual reality, because of its special input/output devices.
- Embedded systems.
- Notebooks, because of their handwriting and speech recognition capabilities.
- Hypermedia for hyperlinked multimedia.

He defines digital television receivers as a combination of embedded systems and hyperlinked multimedia devices, and thus falling into the category of non-WIMP devices. Figure 5.2 shows the differences between WIMP [74] and non-WIMP interfaces. In the first case, the user needs different contexts to perform different activities at the same time (e.g., to browse the Internet while drawing an image), while in the second case only one context is needed (i.e., the TV screen).

Because the WIMP paradigm does not fit digital television, he proposes a new metaphor called the virtual channel. This metaphor, depicted in Figure 5.3, considers traditional broadcast as a set of segments (i.e., videos), which can include extra information (e.g., the score of a football match). A virtual channel, on the other hand, is a "dynamic synthesis of discrete video,



(a)



(b)

Figure 5.2: Comparison Between (a) WIMP and (b) Non-WIMP Interfaces.

graphics and data control at the consumer's digital Set-top Box (STB)" [29]. In order to produce the final output, the STB can use different sources, such as locally stored material, real-time broadcast, and Internet resources.

The basic elements of the virtual channel paradigm, as depicted in Figure 5.4, are the virtual channel, playlists, video clips, and related information. Video clips are defined as distinct A/V items (e.g., a news story) that can include extra information (e.g., text or an application). These video clips are arranged as a continuous flow, forming playlists. Finally, these playlists are manipulated and integrated by the virtual channel. This paradigm is de-

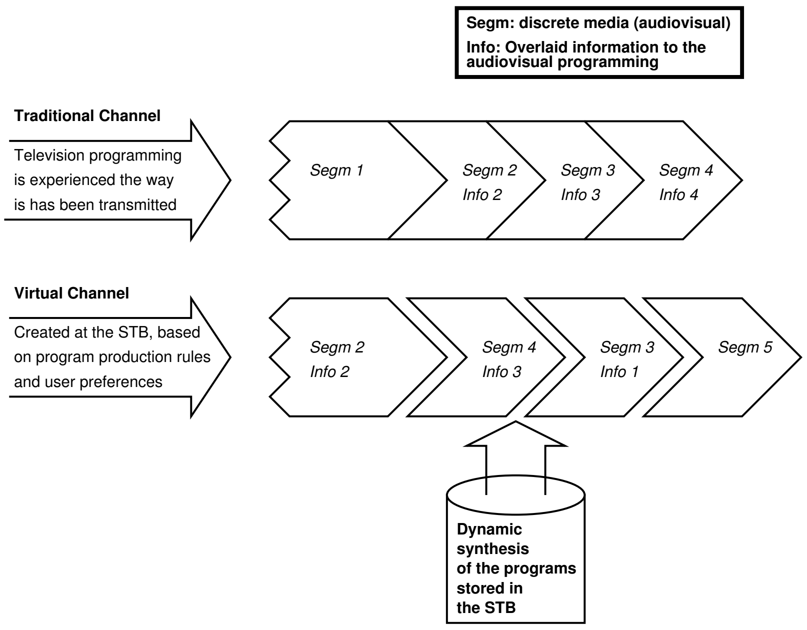

Segm: discrete media (audiovisual)
Info: Overlaid information to the audiovisual programming
Traditional Channel
Television programming is experienced the way is has been transmitted
Segm 1
Segm 2 Info 2
Segm 3 Info 3
Segm 4 Info 4
Virtual Channel
Created at the STB, based on program production rules and user preferences
Segm 2 Info 2
Segm 4 Info 3
Segm 3 Info 1
Segm 5
Dynamic synthesis of the programs stored in the STB

Figure 5.3: Comparison of Traditional and Virtual Channels [29].

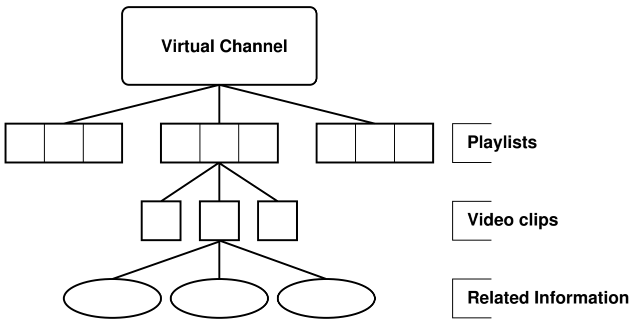


Figure 5.4: Virtual Channel Elements [29].

fined at a high abstraction level, so it does not force any programming language, but it could be implemented using, for example, MPEG-4 or SMIL. Chorianpoulos, in his thesis, provides a high-level implementation for MS TV platform.

Summary

In conclusion, the requirements that the proposed graphics architecture should meet are the following:

- Extensions to GEM
  - **Levels of abstraction:** a layered architecture is preferable, so developers can implement at any level, depending on their goals. Some examples include Java or XML, as discussed in Chapter 2.
  - **Presentation:** visualisation and integration of different media, such as 2D and 3D graphics and A/V content.
  - **User interaction:** various user input mechanisms such as keyboard, keypad, and remote control. Digital television receivers usually lack a pointing device.
- Gaming Capabilities
  - **Input Device:** joystick or game pad.
  - **Temporal Dimension:** the temporal dimension of the applications need to be supported (e.g., animations).
  - **Multiple Scenes:** the game world is composed of different scenes, so a scene manager is required.
- Non-WIMP Metaphor
  - **Metaphor:** the screen metaphor, because the desktop metaphor does not suit devices, such as digital television receivers, intended to show visual content and advanced user interfaces [29]. The screen is considered as a scene composed of different multimedia objects.
  - **Windowing System:** because digital television receivers show one scene at a time, the underlying architecture should be scene-based instead of windows-based. Hence, a scene manager would be more appropriate.

5.3 Graphics Architectures Review

This section includes a review of different graphics architecture models, analysing their suitability for digital television receivers. The models that are covered in this section include classic architecture as defined by Myers [115], Java graphics systems, and scene-based architectures. First, the classic architecture as layered into the windowing system, toolkit, and high-level tools, is studied. The toolkit concept is essential to the Java graphics system that is explained later in this section. Finally, this section concludes with a study on scene-based models such as MPEG-4 and GEM.

Graphical User Interface Classic Architecture

Myers [115] divides the user interface software into three basic components: windowing system, toolkit, and higher-level tools. The windowing system is in charge of monitoring and controlling graphical contexts by separating them into different regions, called windows. The windowing system is divided into a base layer (or window system) and a user interface layer (or window manager) [114]. The former includes the input and output models, while the latter allows the user to control the windows and defines their behaviour. The toolkit is a library of widgets that can be used by any application. It assures consistency, because all applications will have the same look and feel, and it makes the development of Graphical User Interfaces (GUIs) easier. Finally, higher-level tools come in a number of flavours; some examples include event-based, declarative, and constraint languages.

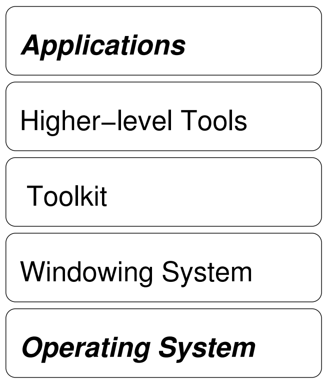


Figure 5.5: Classic Architecture [115].

The windowing system is the interface of the operating system and as such is in charge of managing the input and output models. The input model handles the devices the user can use to interact with the system (e.g., pointing devices), while the output model displays the graphical representation of the user interface. In relation to the output model, the windowing system can either provide a simple imaging model or a sophisticated graphics package (e.g., as with Macintosh) [114]. The major advantage of the first option is its simplicity, and thus its efficiency. In addition, extended graphics functionality can be built on top of the windowing system. The second option, on the other hand, results in a more consistent environment, because all applications use the same graphics package. Moreover, it permits developers to implement more attractive (e.g., 3D graphics-based) user interfaces for the windowing system.

As mentioned before, the toolkit is a library of widgets or user interface elements that simplifies the developer's task. Even though some toolkits provide a collection of procedures, more natural implementations use the object-oriented paradigm. The most important functionalities provided by a toolkit include:

- **Interaction:** to handle user input.
- **Canvas Operations:** provide the rendering region, canvas [128], and

graphics primitives such as the rectangle or triangle.

- **Set of Widgets:** predefined user interface elements (e.g., Button, TextField) and subregions to place them in (e.g., Container, Panel).
- **Graphical Layout:** to control the location of the widgets.

Currently, applications are expected to run on a variety of devices and operating systems. Because developing different versions of the same application for each targeted device is not feasible, a generic or abstract GUI library is needed. An abstract user interface framework is called a virtual toolkit [115] and hides the differences among various platform-specific implementations. A virtual toolkit provides a generic interface for software developers and uses, at runtime, the toolkit installed in the device. For example, AWT is a virtual toolkit that runs on different operating systems such as Macintosh, Windows, and Linux.

For example, Linux systems use X-Window, a client-server architecture, in which X-Server mantains exclusive control of the display and Xlib handles all low-level communication tasks. It follows the motto, ”we provide mechanisms but not policies”, meaning that X-Window does not specify any toolkit or window manager. Currently, the most popular approaches are KDE[1] and GNOME[2], which are complete desktop environments similar to commercial solutions. KDE includes a toolkit called QT[3] and a window manager, Kwin[4]. GNOME includes GTK [5] as a toolkit and does not impose any window manager. Hence, the user can decide which one she prefers; some examples include Sawmill[6], Enlightenment[7], and Fluxbox[8]. Finally, because X-Window has been patched but not upgraded for the last twenty years, X.org[9] has taken responsibility for cleaning up the software and upgrading it to satisfy current needs, offering improvements such as font management, graphics card support, and composite functionality (i.e., the alpha channel). Figure 5.6 depicts the structure of X-Window and shows how the order of the layers defined by Myers do not have to be strict and, for example, the toolkit could be below the window manager.

Toolkit-Based Architecture

Java graphics architecture is based on a virtual toolkit called AWT, which includes support for interaction (i.e., events listeners), canvas operations (i.e., *java.awt.Graphics*), a set of widgets, and a graphical layout. The basic class of AWT is *java.awt.Component*, which represents a widget. Following Sun’s terminology, there are two ways of implementing components: heavyweight and lightweight. In the first, each Java widget uses a native counterpart. AWT contains, as well, a native ”peer” - an object that takes care of

[1]http://www.kde.org/
[2]http://www.gnome.org/
[3]http://www.trolltech.com/
[4]The user can choose to use another window manager.
[5]http://www.gtk.org/
[6]http://sawmill.sourceforge.net/
[7]http://www.enlightenment.org/
[8]http://fluxbox.sourceforge.net/
[9]http://www.x.org/

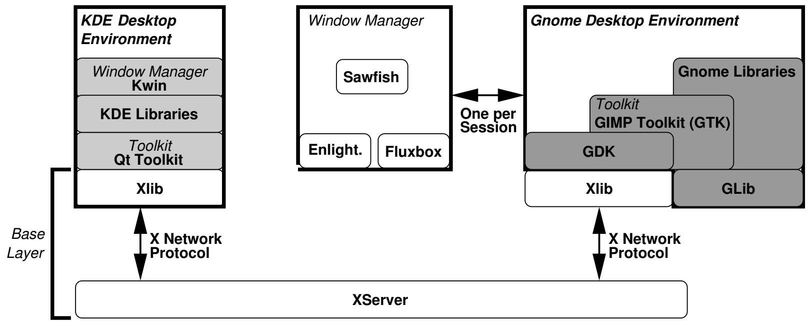


Figure 5.6: Desktop Environments for X-Window.

handling the interaction between the AWT Java object and the peer. In the lightweight approach, the toolkit permits the developer to create her own components. In this case, the drawing of the component content must be done using AWT graphics operations. In addition to *java.awt.Component*, AWT defines *java.awt.Container*, in which components can be placed and layout mechanisms to arrange them are provided.

The class hierarchy of AWT is depicted in Figure 5.7. The main class is *java.awt.Component*, and all widgets and containers are extended from this class. Each container has a reference to a single LayoutManager and any number of components, which can be containers themselves.

