



UNIVERSIDADE TÉCNICA DE LISBOA
INSTITUTO SUPERIOR TÉCNICO

Universal Access to Multimedia Content Based on the MPEG-7 Standard

João Miguel da Costa Magalhães
(Licenciado)

**Dissertação para obtenção do Grau de Mestre em
Engenharia Electrotécnica e de Computadores**

Orientador: Doutor Fernando Manuel Bernardo Pereira

Júri:

Presidente: Doutor Mário Serafim dos Santos Nunes

Vogais: Doutor Fernando Manuel Bernardo Pereira

Doutor Carlos Guilherme Ferreira de Morais Pires

Doutor António José Castelo Branco Rodrigues

Junho 2002

Resumo

Os últimos desenvolvimentos em áreas como as Telecomunicações, a Microelectrónica e as Tecnologias de Informação fazem disparar a quantidade de informação disponível ao utilizador. Um autor pode criar quase instantaneamente conteúdo multimédia utilizando um *scanner* ou uma máquina fotográfica digital. Por outro lado, estes avanços dão origem a terminais móveis com cada vez mais capacidades inovadoras e a acessos com maior largura de banda à Internet, nomeadamente com tecnologia UMTS. Muitas das vezes, o autor pretende partilhar os seus conteúdos, no entanto, a variedade de terminais e redes de acesso serão um problema se o autor pretender que os seus conteúdos sejam acedidos por qualquer pessoa em qualquer local e com qualquer terminal.

Perante este cenário, é compreensível a crescente disparidade entre a qualidade dos conteúdos multimédia e o acesso a estes conteúdos muito ricos através de terminais móveis. “Universal Multimedia Access” é o nome do problema que considera a entrega de qualquer conteúdo para qualquer terminal.

O objectivo da presente tese é o estudo e desenvolvimento de um sistema capaz de efectuar a manipulação e gestão de conteúdos multimédia por forma a disponibilizar a mesma informação a diferentes terminais e redes de acesso.

Palavras-chave

Universal Multimedia Access, UMA, content description, MPEG-7, user environment, MPEG-21, content customization, user context.

Abstract

The exploding variety of multimedia information is nowadays a reality since everyone has a camera, a scanner or another device that almost instantly generates multimedia content. Most often the content author wishes to share its masterpiece with everyone, but the variety of terminals and networks may be a problem if he wants everyone to see his work with the best possible quality. This problem will increasingly grow as industry releases more enabling technologies. Late advances in Telecommunications, Information Technologies and Microelectronic, are creating a set of technologies that will enable the development of an all-new set of services with completely different requirements, terminals with completely different capabilities and networks through which terminals access services in very different conditions.

In the described scenario it is quite perceivable the growing disparity between the content resources needed to consume the content and the terminal capabilities. “Universal Multimedia Access” is the name of the problem that concerns the delivery of any content to any terminal.

This thesis objective is the study and development of a Universal Multimedia Access system capable of managing and customizing content so that any information can be delivered to different terminals and networks.

Keywords:

Universal Multimedia Access, UMA, content description, MPEG-7, user environment, MPEG-21, content customization, user context.

Agradecimentos

Em primeiro lugar quero agradecer ao Prof. Fernando Pereira pela sua dedicação, e acompanhamento extremo ao longo do mestrado, estando sempre disponível para ajudar e aconselhar nas alturas mais conturbadas que povoaram o meu mestrado.

A todos os meus colegas e ex-colegas do Grupo de Imagem do Instituto de Telecomunicações pela sua camaradagem e apoio. Ao meu duplo colega João Ascenso pelas conversas de “caserna” que ajudaram a enfrentar as longas noites de trabalho na tese.

Aos meus pais e irmão, que sempre me apoiaram na realização do mestrado. Um obrigado especial ao meu pai e avô materno por intencionalmente me terem incutido o gosto pelo engenho.

À Anabela por me ter apoiado e ter dado a paz de espírito necessária para a realização da tese.

A todos os meus amigos que estiveram ao meu lado e me apoiaram ao longo do mestrado: Vasco Pereira, Jorge Rosa, Jorge Almeida, Rui Coelho, Pedro Carvalho.

Por fim, agradeço à Siemens, SA, por me ter proporcionado as condições para realizar este trabalho, em particular à Eng^a Leonor Almeida por decidir avançar com desenvolvimento em Portugal na área da presente tese e ao Eng^o Agostinho David por ter albergado no seu departamento o projecto UMA. Agradeço igualmente a todos aqueles que deram contributos construtivos para o projecto UMA, em especial aos meus colegas José Iria, Nuno Abreu e Oscar Trigueiros.

Contents

1	Introduction	1
1.1	Key Technologies	4
1.1.1	Multimedia Content Description	5
1.1.2	User Environment Description	6
1.2	Multimedia Content Customization	7
1.2.1	Content Customization by Selection	9
1.2.2	Content Customization by Adaptation	10
1.3	Objectives and Structure of the Thesis	11
2	UMA Applications and Requirements	15
2.1	Applications	16
2.1.1	Customization Proxy	16
2.1.2	Universal Digital Libraries	17
2.1.3	Universal Mailbox	18
2.1.4	Client-Driven Multimedia Browsing	19
2.2	Relevant UMA-Related Work	20
2.2.1	IBM UMA-Related Experiments	20
2.2.2	Hewlett-Packard UMA-Related Experiments	23
2.2.3	IBM versus HP Content Customization Solutions	27
2.3	Requirements	28
3	Multimedia Content Description	31
3.1	Describing Multimedia Content	31
3.2	MPEG-7 Multimedia Content Description Interface	32
3.2.1	Context and Objectives	32
3.2.2	Scope	33
3.2.3	Applications	36
3.2.4	MPEG-7 Standard Organization	36
3.3	MPEG-7 and UMA applications	45
3.3.1	Media Description Tools	46
3.3.2	Content Structure Description Tools	50
3.3.3	Content Navigation and Access	55
3.4	Summary	61

4	User Environment Description	63
4.1	User Environment Dimensions	63
4.1.1	Terminal	64
4.1.2	Access Network	65
4.1.3	Natural Environment	65
4.1.4	User Preferences	65
4.2	The CC/PP (Composite Capability/Preference Profiles) Framework Solution	66
4.2.1	Profile Structure	67
4.2.2	Attribute Vocabularies	70
4.2.3	Client Profiles	70
4.2.4	Proxy Support	71
4.3	Wireless Application Protocol (WAP) 2.0	73
4.3.1	Architecture	74
4.3.2	Wireless Application Environment (WAE)	75
4.3.3	User Agent Profile	75
4.3.4	CC/PP Technology with WAP	77
4.4	MPEG-21 Multimedia Framework: Digital Item Adaptation	78
4.5	A DIUED Proposal for MPEG-21 DIA	82
4.5.1	Schema tools	84
4.5.2	ServiceProviderType	85
4.5.3	ConsumerType	86
4.6	Summary	96
5	UMA System Design	97
5.1	UMA System Architecture	97
5.1.1	UMA Platform at the Content Server	99
5.1.2	UMA Platform at a Proxy Server	100
5.1.3	UMA Platform at the client	102
5.2	UMA Platform Design	104
5.2.1	Architecture Design	105
5.2.2	Data Structures	109
5.2.3	Event Scenarios	113
5.2.4	Messages	115
5.3	UMA Platform Modules Design	115
5.3.1	GUI	116
5.3.2	Network Interface Manager (NIM)	116
5.3.3	Pipeline Manager/Monitor (PM/M)	116
5.3.4	User Request Processor (URP)	118
5.3.5	Multimedia Content Description Builder (MMCDBuilder)	118
5.3.6	Digital Item User Environment Description Builder (DIUEDBuilder)	118
5.3.7	Content Action Decision (CAD)	119
5.3.8	Content Customization (CC)	119
5.4	Summary	119

6	Content Customization Processing	121
6.1	Content Action Decision Algorithms	121
6.2	Content Customization Algorithms	123
6.3	Image Customization Module	127
6.3.1	Color Temperature Adaptation	129
6.4	Video Customization Module	142
6.5	Summary	143
7	User Interfaces and Test Scenarios	145
7.1	User Interfaces	145
7.1.1	UMA Browser	145
7.1.2	UMA Platform	147
7.1.3	MPEG-7 Description Tool	150
7.2	UMA Test Bed	151
7.2.1	Content Server Test Bed	151
7.2.2	Proxy Server Test Bed	152
7.3	UMA Test Scenarios	153
7.3.1	Multimedia Content and MPEG-7 Descriptions	154
7.3.2	DIUED Scenarios	155
7.4	UMA Results	156
7.4.1	Pocket PC 1 Scenario	156
7.4.2	Pocket PC 2 Scenario	160
7.4.3	WAP Terminal 1 Scenario	161
7.4.4	WAP Terminal 2 Scenario	162
7.5	Summary	165
8	Conclusions and Future Work	167
8.1	Conclusions	167
8.2	Future Work	169
A	References	171
A.1	IETF references	172
A.2	W3C references: CC/PP, RDF and XML Schema	173
A.3	WAP references	173
A.4	3GPP references	174
A.5	Ericsson articles related to Mobile Multimedia	174
A.6	IBM articles related to Universal Multimedia Access	175
A.7	MPEG-7 documents	176
A.8	MPEG-21 documents	178
A.9	Color Temperature	179

List of Figures

Figure 1.1 – Different terminals accessing rich multimedia content through different networks.	2
Figure 1.2 – Conceptual diagram of the UMA Engine.	3
Figure 1.3 – Domains between the content server and the user.	4
Figure 1.4 – Data types composing multimedia content.	5
Figure 1.5 – User environment dimensions.	6
Figure 1.6 – Multimedia content customization example.	8
Figure 1.7 – Enhancing the user experience with adequate content customization.	8
Figure 1.8 – Content customization by selection.	9
Figure 1.9 – Content customization by adaptation.	10
Figure 2.1 – High-level block diagram of the UMA Engine.	15
Figure 2.2 – Customization proxy scenario.	16
Figure 2.3 – Digital libraries can store several content variations.	18
Figure 2.4 – Universal Mailbox scenario.	19
Figure 2.5 – Client-driven multimedia browsing scenario.	20
Figure 2.6 – IBM content adaptation system architecture [1].	21
Figure 2.7 – IBM InfoPyramid for content customization [1].	22
Figure 2.8 – IBM multimedia news system working for different terminals [1].	23
Figure 2.9 – HP adaptive content delivery system [2].	24
Figure 2.10 – HP decision engine architecture [2].	25
Figure 2.11 – Adaptive content delivery to a HP 620 palmtop [2].	26
Figure 3.1 – Sequence of the steps involved in the creation of content descriptions.	33
Figure 3.2 – Scope of MPEG-7.	34
Figure 3.3 – Graphical view of the relation between the different MPEG-7 description elements.	35
Figure 3.4 – Abstract representation of applications using MPEG-7.	35
Figure 3.5 – MPEG-7 terminal architecture.	37
Figure 3.6 – Decomposition of a content description tree into two fragments [86].	38
Figure 3.7 – MPEG-7 Normative Interfaces [86].	38
Figure 3.8 – "Local" and "integrated" coordinate system [88].	40
Figure 3.9 – Real data and interpolation functions [88].	40
Figure 3.10 – Five types of edges for the edge histogram descriptor [88].	41
Figure 3.11 – Examples of various shapes, [88].	42
Figure 3.12 – A 2D visual object (region) and its corresponding shape, [88].	42
Figure 3.13 – (a) Camera track, boom, and dolly motion modes; (b) Camera pan, tilt and roll motion modes.	42
Figure 3.14 – Overview of the MPEG-7 MDS tools [90].	44
Figure 3.15 – Media description tools.	47
Figure 3.16 – Example of image transcoding using Importance Hint information to preserve the fidelity of the face regions.	48
Figure 3.17 – MediaInformationDS elements.	49
Figure 3.18 – Overview of the MPEG-7 tools for describing segments of audiovisual content.	50
Figure 3.19 – Segment DS elements.	51
Figure 3.20 – Mosaic for the "Parliament" sequence constructed from 12 frames [88].	51

Figure 3.21 – Preview of the MPEG-7 tools for describing relations among segments.	53
Figure 3.22 – (a) Example of video segments and regions. (b) Example of a segment relationship graph.	54
Figure 3.23 – Example of HierarchicalSummaryDS containing a summary.	56
Figure 3.24 – Structure of the Hierarchical Summary and Sequential Summary DSs.	57
Figure 3.25 – Example with the SequentialSummaryDS.	57
Figure 3.26 – Space and Frequency Graph decomposes images and audio signals in space and frequency.	58
Figure 3.27 – Variations description tools.	60
Figure 3.28 – Illustration of key concepts in VariationDS.	61
Figure 4.1 – User Environment Description Interface.	64
Figure 4.2 – CC/PP is a RDF application, which in turn is an XML application.	66
Figure 4.3 – Hierarchical organization of a CC/PP profile [40].	67
Figure 4.4 – Example of a CC/PP device profile [40].	68
Figure 4.5 – Example of defaults [40].	68
Figure 4.6 – RDF representation of the Profile in Figure 4.4 [40].	69
Figure 4.7 – Example of a request profile chain [40].	72
Figure 4.8 – Example of proxy behavior description [40].	72
Figure 4.9 – WAP functional model [48].	74
Figure 4.10 – WAP protocol stack [48].	74
Figure 4.11 – Adapting content to a certain user profile.	77
Figure 4.12 – Adapting content in a WAP context.	78
Figure 4.13 – User’s role in the MPEG-21 framework [117].	79
Figure 4.14 – Example of MPEG-21 scenario with several users [117].	80
Figure 4.15 – Proposed DIUED schema (only the consumer dimension is expanded).	84
Figure 5.1 – High-level block diagram of the UMA Engine.	97
Figure 5.2 – Main elements of a UMA System.	98
Figure 5.3 – UMA Platform at the content server.	100
Figure 5.4 – UMA Platform at the proxy server.	101
Figure 5.5 – UMA Engine at the user terminal.	102
Figure 5.6 – UMA Platform building modules.	104
Figure 5.7 – UMA Platform architecture.	107
Figure 5.8 – Data objects model.	110
Figure 5.9 – Event scenario at the reception of an HTTP request.	113
Figure 5.10 – Event scenario at the reception of the content description.	113
Figure 5.11 – Event scenario at the reception of the user environment description.	114
Figure 5.12 – Event scenario at the end of the DIUED and MPEG-7 descriptions parsing.	114
Figure 5.13 – Table of users (sorted by IP address).	117
Figure 5.14 – Table of content (sorted by URL).	117
Figure 6.1 – Example decision process for image customization.	122
Figure 6.2 – Image decision algorithm.	124
Figure 6.3 – Video decision algorithm.	124
Figure 6.4 – The customization modules table.	125
Figure 6.5 – The table of customization modules implements an interface to the set of content processing methods available.	126
Figure 6.6 – Example of color space transformation, from a) to b), and bits per pixel reduction, from b) to c).	128
Figure 6.7 – Uncompressed domain adaptation.	128
Figure 6.8 – Compressed domain adaptation.	129
Figure 6.9 – Light source, illuminant chromaticity and color temperature concepts.	129
Figure 6.10 – Normalized spectral power distribution for blackbody radiators.	130
Figure 6.11 – CIE-xy color system and blackbody locus.	131
Figure 6.12 – Picture with sunset illumination ($T = 3,629$ K).	131
Figure 6.13 – Picture with day light illumination ($T = 6,194$ K).	131
Figure 6.14 – Flowchart for computing the image color temperature.	134

Figure 6.15 – Steps to transform the image color temperature.	140
Figure 6.16 – Color temperature adaptation: a) 3,497 K; b) 5,033 K (source); c) 6,990 K.	141
Figure 6.17 – Color temperature adaptation: a) 4,135 K; b) 7,924 K (source); c) 11,711 K.	141
Figure 6.18 – Color temperature adaptation: a) 2,627 K; b) 3,718 K (source); c) 8,640 K.	141
Figure 6.19 – Color temperature adaptation: a) 2,697 K; b) 4,452 K (source); c) 6,292K.	141
Figure 7.1 – UMA browser interface after retrieving the “My Multimedia Album” content.	145
Figure 7.2 – UMA browser posting a DIUED.	147
Figure 7.3 – UMA Platform Graphical User Interface.	147
Figure 7.4 – UMA Platform configuration options.	149
Figure 7.5 – MPEG-7 description tool posting a description to the MPEG-7 description server.	150
Figure 7.6 – MPEG-7 description tool dialog box.	150
Figure 7.7 – UMA System test bed in content server configuration.	152
Figure 7.8 – UMA System test-bed in proxy server configuration.	153
Figure 7.9 – “My Multimedia Album” content structure.	153
Figure 7.10 – “My Multimedia Album” content: (a) Lisbon pictures collection; (b) video collection.	154
Figure 7.11 – Compaq IPAQ terminal connected to a UMTS terminal.	156
Figure 7.12 – Compaq IPAQ Pocket PC terminal HTTP request.	157
Figure 7.13 – “My Multimedia Album” accessed in a Pocket PC terminal.	157
Figure 7.14 – Image not customized to the Pocket PC display resolution.	159
Figure 7.15 – Image customized to the Pocket PC display resolution.	159
Figure 7.16 – Image customized to the Pocket PC 2 display.	160
Figure 7.17 – Image customized to the WAP terminal 1 display.	161
Figure 7.18 – “My Multimedia Album” in WAP terminals: a) the main page; c) the page with the pictures collection; c) the Pisa pictures collection.	163
Figure 7.19 – Siemens ME 45 WAP terminal HTTP request.	163
Figure 7.20 – Good result of an image adaptation.	164
Figure 7.21 – Poor result of an image adaptation.	164
Figure 7.22 – Visualization of a video customized to the WAP terminal 2.	165

List of Tables

Table 1.1 – Examples of different content, network and terminal characteristics	3
Table 1.2 – Example of image customization by adaptation.	11
Table 3.1 – Types of content customization.	45
Table 3.2 – Media description tools.	47
Table 3.3 – Segment Entity description tools.	52
Table 3.4 – Segment Attributes description tools related to UMA.	53
Table 3.5 – Segment Relation description tools.	54
Table 3.6 – Summarization description tools.	55
Table 3.7 – Hierarchical summaries description tools.	56
Table 3.8 – Sequential summary description tools.	58
Table 3.9 – Views description tools.	58
Table 3.10 – Views decompositions description tools.	59
Table 3.11 – Variations description tools.	60
Table 4.1 – Attribute data type vocabularies [40].	70
Table 4.2 – CC/PP client attribute vocabulary [40].	71
Table 4.3 – Proxy vocabulary names [40].	73
Table 4.4 – Proxy behavior attributes [40].	73
Table 4.5 – DIUEDType element semantics.	85
Table 4.6 – ServiceProviderType descriptor semantics.	86
Table 4.7 – ConsumerType descriptor semantics.	86
Table 4.8 – TerminalType descriptor semantics.	87
Table 4.9 – TerminalSoftware descriptor semantics.	87
Table 4.10 – MediaDecoding descriptor semantics.	88
Table 4.11 – TerminalHardware descriptor semantics.	89
Table 4.12 – DisplayDevice descriptor semantics.	90
Table 4.13 – AudioDevice descriptor semantics.	91
Table 4.14 – AccessNetwork descriptor semantics.	92
Table 4.15 – NetTraffic descriptor semantics.	92
Table 4.16 – NetQoS descriptor semantics.	93
Table 4.17 – NaturalEnvironment descriptor semantics.	93
Table 4.18 – Sensor descriptor semantics.	94
Table 4.19 – SensorTypeCS classification scheme.	94
Table 4.20 – SensorUnitsCS classification scheme.	94
Table 4.21 – UserDescription descriptor semantics.	95
Table 5.1 – Request object attributes.	111
Table 5.2 – User object attributes.	111
Table 5.3 – MMContent object attributes.	111
Table 5.4 – infoStatus object attributes.	112
Table 5.5 – Message types.	115
Table 6.1 – List of customization modules	125
Table 6.2 – Image adaptation operations supported by the image customization module.	127
Table 6.3 – Iso-temperature lines according to Wiseck & Stiles [126].	138

Table 6.4 – Color temperature image customization measures.	142
Table 6.5 – Video adaptation operations supported by the video customization module.	143
Table 7.1 – Characteristics of the content tested.	154
Table 7.2 – User environment tested scenarios.	155
Table 7.3 – Image customization measures for the Pocket PC 1 scenario.	158
Table 7.4 – Video customization measures for the Pocket PC 1 scenario.	159
Table 7.5 – Image customization measures for the Pocket PC 2 scenario.	160
Table 7.6 – Video customization measures for the Pocket PC 2 scenario.	161
Table 7.7 – Image customization measures for the WAP terminal 1 scenario.	162
Table 7.8 – Video customization measures for the WAP terminal 1 scenario.	162
Table 7.9 – Image customization measures for the WAP terminal 2 scenario.	164
Table 7.10 – Video customization measures for the WAP terminal 2 scenario.	165

List of Acronyms

AAC	Advanced Audio Coding
API	Application Program Interface
ATM	Asynchronous Transfer Mode
CAD	Content Action Decision
CC	Content Customization
CC/PP	Composite Capabilities/Preferences Profile
CIF	Common Intermediate Format
CONNEG	Content Negotiation Working Group in the IETF
CT	Color Temperature
DCT	Discrete Cosine Transform
DI	Digital Item
DIA	Digital Item Adaptation
DIUED	Digital Item Usage Environment Description
DVB	Digital Video Broadcasting
GIF	Graphics Interchange Format
GSM	Global System Mobile
GPRS	General Packet Radio System
GUI	Graphical User Interface
HHC	Hand Held Computer
HP	Hewlett-Packard
HTML	Hyper Text Markup Language
HTTP	Hyper Text Transport Protocol
IBM	International Business Machines
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISDN	Intelligent Services Data Network
ISO	International Standards Organization
ISP	Internet Service Provider
JPEG	Joint Picture Experts Group
MP3	MPEG-1 Audio Layer 3
MPEG	Moving Picture Experts Group
MPEG-7	Multimedia Content Description Interface
MPEG-21	Multimedia Framework
MMCD	Multimedia Content Description

PAL	Phase Alternation Line
PC	Personal Computer
PDA	Personal Digital Assistant
PNG	Portable Network Graphics
PM/M	Pipeline Manager Monitor
QCIF	Quarter CIF
RDF	Resource Description Framework
SNR	Signal-to-noise Ratio
TCP	Transport Control Protocol
UAProf	User Agent Profile
UMA	Universal Multimedia Access
UMTS	Universal Mobile Telecommunication System
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URP	User Request Processor
W3C	World Wide Web Consortium
WAE	Wireless Application Environment
WAP	Wireless Application Protocol
WML	Wireless Markup Language
WWW	World Wide Web
xDSL	Digital Subscriber Line
XML	eXtensible Markup Language
XSL	XML Style sheet
XSLT	XML Style sheet Transformation

Chapter 1

Introduction

The exploding variety of multimedia information is nowadays a reality since everyone has a camera, a scanner or another device that almost instantly generates multimedia content. Most often the content author wishes to share its masterpiece with everyone, but the variety of terminals and networks may be a problem if he wants everyone to see his work with the best possible quality. This problem will increasingly grow as industry releases more enabling technologies. Late advances in Telecommunications, Information Technologies and Microelectronics, are creating a set of technologies that will enable the development of an all-new set of services with completely different requirements, terminals with completely different capabilities and networks through which terminals will access services in very different conditions.

With the advances in Telecommunications, new protocols and networks are being standardized, providing larger bandwidth and better access (e.g. Bluetooth, UMTS, xDSL, etc.). However, the different characteristics of these networks may impose restrictions regarding the content delivery process, e.g. related to bandwidth, error protection, quality of service, etc. Networks can be very different in several aspects: in fact, the same content may have to be delivered in a broadcast, a multicast or point-to-point mode; for that the network may provide connection-oriented, or connectionless services or even both types; and some networks may be able to negotiate the quality of the connection according to some criteria while others employ a best-effort policy.

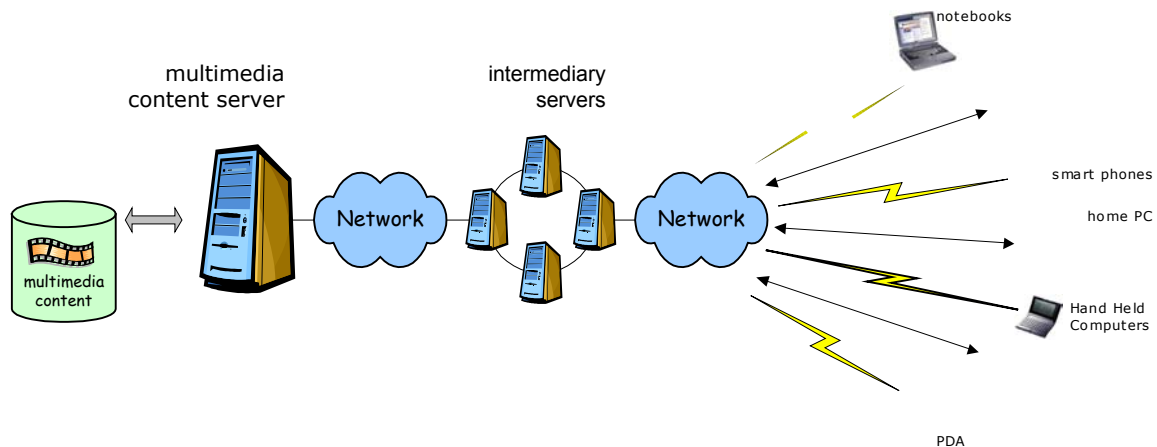
From the Information Technologies area, there are emerging tools enabling the creation and easing the proliferation of increasingly richer multimedia content: images (e.g. JPEG, JPEG2000, GIF, PNG), video (e.g. MPEG-1, 2, and 4, QuickTime, Real Video), audio (e.g. MP3, AAC), interactive content (e.g. MPEG-4, SMIL, W3D), etc. This rich multimedia content is used in many different services, such as Video-on-Demand (VoD), multimedia portals (news, cinema, sports, etc.), digital interactive TV, and communication systems, which are just some of the new applications enabled by emerging Information Technologies. Each service may have very different requirements in terms of network and terminal capabilities.

Advances in Microelectronics are enabling more powerful and varied terminals: workstations, Home PCs, Web TVs, mobile terminals of all types and many others. At one end of this range are high-end platforms with tremendous graphical capabilities, lots of processing power and large storage capacity. In the middle range, terminals with some limitations at certain levels can be found: a Home PC does not have the same disk performance as a workstation, neither the same graphical capabilities. For example, a Web TV is usually connected to a television (PAL resolution) that has less than $\frac{1}{4}$ of the resolution of a workstation monitor. At the other end of the range, there are mobile terminals, with the ability of serving as calendar tools, address books, paging devices, global positioning devices, travel and mapping tools, e-mail clients, and Web browsers. As a result, many new businesses are forming around applications related to bringing

all kinds of information to these terminals. However, the limited capabilities of many of the mobile terminals, make this case a very special one, creating new challenges in the design of applications that allow these terminals to access, store, process and generally consume multimedia information.

Concurrently with the developments above, recent advances in storage and acquisition technologies are driving the creation and dissemination of large amounts of rich multimedia content. As a result of these advances, there is a growing mismatch between the rich content that is available and the capabilities of the terminals and networks available to the users to access and process the content available. Today, when an author creates some multimedia content, he may have to create several variations of the same content, even with different layouts, different image resolutions and even different media types in order to reach the largest possible number of terminals.

This scenario describes today's situation of accessing multimedia content from any terminal as illustrated in Figure 1.1. Typically, when a terminal accesses content to which it was not designed for, the user experience is very poor. The user may experience long delays if the content is adapted ignoring the characteristics of the involved parts making its visual presentation less impacting, etc. The scenario described misses a bridging element between all the components involved, which should take into account their characteristics and assure an efficient and consistent inter-working. In other words, an efficient way to "access any information from any terminal", allowing the delivery of any content (or an adaptation of it) to the user should be provided.



providing closer starting points for the customization; even in the case one of the variations is the customization solution, there is still a matching (UMA) engine which dynamically selects the most adequate content variation following the received user conditions making this process transparent to the user. The UMA approach is rather a dynamic approach where a content variation is customized into different forms, by on-line processing or variation selection, depending on the actual access conditions, e.g. in terms of network and terminal.

Table 1.1 – Examples of different content, network and terminal characteristics

Content	Network	Terminal
Buffering size	Maximum/Average bandwidth	Memory
Streaming bandwidth	QoS	Network protocols
Number of color components	Broadcast/Point-to-Point	Display colors
Spatial resolution	Delay	Display resolution
Temporal resolution		Media formats
Media formats		Input capabilities (e.g. keyboard, mouse)
		Processing power

An application that makes available a customized content variation according to the users conditions is a UMA enabled application. In UMA applications, it would be desirable (to avoid the extraction on-the-fly) to have available descriptions of the parts that have to be matched/bridged – the user characteristics, and the content characteristics – in order to more easily customize the desired content:

- **Description of the content characteristics:** information on the content features, which are instrumental to perform an efficient customization of that content.
- **Description of the user environment:** information on the user conditions which are useful to make available a customized variation of the pretended content.

The Universal Multimedia Access concept involves the idea of content customization based on both the descriptions of the user environment and of the content. Considering a scenario where the content is customized according to these two types of descriptions, it is clear that such a system would have a component implementing the UMA customization that is the **UMA Engine** (see Figure 1.2).

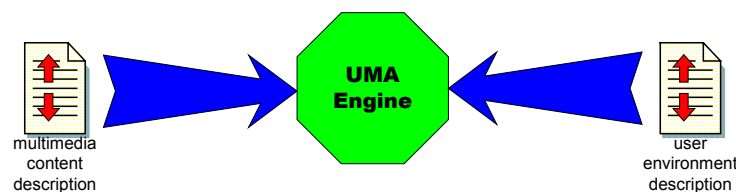


Figure 1.2 – Conceptual diagram of the UMA Engine.

Multimedia content features can be automatically, semi-automatically or manually extracted, depending on the type of features involved; certain types of features can only be extracted by heavy processing of such multimedia content. Content descriptions may also include a link to the multimedia content itself (and typically do).

To accomplish the task of fully describing the user environment, it is necessary to gather information about the user conditions and then provide it to the UMA Engine in order to receive a content variation that is tailored to the concerned user environment. These characteristics may have to be measured (bandwidth, location, etc.) or may be simply available (display color components and video resolution).

For better understanding the scope of the user environment and content descriptions, the diagram from Figure 1.3 represents the end-to-end path from the content server to the user terminal. Generally, information travels through several network domains: from the content server it passes to the transit network, the serving network, the access network and, finally, reaches its target, the user terminal. Figure 1.3 illustrates the following domains:

- **Content server:** the server located at the service provider infrastructure with the content to be accessed.
- **Transit network:** the network linking the content server and the serving network through switching elements, e.g. the IP routers in an IP network.
- **Serving network:** the network where the service provider hosts his servers to provide user services: mail, SMS, news, proxies, gateways, application server, etc; this infrastructure is near the user terminal. In this part of the network, data packages are translated to and from the specific user access network technology (e.g. ISDN or GSM).
- **Access network:** the network between the terminal and the network service provider infrastructure; this may be a GSM, GPRS, or ISDN network.
- **User terminal:** the terminal from which the user is accessing the service/content; it may have very heterogeneous characteristics in terms of processing power, memory, screen size, color components, etc.

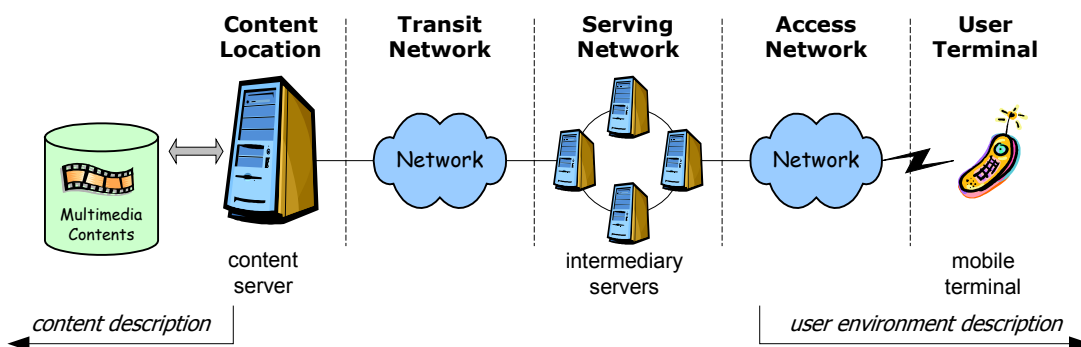


Figure 1.3 – Domains between the content server and the user.

Analyzing Figure 1.3, it is easy to conclude that the major restrictions reside in the access network and user terminal domains: the access network has typically less bandwidth and quality (e.g. high error-rate) than the rest of the path to the content, and the user terminal may have serious limitations in terms of processing capabilities, memory, or power consumption. From the content location to the consuming user, there are three entities with the ability to customize the content: the content server, the intermediary server (one or more together, giving birth to the notion of active networks), and the user terminal. All these three elements may accommodate a UMA Engine; of course, each location has advantages and disadvantages. Independently of the place where the UMA Engine resides, it is essential that a content description and a user environment description are available for matching in order to customize the content for optimum consumption.

1.1 Key Technologies

It is already clear that a multimedia content description and a user environment description are necessary to perform the content customization targeting a certain user. These two inputs must describe some

parameters of both interested sides and must be coded in some standard format since interoperability requirements are very relevant in this context. During the course of this thesis, a research was made to investigate existing technologies that could be used to perform these tasks: two major technologies were identified, MPEG-7 (Multimedia Content Description Interface) for content description and CC/PP (Client Capabilities / Preference Profile) for user environment description.

1.1.1 Multimedia Content Description

Several basic data types can compose a multimedia document: text, image, video, speech, audio, audio and visual synthetic, interactive elements (links, events on user action, mouse over, open new window, etc.) and structural elements (text formatting, images and video location, etc.). The visible and audible data types are text, images, graphics, video and audio (green boxes in Figure 1.4). Structural elements are not visible by themselves (blue box in Figure 1.4); they determine the spatial and temporal organization of the other data types. Interactive elements (brown box in Figure 1.4) provide a way for the user to interact with the content. Figure 1.4 illustrates a possible organization these six data types: content without any interactivity is called *static content* and content with interactivity is called *dynamic content*.

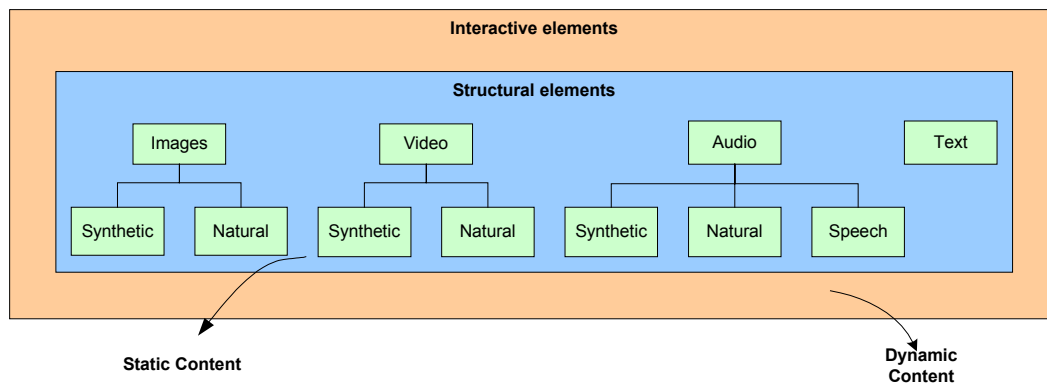


Figure 1.4 – Data types composing multimedia content.

Multimedia content embraces any audio and visual content type. When customizing multimedia content, UMA Engines process or select the content based on its description. If this description is not available, the UMA Engine must retrieve the content and extract the features that are important for the customization process, which involves a decision through matching on what type of adaptation is adequate and the consequent processing or selection phases. The feature extraction process may become a very complex task in some time-critical cases and thus would significantly decrease the efficiency of the UMA Engine if it had to be performed on-line. For example, if the adaptation is for a small screen, requiring the content structure to be rearranged and each image object to be shrunk, the UMA Engine needs to know the image features before deciding the best customization of the content structure. This implies that all image objects have to be made available to the UMA Engine before the decision is made and before the customized content structure is returned to the user.

Content description is a very crucial and valuable input, determining the UMA Engine performance. When the content's author is creating the content and ideally its description, he can provide very useful customization rules and hints that may help on the process (he is by the way the best person to state which is the most adequate adaptation for his own content). These rules and hints will be essential to improve the multimedia content adaptation performance and to help the UMA Engine in the decision process. Moreover multimedia content description must be coded in a format that assures fast access to the features and the process of decoding such descriptions must be as light as possible.

In Chapter 2, some UMA applications are presented and from them a list of requirements for the multimedia content description framework is derived. MPEG (Moving Picture Experts Group) has already performed a similar exercise: UMA applications were used among many other applications to draw the requirements of the newest MPEG standard, the MPEG-7 standard [83]. MPEG-7 is an ISO standard, formally named “Multimedia Content Description Interface”, with the objective to specify a standard way of describing various types of multimedia information, irrespective of their representation format and storage support. MPEG-7 represents information about the content (“the bits about the bits”) and not the content itself.

Among the applications identified to draw the MPEG-7 standard requirements gathering process were UMA applications. As a result, MPEG-7 is the most complete framework available for multimedia content description in general and also in the context of UMA applications. Therefore, MPEG-7 will be used in this thesis to describe multimedia content and thus it will be presented with more detail in Chapter 3.

1.1.2 User Environment Description

In a first approach, the term “user access conditions” implies the terminal and network characteristics; however, in a broader scope, there are many other factors that may influence the user’s access to information. Location, and user preferences (sports, films, etc.) are some other variables that may strongly affect the type of delivered multimedia information. The user environment description gives UMA its real dimension in terms of content customization: the user environment is understood as the set of characteristics that influence and/or condition the user’s access to content. A diversity of elements characterizing the user environment may enter in the equation that rules the content adaptation/delivery policy. Figure 1.5 shows the user environment dimensions to be considered along this thesis.

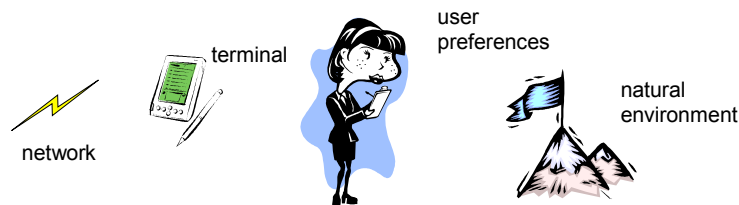


Figure 1.5 – User environment dimensions.

To refer to all the user access dimensions, the term “**user environment**” will be used in the following to designate the set of characteristics that influence and/or condition the user’s access to content, notably:

- **Network characteristics:** describe the network features that condition the multimedia content delivery to the terminal (e.g. bandwidth, delay, latency).
- **Terminal characteristics:** describe the terminal hardware/software characteristics that affect the multimedia content decoding, rendering and visualization (e.g. display capabilities, processing power).
- **User preferences:** describe the “taste” of the user in terms of content (or even more) for various types of services (e.g. sports, movies, food); for this type of information, critical privacy concerns may apply.

- **Natural environment characteristics:** describe natural features of the surrounding environment that may influence the content delivery (e.g. location, temperature).

The user environment description, as the content description, offers the UMA Engine a great advantage because without too much effort the UMA Engine has access to information that otherwise could not be known or could be very difficult to obtain. For example, when a request is sent to the UMA Engine, it must know which is the display resolution; without a description sent by the terminal, this information would have to be obtained through heuristics methods that may be rather inaccurate (e.g. if the request comes from a Palm terminal than it can be assumed that the screen size is 160x160 pixels).

The user environment description format must fulfill the requirements from UMA applications: in Chapter 2, some UMA applications are presented and from them a list of requirements for the user environment description framework is drawn.

A bibliographic research over technologies that could fulfill the requirements demanded by the user environment characterization was made during the course of this thesis. From this research, it was concluded that the W3C Composite Capability/Preference Profiles (CC/PP) framework [39] is today the best technology to describe the user environment if an adequate vocabulary is developed. This study and a detailed description of CC/PP are presented in Chapter 4.

1.2 Multimedia Content Customization

Up to now, the UMA Engine actions were generically referred to as “content customization”; this section provides a clearer view on which types of customization can be performed. Multimedia content has been decomposed in primary data types: text, speech, audio, image, video, audio and visual synthetic, structural and interactive elements. Because some interactive elements are software, it is very difficult to customize this sort of data type. The other data types (static content) can be more easily adapted to the user constraints.

With both descriptions available, a UMA Engine can customize the content by transforming it on the fly or by selecting the content variation in a way that the best possible experience is provided to the user. These two content customization techniques are:

- **Customization by selection:** supposing that several variations of the same content are available, addressing different kinds of user constraints, content selection corresponds to the identification of the most adequate variation from those available, and send it to the user. Existing content variations may include the same or different data types (e.g. video replaced by an image or text converted to speech using a cross-modal transformation). Content selection may also involve sending different information to different users based on their location: for example a different advertising banner for a higher impact.
- **Customization by adaptation:** involves the transformation "on the fly" of the content available, according to some criteria, even if more than one variation is available (but none is adequate enough). This process requires a lot more of computational effort compared with the customization by selection since for some content transformations heavy signal processing algorithms may have to be performed with a low delay.

Content adaptation and selection are the two content customization types of techniques performed by a UMA Engine and can be executed independently or in combination. For example, after a content selection a content adaptation process may be executed still for better adaptation. All these operations have the purpose of improving the user experience when accessing content. For example, much of the

available multimedia content is inadequate for small screen terminals. For instance, take a look at a typical Web news content site: usually a news site has an advertisement part, a search form for finding news, a navigation bar for easier access to the several site sections, the opening news article and a small part that presents results with regular updates (possibly personalized information concerning soccer results, stock exchange values, etc.). Figure 1.6 presents a high level representation of such a Web site in its original form (a single entrance page) and in a customized form (multiple pages). In such type of content, there are parts that are more important than others, and so they should have higher priority when the content is adapted. When the news site is customized to a small screen terminal, the red section with the most relevant information should appear immediately in the first page of the small screen terminal, and the advertisement should appear in all pages. The entrance page has the advertisement, the red section (most important) and two links to the green section and yellow sections.

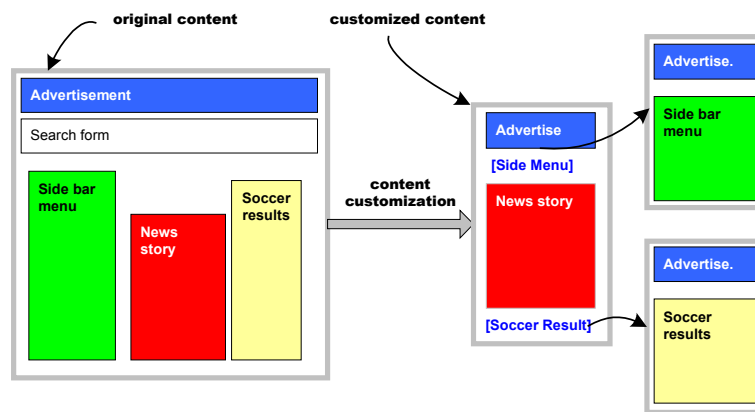


Figure 1.6 – Multimedia content customization example.

Customization is intended to improve the user experience when he accesses information. A very interesting study made regarding the user experience with WAP (Wireless Application Protocol) terminals showed that “the users gave up easily if they had to scroll down to find a piece of information”, [48]. Content customization must be performed in such a manner that it accelerates the user access to the pretended information. Figure 1.7, illustrates the difference between an “intelligent customization” (left side) and a massive adaptation (right side), where all elements are converted, creating a very dense document that might become very difficult to browse through in small screen terminals.

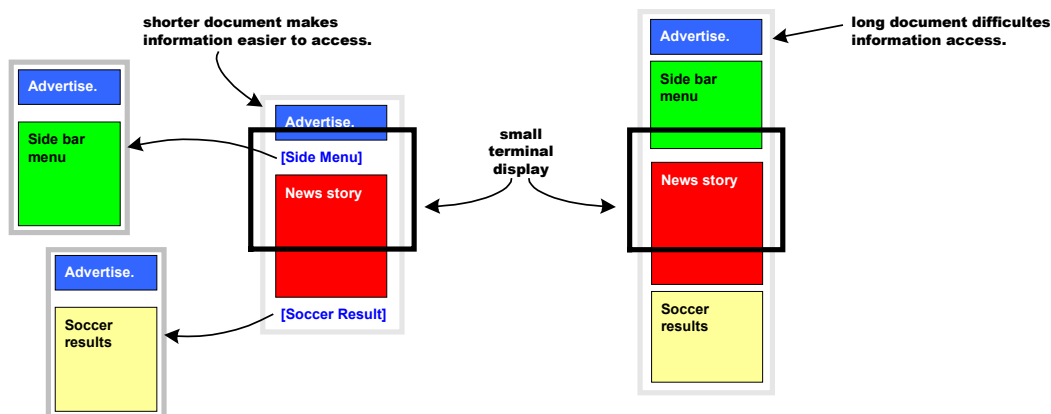


Figure 1.7 – Enhancing the user experience with adequate content customization.

The example in Figure 1.7 shows how content customization can in fact improve the access to information. Multimedia content customization is a key factor to accomplish the user demands by making a variation of the content that perfectly fits the user environment.

1.2.1 Content Customization by Selection

In a UMA enabled application, content selection is based on terminal characteristics, network characteristics, natural environment characteristics and user preferences. The UMA Engine must analyze the user environment description, the content description and then decide which content variation best fits the concerned user environment. This scenario is illustrated in Figure 1.8.

At the moment, content selection is the solution used in many Internet sites; this means that different variations of the same content are created differing on the content requirements (screen size, graphic capabilities) so that certain classes of terminals (e.g. WAP terminals) may access the site. The author of the content must create a variation for each kind of user terminal that he wants to be able to access his content. Nowadays, each variation is accessed through different site addresses or at the site main address, with the user selecting the variation that fits his user environment (e.g. <http://tvradiationnetwork.com/>).

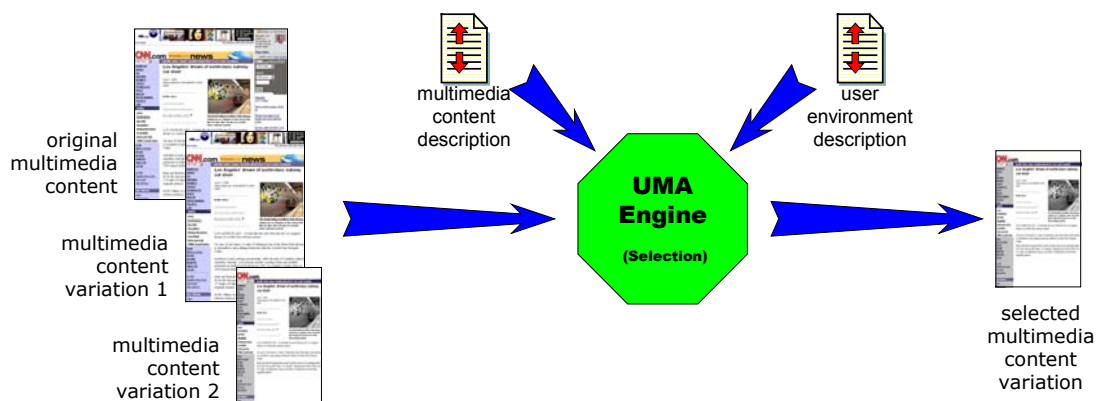


Figure 1.8 – Content customization by selection.

The customization by selection approach has the following advantages:

1. Simple solution.
2. Very fast because the content to send is immediately available.
3. Good approach when dealing with large amounts of data that do not have any regular update and so no human resources are required to maintain the several variations.

And the following disadvantages:

1. One variation per terminal; new terminals have to live with existing variations.
2. When content changes (in some cases, daily), the time and manpower to update all variations may be critical.
3. The costs involved in implementing such a solution might become very high due to the storage and human resources required to maintain the content updated.

Content customization by selection is a very useful solution in UMA applications when it is necessary to deal with existing content variations that contextually change according to the user personal preferences or environment (e.g. location) and when very fast access is required.

1.2.2 Content Customization by Adaptation

The main idea behind this approach is to have one single content source linked to its description including powerful information concerning content encoding conditions, most important parts and other information that can enhance the adaptation process. With the multimedia content and the user environment descriptions available, the UMA Engine has the task of adapting the content in a way to provide the best experience to the user.

Content descriptions carry information that can decrease the time to adapt the content on the fly. Content customization by adaptation involves a trade-off between the time to adapt the content (t_{adapt}), the time to transmit the content (t_{trans}) and the time that the user waits for the content ($t_{\text{wait}} = t_{\text{adapt}} + t_{\text{trans}}$). There are cases where the content data size is reduced by the adaptation and then the delivery of such content to the user is faster, making the overall time smaller even if an adaptation time has to be considered. In other cases, the adaptation process may be very long making the overall time much longer; of course, this mainly depends on the computational resources available at the UMA Engine.

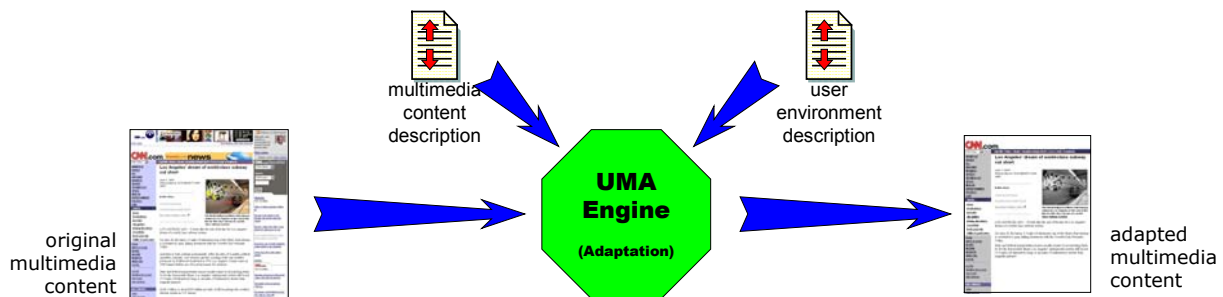


Figure 1.9 – Content customization by adaptation.

The customization by adaptation approach has the following advantages:

1. Storage and manpower costs decrease since only one content variation has to be stored and maintained.
2. Shorter delay to transmit the content since the adapted content data size is typically smaller.
3. Independent of the type of terminal type: if a completely new terminal tries to access the content, the UMA Engine adapts the content according to the new terminal characteristics and thus may satisfy an infinite number of terminal types.

This approach has the following disadvantages:






1. Longer delay to access the content due to the adaptation processing time.
2. Computational resources required at the UMA Engine are much higher due to the adaptation process.

The types of data that may have to be subject to adaptation through processing are all media types, e.g. images, video, audio, and text data, as well as structural and interactive elements. The most demanding data types in terms of processing effort are the media types, notably video.

Table 1.2 shows an example of the adaptations that can be performed over an image. Table 1.2 lists six types of terminals each one with different display resolutions and color capabilities. The image resolution and the number of colors are gradually reduced according to the terminal in question. Consequently, the

image data size also decreases, which causes the access time to be reduced. Depending on the terminal capabilities, the image is adapted and in the last case (for a mobile phone, with no graphical capabilities), the image is replaced by text naming the image. For example, a Hand Held Computer (HHC) does not have the display resolution or the number of colors that a workstation has, so the image resolution/colors are reduced according to its characteristics.

Table 1.2 – Example of image customization by adaptation.

Workstation	Home PC	TV Browser	HHC	PDA	Smartphone
					"Lighthouse"
Image size: 384 x 256	288 x 192	192 x 128	192 x 128	144 x 96	-
Data Size: 289 kByte	56kByte	26kByte	13kByte	2kByte	11 Byte ¹
Colors: 24bit RGB	256 colors	256 colors	16 colors	2 colors	-

Content adaptation implies the existence of one single content version or only a few variations and from it(them) the generation or selection of the most adequate variation that fits the user environment constraints. However content adaptation may require very demanding processing algorithms. This type of content customization is the major topic of this thesis.

1.3 Objectives and Structure of the Thesis

Visual data is the richest information source that humans have, since it is through the vision that humans receive the largest portion of information they consume: it is very important to have a familiar visual interface and type of interactivity on every terminal that a user interacts with (as much as the user environment allows it); for example the Windows PC version and the Windows Pocket PC version have a similar way of interaction. The user should have the best experience for the same information, under the access conditions he/she is constrained to [82]. In a constantly changing world, emerging problems are identified and rather quickly solved by new or existing technologies, creating a new set of services/terminals for the user. This creates the need to keep the interoperability with these new services/terminals so that the user does not have to learn different interfaces for each kind of service/terminal. Universal Multimedia Access is the glue between all these services and terminals by providing a way to bridge different technologies that otherwise could not operate with the same content.

Universal Multimedia Access is a very broad concept that aims to provide the user the best possible experience for a piece of content, taking into account the conditions under which the user accesses the information. Nowadays users want to access the Internet from their mobile terminals, having available the biggest source of information and, in the future, the largest commercial "zone" in the world. The promise, of course, is access to information, anytime, anywhere, from any terminal and to everything, thereby boosting productivity and enabling more efficient and attractive ways to perform tasks.

Due to the growing importance of mobile terminals in today's society, this thesis will focus on the customization of visual content (image and video) to mobile terminals. The main objectives of this thesis and, in fact, its achievements are:

¹ The word "Lighthouse" uses 10 characters plus one more to indicate the end of the string or its length.

- Review the state-of-the-art of UMA applications and solutions in order to know which approaches for the UMA concept have already been deployed and which conclusions were obtained that can be exploited for this thesis.
- Study and use the MPEG-7 standard to code the multimedia content descriptions.
- Investigate the user environment description problem and propose a solution if needed.
- Implement a complete system (the so-called UMA System) capable of testing and proving the UMA concepts applicability for a real scenario.
- Implement a multimedia content customization engine (the so-called UMA Engine) that may be easily integrated in any system capable of intercepting the content requests from the user terminal to the content server. The UMA Engine should be able of customizing:
 - Images based on color component reduction, spatial resolution reduction, region of interest, compression ratio, and color temperature.
 - Video based on color component reduction, spatial resolution reduction, region of interest, and temporal resolution.
 - Using MPEG-7 content descriptions and the user environment description format proposed.
- Test the implemented system in real scenarios with different user environments.

In the context of the mechanisms developed for this thesis, only visual content is processed in the UMA Engine; due to time constraints, the structure of a document and its textual content will not be processed for adaptation. The UMA Engine was designed in such a way that it may be located in a content server or in an intermediary server; more details on this are given in Chapter 5.

The structure of this thesis is as follows:

- **Chapter 2, UMA Applications and Requirements:** several applications and experiments where a UMA Engine plays a central role are described and requirements and conclusions are drawn from these applications/experiments.
- **Chapter 3, Multimedia Content Description:** reviews the general problem of content description; the MPEG-7 standard is presented with a special emphasis on the tools of this standard that are more relevant for the specific objectives of this thesis.
- **Chapter 4, User Environment Description:** reviews the general problem of user environment description, specially the W3C Client Capabilities/Preference Profile standard and the WAP extension vocabulary to CC/PP; a proposal for a user environment description solution under the MPEG-21 umbrella is presented.
- **Chapter 5, UMA System Design:** the implemented UMA System is presented in detail together with the UMA Engine internal architecture; some considerations regarding networking issues are made for two possible network topologies.

- **Chapter 6, Content Customization Processing:** describes the UMA Engine notably the processes used to select and adapt visual content according to the content and user environment descriptions.
- **Chapter 7, User Interfaces and Test Scenarios:** the applications developed during the course of this thesis are presented; the tests performed to the implemented system and the results obtained are described.
- **Chapter 8, Conclusions and Future Work:** states the conclusions derived from the work performed in this thesis and identifies future tasks that can be performed to further improve and develop the work presented.

In this first chapter, a global view of the Universal Multimedia Access challenge was given; this challenge is the starting point of this thesis.

Chapter 2

UMA Applications and Requirements

In order to identify the UMA requirements, this chapter will present and discuss four applications that benefit from a UMA like engine. Chapter 1 already described the complete UMA scenario: the user side, the content side and the customization engine. Figure 2.1 illustrates a high-level block diagram of the UMA Engine, distinguishing four major modules: the content description analyzer, the user environment description analyzer, the decision and adaptation/selection modules.

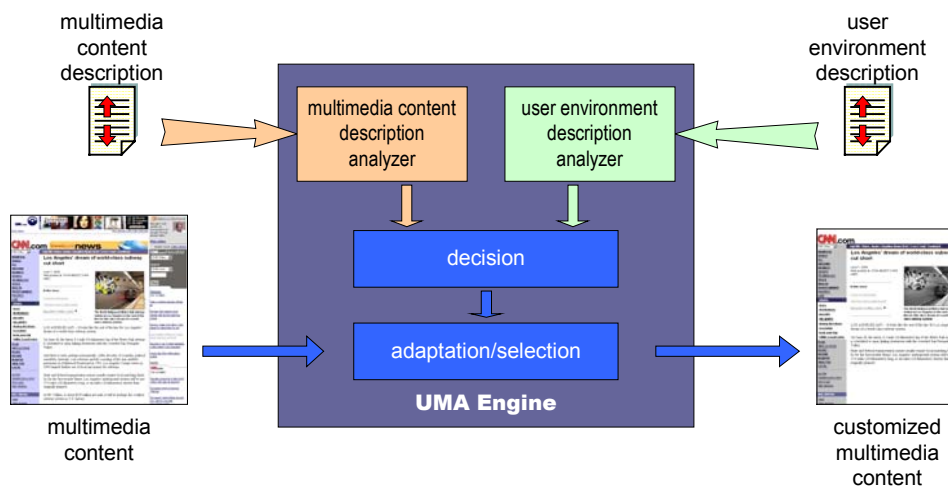


Figure 2.1 – High-level block diagram of the UMA Engine.

The UMA-based applications to be analyzed will be used to define the UMA-related requirements for the following four technologies or modules:

1. Requirements on multimedia content description tools.
2. Requirements on user environment description tools.
3. Requirements on content customization algorithms.
4. Requirements on UMA Engine architecture

It has already been discussed that a UMA Engine may use two primary content customization approaches: content selection and content adaptation. The examples in this chapter illustrate the use of

these two content customization approaches (either in combination or separately) as well as how the availability of both the user environment and multimedia content descriptions can improve the performance of a UMA Engine. In addition to these applications, two important UMA-related experiences (from IBM and HP), presented in several papers, will be analyzed and compared. These two UMA implementations provide important clues for solving some of the more common problems as well as useful experimental results and inputs for this thesis.

2.1 Applications

Within this section, four applications where the UMA Engine plays a major role will be presented:

- **Customization proxy:** the UMA Engine intercepts the content that is being delivered to the user and customizes it.
- **Universal digital libraries:** the UMA Engine is installed in a content server, with all the content that has to be adapted locally available.
- **Universal mailbox:** the UMA Engine is installed in a server that offers the user multiple services. Each service has individual private information for the owner to access under different conditions. In this case, the UMA Engine provides a content customization layer for generic services.
- **Client-driven multimedia browsing:** the UMA Engine is installed in the client. This application is important because it avoids the privacy issues that may be critical for non-client based content customizations.

2.1.1 Customization Proxy

The most direct application of the UMA concept is the provision of World Wide Web access for a wide-range of client devices, such as smart phones, PDAs, hand-held computers, etc. Since the Web has a lot of rich multimedia content, the user terminals often do not have the capabilities necessary for retrieving, processing, storing, and rendering all the rich multimedia content available.

One solution for the above problem is to integrate the UMA Engine in an intermediary server between the content server and the user terminal with the task of accessing multimedia content on behalf of the user terminal and adapt it, on-the-fly, to the user environment. This scenario is illustrated in Figure 2.2 showing an intermediary server acting as a customization proxy.

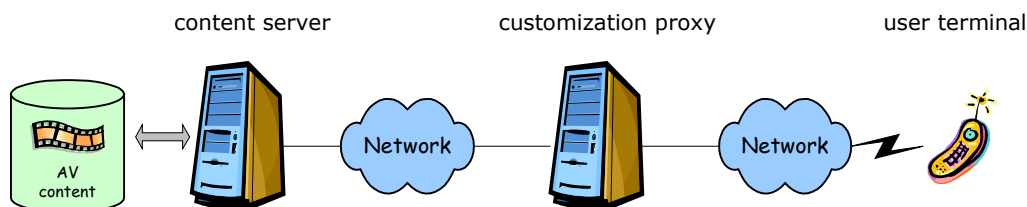


Figure 2.2 – Customization proxy scenario.

In a WAP scenario, this application can reveal very useful: WAP terminals only consume content in a specific format (WML – Wireless Markup Language) and only support certain image formats; for example, they cannot consume common Web content such as HTML, JPEG, GIF, etc. Therefore, such an application is a great improvement for WAP terminals, since they will be able to access a content variation that is adapted to their capabilities.

In general, the customization proxy does not select the content for the user based on user preferences; this task is performed by the content server or by the user terminal (for user privacy protection). The most important task that customization proxies execute is content adaptation. In the case the content descriptions are not available, the proxy must build the content description itself (must extract the features) in order to adapt the content to the user environment. However, in many cases the content descriptions are made available to the proxy server.

Customization proxies (and common proxies) are located in the Internet Service Provider, which means that they are close to the user terminal. This means that customization proxies will serve the clients of a certain ISP, which all have some common characteristics (e.g. NTT DoCoMo serves mobile users and AOL Time Warner serves PC based users). Because the clients of customization proxies follow a certain pattern, the customizations that are performed for each user should all be rather similar. This application example illustrates two aspects: specific methods must be developed for each type of situation (e.g. different coding tools for different types of terminals); moreover a caching mechanism may allow the re-use of previous content customizations improving the UMA Engine performance.

2.1.2 Universal Digital Libraries

Digital libraries deal with the cataloging, searching, browsing and retrieval of multimedia content. Digital libraries are finding application in fields as diverse as education, science, video production, and entertainment. In education, many universities use them to manage multimedia class lectures and related materials. Also many scientific communities generate large amounts of multimedia material that must be managed, such as telescopes or satellite observations.

Information stored in digital libraries is information that is not very commonly updated: once the digital library content is ready for access, it rarely changes; the most likely operation in a digital library is the addition of more content. Another typical characteristic of digital libraries content is the high quality in which the content is stored which produces great amounts of data. Due to the big size of this type of content, adapting such content can be very compromising for the performance of the digital library server and for the user experience, notably if some type of interactivity is involved.

A good approach to improve performance and facilitate the adaptation process is to have two or three variations of the original content that can be nearer to the pretended customization than having only the original material. Moreover, since in this case the UMA Engine is installed in the content server, it is easily possible to store and manage the results of adaptations made to take benefit of them in future adaptations, see Figure 2.3. This causes the need for being able to manage different variations of the same content (which also exists for the customization by selection approach). In this case, content selection becomes the most important task of the UMA Engine, since a good management of the content variations (generated either directly by the author or by UMA Engine) is essential for the overall system performance.

Each content variation must have a description attached to it including the relevant descriptors for adaptation and the relative content value of the variation compared to the original content. In creating multimedia presentations, the content authors can also add information indicating the relative importance of the various content objects within the presentation. This may allow the UMA Engine to improve the creation of the different variations, maximizing the total content value presented to the user within the constraints put by the user environment.

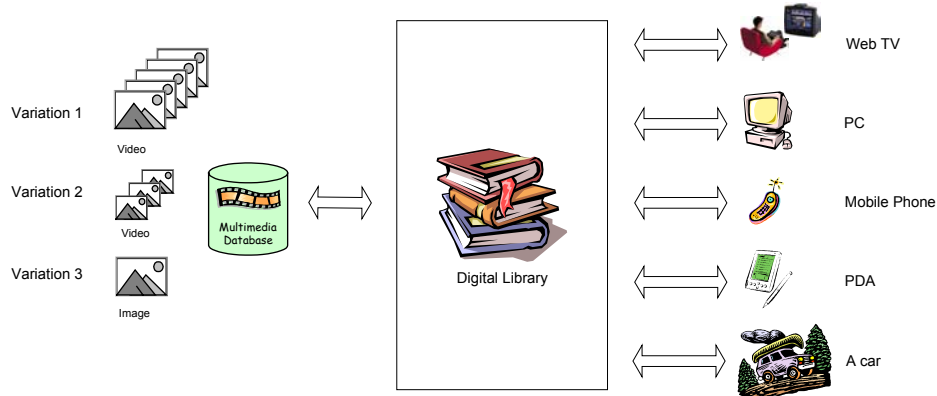


Figure 2.3 – Digital libraries can store several content variations.

2.1.3 Universal Mailbox

In this application, special emphasis is given to the user environment characteristics and how they are important to perform a good content customization. Even with today's rather primitive terminals such as answering machines, mobile phones, and faxes, each operating in one or at most very few different content data types (text, image, video, audio, etc.) the common user encounters many times difficulties in managing the access to information that each particular terminal provides.

Today, one has to read his/her fax and email, listen to his answering machine and talk through his/her telephone in order to interact with information. Tomorrow everybody will still face the same situation but the technology may help the user by centralizing the access to all these services to just one central point: the universal mailbox. One likely trend in future personal computing and communications is that there will not be a single but several terminals and networks available to users, allowing to access information in various forms. Each user, depending on his/her needs, will access the same information in different forms depending on his/her situation (e.g. location), and preferences. The following examples illustrate this scenario:

- **Privacy:** imagine a situation where a person in a public area wants to access his received messages or any given information. For privacy reasons, maybe the user would prefer to have the information read to him on a personal audio device (like a mobile phone) and not presented on a display which would make it easily readable also by others sitting next to him.
- **Attention:** while driving, a map would be more useful if directions and information about the neighborhood are presented in the form of audio rather than visually as this would divert the attention of the driver.
- **Noisy environment:** it may be useful to receive a message left on an answering machine as a written text (for instance, if the message was generated or is being consumed in a very noisy environment).
- **Display and decoding:** in calling a smart camera installed in a high-way to receive information about the fluidity of the traffic, the camera may be accessed by a Web TV or by a mobile terminal, so it is necessary to adapt the video to the terminal displaying and decoding capabilities.
- **Network restrictions:** the user might access a video e-mail and request to hear the voice but only see some key frames of the video (based on the available bandwidth).

All these are use cases that illustrate the need of a good user environment description. In these scenarios, there is a need for an application integrating the access to all these communication services to one single access point: the universal mailbox, see Figure 2.4. The universal mailbox implements a new application model, which helps the user in managing his communication and information access. In a universal mailbox environment, all sorts of communication services are centralized in one point and from there, the user may have access to the services, from any user environment. These services will generate the necessary adapted content, which will be sent to the user no matter which natural environment, terminal or network conditions he/she is using for accessing his universal mailbox.

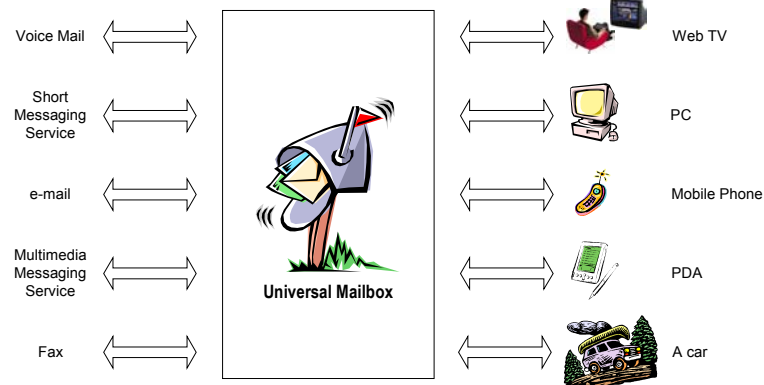


Figure 2.4 – Universal Mailbox scenario.

Using such a universal mailbox, the above problem of accessing many information centers such as voice-mail, SMS, e-mail, fax, etc., is solved. The user just needs to interact with the universal mailbox in order to access all types of information. However since a wide range of user environments with different characteristics may exist, the universal mailbox asks for the user environment description to customize the content.

From the example scenarios, it is concluded that the user environment characteristics play a very important role. In terms of content customization, there are no requirements on content selection because there are no content variations available, and the content adaptation requirements are less demanding than the ones of the customization proxy. The requirements for this application are determined by the fact that the user accepts a lower quality and higher delays in messaging services than for other applications. Another point to focus is the large number of requests that this application can receive possibly creating congestion. Nevertheless, decreasing the quality of service and thus reducing the resources needed to serve a user may solve this situation by serving more users with less quality as the number of users increase.

2.1.4 Client-Driven Multimedia Browsing

So far, the presented applications customize the content in a server removing every content customization task from the client. An application under test by NTT Docomo, illustrates the case of content customization (only by selection) at the client in mobile environments.

In this UMA application, the client starts by accessing a service in a content server (e.g. video library), which will then send the client a list of the available content choices (the content variations description, e.g. video coded with different bit rates) and eventually the tool to perform the selection also considering the user environment conditions. After the client selection decision based on the available choices and his own preferences (and capabilities), the final request for the selected content is then issued to the content server. Figure 2.5 illustrates the client-driven multimedia browsing application. The application relies on the terminal capability to receive and process the content descriptions, e.g. using a Java agent. The mobile

agent is the entity that travels to the mobile terminal and matches the users preferences with the content description and sends the selection decision to the content server.

The main advantage of this application is the way it deals with private data (e.g. the user preferences): here the user never has to send its personal preferences through the network. Moreover this scenario allows distributing some processing tasks to the clients, which some companies believe, may be a good idea, at least for some specific environments/terminals. There are other scenarios where the client side adaptation is also the best solution, but in those cases the terminals typically have more capabilities than a mobile terminal (e.g. a PC or a WebTV) or the user knows that in order for his/her terminal to perform some additional tasks his/her terminal autonomy will be severely reduced (e.g. in a PDA or in a PocketPC).

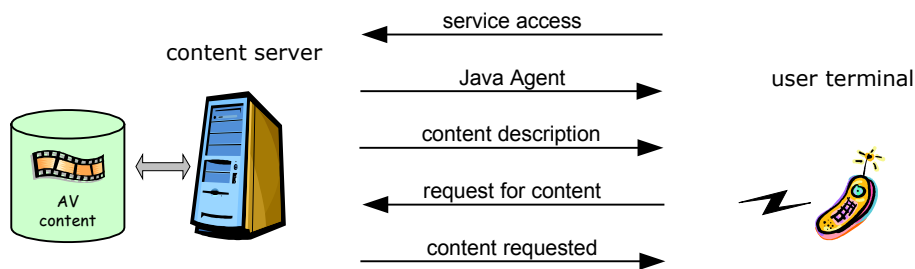


Figure 2.5 – Client-driven multimedia browsing scenario.

This application is quite different from the remaining and poses several challenges. So far only content selection is performed in the client, however with tomorrows mobile terminals capabilities the content customizations may perhaps be performed in the terminal as well.

2.2 Relevant UMA-Related Work

Several important companies are working nowadays on the issue of multimedia content customization. The targeted area is typically to provide Internet access for terminals with rather small resources, typically mobile terminals. Some of the companies that are working in multimedia content customization are IBM [1], Hewlett-Packard [2], NTT Docomo, Ericsson [63], Nokia and Intel. In this section, the two most relevant UMA-related experiments described in the literature will be presented and discussed: the IBM and Hewlett-Packard cases. Both systems divide the UMA problem into four parts even though they may use a different architecture:

1. User environment description.
2. Content description.
3. Content customization decision.
4. Content customization processing.

HP implements the UMA technology as a customization proxy and a content server in [2]. IBM describes the implementation of UMA technology in a Web server [1] even though they also have performed some experiments using a proxy architecture [72].

2.2.1 IBM UMA-Related Experiments

IBM has been developing UMA technology for several years. In the site they dedicate to UMA issues [71], there is a very good overview of IBM UMA-related work. In [77], a digital video library application is discussed, focusing on the real-time adaptation of video for different terminals. In [79], this application is

analyzed from the point of view of accessing large compressed images. In [76], the authors propose a system for accessing Internet content through a customization proxy. And finally in [72], the authors present a very complete study on how transcoding proxies can be efficiently used to overcome network congestion, by discussing several ways to evaluate when the transcoding should be done and which level of quality reduction should be performed. IBM's work on UMA technology has started by integrating it in a Web server and has been evolving with the addition of new functionalities and new solutions to critical problems. The work started by assuming that there were no multimedia content descriptions available and so it would be necessary to analyze the content. Meanwhile, IBM research efforts in this area are visible through a very significant involvement in MPEG-7, CC/PP and WAP standardization projects.

In [1], IBM presents the most complete description of their work, notably the implementation of a system that adapts multimedia content as an extension to a Web server. The IBM system architecture is presented in Figure 2.6. The content server contains the multimedia content to be delivered by the Web server. First, content is analyzed to extract descriptions to be used in guiding subsequent content adaptation and selection processes. Based on the capabilities of the user terminal, different transcoding modules are employed to generate variations of the content in different resolutions and modalities.

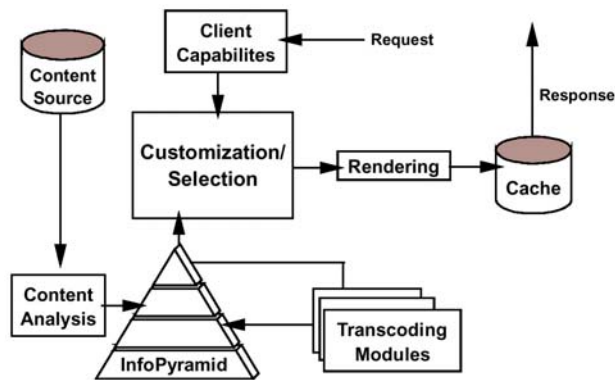


Figure 2.6 – IBM content adaptation system architecture [1].

A description of each module in this architecture follows:

- **Client capabilities:** according to [1], the user environment characteristics are manually entered in the system through a form. The following terminal and network capabilities were considered:
 - Screen (width and height in pixels; number of colors per pixel)
 - Network bandwidth
 - Payload (total amount of bytes that can be delivered to the user terminal for static Web content)
 - Capabilities for displaying/hearing video/image/audio
- **Content source:** this module stores the multimedia content available.
- **Content analysis:** the content is analyzed to extract descriptions that will be useful for the customization process. Two types of content analysis are performed: analysis to know the content semantics (e.g. by analyzing some textual descriptors associated to the content) and to define the resources needed in the terminal and network to deal with the content:

- Content size in bits (important for buffer sizes);
 - Display size (height and width in pixels and area);
 - Streaming bit rate;
 - Color depth;
 - Compression formats;
 - Hardware requirements (e.g. memory).
- **InfoPyramid:** framework for aggregating multimedia content together with its descriptions and the methods and rules for handling the content and generating the content descriptions. The InfoPyramid describes variations in terms of data types (modalities) and in terms of fidelity (e.g. spatial or temporal resolution), as illustrated in Figure 2.7. Methods are used to generate content descriptions (acting as feature extraction); for content adaptation, there are several methods for adapting the content between modalities and fidelities (these transcoding methods are in the transcoding modules). The InfoPyramid is not an active module; it is like a database, which relates the content variations and the methods for the other modules to process them.

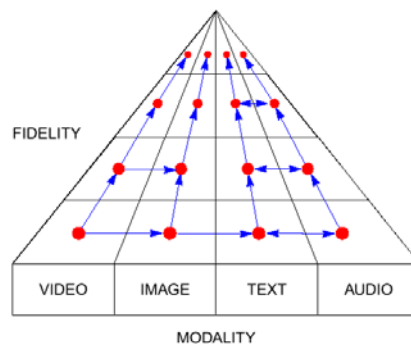


Figure 2.7 – IBM InfoPyramid for content customization [1].

- **Transcoding modules:** these modules are responsible for adapting the content. When a certain content item must be adapted or converted, it is by this module that the content is transformed. Transcoding modules enable to walkthrough the several elements of the InfoPyramid, in terms of modalities and fidelity (Figure 2.7). Transcoding is done off-line.
- **Customization/selection:** this module uses the user terminal characteristics as constraints to pick the best content variation. The best variation is the one that maximizes the content value for the user in question.
- **Rendering:** when the server processes a request, the result from the customization module is an XML document that can be rendered in HTML, WML, or any other format. The HTML page delivered to the user has appropriate textual elements incorporated and links to the right content variations. Thus, subsequent requests to the content objects included in the page do not require any processing overhead at the server.
- **Cache:** each content variation is stored with the associated user environment descriptions. When the system receives a request, it first checks if a request for that content and for a similar user environment has already been done; if so, the content is delivered from the cache.

The decision of the multimedia content to be delivered to the user is made in the Customization module. The decision algorithm uses two parameters: a **content value** for each variation and a corresponding terminal **resource allocation**. A **content value/resource allocation** curve is drawn and afterwards the optimal content variation is selected as the one that maximizes the content value for the used resources.

In [1] multimedia news content was used to demonstrate the content adaptation system. Each news article is composed by a title, a video and the text of the news story. Each content object is injected into its InfoPyramid and is then transcoded to populate the InfoPyramid:

- Video content is compressed at different bit rates and spatial resolutions;
- Video is converted to images by extracting key-frames;
- Images are converted into multiple color depths and resolutions;
- Audio is separated from the video and compressed at multiple bit rates;
- Textual elements of the story are converted into smaller text summaries;

An initial page with some forms is used to manually enter in the UMA system the user environment characteristics. In practice, techniques for the automatic discovering of user environment characteristics could be used. Figure 2.8 shows an example with the delivery of the same news story to a workstation, a PDA and a smart phone. The workstation gets the full text of the story and streaming video; the PDA gets a text summary and two key-frames in 2-bits gray-scale; and finally the mobile phone gets a short title and some audio.



Figure 2.8 – IBM multimedia news system working for different terminals [1].

The experience with the decision algorithms allow to conclude that it might be essential to assign priorities to content objects in composite multimedia content to get useful content adaptation policies. Another conclusion is that a good management of the content variations is also very important for the overall performance. The content descriptions revealed to be very effective in terms of the added value that carry to the decision algorithms.

2.2.2 Hewlett-Packard UMA-Related Experiments

Hewlett-Packard (HP) developed both content server and proxy based content adaptation frameworks. Information available on the Hewlett-Packard UMA-related work is very scarce; however, [2] gives a very

good overview of the UMA problems. This paper describes the implementation of a modular system so that it can be easily extended with new adaptation techniques, and integrated in any system architecture (e.g. proxy-based or content server).

The basic architecture, shown in Figure 2.9, consists in five modules: the first module addresses management and authoring purposes; the second, **Web platform API**, is an interface for integration with other systems (e.g. proxy); and the three most important modules are: the **adaptation** module, where the content is processed, the **decision engine**, which decides the content adaptation actions, and the **user/client/network monitoring**, responsible for determining the user environment characteristics.

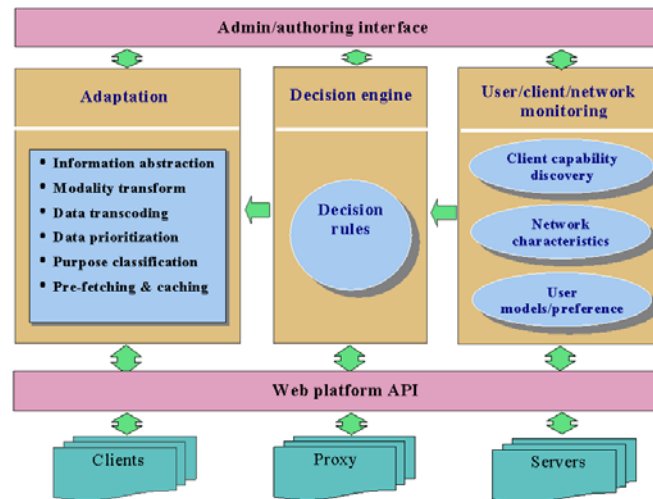


Figure 2.9 – HP adaptive content delivery system [2].

The HP **Adaptation** module includes five content customization techniques, presented in the following [2]:

- **Information abstraction:** technique with the goal to reduce the bandwidth needs for delivering the content by compressing the data, while preserving the information that has highest value to the user.
- **Modality transform:** process of transforming the content from one content data type to another so that the content may become useful for a particular client device not able to consume the first media type (e.g. transforming a video into an image).
- **Data transcoding:** process of converting a data format according to the user terminal capabilities. For example, some client devices may not be able to decode MPEG-4 video; in such cases it is essential to convert the video to a supported format, such as MPEG-1 Video, so that the user may consume the content.
- **Data prioritization:** the goal of data prioritization is to separate the more important parts of the data from the less important parts so that different quality of service levels can be provided when delivering the data through the network.
- **Purpose classification:** a typical Web page contains much information and many content objects that may not be of interest to a user. A portal site usually contains many images of banners, logos, and advertisements. These data often consume a good deal of network bandwidth and, therefore, decrease the efficiency of information delivery. If it is possible to classify the

purpose of each content object in a Web page, the system can improve the efficiency of information delivery by either removing redundant objects or prioritizing them according to their purpose/importance.

All these techniques are rather common. However, the techniques to classify the purpose of each content object, may return some misleading results due to its heuristic nature: the system searches for certain keywords (such as “ad”, “banner”, “promotion”, “advertisement”, etc.) in the file name, link names, etc. and based on that classifies the content in relation to its purpose.

The basic structure of the HP **decision engine** is shown in Figure 2.10. The inputs to the decision engine are the content itself and a description of the content including, for example, content types, content size, and purpose of usage in a Web page. The decision engine first checks the user preferences to see if it needs to remove or substitute any redundant objects (such as replacing a image bullet list by a text bullet list) in order to save bandwidth. After, the decision engine further checks the client capabilities and characteristics of the network that the client uses to connect to the server. Based upon the collected information, the decision engine determines if it needs to launch a particular content adaptation algorithm for a particular piece of content. The decision engine is structured in three stages:

- The first stage involves a binary decision controlling the use of modality transformation or data transcoding.
- The second stage involves a careful examination of various parameters to find out the best trade-off between content adaptation quality losses and download time.
- The final decision-making stage involves data prioritization, e.g. some objects within the content are defined more important than others.

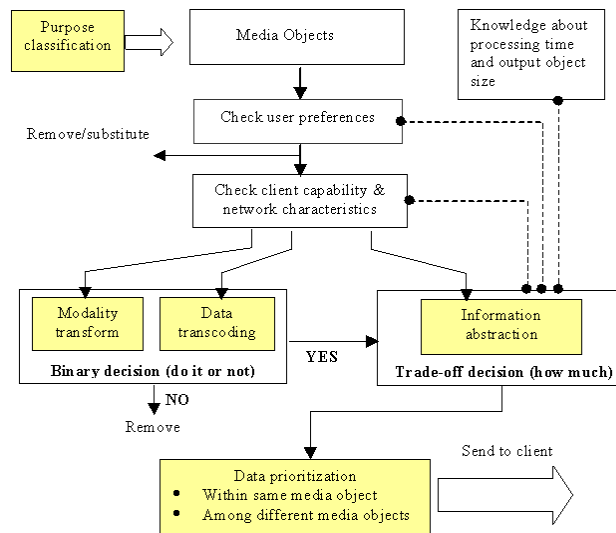


Figure 2.10 – HP decision engine architecture [2].

Figure 2.10 shows how the adaptation module (yellow boxes) and the decision engine are interconnected. The yellow parts in the diagram highlight the different adaptation techniques described above.

The **user/client/network monitoring** module uses heuristic algorithms to look for the pretended characteristics. This system obtains information about the terminal capabilities from the user HTTP (Hyper Text Transport Protocol [22]) request message. For example, if the HTTP request sent by the

terminal describes its browser as “Mozilla/1.22 (compatible; browser 1.0; PalmOS 2.0)” one can know that the terminal screen size has very likely 160x160 pixels because that is the typical screen size on a Palm. This way the system may not need to receive a user environment description even though the information present in the request message is very limited. However this solution seems to be a rather unreliable way of discovering the terminal capabilities. A terminal that uses the same browser (e.g. “Mozilla/1.22 (compatible; browser 1.0; PalmOS 2.0)”) may have different characteristics (e.g. display resolution, colors). Because of this, there is a need for a more efficient way to discover the client capabilities and user preferences, namely the W3C CC/PP standard described in Chapter 4.

The **user/client/network monitoring** module, implements two methods for measuring network bandwidth. The first method calls the TCP/IP protocol stack to know which bandwidth is available. The results from this method result in low bandwidth values at the beginning of the connection but afterwards the returned values tend to be the right values. The second method consists in dividing the size of a message by the measured time that takes to complete the transmission of that message to the terminal. Due to TCP/IP buffering, this method tends to overestimate the network bandwidth; nevertheless the returned values are typically more correct when transmitting large data files.

The user preferences are extracted analyzing the user browsing behavior. An example presented in [2] regards the monitoring of the connection’s living time; if the connections keep ending too soon (before the content has been completely sent), this may indicate that the user is impatient (or the bandwidth is low), and then the adaptation system should adapt more aggressively in order to reduce the response time.

In the same way the system does not need to receive an explicit description of the user environment, it also may not require a content description to be available. For the case of composite content, the absence of content descriptions poses a critical problem to this system because each content object must be retrieved and analyzed before deciding the customization to perform. HP implementation provides an excellent example of how content descriptions are essential when dealing with composite content if a reasonable level of complexity is to be achieved.

Figure 2.11 illustrates HP system results when customizing a Web page to HP palmtop. In the left side, a five-pages-long Web page before content adaptation is delivered to the hand-held-computer while in the right side the page has been reduced to two and a half pages after content adaptation. As can be seen, some banner images on the top of the page were removed.

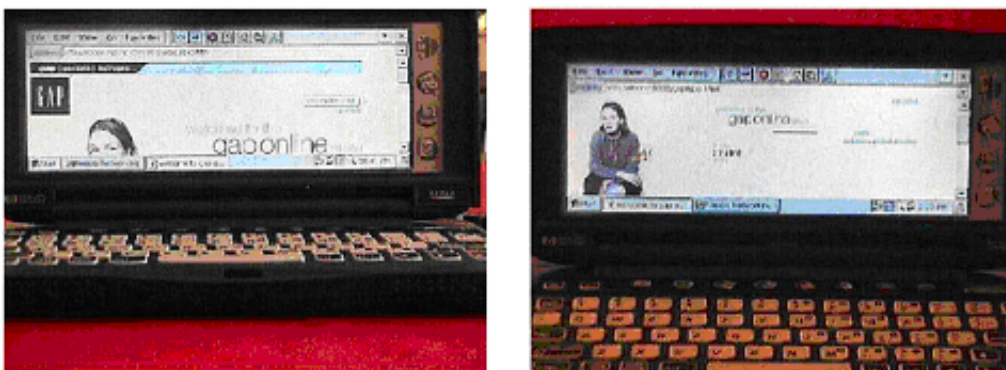


Figure 2.11 – Adaptive content delivery to a HP 620 palmtop [2].

The HP system implements a few content adaptation techniques that mainly focus on the re-authoring of the Web content through transcoding (color component reduction, spatial resolution reduction, compression), video key-frame extraction and HTML re-authoring.

In [2], a framework for adaptive content delivery is presented. The authors proposed discovery methods for detecting the user terminal, user preferences and network characteristics, several techniques to adapt the content and a decision engine to optimize the customized content value to the user environment. An interesting part of this work concerns the **Web Platform API** module, which allows the HP UMA Engine to be used in several situations by other services; in order to provide customized content, these services need only to communicate with the HP UMA Engine through its API.

2.2.3 IBM versus HP Content Customization Solutions

The IBM and HP experiments have a very similar background and the same objective – to deliver any content to any terminal with the best content value. Although the core structure is rather similar, when it comes to implementation issues and to the decision algorithms they differ quite a lot.

Although different, both systems use a architecture with four major modules:

1. **Terminal characteristics discovery:** both systems use heuristic mechanisms to discover the user terminal capabilities. However, through several contributions from IBM to the CC/PP and WAP standards, and analyzing the IBM WebSphere server product [72], it is possible to conclude that the IBM system supports now the WAP terminal description that is today the best available user environment description framework.
2. **Content features extraction:** the IBM system uses a Web server where the content is always available; using off-line processing, the system extracts the content description that is then used to populate the InfoPyramid with several variations created off-line. In the HP solution, there is no separate module targeting content feature extraction since this task is embedded in the adaptation module. Since the HP system extracts the content description in real-time, it has rather limited analysis capabilities. As for the CC/PP standardization efforts, IBM had a very important participation in the MPEG-7 standard development process. As a consequence, it is believed that, in the near future, IBM will support this content description standard in its WebSphere server product line.
3. **Content customization decision:** because InfoPyramid is an efficient way to manage the cached variations, it becomes a precious help to the IBM decision module which has to determine which content variation is delivered to the user. The HP system modular structure facilitates the inclusion of new adaptation algorithms and the consequent modification on the decision strategies.
4. **Content customization processing:** while HP customization is done on the fly, IBM customization is done off-line and thus only variation selection is performed at the time the content requests arrive. More recently [82], IBM also presented different architectures (e.g. proxy-based), which implement real-time content description extraction and customization.

Regarding user environment and multimedia content description, the IBM WebSphere server already uses the CC/PP standard and MPEG-7 will be eventually included soon. These two standards have been developed also with UMA applications in mind what makes them the most adequate solutions to the description problems at hand.

In both solutions (IBM and HP), the content customization modules implement typical compression, summarization, and key-frames extraction techniques that can be used to adapt multimedia content. Content selection using semantic techniques, e.g. to avoid receiving a banner, is also a content customization technique used by both systems.

In the UMA concept, content adaptation is comparable to compressing multimedia content to meet resource constraints imposed by the user environment. However, unlike traditional compression, composite multimedia content is considered and the constraints are not limited to bits or bandwidth, but also include other resources such as screen size, color, or also other dimensions such as user location, in short, the user environment characteristics. UMA efficiency resides in the adaptation techniques and in the decision algorithm that selects the different adaptation techniques to apply to the multimedia content and in that way produce the most valuable multimedia content variation that a specific user can receive.

2.3 Requirements

The objective of Universal Multimedia Access systems is to create different presentations (depending on the user environment conditions) of the same information, from any content-base. The task is to provide the best experience of the same information through an appropriately defined content variation. A typical example is the consumer who receives the same news whether it is through television, newspaper or the Internet. What is sought in this thesis is the capability to create a variety of presentations (content variations) in such way that all these should be derived from the same base content (composition of several multimedia objects), based on the content and user environment descriptions available to provide the best experience to the users.

The applications and experiments described helped in the definition of the UMA requirements and in the design of the UMA System. A list of requirements drawn from the presented applications and related works, organized into four categories, is presented in the following:

Requirements on content description tools:

1. **Media consumption resources:** descriptors on the terminal and network resources needed to consume the concerned content must be supported, e.g. data size, streaming bandwidth, color depth, screen size, number of audio channels, coding format, etc.
2. **Content structure:** descriptors to indicate the structure of the content, (e.g. scene cuts, keywords, points of views, object relations), which allow the creation of important variations of the same content source must be supported.
3. **Content adaptation hints:** descriptors allowing to indicate which are the most relevant parts of the content (spatial or temporal parts) and descriptors on the encoding characteristics/decisions of the described content (for transcoding adaptations) must be supported. These descriptors are important because they give hints to adapt the content knowing which parts are more relevant (e.g. region of interest) and thus preserving the quality on those parts to maximize the final subjective impact.
4. **Content adaptation priorities:** descriptors allowing to suggest which and with which order content customization techniques should be applied to a particular multimedia content object within a composite multimedia item must be supported. These descriptors may be instantiated by the content author, e.g. indicating if it is better to summarize the content, convert it to black/white, or spatially down-sample it.
5. **Content object importance:** descriptors allowing indicating the level of importance of the various object or zones within a composite multimedia item must be supported. This enables the UMA Engine to adapt each content object with different levels of quality according to its importance and the resources available at the user terminal.

6. **Content variation management:** descriptors to allow the management of different content variations must be supported. For example, if a text object has been converted into an audio object using text-to-speech translation, making an audio variation of the text object available, descriptors are needed to indicate its availability and relation to the original object.

Requirements on user environment description tools:

1. **Terminal hardware:** descriptors to characterize the terminal hardware capabilities must be supported, e.g. display size, display color depth, display frame rate capability, audio capabilities, storage space, processing power.
2. **Terminal software:** descriptors to characterize the terminal software capabilities must be supported, e.g. content decoding capabilities and supported content formats.
3. **Network characteristics:** descriptors to characterize the network capabilities must be supported. These descriptors allow describing network characteristics such as network access type (e.g. wireless or phone line), traffic parameters (peak bit rate, minimum bit rate, sustainable bit rate and maximum burst size) and QoS parameters (maximum transfer delay, delay variation and SNR)
4. **Natural environment:** descriptors to characterize the natural environment must be supported, e.g. location, surrounding noise, temperature.
5. **User preferences:** descriptors to characterize the user preferences must exist, e.g. desired confidentiality, user preferred content, demographics (although there are privacy issues related to this last type of feature).

Requirements on content customization algorithms:

1. **Transcoding:** content customization algorithms that can convert from one coding format to another must be supported.
2. **Modality conversion:** content customization algorithms should be able to create variations of a certain media type to a different media type (e.g. a key frame or a text summary from a video).
3. **Quality reduction:** content customization algorithms should be able to create lower quality variations of the source content (e.g. color reduction, spatial resolution reduction).
4. **Summarization:** the UMA Engine should be able to use summary descriptions to build video summaries on-the-fly. Content customization algorithms must be able to produce content variations that are “summaries” of the original content.
5. **Variation management:** content customization algorithms should be able to analyze multiple variations of the same content and select the best variation that serves each user.

Requirements on UMA Engine architecture:

1. **Caching previous customizations:** a UMA Engine should be able to cache and manage previous content customizations so that some variations may be reused in subsequent customizations and that way avoid the repetition of already performed tasks.
2. **Multimedia presentation and synchronization:** a UMA Engine should be able to process composite multimedia content. In composite multimedia content (such as MPEG-4 content), it may be necessary to process several objects before sending any content to the user. When generating a new composite multimedia content item, it is necessary to synchronize each multimedia content object and to reorganize the structure of the composite multimedia piece.
3. **User management:** the UMA Engine should be able to cache internally the user environment description in order to avoid multiple retrievals of the same description.
4. **Real-time processing:** the UMA Engine must be able to handle multiple requests for content customization and dispatch them with an acceptable delay to the user; this means processing multimedia content and user environment descriptions and customizing the content in real-time (eventually with some delay before start streaming but after the streaming has begun with no additional delays or congestions).
5. **Scalability (number of requests):** for some types of services, the UMA Engine must be able to process the customization requests as fast as possible (not necessarily real-time); in this case, the priority is to support the largest number of users requests (possible because for some services the user is more tolerant to longer delays). Thus the system must use a scaling policy that favors the number of served requests in opposition to the smallest response time.
6. **Scalability (response time):** for some types of services, the UMA Engine must be able to process the customization requests as fast as possible. Thus the system must use a scaling policy that favors the response time (eventually approaching real-time) in opposition to the number of serving users. This requirement and the previous one cannot be simultaneously used, so the system must also be capable of using each policy according to its configuration.

The two first requirements' categories are partially satisfied by existing standards as will be seen in Chapter 3 and Chapter 4, respectively. Because there are requirements that are more targeted to specific application domains, the requirements above do not necessarily have to be relevant all simultaneously (although most of them are since they are rather generic). Therefore it is very likely that different UMA Engines may give more emphasis to some of the requirements than others. The content customization algorithms and the UMA Engine architecture requirements will be as much as possible fulfilled by the UMA System architecture that will be presented in Chapter 5 and by the decision and customization algorithms to be presented in Chapter 6.

Chapter 3

Multimedia Content Description

In Chapter 1, the content in a multimedia document was grouped into several categories: text, image, natural video, synthetic video, natural audio, synthetic audio, natural speech, synthetic speech, interactive elements and structural elements. All these data types must be processed by the UMA Engine, and in order to do that, it is required that all the objects in the content have appropriate descriptions. As can be easily understood, the tools to describe each media type may have to be very different and related to the more adequate features to characterize that media type, e.g. color for images and pitch for speech.

This chapter will consider the multimedia content description problem in a UMA scenario and will present the solution that is used in this thesis to describe multimedia content.