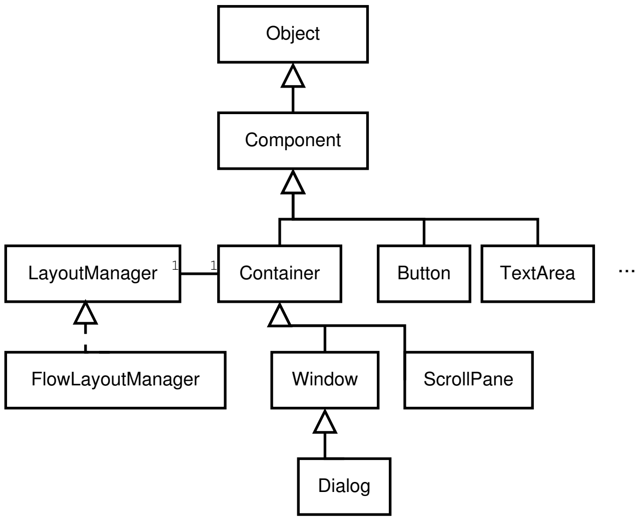


Figure 5.7: UML Class Diagram of a Subset of AWT.

Because of the evolution of Java language, other Java toolkits in addition to AWT are available for developers. The most popular for desktop development is Swing; HAVi is intended for digital television services, and Personal Profile is defined for J2ME Connected Device configuration. All these toolkits rely on the basic AWT principles of *java.awt.Component*, *java.awt.Container*, and *java.awt.Graphics* classes. On the other hand, Connected Limited Device Configuration has defined its own toolkit, called Limited Device User Interface (LCD UI), which does not follow AWT. Figure 5.8 depicts the different Java toolkits.

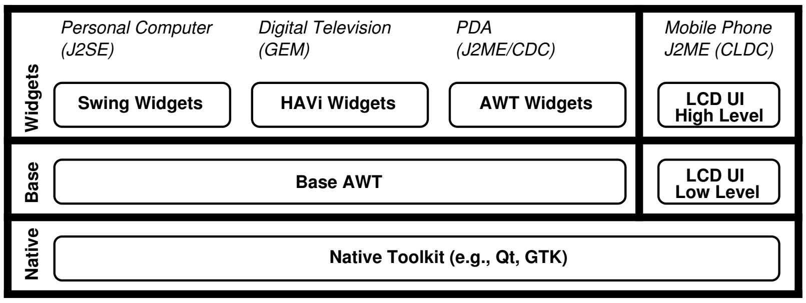


Figure 5.8: Overview of Java Toolkits.

MPEG-4 Graphics Architecture

The MPEG-4 standard provides a way to integrate video, 2D objects, and 3D objects in the same scene by using BIFS. An MPEG-4 scene is defined as a directed graph composed of nodes. Some of the nodes defined in the MPEG-4 standard include[10] [25, 27, 37, 154, 159]:

- **Top:** the root of the graph. It is a description of the scene (e.g., OrderedGroup, Layer2D, and Layer3D).
- **Grouping:** containers of multimedia objects. There are containers for both 2D and 3D nodes (e.g., OrderedGroup), only for 2D nodes (e.g., Transform2D), and only for 3D nodes.
- **Sensor:** nodes capable of detecting events (e.g., Anchor, TimeSensor, and TouchSensor).
- **Graphical Primitives:** the Shape node defines a geometry. It includes two fields (and nodes):
  - Geometry: defines the shape itself (e.g., Circle, Rectangle, Text, and Sphere)
  - Appearance: defines its visual aspect (e.g., texture and material). The textures include media objects such as MovieTexture and ImageTexture.
- **Face:** provides the means to integrate synthetic 3D human-like objects in the scene.
- **Children:** the leaves of the graph. These could be, for example, Grouping, Sensor, and graphical primitives nodes.

In relation to the requirements identified for toolkits, MPEG-4 provides:

[10]http://gpac.sourceforge.net/tutorial

- **Interaction:** MPEG-4 defines Sensors to detect events (e.g., temporal and user input events). In addition, it provides a node called Route to distribute the event from its origin to its destination. Interactions such as Resize, Relocate, Translate, and Remove nodes use the Conditional and Valuator nodes, while more complex interactions can be implemented using ECMAScript (i.e., Script node) or Java.

- **Canvas Operations:** the root of the graph is the rendering region. The developer can then use graphical primitives to define its shape and appearance.

- **Set of Widgets:** MPEG-4 does not define widgets, but the developer can implement any customisable one. For example, by using TouchSensor she can create a button or a textfield by using InputSensor. In addition, as mentioned above, MPEG-4 defines containers to place the objects.

- **Graphical Layout:** in MPEG-4 each object defines its local coordinates, which then are translated into scene coordinates, taking into account the parents of the object. Unfortunately, more complex and automatic layout is not permitted (e.g., a flow layout).

In order to clarify the scene graph composition, Figure 5.9 shows a composition of A/V stream and enhanced information, while Figure 5.10 depicts its scene graph in MPEG-4.

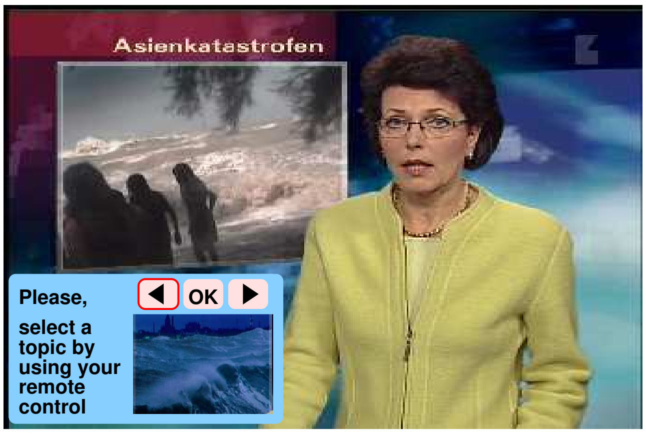
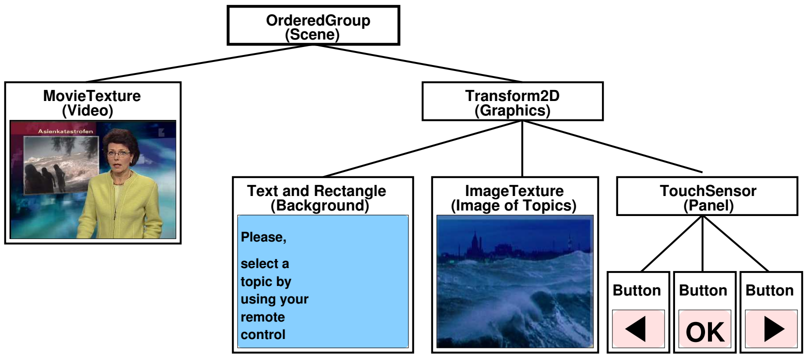


Figure 5.9: Example of A/V Stream and Enhanced Information.

Globally Executable MHP

In GEM, as depicted in Figure 5.11, the television display is composed of three different panes, ordered from back to front: background, video, and graphics layers. The background image, an I-frame or video drips, can be controlled by using HAVi, while the location, resolution, and aspect ratio of the video is controlled by using JMF. The video, in MPEG-2 format, is



OrderedGroup (Scene)
MovieTexture (Video)
Transform2D (Graphics)
Text and Rectangle (Background)
Please, select a topic by using your remote control
ImageTexture (Image of Topics)
TouchSensor (Panel)
Button
Button
Button
OK

Figure 5.10: Elements of Figure 5.9.

hardware decoded and visualised in the video layer. Still, GEM allows presentation of the video as an AWT component rendered in the graphics layer. Finally, the applications follow the AWT painting mechanism, with some alpha channel improvements from HAVi, and are rendered in the graphics layer.

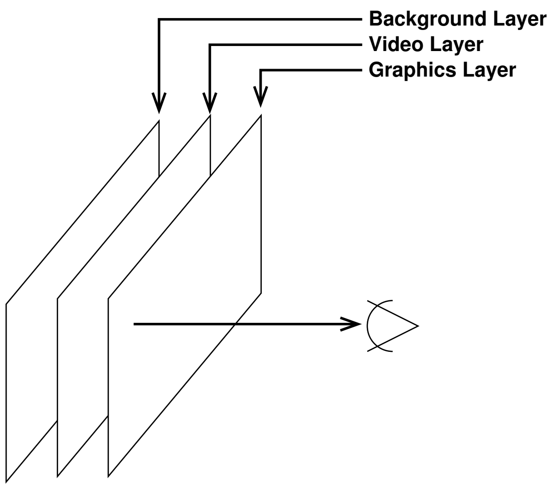


Figure 5.11: MHP Layer Overview [51, 52].

HAVi is the package included in DVB-J for user interface development. HAVi extends Personal Java's AWT package by including a television-specific means of presenting and interacting with applications (e.g., remote control and a specific set of widgets). The most relevant characteristics from HAVi are the following:

- **Lightweight framework:** it defines a set of peerless components, extending *java.awt.Component*. Each component has an associated Look class that is in charge of its rendering, while its feel is defined by a number of interfaces (e.g., Navigable).
- **User Input:** the input device defined is the remote control.

- **Transparency:** since the graphics model is scene based and constructed from three different layers, essential features such as transparency[11] and Z-index are defined in HAVi.

HAVi extends AWT by including a canvas class, called HScene, whose functionality is similar to *java.awt.Window*. In addition, the event mechanism from AWT has been improved to meet digital television needs, for example, to transfer the focus from one widget to another. Finally, a set of widgets are defined. These are lightweight and separate the look, the content, and the feel, according to the Model-View-Controller (MVC) paradigm. Some examples of widget behaviour, defined in HAVi, are visible, navigable, and actionable. Visible widgets, which extend AWT's component, are the base of all widgets. Navigable components are those that can be navigated by using the remote control keys. Actionable components are those that change the state of the application (e.g., buttons). Each widget has, as well, an associated Look class, which is in charge of the rendering. This class can be overridden in order to attach different views to a widget. The class diagram of the HAVi set of widgets is depicted in Figure 5.12.

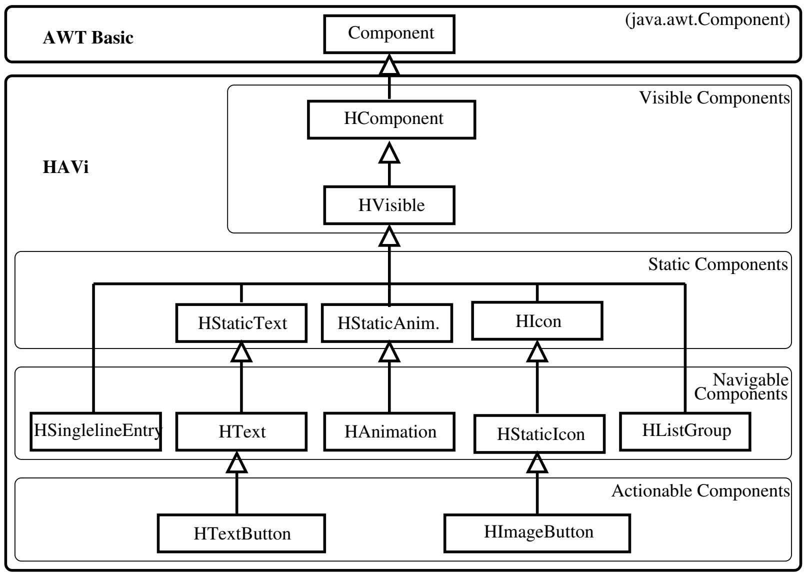


Figure 5.12: HAVi Set of Widgets Overview [P3].

5.4 Proposed Model

This section analyses the suitability of the previous architectures for high-end digital television receivers, taking into account the basic requirements for such devices, as introduced in Section 5.2.

First, the classic architecture, although deriving from the WIMP concepts of the 1980s, can be applied to almost any kind of multimedia system.

[11] In fact DVBColor defines the transparency, but HAVi defines how to treat it.

The main problems for digital television receivers are the following.

- **Pointing Device:** digital television receivers usually do not support a pointing device input mechanism. Hence, the window manager, which heavily relies on it, should be reconsidered.

- **Video and Graphics Integration:** the integration of video and graphics is not straightforward. In addition, it does not include a temporal dimension, but only reacts to user inputs.

- **Windowing Manager:** digital television receivers show one graphical context at a time, making scene-based architecture preferable.

Second, Java Graphics architecture is based on the classic architecture, so it shares similar kinds of problems. For example, the root containers in AWT are *java.awt.Frame* and *java.awt.Window*, which are moved, dragged, minimised, and maximised using a pointing device.

Third, in the personal desktop environment, an MPEG-4 player can be integrated at the base layer of the windowing system, and thus no modifications to the classic architecture are needed. In digital television receivers, an MPEG-4 player can replace the whole architecture, transforming the system from windows based to scene based, which better suits digital television and game console environments. The main benefits of MPEG-4 are seamless integration of 2D and 3D graphics, support for the temporal dimension, the use of scene-based architecture, the definition of different entry points (e.g., MPEG-J and XMT), and several user interaction mechanisms (e.g., navigation between scenes, buttons).