3.1 Describing Multimedia Content

Describing content is the act of gathering special characteristics about the content that may be used in a later process to more easily retrieve, filter or adapt the content in question. Thus a content description is intended to facilitate the access to the pretended content. Descriptions focusing on retrieval and filtering applications are quite different from descriptions targeting content adaptation applications. The content descriptions, which may help universal access systems, focus on the content structure and encoding characteristics: encoding parameters, region of interests, content object importance, video transcoding hints, and video summarization. Descriptions targeting universal access must provide information that enabling the content rendering in any user environment, eventually with transformations in terms of resolution, quality or even modality.

In terms of the difficulty/complexity of the adaptation decision algorithms, single data type content is the easiest to be processed in a UMA scenario. Composite (data types) content and its corresponding structure is in fact the most problematic content that a UMA Engine must deal with. The problem does not reside in the resources required to process the content but in the lack of information about the relations among each content element in the multimedia document (e.g. priorities on the content element disposition in the overall document). Interactive elements are generally embedded within the document that is consumed by the content receiver; therefore the adaptation of these elements involves the knowledge and evaluation of additional characteristics of the user environment such as the input capabilities of the terminal (e.g. keyboard type, mouse).

There are several frameworks to describe content in order to enable more efficient search and filtering of content (through indexed descriptions). In W3C [31] there are several projects addressing this topic; the Dublin Core [9] is another description framework; and also SMPTE [8], DVB [7], IETF and MPEG [5] worked or are working on this type of technology. Each of these organizations is focusing on their natural

technology or application environment, e.g. W3C is targeting Web applications and MPEG audiovisual content.

While some of these frameworks specify a syntax to describe content but do not specify which features (semantics) can be described (e.g. IETF), others specify only some high level features (e.g. Dublin Core, SMPTE and DVB) and finally others are still at the very beginning of solving the process (W3C). MPEG has made a great effort to cover different industry technical requirements in terms of content descriptions (from low level features to high level features) and also very different application domains (TV broadcasters, Internet streaming, mobile applications). As a result of this effort, the MPEG-7 standard is today the most complete, powerful, and stable framework to describe multimedia content and thus will be the solution adopted for this thesis.

3.2 MPEG-7 Multimedia Content Description Interface

MPEG-7 is a new standard [83] developed by MPEG (Moving Picture Experts Group, [5]), the committee that also developed the standards known as MPEG-1 (1991), MPEG-2 (1995) and MPEG-4 (version 1 in 1998 and version 2 in 1999). MPEG-7 addresses the multimedia content description problem at very different levels: it offers a wide range of description tools that range from low-level features such as color and pitch to high-level features such as the name of the characters in a scene. One of the MPEG-7 target applications is universal access to multimedia content, which makes MPEG-7 particularly interesting for the problem addressed in this thesis.

MPEG-7 aims at offering a comprehensive set of audiovisual description tools to create descriptions, which will form the basis for applications enabling the needed quality of access to content, which implies adequate database solutions, high-performance content identification, intellectual property management and fast, accurate and personalized filtering, searching and retrieval.

3.2.1 Context and Objectives

The multimedia content scenario presented in Chapter 1 brings several challenges: the vast amounts of information being nowadays generated make it very difficult to manage the information in an efficient way; moreover the access to information in useful time (searching or filtering information) and the easy access to audiovisual content when in constrained environments (limited bandwidth or computing resources) also need a solution.

Indexing applications must be able to analyze audiovisual content and extract key characteristics that can be encoded using the MPEG-7 format (the description). A feature in the MPEG-7 context is a key characteristic that contains relevant information regarding the analyzed/described content. Applications will extract the features through content analysis by three ways:

- **Automatically:** the system can extract the feature directly in an automatic and objective way;
- **Semi-automatically:** the extraction process is automatically performed and complemented/corrected with the help of a human;
- **Manually:** a human manually instantiates the feature value in the description.

There are two major types of features: low-level features and high-level features. Low-level features are easily extracted by automatic methods while high-level features require complex analysis algorithms or are most likely impossible to extract automatically due to the semantic dimension they involve. Figure 3.1 illustrates a combined process of high-level and low-level feature extraction. This Figure shows the way that low-level features are used to derive high-level features: through a step-by-step analysis; at each step,

the analysis abstracts high-level features from lower-level features of a previous analysis step. Using a feedback chain, the previously acquired knowledge is used to enable the improvement/tuning of the extraction results.

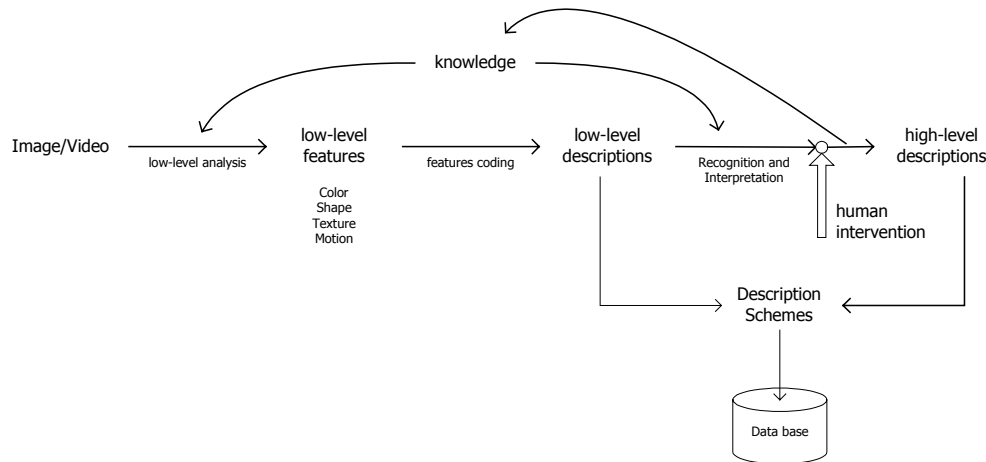


Figure 3.1 – Sequence of the steps involved in the creation of content descriptions.

High-level features can be fully derived by human intervention, or derived just under human supervision while low-level features can be automatically extracted, sometimes even in real-time. Therefore, MPEG-7 description tools allow creating descriptions of content including low-level and high-level description components such as:

- Information on the content media features (storage format, encoding);
- Low-level characteristics such as color, texture, sound timbre, melody;
- Structural information on spatial, temporal or spatial-temporal components of the content such as scene cuts, segmentation in regions, region motion tracking;
- Conceptual information regarding the reality captured by the content such as definition of objects and events, interactions among objects;
- Information related to the usage of the content such as copyright pointers, usage history, broadcast schedule;
- Information related to the content creation and production processes such as director name, title, date;

MPEG-7 provides a set of description tools intended to characterize the audiovisual content in terms of the type of features listed above. These description tools are independent of the content format; in fact, the content described can be in any format: analog, PCM, MPEG-1, -2 or -4 or any other format. The standard separates the descriptions from the content but provides linking mechanisms between the content and the descriptions since there may exist more than one description for the same content; these links must work in both directions.

3.2.2 Scope

Previous MPEG standards were defined in such a way that the minimum number of tools was normatively specified in order to leave enough space for competition without preventing interoperability.

MPEG-7 also follows this approach and only standardizes the description format (syntax and semantics): applications are responsible for the feature extraction and consumption mechanisms where no normative rules are to be followed. Figure 3.2 shows a simplified block diagram of the MPEG-7 application chain. This chain includes feature extraction (description creation), the description itself (eventually to be delivered), and the search engine (description consumption).

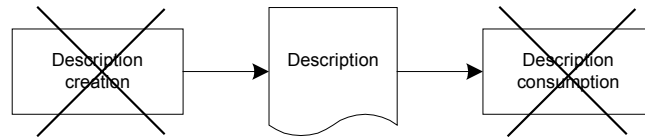


Figure 3.2 – Scope of MPEG-7.

To fully exploit the possibilities of MPEG-7 description tools, automatic extraction of features (or descriptions) will be extremely useful. It is also clear that automatic extraction is not always possible. The higher the level of abstraction, the more difficult automatic extraction is, and interactive extraction tools allowing to input human guidance in the process will be of good use. However, independently of how useful they are, neither automatic nor semi-automatic feature extraction algorithms are in the scope of the standard. The main reason is that their standardization is not required to allow interoperability, while leaving them free, creates space for industry competition and technical evolution.

The types of description tools specified by the MPEG-7 standard [85] are:

- **Descriptors (D):** represent a feature, and define the syntax and semantics of the feature representation. Possible descriptors are: a color histogram, the average of the frequency components, a motion field, the text of the title, etc.
- **Description Schemes (DS):** specify the structure and semantics of the relationships between their components, which may be both Descriptors, and Description Schemes. Possible Description Schemes are: a movie, temporally structured as scenes and shots, including some textual descriptors at the scene level, and color, motion and audio descriptors at the shot level.
- **Description Definition Language (DDL):** allows the creation of new Description Schemes and possibly Descriptors, as well as the extension of existing Description Schemes.
- **Systems tools:** support the multiplexing of descriptions, synchronization of descriptions with the associated content, binary representation for efficient storage and transmission, management and protection of intellectual property, etc.

To provide a better understanding of the terminology introduced above, Figure 3.3 illustrates their use. Figure 3.3 illustrates the way Descriptors and Description Schemes are organized; a root element starts the tree describing a certain piece of multimedia content. Below the root element, there can be several Descriptors and Description Schemes characterizing the content. By defining only a minimum set of restrictions, MPEG-7 leaves a huge space for new applications to explore these new possibilities in terms of audiovisual description. Even though MPEG-7 supplies a very complete list of descriptors, new descriptors are already under study to be included in future versions of MPEG-7 in order to provide additional description functionalities.

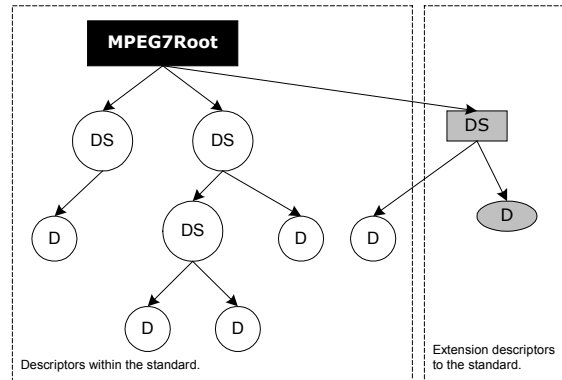


Figure 3.3 – Graphical view of the relation between the different MPEG-7 description elements.

Figure 3.4 includes a block diagram showing how descriptions are generated and consumed. After extracting or human annotating the features, they are coded in the MPEG-7 format so that they can be consumed later by any MPEG-7 enabled application. This chain can be described in the following way:

- Content is analyzed and features are extracted (1) to generate a content description (2).
- Using an MPEG-7 enabled framework, the content description is coded using the MPEG-7 standard format (3).
- If desired, the description is linked to the content it describes (4).
- Again using an MPEG-7 enabled framework, an application decodes the MPEG-7 standard content description (5).
- An application uses the MPEG-7 content description to easily retrieve, filter, manage, etc. the content (6).

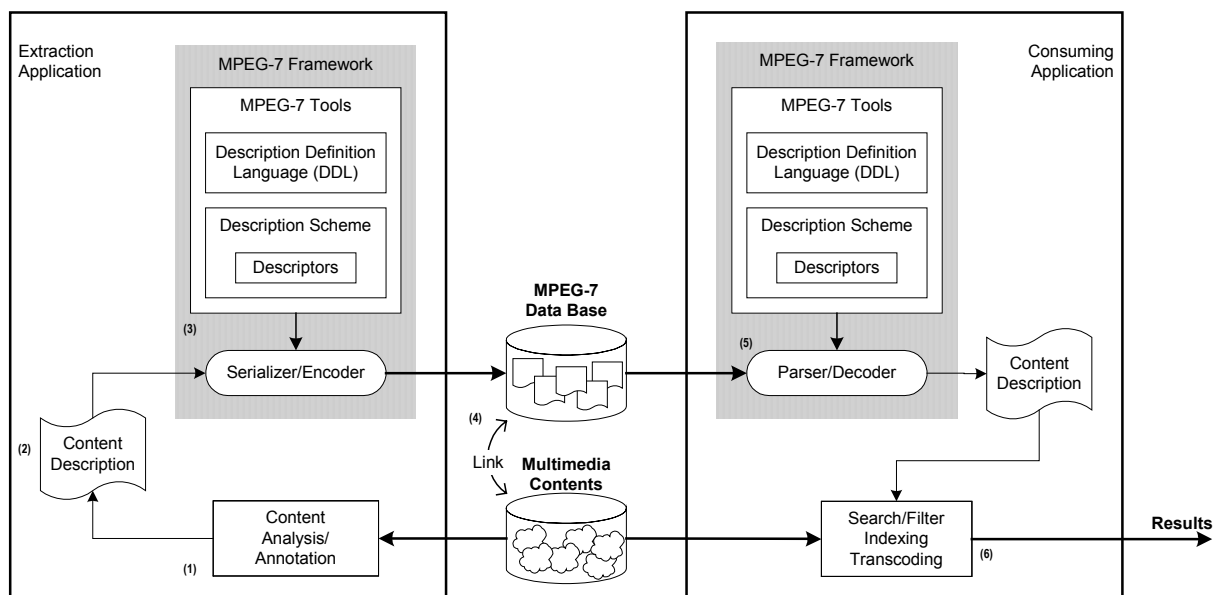


Figure 3.4 – Abstract representation of applications using MPEG-7.

The features are extracted or generated by analyzing the content. The description is associated with the content itself, in order to allow fast access to content. MPEG-7 descriptions do not need to be located in the same place as the content (descriptions are independent of the content).

3.2.3 Applications

MPEG-7 enabled applications can be divided into three categories:

- **Pull applications:** applications where the user wants to find information that may reside in databases or in information servers and actively looks for this information.
- **Push applications:** applications such as in broadcast environments where the user needs to filter from all the content he receives, the content that he effectively wants to consume.
- **Content adaptation applications:** applications where the content is customized to the conditions that the user is using to access to the information (e.g. limited bandwidth).

An application may create and/or consume content descriptions and encode them in the MPEG-7 standard format so that it may interoperate with other applications. The specific way MPEG-7 data will be used to answer user queries is outside the scope of the standard. Any type of audiovisual material may be retrieved by means of any type of query material (query by example). This means, for example, that video material may be queried using video, music, speech, etc. It is to the search engine to match the query data to the MPEG-7 audiovisual descriptions.

3.2.4 MPEG-7 Standard Organization

The MPEG-7 standard is organized into the following parts:

- **Part 1: Systems:** includes the tools that are needed to prepare MPEG-7 descriptions for efficient transport and storage (binarization) and to allow synchronization between content and descriptions as well as the tools related to managing and protecting intellectual property [86]
- **Part 2: Description Definition Language:** includes the language for defining new Description Schemes and eventually also new Descriptors [87].
- **Part 3: Visual:** includes the Descriptors and Description Schemes dealing only with visual data [88].
- **Part 4: Audio:** includes the Descriptors and Description Schemes dealing only with audio data [89].
- **Part 5: Multimedia Description Schemes:** includes the Descriptors and Description Schemes dealing with generic features and multimedia data [90].
- **Part 6: Reference Software:** includes a software implementation of the tools specified by the first five parts of the MPEG-7 Standard [91].
- **Part 7: Conformance Testing:** includes guidelines and procedures for testing the conformance of MPEG-7 implementations and streams.

The MPEG-7 Reference Software is a compilation of all software used in the MPEG core experiments² to develop and validate the MPEG-7 standard. MPEG-7 Conformance Testing is still in a very initial phase. In the following sections, the first five parts of the standard will be briefly introduced since these are the parts where the MPEG-7 technology is defined.

3.2.4.1 MPEG-7 Systems

MPEG-7 systems architecture defines the basic structure through which MPEG-7 descriptions must be exchanged. Descriptions can be exchanged both in textual format using the DDL language specified in Part 2: DDL or in binary format using the compression tool specified in Part 1: Systems. There is a direct mapping between the textual and binary formats which is defined in Part 1: Systems. The terminal architecture, the access units and the normative interfaces are the main elements of MPEG-7 Systems:

- **Terminal architecture:** the architecture of a terminal making use of MPEG-7 descriptions is presented in Figure 3.5. Only the Delivery Layer and the Compression Layer are specified in the standard:
 - The **Delivery Layer** provides to the Compression Layer MPEG-7 elementary streams. MPEG-7 elementary streams consist in consecutive individually accessible portions of data named *Access Units* which contain information of different nature: schema (DS) information defining the structure of the MPEG-7 descriptions, and description information that is either the complete description of the multimedia content or fragments of that description.
 - At the **Compression Layer**, the flow of Access Units is parsed, and the content description is reconstructed. The standard does not mandate the reconstruction of a textual representation as an intermediate step of the decoding process.

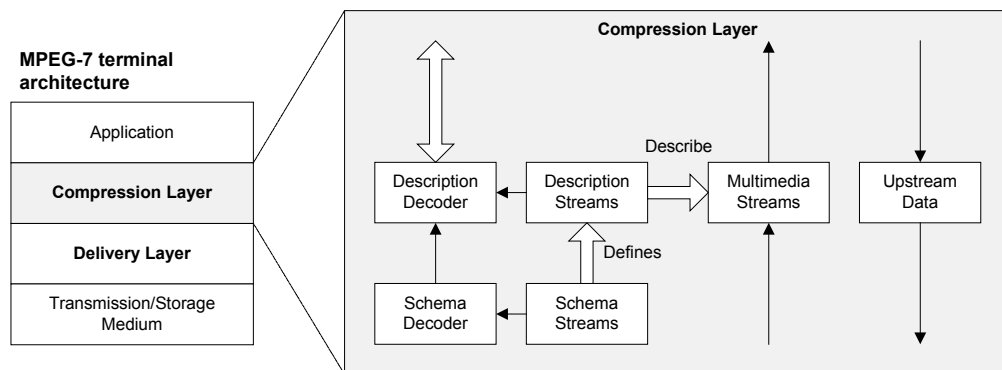


Figure 3.5 – MPEG-7 terminal architecture.

- **Access Units:** an MPEG-7 description has a tree organization where each branch can be fragmented into smaller trees. The Access Unit definition borrows from the generalization of this type of tree fragmentation (it corresponds to these description fragments, as in Figure 3.6). An Access Unit is an MPEG-7 systems data type to which timing information can be attached; two Access Units shall not carry the same timing information. This timing information can, for example, be used for

² A core experiment verifies the inclusion of a new technique or set of techniques in MPEG standards. In a core experiment there are multiple, independent, directly comparable experiments, performed to determine the validity or not of the proposed technique.

synchronization purposes between the transmitter and the receiver; also the transport layer packets usually carry this timing information.

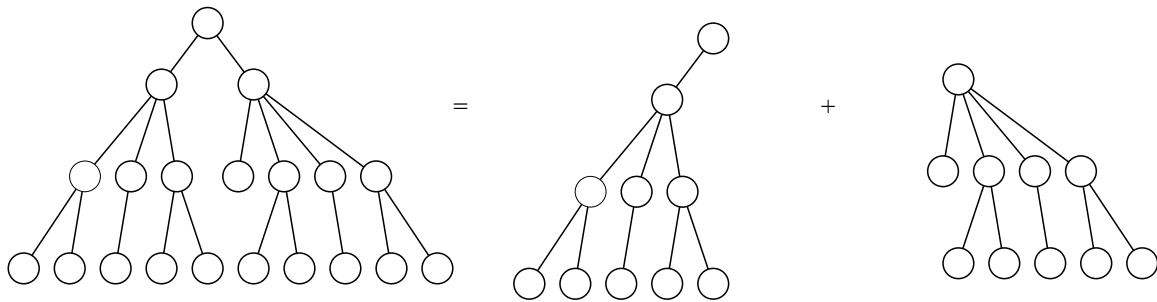


Figure 3.6 – Decomposition of a content description tree into two fragments [86].

- **Normative interfaces:** MPEG-7 has two normative interfaces, one textual and another binary, as illustrated by Figure 3.7. An MPEG-7 encoder transforms the content features into a description compliant with the standard. The textual/binary interfaces define the format of the textual/binary access units. The MPEG-7 decoder transforms MPEG-7 compliant data contained in access units into the application specific content description format.
 - **Textual Format interface:** this interface describes the format of the textual access units. The MPEG-7 Textual Decoder/Encoder (XML Schema) consumes/generates a flow of such Access Units.
 - **Binary Format Interface:** this interface describes the format of the binary access units. The MPEG-7 Binary Decoder/Encoder (**BiM**) consumes/generates a flow of such Access Units.

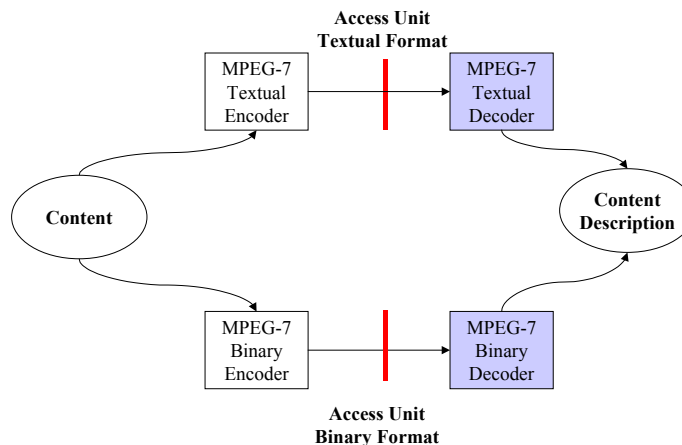


Figure 3.7 – MPEG-7 Normative Interfaces [86].

3.2.4.2 MPEG-7 Description Definition Language

The Description Definition Language forms a core part of the MPEG-7 standard since it provides the solid descriptive foundation by which standard Descriptors and Description Schemes are defined and users can create their own Description Schemes. The DDL is able to express spatial, temporal, structural and conceptual relationships between the elements of a DS, and between DSs. It provides a rich model for links and references between one or more descriptions and the data that they describe. In addition, it is platform and application independent and human-and machine-readable.

At the 51st MPEG Meeting in Noordwijkerhout (NL), in March 2000, it was decided to adopt the XML Schema Language as the MPEG-7 DDL. The complete specification of XML Schema can be found at <http://www.w3c.org>. However, since XML Schema is not a language specifically designed for audiovisual content description, certain extensions were necessary in order to satisfy all of the MPEG-7 DDL requirements [85].

3.2.4.2.1 XML Schema

XML Schema is a W3C recommendation allowing developers to precisely define the structures of their own XML-based formats. XML is a set of rules (such as guidelines and conventions) for designing text formats that let developers structure his own data. XML makes it easy for a computer to generate data, read data, and ensure that the data structure is unambiguous. XML Schema defines constraints and syntax of the XML data. The XML Schema recommendation is divided in three parts:

- **XML Schema Part 0: Primer:** non-normative document intended to provide an easily readable description of the XML Schema facilities; it is oriented towards quickly understanding how to create schemas using the XML Schema language.
- **XML Schema Part 1: Structures:** specifies the XML Schema definition language, which offers facilities for describing the structure and constraining the contents of XML 1.0 documents.
- **XML Schema Part 2: Datatypes:** specifies facilities for defining datatypes to be used in XML Schemas; the datatype language, which is itself represented in XML 1.0, provides a way for specifying datatypes on elements and attributes.

3.2.4.2.2 MPEG-7 Extensions to XML Schema

The following features have been added to the XML Schema Language specification in order to satisfy specific MPEG-7 requirements:

- Structural modifications to XML Schema has been made in order to allow the use of two new data types:
 - **Array and matrix data types:** both fixed size and parameterized size; using the *list* data type, two methods are provided for specifying sizes of (1D) arrays and multi-dimensional matrices.
- Two new data types were defined, which did not required structural modifications of the XML Schema:
 - **basicTimePoint:** specifies a time point according to the Gregorian dates, day time and the Time Zone (TZ); the format is based on the ISO 8601 standard.
 - **basicDuration:** specifies the duration of a time period according to days and time of day; the format is based on the ISO 8601 standard.

3.2.4.3 MPEG-7 Visual

The visual description tools included in MPEG-7 Part 3 consist of basic structures and descriptors that cover visual features in the terms of color, texture, shape and motion. In the following, only the most relevant descriptors are presented:

– **Basic structures:** the five basic structures used by the MPEG-7 visual descriptors are:

- **Spatial 2D coordinates:** defines a 2D spatial coordinate system to be used by reference in other Ds/DSs, when relevant. It supports two kinds of coordinate systems: “local” and “integrated” (see Figure 3.8). In a “local” coordinate system, the coordinates used for the calculation of the description are mapped to the current coordinate system applicable. In an “integrated” coordinate system, each image (frame) of e.g. a video sequence may be mapped to different areas with respect to the first frame of a shot or video.

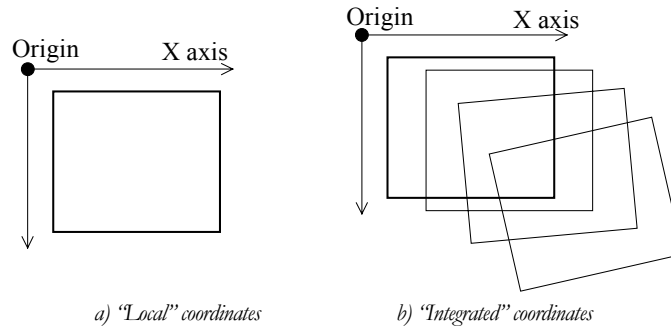


Figure 3.8 – "Local" and "integrated" coordinate system [88].

- **Temporal interpolation:** characterizes temporal interpolation using connected polynomials, to approximate multi-dimensional variable values that change with time, such as object position in a video sequence. In Figure 3.9, 25 real values are represented by five linear interpolation functions and two quadratic interpolation functions.

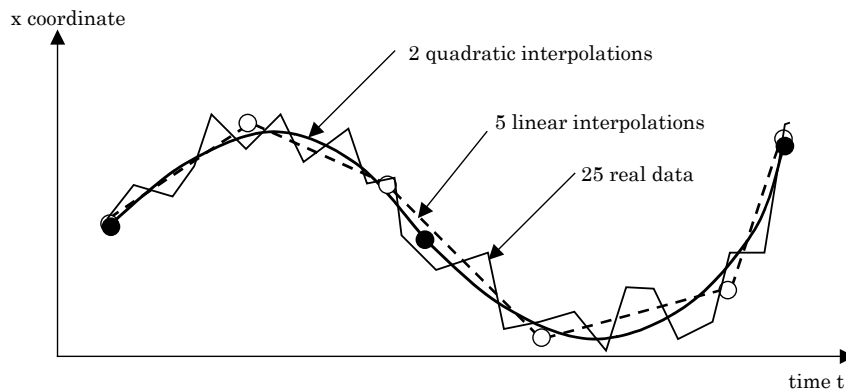


Figure 3.9 – Real data and interpolation functions [88].

- **Grid layout:** provides a splitting of the image into a set of rectangular regions, so that each region can be described separately.
 - **Time series:** describes a temporal series of descriptors in a video segment and provides image to video frame and video frame to video frame matching functionalities.
 - **Multiple view:** specifies a structure combining 2D descriptors representing a visual feature of a 3D object seen from different view angles.
- **Color Descriptors:** there are seven MPEG-7 color descriptors defined:

- **Color space:** defines the color space used in MPEG-7 color based descriptions; the following color spaces are supported: RGB, YCbCr, HSV, HMMD, linear transformation matrix with reference to RGB, monochrome.
 - **Color quantization:** defines the uniform quantization of a color space.
 - **Dominant color:** specifies a set of dominant colors in an arbitrarily shaped region.
 - **Scalable color:** defines a color histogram in the HSV color space, encoded by a Haar transform; its binary representation is scalable in terms of bin numbers and bit representation accuracy over a broad range of data rates.
 - **Color layout:** specifies the spatial distribution of colors for high-speed retrieval and browsing; it can be applied either to a whole image or to any part of an image, including arbitrarily shaped regions.
 - **Color structure:** captures both color content (similar to that of a color histogram) and the structure of this content via the use of a structuring element composed of several image samples.
 - **GoF/GoP color:** defines a structure for representing the color features of a collection of (similar) images or video frames by means of the scalable color descriptor defined above. The collection of video frames can be a contiguous video segment or a non-contiguous collection of similar video frames.
- **Texture Descriptors:** there are three MPEG-7 texture descriptors defined:
- **Homogeneous texture:** provides a precise quantitative description of a texture by representing the energy and energy deviation values extracted from a frequency layout.
 - **Texture browsing:** provides a perceptual characterization of a texture, similar to a human characterization, in terms of regularity, coarseness and directionality; it is useful for representing homogeneous texture for browsing type applications.
 - **Edge histogram:** represents the spatial distribution of five types of edges, namely four directional edges and one non-directional edge; since edges play an important role in image perception, it can be useful for retrieving images with similar semantic meaning. See Figure 3.10.

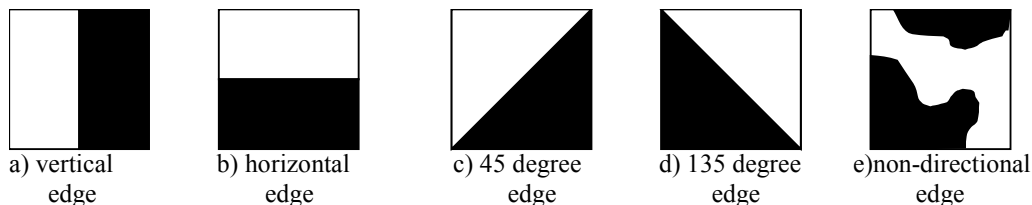


Figure 3.10 – Five types of edges for the edge histogram descriptor [88].

- **Shape Descriptors:** there are three shape descriptors:
- **Region-based:** makes use of all pixels constituting the shape and can describe any shape, i.e. not only a simple shape with a single connected region but also a complex

shape consisting of several disjoint regions (see Figure 3.11). The MPEG-7 region-based descriptor is based on the Angular-Radial Transform (ART).



Figure 3.11 – Examples of various shapes, [88].

- **Contour-based:** describes a closed contour of a 2D object or region in an image or video sequence. Figure 3.12 illustrates a contour based shape of a certain object. The contour-based descriptor is based on the Curvature Scale Space (CSS) algorithm.

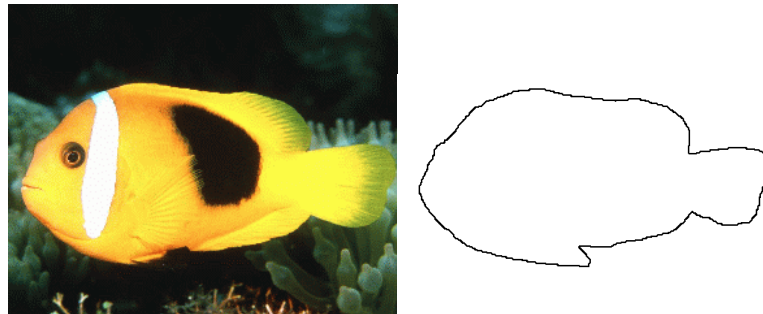


Figure 3.12 – A 2D visual object (region) and its corresponding shape, [88].

- **3D shape:** provides an intrinsic shape description of 3D mesh models, based on the histogram of 3D shape indexes representing local curvature properties of the 3D surface.
- **Motion Descriptors:** there are four MPEG-7 motion descriptors:
- **Camera motion:** characterizes 3D camera motion based on 3D camera motion parameter information, which can be extracted or generated by capture devices. Figure 3.13 illustrates the types of camera movements described by the camera motion descriptor.

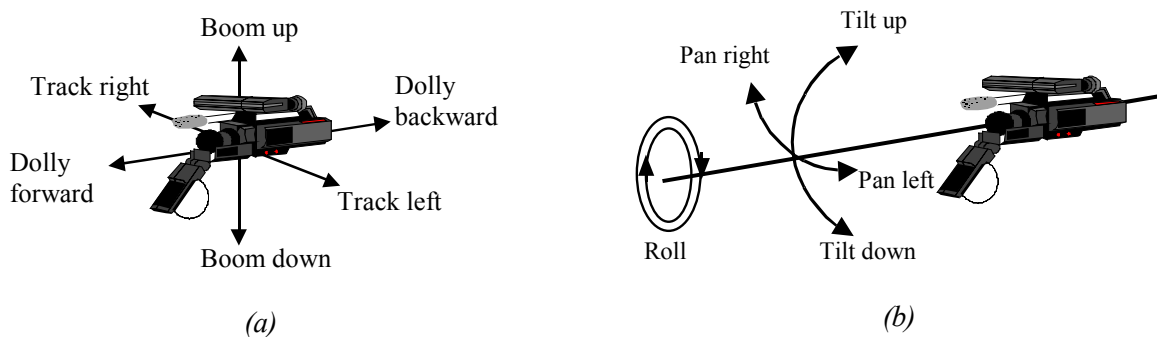


Figure 3.13 – (a) Camera track, boom, and dolly motion modes; (b) Camera pan, tilt and roll motion modes.

- **Motion trajectory:** defines the spatial-temporal localization of a representative point of a moving object (or region).

- **Parametric motion:** characterizes the evolution of arbitrary shaped regions over time in terms of a 2D geometric transform.
- **Motion activity:** captures the intuitive notion of “intensity of action” or “pace of action” in a video segment.
- **Localization:** there are two MPEG-7 localization descriptors defined:
 - **Region locator:** describes the localization of regions within images or frames by specifying them with a brief and scalable representation of a box or a polygon.
 - **Spatial temporal locator:** describes spatial-temporal regions in a video sequence, such as moving object regions, providing a localization functionality.

These are the main descriptors defined in the Visual Part of MPEG-7; for the other description tools refer to Part 3 of the MPEG-7 standard [88].

3.2.4.4 MPEG-7 Audio

This part of the MPEG-7 standard (Part 4: Audio) [89] specifies description tools for audio data. The audio description tools are applicable to all forms of audio: music, speech, sound effects, soundtracks, and any mixtures of these. The MPEG-7 Audio standard considers two major parts:

- **Audio Framework:** collection of tools and low-level descriptors intended as a framework for construction of many applications.
 - **Scalable Series:** efficient representation for series of feature values.
 - **Low-level audio descriptors:** collection of low-level audio descriptors, many built upon the Scalable Series tool.
 - **Silence:** descriptor for silence.
- **High-level Tools:** set of tools more geared towards more specific applications; these tools may make use of the Audio framework tools listed above.
 - **Timbre Description:** collection of description schemes addressing the perceptual features of instrument sounds.
 - **Sound Recognition:** collection of descriptors and description schemes defining a general mechanism suitable for handling sound effects.
 - **Spoken Content:** set of description schemes representing the output of an Automatic Speech Recognition (ASR) system.
 - **Melody Contour:** description scheme allowing retrieval of musical data.
 - **Melody:** more general description framework for melody.

As already said in Chapter 1, this part of the MPEG-7 standard will not be used in this Thesis since the UMA Engine implemented processes only visual information.

3.2.4.5 MPEG-7 Multimedia Description Schemes

The MPEG-7 DSs expand on the MPEG-7 Descriptors by combining individual Descriptors as well as other DSs within more complex structures and by defining the relationships among the component Descriptors and DSs. Typically, the multimedia description schemes refer to all kinds of multimedia consisting of audio, visual and textual data, whereas the domain specific Descriptors or DSs refer specifically to the audio or visual domains. The Multimedia Description Schemes (MDS) in Part 5 [90] of the MPEG-7 standard are organized in the areas shown in Figure 3.14.

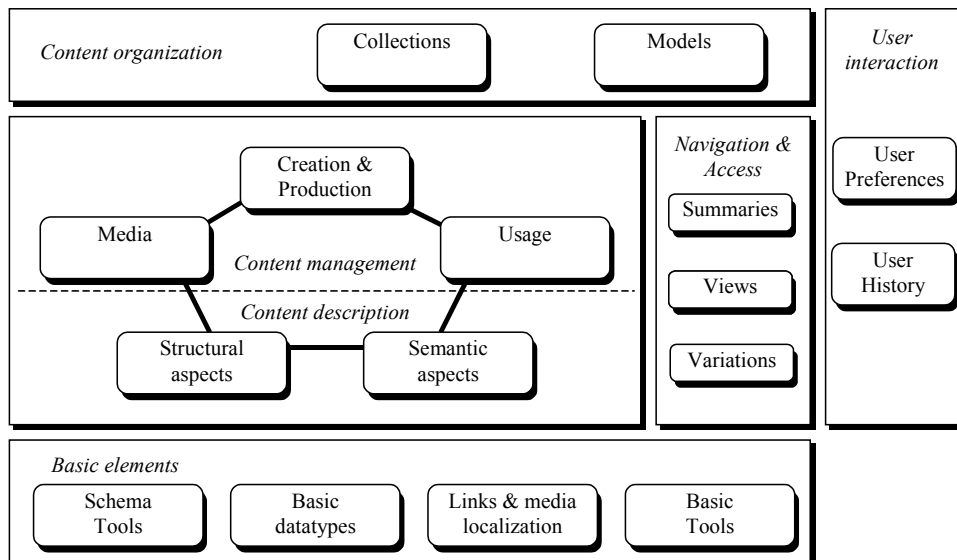


Figure 3.14 – Overview of the MPEG-7 MDS tools [90].

- **Basic Elements:** include fundamental constructs which are used repeatedly throughout the definition of the MPEG-7 DSs. Many of the basic elements provide specific data types and mathematical structures, such as vectors, matrices and histograms, which are important for audiovisual content description. Schema tools are intended to facilitate the creation and packaging of MPEG-7 descriptions.
- **Content Description:** describe the structure (regions, video objects, frames and audio segments) and semantics (objects, events, abstract notions) of audiovisual content.
- **Content Management:** describe different aspects of creation and production, media coding, storage and file formats and content usage.
- **Content Organization:** organize and model collections of audiovisual content.
- **Navigation and Access:** facilitate browsing and retrieval of audiovisual content by defining summaries, partitions, decompositions, and variations of audiovisual material.
- **User Interaction:** describe user preferences pertaining to the consumption of the multimedia material.

The most important MPEG-7 Multimedia Description Schemes for the purpose of data customization applications are included in the Navigation and Access group which will be presented with more detailed in the next section.

3.3 MPEG-7 and UMA applications

This section will present the MPEG-7 description tools that are important for UMA applications. Many of the features relevant for UMA applications can be automatically extracted while other features must be manually introduced/annotated in the content description (such as the name of an actor in a movie). A feature extraction system may be controlled/tuned to extract the most relevant set of features for the target application that will consume the description and for the content that is being analyzed (for example, a football match).

In this thesis, only the visual media types are considered in more detail in terms of content description (image and video). MPEG-7 provides description tools that will be used by the UMA Engine to facilitate the content customization process. Table 3.1 presents a non-exhaustive list of content adaptation operations that a UMA Engine may perform to the basic media types. The adaptation alternatives in Table 3.1 are classified according to the concerned data type and the type of operation in question.

Table 3.1 – Types of content customization.

	Video	Image	Audio	Text
Summarization	Key-frame extraction Shots selection based on some criteria Audio extraction Video-to-text Removal (if nothing better may be done)	Image-to-text Removal (if nothing better may be done)	Audio summarization Audio-to-text Removal (if nothing better may be done)	Text summarization Text-to-audio Table-to-list Removal (if nothing better may be done)
Quality reduction	Video frame rate reduction Video spatial resolution reduction Quantization step reduction Color-depth reduction	Image spatial resolution reduction Color-depth reduction Data size reduction (by increasing compression rate)	Audio sub-sampling Stereo-to-mono conversion Quantization step reduction	Outlining Font size reduction Text white space removal
Transcoding	Format conversion	Format conversion	Format conversion	Format conversion

The content adaptation solutions are divided into three major categories:

- **Summarization:** this type of customization involves the temporal reduction of a certain audiovisual material and/or modality conversion (e.g., video to image). It creates a smaller variation (in terms of temporal duration) of the source content by removing temporal segments that are less relevant according to some criteria (e.g. user preferences). These customizations may involve also modality conversion, (e.g. using only the audio of a news TV program or key-frames from a video). Modality conversion is also used when a terminal is not capable of handling certain media types (e.g. video) but it is capable of handling other media types (e.g. audio or images).
- **Quality reduction:** in this type of adaptations, content quality is reduced through transformations directly applied at the bitstream level (e.g. spatial, temporal, or frequency transformations). A temporal transformation may imply for video a reduction on the number

of video frames per second, for example changing from 25 fps to 10 fps; for audio, it may imply a reduction on the number of samples per second. A spatial transformation may imply a reduction on the number of color components or on the spatial resolution. A frequency transformation may imply the increase of the compression ratio, e.g. by removing the higher frequency DCT coefficients. Other bitstream transformations may fall under this category (e.g. quantization step increase).

- **Transcoding:** involves the conversion of the original coding format to another coding format (e.g. JPEG to GIF).

The content descriptors which are more often needed to perform the adaptations listed above are all related to content encoding parameters, e.g. format, image pixel width and height, video frame-rate, etc. However, more complex descriptors may be needed, for example to guide the UMA Engine on how to do modality conversions (e.g. region of interests, a text summary of a video, key frames).

MPEG-7 MDS basic elements (Schema tools, basic data types, links and media localization) provide a foundation for the development of the description tools; however, because the presentation of this part would be very extensive, it is not presented in this thesis. MPEG-7 descriptors and description schemes that can be more relevant to help the UMA Engine in the content selection/adaptation process are grouped into three categories:

- **Media description tools:** description of the media; typical features include the storage format, the encoding of the multimedia content, and the identification of the media.
- **Content structure description tools:** describe the audiovisual content from the viewpoint of its structure; the content is organized in segments that represent its spatial, temporal or spatial-temporal structure.
- **Content navigation and access:** tools that facilitate the browsing and retrieval of audiovisual content and the management of different versions of the same content.

The description tools for UMA applications presented next, are already part of the standard. Meanwhile other tools are under consideration by the MPEG Committee. One of the tools that is not yet part of the MPEG-7 standard but is under development for inclusion in MPEG-2 Version 2 is the color temperature descriptor that will be presented in Chapter 6.

The presentation of the MPEG-7 description tools will cover the most relevant aspects and concepts; however not every attribute of each description tool is detailed in this thesis. For a complete specification the reader may refer to the MPEG-7 Part 5 document [90].

3.3.1 Media Description Tools

Audiovisual content described using MPEG-7 description tools can be available in different modalities, and coding formats, and multiple instances (variations) of the source content may exist. For example, a concert may have been recorded in two different formats: audio only and audiovisual. This idea is illustrated in Figure 3.15.

Figure 3.15 illustrates a case where content is produced or recorded in different modalities, so each individual media profile exists to describe the different encoding features and storage media information. From the produced content (the source), several other content copies may be created, therefore creating different instances with different encoding profiles or formats of the same content source.

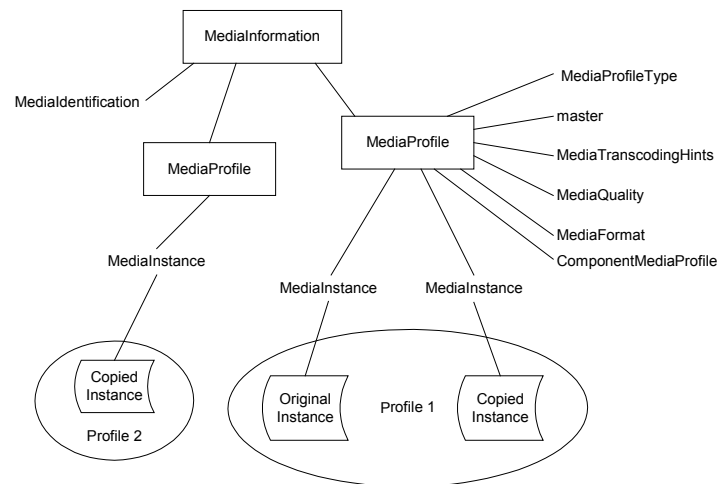


Figure 3.15 – Media description tools.

The most important media description tools depicted in Figure 3.15 are:

- **Media Information:** the media encoding and media location related features are all described by the Media Information DS. One description instance of the DS will be attached to one content item to describe it. The DS includes an identifier of the content item and a set of descriptors for the format of the item.
- **Media Profile:** one content item can have one or more media profiles that correspond to different media encoding parameters of the same content (e.g. different bit rate, different frame rate). One of the profiles is the original one, called *master profile*. It corresponds to the initially created or recorded profile. The others are derived from the master. Even if the audiovisual content is encoded with the same encoding software or hardware but with different parameters, different media profiles are created.
- **Media Instance:** content can be instantiated as physical entities called media instances (e.g. the location of a tape stored in the analogue archive of a broadcaster). An identifier and a locator specify a media instance.

Figure 3.17 shows how the Media Description tools are organized under the MediaInformationDS; these tools include: MediaIdentificationDS and MediaProfileDS that hold every descriptor containing media specific information, MediaQualityD, MediaInstanceD, MediaFormatD and MediaTranscodingHintsD. The other description tools are intended to facilitate the management of each variation of the content.

The media description tools include the following DSs and Ds:

Table 3.2 – Media description tools.

Description Tool	Functionality
MediaInformationDS	Describes information regarding the physical format of the audiovisual data.
MediaIdentificationD	Description tool that is specific to the identification of the audiovisual content, independently of the different available instances (e.g. a medical image, speech).
MediaProfileDS	Includes different description tools that allow the description of one profile of the media audiovisual content being described. The profile concept refers to the different variations that can be produced from an original or master media depending of on the options chosen for the coding, storage format, etc. This is different from the concept of profile in MPEG-2 and MPEG-4.

MediaInstanceD	Identifies and locates one MediaInstance of the MediaProfile (e.g. under some URL or the location of tapes stored in the analogue archive of a broadcaster).
MediaFormatD	Describes the coding format parameters of the MediaProfile (e.g. spatial resolution, audio coding format).
MediaTranscodingHintsD	Specify transcoding hints for the media being described. The purpose of this D is to improve quality and reduce complexity for transcoding applications (e.g. region of interest).
MediaQualityD	Describes the quality rating information of the MediaProfile (e.g. visual defects, objective, subjective).

The MediaProfileDS contains two of the most important descriptors for UMA applications: the MediaTranscodingHintD and the MediaFormatD.

The MediaTranscodingHintD provides important information regarding coded video content. This Descriptor holds information on the most relevant zones (Regions Of Interest - ROI) on a video, and the recommended integer pixel search range for motion vectors; this data may be instrumental for transcoding when a video must be re-encoded with other parameters. In Figure 3.16 there is an example showing how the MediaTranscodingHintD can benefit the adaptation process through the use of regions of interest.



Figure 3.16 – Example of image transcoding using Importance Hint information to preserve the fidelity of the face regions.

The other descriptor within MediaProfileDS is the MediaFormatD which holds information regarding the encoding of the media file, such as the encoding format, image spatial resolution, audio sampling rate, the used bit rate, and so on.

The ControlledTermUseType is a generic structure used to classify items with a term that exists in a controlled list of terms. These lists of controlled terms are called Classification Schemes and are listed in an annex of MPEG-7 Part 5 MDS [90].

The MediaFormat Descriptor gathers information related to the media content such as the file format, coding parameters, bandwidth required, etc. Two of the most important coding properties for an audiovisual stream are the audio and visual coding formats used: these fields of the MediaFormatD make reference to Classification Schemes defined in the MPEG-7 MDS where lists of coding formats are available. However, during this thesis it was detected that the MPEG-7 audio and visual coding formats classification schemes were very incomplete and, in some cases, incorrect. Moreover, since there are other types of media content that do not fall into either the category of audio or visual, e.g., graphics and scene composition streams, more descriptor components besides the VisualCoding and AudioCoding components should be defined for MediaFormatD in order to cover these cases (GraphicCodingFormat, SceneCodingFormat and GenericCodingFormat).

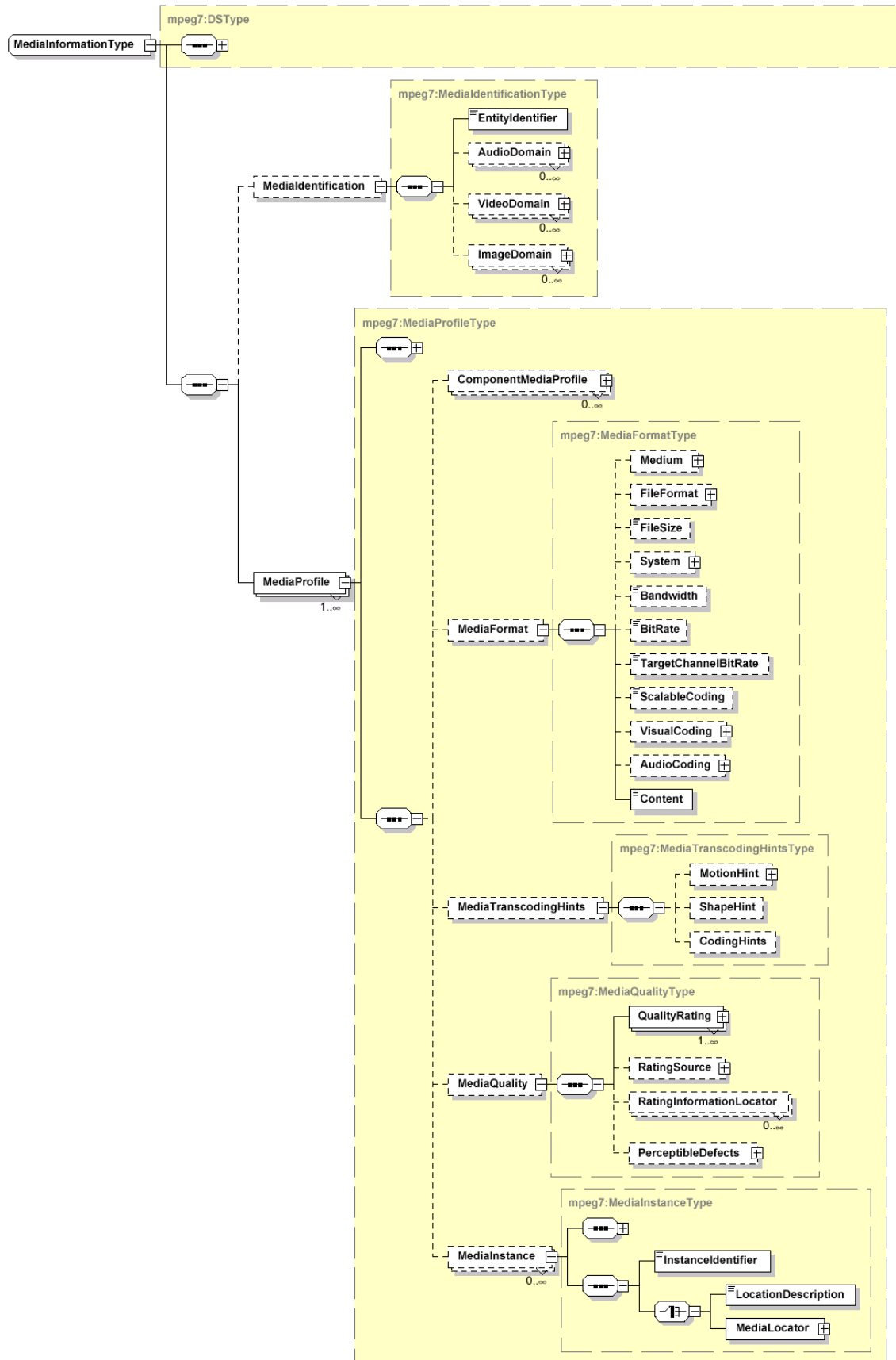


Figure 3.17 – MediaInformationDS elements.

Consequently, the author of this thesis proposed to MPEG-7 improved Classification Schemes for VisualCoding and AudioCoding and new Classification Schemes for GraphicCodingFormat, SceneCodingFormat and GenericCodingFormat. [116]; these proposals were fully accepted for integration in the MPEG-7 standard.

3.3.2 Content Structure Description Tools

Content structure description tools allow describing the spatial-temporal structure of the content, by referring to temporal segments of audio and video, spatial regions of images, temporally moving regions of video, spatial-temporal segments of audiovisual content and panoramic compositions of video. Some of these tools can also be applied to non-audiovisual content such as HTML. However, because these tools were not designed with HTML content in mind, there would be some restrictions.

The content structure description tools are grouped into the following categories:

- **Segment Entity Tools:** describe spatial-temporal segments of audiovisual content and hierarchical structural decompositions of segments.
- **Segment Attribute Tools:** describe attributes of segments related to spatial -temporal masks and importance for matching and point of view.
- **Segment Relation Tools:** describe structural relations among segments such as spatial -temporal relations.

3.3.2.1 Segment Entity Description Tools

The tools presented in this section are used to describe spatial, temporal, and spatial-temporal segments of audiovisual content, and hierarchical structural decompositions of segments. The SegmentDS provides the abstract base definition from which specialized segment entity tools are derived. Figure 3.18 shows the DSs that are derived from SegmentDS in order to describe any audiovisual segment.

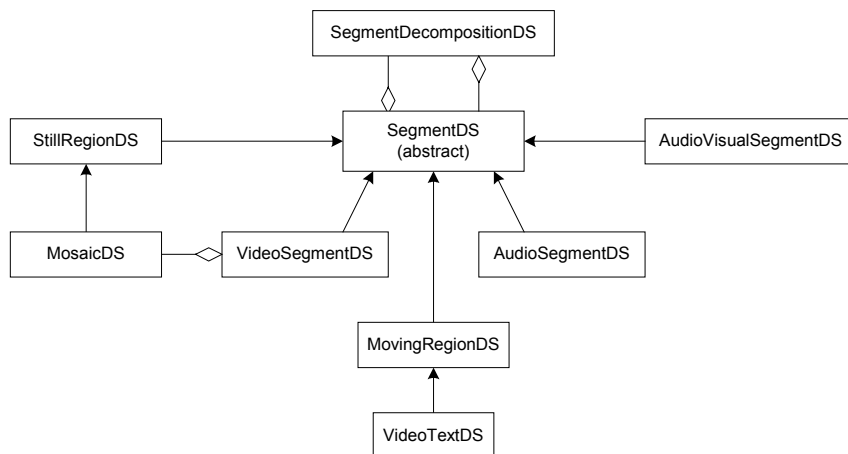


Figure 3.18 – Overview of the MPEG-7 tools for describing segments of audiovisual content.

All DSs from this group are concerned with segments of audiovisual content such as audio (AudioSegmentDS), image (StillRegionDS), and video (VideoSegmentDS). SegmentDS has the structure illustrated by Figure 3.19; each DS deriving from SegmentDS will add its own specific elements to describe the type of segment that it is featured for. The most important elements are: MediaInformation where the description tools presented in Section 3.3.1 are used; PointOfView which uses the description

tools described in Section 3.3.2.1; and Relation where segment relations can be described using the description tools presented in Section 3.3.2.3.

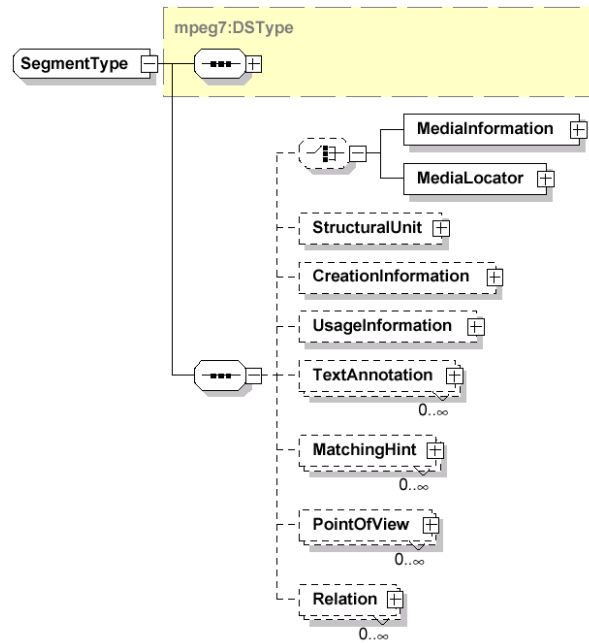


Figure 3.19 – Segment DS elements.

The MosaicDS is a very special DS that derives from SegmentDS: it describes a mosaic or panoramic view of a video segment. Aligning together and warping the frames of a video segment upon each other using a common spatial reference system is a possible process to build a mosaic. This gives a panoramic view of the scene with a single image as illustrated in Figure 3.20.



Figure 3.20 – Mosaic for the “Parliament” sequence constructed from 12 frames [88].

Beside its use for visual summarization, mosaics offer important means for performing search and comparison of video sequences. Image features can be extracted from the mosaic yielding a stable description of a whole video shot. Also the MosaicDS itself can be used as a good summarizing description of a video sequence, because it describes the global motion in the video and thus is suitable for searching and comparison of videos. Another functionality quite valuable in UMA applications is the use

of panoramic views when the terminal has no support for video. Through the MosaicDS, a UMA Engine can construct on the fly a panoramic view from the video frames (or use the mosaic available in the description) and deliver it to the terminals that do not support video (or as a replacement while the video is being adapted).

The Segment Entity description tools include the following DSs and Ds:

Table 3.3 – Segment Entity description tools.

Description Tool	Functionality
SegmentDS	Provides an abstract type from which the specialized SegmentDS are derived. The SegmentDS defines the properties of segments.
SegmentDecompositionDS	Describes the hierarchical decomposition of a segment into sub-segments. The SegmentDecompositionDS can describe a tree-structured hierarchical decomposition segment such as a table of contents for the video.
AudioVisualSegmentDS	Extends from the SegmentDS. Describes a spatio-temporal segment of audiovisual content, which may correspond to a set of pixels from an arbitrary sequence of video frames and a set of audio samples.
VideoSegmentDS	Extends the SegmentDS. Describes a temporal interval or segment of video, which can correspond to an arbitrary sequence of frames, a single frame, or even the full video sequence.
AudioSegmentDS	Extends from SegmentDS. Describes a temporal interval or segment of audio, which can correspond to an arbitrary sequence of audio samples, a single sample, or even the full audio sequence.
StillRegionDS	Extends the SegmentDS. The StillRegionDS describes a spatial range or region of an image or video frame, which can correspond to an arbitrary set of pixels of the image or video frame, a single pixel, or even the full image or video frame.
MovingRegionDS	Extends from the SegmentDS. Describes a spatio-temporal area of video, which can correspond to a set of pixels from an arbitrary sequence of video frames, one pixel of one video frame, one full video frame, or even the full video sequence.
MosaicDS	Extends the StillRegionDS. Describes a mosaic or panoramic view of a video segment. A mosaic can be constructed by aligning together and warping the frames of a VideoSegment upon each other using a common spatial reference system.
VideoTextDS	Extends from the MovingRegionDS. Describes a region of video or an image corresponding to text or captions.

3.3.2.2 Segment Attribute Description Tools

This type of description tools is used for describing various attributes of segments related to importance for matching and point of view content access.

From this group of tools, only the PointOfView Descriptor is important for content customization applications. This Descriptor provides a way to describe the relative importance of segments given a specific viewpoint. For example, for video segments of a football match between Team A and Team B, PointOfViewD descriptions provide the relative importance of each video segment for the fans of each team. This is very important, for example when it comes the time to make a summary of the match for fans of the two teams. The content creator or an automatic system assigns the relative importance values to the video segments. This Descriptor can be used to summarize audiovisual content dynamically depending on required viewpoints and the duration each user requests.

Table 3.4 – Segment Attributes description tools related to UMA.

Description Tool	Functionality
PointOfViewD	Describes the relative importance of segments given a specific viewpoint.

The Segment Attribute description tools include more DSs and Ds which however are not so important for UMA related purposes.

3.3.2.3 Segment Relation Description Tools

These description tools are designed to describe relations among audiovisual segments. When a UMA Engine is processing audiovisual content, these description tools are not very useful, because the order (spatial and/or temporal) of audiovisual segments is never changed. However, when a UMA Engine is adapting other type of structured content (e.g. HTML), tools to describe (spatial) segment relations are fundamental.

Similarly to the SegmentDS, the Segment Relation DSs derive all from one base DS, the SegmentRelationDS. From this DS, others DSs are derived to describe temporal, spatial and spatial-temporal relations. Figure 3.21 shows the organization of the tools for describing relations among segments (the Binary...DSs are intended to describe the relation between two segments while the NArY...DSs are used to describe relations between two or more segments).

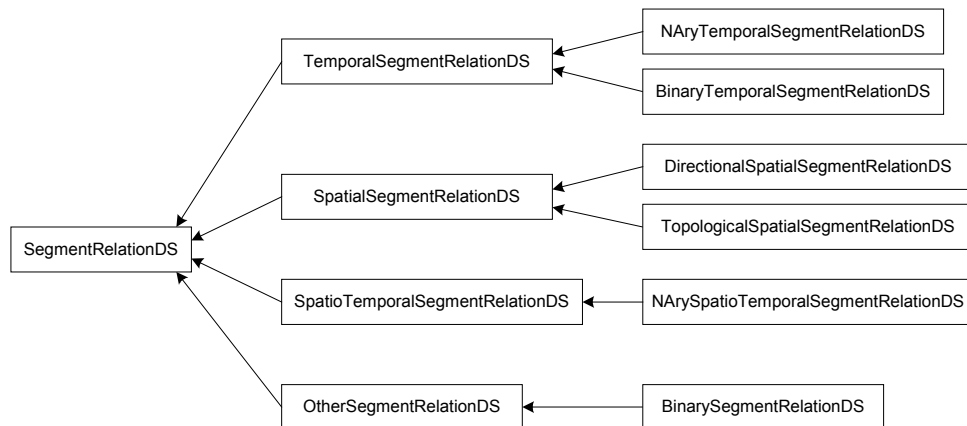


Figure 3.21 – Preview of the MPEG-7 tools for describing relations among segments.

To illustrate the use of Segment Relation DSs, consider the example shown in Figure 3.22 [90]. This example shows an excerpt from a football match. Two video segments, one still region and three moving regions have been defined. Figure 3.22 a) shows two key-frames for each video segment, with each region well identified in each key-frame (ball, player, goal and goalkeeper). A graph describing the structure of the content and thus the relations among the segments is shown in Figure 3.22 b). The video segments defined are:

- **Dribble & Kick:** includes the ball, the goalkeeper and the player. The ball remains close to the player who is moving towards the goalkeeper. The player appears on the right of the goalkeeper.
- **Goal Score:** includes the same moving regions from Dribble & Kick plus the still region called goal. In this part of the sequence, the player is on the left of the goalkeeper and the ball moves towards the goal.

This simple example illustrates the flexibility of this kind of representation. Note that this description is mainly structural because the relations specified in the graph edges are purely physical with the nodes representing segments, which are entities described by creation, usage and media information as well as low-level descriptors such as color, shape and motion. The only explicit semantic information is available from the textual annotation, where keywords such as ball, player, or goalkeeper can be specified.

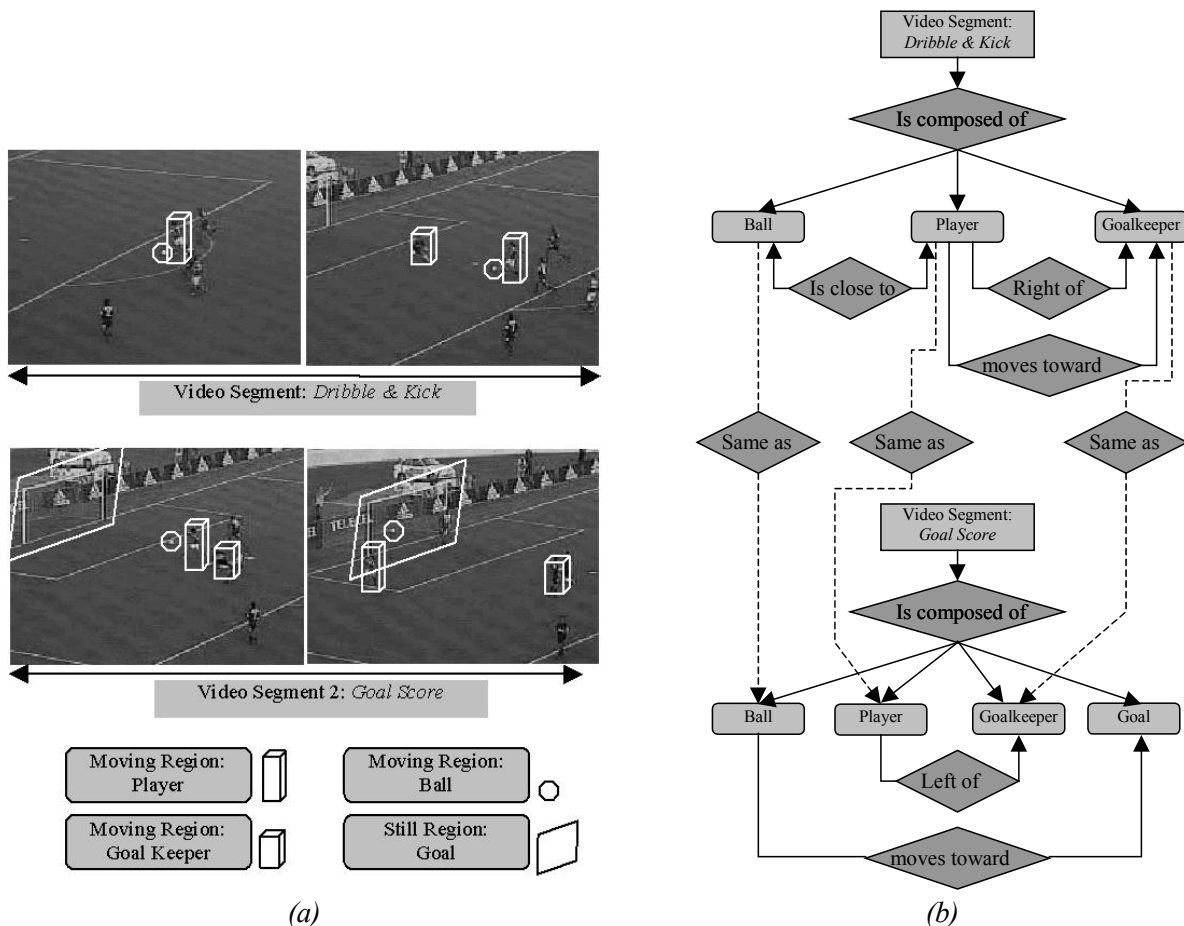


Figure 3.22 – (a) Example of video segments and regions. (b) Example of a segment relationship graph.

Segment relations that can help a UMA Engine are provided through the DirectionalSpatialSegmentRelationDS, which describes spatial relations between two segments (e.g. *left*, *above*, *south*, and *north*). Directional spatial relations describe how the segments are placed and relate to each other in a 2D or 3D space. This DS is specific to audiovisual content, and it would have to be modified to be used with HTML data.

The MPEG-7 Segment Relation description DSs and Ds are:

Table 3.5 – Segment Relation description tools.

Description Tool	Functionality
SegmentRelationBaseDS	It is the abstract type that describes a relation among segments.
SegmentRelationDS	Extends from SegmentRelationBaseDS. Describes a relation among segments.
TemporalSegmentRelationDS	Extends from the SegmentRelationDS. Describes temporal relations among segments.
SpatialSegmentRelationDS	Extends from the SegmentRelationDS. Describes spatial relations among

	segments.
SpatioTemporalSegmentRelationDS	Extends from the SegmentRelationDS. Describes spatial-temporal relation among segments.
OtherSegmentRelationDS	Extends from the SegmentRelationDS. Describes a relation that is not spatial, temporal, or spatial-temporal among segments (e.g. one segment is the annotating another).

3.3.3 Content Navigation and Access

This group of Description Schemes is used to facilitate the browsing and retrieval of audiovisual content. The tools defined in this section are organized into the following groups:

- **Summarization:** to improve browsing and navigation, summarization tools enable the description of audiovisual summaries and abstracts.
- **Views and View Decompositions:** tools for describing views and view decompositions of images, video, and audio signals in time and frequency.
- **Variations:** tools for describing the associations or relationships between different variations of audiovisual programs.

3.3.3.1 Summarization Tools

These description tools facilitate the discovery, browsing, navigation and visualization of audiovisual content. With summarization tools, multiple summaries of the same content can be specified at different levels of detail, without the need for generating or storing multiple variations of the content. The SummarizationDS is the root element of a collection of summaries related to the same audiovisual content. The SummaryDS describes each summary within SummarizationDS; SummaryDS is an abstract DS, from which two DSs are derived, allowing the creation of two different types of summaries:

- **Hierarchical Summaries:** the information is organized into successive levels, each one describing the audiovisual content at a different level of detail.
- **Sequential Summaries:** provides a sequence of images or video frames, possibly synchronized with audio, which may compose a slide show or audiovisual skim.

The DSs that match the above types of summaries are the HierarchicalSummaryDS and the SequentialSummaryDS, respectively, see Figure 3.24.

Table 3.6 – Summarization description tools.

Description Tool	Functionality
SummarizationDS	Specifies a set of summaries to enable rapid browsing, navigation, visualization and sonification of audiovisual content. Each audiovisual content may have many different summaries inside this DS.
SummaryDS	Abstract DS from which other specific summary description schemes are derived. A summary may be associated with an entire audiovisual program or with a segment from a program. A summary may contain references to the description of the original/source audiovisual content.

3.3.3.1.1 Hierarchical Summary DS

The HierarchicalSummaryDS is constructed around the generic notion of audiovisual temporal segments, using the HighlightSegmentDS. Each of these segments contains references (through MediaInformationDS) to the audiovisual data, to provide access to the annotation referring to key-video clips or key-audio clips, to key-frames or to key-sounds and may also contain textual annotation referring to key-themes. These audiovisual segments are grouped into summaries, or highlights, using the HighlightSummaryDS.

Figure 3.23 illustrates the use of the HierarchicalSummaryDS description schemes. In this figure, the HierarchicalSummaryDS contains two HighlightSummaryDSs, where the first summary consists of two HighlightSegmentsDS and the second summary consists of three HighlightSegmentsDS. Such summaries may correspond to two different themes (e.g. a brief and a detailed football game summary) and could provide alternative views on the original audiovisual content. The HighlightSummaryDS is recursive in nature, enabling summaries to contain other summaries. This capability can be used to build a variety of hierarchical summaries, e.g. a summary of the best shots of the user's favorite team.

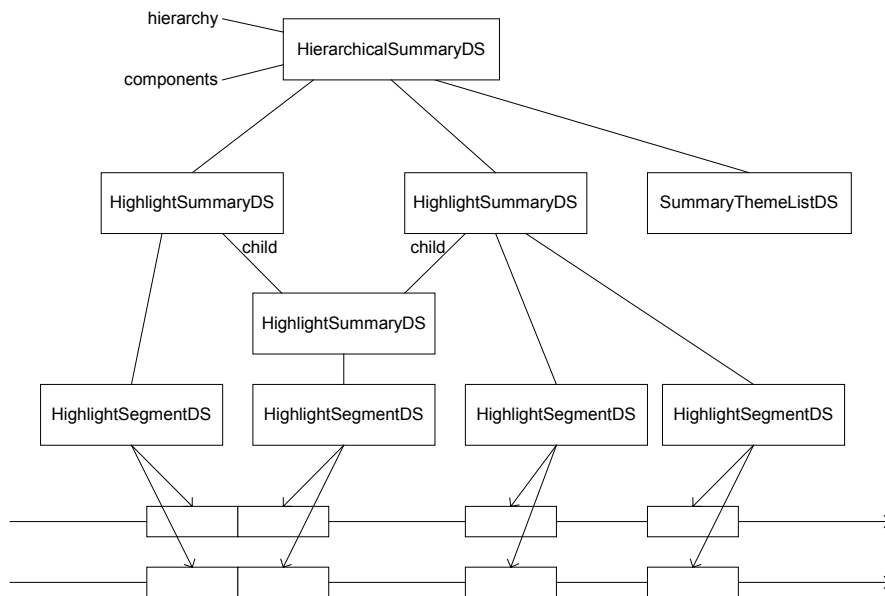


Figure 3.23 – Example of HierarchicalSummaryDS containing a summary.

In Table 3.7 a brief description of each DS and D in the HierarchicalSummaryDS structure is presented.

Table 3.7 – Hierarchical summaries description tools.

Description Tool	Functionality
HierarchicalSummaryDS	Describes a group of audiovisual summaries, where each summary may consist of a hierarchy of components. Elements in a hierarchy are described using the HighlightSummaryDS.
SummaryThemeListDS	Specifies a set of textual themes or events associated with summaries of audiovisual content. Such themes may be associated with particular key-frames, key-video clips, key-audio clips or with entire summaries.
HighlightSummaryDS	Describes an audiovisual summary by referring to a sequence of audiovisual segments or images. A HighlightSummary may correspond to a summary, for example, with a particular time duration or a particular set of events. Embedding HighlightSummary elements recursively can form a coarse-to-fine hierarchy of summaries.
HighlightSegmentDS	Provides access to an audiovisual segment of audiovisual content. This DS may be used to refer to a key-video clip, a key-audio clip, key-frames and key-sounds associated with an audiovisual

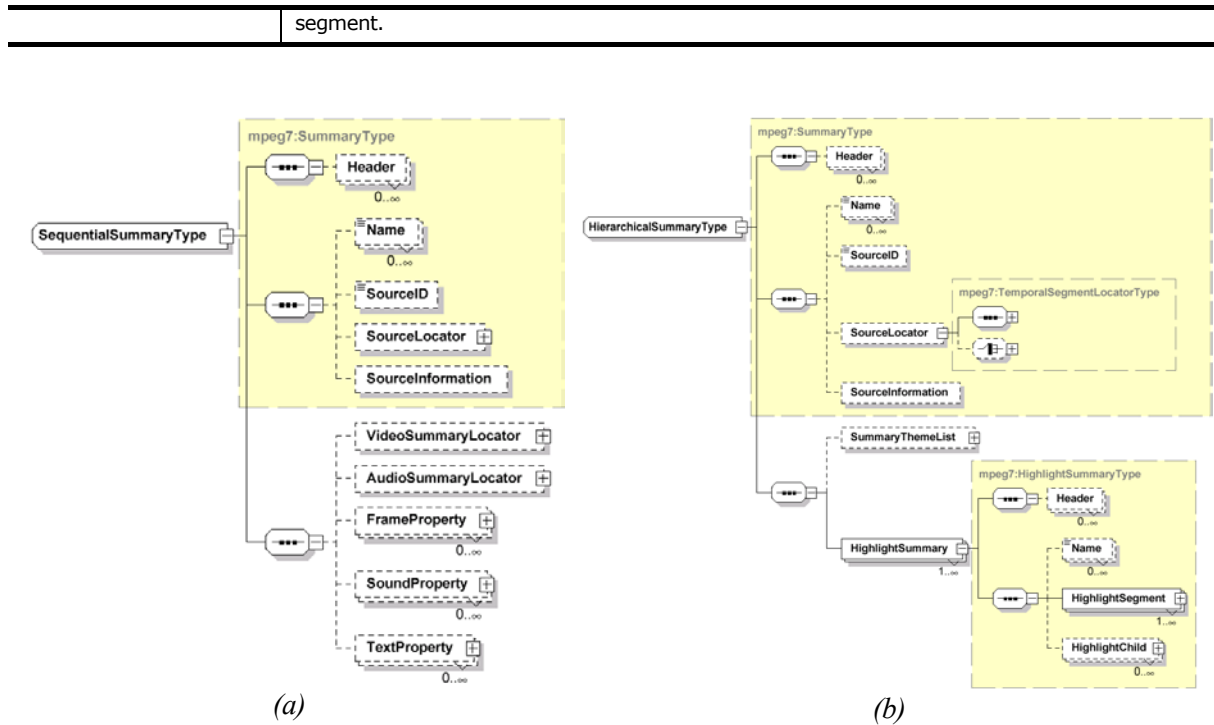


Figure 3.24 – Structure of the Hierarchical Summary and Sequential Summary DSs.

3.3.3.1.2 Sequential Summary DS

The SequentialSummaryDS consists of multiple sequences or tracks of audiovisual segments describing a particular modality of time-varying audiovisual data. In particular, this DS can contain a sequence of segments describing images or video frames, a sequence of segments describing audio-clips, and a sequence of segments describing textual information associated with parts of the audiovisual data.

Each segment describes the temporal features, as well as other properties, of a frame (FramePropertyD), of an audio-clip (SoundPropertyD) or of the textual information (TextPropertyD). Furthermore, the elements in each track or sequence may be synchronized across them.

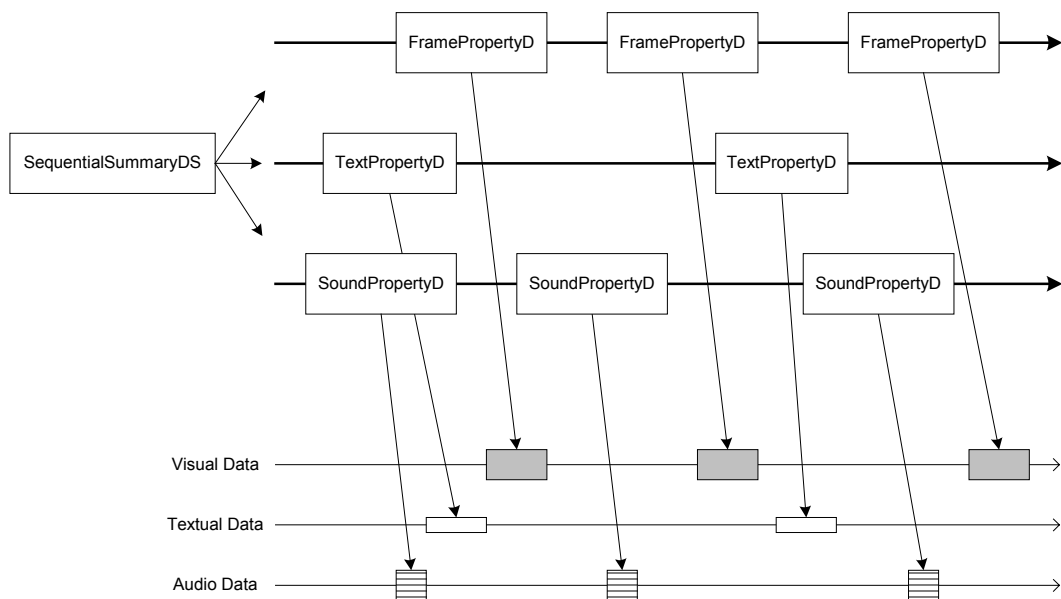


Figure 3.25 – Example with the SequentialSummaryDS.

In Figure 3.25, the SequentialSummary DS describes the properties of three tracks associated with the audiovisual content: video frames, textual annotation and audio clips. Each video frame is described by FramePropertyD, each annotation by TextPropertyD and each audio clip by SoundPropertyD.

In Table 3.8, the Ds and DSs used in sequential summaries are briefly described.

Table 3.8 – Sequential summary description tools.

Description Tool	Functionality
SequentialSummaryDS	Specifies a single audiovisual summary, which may contain a sequence of images or video frames, possibly synchronized with audio, composing a slide-show or audiovisual skim.
FramePropertyD	Describes certain properties associated with an image or video frame in the original audiovisual content. These properties may specify the location of the image or video frame possibly by referring to the original audiovisual content.
SoundPropertyD	Describes certain properties associated with an audio clip or the components of an audio slide show. These properties may specify the location of the audio clip, as a separate file and/or possibly by referring to the original audiovisual content.
TextPropertyD	Describes certain properties of textual information associated with an audiovisual summary.

3.3.3.2 Views

Views description tools correspond to a region of an image, video or audio signal in multi-dimensional space or frequency, which is defined in terms of a space and frequency division. For example, a low-resolution view of a segment of an audio signal provides a particular kind of View of the audio signal in time and temporal-frequency.

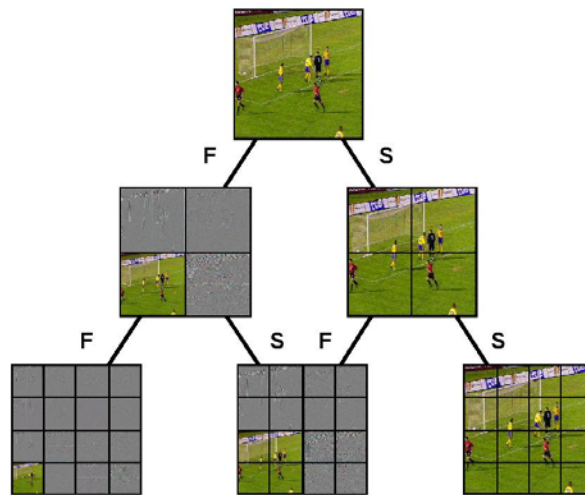


Figure 3.26 – Space and Frequency Graph decomposes images and audio signals in space and frequency.

Figure 3.26 illustrates the decomposition of images in space and frequency, which enables efficient multi-resolution access, and progressive retrieval of the image data. The views description tools group includes several cases of multi-resolution access to content; Table 3.9 lists the existing DS and D tools.

Table 3.9 – Views description tools.

Description Tool	Functionality
ViewDS	Provides an abstract type for the specific types of views.
SpaceViewDS	Describes a multi-dimensional view of an audiovisual signal in space and/or time. In general, a

	SpaceView is specified in terms of a multi-dimensional partition in the spatial or temporal domain of the image, video or audio signal.
FrequencyViewDS	Describes a multi-dimensional view of an audiovisual signal in frequency. In general, a FrequencyView is specified in terms of a multi-dimensional partition in the spatial-frequency and/or temporal-frequency domain of the audiovisual signal.
SpaceFrequencyViewDS	Describes a multi-dimensional view of an audiovisual signal in space and frequency. In general, a SpaceFrequencyView is specified in terms of multi-dimensional partitions in the spatial and/or temporal domain in addition to the spatial-frequency and/or temporal-frequency domain of the audiovisual signal.
ResolutionViewDS	Describes a multi-dimensional low-resolution view of an audiovisual signal. Conceptually, a ResolutionView is a specific kind of FrequencyView that corresponds to the low-frequency portion of the frequency plane. A ResolutionView is specified in terms of the sampling factor on each dimension.
SpaceResolutionViewDS	Describes a multi-dimensional view of an audiovisual signal, which corresponds to a low-frequency region from the frequency plane.

3.3.3.3 Views Decomposition

Views decomposition description tools correspond to an organized set of views that provides a structured decomposition of an image, video or audio signal in a multi-dimensional space and/or frequency. For example, the spatial quad-tree decomposition of an image into a hierarchy of views as shown in Figure 3.26.

Table 3.10 – Views decompositions description tools.

Description Tool	Functionality
ViewDecompositionDS	Abstract base class from which specific types of view sets and tree and graph based decompositions are derived.
ViewSetDS	Describes a set of Views, which has properties indicating completeness and redundancy in covering the space and/or frequency planes.
SpaceTreeDS	Describes a tree-based spatial decomposition of an audiovisual signal. A SpaceTree consists of SpaceTree nodes, which can optionally contain a SpaceView. Each node also has a number of children, which are also SpaceTree nodes, which can optionally contain a SpaceView.
FrequencyTreeDS	Describes a tree-based decomposition in frequency of an audiovisual signal.
SpaceFrequencyGraphDS	Describes a directed acyclic graph-based decomposition of an audiovisual signal in space and frequency. This graph organization is composed by SpaceFrequencyGraph nodes, which can optionally contain a SpaceFrequencyView.
VideoViewGraphDS	Describes a directed acyclic graph-based decomposition of an image, video or audio signal in spatial and/or temporal-frequency. This graph organization is composed by VideoViewGraph nodes, which can optionally contain a FrequencyView.

3.3.3.4 Variations

Variations tools serve important content management functionalities by tracking the variations of audiovisual content that result from various types of multimedia processing such as summarization, translation, resolution reduction, revision and so forth.

Figure 3.27 presents the structure of the two DSs defined in this group of description tools: VariationSetDS and VariationDS. The VariationSetDS holds a description of the source content and groups several variations of the same source. Having a piece of audiovisual content, several variations of this content can be obtained by the VariationSetDS that groups the various variations descriptions and the source description itself. Each content variation is described by the VariationDS.

Table 3.11 – Variations description tools.

Description Tool	Functionality
VariationSetDS	Specifies the source content and the set of existing variations.
VariationDS	Describes different versions of the same audiovisual source. Variations may be generated in a number of different ways or reflect revisions of the source audiovisual content.

A VariationDS holds all the information contained in the corresponding media type segment descriptor (see Section 3.3.2). Besides this information it also points to the description of the source content and describes the difference between the source and the adapted variation (in the VariationRelationship parameter).

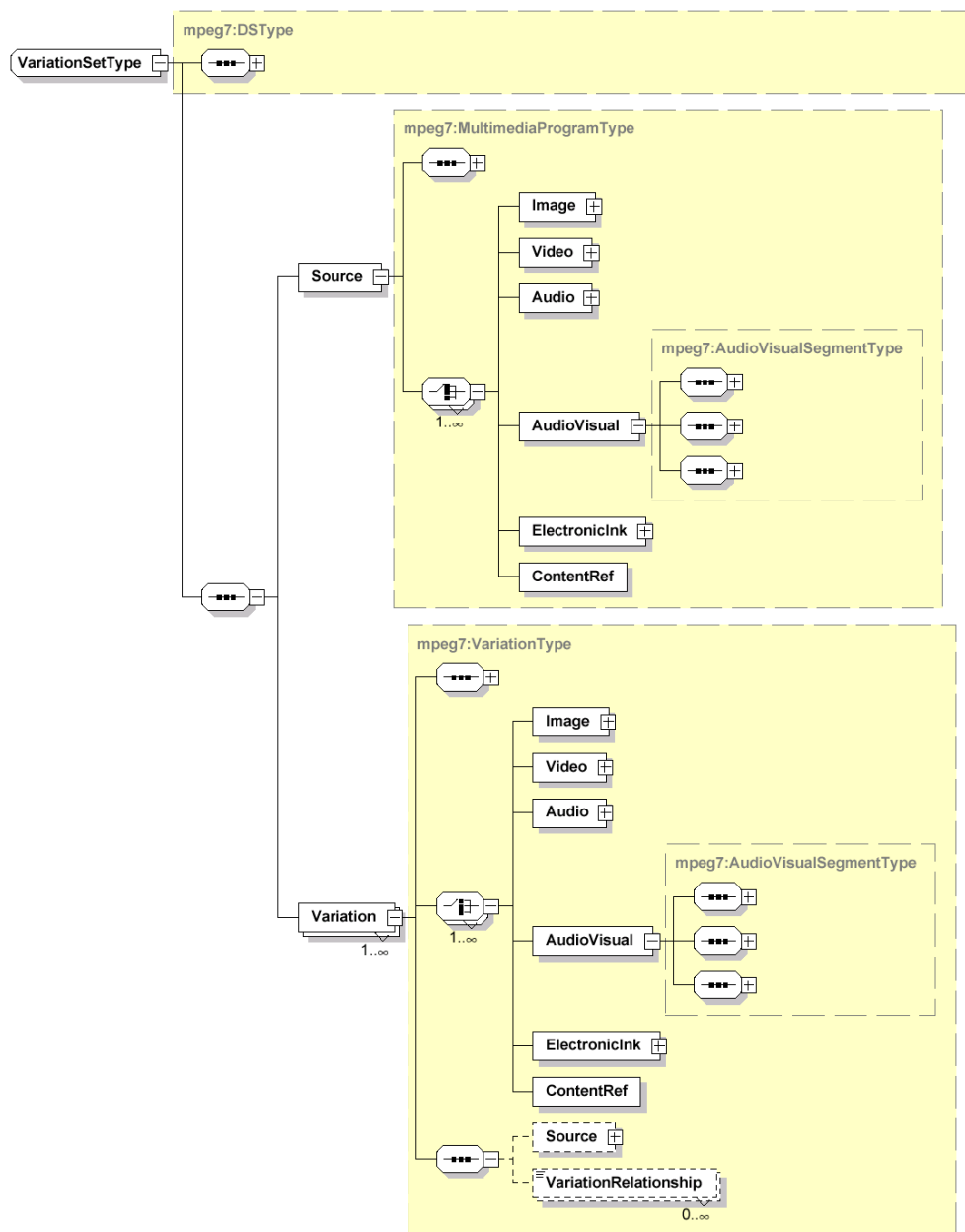


Figure 3.27 – Variations description tools.

Figure 3.28 illustrates how these two DSs can improve the versioning management of the generated content variations. The VariationSetDS describes the source audiovisual content, and holds a VariationDS for each content adaptation that is stored (the example shows two). One of the VariationDSs is describing a variation with a lower spatial resolution and a lower frame-rate; the second VariationDS describes a variation that is only a key-frame. Each variation description includes a specific fidelity value indicating the fidelity of the variation program with respect to the source program.

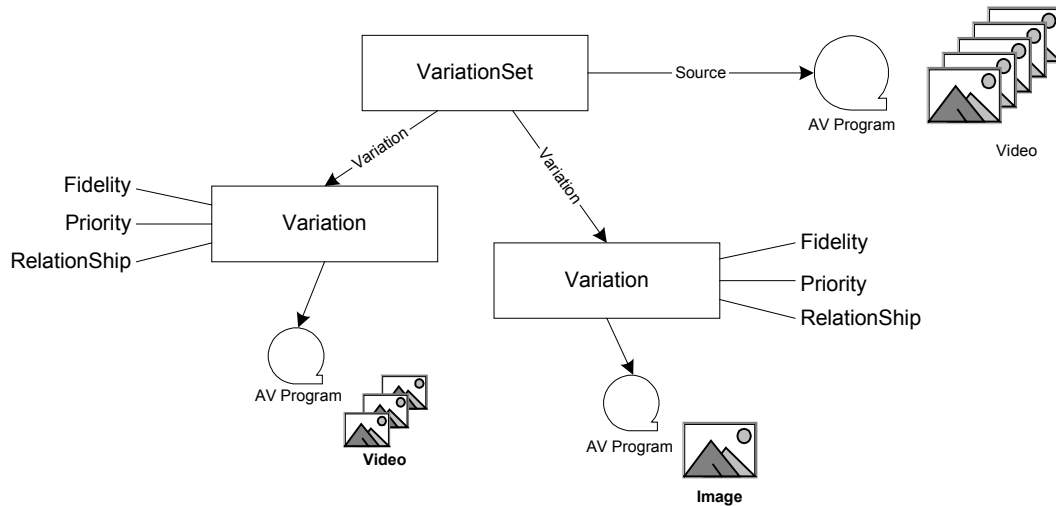


Figure 3.28 – Illustration of key concepts in VariationDS.

3.4 Summary

Content descriptions play a very important role in a UMA System as was discussed in Chapter 1. Without them the content customization process would be much more difficult and the results would offer a poorer user experience.

The analysis of the content description problem made in Section 3.1 focused on the UMA enabling description tools. From the bibliographic research made, it is possible to conclude that the MPEG-7 standard, presented in Section 3.2, is the most complete solution currently available to describe audiovisual content. The most relevant MPEG-7 description tools to enable UMA applications were covered in Section 3.3. Structured content such as multi-object MPEG-4 streams is outside the scope of the work for this thesis in terms of content adaptation mechanisms; nevertheless, Section 3.3.2 presented the description tools that can be used in a UMA System to handle structured content. Moreover, there are types of structured content that cannot be handled with present MPEG-7 descriptions such as HTML.

The UMA Engine implemented in this thesis uses the Media Description tools and the DSs and Ds under VariationSetDS that were presented in this chapter. The descriptors below VariationSetDS provide the essential tools for managing and describing the information that shall be used to perform the customizations implemented in the context of this thesis.

Chapter 4

User Environment Description

The user environment concept, presented in Chapter 1, is understood as the set of characteristics that influence and/or condition the user access to content. A diversity of elements characterizing the user environment may enter in the equation that rules the content server delivery policy. In Chapter 2, the universal mailbox application illustrates a real life scenario where the user environment strongly conditions the way the user has to receive the multimedia information.

This chapter will discuss the user environment characterization problem, particularly by studying the currently available and emerging candidate solutions. The WAP standard will also be presented, as this is one of the most motivating scenarios for the UMA Engine. Moreover a technical solution to address the problem of user environment description will be proposed. Both the discussion of the current candidate solutions [123], as well as the proposal [124] for a solution has been presented by the author of this thesis to the MPEG standardization group. Following these contributions, MPEG decided to address the problem of user environment description as Part 7 of the MPEG-21 standard [117], which is currently under development, entitled Digital Item Adaptation [120].

4.1 User Environment Dimensions

In Chapter 1, it was discussed that the variety and amount of multimedia content is becoming a huge problem when searching for information in useful time. Another problem discussed was the number of different terminals and conditions under which the user accesses information. For the content side, there is a solution available to describe the multimedia content to be accessed; this solution is the MPEG-7 standard presented in Chapter 3. The objective of this chapter is the pro-active discussion of a similar solution for the description of the user environment. This solution will act as an interface between the specific user consumption conditions and the content available on the other side. Figure 4.1 illustrates the idea.

In the list of requirements that concluded Chapter 2 there is a part concerning the user environment description notably considering the following dimensions: the user **terminal** capabilities (software and hardware), the **access network** characteristics, the user **natural environment** features and the **user preferences**. With an interface providing information on the user environment key elements, as illustrated in Figure 4.1 and required by the list of requirements identified in Chapter 2, it would be easier for an application to customize its content and services to the user's environment conditions. Moreover it would improve applications portability since an application could be developed so that it checks the conditions in which it is running and behaves accordingly.

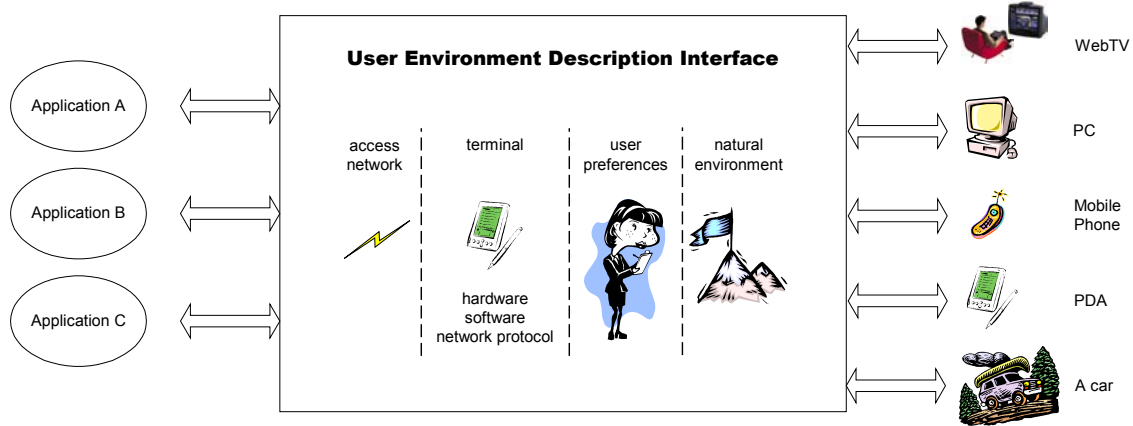


Figure 4.1 – User Environment Description Interface.

Using a broader concept, the user environment characterization concerns all relevant information regarding the user and that can be described and used with benefit even for other purposes beside content customization. For example, the list of available payment cards may be part of the user environment characterization but this is not so important for the task of a UMA Engine. However, in this thesis, only user environment characteristics that are directly related to content adaptation or content selection will be covered.

In the following sections, the user environment key dimensions that are relevant for the improvement of the performance and the quality of the content customization process will be discussed. These dimensions have been proposed and accepted as the major dimensions for the usage environment description problem as defined in the Requirements document and Call for Proposals issued by MPEG in the context of the development of MPEG-21 Part 7: Digital Item Adaptation [120].

4.1.1 Terminal

The most commonly used software with which a user accesses multimedia content is a Web browser and its plug-ins. The browser is dependent on the hardware and software on the top of which it is running. Regarding the terminal, the most relevant parts to characterize are the **hardware** and the **software**. These parts are addressed in the following sections.

4.1.1.1 Hardware

The terminal's hardware is one of the most important elements in the terminal performance and capabilities. A colored image makes no sense in a black and white terminal, so the physical elements that compose the terminal must be fully described in order to negotiate or customize the content appropriately.

Some of the most important terminal hardware elements are:

- Video capabilities, e.g. display resolution/display size/number of colors.
- Audio capabilities, e.g. number of output channels.
- Input capabilities, e.g. keyboard type.
- Terminal CPU, e.g. PowerPC, ARM.
- Terminal memory, e.g. 256 kbyte, 1 Mbyte.