Fourth, the graphics architecture of GEM is based on three layers: background, video, and graphics. These layers are not integrated, but rather use overlaying. Hence, GEM expects the use of transparency for composition purposes. However, as described in Subsection 4.2, some extensions such as real integration of video and services in the graphics layer and 3D graphics support should also be considered.

Taking these issues into account, a graphics model for digital television receivers is defined as depicted in Figure 5.13. The model is open and extensible, meaning that different configurations can be obtained depending on the device requirements. The model is also layered and takes into account the threshold (i.e., difficulty of use) and ceiling (i.e., expressive power) measures [116] of each layer. The layers are:

- **Hardware Abstraction (HAL):** renders the final graphics output.

- **Graphical Context:** a cross-platform abstraction of a rendering region. In terms of Olsen [128], it is the abstract canvas. In addition, it provides a set of multimedia output primitives and receives input events.

- **Graphical Environment:** provides the means to control different graphical contexts (e.g., visibility), translating and delegating inputs to the correct context.

- **GUI Toolkit:** provides a set of ready-made user interface widgets.

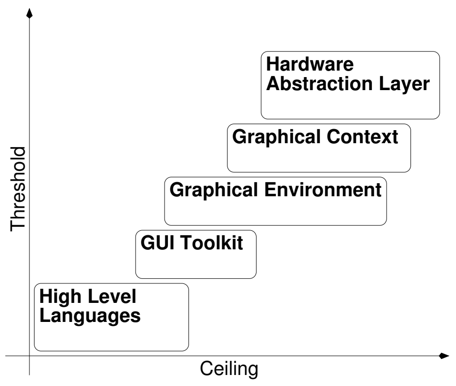


Figure 5.13: Proposed Model [P9].

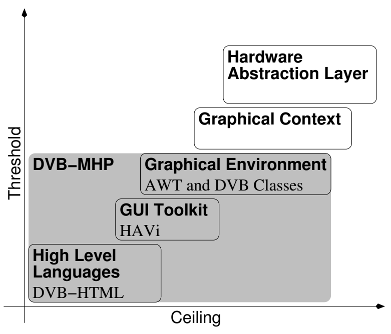
- **High-Level Languages (HLL):** includes declarative languages. These are easy and quick to apply, but restricted in their expressive power.

The proposed model is based on the classic architecture, but includes certain concepts from MPEG-4 and GEM:

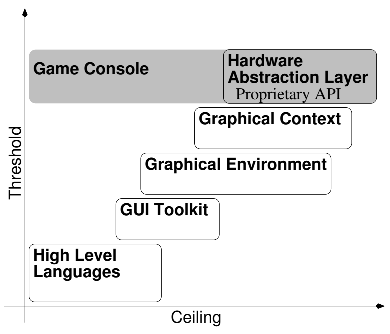
1. The User Interface Layer of the windowing system is renamed the graphical environment, because the windows-based architecture does not apply anymore in digital television receivers. Hence, the graphical environment corresponds to a particular scene.
2. The base layer of the windowing system is explicitly divided into two layers:
   (a) HAL: in charge of rendering the final graphics output.
   (b) Graphical Context: apart from the abstract canvas, it provides libraries for 2D and 3D graphics and video rendering.
3. Temporal Dimension: applications can be synchronised with video content, for example, by HLL such as SMIL or Timesheets.

The proposed model takes into account the expressive power and difficulty of use of each of the layers. Hence, developers can choose the appropriate tool when developing a service. For example, limited-interaction information services can be implemented using high-level Languages, while resource-consuming 3D games can be implemented using HAL tools.

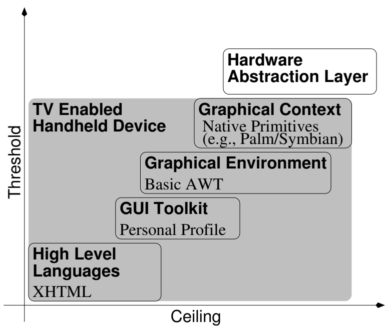
In order to generalise the model, its suitability for other media stations is analysed. First, an MHP-compliant digital television receiver implements the graphical environment and GUI toolkit as DVB-J, and the HLL as DVB-HTML. Second, a game console provides those layers closer to the hardware and can support, for example, HLL to get online help for a particular game. Finally, a television-enabled personal digital assistant using DVB-H and a handheld device with game console capabilities implement all the layers of the model. These specific cases are depicted in Figure 5.14. In conclusion, the proposed model is suitable for all kinds of media stations and devices intended for entertainment, while other devices (e.g., those intended for information management) follow the classic architecture.



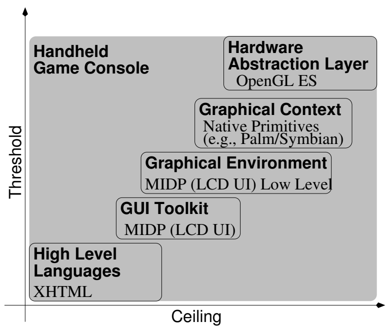
(a) MHP Compliant Receiver.



(b) Game Console.



(c) TV Enabled Handheld Device.



(d) Handheld Device with Game Console Capabilities.

Figure 5.14: Configurations of the Model.

5.5 Summary

This chapter has proposed a graphics architecture for digital television receivers taking into account their specific requirements. First, the requirements derived from their software architecture and their nature as television and entertainment devices were presented. These requirements included, for example, seamless integration of multimedia objects (e.g., video and 3D graphics). Then, different graphics architectures models such as windows, toolkit, and scene based were reviewed and their suitability for digital television receivers was analysed. Based on this analysis, a graphics architecture model for digital television, based on that used in desktops, has been defined. This architecture is divided into five layers. First, the HAL composes and renders the final scene. Second, the graphical context defines a scene composed of video, 2D graphics, and 3D graphics. Third, the graphical environment is in charge of controlling different scenes. Fourth, a GUI toolkit provides a set of widgets for easy creation of user interfaces. And fifth, HLL are declarative tools that do not require programming skills, thus making application development easy and fast. The proposed model takes into account the expressive power and difficulty of use of each of the layers. Hence, developers can choose the appropriate tool when developing a service. For example, limited-interaction information services can be implemented by using HLL, while resource-consuming 3D games can be implemented using HAL tools.

6 REFERENCE IMPLEMENTATIONS: OTADIGI AND UBIK

Chapter 6: Reference Implementations
(Concept Implementation)
Otadigi: DTV Broadcast System
Ubik: Configurable DTV Receiver

Note: Section 6.2 is an extended version of [P9].

In order to examine the topics presented in this thesis, two reference implementations were developed: Otadigi and Ubik. Otadigi has been introduced in Chapter 2 as an example of a digital terrestrial television network, while Ubik is a prototype implementation of a digital television receiver. Otadigi permitted me the study of the current state, potential, and limitations of commercial television receivers. On the other hand, Ubik allowed me to look towards the future and to experience how the proposed extensions to GEM can be implemented.

6.1 Reference Implementation 1: Otadigi

I have been a member of the Otadigi technical committee since 2002. I have taken decisions and collaborated in the set up of the system. In addition, I have been in charge of developing the services that are transmitted in the network. Some of the topics studied with Otadigi include:

- Development of DVB-J applications.
- Usage of high-level languages such as SMIL.
- Usage of the interaction channel.

Services

The first service developed for the Otadigi network was a simple demo application[1], depicted in Figure 6.1. The reasons for implementing such an application were, first, to put into practise the theory learnt about the MHP standard, and, second, to test if the set of widgets described in [P3] could be used in commercial receivers. This application was presented in [P2] running on a demo version of a digital television software stack and in [P6] running on a commercial receiver.

In addition to the simple application, a Navigator[2] was implemented, as depicted in Figure 6.2. This application was presented in [P2] running on a

[1]The original author of this application is C. Peng. I have upgraded it for use in a commercial receiver. The source code of the application can be downloaded from the OpenMHP project (http://www.openmhp.org).

[2]The original author of this application is C. Peng. I have upgraded it for commercial receiver. The source code of the application can be downloaded from the OpenMHP project (http://www.openmhp.org).

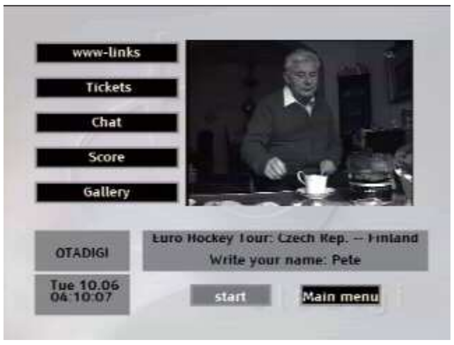


Figure 6.1: DVB-J Application Running in I-can [P6].

demo version of a digital television software stack and in [P6] running on a commercial receiver.

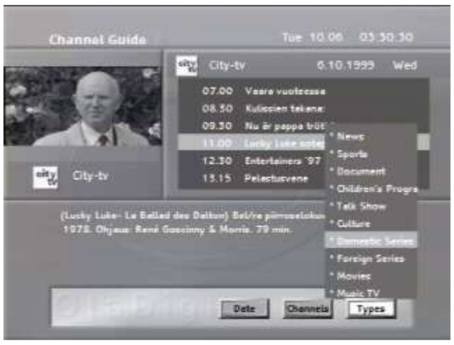


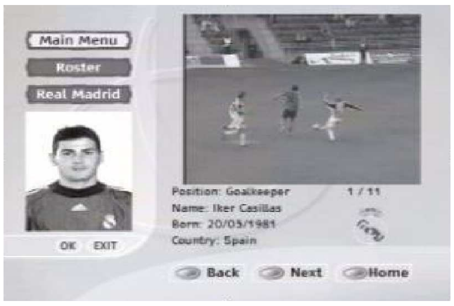
Figure 6.2: EPG Running in I-can [P6].

Finally, in order to study high-level languages, a SMIL[3] player for commercial receivers was studied, and two possible usages in the digital television environment were analysed: enhanced television and a Super Teletext service. Figure 6.3 (a) shows an enhanced service that provides extra information about a football match, while Figure 6.3 (b) presents an implementation of a Super Teletext service. Both services can access the information by retrieving the object carousel or using the interaction channel. The services were presented in [P5] and the SMIL player implementation was presented in [P4].

Results

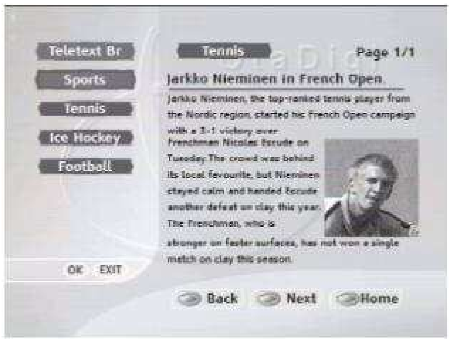
Having the opportunity to play around with a real broadcast environment was a positive experience in terms of the findings of this thesis. It provided experience and knowledge of the television production chain (i.e., development of services, transmission, and reception). The most relevant

[3]The original author of the SMIL player is K. Pihkala. J. L. Lamadon and me developed the services presented in this subsection. The SMIL player can be downloaded from X-Smiles (http://www.xsmiles.org).



Position: Goalkeeper
Name: Iker Casillas
Born: 20/05/1981
Country: Spain

(a) Enhanced Television.



Jarkko Nieminen in French Open.

Jarkko Nieminen, the top-ranked tennis player from the Nordic region, started his French Open campaign with a 3-1 victory over Frenchman Nicolas Escude on Tuesday. The crowd was behind its local favourite, but Nieminen stayed calm and handed Escude another defeat on clay this year. The Frenchman, who is stronger on faster surfaces, has not won a single match on clay this season.

(b) Super Teletext.

Figure 6.3: SMIL Player Running in I-can [P5].

conclusions derived from the implementation and transmission of the services mentioned is that current commercial receivers[4] are not adequate for interactive television. For example, major problems were faced when implementing the SMIL player and services. First of all, current receivers lack the appropriate XML parsers (i.e., the parsers have to be downloaded)[5] and storage capacity and provide slow processing power, which resulted in high application startup times. Second, because the return channel was an analogue modem, using it for downloading content was not an alternative. Other results, such as the use of I-frames and methods of controlling the video size and location, were already mentioned in Chapter 2.