4.1.1.2 Software

The software may be organized in three major elements:

- Operating System (OS).
- Application receiving the content which should describe the constraints under which the content will be rendered; for example, for a video, the size of the window where the video will be played.
- Multimedia decoding software; these modules are used by an application to decode and render the content within the application space (visual or audio).

Each one of these elements must have a full description of its capabilities and enough information on them must be supplied to the UMA Engine in order that it can adequately adapt the content.

4.1.2 Access Network

The access network can be the cause of very annoying effects for the user: delay, bandwidth shortage, channel errors, etc. The access network should be described to prevent as much as possible delays or pauses in the content rendering. Both the network physical characteristics and the network protocol characteristics should be described.

In terms of the physical access network, there are basic characteristics such as the peak bit rate, medium bit rate, and bit error rate that must be described in the user environment.

The network protocol is also very important in accessing the content appropriately (for example, for Quality of Service support); it may also allow only some types of multimedia content or services (for example, WAP only supports black&white bitmaps). The ability to stream content in real-time is also very important. Another hot topic at the network protocol level is the need to describe the billing models that the network supports in order to negotiate network resources (dynamic bandwidth, for example).

4.1.3 Natural Environment

This user environment dimension includes the natural features of the surrounding user environment that may influence the content adaptation: location is the most obvious natural environment feature that can be used to create new services (nowadays some of the industry biggest investments are being made in location based services). User location is a variable from the user surrounding environment but this is just one of several measures that can be used to improve and deploy new services. Temperature, pressure, altitude, time, and date are other natural environment physical variables that might be included in the user environment description. A UMA Engine can use these characteristics for content adaptation and/or selection purposes, e.g. making the type of advertising change with the temperature, the famous ice cream versus hot chocolate problem.

4.1.4 User Preferences

So far the technology that users employ to access content and the natural conditions under which they do it have been discussed, but the last element in this chain should not be forgotten: the user itself, a human being with his preferences and behaviors. MPEG-7 user preferences description tools [90] address these characteristics supplying information to search and filter information taking into account the user specific preferences. Information regarding user preferences included in a user description might be more general than just multimedia content preference, such as food and accommodation preferences for advertising or retrieval. As in the previous section, a UMA Engine may use these characteristics for content adaptation and selection purposes (for example, for building a movie summary).

4.2 The CC/PP (Composite Capability/Preference Profiles) Framework Solution

Having described the user environment problem to be addressed, it is important to review and study the technologies available, which may completely or partly solve the problem at hand. Although there is not much relevant technology available to address the problem described in the previous section, the Composite Capabilities/Preference Profile framework outstands as the most interesting technology, enabling the creation of tools to solve the problem in question.

In 1998, the World Wide Web Consortium (W3C) created the Mobile Access Interest Group to study the specific requirements of mobile terminals to access the World Wide Web. With a strong involvement from the Wireless Application Protocol (WAP) Forum [46] and contributions from the Internet Engineering Task Force CONNEG Working Group [25], a new W3C framework was developed, the Composite Capabilities/Preference Profile framework.

In the W3C Mobile Access document [38], the opening text states:

“The goal of the CC/PP framework is to specify how client devices express their capabilities and preferences (the user agent profile) to the server that originates content (the origin server). The origin server uses the “user agent profile” to produce and deliver content appropriate to the client device. In addition to computer-based client devices, particular attention is being paid to other kinds of devices such as mobile phones.”

From this paragraph, the final goal of the CC/PP framework becomes clear: the description of the user devices capabilities and his preferences.

CC/PP is a user side framework for content negotiation; it allows representing a collection of data, which describes the hardware and software capabilities, and the user preferences. CC/PP is not only intended for mobile terminals, since the standard is general enough to describe any user environment (called *user profile* in CC/PP terminology).

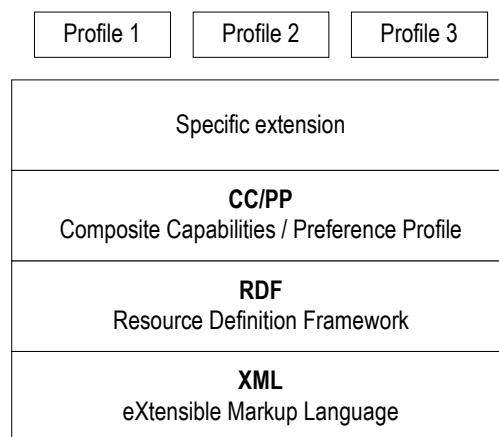


Figure 4.2 – CC/PP is a RDF application, which in turn is an XML application.

Figure 4.2 illustrates how CC/PP derives from XML/RDF technologies. RDF (Resource Description Framework) [37] was designed as a general-purpose metadata description language. XML is a descriptive language that only imposes syntactic constraints. RDF is an application of XML that imposes needed structural constraints to provide unambiguous methods of expressing semantics. RDF additionally provides the means for publishing both human-readable and machine-understandable vocabularies designed to encourage the reuse and extension of metadata semantics among disparate information communities. RDF can be used in a variety of application areas; for example, in *resource discovery* to provide

better search engine capabilities, in *cataloging* for describing the content and content relationships available at a particular Web site, page, or digital library, by *intelligent software agents* to facilitate knowledge sharing and exchange.

The CC/PP framework is divided into several parts that describes its organization and how it must be used:

- **Structure:** defines the profiles' organization.
- **Attribute vocabularies:** define basic vocabularies to be used when defining profiles.
- **Client profiles:** define how the user profiles are implemented.
- **Proxy support:** defines how the profiles should be extended to include proxy support.
- **Extensibility and namespaces:** define how the CC/PP framework should be extended to include more vocabularies.

The CC/PP framework uses the term *user profile* with the same meaning of *user environment description*.

4.2.1 Profile Structure

The structure by which profile characteristics are organized is a tree like data model. Top-level branches are the components described in the profile. Each major component may have a collection of attributes or preferences. A CC/PP profile is broadly constructed as a 2-level hierarchy including:

- **Components:** the high level items described in a profile;
- **Attributes:** the smaller items under each component in the profile.

Initial branches of the CC/PP profile tree describe major components of the user environment. Possible components are the hardware platform, the software platform and the application that retrieves and presents the content to the user (for example, a browser). A graphical representation of the top of a CC/PP tree based on three components, would look as shown in Figure 4.3:

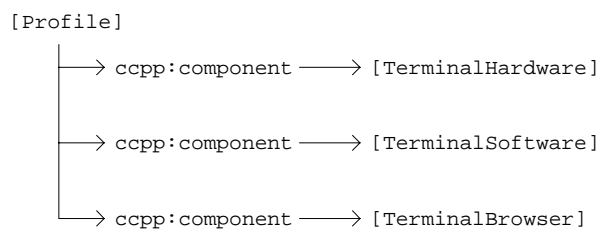


Figure 4.3 – Hierarchical organization of a CC/PP profile [40].

The description of each component is a sub-tree whose branches are the capabilities or preferences associated with that component. RDF allows modeling a large range of data structures, including arbitrary graphs. This could lead to a very complex user profile, which would be very hard to process efficiently. However, typical capabilities can often be described using a small number of CC/PP attributes, each having a simple, atomic value.

Figure 4.4 illustrates a tree structure with a few components and their attributes. Each component has a type from which its class definition is inherited. The graph in Figure 4.4 has the RDF textual

representation shown in Figure 4.6. In the first element of the RDF textual description, the different layers that are used in that description are declared: the first is XML, then RDF, then CC/PP and the WAP specific vocabulary (this embedding mechanism is called namespace and will be further explained in Section 4.2.1.2).

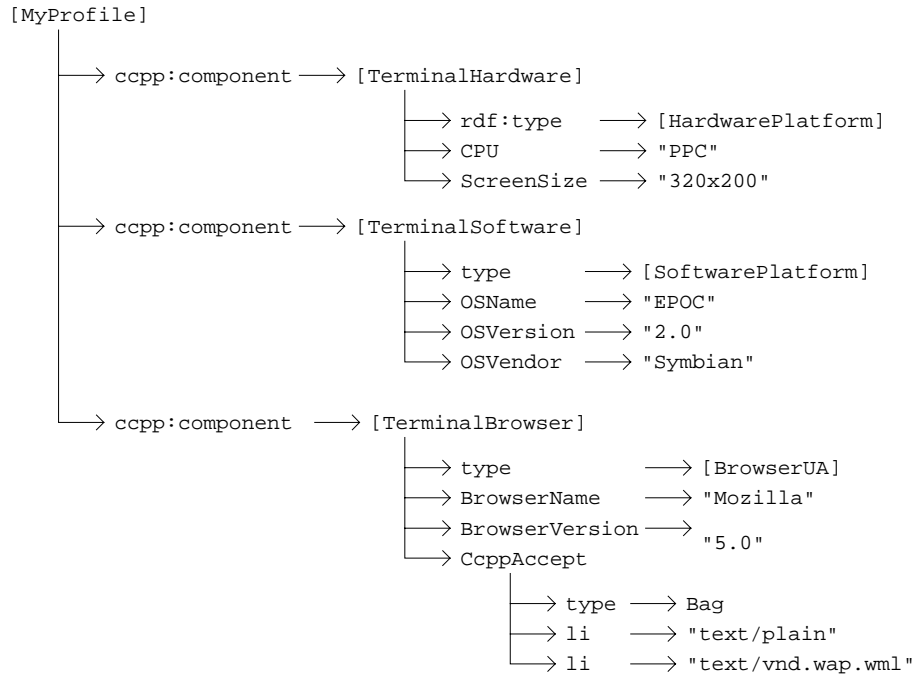


Figure 4.4 – Example of a CC/PP device profile [40].

4.2.1.1 Defaults

Each CC/PP component may provide a separate, subordinate collection of default capabilities (attributes). This collection of Defaults can either be a separate RDF document that is named via a Uniform Resource Identifier (URI) [30] or the collection can be explicitly listed in the profile. If a capability described in a collection of Defaults is also described as a non-default capability, the non-default description prevails. For example, a hardware vendor may provide default profiles describing the physical properties of each model of his products. However, the current owner of the product may be able to add options, such as additional memory, persistent storage or additional I/O devices that add new capabilities or change the values of some original capabilities.

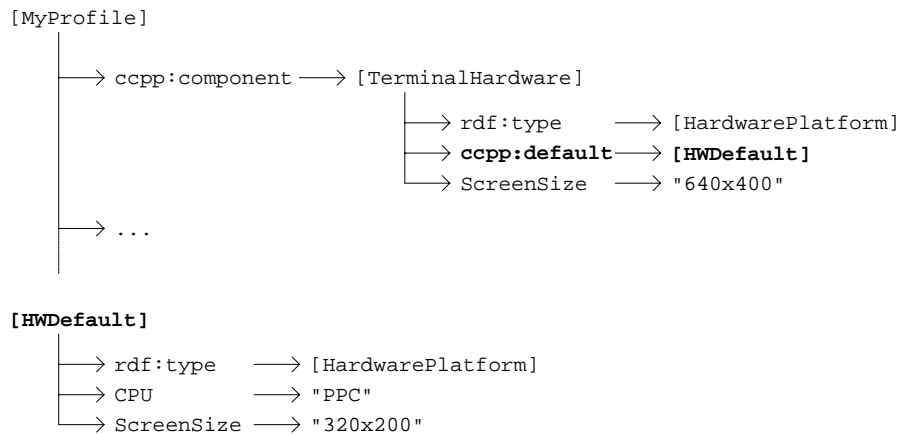


Figure 4.5 – Example of defaults [40].

This use of an externally referenced graph which is used to describe default properties makes possible some important optimizations. As a separate document, it can reside at a separate location and it can be separately cached. This is particularly useful in wireless environments such as cellular networks, where the profiles may be large and the link connected to the client may be slow and expensive.

Defaults can be encoded inline or as separate documents referred to via a URI. It is the responsibility of any server interpreting a CC/PP profile to combine profiles with any externally referenced Defaults in such a way that it will be able to correctly interpret the profile. A profile with inline Defaults is logically equivalent to a profile with the same non-default data and an externally referenced Defaults document. Figure 4.5 shows a simple graph profile using Defaults.

4.2.1.2 Extensibility and Namespaces

Fundamental to the design of the CC/PP framework is the idea that new client attributes can be defined, as needed, through the introduction of new vocabularies. New vocabularies are introduced through XML namespaces.

Vocabulary extensions are used to describe more detailed information that cannot be described using the core vocabulary. Any application or operational environment that uses CC/PP may define its own vocabulary extensions, but wider interoperability is achieved if vocabulary extensions are defined that can be used more generally. Any CC/PP expression can use terms drawn from an arbitrary number of different vocabularies; so there is no restriction caused by re-using terms from an existing vocabulary rather than defining new names to identify the same information.

```
<RDF xmlns="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"

  xmlns:uaprof="http://www.wapforum.org/UAPROF/ccppschema-19991014#" >
  <Description about="http://www.example.com/MyProfile">
    <ccpp:component>
      <Description about="http://www.example.com/TerminalHardware">
        <type resource="http://www.example.com/Schema#HardwarePlatform" />

        <uaprof:ScreenSize>320x200</uaprof:ScreenSize>
      </Description>
    </ccpp:component>
    <ccpp:component>
      <Description about="http://www.example.com/TerminalSoftware">

        <uaprof:OSName>EPOC</uaprof:OSName>
        <uaprof:OSVersion>2.0</uaprof:OSVersion>
        <uaprof:OSVendor>Symbian</uaprof:OSVendor>
      </Description>
    </ccpp:component>

    <Description about="http://www.example.com/Browser">
      <type resource="http://www.example.com/Schema#BrowserUA" />
      <uaprof:BrowserName>Mozilla</uaprof:BrowserName>
      <uaprof:BrowserVersion>5.0</uaprof:BrowserVersion>
      <uaprof:CcppAccept>

        <li>text/plain</li>
        <li>text/vnd.wap.wml</li>
      </Bag>
    </uaprof:CcppAccept>
    </Description>
  </ccpp:component>
</Description>
</RDF>
```

Figure 4.6 – RDF representation of the Profile in Figure 4.4 [40].

The CC/PP framework takes advantage of the XML namespace mechanism. Consider the namespace declarations from the example in Figure 4.6. The first two namespaces declarations (with the `xmlns` command) are for RDF usage. The third declaration names a CC/PP core attribute vocabulary and the corresponding classes; it includes the complete vocabulary specified in the CC/PP framework. The fourth namespace declaration includes application domain specific vocabulary. This is the case when an extension is made on top of the CC/PP core vocabulary. In this case, the User Agent Profile (UAProf) group of the WAP Forum defines the domain specific vocabulary [50].

Any application may define its CC/PP vocabulary extension and publish it under a certain URL. For a coherent use of the vocabulary, it must be unambiguous. Moreover the interpretation of each name from the vocabulary must be unique. Many extension vocabularies can be drawn from existing applications and protocols; already existing vocabularies are WAP UAProf [50] and IETF CONNEG [27].

4.2.2 Attribute Vocabularies

This section describes the basic data types that are available for the values associated to a CC/PP attribute name. All CC/PP attributes must be defined with values that can be treated as a *simple attribute value* or as a *complex attribute value*. All simple CC/PP attribute values are represented as literal text values (in XML elements or XML attributes, according to the rules for RDF literal object values). In addition to the simple data values, a CC/PP attribute may have a complex value expressed in the form of a resource with its own collection of RDF properties and associated values.

Data types defined in CC/PP are presented in Table 4.1.

Table 4.1 – Attribute data type vocabularies [40].

Data type		Description
Simple data type	URI	A URI is represented as a text string, but is subject to comparison rules set out in IETF RFC 2396, which may require the 'absolutization' of the URI.
	Text values	A text value is a string that is used to describe or identify some specific feature value.
	Tokens	Tokens are case insensitive text values using a constrained subset of US-ASCII characters, generally used to name an enumerated set of values.
	Integer number	Integer numbers may be positive, zero or negative.
	Rational number	A rational number is expressed as a ratio of two integer numbers.
Complex data type	Set of values	A set of values consists of zero, one or more values, all different and whose order is not significant.

4.2.3 Client Profiles

A client profile follows the standard CC/PP profile organization with components and attributes. A CC/PP client profile describes user environment characteristics (or capabilities) hierarchically by using first the components and after the attributes. A vocabulary specific to the application may have to be defined as explained in the previous section.

When using names from the core vocabulary or an extension vocabulary, it is important that all system components (clients, servers, proxies, etc.) that generate or interpret the names, apply a common meaning to the same name. For this, it is a good policy to use the same name to refer to the same feature in different components, even if they do not belong to the same application. This procedure increases the inter-working possibilities across applications that use capability information. The core vocabulary is intended to structure the profile, define some attributes data types and add support to active proxies.

CC/PP does not define any set of features to describe information concerning the user device capabilities. These features must be defined in an extension vocabulary. The CC/PP specification defines however a vocabulary intended to describe the printing, display and fax capabilities of a device, see section 4.2.3.1. This is a very limited vocabulary and is defined for compatibility reasons with CONNEG [27].

4.2.3.1 Client Vocabulary

The vocabulary defined here is not a mandatory part of the core CC/PP format specification. This vocabulary is defined for use by CC/PP aware applications that may need to describe certain common features. Designers of CC/PP applications who need to describe such features are encouraged to use this vocabulary rather than defining new terms.

Table 4.2 – CC/PP client attribute vocabulary [40].

Attribute	Description
deviceIdentifier	A URI that serves as an identifier of the client device or user agent type.
type	A MIME content type that can be accepted and presented by a client.
schema	A URI that identifies a schema that is recognized by the client.
charWidth	The width of the character display.
charHeight	The number of lines of text that can be displayed.
charSet	A character set that can be rendered.
pix-x	The number of horizontal pixels that can be displayed.
pix-y	The number of vertical pixels that can be displayed.
color	An indication of the color capabilities.

The client attribute vocabulary defined in Table 4.2 may be used to identify some common properties associated with client devices that print or display visual information, such as text or graphics. The vocabulary is described using an XML namespace, published under the `<http://www.w3.org/2000/07/04-ccpp-client#>` XML namespace identifier.

4.2.4 Proxy Support

It may be that an intervening network element, such as a customization proxy, has to enhance the client profile with capabilities that it is able to execute on behalf of the client. For instance, if a client is only able to receive GIF images, a customization proxy may be able to convert JPEG images to GIF images, and so the user profile must be enhanced with the extra customization proxy capabilities. CC/PP offers two mechanisms to ease this process:

- **Capability chaining:** a proxy's role as a content modifying component between client and server is represented by chaining a description of the proxy's behavior to the downstream client description. For any given request containing a CC/PP profile, a new profile is dynamically created that refers to a CC/PP description of the proxy and to the CC/PP capabilities in the request received by the proxy. For this purpose, the "proxyProfile" and "nextProfile" properties are introduced. In Figure 4.7, the proxy profile capability chaining is illustrated with the use of these two properties to include both the client and the proxy profiles in the request. The proxy profile may include hardware and operating system components as well but, in practice, it is unlikely that there is any advantage for the proxy to advertise this type of characteristics.
- **Proxy behavior:** a proxy may convert or interpret data for a client (add capabilities), or impose policy constraints (block capabilities). For that purpose, the properties "proxyAllow" and "proxyBlock" are used to add or remove capabilities, respectively.

Figure 4.7 shows a full example of both capability chaining and proxy behaviour functionalities. “MyProxyProfile” lists its transcoding capabilities as a property of “proxyBehaviour”. Structurally, a proxy profile mirrors the structure of a client profile, in terms of the components to which proxy behaviors apply. Designers of CC/PP applications that need to deal with proxy behaviors may define their own schemas to describe the characteristics of their proxy but they are encouraged to use the vocabulary in the next section rather than define new structures.

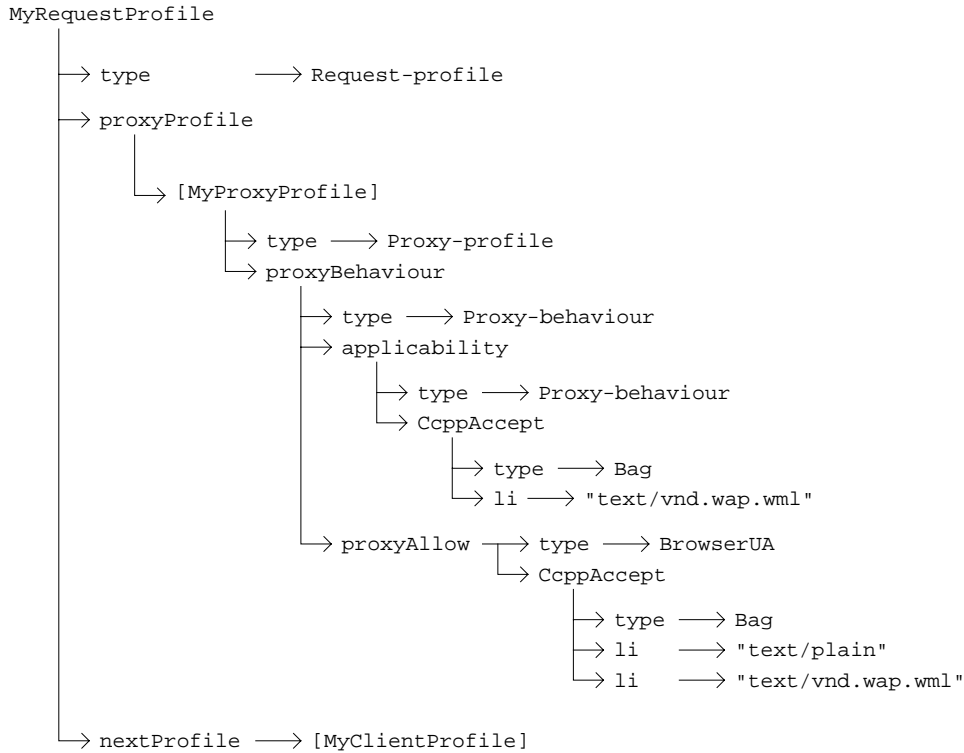


Figure 4.7 – Example of a request profile chain [40].

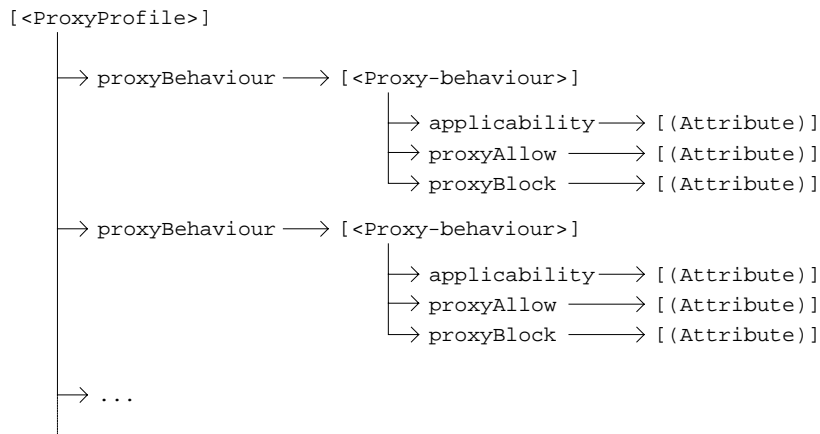


Figure 4.8 – Example of proxy behavior description [40].

4.2.4.1 Proxy Vocabulary

The proxy vocabulary defined here is not a mandatory part of the CC/PP specification, but is defined for use by CC/PP aware applications that may need to deal with proxies which may play an active role in content handling. For the purpose of the CC/PP specification, a proxy is a component that sits on the network path between a client and an origin server, and modifies or filters the content passed towards the

client. This in turn affects what the origin server may provide in response to a given client request, so the CC/PP information needs to be augmented with information corresponding to the proxy's behavior.

The core proxy vocabulary names below are described using XML namespace local parts, which are further qualified by the XML namespace identifier `<http://www.w3.org/2000/07/04-ccpp-proxy#>`.

Table 4.3 – Proxy vocabulary names [40].

Attribute	Description
Profile	This class represents any CC/PP profile that can be delivered to an origin server.
Request-profile	This is a subclass of the CC/PP profile that is used to link a proxy profile to a downstream request or client profile.
Proxy-profile	This class represents the capabilities and filtering behavior of a given proxy.
Client-profile	This class represents the capabilities of a given client.
proxyProfile	This property is applied to a Request-profile instance, and indicates a Proxy-profile that is applied to the CC/PP profile associated with the corresponding request.
nextProfile	This property is applied to a Request-profile instance, and indicates a downstream Request-profile or Client-profile with which new proxy behavior is combined.

Table 4.4 – Proxy behavior attributes [40].

Attribute	Description
Proxy-behaviour	This class represents a description of a single aspect of a proxy's behavior.
proxyBehaviour	This property is applied to a proxy capability description, and references a Proxy-behavior instance.
Applicability	Indicates a set of component values that must exist in a request in order a proxy-behavior is applied to the concerned request. If the "Applicability" values do not match those available in the request then this proxy-behavior is ignored.
proxyAllow	Indicates a Component value that specifies one or more attribute values that are included by the corresponding Proxy-behavior in the CC/PP capabilities of a request.
proxyBlock	Indicates a Component value that specifies one or more capability attributes that are removed from the CC/PP capabilities of a request, if present.

4.3 Wireless Application Protocol (WAP) 2.0

First mobile services had to deal with serious limitations at different levels, bandwidth being the most critical one. New protocols had to be developed in order to adapt existing protocols to the restrictions of mobile networks. The Wireless Application Protocol Forum (<http://www.wapforum.com>) objective is to create specifications that give a solution to the above problem. First version of WAP was released in 1998, while for third generation mobile networks, WAP Forum released version 2.0 in November 2001 [48].

WAP specifications cover many topics, ranging from Multimedia Messaging Services (MMS) to transport level security; however, in terms of UMA applications, the following three items are the most relevant:

- **Architecture:** provides a perspective on how the servers existent in a mobile network system interwork;
- **Wireless Application Environment (WAE):** allows the development of applications by providing an environment with a set of tools capable of handling certain content, security, messaging, etc;

- **User Agent Profile (UAProf):** extension to the CC/PP framework that enables a WAP device to describe itself to some proxy or content server.

Besides of the importance of WAP as a global system, it is also the most complete system where the problem of user environment description had to be solved in practice. For this reason, the WAP solution will be analyzed in the following.

4.3.1 Architecture

Whenever possible, existing standards have been adopted as the starting point for the WAP technology. Moreover existing solutions have been adopted with optimizations and extensions in order to match the characteristics of the wireless environment. The first example of technology re-use is illustrated in Figure 4.9: the WAP functional model enhances the Web model by providing a push mechanism while maintaining the same request/response mechanism between the device and the Web server.

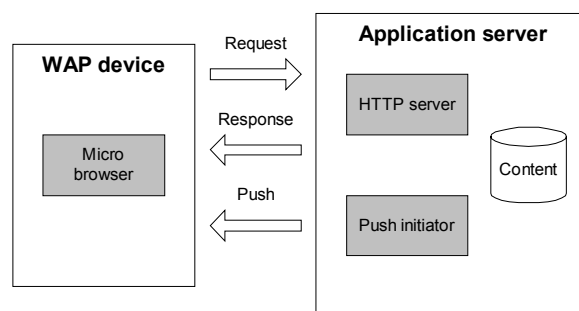


Figure 4.9 – WAP functional model [48].

In the first releases of the WAP specifications, a very restrictive solution was designed due to the very though limitations of GSM and GPRS networks: for this, a very optimized protocol stack was specified. With third generation mobile networks, limitations are not so tight, and so more common protocols can be used, more precisely the Wireless-TCP/IP and Wireless-HTTP protocols. Figure 4.10 illustrates the WAP version 2.0 protocol stack. In WAP version 2.0, the standards HTTP, TCP and IP protocols have been adopted in their wireless versions, since they provide a more efficient use of the available bandwidth by using lossless compression techniques to decrease the size of the transmitted packets.

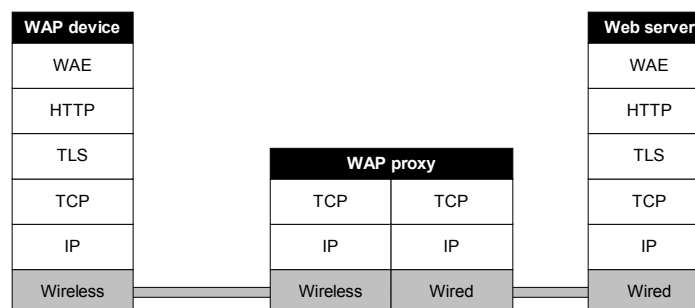


Figure 4.10 – WAP protocol stack [48].

In terms of the network architecture, WAP defines some services that need to be available to be able to take benefit of some of the WAP functionalities. The most important service is the User Agent Profile repository, which is used to store the user profiles; this avoids sending and receiving the profile/description for every request.

4.3.2 Wireless Application Environment (WAE)

The Wireless Application Environment [49] provides a general-purpose application environment based on a combination of the World Wide Web, Internet and mobile telephony technologies. The primary objective of the WAE is to establish an interoperable environment allowing operators and service providers to build applications and services that can reach a wide variety of different wireless platforms in an efficient and useful manner.

WAP defines a markup language that is targeted to devices with limited capabilities: the Wireless Markup Language (WML). Thus, any content received by a WAP device must come within a WML page.

The WAE most relevant elements are:

- **User-Agent:** micro-browser environment containing or allowing for markup (including WML and XHTML), scripting, style-sheet languages, and telephony services and programming interfaces, all optimized for use in hand-held mobile terminals.
- **Content Formats:** support a set of well-defined data formats, such as color images, audio, video, animation, phone book records, and calendar information.
- **Push:** provides a general mechanism for the network to initiate the transmission of data to applications resident on WAP devices.
- **Multimedia Messaging:** the Multimedia Message Service (MMS) provides for the transfer and processing of multimedia messages such as email and instant messages to WAP devices.

Applications running in the WAE are responsible for requesting the content; for each request, the device profile is sent together.

4.3.3 User Agent Profile

The user agent profile (UAProf) was designed to describe the characteristics of the device, its preferences for certain content formats, and user preferences related to WAP specific services [50]. The description features include the hardware and software characteristics of the device as well as information about the network to which the device is connected. The user agent profile includes information useful for content formatting purposes and although not a full solution it is the most complete solution currently available to address the problem of user environment description in the context of UMA applications.

The user agent profile extends the CC/PP framework by defining a set of components, its attributes and the semantics of these attributes for WAP terminals. The CC/PP extensibility guidelines presented previously in this Chapter are also valid for UAProf to assist in extending the schema with new components.

The UAProf specification [50] assumes that:

- The information contained within the profile is provided in the HTTP request on behalf of the user who will be receiving the content contained in the corresponding HTTP response.
- The CC/PP repository storing the profile is secure, meaning that it does not permit unauthorized modification to corrupt profile information. The functionality associated with this repository may be implemented in a WAP gateway or intermediate proxy or as a separate standalone element in the network.

- An implicit chain of trust exists between the user terminal and the origin server. The integrity of the profile is maintained (in other words, not compromised) as it is transmitted through or cached within the network. It is assumed that the network elements that contribute with descriptions to the profile are trusted (through CC/PP profile chaining capability). Network elements will not assemble a "history" about users by tracking the deltas in their profile over time.

Use of the profile by the origin server is intended to enhance the user's experience. Therefore, should the profile be discarded, corrupted, or otherwise inaccessible, the origin server will still provide and deliver content to the user in a best-efforts fashion.

Because WAP User Agent Profiles are based on CC/PP, they include a set of components and attributes; the major key components and the corresponding description features are presented in the following:

- **HardwarePlatform:** a collection of properties that adequately describe the hardware characteristics of the terminal; this includes the type of device, model number, display size, input and output methods, etc.
- **SoftwarePlatform:** a collection of attributes associated with the operating environment of the device; these attributes provide information on the operating system software, video and audio encoders supported by the device, and user's preference on language.
- **BrowserUA:** a set of attributes to describe the HTML browser application
- **NetworkCharacteristics:** information about the network-related infrastructure and environment such as bearer information; these attributes can influence the resulting content, due to the variation in capabilities and characteristics of various network infrastructures in terms of bandwidth and device accessibility.
- **WapCharacteristics:** a set of attributes pertaining to WAP capabilities supported on the device; this includes details on the capabilities and characteristics related to the WML Browser, etc.
- **PushCharacteristics:** a set of attributes pertaining to Push specific capabilities supported by the device; this includes details on supported MIME-types, the maximum size of a push-message shipped to the device, the number of possibly buffered push-messages on the device, etc.

The User Agent Profile is transmitted for every request in two possible ways:

- Passing the profile entirely in the request;
- Passing the URL of the location where the profile is and adding just the differences, if any;

UAPProf can be processed by any network element (e.g. proxy or gateway) through which the request passes; that network element can use the CC/PP mechanisms to modify the original profile in order to build a customized response to the request. WAP User Agent Profiles use the extension to the CC/PP core vocabulary presented above to satisfy the specific requirements of WAP terminals. UAPProf does not go much far in terms of characteristics related to user preferences or natural environment, since it is a solution much focused on the limited resources mobile devices environment.

4.3.4 CC/PP Technology with WAP

This section will present the use of CC/PP technology with WAP. The WAP Forum architecture is based on a proxy server, which acts as a gateway to the optimized protocol stack for the mobile environment. The WAP gateway is the bridging point between the WAP device and the backbone network where the application servers and services are located. On the wireless side of the communication, there is an optimized protocol to establish service sessions, and an optimized transmission protocol (W-TCP and W-HTTP); on the fixed side of the connection, the protocol used is HTTP.

UAProf is used to perform content adaptation in a content server or in an intermediary server. The following examples illustrate the use of CC/PP and UAProf in both situations:

- **Content server adaptation:** in case the content descriptions exist, there will be some processing inside the content server, as the user profile information needs to be matched with the content description to check if the terminal is able to receive that content (or if some adaptation is needed). The processing at the server is not defined since CC/PP does not standardize the matching algorithms. This situation is shown in Figure 4.11, where no WAP gateway (or proxy) is included in the diagram.

The sequence of the process is:

1. HTTP request with device description.
2. Content description is matched against device description to decide on the best content representation.
3. Content is adapted, if needed.
4. HTTP response to user device with adapted content.

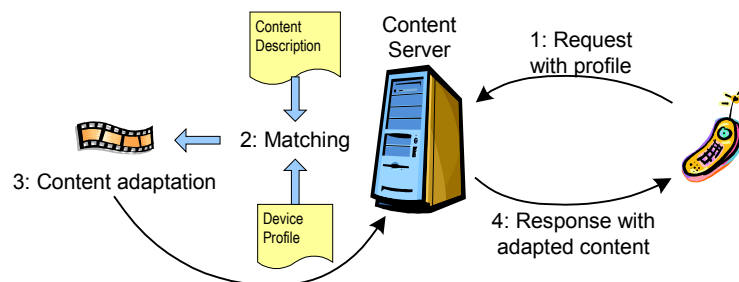


Figure 4.11 – Adapting content to a certain user profile.

- **Gateway adaptation:** this usage case involves a mixed architecture with an active proxy in the network. The use of CC/PP profiles in the WAP UAProf was already described in this chapter. When a User Agent Profile is used with a WAP terminal, the following sequence of steps happens, see Figure 4.12:
1. HTTP request with device description or difference relative to a specified default device description.
 2. WAP Gateway caches HTTP header and composes the final terminal description (using the cached header as defaults, and differences from the client); UAProf values can change at setup or resume of session.

3. Gateway passes request to server.
4. Server returns content (with content description, if it exists).
5. Gateway adapts the content according to the device description (and content description, if it exists).
6. Response by the WAP gateway in W-HTTP with adapted content.

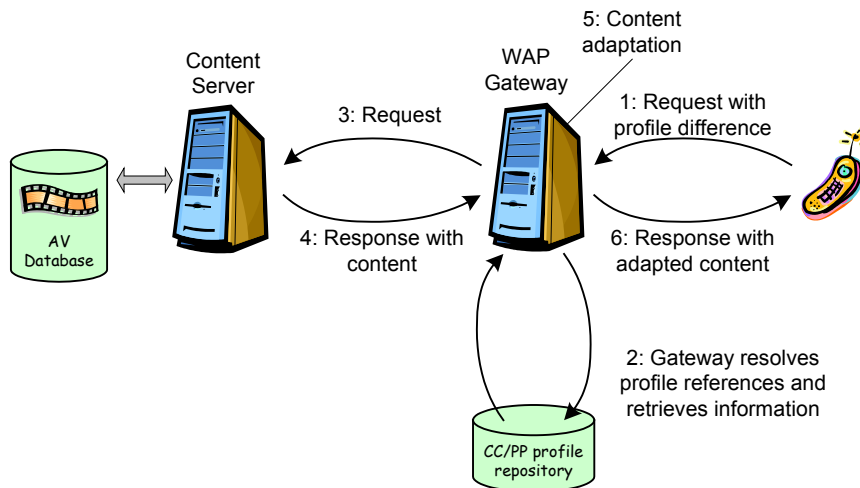


Figure 4.12 – Adapting content in a WAP context.

WAP specifications provide a good idea on how a UMA System must interact with the user terminal in a Web scenario. On the other hand, CC/PP offers a good framework for user environment description and provides some mechanisms that are crucial in wireless links (defaults and capabilities chaining).

4.4 MPEG-21 Multimedia Framework: Digital Item Adaptation

Currently, the end-user has multimedia content available through any terminal, network, anywhere and anytime even if in many cases he does not have the resources required to access certain types of content. To reach the end-user, the content has to travel a value-chain that starts in the content creator and passes through several other users that have their role in the task of getting the content to the end-user. All these users make use of multimedia technology to improve its working process, and each one of them has its own domain of action. A domain of action is understood as the community where the user, understood as any player in the content chain, “lives”: in a user’s community there are certain rules, procedures, interests and even content formats. However, there is today no way that all users and/or communities may communicate and interact in an efficient way, notably regarding multimedia information. Thus a common framework is needed in order to improve the co-operation between users and support a more efficient implementation of the models, rules, procedures, interests and content formats. MPEG-21³ is an initiative of MPEG formally called “Multimedia Framework” proposes to build and specify this framework [117].

The MPEG-21 framework is intended to be a framework for multimedia delivery and consumption, where its users may interact and exchange multimedia content. In MPEG-21, the main elements are the *User* and the *Digital Item*, Figure 4.13. An MPEG-21 user is any entity that interacts within the framework: it can be a content creator, a content distributor, or a content consumer (the user can have any function

³ MPEG-21 work started in October 1999 and the parts under development are planned to end in May 2003.

that is announced by its description). In order for users to be able to interact, interfaces must be defined so that transactions, rights of use, content, value, authorization and other information be exchanged. Aggregated with the content that users exchange there is also its corresponding description and unique identification (through secure mechanisms such as watermarking); this set of elements is the so called *Digital Item*.

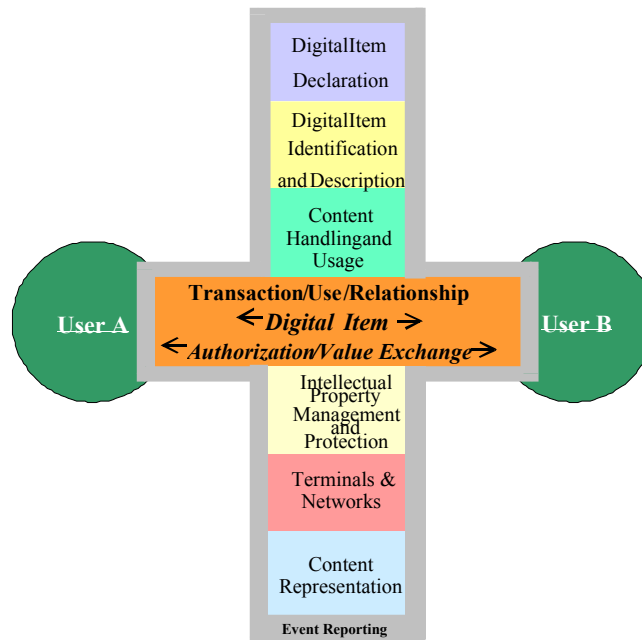


Figure 4.13 – User's role in the MPEG-21 framework [117].

The MPEG-21 Multimedia Framework has identified seven key elements needed to support the multimedia delivery chain, the relationships between users and the operations supported by them:

- **Digital Item Declaration:** uniform and flexible abstraction and interoperable schema for declaring Digital Items;
- **Digital Item Identification and Description:** framework for identification and description of any entity regardless of its nature, type or granularity;
- **Content Handling and Usage:** provides interfaces and protocols that enable creation, manipulation, search, access, storage, delivery, and (re)use of content across the content distribution and consumption value chain;
- **Intellectual Property Management and Protection:** defines the means to enable content to be persistently and reliably managed and protected across a wide range of networks and devices;
- **Terminals and Networks:** relates to the ability to provide interoperable and transparent access to content across networks and terminals;
- **Content Representation:** relates to the way media resources are represented;
- **Event Reporting:** metrics and interfaces that enable the users to understand more precisely what is happening within the framework;

The framework elements represent the various areas where standards are needed to guarantee the transparent and augmented use of multimedia resources across a wide range of networks, devices, and communities, notably for trading (of bits). For this, MPEG-21 will consider three phases [117]:

- **Define big picture:** understanding how the components of the framework are related and identify where gaps in the framework exist;
- **Fill the gaps:** developing new specifications where needed;
- **Integration:** achieving the integration of standards to support harmonized technologies for the management of multimedia content;

Whenever standards already available fulfill the identified requirements for a certain problem, they will be used instead of defining new ones; for example, the content may be encoded in JPEG, MPEG-2, MPEG-4, etc. and the content description may be formatted using MPEG-7.

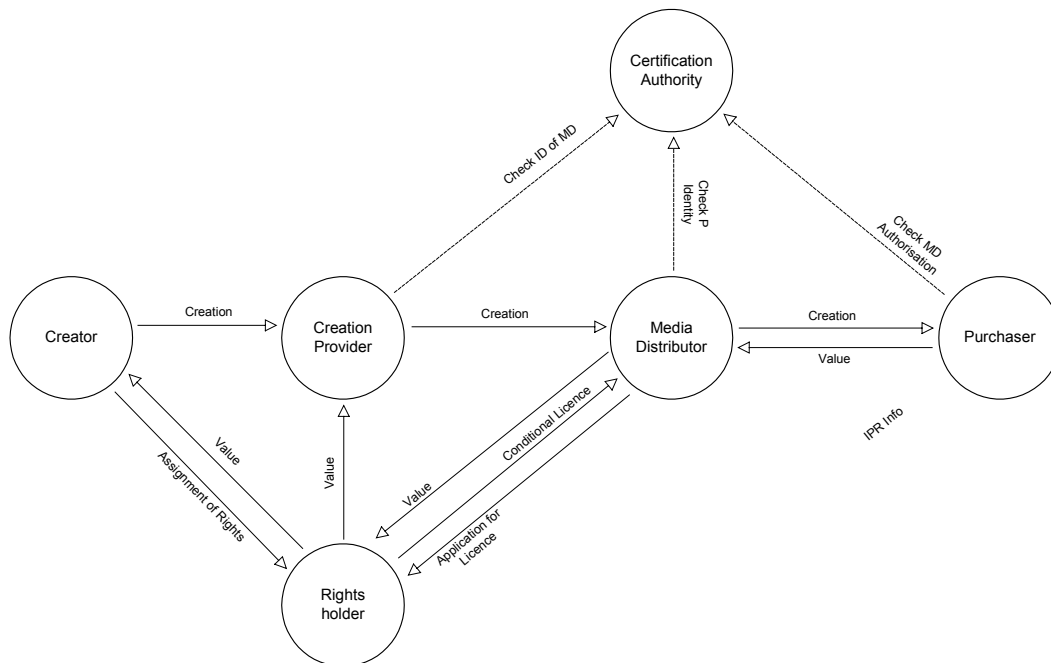


Figure 4.14 – Example of MPEG-21 scenario with several users [117].

Figure 4.14 illustrates an MPEG-21 framework where several users interact. In this example, the end-user, the **Purchaser**, wishes to buy or access some creation, a Digital Item, that has been created by another user in the MPEG-21 framework, the **Creator**. This user has delivered its Digital Item, to some entity that will be responsible for providing the Digital Item to any other user that may want it, this user is the **Creation Provider**. When the Purchaser contacts the Creation Provider and makes the deal, another user enters in the value-chain to transport the Digital Item from the Content Provider to the Purchaser. This user is responsible for distributing Digital Items to the users retrieving it; this is the **Media Distributor**. In this value-chain, other types of users must exist to enforce the Intellectual Property protection and to assure that the Digital Item that the end-user receives is a legitimate copy of the original work; these users are, respectively, the **Rights Holder** and the **Certification Authority**. This example serves only the purpose of explaining how the MPEG-21 concepts should be used in a real scenario.

MPEG-21 has established a work plan regarding the technological gaps where further standardization is needed which considers at the moment six activities that will correspond to Parts 2 to 7 of MPEG-21; Part 1 corresponds to a Technical Report where the vision and objectives of MPEG-21 are presented. The seven parts of the MPEG-21 standard currently adopted are:

- **Part 1, Vision, Technologies and Strategy:** describe the multimedia framework and its architectural elements together with the functional requirements for their specification;
- **Part 2, Digital Item Declaration:** presents the basic definitions and concepts to form a useful model for defining Digital Items;
- **Part 3, Digital Item Identification and Description:** specifies how to uniquely identify and describe Digital Items;
- **Part 4, Intellectual Property Management and Protection (IPMP):** defines an interoperable framework for Intellectual Property Management and Protection;
- **Part 5, Rights Data Dictionary:** defines a dictionary of key terms which are required to describe rights of users, including intellectual property rights;
- **Part 6, Rights Expression Language:** specifies a machine-readable language that can declare rights and permissions using the terms defined in the Rights Data Dictionary;
- **Part 7, Digital Item Adaptation:** defines tools relevant for the adaptation of Digital Items under any conditions.

Part 2: Digital Item Declaration (DID) defines the model and the syntax and semantics that must be used to define Digital Items. Within the DID model, a Digital Item is the digital representation of “a work”, and as such, it is the *thing* that is acted upon (coded, managed, described, exchanged, collected, paid for, etc.) within the model. The DID is defined through a normative XML schema comprising the entire grammar of the Digital Item Declaration representation in XML.

Part 3: DIID defines a framework for Digital Item identification (how to assign a unique ID to a Digital Item) and description.

Parts 4, 5 and 6 are all mostly related with the management of rights and intellectual property aspects that go beyond the scope of this thesis, although there are critical rights’ issues associated to content customization, e.g. will the content creators or rights holders accept any type of customization to be performed on their content?

Part 7, Digital Item Adaptation started as the result of the discussion made during a few months in the UMA Ad-hoc group where the author of this thesis very actively participated in the definition of the DIA requirements [122]. After this activity, MPEG recognized the importance to address the problem of user environment description and started to gather the corresponding requirements. Initially this part was to be called Digital Item Usage Environment Description (DIUED) but later its scope was expanded in order to embrace any tool that can be used to perform content adaptation. Of course, user environment description tools are still the core of the Call for Proposals issued in order to collect technology to start the MPEG specification process. It is also important to highlight that since MPEG-21 is about the easy trading of multimedia content, all the Digital Item adaptation tools to be specified have to be particularly powerful and efficient regarding this type of content.

Although some parts of the MPEG-21 standard are already close to finalized, Part 7 is at a very initial phase. After the Call for Proposal issued in December 2001, a final Call for Proposals was issued in March 2002 as an output of the 59th MPEG meeting. Between the 1st and the 25th April 2002, the MPEG committee will receive the proposals answering to the Call. At the 60th MPEG meeting, in May 2002, the proposals will be evaluated. As an output of this meeting, the first version of the DIA Working Draft will be issued; the second version will be issued at the 61st MPEG meeting and the third version at the 62nd MPEG meeting in October 2002. The Committee Draft will be published in December 2002, at the 63rd MPEG meeting, and the Final Committee Draft at the following meeting. Finally, MPEG-21 Part 7: Digital Item Adaptation will reach the Final Committee Draft status in July 2003 at the 64th MPEG meeting.