6.2 Reference Implementation 2: Ubik

After studying the limitations of commercial receivers, I started the development of a prototype. This implementation, called Ubik, was presented in [P7, P8, and P9]. There were two reasons for creating this implementation. First, an open platform, unlike commercial receivers, in which to test the different profiles presented in Chapter 4 was available. Second, the graphics model proposed in Chapter 5 was applied to a real environment, thus demonstrating the feasibility of the model. This section describes Ubik, summarising the findings presented in [P9]. First, the implementation decisions made during Ubik's development are explained. Second, the services Ubik can provide are described, thus showing the implementation's advantages from the viewpoint of the user.

System Architecture

The configuration of Ubik, depicted in Figure 6.4, is composed of the following layers:

- **Operating System and TV Reception:** Ubik is a Linux-based terminal. Linux was selected because it is open and configurable. In addition, its kernel version 2.6 includes modules for DVB-T reception and drivers for some TV cards, such as Nova-t PCI. Moreover, open-source projects such as linuxtv and linuxstb provide software intended for tuning channels (e.g., *tzap* or *dvbtune*) and downloading content (e.g., *dvbdata*). Ubik uses tzap to tune to the TV channels and libsoftmpeg to decode the DVB A/V content and pipe it to the HAL layer (i.e., DirectFB).
- **Hardware Abstraction Layer:** in Ubik, DirectFB is in charge of rendering the final graphics output using the framebuffer device, which frees the system from X-Window. In addition to DirectFB, Ubik can use OpenGL primitives and thus directly access the 3D graphics card drivers. Finally, DirectFB composes and renders the final graphics output as a combination of DVB A/V content, graphical context output, and OpenGL primitives.

[4]The transmissions were carried out during the year 2003.

[5]April 2005: MHP 1.1.2 [58] includes an XML-parsing API based on the one used for Java in mobile phones.

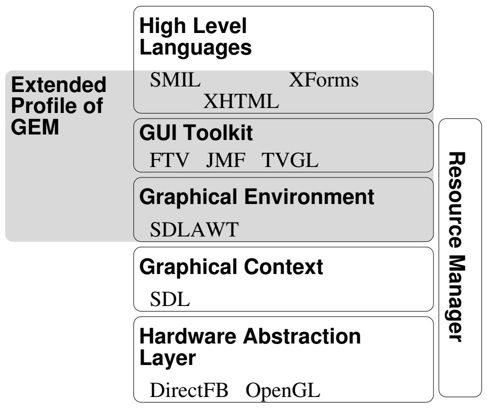


Figure 6.4: Ubik's Configuration (Please note that the order of layers is reversed from the model presented in Figure 5.13 because the threshold and ceiling are not shown here.) [P9].

- **Graphical Context:** the graphical context of Ubik is a cross-platform library called SDL. SDL provides multimedia primitives and an abstract canvas called *SDLSurface*, and supports different input devices. SDL can run on top of DirectFB, taking advantage of the graphics card hardware acceleration. In addition, it allows the use of OpenGL for 3D graphics, generating a *SDLSurface* when rendering. Finally, Ubik provides tools for playing any A/V content format (apart from the DVB stream) by generating a *SDLSurface* directly from video memory.

- **Graphical Environment:** following the GEM specification, the graphical environment of the system consists of a subset of AWT. It does not include any graphical widgets, because those are considered part of the GUI toolkit, but only those Java classes required to paint and get events. For Ubik, it was implemented in a new back-end for Kaffe's[6] AWT running on top of SDL.

- **GUI Toolkit:** in Ubik, developers can use three kinds of objects: widgets, video, and 3D objects. First, as specified by GEM, Ubik provides both its own implementation of HAVi, a package called Future TV (FTV), and JMF support. FTV implements a number of widgets and can handle television-specific input events. It will be available as part of the open-source OpenMHP[7] initiative. JMF, on the other hand, was implemented by extending Sun's JMF, which includes a new renderer plug-in, SDLRenderer, used instead of the commonly used XlibRenderer. Finally, a Java package binding OpenGL functionality, TVGL, was implemented for Ubik. This package permits the use of 3D graphics primitives, such as textures, at the Java level.

- **High-Level Languages:** in order to support the XML-based content,

[6]http://www.kaffe.org
[7]http:www.openmhp.org

a Java-based cross-platform XML user agent, X-Smiles[8], was integrated into Ubik. X-Smiles supports different W3C recommendations such as XHTML, XForms, and SMIL.

Apart from the layers described above, the resource manager is in charge of controlling the different layers of the system (i.e., the DVB A/V stream, native, Java, and XML layers). It also provides inter-process communication mechanisms to allow higher-level tools, such as HAVi or JMF, to control lower-level functionalities (e.g., the location of the DVB A/V content).

The software architecture of Ubik is depicted in Figure 6.5, showing the components that have been actually used for the architecture presented in Figure 5.1. The basic principle used in its implementation was to reuse as many open-source projects as possible. One of the benefits of my approach is that Ubik is modular, meaning that other components can be used without altering the whole system. For example, SDL was included as a cross-platform library, but it can be removed for those receivers in which the manufacturer does not need to provide game capabilities. If SDL is not used, the Java environment just needs a back-end for DirectFB. Another example is that the use of libsoftmpg can be replaced by a combination of dvbstream[9] and a video player (e.g., MPlayer[10] and Xine[11]). Because these video players already embed the composition of video and 2D objects, they were difficult to adapt to Ubik's special needs. At the higher levels, other browsers such as Ortikon[12] or Espial[13] can be used instead of X-Smiles, but they do not support XML compound documents. In conclusion, by reusing a number of open-source projects, the whole system was constructed using a reduced number of people and at a minimal cost (e.g., using a PC and Nova-t PCI card). Other additional costs for commercial production will be the license fees of, for example, MHP.

Services

In order to test whether Ubik meets the requirements identified for High-End Interactive television terminals, several services were implemented based on those requirements. These services include:

- **DVB A/V:** users can normally watch TV and zap among channels.
- **Native Rendering:** Ubik can integrate any kind of multimedia object. Some practical uses of this feature include enhanced information over the DVB A/V content as depicted in Figure 6.6, attractive resident applications (e.g., 3D games), and subtitles. Ubik also includes a simple menu, which allows the user to launch any service mentioned in this subsection while watching a TV channel.
- **Precompiled Native Game:** apart from resident games, the user can access a portal offering different 3D graphics-based games. Currently,

[8]http://www.xsmiles.org
[9]http://sourceforge.net/projects/dvbtools
[10]http://www.mplayer.org
[11]http://xinehq.de
[12]http://www.ortikon.com
[13]http://www.espial.com

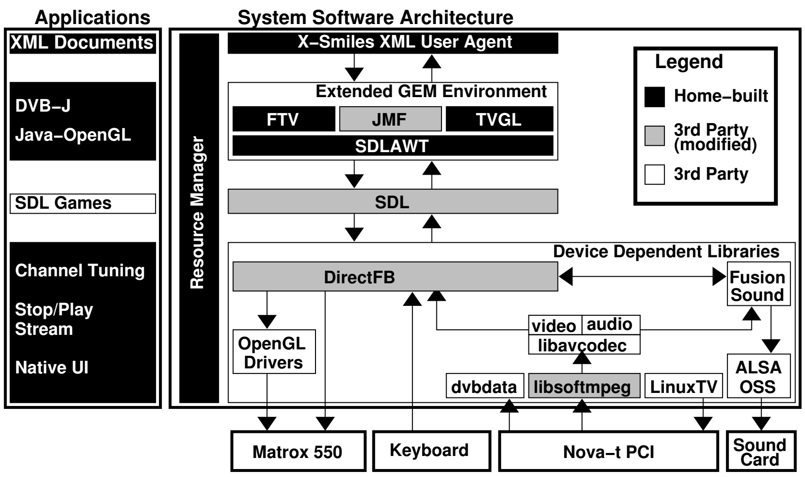
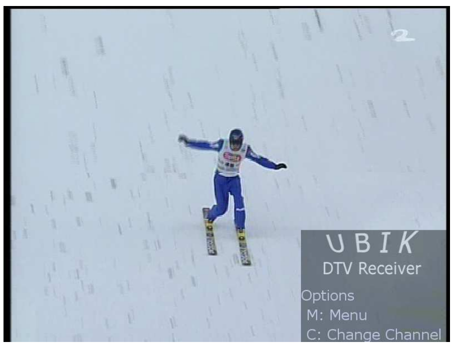
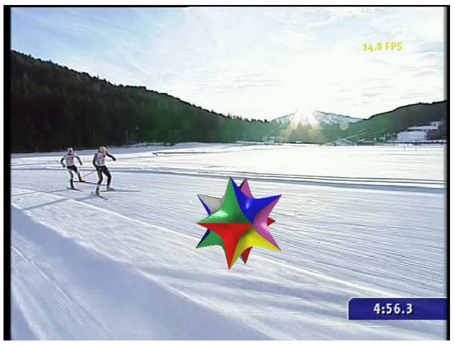


Figure 6.5: Ubik's Software Architecture [P9].



(a) Enhanced Information Overlay



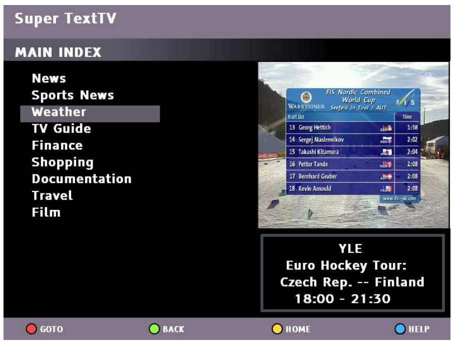
(b) Transparent 3D Graphics Demo

Figure 6.6: Services Running in Ubik (Native Rendering) [P9].

several games are available for SDL, the graphical environment of Ubik. Figure 6.7 shows one of them.

Figure 6.7: Tuxracer (http://tuxracer.sourceforge.net) Running in Ubik [P9].

- **DVB-J Application:** the user can run any DVB-J application in Ubik. For example, Figure 6.8 depicts a DVB-J-based Super Teletext.



YLE
Euro Hockey Tour:
Czech Rep. -- Finland
18:00 - 21:30

Figure 6.8: DVB-J-Based Super Teletext Running in Ubik [P9].

- **XML Based Content:** the digital television user can access declarative content as well. Ubik provides the two profiles discussed in Chapter 4:
  - Basic: intended for information services. This profile is based on XHTML and CSS. A typical use case is the Super Teletext service, as depicted in Figure 4.7.
  - Internet Access: Ubik includes support for XML compound documents, combining recommendations from XForms (user interaction) and SMIL (temporal and spatial dimensions). Another option, CSS + XHTML 2.0 + Timesheets, is supported as well. In both cases, XForms was selected for user interaction over

scripting because HLL are intended for fast and easy development (i.e., low threshold and ceiling). For example, Figure 4.8 depicts an e-learning Web portal running in Ubik. The portal contains lectures in which video and slides are synchronised. When a lecture is over, an exam developed in XForms is administered to the student.

6.3 Summary

This chapter has described two reference implementations used to study the concepts proposed in this thesis. First, Otadigi is a digital television network supporting DVB-T that has been used to study the current state of commercial receivers presented in Chapter 2. Otadigi permitted me to identify the limitations of current MHP implementations. Some examples include the limited interaction channel capabilities and the lack of resident XML parsers, which forces the service provider to broadcast them along with the XML based documents. Second, Ubik is a prototype implementation of a High-End Interactive television receiver. It is based on the graphics architecture model proposed in Chapter 5, thus validating the suitability of the model. Finally, several representative services of the profiles for digital television described in Chapter 4 were tested in Ubik.

7 CONCLUSIONS

First, this thesis aimed to show that the decision about which content authoring format to use when implementing services depends on the development purpose. For example, certain simple applications (e.g., advertisements) can be coded using XML-based languages, but resource-demanding applications (e.g., Doom-type interactive games) should be coded at a different level. Thus, a graphics architecture model for media stations was proposed. The architecture is divided into different layers, depending on their difficulty of use and expressive power. Second, this thesis is based on the GEM standard, a worldwide alternative for digital television receivers. Because the capabilities of devices are evolving constantly, this thesis proposed a set of extensions to GEM. Finally, because manufacturers need to differentiate their products, this thesis recommended improvements to their graphics architectures, such as convergence with game consoles.

Currently, the main concern regarding digital television is ensuring a smooth transition from its analogue counterpart, for which the timely definition of a viable, open, and platform-independent middleware (i.e., horizontal-market) standard is a key issue. Such a standard should, as well, take into account that the transition will follow gradual steps and that different user groups will have different demands. With these goals in mind, different regional consortia have proposed their own standards, such as the DVB-defined MHP. In order to provide a worldwide solution, DVB has published GEM. GEM is a valid starting point that takes into account the American and Japanese initiatives, but only considers a procedural environment, DVB-J. Hence, the declarative environment and advanced functionality (e.g., 3D graphics) remain to be standardised.