Although it is expected that many proposals will be made to MPEG in the sequence of the Call for Proposals, it is clear that the CC/PP technology with the WAP UAProf vocabulary would be a good starting point for developing a complete multimedia user environment description solution. Since UMA scenarios may consider any type of user environments (e.g. Web TV, Workstation, mobile terminal, etc.), the UAProf vocabulary must be extended and new characteristics must be added. In the following section, a proposal will be made for a user environment description solution, which uses the WAP UAProf specification as the starting point.

4.5 A DIUED Proposal for MPEG-21 DIA

One of the major problems felt at the starting of this thesis has been the lack of a standard, and rather complete solution, for the description of user environments. With this solution, context-aware applications can be much improved and new applications may develop. Following the recognition of this need, the author of this thesis made a first technical proposal to MPEG [124], in July 2001, to address the problem in the context of the MPEG-7 standard. At that time, MPEG recognized the technical value of the proposal but did not want to address the user environment description problem in the context of MPEG-7 since it considered MPEG-7 to exclusively address content description issues. Anyway following this and other contributions, and the interest developed within the group, MPEG decided to create Part 7 of the MPEG-21 standard to address first the issue of Digital Item Usage Environment Description and finally the issue of Digital Item Adaptation. As a consequence, the proposal made in this section is already an evolution of the proposal made in the context of MPEG-7, to consider the more wide scope of the MPEG-21 standard, notably the broad understanding of the concept of *user*.

In this context, the proposal by the author of this thesis is a set of descriptors, using XML Schema, that should enable the MPEG-21 standard to provide the first complete and generic user environment (or usage context) description framework where all the user environment aspects are described and thus can be used in very different applications. The proposal considers two basic description dimensions: consumer and service provider, which will be motivated in the following.

From the previous section, it became clear that users must interact and must be aware of each other. In order for one user to know which functionalities another user may offer, the later provide a description of his characteristics, notably in terms of functions provided. Although, the end-user or consumer case is typically the most common, a powerful multimedia framework must consider other types of users, notably those that are not consuming but providing a service, e.g. transcoding or summarization. Therefore, this proposal separates the user's function into two major categories: consumer and service provider. Under the service provider category, there may exist almost any type of user: For example, a UMA Engine may be a user in the framework that offers the function of customizing content for the other users. Moreover, the same user may behave as a consumer and as a service provider simultaneously, implying that the same user should provide description(s) including the two roles that he can perform. Therefore a user is classified in two distinct functions and thus description dimensions:

- **Consumer:** this dimension applies to the user that receives content for consumption and wishes to have the content tailored to its user environment, the so-called *end-users*.
- **Service provider:** this dimension applies to the users that do not consume content as end-users but perform any action regarding the content, in principle easing the consumption task for an end-user. A service provider can be a user that customizes content, authenticates content, distributes content and so on.

In some circumstances it may be relevant to consider the notion of *community* understood as an interacting group of users sharing something in common that is defined in terms of user characteristics (e.g. preferences or demographic features). In this case defining the community characteristics, e.g. group of users that like cowboys movies, like sports and have less than 30 years or even that like food made with cod fish (if not only content-related preferences are considered). Communities are associated with a ‘specialization’ of users’ interests; thus, through a community a user, consumer or service provider, may more easily find and contact other users with whom he may exchange valuable information, multimedia content or not. To facilitate this type of functionality, a user description may include a field listing the communities to which that user belongs.

A possible service provided by a Service Provider may be precisely making available the membership of these communities; these membership lists may be sold or made public if built based on user descriptions for which permission was given for the corresponding users to be listed as part of the community in question. From a service provider point of view, this notion of community may be rather powerful, e.g. to inform a highly specialized public (e.g. community of professional photographers) about the services provided that can indeed be very specialized (e.g. printing digital photos into large posters or the arrival of a new type of camera).

Figure 4.15 illustrates the proposed schema definition, covering the two description dimensions presented. The proposal is fundamentally a DIUED proposal targeting the user environment description for content customization purposes; however other requirements are also covered by the proposal although at a rather basic level. This means that some place holders for future descriptors may be created, indicating that this proposal is prepared to be combined with other proposals which consider in detail other user environment description areas such as digital rights management.

The proposal is not based on CC/PP but on XML Schema since MPEG already adopted XML Schema as the basis for the MPEG-7 Description Definition Language (DDL) and for the MPEG-21 Digital Item Declaration Language (DIDL). One of the advantages of using XML Schema for MPEG-21 DIA is that other XML based user environment descriptions, that are not compatible with the framework defined here (e.g. WAP UAProf), can easily be transformed into an MPEG-21 DIA compliant syntax (not semantics). This is an advantage since XML Schema based languages are becoming very popular.

The following sections will present the structures and data types that are defined by the DIUED Schema definition to characterize the user environment.

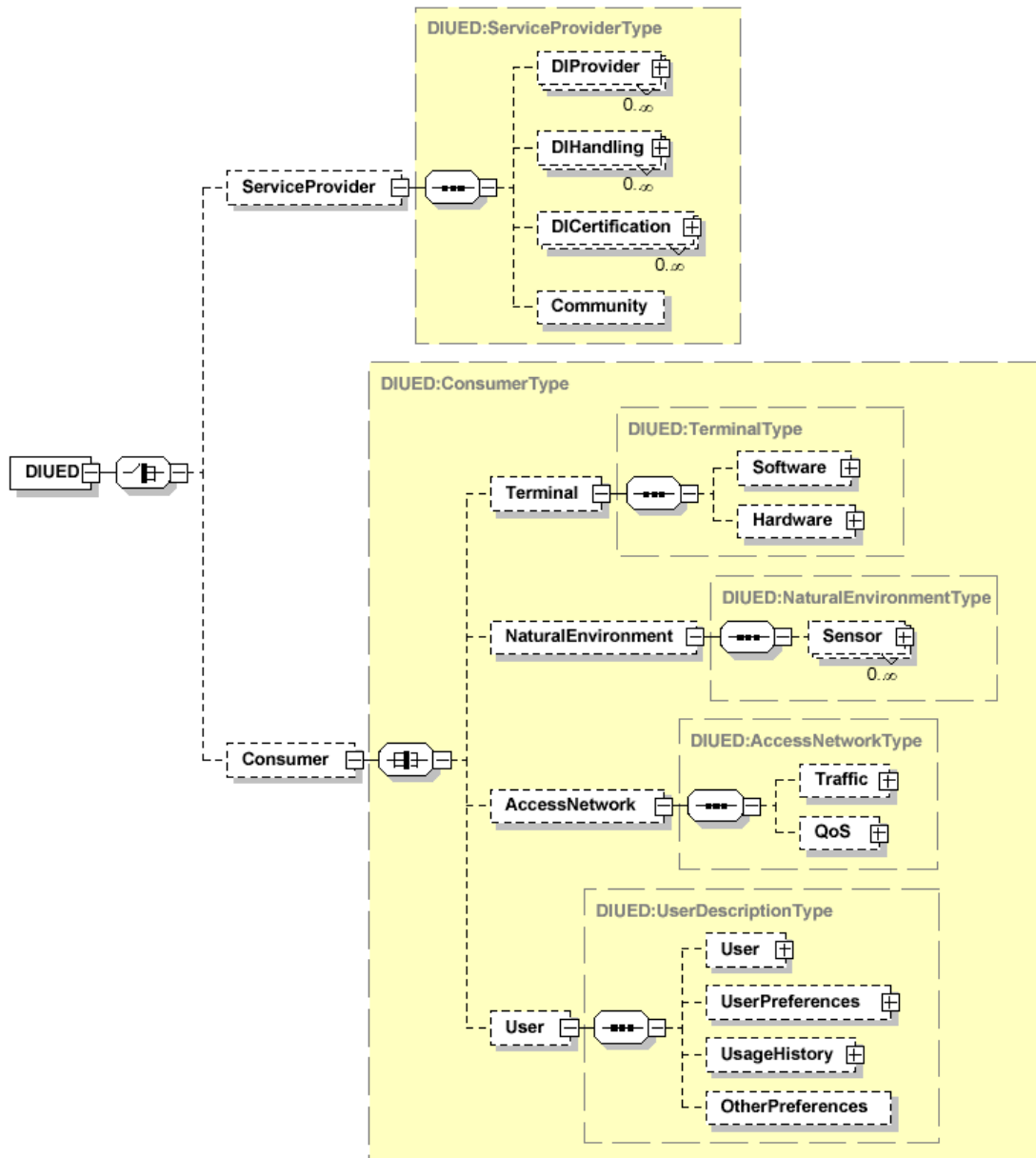


Figure 4.15 – Proposed DIUED schema (only the consumer dimension is expanded).

4.5.1 Schema tools

The DIUED Schema defines a set of descriptors that can be used to characterize the user environment. This schema defines the basic tools that must exist to create the description.

MPEG-7: Part 5, Multimedia Description Schemes, offers many tools that can be used in the DIUED Schema, such as the user preferences; therefore the MPEG-7 MDS Schema are imported⁴ into the DIUED Schema definition. In this context, most of the tools are inherited from the MPEG-7 MDS Schema, and only a new root element is defined in the DIUED Schema.

⁴ In XML Schema two schemas can be linked by `import` (the imported schema will be known as a sub-schema) and `include` (both schemas work at the same level, the included schema will act as the basis for the new extension).

4.5.1.1 DIUEDType

The initial wrapper or root element of every description identifies the type of description that is inside it; in this case, DIUEDType is the root element for the DIUED Schema. DIUEDType contain a description of one of the two types entities that were defined: the consumer, and the service provider. It is possible, for a user to include in its description the communities to which he belongs; this is possible through each user respective descriptor. When a user has multiple functionalities that cannot be included in the same description, he uses the description that is more adequate to the situation (e.g. consumer or service provider).

The DIUED fields are listed and its semantics presented in Table 4.5; the DIUED XML syntax is as follows:

```
<!--=====-->
<!-- Digital Item Usage Environment Description -->
<xsd:element name="DIUEDType">
  <xsd:complexType>
    <xsd:choice>
      <xsd:element name="ServiceProvider" type="DIUED:ServiceProviderType" minOccurs="0"/>
      <xsd:element name="Consumer" type="DIUED:ConsumerType" minOccurs="0"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
```

Table 4.5 – DIUEDType element semantics.

Name	Definition
DIUEDType	Root element for the user environment description with the following basic dimensions: <ul style="list-style-type: none"> – Service Provider – Consumer
ServiceProvider	Describes a service provided by a service provider.
Consumer	Describes the end-user environment characteristics.

4.5.2 ServiceProviderType

The ServiceProviderType descriptor is used to describe a user that is acting as a service provider. For the reasons already mentioned, the proposal for this descriptor provides only the basis for other works to proceed in describing this type of users. The present descriptor considers only the characterization of the content provided, of the content handling capabilities (content customization), and of the content certification capabilities.

The ServiceProviderType descriptor fields are listed and its semantics presented in Table 4.6; the ServiceProviderType descriptor XML syntax is as follows:

```
<!--=====-->
<!--Definition of ServiceProvider -->
<xsd:complexType name="ServiceProviderType">
  <xsd:sequence>
    <xsd:element name="DIProvider" type="DIUED:DIProviderType" minOccurs="0"
      maxOccurs="unbounded"/>
    <xsd:element name="DIHandling" type="DIUED:DIHandlingType" minOccurs="0"
      maxOccurs="unbounded"/>
    <xsd:element name="DICertification" type="DIUED:DICertificationType" minOccurs="0"
      maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
```

```

<xsd:element name="CommunityCharacterization"/>
</xsd:sequence>
</xsd:complexType>

```

Table 4.6 – *ServiceProviderType* descriptor semantics.

Name	Definition
ServiceProviderType	Describes a Service Provider.
DIContentProvider	Describes the Digital Items provided by this service provider, in terms of content theme and content description (e.g. sports photos with annotations).
DICertification	Describes the Digital Items certification capabilities of this user (e.g. every digital items provided by this user contains a secure identification, possibly recurring to watermarking techniques, that assures the digital item origin).
DIHandling	Describes the Digital Item handling/delivering capabilities of this service provider in terms of content customization and delivery capabilities (e.g. summarization and real time 128kbit/s streaming, CIF to QCIF adaptation and only download).
CommunityCharacterization	For a service provider that is providing community characterizations, this parameter describes the usage characteristics of the members of that community (e.g. the genre of content they like).

4.5.3 ConsumerType

The ConsumerType descriptor includes the whole set of end-user environment characteristics. It aggregates a set of descriptors that should gather all the relevant information about the end-user environment organized in four main dimensions: Terminal, Access Network, Natural Environment and User Preferences. Note that other fields may be added since the proposed fields do not address all the characteristics mentioned in Section 4.1. For example, descriptors that cover the terminal input capabilities, such as the type of keyboard, the type of pointing device, etc. have not been yet considered.

The ConsumerType descriptor fields are listed and its semantics presented in Table 4.7; the ConsumerType descriptor XML syntax is as follows:

```

<!--=====-->
<!--Definition of Consumer -->
<xsd:complexType name="ConsumerType">
  <xsd:all>
    <xsd:element name="Terminal" type="DIUED:TerminalType" minOccurs="0"/>
    <xsd:element name="AccessNetwork" type="DIUED:AccessNetworkType" minOccurs="0"/>
    <xsd:element name="NaturalEnvironment" type="DIUED:NaturalEnvironmentType" minOccurs="0"/>
    <xsd:element name="User" type="DIUED:UserDescriptionType" minOccurs="0"/>
  </xsd:all>
</xsd:complexType>

```

Table 4.7 – *ConsumerType* descriptor semantics.

Name	Definition
ConsumerType	Specifies the consumer environment characteristics.
Terminal	Describes the consumer terminal characteristics.
AccessNetwork	Describes the consumer access network characteristics.
NaturalEnvironment	Describes the consumer natural environment characteristics.
UserCharacterization	Describes the consumer preferences and eventually other characteristics.

4.5.3.1 TerminalType

The TerminalType descriptor is one of the four components of the user environment; it is designed to describe the user terminal capabilities.

The TerminalType descriptor fields are listed and its semantics presented in Table 4.8; the TerminalType descriptor XML syntax is as follows:

```
<!--=====-->
<!--Definition of Terminal -->
<xsd:complexType name="TerminalType">
  <xsd:sequence>
    <xsd:element name="Software" type="DIUED:TerminalSoftwareType" minOccurs="0"/>
    <xsd:element name="Hardware" type="DIUED:TerminalHardwareType" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
```

Table 4.8 – TerminalType descriptor semantics.

Name	Definition
TerminalType	Describes the user terminal characteristics.
Software	Describes the terminal software related characteristics.
Hardware	Describes the terminal hardware related characteristics.

4.5.3.1.1 TerminalSoftwareType

The TerminalSoftwareType descriptor allows describing the software that is available at the user terminal. For UMA applications, only the software decoding capabilities are of interest. The DRM (Digital Rights Management) field of this descriptor is proposed because this is an important descriptor that will need further work in the future, notably in relation to content customization. The SoftwarePlatform field has been included only for the purpose of compatibility with the WAP specifications.

The TerminalSoftwareType descriptor fields are listed and its semantics presented in Table 4.9; the TerminalSoftwareType descriptor XML syntax is as follows:

```
<!--=====-->
<!--Definition of Terminal Software -->
<xsd:complexType name="TerminalSoftwareType">
  <xsd:sequence>
    <xsd:element name="SoftwarePlatform" type="DIUED:SoftwarePlatformType" minOccurs="0"/>
    <xsd:element name="MediaDecoding" type="DIUED:MediaDecodingType" maxOccurs="unbounded"/>
    <xsd:element name="DRM" type="DIUED:DRMType" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
```

Table 4.9 – TerminalSoftware descriptor semantics.

Name	Definition
TerminalSoftwareType	Describes the terminal software related characteristics.
SoftwarePlatform	Describes the terminal Operating System (e.g. WAP, PocketPC).
DRM	Describes the Digital Rights Management capabilities of the terminal software (e.g. the capabilities of a mobile phone with MP3 player with DRM functionality preventing illegal playing and copying).
MediaDecoding	Describes the content decoding capabilities available at the user terminal (e.g. MPEG-1 format, Flash format).

4.5.3.1.1.1 MediaDecodingType

A user can install almost any plug-in in his terminal, notably for decoding purposes, thereby enhancing the terminal with further decoding capabilities. Thus, it is useful to inform the content server about the available decoding capabilities, thereby enabling the content server or any intermediate server to improve the content delivery and the final user experience. The solution here proposed is analogous to the MPEG-7 MediaFormatD descriptor (for compatibility reasons) but now applied to the decoding plug-ins and not to the content. The MediaDecodingD is a descriptor to indicate which decoding plug-ins (typically more than one, e.g. audio and video) are available at the terminal. The ControlledTermUseType is an MPEG-7 tool (see Section 3.3.1), which is used to identify a controlled term from a Classification Scheme.

The MediaDecodingType descriptor fields are listed and its semantics presented in Table 4.10; the MediaDecodingType descriptor XML syntax is as follows:

```
<!--=====-->
<!--Definition of MediaDecoding -->
<xsd:complexType name="MediaDecodingType">
  <xsd:sequence>
    <xsd:element name="Identifier" type="string"/>
    <xsd:element name="Type">
      <xsd:complexType>
        <xsd:complexContent>
          <xsd:extension base="mpeg7:ControlledTermUseType">
            <xsd:attribute name="Type" use="default" value="other">
              <xsd:simpleType>
                <xsd:restriction base="string">
                  <xsd:enumeration value="image"/>
                  <xsd:enumeration value="audio"/>
                  <xsd:enumeration value="video"/>
                  <xsd:enumeration value="audiovideo"/>
                  <xsd:enumeration value="graghic"/>
                  <xsd:enumeration value="other"/>
                </xsd:restriction>
              </xsd:simpleType>
            </xsd:attribute>
          </xsd:extension>
        </xsd:complexContent>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="FileFormat" type="mpeg7:ControlledTermUseType"/>
    <xsd:element name="VisualDecodingFormat" type="mpeg7:ControlledTermUseType" minOccurs="0"/>
    <xsd:element name="AudioDecodingFormat" type="mpeg7:ControlledTermUseType" minOccurs="0"/>
    <xsd:element name="GraphicDecodingFormat" type="mpeg7:ControlledTermUseType"
      minOccurs="0"/>
    <xsd:element name="SceneDecodingFormat" type="mpeg7:ControlledTermUseType" minOccurs="0"/>
    <xsd:element name="OtherDecodingFormat" type="mpeg7:ControlledTermUseType" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
```

Table 4.10 – MediaDecoding descriptor semantics.

Name	Definition
MediaDecodingType	Describes the decoding capabilities of a certain decoder.
Identifier	An identifier of the media decoder at the terminal.
Type	Indicates the content type supported by the media decoder: <ul style="list-style-type: none"> ▪ <i>audio</i> – the content type accepted by the described media decoder is audio. ▪ <i>image</i> – the content type accepted by the described media decoder is image. ▪ <i>visual</i> – the content type accepted by the described media decoder is

	<p>visual.</p> <ul style="list-style-type: none"> ▪ <i>audiovisual</i> – the content type accepted by the described media decoder is audiovisual. ▪ <i>graphics</i> – the content type accepted by the described media decoder is graphics. ▪ <i>other</i> – the content type accepted by the described media decoder is neither of the previous content types (OtherDecodingFormat is used).
FileFormat	Describes the supported file format, e.g. MP4, AVI, QuickTime; an example of a Classification Scheme is MPEG7FileFormatCS presented in Chapter 3.
VisualDecodingFormat	Describes the visual coding format supported by the media decoder, e.g. H.261, MPEG-4 Main Level@Main Profile, H.263, JPEG; an example of CS is MPEG7VisualCodingFormatCS.
AudioDecodingFormat	Describes the audio/speech coding format supported by the media decoder, e.g. MPEG-1 Layer III, MPEG-4 Audio AAC@Level1, G.728, an example of CS is MPEG7AudioCodingFormatCS.
GraphicDecodingFormat	Describes the graphics coding format supported by the media decoder, e.g. VRML, Flash; an example of CS is MPEG7GraphicCodingFormatCS.
SceneDecodingFormat	Describes the scene coding format supported by the media decoder, e.g. MPEG-4 BIFS, SMIL; an example of CS is MPEG7GraphicCodingFormatCS.
OtherDecodingFormat	Describes the coding format for a content type not considered above; for example, the supported HTML version.

4.5.3.1.2 TerminalHardwareType

The TerminalHardwareType descriptor allows to describe the hardware that is available at the user terminal. For UMA applications, only the display and audio hardware capabilities are of interest. The HardwarePlatform field has been included only for the purpose of compatibility with the WAP specifications.

The TerminalHardwareType descriptor fields are listed and its semantics presented in Table 4.11; the TerminalHardwareType descriptor XML syntax is as follows:

```
<!--=====-->
<!--Definition of Terminal Hardware -->
<xsd:complexType name="TerminalHardwareType">
  <xsd:all>
    <xsd:element name="HardwarePlatform" type="DIUED:HardwarePlatformType" minOccurs="0"
maxOccurs="unbounded"/>
    <xsd:element name="DisplayDevice" type="DIUED:DisplayDeviceType" maxOccurs="unbounded"/>
    <xsd:element name="AudioDevice" type="DIUED:AudioDeviceType" maxOccurs="unbounded"/>
  </xsd:all>
</xsd:complexType>
```

Table 4.11 – TerminalHardware descriptor semantics.

Name	Definition
TerminalHardwareType	Describes the terminal hardware related characteristics.
HardwarePlatform	Describes the hardware characteristics not related to display and audio.
DisplayDevice	Describes the display capabilities associated with a certain display device of the user terminal.
AudioDevice	Describes the audio capabilities associated with a certain audio output device of the user terminal.

4.5.3.1.2.1 DisplayDeviceType

The features covered by this descriptor are all related to the display capabilities of the device, e.g. screen pixel resolution, bits per pixel, color capabilities. Notice that a terminal can have different display devices, each one with different characteristics. Therefore, the DisplayDeviceCapabilities descriptor

targets the description of the capabilities of all existing display devices at the terminal. If multiple display devices are available and described, the UMA adaptation may select the best one or even different views for each display device (e.g. monitor and projector).

The DisplayDeviceType descriptor fields are listed and its semantics presented in Table 4.10; the DisplayDeviceType descriptor XML syntax is as follows:

```
<!--=====-->
<!--Definition of Display Device -->
<xsd:complexType name="DisplayDeviceType">
  <xsd:sequence>
    <xsd:element name="DeviceIdentifier" type="string"/>
    <xsd:element name="Height" type="nonNegativeInteger"/>
    <xsd:element name="Width" type="nonNegativeInteger"/>
    <xsd:element name="RefreshRate" type="nonNegativeInteger"/>
    <xsd:element name="Resolution" type="decimal"/>
    <xsd:element name="AspectRatio" type="decimal"/>
    <xsd:element name="BitsPerPixel" type="nonNegativeInteger"/>
    <xsd:element name="ColorDomain" type="mpeg7:ControlledTermUseType"/>
  </xsd:sequence>
</xsd:complexType>
```

Table 4.12 – DisplayDevice descriptor semantics.

Name	Definition
DisplayDeviceType	Describes the display device characteristics.
DisplayDeviceIdentifier	Identifier of the device in the terminal.
DisplayHeight	Indicates the height of the display in pixels.
DisplayWidth	Indicates the width of the display in pixels.
DisplayRefreshRate	Indicates the refresh rate of the display device.
DisplayResolution	Indicates the resolution of the display device in dpi (dots per inch).
DisplayAspectRatio	Indicates the optical display aspect ratio.
DisplayBitsPerPixel	Indicates the number of bits per the display device pixel.
DisplayColorDomain	Indicates the color domain capabilities of the display device. The values of the color domain are defined as follows: <ul style="list-style-type: none"> ▪ <i>Binary</i> – The color capability is binary (black and white). ▪ <i>Color</i> – The display device is color capable. ▪ <i>Graylevel</i> – The color capability is gray level.

4.5.3.1.2.2 AudioDeviceType

This descriptor is very similar to the previous descriptor. A terminal can have several audio devices, each of them with different characteristics. Therefore, the AudioDeviceCapabilitiesD targets the description of the capabilities of all the audio devices available at the terminal.

The AudioDeviceType descriptor fields are listed and its semantics presented in Table 4.13; the AudioDeviceType descriptor XML syntax is as follows:

```

<!--Definition of Audio Device -->
<xsd:complexType name="AudioDeviceType">
  <xsd:sequence>
    <xsd:element name="DeviceIdentifier" type="string"/>
    <xsd:element name="Channels" type="nonNegativeInteger"/>
    <xsd:element name="front" type="nonNegativeInteger"/>
    <xsd:element name="rear" type="nonNegativeInteger"/>
    <xsd:element name="lfe" type="nonNegativeInteger"/>
    <xsd:element name="RenderRate" type="nonNegativeInteger"/>
    <xsd:element name="BitsPerSample" type="nonNegativeInteger"/>
  </xsd:sequence>
</xsd:complexType>

```

Table 4.13 – AudioDevice descriptor semantics.

Name	Definition
AudioDeviceType	Describes the audio device characteristics.
AudioDeviceIdentifier	Identifier of the device in the terminal.
AudioChannels	Indicates the number of audio channels in the device.
Front	Indicates the number of front channels.
Side	Indicates the number of side channels.
Rear	Indicates the number of rear channels.
Lfe	Indicates the number of LFE (Low Frequency Enhancement) channels.
AudioRenderRate	Indicates the sample rendering rate capability in Hz.
AudioBitsPerSample	Indicates the audio sample accuracy in bits per audio sample.

4.5.3.2 AccessNetworkType

Network characteristics are very important for applications that stream content in real-time, e.g. using MPEG-4 audio and video, as well as for UMA applications customizing the content according to the network conditions.

Because describing the end-to-end network connection is very difficult, the proposed descriptors only describe the access network, even though if required they can easily be applied to the end-to-end path. Network characteristics have been divided in two classes: a traffic descriptor related to the amount and type of resources, and a QoS descriptor expressing the quality of the connection. The AccessNetworkType descriptor gathers all the information about the network characteristics.

The AccessNetworkType descriptor fields are listed and its semantics presented in Table 4.14; the AccessNetworkType descriptor XML syntax is as follows:

```

<!--Definition of Access Network-->
<xsd:complexType name="AccessNetworkType">
  <xsd:sequence>
    <xsd:element name="Traffic" type="DIUED:NetTrafficType" minOccurs="0"/>
    <xsd:element name="QoS" type="DIUED:NetQoSType" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>

```

Table 4.14 – AccessNetwork descriptor semantics.

Name	Definition
AccessNetworkType	Describes the access network characteristics.
Traffic	Describes the network traffic capabilities associated with the available access network.
QoS	Describes the Quality of Service capabilities associated with the available access network.

4.5.3.2.1 NetworkTrafficType

This descriptor is intended to describe the traffic characteristics of the access network as mentioned in Section 4.5.3.2.

The NetworkTrafficType descriptor fields are listed and its semantics presented in Table 4.15; the NetworkTrafficType descriptor XML syntax is as follows:

```
<!--=====-->
<!--Definition of Net Traffic -->
<xsd:complexType name="NetTrafficType">
  <xsd:sequence>
    <xsd:element name="PeakBitRate" type="nonNegativeInteger"/>
    <xsd:element name="MaximumBitRate" type="nonNegativeInteger"/>
    <xsd:element name="MinimumBitRate" type="nonNegativeInteger"/>
    <xsd:element name="SustainableBitRate" type="nonNegativeInteger"/>
    <xsd:element name="BurstSize" type="nonNegativeInteger"/>
  </xsd:sequence>
</xsd:complexType>
```

Table 4.15 – NetTraffic descriptor semantics.

Name	Definition
NetworkTraffic	Describes the access network traffic characteristics.
PeakBitRate	Indicates the peak bit rate available.
MaximumBitRate	Indicates the maximum bit rate available.
MinimumBitRate	Indicates the minimum bit rate available.
SustainableBitRate	Indicates the average bit rate available.
BurstSize	Indicates the total amount of successive data packets that can be sent at PeakBitRate.

4.5.3.2.2 NetworkQoSType

This descriptor is intended to describe the QoS of the access network as mentioned in Section 4.5.3.2.

The NetworkQoSType descriptor fields are listed and its semantics presented in Table 4.16; the NetworkQoSType descriptor XML syntax is as follows:

```
<!--=====-->
<!--Definition of Net QoS -->
<xsd:complexType name="NetQoSType">
  <xsd:sequence>
    <xsd:element name="TransmissionDelay" type="decimal"/>
    <xsd:element name="TransDelayVariation" type="decimal"/>
    <xsd:element name="ErrorBitRate" type="decimal"/>
  </xsd:sequence>
</xsd:complexType>
```


Table 4.16 – NetQoS descriptor semantics.

Name	Definition
NetQoSType	Describes the access network traffic characteristics.
TransmissionDelay	Indicates the delay in the transmission of one bit from one side of the access network (user terminal) to the other.
TransmissionDelayVariation	Indicates the maximum variation of the transmission delay in ms.
ErrorBitRate	Indicates the bit error rate in the connection.

4.5.3.3 NaturalEnvironmentType

The NaturalEnvironment descriptor is designed to include multiple measures of variables characterizing the user surrounding physical world, e.g. location, temperature, pressure. This descriptor is based on a generic Sensor descriptor which may be instantiated for different physical world features.

The NaturalEnvironmentType descriptor fields are listed and its semantics presented in Table 4.17; the NaturalEnvironmentType descriptor XML syntax is as follows:

```
<!--Definition of Natural Environment-->
<xsd:complexType name="NaturalEnvironmentType">
  <xsd:sequence>
    <xsd:element name="Sensor" type="DIUED:SensorType" minOccurs="0" maxOccurs="unbounded" />
  </xsd:sequence>
</xsd:complexType>
```

Table 4.17 – NaturalEnvironment descriptor semantics.

Name	Definition
NaturalEnvironmentType	Describes the user natural environment.
Sensor	Describes one or more measures of the physical world.

4.5.3.3.1 SensorType

In the natural environment there are many variables that can be measured. Each measure can be taken in certain units and some of the measured variables can use the same units. Instead of defining one descriptor for each type of variable, a generic descriptor has been defined that can be used to measure any variable. For this, two Classification Schemes have been defined as indicated in Chapter 3: one for the type of variable being measured (e.g. “location”, “velocity”) and another for the units (e.g. “m”, “°C”). Each natural environment feature should of course have its value expressed in the correct units. A combination of several of these descriptors can be arranged to express conditions such as “at X place, with Y temperature, in day Z”.

The SensorType descriptor fields are listed and its semantics presented in Table 4.18; the SensorType descriptor XML syntax is as follows:

```
<!--Definition of Sensor -->
<xsd:complexType name="SensorType">
  <xsd:sequence>
    <xsd:element name="SensorIdentifier" type="mpeg7:ControlledTermUseType" />
    <xsd:element name="Units" type="mpeg7:ControlledTermUseType" />
    <xsd:element name="Value" type="Real" />
  </xsd:sequence>
</xsd:complexType>
```

```

</xsd:sequence>
</xsd:complexType>

```

Table 4.18 – Sensor descriptor semantics.

Name	Definition
SensorIdentifier	Indicates the type of the sensor, e.g. speed (a first proposal for SensorTypeCS is made).
Units	Indicates the units in which the measure is given, e.g. km/h (a first proposal for SensorUnitsCS is made).
Value	Indicates the value of the natural environment feature (sensor) expressed in the indicated units.

The type of measure that the sensor provides is indicated by the SensorIdentifier field, which references an ID in the SensorTypeCS. In the same way, the units for the sensor are indicated in the Units field, which references an ID in the SensorUnitsCS.

First proposals for the SensorTypeCS and SensorUnitsCS are made in the following tables:

Table 4.19 – SensorTypeCS classification scheme.

```

<ClassificationScheme uri="urn:mpeg:MPEG7SensorTypeCS"
  domain="//UserEnvironment/Sensor/SensorIdentifier">
  <Term termId="1">
    <Name xml:lang="en">Location</Name>
    <Definition xml:lang="en">Location</Definition>
  </Term>
  <Term termId="2">
    <Name xml:lang="en">Velocity</Name>
    <Definition xml:lang="en">Velocity</Definition>
  </Term>
  <Term termId="3">
    <Name xml:lang="en">Temperature</Name>
    <Definition xml:lang="en">Temperature</Definition>
  </Term>
  <Term termId="4">
    <Name xml:lang="en">Altitude</Name>
    <Definition xml:lang="en">Altitude</Definition>
  </Term>
  <Term termId="5">
    <Name xml:lang="en">Pressure</Name>
    <Definition xml:lang="en">Pressure</Definition>
  </Term>
  <Term termId="6">
    <Name xml:lang="en">Humidity</Name>
    <Definition xml:lang="en">Humidity</Definition>
  </Term>
  <Term termId="7">
    <Name xml:lang="en">Time</Name>
    <Definition xml:lang="en">Time</Definition>
  </Term>
  <Term termId="8">
    <Name xml:lang="en">Date</Name>
    <Definition xml:lang="en">Date</Definition>
  </Term>
</ClassificationScheme>

```

Table 4.20 – SensorUnitsCS classification scheme.

```

<ClassificationScheme uri="urn:mpeg:MPEG7SensorTypeCS"
  domain="//UserEnvironment/Sensor/SensorUnits">
  <Term termId="1.1">

```

```

    <Name xml:lang="en">m</Name>
    <Definition xml:lang="en">meters</Definition>
  </Term>
  <Term termId="1.2">
    <Name xml:lang="en">s</Name>
    <Definition xml:lang="en">seconds</Definition>
  </Term>
  <Term termId="1.3">
    <Name xml:lang="en">m/s</Name>
    <Definition xml:lang="en">meters per second</Definition>
  </Term>
  <Term termId="1.4">
    <Name xml:lang="en">°C</Name>
    <Definition xml:lang="en">centigrade degrees</Definition>
  </Term>
  <Term termId="1.5">
    <Name xml:lang="en">°K</Name>
    <Definition xml:lang="en">kelvin degrees</Definition>
  </Term>
  <Term termId="1.6">
    <Name xml:lang="en">Pa</Name>
    <Definition xml:lang="en">pascal</Definition>
  </Term>
</ClassificationScheme>

```

4.5.3.4 UserDescriptionType

The UserDescriptionType descriptor allows collecting information about the user that is receiving the content or is using some service in an MPEG-21 framework. The user while a person can be described by several characteristics: its demographics; its preferences; and by its usage history. These four items are the relevant information described by the UserDescriptionType.

The UserDescriptionType descriptor fields are listed and its semantics presented in Table 4.18; the UserDescriptionType descriptor XML syntax is as follows:

```

<!--=====-->
<!--Definition of User Description -->
<xsd:complexType name="UserDescriptionType">
  <xsd:sequence>
    <xsd:element name="User" type="mpeg7:PersonType" minOccurs="0"/>
    <xsd:element name="UserPreferences" type="mpeg7:UserPreferencesType" minOccurs="0"/>
    <xsd:element name="UsageHistory" type="mpeg7:UsageHistoryType" minOccurs="0"/>
    <xsd:element name="OtherPreferences"/>
  </xsd:sequence>
</xsd:complexType>

```

Table 4.21 – UserDescription descriptor semantics.

Name	Definition
UserDescriptionType	Describes the user characteristics.
User	Describes the user's demographic information (if not against privacy rules).
UserPreferences	Specifies preferences pertaining to the consumption of multimedia content by a particular user (already defined in MPEG-7).
UsageHistory	Specifies user's multimedia content consumption history (already defined in MPEG-7).
OtherPreferences	Describes the user preferences in terms of the communities to which he belongs.

Concerning the fields proposed for the UserDescription descriptor, the MPEG-7 standard already includes description tools to address the description of the user preferences and usage history. Following the MPEG-21 policy of using as much as possible the available MPEG-7 tools, it is also proposed here

to use the MPEG-7 user preference tools for interoperability reasons. However, if new types of preferences are needed, such as food and service preferences, new descriptors (not related to content) must be defined.

4.6 Summary

The user environment description problem was addressed in this Chapter, notably by presenting the technology which seems to be today's best existing solution for the problem: CC/PP and WAP UAProf. However, this solution has some drawbacks and limitations, as was discussed, and it is very domain specific. Due to its importance, the user environment description problem was presented to MPEG which acknowledged its relevance and decided to issue a Call for Proposals in the context of its most recent standard: the MPEG-21 Standard. Since MPEG-21 is divided into several rather independent parts, and the user environment description tools will build Part 7, other standardization organizations may easily reference the MPEG-21 user environment description tools and companies may adopt these tools even without using the rest of the MPEG-21 technologies.

The most valuable result from this Chapter is the definition of a user environment description solution that will very likely be sent to the MPEG Committee as a response to the MPEG-21 Call for Proposals on Digital Item Adaptation.

Until today WAP UAProf is the only standard providing tools to describe user environments. In this chapter the need for a standard providing the tools to describe the user environment was discussed and a proposal has been made. Recently, W3C has recognized the same need as presented by this thesis and has merged the Mobile Access Activity [38] with the Television and the Web Activity [41] into a single group: the Device Independent Activity [42]. The goals of the W3C Device Independence are similar to the objectives of this chapter since the same problem was identified: the lack of a generic framework to describe user environments.

Chapter 5

UMA System Design

The high-level block diagram presented in Figure 5.1 shows the basic architecture of a UMA Engine, highlighting the main modules and the key technologies. The applications' analysis performed in Chapter 2 and the technologies presented in Chapters 3 and 4 were used to design the UMA Engine and the overall UMA System that were implemented in the context of this thesis. The complete UMA System is composed by several elements (see Figure 5.2), the most important of which is the element where the UMA Engine is included, the so-called UMA Platform.

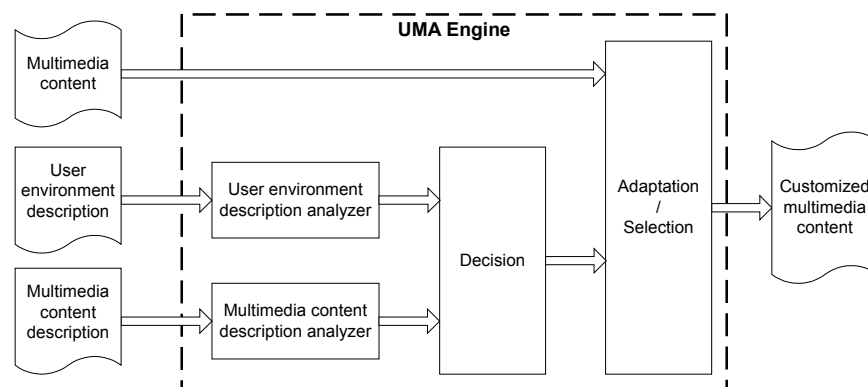


Figure 5.1 – High-level block diagram of the UMA Engine.

This chapter will cover the UMA System and the UMA Platform architecture design and implementation for this thesis. Content processing inside the UMA Platform will be addressed in Chapter 6. This chapter is organized in the following way: in a first section the UMA System architecture is presented; then the next section covers the UMA Platform design; and finally the UMA Platform modules are described.

The system architecture designed in this chapter was implemented using Visual C++ 6.0 under Windows 2000; whenever a Web server was needed, an Apache Web server was used.

5.1 UMA System Architecture

Since the UMA Platform itself is only intended to perform content customization, other elements must be added in the overall system so that a UMA-based application may be built. The purpose of this section is to define which network elements compose the UMA System and their functions. Figure 5.2 illustrates the UMA System and its major elements; those in bold have been implemented in the context of this thesis.

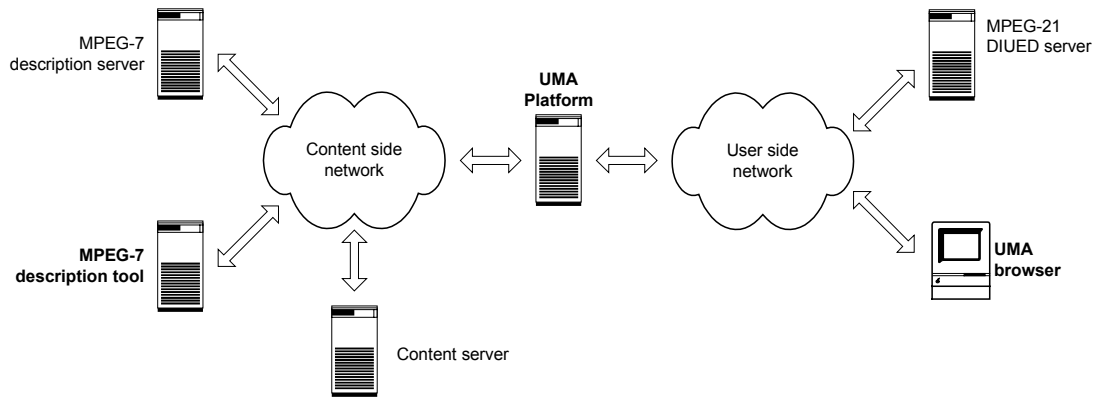


Figure 5.2 – Main elements of a UMA System.

The function of each element in the UMA System is:

- **Content server:** this element will act as the content source.
- **MPEG-7 description tool:** this element is used to analyze the multimedia content available at the content server and generate an MPEG-7 description that will be saved locally or posted into an MPEG-7 description server.
- **MPEG-7 description server:** this element stores the MPEG-7 descriptions received from the MPEG-7 description tool. In this database, one MPEG-7 description exists per each URL (content location). This element provides the UMA Platform with the MPEG-7 description for the pretended piece of content.
- **MPEG-21 DIUED server:** this element stores the user environment description (DIUED) received from the UMA browser. In this database, one DIUED per IP address exists. This element provides the UMA Platform with the DIUED for the pretended user⁵.
- **UMA Platform:** this element corresponds to the application implementing the content customization required to provide the best experience to the user for the content he/she asked. It will act as a content customization server.
- **UMA browser:** this element includes a Web browser used to access content and allows the user to manage his/her usage environment description (DIUED in MPEG-21) through appropriate menus and send it to an MPEG-21 DIUED server or directly to the UMA Platform.

The content server, the MPEG-7 description server and the MPEG-21 DIUED server are mere Web servers (Apache Web server [10]), which implement the HTTP protocol allowing other applications to store content in the server (through the POST command). The UMA browser, the UMA Platform and the MPEG-7 description tool are applications that were implemented in the context of this thesis. The UMA Platform implements the customization engine, while the other two tools allow the creation of descriptions to be consumed by the UMA Platform. The UMA browser and the MPEG-7 description

⁵ The user is here understood as the combination of the terminal, network and natural environment characteristics with the user preferences.

tool are presented and explained in Chapter 7 while the UMA Platform is presented and explained in Chapter 5 (the internal organization) and in Chapter 6 (the content processing).

The MPEG-21 DIUED server works as a WAP UAProf server [50]: this server contains the base user environment description; for every content request, the terminal sends only the differences relatively to the base DIUED and a reference to his base DIUED. The base DIUED describes the static user environment characteristics that never change; for example, contains the description of a certain terminal model published by the terminal manufacturer. The user can then send multiple requests for content without the UMA Platform having to retrieve the base DIUED for each request (only non-static characteristics of the user environment, e.g. associated to the natural environment features, are received with each request). In this case, just the necessary MPEG-7 content description will be looked for. Therefore, comparing the average time consumed for retrieving the required DIUEDs with the average time consumed for retrieving the required MPEG-7 description, it can be concluded that the average time consumed by DIUED retrieving is typically much lower.

In principle, content customization can be performed in three different network locations: at the client, at the content server or somewhere in the middle (at an intermediary server). This means the UMA Engine would be located in one of these three locations. In the following sections, the advantages and disadvantages of the three possible configurations will be discussed:

- UMA Platform at the content server
- UMA Platform at an intermediary (proxy) server
- UMA Engine at the client

The main issues to consider include how deployable is the system (in terms of integration in an existing network), how efficient is the usage of bandwidth, CPU, and memory, how the configuration may influence the content adaptation performance, and how copyright protection and privacy requirements are dealt with. Both the content server and proxy server configuration were implemented in the context of this thesis.

5.1.1 UMA Platform at the Content Server

When an author publishes content on a content server that has certain functionalities available, he/she will use them to improve its content presentation (e.g. Active Server Pages [14] from Microsoft and Java Server Pages [15] from Sun Microsystems). If that content server has a UMA Platform, the author knows that certain types of content customization mechanisms will be available. Thus, this configuration provides more author control by tying the content authoring process to the content customization, allowing the author to provide hints (descriptions) on the adaptations for different circumstances (for example, if there is a region of interest or if the image should be replaced by text and never adapted in terms of spatial resolution). An author can also preview the adapted outcome under different viewer preferences and conditions and thus instruct which adaptations can be performed or not. This greatly eases the authoring process for heterogeneous environments, as the author needs only to create the content once, and the UMA Platform automatically generates the appropriate content variation to be delivered.

Figure 5.3 illustrates the content server configuration. Since the UMA Platform hosts the content locally, most probably the content descriptions will also be available locally as it may have been created by the same entity: the author with a description tool.

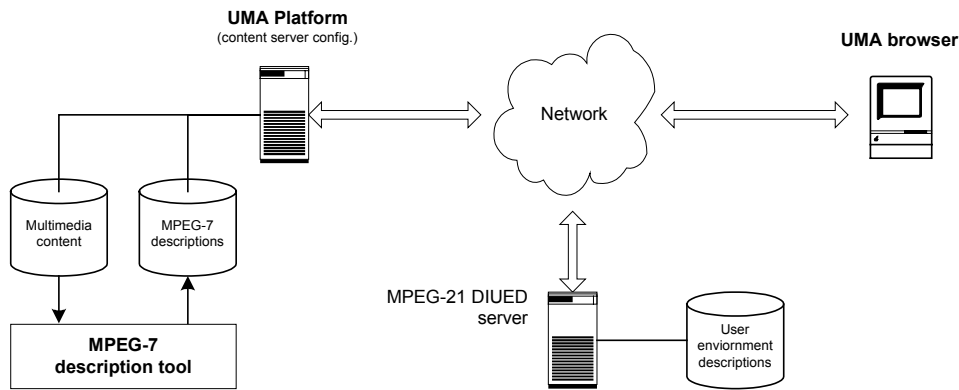


Figure 5.3 – UMA Platform at the content server.

The content server-based configuration has the advantage to ease the dynamic content adaptation (on-the-fly) because the content is locally available. The content server configuration provides the best solution for a secure environment, such as in e-commerce applications, where content is usually encrypted. Because the content travels encrypted from the content server to the end-user, only the content server may perform the content adaptation and afterwards the content encryption. This architecture also benefits real-time streaming environments where the adaptation is done while the converted content is being delivered.

In addition to the technical issues, it is necessary to consider the rights management and business implications of adaptive content delivery. Adapting content authored by someone else may incur in rights infringement. With the content server configuration, this liability can be avoided in some circumstances, e.g. if the content provider owns the content and the UMA Platform what is in this case more likely.

5.1.2 UMA Platform at a Proxy Server

In a proxy-based adaptation, the user takes the UMA Platform as its proxy, which will then make the request to the content server on behalf of the user. The content server response (in this case unable in principle to perform content customization) is then received by the UMA Platform, which afterwards decides and performs the adaptation (if needed), and then sends the transformed content back to the client. It is usually assumed that the bandwidth between the UMA Platform and the server is much higher than between the client and the UMA Platform, so the time to download the original content from the server to the UMA Platform is much lower. This is true, for example, when a proxy resides at the point-of-presence (POP) of an Internet Service Provider, as well as when the user is connecting through a slow modem or a wireless link.

For example, the proxy may transform existing Web content based on its HTML structure and MPEG-7 description. Figure 5.4 illustrates this network configuration: after receiving the content, the UMA Platform fetches the MPEG-7 description and the MPEG-21 DIUED, and based on them decides which content customization must be performed.

A proxy-based architecture makes it easy to place the UMA Platform geographically close to the clients; for example, at an ISP (Internet Service Provider) it may be used to serve multiple users. Adapting on a proxy means that existing content servers are kept unchanged because each UMA Platform can transform content for many servers, thus achieving more economy of scale than with the previous architecture.

Because a proxy takes the content from many servers, there is content with widely varying characteristics, created with many different authoring tools (e.g. Web pages). Thanks to content descriptions including adaptation hints provided by the author, it is again possible to have a good author control on the outcome

of the adaptation with the proxy server based configuration, since he/she can characterize the content and determine which alterations "look good" for that content. However, because the author and the UMA Platform (proxy) can be from different entities, he/she cannot predict if a UMA Platform is in the way between the content and the end user. Even if a UMA Platform exists in the content path, the author does not know which types of content customization are available at the UMA Platform.

When needed, the UMA Platform analyses the content and generates an MPEG-7 description that can be used in future adaptations of the same content. This process is depicted in Figure 5.4 through the inclusion of the MPEG-7 description tool module. Content providers that wish to give the task of content customization to a third party can supply an MPEG-7 description database that will improve the content customization quality of their Web sites, without having to design new content for a new market segment (e.g. PDAs, Smart Phones).

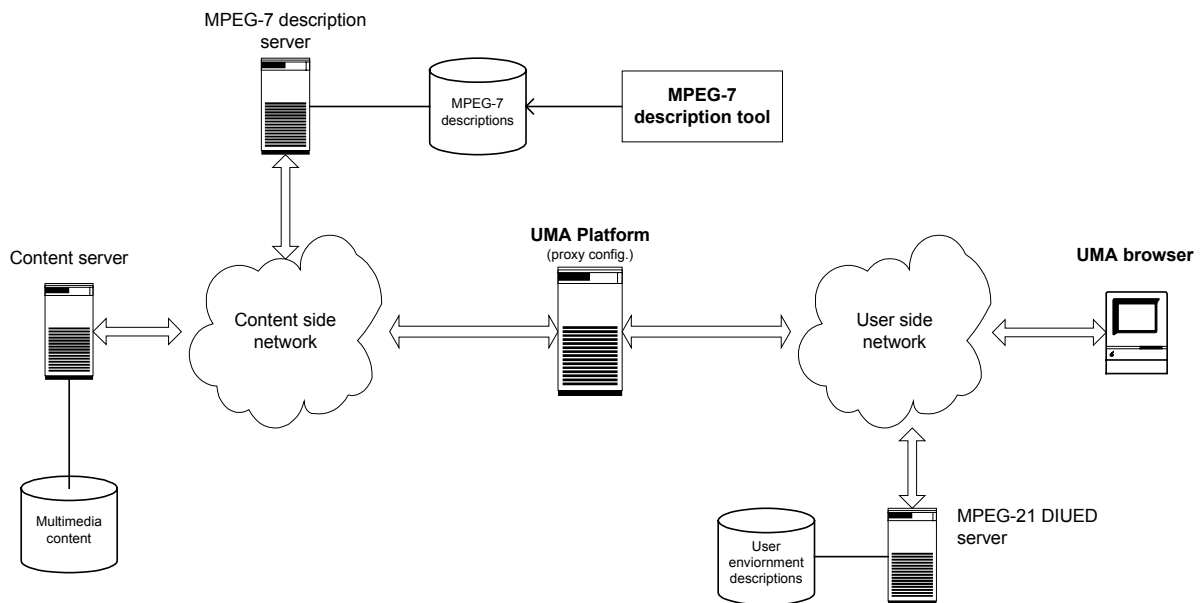


Figure 5.4 – UMA Platform at the proxy server.

Even when the content descriptions do not exist, the content customization may be performed although with less quality, since the descriptions must be extracted in real-time (and thus the customization decisions are not previously tested). So, the author of the multimedia content may find the resulting adaptation unpleasant, as it may be difficult to reliably ensure that an automatically adapted piece of content looks esthetically pleasant and efficient in terms of conveying the original information. For secure or proprietarily encoded content (e.g. QuickTime), the organization deploying the proxy server will need to coordinate with the content provider in order to access the content (in an open format) and its description for performing the adaptation. For real-time streaming services, the adaptation is a rather complex problem, because the UMA Platform is retrieving the content, adapting the content and streaming the content simultaneously.

The issue of copyright infringement becomes critical in a proxy-based system, since an author may have little control on the content adaptation to be performed (in the case where there is no knowledge by the content owner of the adaptation capabilities of the UMA Platform in proxy configuration). For example, a common filter is that of blocking advertisement logos. Considering that many free Web services rely on advertising revenues, these content providers and their advertising partners will likely be displeased by the decreased hit rates of the ads due to the ad-blocking filter.

In the content server configuration, the multimedia content database and the MPEG-7 description database typically share the same machine. This approach is the most common, as very likely the content provider will also deliver the content descriptions. In both configurations, the content provider may also not be able to deliver the content descriptions, and thus, a separation of the MPEG-7 description server from the content server may happen. One benefit of running the MPEG-7 description tool in a dedicated system is the extra resources available to work continuously in the analysis, generation and improvement of the MPEG-7 descriptions concerning the most accessed content. Descriptions' improvement can be achieved by more complex analysis and by human intervention on the extracted features. This process will increase the customization performance and the quality of the resulting content.

5.1.3 UMA Platform at the client

There are situations in which the terminal may have enough capabilities to participate in the content customization process, for example selecting which content variation from those available must be received and, in some cases, even performing low-complexity content adaptations. Although these situations are not very common in mobile environments, this solution has a very important advantage over the previous ones: the user privacy questions are not relevant since the user preferences information does not leave the terminal. Figure 5.5 illustrates this configuration. The content server provides the content and the MPEG-7 description server provides the content descriptions to the terminal that is implementing a UMA Engine. Based on the content description, e.g. informing that there are a few variations of the pretended content, the terminal decides which is the most adequate to get based on the user environment conditions.

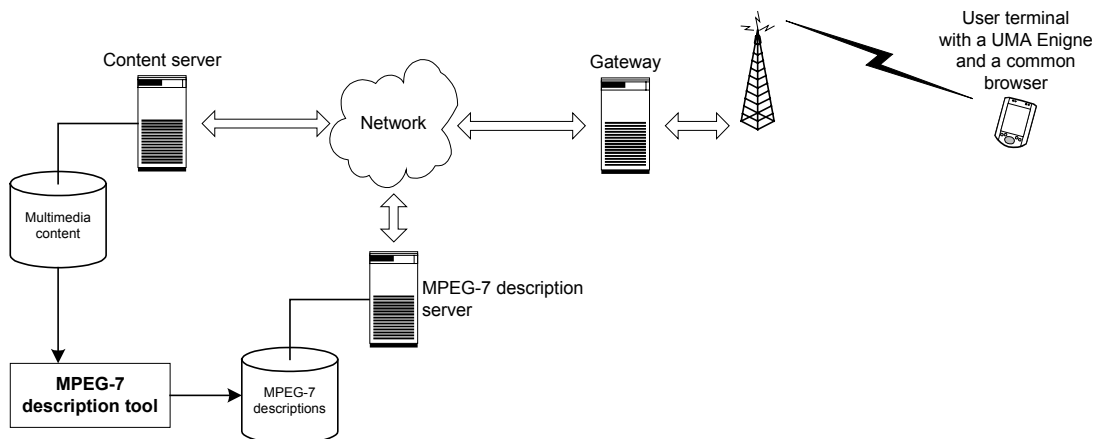


Figure 5.5 – UMA Engine at the user terminal.

In terms of system architecture, the main difference between the content server and the proxy server configuration is where the content customization is done and consequently from where the content is retrieved. In contrast with the previous configurations, this one follows a different architecture design. For example, the UMA Engine may be based on two different architectures:

- The UMA Engine is physically divided in two parts: the decision process is performed at the user terminal and the content customization (if any) is performed at a content or proxy server.
- The UMA Engine is completely implemented at the user terminal: both the decision and the content customization are performed at the terminal.

Since the decision process is always performed at the terminal, the user preferences do not have to travel in the network as for the previous configurations. This provides a completely secure way of using the user environment information. Given that the user environment description is never transmitted over the

network, there is no risk that other entities get this information. Is it true that for the previous configurations the user environment description may be sent encrypted, but it is always less secure than not sending it at all.

The author control over the adaptation process is the same as in the proxy configuration. The author may publish the content and the corresponding descriptions, but since he/she does not know the type of UMA Engine that will process its descriptions, he does not know the behavior of the UMA Engine working at the user terminal. Therefore the content customization result will most probably have a lower quality than for the content server configuration.

The content description plays a very important role in this configuration. The content description will be transmitted to the terminal, using bandwidth and memory, and the terminal will have to analyze it to decide the best content customization to perform. Because the decision is always made at the user terminal (where the content itself is not available), the content description must be powerful enough for the UMA Engine to decide the best content customization that fits his case. As can be understood from this explanation, the content description created for consumption in a terminal with limited resources or in a server is very different, which means that in this case the content descriptions to be used must take into account the environment under which they will be consumed; they must be powerful but simple which may be a difficult target to achieve.

This configuration consumes more bandwidth because it needs to send the terminal the content description.

Also this configuration poses a severe drawback in terms of terminal autonomy. Mobile terminals have limited resources (notably power resources) and, in many situations, it is preferable to ask a (fixed) server to perform all the heavy tasks for the terminal. Because with this configuration more tasks are performed at the terminal, the terminal autonomy will decrease according to the customizations. Even though, with today's advances, it is possible for a terminal to perform some content adaptations.

The allocation of content customization tasks between the server and the terminal involves several factors that must be weighted:

- The server transfers to the terminal the task of deciding the best selection or customization. By doing so, the server gets the task of transferring the content description to the terminal and receiving its decision response, which reduces the advantages of giving the adaptation/selection task to the terminal.
- When the UMA Engine is at the server, the resources available for content customization are known; however, if the UMA Engine is at the user terminal, the decision algorithm does not know if the server has available resources to perform the decided customization. This may lead to a server overload.

All these factors must be evaluated in order to know if the distribution of tasks is a reasonable solution or not and if it is the most adequate solution for every situation.

In scenarios where the terminals have large capabilities, this approach is in fact used such as for Web TV terminals where the terminal is a common PC with DVB [7] software incorporated. Unless mobile terminals start to have more capabilities, enabling them to process multimedia content, the heaviest tasks of content customization will have to keep being performed at (fixed) servers.

5.2 UMA Platform Design

This section presents the design of the UMA Platform based on the previous analysis. The implemented UMA Platform can be configured as a content server or as a proxy server based. Several modules compose the UMA Platform (see Figure 5.6): the UMA Engine, the modules required by the UMA Engine to process content and their descriptions, and the interfacing modules (with the Administrator and with the network).

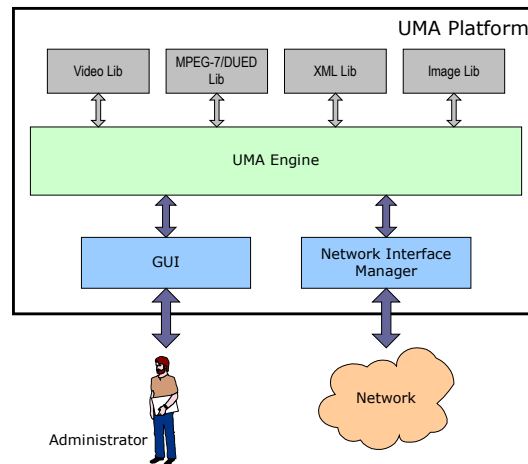


Figure 5.6 – UMA Platform building modules.

Figure 5.6 shows the main modules of the UMA Platform:

- **Graphical User Interface (GUI):** this module monitors and administrates the interface to the Platform. It is highly platform dependent and only the Platform administrator should have access to it.
- **Network Interface Manager (NIM):** the Network Interface Manager is responsible for the network communications between the UMA Platform, and the other UMA System elements. It is also responsible for retrieving the descriptions. Provides all the functions to interface with the network, e.g. an HTTP network, and enables the Platform to be configured as a content server or as a proxy server (at least for the implemented UMA Platform).
- **UMA Engine:** this is the content customization module; it includes several modules running in sequence and parallel.
- **MPEG-7 / DIUED Lib:** this module is a software library that offers an MPEG-7/DIUED API to read (parse) and write (serialize) MPEG-7/DIUED descriptions. Because the DIUED and MPEG-7 use the same base technology for description representation (XML Schema), this library is the same for both MPEG-7 and MPEG-21 DIUED APIs.
- **XML Lib:** this is the module responsible for parsing XML data (to read the MPEG-7 and DIUED descriptions). The *Xerces C++* software library [10] that provides all the XML Schema tools is used as the XML parser for the platform.

- **Image Lib:** this module provides a set of tools to process images. The *Image Magick* software library [12] together with the implemented color temperature adaptation algorithms are used by the Platform to perform the image processing functions.
- **Video Lib:** this module provides a set of tools to process video. For the UMA Platform implemented, this is an adapted version of an *ISO MPEG-2/MPEG-1* encoder and decoder [13], which enables the transcoding of MPEG-2/MPEG-1 video streams.

5.2.1 Architecture Design

Chapter 2 defined a list of requirements on the UMA Engine architecture that were taken into account for the design of the UMA Platform, notably:

1. Caching previous customizations
2. User management
3. Multimedia presentation and synchronization
4. Real-time processing
5. Scalability (number of requests)
6. Scalability (response time)

The first and second requirements regard the re-use of previous content customizations and previously retrieved descriptions; these requirements are immediately satisfied by the implementation of caching tables to store the information that a request generates: content and user environment descriptions, a reference to the source content, and a reference to the customized content. The third requirement is related to composite content: the decision module in the UMA Engine must retrieve information regarding every element that composes the content. This means that the Engine must provide a mechanism to relate several requests that concern the same composite content (requests dependencies on other requests).

The fifth requirement defines scalability in terms of number of users, that is, the system should be able to simultaneously process multiple requests; in this case the priority is the number of simultaneous requests. The sixth requirement defines scalability in terms of response time, that is, the system should be able to dispatch simultaneous requests within a minimum time; in this case the priority is the response time for each request. Because these requirements cannot be implemented simultaneously, a best-effort mechanism was implemented which corresponds to the sixth requirement.

Having these requirements in mind, a pipeline architecture was chosen for the UMA Engine so that the processing of each request can be easily split into several successive steps, thus allowing the UMA Engine to process several requests simultaneously. With this architecture it is possible to distribute the processing among several entities (threads, processes or even computers). This pipeline architecture is presented in more detail in Figure 5.7.

The main advantages of using a pipeline architecture for the UMA Engine are:

- **Optimal for multiple requests:** due to its pipeline nature, the UMA Engine can handle several user requests simultaneously.

- **Request processing time:** the tasks are independent and elementary so that some of them can be executed in parallel (for example, the multimedia content and the user environment descriptions' are partly processed in parallel);
- **Distributed computing:** each step of the pipeline can be easily modified to be executed in different machines;
- **Robustness to software errors:** if a software error occurs in a module, then that module can be removed and a new module enters in the pipeline, and only the request that was being processed in that module is lost.

As mentioned before, the GUI and the Network Interface Manager are outside the UMA Engine. Observing the block diagram in Figure 5.7, it is clear that the UMA Engine has a pipeline-based structure. It is now possible to analyze in the diagram of Figure 5.7 the modules that reside inside the UMA Engine:

- **Pipeline Manager/Monitor (PM/M):** this is the only module interfacing with the GUI, which means that all status and control messages to/from the GUI are processed by this module. When the processing of a request is complete, this module updates the caching tables with references to the customized content and descriptions. The management of the caching data tables is the responsibility of this module; all the other modules may only read from these caching tables.
- **User Request Processor (URP):** when a new request arrives at the UMA Platform, this module examines the caching tables in order to reuse information that has been previously retrieved or generated (e.g. content description or content variations). This module may also transform a non-DIUED format user environment description (for example, a WAP User Profile) into a DIUED format description so that the Customization Action Decision module may process it. To gather more information about the user, this module may also process the HTTP headers.
- **Multimedia Content Description Builder (MMCDBuilder):** this module parses the MPEG-7 descriptions into internal data memory structures. The MPEG-7/DIUED Lib is used by this module to process the description data.
- **Digital Item Content Description Builder (DIUEDBuilder):** this module parses the MPEG-21 DIUED descriptions into internal data memory structures. The MPEG-7/DIUED Lib is used by this module to process the description data.
- **Multimedia Content Buffer:** this module stores the multimedia content retrieved from the content server. This is the content source to be adapted by the customization operations.
- **Content Description Buffer:** this module stores in the internal data structures the MPEG-7 content description after being parsed.
- **User Environment Description Buffer:** this module stores in the adequate internal data structures the user environment description after being parsed.

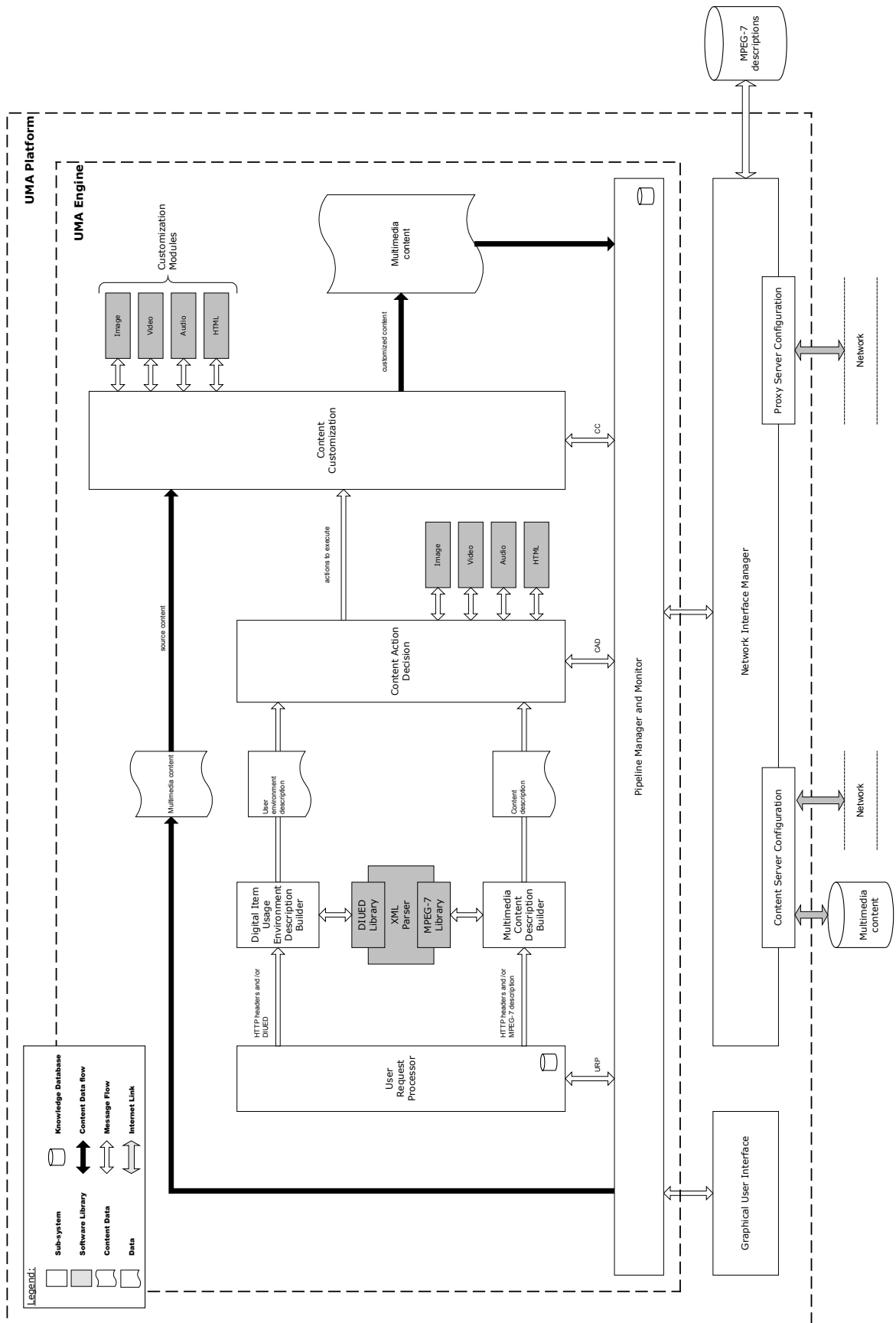


Figure 5.7 – UMA Platform architecture.

- **Customization Action Decision (CAD):** the Customization Action Decision module matches the content description with the user environment description to decide which transformations must be performed to the content, taking into account the customization processes available.
- **Content Customization (CC):** this module performs all the content processing operations decided by the CAD module; it includes several customization modules each one corresponding to the various media types that can be adapted. Depending on the decided actions, the corresponding customization module is called to perform the content adaptation.
- **Customized Multimedia Content Buffer:** this buffer stores the content variation that will be delivered to the user.

The pipeline architecture is associated to its four successive stages: 1) the URP; 2) the DIUEDBuilder and the MMCDBuilder; 3) the CAD; and 4) the CC. The descriptions' builder stage is divided in two branches: one for the parsing of the content description and another for the parsing of user environment description. The two branches work independently so that each description may be processed independently and in parallel with the other.

A simplified walkthrough of the process performed to satisfy a user request may help to understand the proposed UMA Platform architecture:

- (1) Using the **UMA browser**, the user sends its user environment description to the **MPEG-21 DIUED server** so that the UMA Platform may get it.
- (2) As soon as the **Network Interface Manager** receives a request for content from the user, it forwards the request to the **User Request Processor**.
- (3) After receiving a user request, the **User Request Processor** module asks the **Network Interface Manager** to retrieve the DIUED and the content description.
- (4) After receiving the content description, the **MMCDBuilder** module parses the description into adequate internal data structures and sends it to the **Content Action Decision** module.
- (5) After receiving the user environment description, the **DIUEDBuilder** module parses the description into adequate internal data structures and sends it to the **Content Action Decision** module.
- (6) At this stage, the **Content Action Decision** module matches the content description with the user environment description, and decides which customization actions must be performed to the content to provide the user with the best possible experience for the content requested. After deciding the content customization actions, the **CAD** informs the **Content Customization** module.
- (7) When the **Content Customization** module receives the content customization actions to perform, it simply calls the customization modules to process the content and generate the customized multimedia content. Before starting the content customization process, the **Network Interface Manager** is informed that he can start sending the content to the user while it is being adapted.

- (8) Finally, when the content has been completely adapted by the **Content Customization** module and transmitted by the **Network Interface Manager**, a reference to the adapted content is stored in the caching tables for later use.

Each content request has its own context: the request itself, internal references to the user environment and content descriptions and internal references to the content itself. Whenever a module sends a message to another, the request context accompanies the message, thus eliminating any concurrency problem between the modules regarding the access to memory, e.g. the caching tables.

Between the URP and the CAD there are two parallel paths for the two types of descriptions. When both descriptions are parsed, the CAD matches the characterization of the user side with that of the content side. Because of this parallel processing, a problem arises: the synchronization of the Content Action Decision module. Regarding the request context integrity, there is no problem in having the MMCDBuilder and the DIUEDBuilder using the same request context because they have to process distinct data within the same request: the MPEG-7 data and the DIUED data. There is also a case when two modules access a request context simultaneously: while CC is generating the customized content (writing) the NIM is streaming the generated content (reading). These are the two only situations when two modules are accessing the same request context, for the remaining cases, the rule is: a request context is only accessed by one module.

Note that in Figure 5.7 the software libraries are not outside the UMA Engine. In Figure 5.7 all UMA Platform software libraries are marked in gray. The MPEG-7/DIUED Lib uses the XML Parser library and the remaining libraries are used by the content customization modules.

5.2.2 Data Structures

The UMA Engine data objects model presented in Figure 5.8 shows the organization of all data related to a single user request. This diagram illustrates the way data objects relate to each other; all data structure is centered on a Request data object. The main data objects used in the implementation made for this thesis are:

- **Request:** used as a base data object to organize all data related to a single user request.
- **User:** gathers all information about the user who sent a request to the UMA Platform.
- **MMContent:** gathers all information about the content requested by the user.

A data object not present in the diagram of Figure 5.8 is the `infoStatus` object used to inform the GUI of each module's status. There are other data objects that are not implemented in the UMA Engine module but in the MPEG-7/DIUED Library. These data objects are used to instantiate the descriptors of the content descriptions and user environment descriptions.

5.2.2.1 Request

This data object is used as the base element to organize all data related to a single user request. lists all the Request data object attributes. Each user request regards one multimedia content object and its description. This data object is instantiated every time the Network Interface Manager receives an HTTP command asking for content, in the configured port. All other data objects are referenced by this one: when a message is passed from one module to another, the reference to Request is passed, thus assuring that only one module holds the reference to that Request object. This procedure may be understood as a token that is passed from one module to another and thus at any given moment the token is held by

only one module. Modules retain no data related to a request; every data concerning a request is held in the Request object (and by the referenced objects, see Figure 5.8).

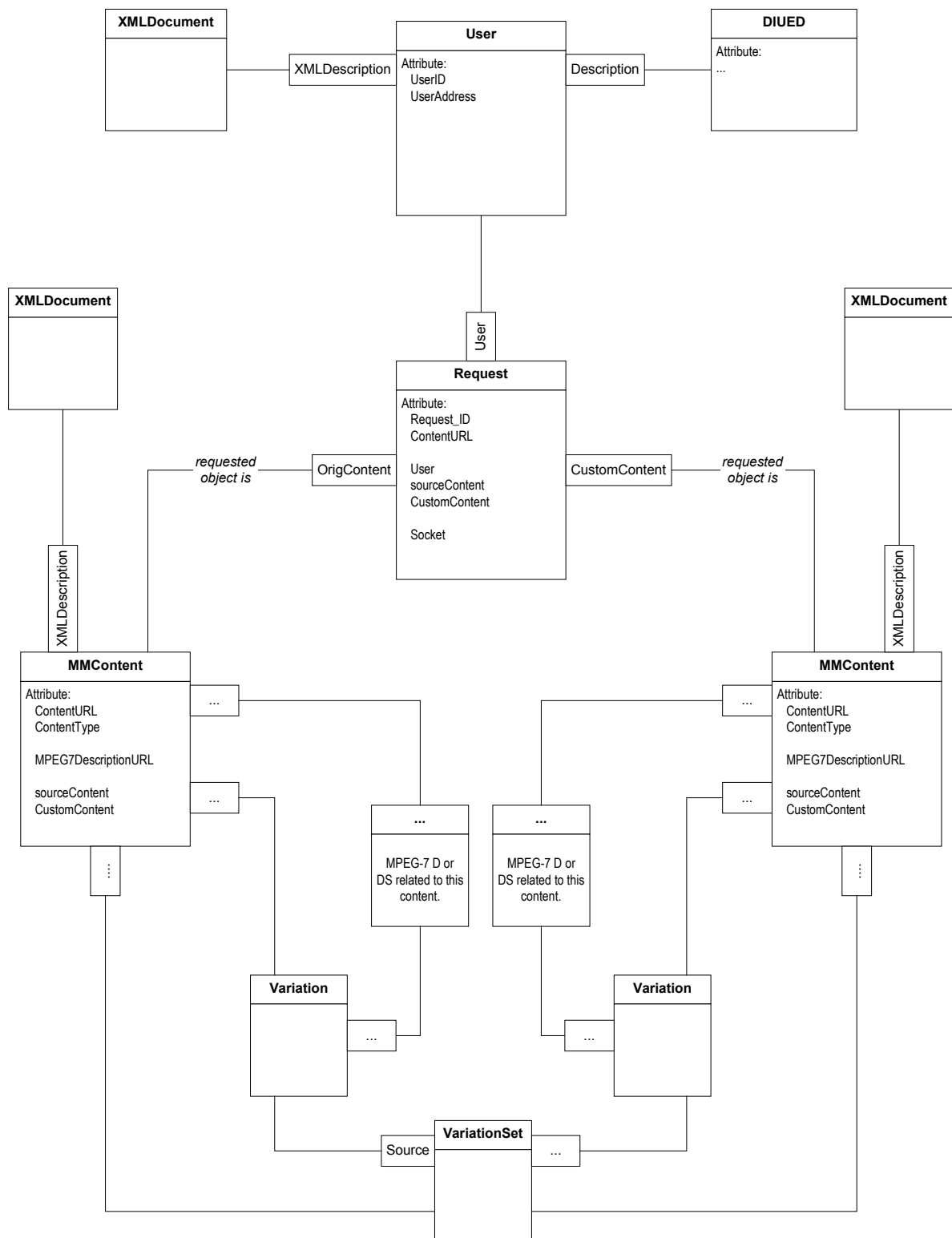


Figure 5.8 – Data objects model.

Table 5.1 – Request object attributes.

Request attributes	
ContentURL	String with the URL for the requested content.
RequestingUser	Reference to the User data object.
sourceContent	Internal reference to the MMContent object corresponding to the source multimedia content.
customContent	Internal reference to the customized multimedia content.
DIUEDready	Boolean indicating that the DIUED processing is complete.
MMCDready	Boolean indicating that the content description processing is complete.
ActionStack	Instructions from the CAD to the CC.

5.2.2.2 User

This data object aggregates all information on the user who is requesting content. Table 5.2 lists all the User data object attributes. The User data object stores the user IP address and the IP port of the socket of the user terminal. A reference to the user environment description is stored in `DIUEDurl`; when this description has been processed, a reference to the parsed description is put in `DIUED`. After the request is completed, the User data object and MMContent data object are stored in caching tables for later use in the context of other requests.

Table 5.2 – User object attributes.

User attributes	
UserIPAddress	User IP address.
UserIPPort	Associated socket port.
DIUEDurl	URL for the DIUED.
XMLDocBuffer	DIUED in XML format.
DIUED	Internal Reference to the root element of the XML DIUED.

5.2.2.3 MMContent

This data object aggregates all information on the relevant multimedia content. Table 5.3 lists all the MMContent data object attributes. Each MMContent data object concerns a certain multimedia content that is referenced by the `contentURL`. The content data is accessed through the internal reference `ContentData`. The content media type is identified by the `contentType` attribute. The content description is located at the URL designated by `descriptionURL`. When the content description has been parsed, internal references to the instantiated descriptors and description schemes are stored in their respective fields.

Table 5.3 – MMContent object attributes.

MMContent attributes	
ContentURL	URL for the requested content.
ContentType	String with the multimedia content media type.
ContentData	Internal reference to the multimedia content data.
DescriptionURL	URL for the multimedia content description.

XMLDocBuffer	MPEG-7 content description in XML format.
Source	Internal reference to the original content.
VariationSet	Internal Reference to the MPEG-7 VariationSet DS data regarding MMContent.
ThisVariation	Internal Reference to the MPEG-7 Variation DS regarding MMContent.
MediaLocator	Internal MPEG-7 Descriptor; see Chapter 3.
MediaInformation	Internal MPEG-7 Description Scheme; see Chapter 3.
MediaProfile	Internal MPEG-7 Description Scheme; see Chapter 3.
MediaFormat	Internal MPEG-7 Descriptor; see Chapter 3.
MediaInstance	Internal MPEG-7 Descriptor / Description Scheme; see Chapter 3.
SequentialSummary	Internal MPEG-7 Description Scheme; see Chapter 3.
HierarchicalSummary	Internal MPEG-7 Description Scheme; see Chapter 3.
HTTPHeaders	HTTP headers to be added in the response to the content request, including the content media type and content size.

5.2.2.4 infoStatus

Every module uses this data object to inform the PM/M that a message is being sent from one module to another. The PM/M will then forward the `infoStatus` message to the GUI. Table 5.4 lists all the `infoStatus` attributes. This data class holds a copy of the information related to a message that is sent between two modules.

Table 5.4 – infoStatus object attributes.

infoStatus attributes	
MessageTo	Sender module identification.
MessageFrom	Receiver module identification.
MessageType	Type of message (see Table 5.5).
MessageTime	Time when the message was sent.
ContentURL	URL of the content requested.
UserIP	IP address of the requesting user.

5.2.2.5 VariationSet

This data class corresponds to the MPEG-7 VariationSet description scheme presented in Chapter 3. This DS is one of the several DSs and Ds implemented in the MPEG-7/DIUED Library software.

5.2.2.6 Variation

This data class corresponds to the MPEG-7 Variation description scheme. This DS is one of the several DSs and Ds implemented in the MPEG-7/DIUED Library software.

5.2.2.7 DIUEDConsumer

This data class corresponds to the DIUED descriptor proposed in Chapter 4. This descriptor is one of the several DIUED descriptors implemented in the MPEG-7/DIUED Library software. The DIUED is a schema that uses some of the MPEG-7 tools (e.g. user preferences) and shares a common format (XML Schema); therefore, the DIUED and the MPEG-7 libraries has been implemented as in the same library.

5.2.3 Event Scenarios

Four scenarios were used to design all the messages and the dynamics of each module within the UMA Platform.

5.2.3.1 Event Scenario 1: User request

When the Network Interface Manager receives a content request via the HTTP protocol, it sends a MSG_REQUEST message to the URP module. The base DIUED and the differences received with the request are used to process the request. The URP analyzes the request to check if it has cached the base DIUED of the requesting user, the content itself or its MPEG-7 description. If it does not have one of the descriptions, the URP module sends a MSG_GET_MMCD or a MSG_GET_DIUED to the Network Interface Manager to get the missing description(s). This scenario is depicted in Figure 5.9.

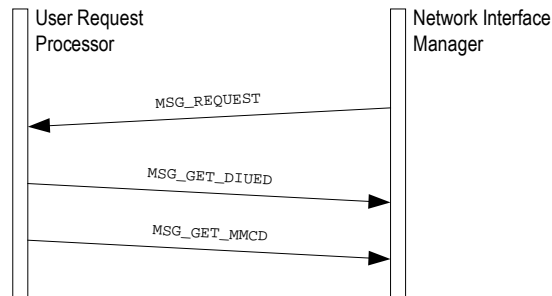


Figure 5.9 – Event scenario at the reception of an HTTP request.

5.2.3.2 Event Scenario 2: Multimedia Content Description Processing

When the relevant MPEG-7 description is retrieved (from a URL or from a local cache), a message is received by the URP, which then forwards the description (and the HTTP headers) to the MMCDBuilder with a MSG_PARSE_MMCD message. The MMCDBuilder parses the description into memory and checks the description validity (no semantic errors, e.g. the MediaLocatorD must reference the content being processed) so that it may be used. When the content description parsing is finished, the MMCDBuilder sends a MSG_MMCD_READY message to the CAD so that this module may analyze it and decide the content customization actions. This scenario is depicted in Figure 5.10.

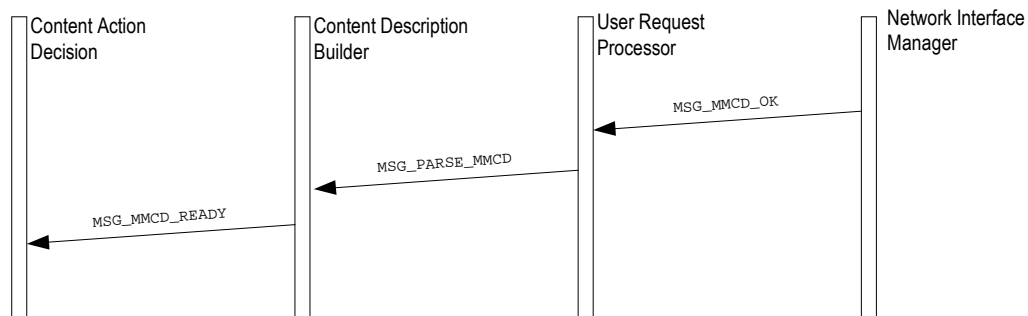


Figure 5.10 – Event scenario at the reception of the content description.

If no MPEG-7 description is available at a URL or in a local cache, the message MSG_NO_MCD is sent instead of MSG_MMCD_OK; the rest of the process is similar but now only the HTTP headers are processed to get knowledge about the content (by extrapolating some characteristics of the content).

5.2.3.3 Event Scenario 3: User Environment Description Processing

When the DIUED is retrieved (from some URL or from a local cache), a message MSG_DIUED_OK is received by the URP which then forwards the description (and the HTTP headers) to the DIUEDBuilder with a message MSG_PARSE_DIUED. The DIUEDBuilder parses the descriptions into memory and checks its validity so that it can be used. When the DIUED parsing is finished, the DIUEDBuilder sends a message to the CAD so that this module can analyze it and decide the content actions. This scenario is depicted in Figure 5.11.

If there is no DIUED available in the MPEG-21 DIUED server or in local cache, a default DIUED with predefined values is used instead and the rest of the process is equal.

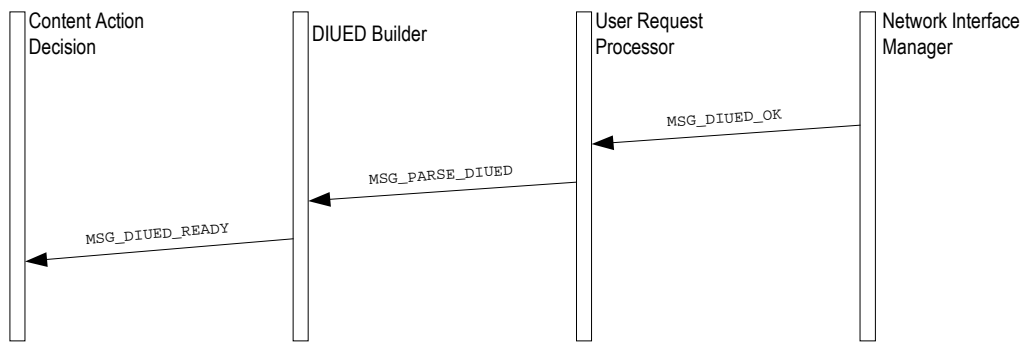


Figure 5.11 – Event scenario at the reception of the user environment description.

5.2.3.4 Event Scenario 4: Multimedia Content Processing

When the CAD module receives both the MSG_MMCD_READY and MSG_DIUED_READY messages, the decision process is initiated. After the decisions have been made, the CAD sends the list of adaptation actions to perform to the CC module with a MSG_CONTENT_ACTIONS message. The CC module will then send an MSG_GET_CONTENT to the Network Interface Manager to retrieve the selected content variation. When the content is retrieved, the Network Interface Managers signals the CC with an MSG_CONTENT_OK message so that the adaptation process may start if there is still any adaptation to do on the variation selected. This scenario is depicted in Figure 5.12.

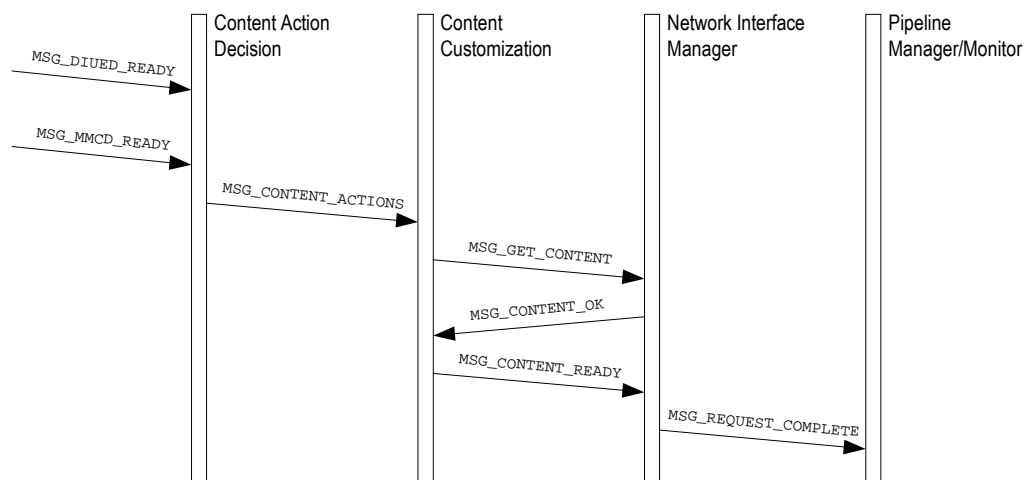


Figure 5.12 – Event scenario at the end of the DIUED and MPEG-7 descriptions parsing.

Before the content customization is started, the CC sends a `MSG_CONTENT_READY` to the Network Interface Manager, so that both modules start working in parallel. While the customized content is being generated by the CC, the NIM is sending the user the parts of the content that become ready. When the content adaptation is complete, a flag is set in the request context, which will inform the NIM that the request is finished.

When the NIM finishes sending the content to the user, it will send a `MSG_REQUEST_COMPLETE` message to the PM/M so that this one may insert the relevant data into the caching tables.

5.2.4 Messages

All modules communicate through messages, which state the action that must be performed on the handle of the request passed in the message. In order to send a message, four parameters are required: `msgType`, `msgTo`, `req` and `dbgMSG`.

The first parameter `msgType` indicates the type of message that is being sent; Table 5.5 lists all existing types of messages. The second parameter, `msgTo`, identifies the module to which the message is intended. The third parameter, `*req`, is the reference to the request context to which the message concerns and the last parameter, `dbgMSG`, is only free text for debugging purposes.

Table 5.5 – Message types.

Message	Description
<code>MSG_START</code>	Every module that receives this message must start processing.
<code>MSG_STOP</code>	Every module that receives this must stop its processing.
<code>MSG_STATUS</code>	The module that sent this message is informing the PM/M about its status.
<code>MSG_REQUEST</code>	The URP must process the received HTTP request.
<code>MSG_GET_DIUED</code>	The NIM must retrieve the Digital Item Usage Environment Description.
<code>MSG_DIUED_OK</code>	The NIM has retrieved the Digital Item Usage Environment Description.
<code>MSG_NO_DIUED</code>	The NIM could not retrieve the Digital Item Usage Environment Description.
<code>MSG_GET_MMCD</code>	The NIM must retrieve multimedia content description.
<code>MSG_MMCD_OK</code>	The NIM has retrieved the multimedia content description.
<code>MSG_NO_MMCD</code>	The NIM could not retrieve the multimedia content description.
<code>MSG_PARSE_MMCD</code>	The MMCDBuilder must parse the content description.
<code>MSG_PARSE_DIUED</code>	The DIUEDBuilder must parse the user environment description.
<code>MSG_MMCD_READY</code>	The CAD receives the parsed MMCD.
<code>MSG_DIUED_READY</code>	The CAD receives the parsed DIUED.
<code>MSG_CONTENT_ACTIONS</code>	The CAD sends the content actions to the CC.
<code>MSG_GET_CONTENT</code>	The NIM must retrieve the content.
<code>MSG_CONTENT_OK</code>	The NIM has retrieved the content.
<code>MSG_NO_CONTENT</code>	The NIM could not retrieve the content.
<code>MSG_CONTENT_READY</code>	The NIM may start sending the content while it is being generated.
<code>MSG_REQUEST_COMPLETE</code>	The NIM informs the PM/M that the request is completed.

5.3 UMA Platform Modules Design

Designing the UMA System involved the identification of the already listed modules and the definition of the data classes. After the basic architecture was defined, the messaging among the modules was designed.

For this, the event scenarios were used to assist in the design of each module and of the messaging. This section will provide a more detailed description on each module of the UMA Engine.

5.3.1 GUI

The GUI makes available to the UMA Engine Administrator a set of functionalities, which let him monitor and control the UMA Platform. The rest of the UMA Engine is independent of the GUI since it runs in separate threads, which communicate through messages that have been designed specifically for this platform. When the UMA Platform processing is started only this module exists; the Administrator must start the UMA Engine through the `Start UMA Engine` option. This action starts the PM/M, which will create all the others modules.

The complete list of available options that the GUI offers is presented with more detail in Chapter 7 when the visual interfaces will be presented.

5.3.2 Network Interface Manager (NIM)

The Network Interface Manager is the module that intercepts the user requests so that the UMA Engine can process them. This module was designed to provide the network integration to the UMA Engine, so that it could be tested in a real network scenario. With this module, the proxy server and content server based configurations can be easily used because the NIM retrieves the content from a local disk or from a URL.

If this module was not included in the UMA Engine, the content would be exclusively retrieved from the local hard disk and the requests would have to be received through the GUI instead of considering a more realistic scenario as it happens. The NIM is essentially an implementation of parts of the HTTP protocol to enable the UMA Engine to perform the following functions:

- Receiving user requests (listening to the HTTP command GET on an IP port of the computer).
- Receiving user environments posts (listening to the HTTP command POST on an IP port of the computer); this enables the UMA browser to send the DIUED directly to the UMA Platform.
- Retrieving content from any accessible URL when another module requests it;
- Retrieving descriptions from any accessible URL when another module requests it;
- Streaming the adapted content back to the user when the CC sends the adequate message;

In order to integrate the UMA Engine in any other system that would require a content customization engine (e.g. a multimedia messaging system), only this module would have to be modified.

5.3.3 Pipeline Manager/Monitor (PM/M)

The Pipeline Manager/Monitor main task is to manage and monitor the UMA Engine. When any other module sends or receives a message, the Pipeline Manager/Monitor automatically receives a message informing about it. With this method, the PM/M tracks the state of all requests and modules in the pipeline. All the tracked messages are passed to the GUI module for better human control. The managing tasks of this module are:

- Start/create new modules;

- Stop modules;
- Managing the caching tables.

When the Network Interface Manager completes a request by sending the customized content to the user, the PM/M receives the context of that request. A reference to the data generated or retrieved to satisfy a request may then be inserted in caching tables for later use. Only the PM/M is allowed to write in the caching tables, while all the others modules can only read from them. There are two caching tables:

- (1) **Table of users** - Table with the `User` objects corresponding to a certain IP, see Figure 5.13;
- (2) **Table of content** - Table with references to processed content sorted by URL and corresponding description, Figure 5.14;

The table of users is used to cache base DIUED descriptions that were retrieved from the MPEG-21 DIUED server. Each request sends the difference relatively to the base DIUED that is stored in the table.

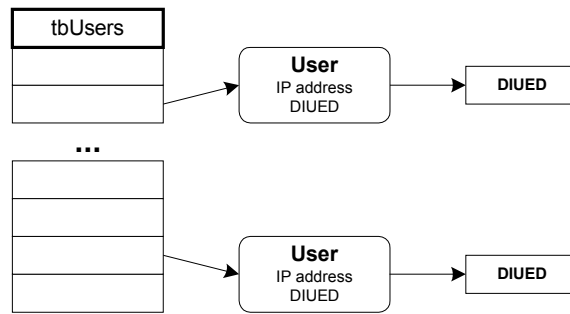


Figure 5.13 – Table of users (sorted by IP address).

The table of content stores `MMContent` objects that reference the source content data, the content description and the existing variations, as can be seen in Figure 5.14. `MMContent` objects in this table concern the source content exclusively. References to `MMContent` objects regarding content variations are not inserted directly in the table. Instead, the source `MMContent` object references its `Variation` objects, which reference the `MMContent` objects resulting from previous customizations. Figure 5.14 illustrates these cases, where a source content has several variations that can be located through the descriptions of the content variations.

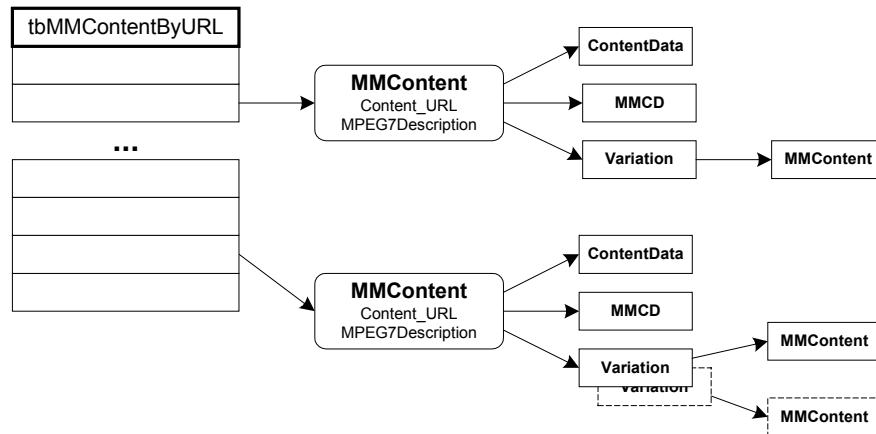


Figure 5.14 – Table of content (sorted by URL).

The PM/M could also implement a watchdog function, since through the `infoStatus` message it can know the state of each module; if a module is taking too long to perform some task than probably the module has entered in some endless loop (or any other software error) and it should be stopped in order to prevent the system collapse.

5.3.4 User Request Processor (URP)

The URP module processes the requests from the user and is responsible for using the caching tables for performance improvement. This module has also the task of gathering all information about the requesting users and requested URLs (multimedia content). The functions for this module are:

- Checking if the pretended content and user environment descriptions are referenced in the caching tables.
- Asking the Network Interface Manager to retrieve the relevant content description from the MPEG-7 description server.
- Asking the Network Interface Manager to retrieve the user environment description from the MPEG-21 DIUED server.
- If necessary, analyzing the request HTTP headers present in the user GET command to extrapolate information about the user.
- If there is no content description available, asking the Network Interface Manager to retrieve the content HTTP headers and analyze these HTTP headers to extrapolate information about the content.
- Sending the retrieved user environment description to the DIUEDBuilder to be parsed.
- Sending the retrieved content description to the MMCDBuilder to be parsed.