This thesis proposed a declarative environment for GEM. Because the MHP solution, DVB-HTML, was too complex, the proposed declarative environment can be divided into two different profiles. The first one, intended for information services that only require visualisation of content (e.g., Super Teletext), is based on XHTML and CSS, but does not include any scripting or forms, and as such only allows browsing of information. The second is a more complex profile that supports advanced user interaction (i.e., XForms) and the temporal dimension (i.e., SMIL) of XML documents. A representative example of a service for the second profile is an educational Web site that contains lectures synchronised with a slide show. In addition to the declarative environment, this thesis defined a new procedural profile by extending DVB-J with 3D graphics capabilities. Moreover, this thesis recommended its implementation as a thin layer wrapping OpenGL ES functionality.

After defining next-generation digital television receivers, this thesis studied their graphics architectures and proposed a model taking into account their specific requirements. Myers has defined the graphics architecture of desktop devices using the following layers: windowing system, toolkit, and high-level tools. This architecture divides the screen into different regions, or windows, and implies the simultaneous performance of different activities (e.g., surfing the Web and writing a document). Since digital television is intended to visualise one context at a time (e.g., watching television or playing a game), this thesis proposed a modified version of the Myers architecture. The proposed layered model is composed of five layers: the hardware abstraction layer is in charge of rendering the final graphics output; the graphical context is a cross-platform abstraction of the rendering region and provides graphics primitives (e.g., rectangles and images); the graphical environment provides the means to control different graphical contexts; the GUI toolkit is a set of ready-made user interface widgets and layout schemes; and high-level languages are easy-to-use tools for developing simple services.

Finally, in order to understand all the components constituting the digital television chain (i.e., content provider, broadcaster, and receiver), two implementation threads were followed: Otadigi and Ubik. Otadigi is a digital television broadcast system operating in the Otaniemi area of Helsinki, Finland. I am part of Otadigi's technical committee and thus works on its development and makes decisions about the equipment used. In addition, I have been in charge of developing services for Otadigi that are accessed by commercial receivers, resulting in an understanding of the current limitations and potential usages of digital television. Ubik, on the other hand, is a prototype implementation of a configurable digital television receiver based on the proposed graphics architecture model. By designing and implementing Ubik, I had the opportunity to research features not available in current commercial systems, such as 3D graphics support, streamed video, and the use of XML-based languages.

Table 7.1: Research Questions, Answers, and Validation.

Research Question	Answer	Validation
1. What are the limitations of current digital television receiver implementations?	For example, WWW access, Streamed Media, and 3D Graphics support.	Concept Implementation (Otadigi) and Literature Review.
2. How can the GEM standard be extended so it defines a declarative environment?	Procedural (GEM) and Declarative (XHTML + CSS, XHTML 2.0 + SMIL).	Concept Implementation (Ubik).
3. How can the GEM standard can be extended so it includes support for all kinds of multimedia objects (2D, 3D, and video) and WWW access?	3D Graphics extension to GEM Procedural Environment.	Concept Implementation (Ubik).
4. How can manufacturers, following GEM, differentiate their products?	3D Graphics extension to GEM implemented as Java bindings of OpenGL.	Concept Implementation (Ubik).
5. What graphics architecture is needed to implement GEM and the extensions proposed in this thesis? Is this architecture valid for other media stations?	Proposed Model of Five Layers.	Concept Implementation (Ubik).

8 SUMMARY OF PUBLICATIONS AND AUTHOR'S CONTRIBUTION

This chapter gives a brief introduction to the publications included in this thesis and describes the contributions of the author to each. The research intention was to study graphics software architecture for digital television receivers. Because of this reason, the author followed two main implementation threads: Otadigi and Ubik. Otadigi is a digital television broadcast system operating in the Otaniemi area of Helsinki, Finland. The author is part of Otadigi's technical committee and thus works on its development and makes decisions about the equipment used. He studied how to implement and broadcast services in such a network, which was accessed by commercial receivers. Ubik, on the other hand, is a prototype implementation of a high-end digital television receiver based on the proposed graphics architecture model. By designing and implementing Ubik, the author had the opportunity to research features not available in current commercial systems, such as 3D graphics support, streamed video, and the use of XML based languages. In addition, he studied in detail the different software layers of a multimedia device (i.e., the operating system, native, Java, and XML layers). Since both environments are complementary, the author had the opportunity to work with all the components constituting the digital television chain (i.e., content provider, broadcaster, and receiver). The author researched not only the current state of the field, but also further technological improvements. In order to do so, collaboration with many people was needed. Kari Pihkala was the original developer of SMIL, and Mikko Honkala was the original developer of XForms. Carlos Herrero was in charge of Otadigi's object carousel, and Jean Luc Lamadon collaborated with the author in developing Otadigi's services. Chengyuan Peng was the original developer of digital television applications such as the Navigator, Super Teletext, and ice hockey applications. Finally, Juha Vierinen collaborated with the author on Ubik's DVB-T reception (Nova-t drivers and DirectFB rendering of the video stream).

- **Publication [P1]** This paper presented a study of the system software for digital television at a moment when no MHP-compliant digital television receivers were commercially available. In addition to the theoretical study, a reference implementation was described. The author's contribution was the migration of the applications to the Linux and Kaffe platforms. The actual importance of the paper, in terms of its relation to this thesis, resides in the author's discovery of the main problems he has tried to solve during his research. Some examples include the lack of digital television widgets, the lack of graphics capabilities such as Z-ordering and the alpha channel, and also the integration of video and graphics in the same context.

  Author's overall contribution to the paper: 20%.

- **Publication [P2]** This paper presented a system integration model for running different digital television applications in a prototype set-top box. The integration model includes hardware, middleware (including operating system, Java virtual machine, APIs, and application manager), and applications. The author migrated the applications to the Linux and Kaffe platforms. This article and [P1] are included in this thesis because they constitute the starting point of the author's research.

Author's overall contribution to the paper: 30%.

- **Publication [P3]** This paper was the result of the author's Master's Thesis. The contributions of this paper are the study and implementation of a graphical user interface framework for digital television. This framework was written in Java, it is lightweight, and it fits digital television requirements (e.g., remote control use). It is based on MHP's graphics standard (i.e., HAVi). The author wrote the paper and implemented the FTV package. The results of this paper were widely used in later publications.

  Author's overall contribution to the paper: 90%.

- **Publication [P4]** This article presented the design and implementation of a portable SMIL player. The player was written in Java and can be executed on top of AWT, Swing, and FTV (i.e., an implementation based on HAVi standard) GUI frameworks. Thus, it can be run on various platforms, including PDAs, PCs, and digital television receivers. The author wrote Section 2 and half of sections 1 and 4. In addition, he designed and implemented the FTV GUI framework.

  Author's overall contribution to the paper: 40%.

- **Publication [P5]** In this article, different uses of a SMIL player in the digital television environment were studied. Two case studies were presented: Super Teletext and an enhanced program service. The main benefits of SMIL used in the digital television environment are the inclusion of the time dimension in applications, ease of service development, and the convergence of broadcast and the Internet. The main problem encountered was the technological immaturity of commercial digital television receivers. Some examples include the lack of an effective return channel mechanism (i.e., only analogue modem was available) and of local XML parsers (i.e., the parsers had to be broadcast, which increased the size of the package). For this paper, the author ported the SMIL player presented in [P4], converting it into an Xlet. He wrote sections 1 and 6, half of Section 5, and more than a third of sections 2 and 4.

  Author's overall contribution to the paper: 40%.

- **Publication [P6]** This paper presented the implementation of a DVB-T network called Otadigi, the implementation of two services, and different ways of improving their delivery. The main contribution of the paper was to test previously developed applications, presented in [P2], in commercial MHP-compliant receivers, as well as to study the behaviour of such receivers. The author is a member of Otadigi's technical committee (i.e., he works on its development and makes decisions about the equipment used) and ported the two services presented in the paper. He wrote sections 3 and 5 in their entirety.

Author's overall contribution to the paper: 50%.

- **Publication [P7]** The main contributions of this paper were, first, the definition of a valid XML based language (i.e., the SMIL + XForms profile) to develop interactive multimedia applications. Second, it presented the development of a Component Factory to seamlessly support this profile in a variety of devices (i.e., there is no necessity to implement a unique version for each targeted platform). And, finally, it described the inclusion of this profile in an ongoing digital television prototype platform for digital television reception, called Ubik. The author designed and implemented Ubik and integrated the XML user agent. He wrote sections 1 and 2, Subsection 3.3, and half of Section 5.

  Author's overall contribution to the paper: 50%.

- **Publication [P8]** This paper presented a layered graphics software architecture for multimedia terminals. The architecture takes into account multimedia requirements such as visualisation (e.g., an alpha channel between graphics layers), the temporal dimension provided by the SMIL player, and seamless integration of graphics and video. In addition, the architecture is open, so different configurations (e.g., MHP-compliant receiver or game console) can be obtained. Finally, a prototype implementation based on the model, called Ubik, was described. The author designed and developed Ubik, and he wrote sections 2, 3, 5 (except Subsection 5.1), 6, and 7, and half of Section 4.

  Author's overall contribution to the paper: 70%.

- **Publication [P9]** The contributions of this paper can be divided into those of commercial, standardisation, and theoretical value. First, commercially it proved that the transformation of the digital television receiver into a multimedia platform while keeping its own nature (i.e., watching television as the main application) is feasible. Second, this paper proposed some extensions to the GEM standard while remaining within the MHP framework (e.g., its extension so 3D graphics are supported). Moreover, this paper recommended the support of integrated emerging W3C standards such as XHTML, SMIL, and XForms for digital television receivers' browsers. Finally, at the theoretical level, this paper took a new look at the topic of user interface software tools from the perspective of non-desktop multimedia devices. The author wrote the paper and designed and implemented Ubik.

  Author's overall contribution to the paper: 90%.

In order to provide a clearer understanding of the path followed by the author during his research, Figure 8.1 depicts the relationships between the publications.

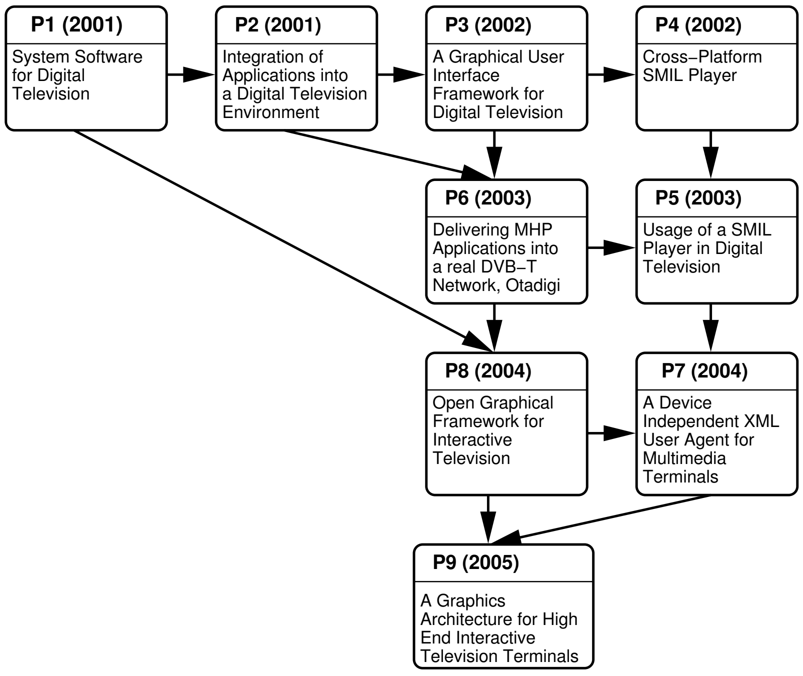

P1 (2001)
System Software for Digital Television
P2 (2001)
Integration of Applications into a Digital Television Environment
P3 (2002)
A Graphical User Interface Framework for Digital Television
P4 (2002)
Cross−Platform SMIL Player
P6 (2003)
Delivering MHP Applications into a real DVB−T Network, Otadigi
P5 (2003)
Usage of a SMIL Player in Digital Television
P8 (2004)
Open Graphical Framework for Interactive Television
P7 (2004)
A Device Independent XML User Agent for Multimedia Terminals
P9 (2005)
A Graphics Architecture for High End Interactive Television Terminals

Figure 8.1: Relation of Previous Publications to the Thesis.

BIBLIOGRAPHY

[1] T. A. Aleem. *A Taxonomy of Multimedia Interactivity*. Doctoral dissertation, The Union Institute, USA, September 1998.

[2] M. Altheim et al. Modularization of XHTML. W3C Recommendation, April 2001.

[3] ARIB. Application execution engine platform for digital broadcasting. ARIB STD-B23 v1.0, June 2003.

[4] ARIB. Data coding and transmission specification for digital broadcasting. ARIB STD-B24 v3.2, June 2003.

[5] ARIB. Receiver for digital broadcasting. ARIB STD-B21 v4.2, October 2003.

[6] ARIB. Coding, transmission and storage specification for broadcasting system based on home servers. ARIB STD-B38 v1.2, December 2004.

[7] ARIB. Video coding, audio coding, and multiplexing specifications. ARIB STD-B32 v1.5, February 2005.

[8] ATSC. DTV application software environment - level 1 (DASE-1). ATSC Doc. A/100, March 2003.

[9] ATSC. ATSC candidate standard: Advanced common application platform (ACAP). ATSC Doc. A/101A, February 2004.

[10] ATSC. ATSC standard: Digital television standard, revision c, including amendment n. 1. ATSC Doc. A/53C, July 2004.

[11] J. Axelsson et al. XHTML 2.0: The extensible hypertext markup language. W3C Working Draft, July 2004.

[12] P. Bar-Haim and S. Wald. *The NDS Guide to Digital Set-top Boxes*. NDS, third edition, 2002.

[13] U. Black. *Mobile and Wireless Networks*. Prentice Hall, Upper Saddle River (NJ), 1996.

[14] D. Blythe and A. Munshi. OpenGL ES common/common lite profile specification. Khronos Group Inc, version 1.1.02, November 2004.

[15] S. Boll. *ZYX, Towards Flexible Multimedia Document Models for Reuse and Adaptation*. Doctoral dissertation, University of Vienna, Austria, August 2001.

[16] B. Bos, H. W. Lie, C. Lilley, and I. Jacobs. Cascading style sheets, level 2, CSS2 specification. W3C Recommendation, May 1998.

[17] E. Boyle et al. The creation of MPEG-4 content and its delivery over DVB infrastructure. In *Proceedings of the first Joint IEI/IEEE Symposium on Telecommunications Systems Research*, Dublin, Ireland, 2001.

[18] T. Bray et al. Extensible markup language (XML) 1.0 (3rd edition). W3C Recommendation, February 2004.

[19] M. H. Butler. Current technologies for device independence. Technical Report HPL-2001-83, Hewlett Packard Laboratories, March 2001.

[20] CableLabs. Opencable application platform specification (OCAP) 2.0 profile. OpenCable OC-SP-OCAP2.0-I01-020419, April 2002.

[21] CableLabs. Opencable application platform specification (OCAP) 1.0 profile. OpenCable OC-SP-OCAP1.0-I14-050119, January 2005.

[22] B. Calder and Courtney J. Java TV API technical overview: the Java TV API whitepaper version 1.0. Sun Microsystems, White Paper, 2000.

[23] CENELEC. Standardization in digital interactive television. Strategy and recommendations for a standardisation policy supporting the effective implementation of framework directive 2002/21/EC and the establishment of required interoperability levels in digital interactive television. CENELEC, April 2003.

[24] CENELEC. Standardization work programme in support of digital interactive television and the effective implementation of article 18 of directive 2002/21/EC. ETSI TR 102 282 v1.1.1, February 2004.

[25] K.-A. Cha and S. Kim. MPEG-4 studio: An object-based authoring system for MPEG-4 contents. *Multimedia Tools and Applications*, 25(1):111–131, January 2005.

[26] M.-L. Champel. MHP 1.1: DVB-HTML part. Thomson, Multimedia Presentation, October 2001.

[27] L.-T. Cheok and A. Eleftheriadis. SMIL vs MPEG-4 BIFS. Department of Electrical Engineering, Columbia University, Technical Report, January 2002.

[28] L. Chiariglione. Impact of MPEG standards on multimedia industry. *Proceedings of the IEEE*, 86(6):1222–12227, June 1998.

[29] K. Chorianpoulos. *Virtual Television Channels: Conceptual Model, User Interface Design and Affective Usability Evaluation*. Doctoral dissertation, Athens University of Economics and Business, Greece, May 2004.

[30] W. S. Ciciora. Inside the set-top box. *IEEE Spectrum*, 33(4):70–75, April 1995.

[31] P. C. Clements. Coming attractions in software architecture. Carnegie Mellon University (CMU), Technical Report, CMU/SEI-96-TR-008, ESC-TR-96-008, January 1996.

[32] C. Concolato and J.-C. Dufourd. Comparison of MPEG-4 BIFS and other multimedia description languages. In *Proceedings of the Workshop and Exhibition on MPEG-4*, San Jose (CA), 2002.

[33] J. Cosmas et al. Customtv with MPEG-4 and MPEG-7. In *IEEE Colloquium on Interactive Television*, volume 9, pages 1–6, 1999.

[34] J. Cosmas et al. Customized television: standards compliant advanced digital television. *IEEE Transactions on Broadcasting*, 48(2):151–158, 2002.

[35] Counterpoint. Digital television - consumers' use and perception. A report on research study. Oftel, August 2001.

[36] A. Daniels. The multimedia home platform: Creating a multimedia world in the STB. CommsDesign online technical resource center, (http://www.commsdesign.com/design_corner/OEG20020918S0011), September 2002.

[37] P. Daras, I. Kompatsiaris, and T. Rapits. An MPEG-4 tool for composing 3D scenes. *IEEE Multimedia*, 11(2):58–71, 2004.

[38] C. Declerck. *Introduction to MHP*. Focal Press, To be published during 2005.

[39] M. A. Dolan. Report on television data applications. National Institute of Standards and Technology (NIST), Report NIST GCR 01-818, July 2001.

[40] J.-L. Droicourt. Integra architecture-anatomy of the interactive set-top box, how it works, and what it means to the consumer. In *Proceedings of the International Broadcasting Convention (IBC), Conference Publication N. 428*, pages 272–276. IBC, September 12-16 1996.

[41] Digital TV Group (DTG). Digital terrestrial television MHEG-5 specification. v1.06, May 2003.

[42] DTI. Digital television for all - a report on usability and accessible design. DTI, September 2003.

[43] M. Dubinko et al. XForms 1.0. W3C Recommendation, 2003.

[44] J. S. Dumas and J. C. Redish. *A Practical Guide to Usability Testing*. Ablex Publishing Corporation, Norwood (NJ), 1993.

[45] S. Dustdar. Multimedia information systems applications - a taxonomy and three case studies. In *Proceedings of the 1997 International Conference on Multimedia Computing and Systems (ICMCS '97)*, pages 654–655. IEEE Computer Society Press, 1997.

[46] DVB. Digital video broadcasting (DVB); specification for conveying ITU-R system b teletext in DVB bitstreams. ETSI EN 300 472, v1.3.1.

[47] DVB. Digital video broadcasting (DVB); framing structure, channel coding and modulation for 11/12 GHz satellite services. ETSI EN 300 421 v1.1.2, August 1997.

[48] DVB. Digital video broadcasting (DVB); interaction channel through Public Switched Telecommunications Network (PSTN) / Integrated Services Digital Networks (ISDN). ETSI EN 300 801, August 1997.

[49] DVB. Digital video broadcasting (DVB); framing structure, channel coding and modulation for cable systems. ETSI EN 300 429 v1.2.1, April 1998.

[50] DVB. Digital video broadcasting (DVB); subtitling systems. ETSI EN 300 743 v1.2.1, October 2002.

[51] DVB. Digital video broadcasting (DVB); multimedia home platform (MHP) specification 1.0.3. ETSI TS 101 812 v1.3.1, June 2003.

[52] DVB. Digital video broadcasting (DVB); multimedia home platform (MHP) specification 1.1.1. ETSI TS 102 812 v1.2.1, June 2003.

[53] DVB. Digital video broadcasting (DVB); framing structure, channel coding and modulation for digital terrestrial television. ETSI EN 300 744 v1.5.1, November 2004.

[54] DVB. Digital video broadcasting (DVB); globally executable MHP (GEM). ETSI TS 102 819 v1.2.1, May 2004.

[55] DVB. Digital video broadcasting (DVB); specification for service information (SI) in DVB systems. ETSI EN 300 468 v1.6.1, November 2004.

[56] DVB. Digital video broadcasting (DVB); transmission system for handheld terminals (DVB-H). Final Draft ETSI EN 302 304 v.1.1.1, June 2004.

[57] DVB. Digital video broadcasting (DVB); implementation guidelines for the use of video and audio coding in broadcasting applications based on the MPEG-2 transport stream. ETSI TS 101 154 v1.7.1, June 2005.

[58] DVB. Digital video broadcasting (DVB); multimedia home platform (MHP) specification 1.1.2. DVB Document A068 Rev. 1, April 2005.

[59] ECMA-262. Ecmascript language specification. European Computer Manufacturers Association (ECMA) (3rd Edition), December 1999.

[60] R. Koenen (ed.). MPEG-4 overview. Document ISO/MPEG N2725, March 1999.

[61] J. Ferraiolo et al. Scalable vector graphics (SVG) 1.1. W3C Recommendation, January 2003.

[62] D. Flanagan. *Java in a Nutshell: a Desktop Quick Reference*. O'Reily & Associates, Reading (MA), 1997.

[63] MPEG-4 Industry Forum. MPEG-4 - the media standard: the advanced multimedia coding. Document m4-out-20027-R3.pdf, November 2002.

[64] B. Fox. Digital TV comes down to earth. *IEEE Spectrum*, 35(10):23–29, October 1998.

[65] B. Fox. Digital TV rollout (US digital terrestrial TV). *IEEE Spectrum*, 38(2):65–67, February 2001.

[66] D. Garlan. Software architecture: a roadmap. In A. Finkestein, editor, *The Future of Software Engineering*. ACM Press, 2000.

[67] D. Garlan and M. Shaw. An introduction to software engineering. In V. Ambriola and G. Tortora, editors, *Advances in Software Engineering and Knowledge Engineering*, volume 2, pages 1–39. World Scientific Publishing Company, Singapore, 2000.

[68] J. H. A. Gelissen. The ITEA project europa, a software platform for digital CE appliances. In *Proceedings of the IEEE International Conference on Multimedia and Expo*, pages 583–586, Tokyo, Japan, 2001. IEEE.

[69] A. Gil et al. Internet-TV convergence in DVB-MHP. In *Proceedings of the International Symposium on Video/Image Processing and Multimedia Communications*, pages 447–451. IEEE, 2002.

[70] A. Gil et al. Surfing the web on TV: the MHP approach. In *Proceedings of the International Conference on Multimedia and Expo*, pages 285–288. IEEE, 2002.

[71] A. Gil et al. The MHP (multimedia home platform) framework for web access through digital TV. In *Proceedings of the International Conference on Web Engineering*, pages 523–524. Springer Verlag, July 2003.

[72] R. L. Glass, V. Ramesh, and I. Vessey. An analysis of research in computing disciplines. *Communications of the ACM*, 47(6):89–94, June 2004.

[73] R. Gordon and S. Talley. *Essential JMF: Java Media Framework*. Prentice Hall, Upper Saddle River (NJ), 1999.

[74] M. Green and R. Jacob. Siggraph '90 workshop report: Software architectures and metaphors for non-wimp user interfaces. *ACM SIGGRAPH Computer Graphics*, 25(3):229–235, July 1991.

[75] F. Halsall. *Data Communications, Computer Networks and Open Systems*. Addison-Wesley, Reading (MA), fourth edition, 1996.

[76] U. Hansman, L. Merk, M. S. Nicklous, and T. Stober. *Pervasive Computing Handbook*. Springer-Verlang, Berlin (Germany), 2001.

[77] R. S. Heller and C. D. Martin. A media taxonomy. *IEEE Multimedia*, 2(4):36–45, Winter 1995.

[78] R. S. Heller, C. D. Martin, N. Haneef, and S. Gievska-krliu. Using a theoretical multimedia taxonomy framework. *ACM Journal of Educational Resources in Computing*, 1(1):article number 6, 2001.

[79] A. L. Hors et al. Document object model (DOM) level 2 core specification - version 1.0. W3C Recommendation, November 2000.

[80] P. Hoschka et al. Synchronized multimedia integration language (SMIL) 1.0. W3C Recommendation, June 1998.

[81] P. Hoschka et al. Synchronized multimedia integration language (SMIL) 2.0. W3C Recommendation, August 2001.

[82] T. R. Hurley. Evolution of the digital set-top box. In *Proceedings of the International Broadcasting Convention (IBC), Conference Publication N. 428*, pages 277–288. IBC, September 12-16 1996.

[83] K. Illgner and J. Cosmas. System concept for interactive broadcasting consumer terminals. In *Proceedings of the International Broadcasting Convention (IBC)*, Amsterdam, The Netherlands, September 2001.

[84] ISO/IEC. Information technology - coding of multimedia and hypermedia information: MHEG object representation, base notation (ASN.1). ISO/IEC 13522-1 (MHEG-1), 1997.

[85] ISO/IEC. Information technology - coding of multimedia and hypermedia information: Support for base-level interactive applications. ISO/IEC 13522-5 (MHEG-5), 1999.

[86] ITU-T. Worldwide common core - application environment for digital interactive television services. ITU-T J.200, March 2001.

[87] ITU-T. Harmonization of procedural content formats for interactive TV applications. ITU-T J.202, May 2003.

[88] ITU-T. Harmonization of declarative content format for interactive TV applications. ITU-T J.201, July 2004.

[89] R. J. K. Jacob. User interfaces. In A. Ralston, E. D. Reilly, and D. Hemmendinger, editors, *Encyclopedia of Computer Sciences*. Grove Dictionaries Inc, fourth edition, 2000.

[90] J. Jones. DVB-MHP / Java data transport mechanisms. In *Proceedings of the 40th International Conference on Tools Pacific*, volume 10, pages 115–121, Sydney, Australia, 2002. Australian Computer Society.

[91] W. T. Kate, P. Deunhouwer, and R. Clout. Timesheets - integrating timing in XML. In *Proceedings of the WWW9 Workshop: Multimedia on the Web*, Amsterdam, The Netherlands, 2000. ACM.

[92] The MHP Knowledge Project Knowledge Database. Analysis of the current situation. The MHP Knowledge Project, IST-507442, Report D1, v1.0, April 2004.

[93] E. A. Lee. Embedded software. In M. Zelkowitz, editor, *Advances in Computers*, volume 56. Academic Press, London, 2002.

[94] T. Lewis. Information appliances: Gadget netopia. *Computer*, 31(1):59–68, 1998.

[95] Z.-N. Li and M. S. Drew. *Fundamentals of Multimedia*. Prentice Hall, Upper Saddle River (NJ), 2004.

[96] H. W. Lie and B. Bos. Cascading style sheets, level 1. W3C Recommendation, December 1999.

[97] T. Lindhom and F. Yellin. *The Java Virtual Machine Specifications*. Addison Wesley, Reading (MA), 1997.

[98] G. Lutteke. The DVB multimedia home platform. DVB Project, February 2001.

[99] G. Lutteke. Multimedia home platform - concept and impact. MHP Cable Workshop, January 30th 2002.

[100] M. Magnor. 3D-TV - the future of visual entertainment. In *Proceedings of the Workshop on Multimedia Databases and Image Communication*, pages 1–8, Salerno, Italy, 2004.

[101] C. Malerczyk, K. Klein, and T. Weibeisek. 3D reconstruction of sports events for digital TV. In *Proceedings of the eleventh International Conference in Central Europe on Computer Graphics, Visualization, and Computer Vision*, Plzen, Czech Republic, 2003. WSCG.

[102] C. Marrin, R. Myers, J. Kent, and P. Broadwell. Steerable media: interactive television via video synthesis. In *Proceedings of the sixth international conference on 3D Web Technology: Virtual Reality Modeling Language Symposium*, pages 7–14, Paderbon, Germany, 2001. ACM.

[103] H. Matsumura and H. Katoh. Digital broadcasting systems. *Broadcast Technology*, 18:12–17, Spring 2004.

[104] S. McCarron et al. XML events. W3C Recommendation, October 2003.

[105] N. Medvidovic and M. Mikic-Rakic. Programming-in-the-many: A software paradigm for the 21st century. In *Proceedings of the Workshop on New Visions for Software Design and Productivity: Research and Applications*, 2001.

[106] N. Medvidovic, M. Mikic-Rakic, N. Mehta, and S. Malek. Software architectural support for handheld computing. *IEEE Computer*, 36(9):66–73, September 2003.

[107] T. Meyer-Boudnik and W. Effelsberg. MHEG explained. *IEEE Multimedia*, 2(1):26–38, Spring 1995.

[108] MHP. MHP - multimedia home platform. White Paper, April 2003.

[109] MHP. MHP platform harmonisation. White Paper, April 2004.

[110] M. Mikic-Rakic and N. Medvidovic. Architectural-level support for software component deployment in resource constrained environments. In *Proceedings of the IFIP/ACM Working Conference on Component Deployment*, pages 31–50, 2002.

[111] M. Milenkovic. Delivering interactive services via a digital TV infrastructure. *IEEE Multimedia*, 5(4):34–43, October-December.

[112] S. Morris and A. Smith-Chaigneau. *Interactive TV Standards: a guide to MHP, OCAP, and JavaTV.* Focal Press, To be published during 2005.

[113] T. Murakami. Overview: the present and future of digital TV broadcasting. *Mitsubishi Electric Advance*, 85.

[114] B. Myers. A taxonomy of window manager user interfaces. *IEEE Computer Graphics and Applications*, 8(5):65–84, September 1988.

[115] B. Myers. Graphical user interface programming. In Allen B. Tucker, editor, *CRC Handbook of Computer Science and Engineering*, chapter 48. CRC Press, Inc., Boca Raton (FL), second edition, 2004.

[116] B. Myers, S. E. Hudson, and R. Pausch. Past, present, and future of user interface software tools. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 7(1):3–28, march 2000.

[117] N. Negroponte. *Being Digital.* Random House Inc., New York (NY), 1996.

[118] Netscape. Client-side javascript reference, version 1.3. Netscape Communication Corporation, 1999.

[119] Y. Neuvo and J. Yrjanainen. Wireless meets multimedia - new products and services. In *Proceedings of the International Conference on Image Processing*, volume 1, pages I–1–I–4. IEEE, 2002.

[120] J. C. Newell. The DVB MHP internet access profile. British Broadcasting Corporation (BBC), R&D White Paper, January 2002.

[121] J. C. Newell. An introduction to MHP 1.0 and MHP 1.1. British Broadcasting Corporation (BBC), R&D White Paper, January 2002.

[122] J. Nielesen. *Usability Engineering*. Academic Press, Boston (MA), 1993.

[123] Y. Nishida. Digital multimedia broadcasting expert group progress report. Expert Group Progress Report to ASTAP-8, August 2004.

[124] D. A. Norman. *The Design of Everyday Things*. Doubleday Currency, New York (NY), 1990.

[125] D. A. Norman. *The Invisible Computer*. MIT Press, Cambridge (MA), 1998.

[126] D. A. Norman and S. W. Draper. *User Centered Systems Design: New Perspectives on Human-Computer Interaction*. Lawrence Erlbaum, Hillsdale (NY), 1986.

[127] G. O'Driscoll. *The Essential Guide to Digital Set-top boxes and Interactive TV*. Prentice Hall, Upper Saddle River (NJ), 2000.

[128] D. R. Olsen. *Developing User Interfaces*. Morgan Kaufmann, San Francisco (CA), 1998.

[129] D. R. Olsen. Interacting in chaos. *Interactions*, 6(5):42–54, 1999.

[130] S. Pekowsky and R. Jaeger. The set-top box as multi-media terminal. *Transactions on Consumer Electronics*, 44(3):833–840, August 1998.

[131] S. Pemberton et al. XHTML basic. W3C Recommendation, December 2000.

[132] S. Pemberton et al. XHTML 1.1: Module-based XHTML. W3C Recommendation, May 2001.

[133] S. Pemberton et al. XHTML 1.0: The extensible hypertext markup language (2nd edition). W3C Recommendation, August 2002.

[134] C. Peng. *Digital Television Applications*. Doctoral dissertation, Helsinki University of Technology, Finland, November 2002.

[135] F. Pereira and T. Ebrahimi. *The MPEG-4 Book*. Prentice Hall, Upper Saddle River (NJ), 2002.

[136] F. Pereira and R. Koenen. MPEG-4 - opening new frontiers to broadcast services. *EBU Technical Review*, pages 28–35, Spring 1999.

[137] P. Perrot. DVB-HTML - an optional declarative language within MHP 1.1. *EBU Technical Review*, September 2001.

[138] C. Phanouriou. *UIML: A Device-Independent User Interface Markup Language*. Doctoral dissertation, Virginia Polytechnic Institute and State University, USA, September 2000.

[139] J. Piesing. The DVB multimedia home platform - MHP. In *Proceedings of the IEEE Colloquium on Interactive Television*, volume 2, pages 1–6. IEEE, 1999.

[140] K. Pihkala. *Extensions to the SMIL Language*. Doctoral dissertation, Helsinki University of Technology, Finland, November 2003.

[141] M. Pohja, M. Honkala, and P. Vuorimaa. An XHTML 2.0 implementation. In *Proceedings of the Fourth International Conference on Web Engineering, ICWE2004*, pages 402–415. Springer Verlag, 2004.

[142] R. Pulles and P. Sasno. A set top box combining MHP and MPEG-4 interactivity. In *Proceedings of the second European Union Symposium on Ambient Intelligence*, pages 31–34, Eindhoven, The Netherlands, 2004.

[143] H. Purchase. Defining multimedia. *IEEE Multimedia*, 5(1):8–15, January-March 1998.

[144] R. Rafey et al. Enabling custom enhancements in digital sports broadcasts. In *Proceedings of the sixth internation conference on 3D Web Technology: Virtual Reality Modeling Language Symposium*, pages 101–107. ACM, 2001.

[145] D. Ragget et al. HTML 4.0.1 specification. W3C Recommendation, December 1999.

[146] V. Ramesh, R. L. Glass, and I. Vessey. Research in computer science: an empirical study. *The Journal of Systems and Software*, 70(1-2):165–176, February 2004.

[147] SPB. Rao, BP. Dhanakoti, and P. Vuorimaa. Bluetooth enabled digitalTV set-top box. In *Proceedings of the International Signal Processing Expo & Conference (ISPC2004)*, Santa Clara (CA), September 27-30 2004.

[148] U. Reimers. Digital television broadcasting. *IEEE Communications Magazine*, 36(1):104–110, February 1998.

[149] M. C. Revett and G. J. South. Consumer devices for ecommerce access. *BT Technology Journal*, 17(3):112–123, 1999.

[150] M. Robin and M. Poulin. *Digital Television Fundamentals: Design and Installation of Video and Audio Systems*. McGraw-Hill, New York (NY), 1998.

[151] C. P. Sandbank. Digital television in the convergent environment. *IEEE Computer Graphics and Applications*, 21(1):32–36, January-February 2001.

[152] E. M. Schwalb. *ITV Handbook: Technologies and Standards*. Prentice Hall, Upper Saddle River (NJ), 2003.

[153] M. Shaw. The coming-of-age of software architecture research. In *Proceedings of the International Conference on Software Engineering*, pages 656–664. ACM, 2001.

[154] J. Signes, Y. Fisher, and A. Eleftheriadis. MPEG-4's binary format for scene description. *Signal Processing: Image Communication Journal*, 15(4-5):321–345, January 2000.

[155] SMPTE. Declarative data essence – transitional. SMPTE Technology Committee D27 on Data Essence Technology, Proposed SMPTE 397M, April 2003.

[156] W. Stallings. *Data and Computer Communications*. Prentice Hall, Upper Saddle River (NJ), sixth edition, 2000.

[157] A. S. Tanenbaum. *Computer Networks*. Prentice Hall, Upper Saddle River (NJ), fourth edition, 2003.

[158] R. S. Tannenbaum. *Theoretical Foundations of Multimedia*. Computer Science Press, New York (NY), 1998, Fourth printing 2001.

[159] S. M. Tran, M. Preda, F. J. Preteux, and K. Fazekas. Exploring MPEG-4 BIFS features for creating multimedia games. In *Proceedings of the IEEE International Conference on Multimedia and Expo*, pages 429–432, Baltimore, MD, 2003. IEEE.

[160] I. Vessey, V. Ramesh, and R. L. Glass. A unified classification system for research in the computing disciplines. *Information and Software Technology*, 47(4):245–255, March 2005.

[161] R. Want, G. Borriello, T. Pering, and K. I. Farkas. Disappearing hardware. *IEEE Pervasive Computing*, 1(1):36–47, 2002.

[162] M. Weiser. The computer of the 21st century. *Scientific America*, 265(3):94–104, 1991.

[163] M. Weiser. Some computer science numbers in ubiquitous computing. *Communications of the ACM (CACM)*, 36(7):75–84, 1993.

[164] M. Weiser. The world is not a desktop. *Interactions*, 1(1):7–8, 1994.

[165] J. G. Wijnstra. Supporting diversity with component frameworks as architectural elements. In *Proceedings of the International Conference on Software Engineering*, pages 50–59. ACM, 2000.

[166] M. Williams. *A Taxonomy of Media Usage in Multimedia*. Doctoral dissertation, Nova Southeastern University, USA, May 2003.

[167] T. Worthington. Internet-TV convergence with the multimedia home platform. Communications Research Forum, September 2001.

APPENDIX A: INSTALLING UBIK

Ubik is a prototype implementation of a configurable digital television receiver. I have implemented it as a proof of concept of the contributions included in this thesis. Ubik can be downloaded, as open-source code, from http://sourceforge.net/projects/ubik and http://sourceforge.net/projects/kaffesdl.

This appendix includes information about how to install Ubik. Nevertheless, please read the README, INSTALL, and DEPENDENCIES files before installation. In case of any problems or suggestions, please contact me at pcesar@tml.hut.fi. Good luck!

Ubik is configurable and open, meaning different models of the receiver can be provided according to the user's needs. Some available configurations include Basic Broadcast (watching TV only), Enhanced Broadcast (enjoying Java applications), Basic Interactive (accessing XML based information services), Internet Access Interactive (running complex XML based applications), and High-End Interactive (3D graphics and streamed video support). Its openness derives from its reliance on a number of open-source projects, such as Gentoo[1], OpenGL[2], DirectFB[3], SDL[4], Kaffe[5], Java OpenGL[6], and X-smiles[7].

Installation procedure

There is plenty of information about Linux available on the Web, but the best guide I can think of is Gentoo. In order to enable DVB-T reception in Linux, please refer to Appendix B. DirectFB is a C-based package intended to effectively render graphics through the framebuffer, so X-Window is not needed. Ubik utilises a modified version of DirectFB 0.9-21 (modifications are included in the distribution). SDL is a cross-platform abstraction layer of the hardware graphics, and thus runs on top of DirectFB. Ubik uses a modified version of SDL 2.6 (modifications are included in the distribution). Kaffe's AWT package has been modified, so it paints and handles events using SDL primitives. In addition to the Kaffe 1.1.3 virtual machine, Ubik uses Jikes 1.14 as compiler. As a summary, Ubik's graphics system software relies on SDL surfaces as a shared graphics context for applications (video, 2D, and 3D graphics). Once the surface is composite, it is rendered using DirectFB, which has access to the graphics drivers.

Finally, a number of services can be enjoyed using Ubik. Some examples include:

- Watching a broadcasted A/V stream

[1]http://www.gentoo.org
[2]http://www.opengl.org
[3]http://www.directfb.org
[4]http://www.libsdl.org
[5]http://www.kaffe.org
[6]http://www.jogl.org
[7]http://www.xsmiles.org

- Playing SDL games such as Tuxracer or Doom
- Using DVB-J applications such as Super Teletext or Navigator
- Watching DVB-3DJ applications such as commercials
- Browsing simple XML based information services using XHTML 1.0 and CSS
- Interacting with complex XML based applications using XForms + SMIL/Timesheets

APPENDIX B: HOW TO BUILD A LINUX-BASED DTV RECEIVER

Note: this document is based on the following Web pages:

- http://www.stanford.edu/ bescoto/linuxtv/
- http://software.newsforge.com/software/04/03/08/165203.shtml/
- http://nrg.joroinen.fi/dvb-minihowto/index.html

and a number of forums including MPlayer, dvblinux, linuxstb, etc., but - as always - the continuity of Web pages is not guaranteed.

Introduction

This appendix tries to explain how you can create a digital television receiver. The only requirements are the following:

- Linux operating system (kernel 2.6)
- DBV-T reception card (e.g., Nova-t PCI)

The first step is to install Linux. A good alternative is Gentoo[8], which includes high-quality reference documentation, so you can learn the operating system at the same time.

Linux Kernel: DVB, Video for Linux, I2C, and ALSA Sound

There are two main options today. One is kernel 2.6, which includes the modules for DVB-T reception. The other is kernel 2.4, in which case you need to download the modules from www.linuxtv.org. This appendix assumes that you want to install the latest version (kernel 2.6), and you might also need certain patches for it (download them from www.linuxtv.org and run "*installpatch*"). The actual options you need for Nova-t PCI are the following:

- Device Drivers
  - Video for Linux
    - Philips SAA7134 Support
    - Philips Semiconductor 'dcp7146 Demonstration Board
  - I2C
    - I2C Device Interface
    - I2C Hardware Bus Support
      - VIA 82C5863
      - VIA 82C596/82C686/823x
    - Hardware Sensors Chip Support

[8] http://www.gentoo.org

            · VIA686A

  - Sound

      * Advanced Linux Sound Architecture
      * Open Sound System

  - Digital Video Broadcasting Devices

      * DVB For Linux
      * DVB Core Support
      * Front-ends with External TDA1004x Demodulators (QFDM)
      * AV7110 Cards
      * Budget Cards
      * Budget Cards with Onboard CI Connector
      * Budget Cards with Analog Video Input
      * AV7110 Cards with Budget Path

Then, type the following commands in the given order:

1. "*modprobe tad1004x*"

2. "*modprobe budget*"

3. "*modprobe budget_ci*"

4. "*modprobe budget_patch*"

The most informative feedback is provided by typing "*dmesg*". It should give you the following information:

```
Linux video capture interface: v1.00
saa7146: register extension 'dvb'.
eepro100.c:v1.09j-t 9/29/99 Donald Becker
http://www.scyld.com/network/eepro100.html
eepro100.c: $Revision: 1.1 $ 2000/11/17
Modified
by Andrey V. Savochkin <saw@saw.sw.com.sg>
and others
eth0: 0000:00:0b.0, 00:02:B3:9B:D5:71, IRQ 10.
  Board assembly 751767-004, Physical
  connectors present: RJ45
  Primary interface chip i82555 PHY #1.
    Secondary interface chip i82555.
  General self-test: passed.
  Serial sub-system self-test: passed.
  Internal registers self-test: passed.
  ROM checksum self-test: passed (0x3258698e).
saa7146: register extension 'budget dvb'.
saa7146: register extension 'budget_ci dvb'.
saa7146: found saa7146 @ mem f88b4000
(revision 1, irq 9) (0x13c2,0x1011).
DVB: registering new adapter
(TT-Budget/WinTV-NOVA-T  PCI).
tda1004x: Detected Philips TDA10045H.
tda1004x: Detected Philips TDM1316L tuner.
DVB: registering frontend 0:0
(Philips TDA10045H)...
TT-Budget/WinTV-NOVA-T PCI adapter 0 has
MAC addr = 00:d0:5c:23:0f:bf
saa7146: register extension 'budget_patch dvb'.
```

3D Graphics

This is a little tricky if you are not planning to use X-Window. Even though it seems odd, the agpart module (for 3D graphics) should not be loaded. Then, enable the following modules for Matrox Framebuffer (M option):

- Device Drivers
  - Character Devices
  - Intel/AMD/VIA HW Random Number Generator Support
  - /dev/agpgart (AGP support)
  - VIA Chipet Support
  - Matrox g200/g400

Next, type:

1. ”*rmmod agpgart*”
2. ”*modprobe via_agp*”

3. ”*modprobe matroxfb_base*”

The resulting dmesg should be:

```
Linux agpgart interface v0.100 (c) Dave Jones
agpgart: Detected VIA Twister-K/KT133x/KM133
chipset
agpgart: Maximum main memory to use for agp
memory: 816M
agpgart: AGP aperture is 64M @ 0xe4000000
matroxfb: Matrox G550 detected
matroxfb: MTRR's turned on
matroxfb: 640x480x8bpp (virtual: 640x65536)
matroxfb: framebuffer at 0xE2000000, mapped
to 0xf8ad2000, size 33554432
fb0: MATROX frame buffer device
fb0: initializing hardware
i2c_adapter i2c-2: Registered as minor 2
i2c_adapter i2c-2: registered as adapter #2
i2c_adapter i2c-3: Registered as minor 3
i2c_adapter i2c-3: registered as adapter #3
i2c_adapter i2c-4: Registered as minor 4
i2c_adapter i2c-4: registered as adapter #4
[drm] Initialized mga 3.2.0 20040319 on minor
0: Matrox G550
(AGP)agpgart: Found an AGP 2.0 compliant
device at 0000:00:00.0.
agpgart: Putting AGP V2 device at 0000:00:00.0
into 4x mode
agpgart: Putting AGP V2 device at 0000:01:00.0
into 4x mode
```

Tuning and Zapping

In order to watch your installed embedded Linux digital TV receiver, the first thing is to try tuning and zapping the channels. In order to try your system and acquire essential information about the broadcast, download dvbtune from linuxstb[9]. Install it (”*./configure*”, ”*make*”, ”*make install*”) and run:

1. ”*dvbtune -f frequency -i >tables.xml*”

It will write into the tables.xml file the data information for a given frequency. For example, in Finland you can find frequencies listed by city on Digita's Web page[10]. In the sample case that follows, the frequency was set to 562000000, resulting in access to the multiplexer called A (YLE). The following output was obtained:

[9]http://www.linuxstb.org

[10]http://www.digita.fi/english/digita_alasivu.asp?path=1841;2080;2130;2132

```
<service id="17" ca="0">
<description tag="0x48" type="1"
provider_name="YLE" service_name="YLE TV1" />

<stream type="2" pid="512">
...
</stream>

<stream type="4" pid="650">
...
</stream>

<stream type="11" pid="3110">
...
</stream>

...

<stream type="11" pid="3116">
...
</stream>

</service>
```

Even though tables.xml is quite an ugly file, it is not difficult to interpret if you understand some basic tags:

- <service>: a program (a channel in the analogue world like BBC)

- <stream>: a MPEG-2 data stream

  - type=2: video
  - type=4: audio
  - type=6: subtitles or teletext
  - type=11: application

Now you can understand the sample tables.xml file: the program (channel) called ”YLE TV1” has the program number 17, is provided by YLE, and includes video (pid=512), audio (pid=650), and some applications (e.g., pid=3110).

Once you have identified all the channels and related information, you can create your own channels.conf. The channels.conf file describes all the transmission information of a given channel, including its name, frequency, bandwidth, error correction, and audio and video PIDs. An extract of the channels.conf for the Helsinki region looks like this:

```
subtv:658000000:INVERSION_OFF:BANDWIDTH_8_MHZ:
FEC_2_3:FEC_NONE:QAM_64:TRANSMISSION_MODE_8K:
GUARD_INTERVAL_1_8:HIERARCHY_NONE:353:609

mtv3:658000000:INVERSION_OFF:BANDWIDTH_8_MHZ:
FEC_2_3:FEC_NONE:QAM_64:TRANSMISSION_MODE_8K:
GUARD_INTERVAL_1_8:HIERARCHY_NONE:304:560

yle1:562000000:INVERSION_OFF:BANDWIDTH_8_MHZ:
FEC_2_3:FEC_NONE:QAM_64:TRANSMISSION_MODE_8K:
GUARD_INTERVAL_1_8:HIERARCHY_NONE:512:650

otadigi:746000000:INVERSION_OFF:BANDWIDTH_8_MHZ:
FEC_2_3:FEC_NONE:QAM_16:TRANSMISSION_MODE_8K:
GUARD_INTERVAL_1_4:HIERARCHY_NONE:101:102
```

Next, after creating the channels.conf, you can install the tzap application provided by the linuxtv[11] project. This program allows you to zap any channel. For example, you can tune to ”YLE TV1” by typing ”*tzap -c channels.conf yle1*”. This is how tzap works:

1. ”*tzap -c channels.conf channel-name*”

Watching Video and Audio
There are at least two alternative ways to watch the A/V stream:

- You can install a normal video player, such as MPlayer, and pipe the A/V stream to it by typing: ”*dvbstream -ps -o video_pid audio_pid — mplayer -cache 1024 -*”
- Or you can use DirectFB and install the libsoftmpeg package. In this case, first tzap (”*tzap -r yle1*”) to the desired channel and then type ”*demo/dfb_ts /dev/dvb/adapter0/dvr video_pid audio_pid*”

Downloading Applications
In order to download applications from the object carousel, there is an application called dvbdata. You can use it as follows:

1. ”*dvbdata -f frequency -pnr 1 -n program-name* ”

[11] http://www.linuxtv.org