5.3.5 Multimedia Content Description Builder (MMCDBuilder)

The MMCDBuilder module parses and validates the MPEG-7 descriptions. For that it uses the MPEG-7/DIUED Library. The main functions of this module are:

- Parsing the content description to check its syntactic validity;
- Analyzing the content description to check its semantic validity (no parameter with invalid values, e.g. bits per pixel 0);
- Instantiating internal structures with the content descriptors that are relevant for the analysis to be performed by the UMA Engine.

The result of this process is a tree of objects stored in memory containing the UMA related information present in the MPEG-7 description. An internal reference to the appropriate data structures is then passed to the Content Action Decision module.

5.3.6 Digital Item User Environment Description Builder (DIUEDBuilder)

The DIUEDBuilder module parses and validates the DIUED. For that it uses the MPEG-7/DIUED Library. The main functions of this module are:

- Parsing the user environment description to check its syntactic validity;
- Analyzing the user environment description to check its semantic validity (no parameter with invalid values, e.g. a screen resolution of -10×200);
- Instantiating internal structures with the user environment descriptors that is relevant for analysis to be performed by the UMA Engine.

When there is no DIUED available for the concerned terminal, the DIUED Builder will examine the HTTP headers included in the content request and with that information will internally build a valid DIUED description. In this situation the information concerning the terminal can be very difficult to obtain, because not every terminal sends enough information, for example, a Pocket PC sends only the pixel width and height and the number of colors; for some applications this might not be enough.

The result of this process is a tree of objects stored in memory containing the DIUED description. An internal reference to the appropriate data structures is then passed to the Content Action Decision module.

5.3.7 Content Action Decision (CAD)

The Content Action Decision module is concerned with the decision of what type of adaptation operations must be performed to the multimedia content. This module knows exactly which adaptation methods are available in the following pipeline step and thus must decide the content operations based on the available methods, after matching the content description with the user environment description. This module will be detailed in Chapter 6.

5.3.8 Content Customization (CC)

The Content Customization module executes the content customization actions ‘ordered’ by the CAD module. This module performs the signal processing tasks and thus the re-purposing of the multimedia content. This module may integrate several libraries providing functions for the manipulation of different types of content: in the UMA System implemented, images are handled with the *Image Magick* library [12], and video with an adapted version of the ISO MPEG-2 encoder/decoder [13].

This module is also responsible for asking the Network Interface Manager to retrieve the selected content variation in order to perform the necessary adaptations when this content becomes available. This module will be detailed in Chapter 6.

5.4 Summary

All the studies made in the previous chapters culminated in this chapter with the design of UMA System and UMA Engine architectures presented in Figure 5.2 and Figure 5.7, respectively. This chapter presented an architecture analysis and detailed the process to design and implement the UMA Engine. The architecture adopted is flexible since the entire customization process is highly modular which allows a high optimization and performance tuning according to the results pretended.

The Content Action Decision and the Content Customization modules are highly dependent on each other: the first one must have an up to date knowledge of the adaptation methods available; in a more sophisticated implementation, the decisions should take into account the processing queue of the second module. The content processing modules are the most important of the UMA Platform; the following chapter will present the content customization processes that are performed in these modules.

Chapter 6

Content Customization Processing

In the previous Chapter, the internal organization of the UMA Platform as well as the function of each block that composes it was presented. Since Chapter 1 states that the UMA Engine must be able to handle all the basic media types (text, audio, image and video), the UMA Engine was designed so that it is architecturally prepared to adapt any media type: it has a modular structure that allows any type of content to be processed by the corresponding adaptation methods.

This Chapter will present the entire content customization process performed by the two modules presented in Chapter 5: the Content Action Decision and the Content Customization modules. Besides these two major modules, two data type specific customization modules implementing image and video adaptation methods are also described.

6.1 Content Action Decision Algorithms

The Content Action Decision module is responsible for the decision of the precise customization methods that are to be performed on the content to be adapted. This module knows exactly which adaptation methods are available in the Content Customization module and must decide the content adaptation actions based on the available customization operations, the MMCD and the DIUED (the content and user environment descriptions). The actions are executed by the Content Customization module, which receives the list of actions to perform through a buffer.

When both descriptions are received, the Content Action Decision module initiates the decision process. First, it checks the UMA Engine state (configured using the “UMA Platform Properties” dialog box), and according to its state it acts accordingly:

- **Bypass mode:** the UMA Engine is configured so that no customization operations are to be performed and thus the content is transmitted without any modifications (bypassed).
- **Normal mode:** the UMA Engine is configured so that the content is customized using the actions determined after the analysis of both the content and user environment descriptions and taking into account the available adaptation operations.

The relevant UMA Engine configuration mode in terms of content adaptation is the normal mode. Therefore in this Chapter the normal mode is always assumed to be the one selected.

When starting the customization decision process, the media type of the content in question is checked and depending on it, the corresponding decision function is called: in the context of this thesis, decision

functions for image and video content were implemented; a simple scheme illustrating the decision process is presented in Figure 6.1.

All decisions taken by each of these functions are stored in two buffers to be passed to the Content Customization module: the first buffer is called *Action_Stack* and it is used for storing the content adaptation actions; the second buffer is called *Action_Parameter_Stack* and it is used for storing the content adaptation actions' parameters regarding each content adaptation action, e.g. the new spatial resolution for a spatial resolution conversion.

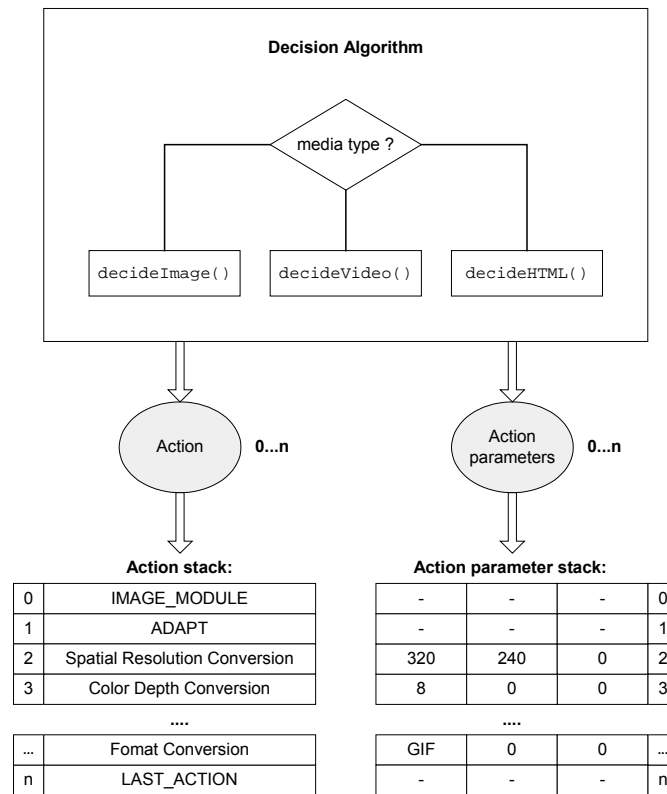


Figure 6.1 – Example decision process for image customization.

A **customization module** is like a plug-in, which implements a set of operations to adapt a certain type of content. The Content Customization module has a table where several customization modules are listed and thus available to perform the content customizations (in the next section this will be detailed). Each decision function must first select which customization module will be used to handle the concerned content and after select the relevant adaptation actions and corresponding parameters for the selected customization module.

The *Action_Stack*'s first two entries are intended as commands for the Content Customization module's logic: the first entry selects the specific content customization module that must process the content (in the example above it is the *IMAGE_HANDLER* which means that an image is to be processed), and the second entry tells the Content Customization module how the content adaptation must be performed.

For this, the second entry of the *Action_Stack* can have one of the following three values:

- **ADAPT**: the content customization decisions are to be applied to the original content;

- **VARIATION:** an available content variation must be used and the content customization actions performed over this variation;
- **REMOVED:** the content is to be removed; this corresponds, for example, to the case where a certain media object in the context of composite content is not to be used.

The second part of the output of the content action decision module corresponds to the specific adaptation actions and corresponding parameters, as illustrated in Figure 6.1. While the decision algorithm starts generating the adaptation actions, they will be stacked in the `Action_Stack`; after the last adaptation action has been stacked, the algorithm inserts the `LAST_ACTION` key value that closes the `Action_Stack`. As shall be seen in the next section, the order in which the actions are inserted in the stack is not relevant for the adaptation processes implemented but it could be for other adaptation methods.

The adaptation decision algorithms implemented in this thesis for image and video customization are presented in Figure 6.2 and Figure 6.3, respectively:

- The **image decision algorithm** is rather simple but it is general enough to be applicable to any image format (e.g. JPEG, GIF): the proposed decision algorithm considers as image adaptation parameters, the spatial resolution (height and width in pixels), color domain and number of bits per pixel of the image. More complex decision algorithms could have been implemented by considering other adaptation solutions such as scalable coding formats, or decreasing the image quality (changing the quantization step) to match the maximum delay permitted by the user (image retrieval time).
- The **video decision algorithm** starts by executing the same algorithm as for the images but in this case applied to the frame spatial resolution and color characteristics taking into account the description of the user environment. After this first step, the ratio between the customized frame data size, `newFrameByteSize`, and the source frame data size, `oldFrameByteSize`, is calculated as `downRatio`. Then the customized video bit rate, `newVideoBitRate`, is estimated to be equal to the source video bit rate multiplied by the `downRatio` factor. This is a rather rough approximation to know which should be the new video bit rate in order to maintain the same compression ratio as before. Then the `newVideoBitRate` is compared with the `networkBitRate` and the lower value is used to encode the customized video.

6.2 Content Customization Algorithms

The Content Customization module described in this section executes the instructions received from the Content Action Decision module. The Content Customization module is responsible for the entire signal processing tasks applied to the content in order to get the pretended tailored content. The Content Customization module includes a table that groups the adaptation methods to process the content in terms of data types, e.g. image, video, etc. These groups of methods are called customization modules. Each customization module groups adaptation methods concerning a media type. The table of customization modules has sixteen entries each one corresponding to a pre-defined customization module as illustrated in Figure 6.4.

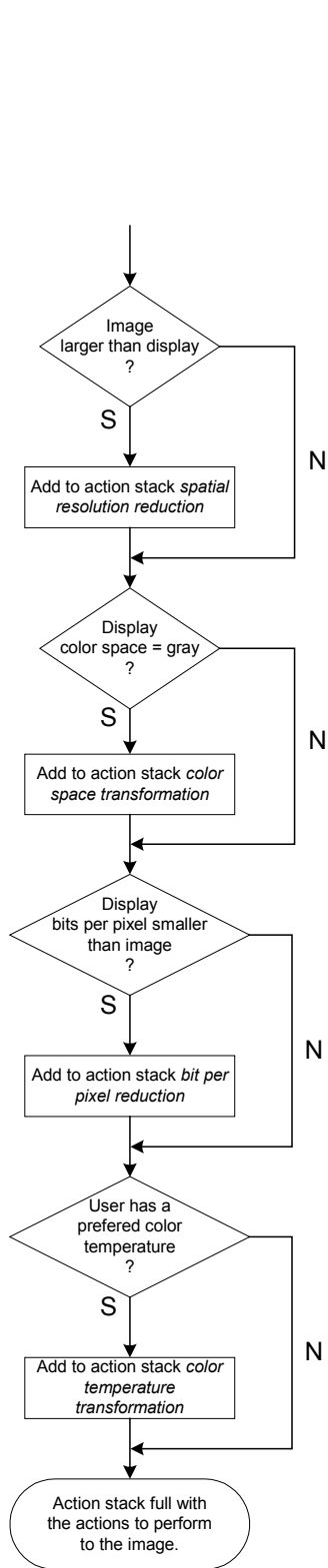


Figure 6.2 – Image decision algorithm.

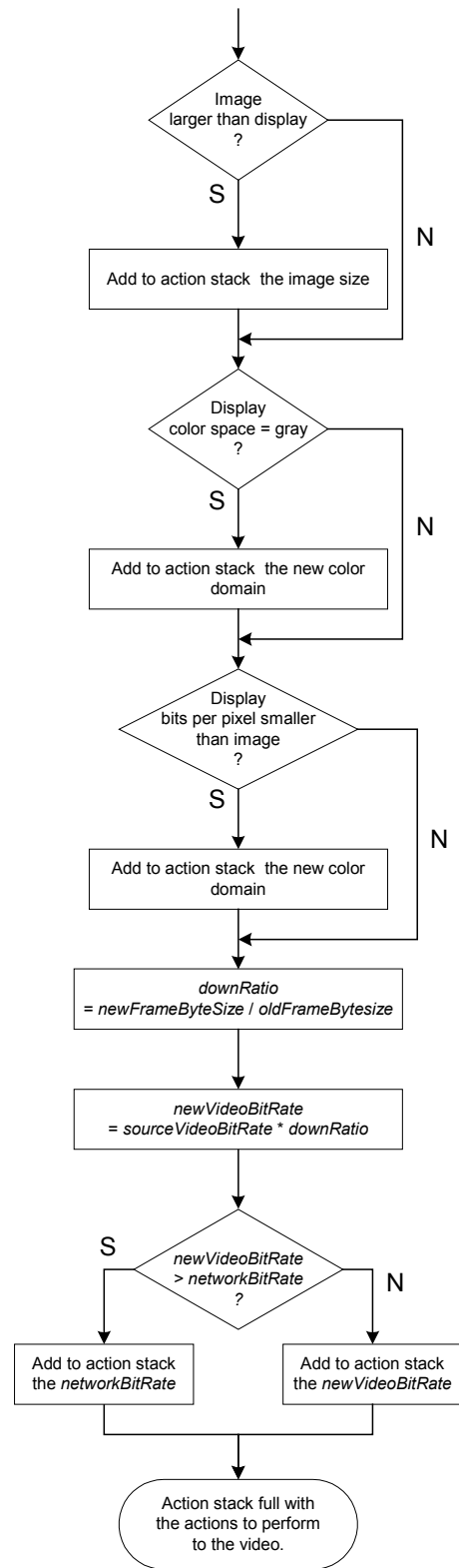


Figure 6.3 – Video decision algorithm.

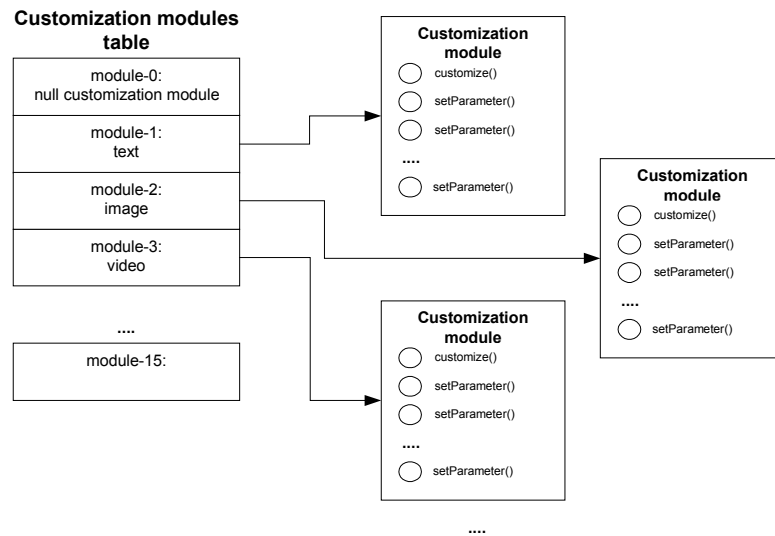


Figure 6.4 – The customization modules table.

A complete list of the customization modules already considered in the customization modules' table is presented in Table 6.1. As can be observed in this table, not all customization module entries have already a predefined function: there are spare entries in the table that have no adaptation task assigned for the moment preparing the table for the inclusion of additional customization modules. This offers a development environment and easy integration of new customization algorithms that may be developed. From the attributed customization module entries only three have been implemented: the first one, called NULL, is the one corresponding to the case when the content needs no adaptation action; the others implemented correspond to image and video content.

Table 6.1 – List of customization modules

Customization Modules	Customization methods types	Status
Module 0	NULL (Bypass content)	Implemented
Module 1	Text adaptation methods	-
Module 2	Image adaptation methods	Implemented
Module 3	Video adaptation methods	Implemented
Module 4	Audio adaptation methods	-
Module 5	Speech adaptation methods	-
Module 6	Structure adaptation methods	-
Module 7	Interactive elements adaptation methods	-
Module 8 to Module 15	-	-

Concerning the customization modules, there are two main methods to access the pretended customization module:

- `setParameter()`: for each adaptation action, the customization module must have the parameters that will guide that action; with `setParameter()`, the Content Customizer configures the parameters for each specific adaptation method.

- `customize()`: after the Content Customizer has set all the actions and the corresponding parameters, the `customize()` method is used to execute the content adaptation processing, action by action, sequentially.

The Content Customization module has a distributor that fetches the actions from the `Action_Stack` and passes them to the corresponding customization module to process the content. For each media type, a corresponding customization module must exist. If the media type does not have the corresponding customization module, the Content Action Decision module will select the NULL customization module.

When the Content Customization module creates the customization modules table, the image and video customization modules are instantiated and inserted in the table. Even though there is space for more customization modules, only these two are used.

Figure 6.5 presents the table of customization modules and the way the `Action_Stack` and the `Action_Parameter_Stack` are used to index the table:

- 1) The content of the first position of the `Action_Stack` selects the customization module that will process the multimedia content;
- 2) The following position of the `Action_Stack` tells the Content Customizer which content must be adapted (the source content or a content variation); the Content Customizer asks then to the Network Interface Manager to retrieve the content;

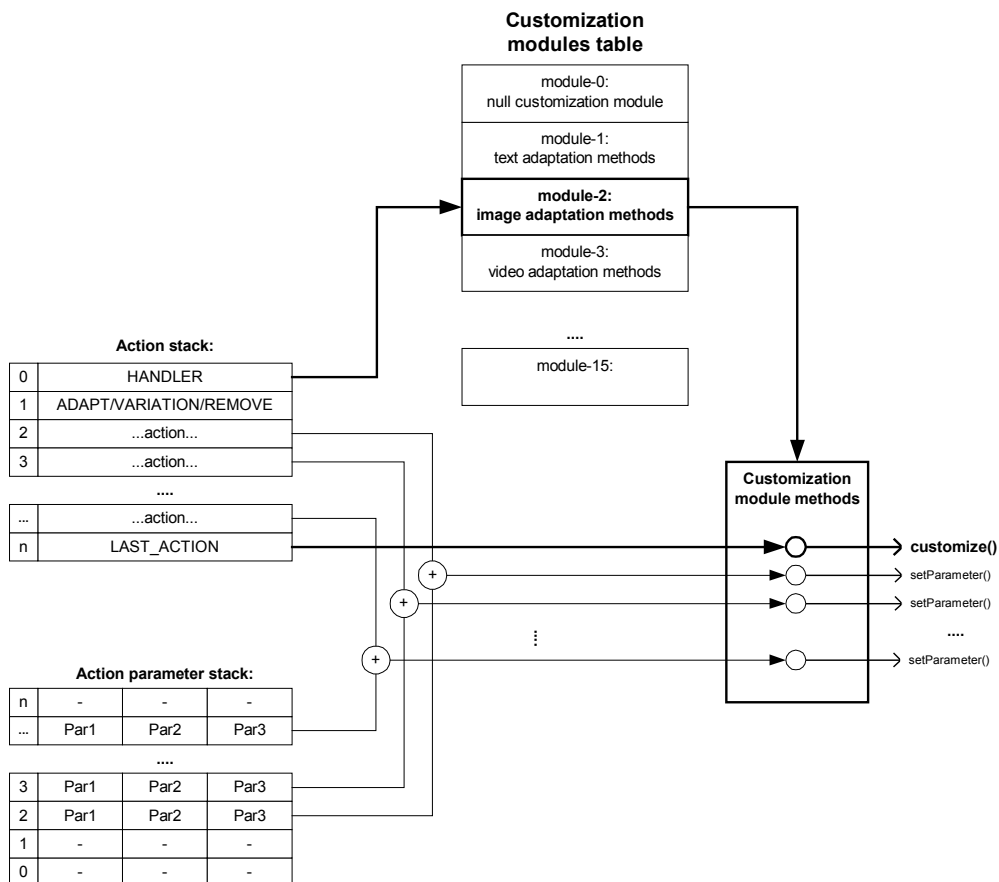


Figure 6.5 – The table of customization modules implements an interface to the set of content processing methods available.

- 3) After the content has been retrieved, the Content Customizer gets the list of adaptation actions from the `Action_Stack` and the corresponding list of parameters from the `Action_Parameter_Stack` and uses the `setParameter()` method to load the content actions and parameters into the relevant customization module.
- 4) Next, the Content Customizer forwards the content request that it is processing to the Network Interface Manager (NIM), right before it calls the `customize()` method. This way, both modules will be working together on the same request: the NIM will start sending the customized content to the user while the Content Customizer is writing it into the memory, thus preventing long delays as would happen if the transmission would start only after the completion of the content customization process. This functionality is less relevant for images, but it is quite important for video, because the video is played while it is being adapted.
- 5) Finally, the Content Customizer executes the `customize()` method. This method will execute all adaptation methods that were set previously using the `setParameter()` method.

The adaptation methods are not accessed individually: they are all sequentially executed inside the `customize()` method; this solution avoids that different adaptation methods perform the same operations unnecessarily, in this case, decode and encode the source content.

6.3 Image Customization Module

The image customization module implements a set of adaptation methods to process image content, as presented in Table 6.2. The image adaptation algorithms have been implemented using the Image Magick library [12] that offers a collection of basic image processing algorithms. Each adaptation operation must be configured by an appropriate method, `setParameter()`, which sets the parameters for the concerned adaptation. After all the adaptation operations have been set, a final method, `customize()`, is called to execute all the configured adaptation operations.

Table 6.2 – Image adaptation operations supported by the image customization module.

Customization method	Description	Status
Color depth reduction	Reduces the number of bits per pixel.	Implemented
Color space transformation	Reduces the color space (from color space to gray scale).	Implemented
Spatial resolution reduction	Reduces the image spatial resolution.	Implemented
Spatial resolution reduction using a Region of interest	Reduces the spatial resolution considering a region of interest.	Implemented
Compression ratio increase	Increases the compression ratio (lose quality/ reduce retrieval time)	Implemented
Format conversion	Converts the image to another format, (e.g. from JPEG to GIF).	Implemented
Replace by variation	Selects another variation.	Implemented
Color temperature transformation	Customization according to color temperature preference.	Implemented
Operation 9 to Operation 15	Free.	-

Figure 6.1 illustrates two adaptation methods: the bits per pixel reduction and the color space transformation.

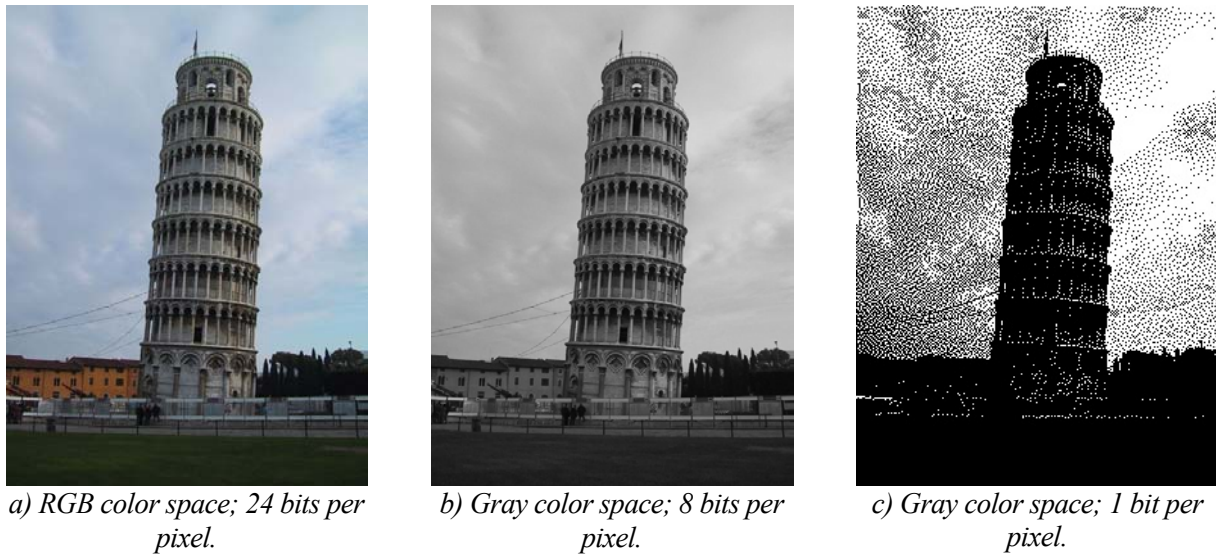


Figure 6.6 – Example of color space transformation, from a) to b), and bits per pixel reduction, from b) to c).

Only when the `customize()` method is executed, the content is processed. The `customize()` method performs the following steps:

1. opens and decodes the image into memory using Image Magick primitives;
2. using the new parameters the image is adapted using Image Magick primitives; and
3. finally, the image is encoded in the output format using Image Magick primitives.

Figure 6.7 illustrates the steps above. The `customize()` method uses a simple algorithm for performing image content customization. The performance in terms of execution time and memory consumption that this method offers is rather poor because it processes the content in the uncompressed domain and therefore requires large computational resources.

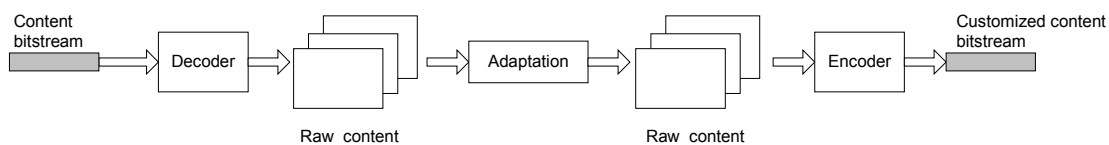


Figure 6.7 – Uncompressed domain adaptation.

With uncompressed domain adaptation methods, images, video and audio require large resources in terms of memory and CPU usage to be processed. This approach is inefficient because the content must be passed into the uncompressed domain to be adapted and then recompressed. Besides the high computational costs, this solution also implies a quality reduction after several adaptations are performed due to the accumulation of the coding (quantization) errors. Therefore, to improve the performance of the UMA Engine, it would be vital that adaptation methods able to process the content in the compressed domain or semi-compressed domain (see Figure 6.8) are adopted. Such a solution would increase the quality of the adapted content and decrease the resources required to customize the content.

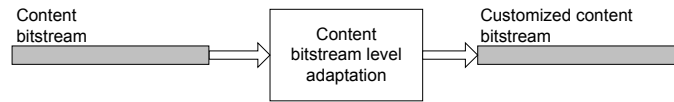


Figure 6.8 – Compressed domain adaptation.

Compressed domain adaptation methods would be faster and would consume much less memory; however, this type of processing is more complex in terms of implementation and thus was considered not feasible within the time limits of this thesis.

6.3.1 Color Temperature Adaptation

The color temperature is a measure that relates a color with a temperature. A light source, which is illuminating a scene, has several characteristics that can be measured. Such a light source is usually called the illuminant of a scene: one of the possible measures that can be performed is the color of that light, which is called the illuminant chromaticity. Another measure that can be made from the scene illuminant is the color temperature of that light (the illuminant color temperature). The illuminant color temperature can be measured directly from the light source using hardware sensors. However, with a digital image of the scene, it is possible to measure the illuminant chromaticity and from this value calculate the illuminant color temperature of the scene illuminant. Figure 6.9 illustrates the described problem. From now on when the term “color temperature” is used in this text with the meaning of “scene illuminant color temperature”.

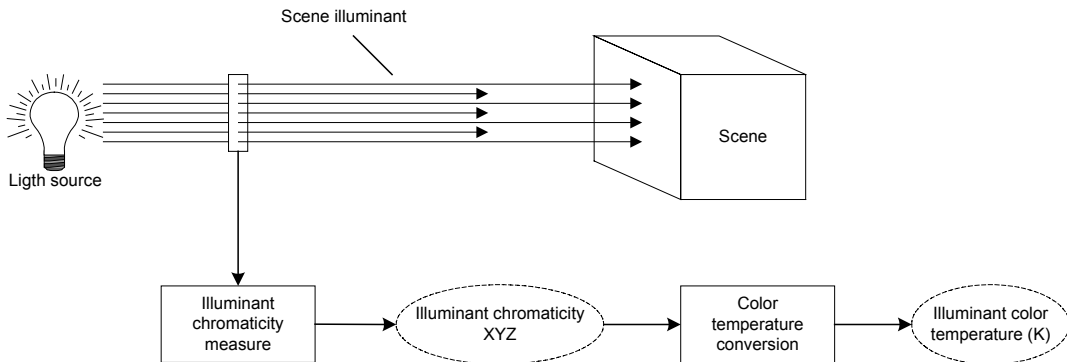


Figure 6.9 – Light source, illuminant chromaticity and color temperature concepts.

Color temperature is a measure, which humans can relate to a feeling of warm or cold in an image. This measure is used in UMA applications to customize the image color temperature according to the user preferred color temperature. In order to have this functionality, a color temperature measurement algorithm (to instantiate the descriptor) and a color temperature transformation algorithm (to perform color temperature adaptations) have been implemented in the context of this thesis, notably in the MPEG-7 description tool and in the UMA Platform, respectively.

MPEG is developing a low-level visual descriptor of color temperature and the corresponding user preference descriptor to be included in MPEG-7 Version 2. This novel low-level descriptor offers the UMA Engine the capability to adapt image and video data according to the user preference in terms of color temperature.

6.3.1.1 Fundamentals on Color Temperature

Color temperature is a term used to describe properties of light sources. In order to define color temperature, the concept of a *blackbody radiator* must be defined. A blackbody is a theoretical object, which

is a perfect radiator – it absorbs all the incident radiation, and reradiates that energy with complete efficiency. As the temperature of a blackbody rises, it radiates energy in the visible range, first red, changing to orange, yellow, white and, finally, bluish. These colors plot as a curved line in the CIE⁶ chromaticity diagram, see Figure 6.11. This curve is known as the *blackbody locus*.

Note that the color temperature is not a measure of the physical temperature of the light source. It is the temperature of the blackbody radiator when the color appearance is the same as the source being tested. Ironically, a low color temperature corresponds to what is considered a warm appearance (red and orange) while a high color temperature corresponds to a cool appearance (bluish light).

The theoretical model of a blackbody is the reference to which any other light source is compared. The physicist that derived this model was Max Planck. The equation of the spectral radiant power of a blackbody radiator as a function of the temperature T and wavelength λ is given by:

$$M = c_1 \cdot \lambda^{-5} \cdot \left(e^{\left(\frac{c_2}{\lambda T} \right)} - 1 \right)^{-1}$$

where $c_1 = 3.7418 \times 10^{-16}$ W/m² and $c_2 = 3.7418 \times 10^{-2}$ W/K. This equation describes the maximum theoretical power that can be emitted at a given wavelength and temperature. From Planck's equation, it is possible to determine the energy that a certain light emitter radiates at a given wavelength. The spectral power distributions of a blackbody shown in Figure 6.10 are normalized to the spectral power corresponding to a wavelength of 550 nm. This power is normalized to 550 nm because this wavelength is near to the peak luminance response of the human eye. It can be observed that as the blackbody temperature is increased, the output in the blue increases and the output in the red decreases (high and low wavelengths, respectively).

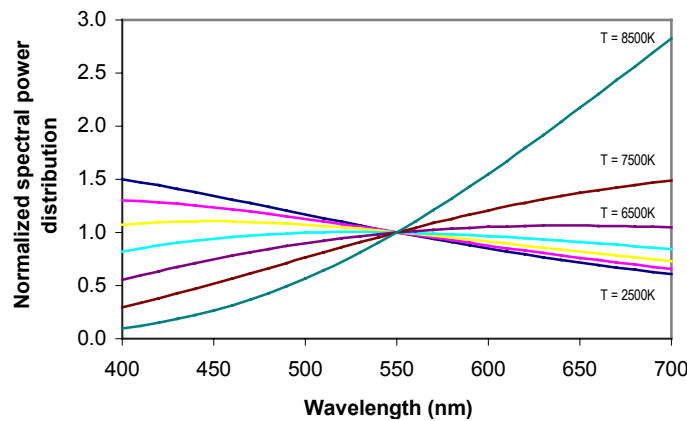


Figure 6.10 – Normalized spectral power distribution for blackbody radiators.

The diagram in Figure 6.11 is obtained by converting the spectral power distribution data from Planck's equation to chromaticity coordinates plotted in a CIE diagram. The arc on the plot shows the change in chromaticity coordinates from 1,000 K to 10,000 K. The CIE diagram also allows colors near the blackbody locus to be defined as correlated color temperature. The mechanism for computing the

⁶ The Commission International de l'Eclairage is the technical body that handles many of the problems related to color perception and color standards.

correlated color temperature for colors out of the arc is nontrivial. This method is described in [126] and was implemented in this thesis to calculate the color temperature.

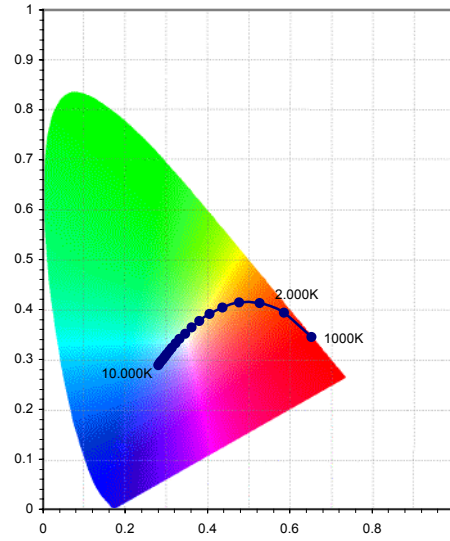


Figure 6.11 – CIE-xy color system and blackbody locus.

The illuminant chromaticity corresponds to a point in the CIE-xy diagram. Since it is impossible for any light color coordinates to fall over the blackbody locus, there are methods (described later) to correlate a color with a blackbody locus color. Thus calculating a correlated color temperature of the scene illuminant. This is a very high level presentation of how an image color temperature is calculated from a scene illuminant chromaticity.

6.3.1.2 Color Temperature in MPEG-7

For a given digital image, the color temperature can be calculated directly from the image. Since illumination is an important factor that affects the user's feeling of an image, each person feels differently on two images of the same content with different color temperatures. Therefore, there exist personal preferences in terms of image color temperature. In Figure 6.12, the source image looks to be taken under twilight while in Figure 6.13 it looks to be taken under daylight.



Figure 6.12 – Picture with sunset illumination
($T = 3,629\text{ K}$).



Figure 6.13 – Picture with day light illumination
($T = 6,194\text{ K}$).

Some people prefer the first image because of the warm feeling while others may prefer the second image with a cool and bright feeling. In fact, the visual impression from the images in Figure 6.12 (source image) and Figure 6.13 (transformed image) is rather different although the only difference between them is in terms of its color temperature. It is also known that there is a considerable difference between the color

temperature preferences for Oriental and European people. While Europeans usually prefer temperatures around 5,000 K, Oriental people prefer temperatures around 9,300 K, [131].

There are several methods to transform the original image with a certain color temperature into another image with a different color temperature. To perform a color temperature customization, information regarding three items is required:

- **Color temperature of the input image and gamma correction parameters related to its capturing device** – the color temperature can be directly obtained from the image or from an MPEG-7 description with the MPEG-7 color temperature descriptor instantiated; the gamma correction characteristics of the device that captured the image can be obtained from the MPEG-7 descriptors regarding media information.
- **Color temperature user preferences** – the user preferences in terms of color temperature can be obtained by a direct user input or by asking the user to select from a list of examples of color temperature transformation results the one that he/she prefers, i.e. selection by choice.
- **Display device features** – to achieve a precise color temperature transformation, some features regarding the display device are needed such as the gamma correction information and the color temperature of the display device. This information may be obtained from the display device driver. Furthermore, if the user preference on color temperature was obtained by selecting examples (as usually is), the display device feature information would be optional, because the device characteristics are already taken into account in the user selection.

Methods to measure the color temperature of an image and adapt it to the user preferences have been proposed to MPEG by the Samsung Advanced Institute of Technology [130]. Both algorithms have been implemented in this thesis and will be presented after introducing the specification of the MPEG-7 color temperature descriptor.

6.3.1.2.1 ColorTemperatureD

In this thesis, the MPEG-7 descriptor was used to specify image color temperatures and user color temperature preferences. The MPEG-7 color temperature descriptor semantics is quite simple: the ColorTemperatureValue parameter represents the color temperature of the given image/region. The range of the color temperature is [1,667; 25,000] in Kelvin units. The descriptor syntax is as follows:

```
<!-- ##### -->
<!-- Definition of MPEG-7 ColorTemperatureType -->
<!-- ##### -->
<complexType name="ColorTemperatureType" final="#all">
  <complexContent>
    <extension base="mpeg7:VisualDType"/>
    <sequence>
      <element name="ColorTemperatureValue"
        type="mpeg7:unsigned15"/>
    </sequence>
  </extension>
</complexContent>
</complexType>
```

In a similar way, the user may specify his/her preference in terms of color temperature using this descriptor; however the descriptor is used under the UserPreferencesDS that aggregates within MPEG-7 all user preferences related information.

6.3.1.3 Measuring Color Temperature

A method for extracting the scene illuminant chromaticity from an arbitrary scene can be implemented following two different approaches: hardware or software. The first method uses sensors to measure the scene illuminant chromaticity directly, while the second starts from the image obtained from the input devices, e.g. a camera, and uses an algorithm to calculate the illuminant chromaticity. The software method proposed to MPEG [129] can be used for the estimation of the illuminant chromaticity and combines both the **highlight** and **perceived illumination** algorithms explained in the following.

Every physical body has two surface reflection characteristics: the diffuse and the specular reflection. When a surface is illuminated with a certain light source, part of that light is reflected with the same angle: this is the **specular reflection**. The other part of the light will initially penetrate into the object and will be after, partly absorbed and partly scattered back to the surface: this is the **diffuse reflection**. Specular reflection depends on the characteristics of the surface considered; it will be small for a matt surface and large for a glossy surface. The sum of these two types of reflection gives the total reflected light from an arbitrary colored surface. Under certain conditions of direct illumination and viewing geometries, some surfaces show **highlights**, (e.g. in the human skin this effect is easily observed corresponding to the white zones on the tip of the nose). On [133] Shafer's proposes a method to measure the highlights; from this method, the illuminant chromaticity can be calculated. This is what is sometimes called the *highlight method*.

Perceived illumination is a method to estimate the illuminant chromaticity from a color image by selectively excluding self-luminous regions. Humans can feel the global tone when looking at some arbitrary scene and exclude the light that is reflected by some surfaces. This tone information is proportional to the scene illuminant chromaticity.

The self-luminous regions correspond to the areas in the scene that do not correspond to passive surface reflection areas. Self-luminous regions can also be thought of as active reflectors like a lamp itself, the light passing through an aperture in a wall, or some specular reflection of an arbitrary surface. In general, the component average measure (in the XYZ color space) in self-luminous regions causes some error in the output value of the illuminant chromaticity. If these areas are excluded from the illuminant chromaticity calculation, the accuracy for the estimation will be increased. The estimation results are expressed as color coordinates, such as CIE-xy color space. Since the measure pretended is a color temperature, the chromaticity coordinates are correlated with the blackbody locus coordinate corresponding to a certain color temperature.

The method that has been proposed to MPEG allows to estimate the illuminant chromaticity by combining both the **perceived illumination** and **highlight** algorithms. The perceived illumination approach can provide a stable candidate range for the estimation of the illuminant chromaticity but the accuracy is not high and depends on the image content. The highlight method is not dependent on the image content and gives accurate values for the scene illuminant chromaticity, but it is difficult to select the final solution among those possible. The method for extracting the scene illuminant color temperature of a color image is presented in Figure 6.14. At least part of this method will be standardized by MPEG in MPEG-7 Version 2.

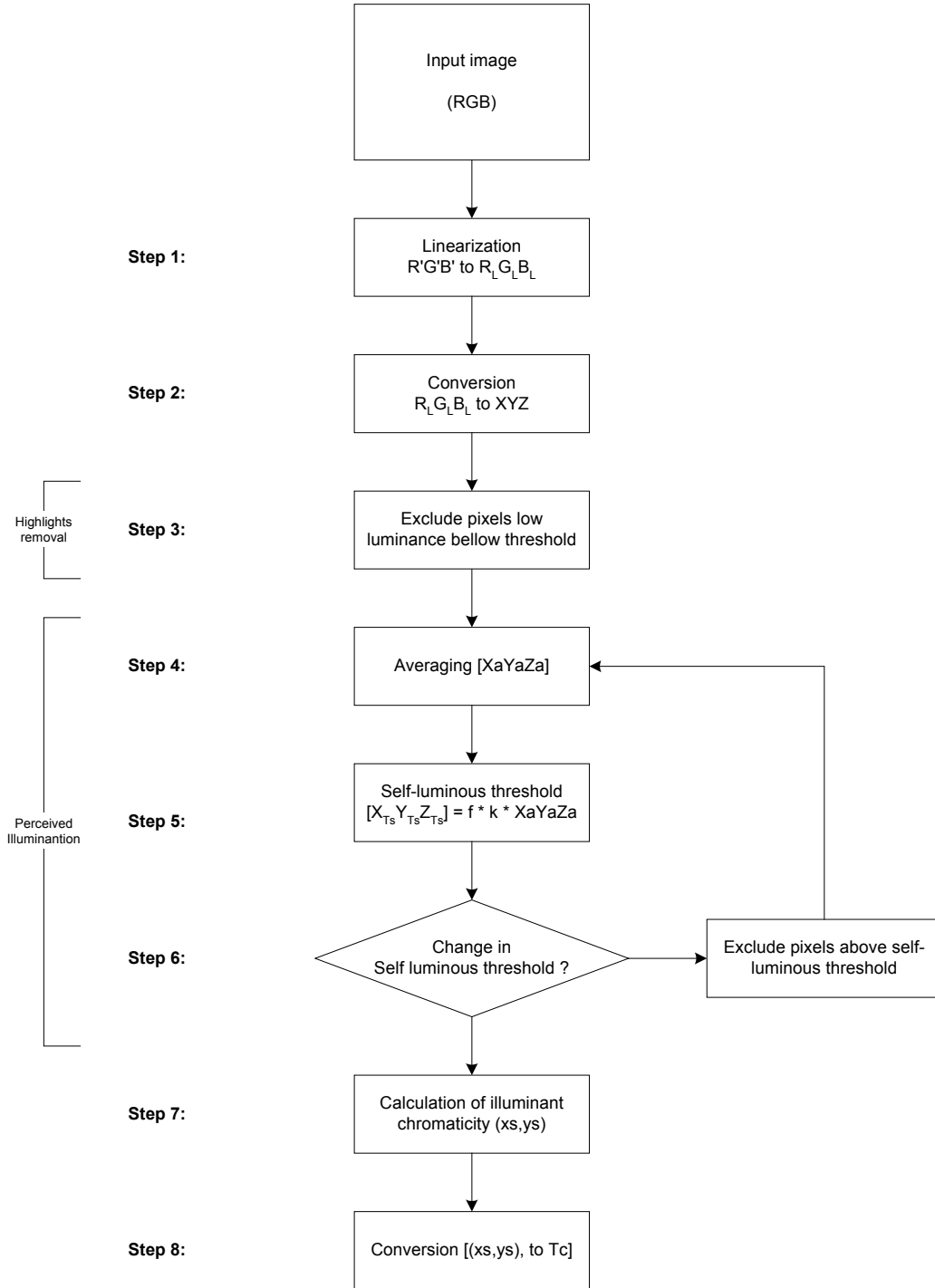


Figure 6.14 – Flowchart for computing the image color temperature.

Each of the steps involved in the calculation of the color temperature for an image is detailed next [128]:

- **Step 1:** Obtain the linearized $R_L G_L B_L$ by the inverse gamma correction of the $R'G'B'$ input (original image), which is gamma-corrected for the display devices (it is assumed that the input image is gamma-corrected in the range of 0 to 255):

$$\text{if } R'(i, j), G'(i, j), B'(i, j) \leq 0.03928 \times 255.0$$

$$R_l(i, j) = \left(\frac{R'(i, j)}{255} \right) \div 12.92$$

$$G_l(i, j) = \left(\frac{G'(i, j)}{255} \right) \div 12.92$$

$$B_l(i, j) = \left(\frac{B'(i, j)}{255} \right) \div 12.92$$

else $(R'(i, j), G'(i, j), B'(i, j)) > 0.03928 \times 255.0$

$$R_l(i, j) = \left[\frac{\left(\frac{R'(i, j)}{255} \right) + 0.055}{1.055} \right]^{2.4}$$

$$G_l(i, j) = \left[\frac{\left(\frac{G'(i, j)}{255} \right) + 0.055}{1.055} \right]^{2.4}$$

$$B_l(i, j) = \left[\frac{\left(\frac{B'(i, j)}{255} \right) + 0.055}{1.055} \right]^{2.4}$$

where (i, j) are the pixel coordinates.

- **Step 2:** Convert the linearized R_l, G_l, B_l into the XYZ color space using the conversion matrix M

$$\begin{bmatrix} X(i, j) \\ Y(i, j) \\ Z(i, j) \end{bmatrix} = M \bullet \begin{bmatrix} R_l(i, j) \\ G_l(i, j) \\ B_l(i, j) \end{bmatrix} \quad \text{where } M = \begin{bmatrix} 0.4124 & 0.3576 & 0.1805 \\ 0.2126 & 0.7152 & 0.0722 \\ 0.0193 & 0.1192 & 0.9505 \end{bmatrix}$$

- **Step 3:** Exclude the regions that have a luminance value smaller than the low luminance threshold (T_{ll})

$$\begin{cases} Y(i, j) < T_{ll}, & p(i, j) = 0 \\ \text{otherwise}, & p(i, j) = 255 \end{cases}$$

where the matrix $p(i, j)$ is a buffer keeping track of the pixels that shall be used in the following algorithm calculations. The low luminance threshold corresponds to 5% of the maximum allowable luminance value (e.g., if the maximum luminance is 1, then $T_{ll} = 1 \times 5\% = 0.05$).

- **Step 4:** From this step to step 6, the illuminant chromaticity is calculated iteratively until a stable measure has been reached. The average of the XYZ values for all pixels with $p(i, j) = 255$ is

calculated. The value $remain_p$ corresponds to the number of all pixels that have not been excluded (have $p(i,j) = 255$).

$$X_a = \frac{1}{remain_p} \sum_{i=0}^{row-1} \sum_{j=0}^{col-1} \left(X(i,j) \cdot p(i,j) / 255 \right),$$

$$Y_a = \frac{1}{remain_p} \sum_{i=0}^{row-1} \sum_{j=0}^{col-1} \left(Y(i,j) \cdot p(i,j) / 255 \right),$$

$$Z_a = \frac{1}{remain_p} \sum_{i=0}^{row-1} \sum_{j=0}^{col-1} \left(Z(i,j) \cdot p(i,j) / 255 \right).$$

- **Step 5:** Compute the self-luminous regions threshold: $X_{T_s}, Y_{T_s}, Z_{T_s}$

$$X_{T_s} = f \times k \times X_a,$$

$$Y_{T_s} = f \times k \times Y_a,$$

$$Z_{T_s} = f \times k \times Z_a.$$

where $f \times k$ corresponds to the estimated illuminant level; according to [129], 3.0 is an adequate value for $f \times k$.

- **Step 6:** If $X_{T_s}, Y_{T_s}, Z_{T_s}$ have the same values of the previous iteration, go to **Step 7**; otherwise exclude from the computations for each X,Y,Z component the pixels that have a pixel value above the self-luminous threshold and repeat **Step 4** to **Step 6**.

If $(X_{T_s}(t) = X_{T_s}(t-1), Y_{T_s}(t) = Y_{T_s}(t-1), Z_{T_s}(t) = Z_{T_s}(t-1))$ {

go to **Step 7**.

} else {

$$\begin{cases} X(i,j) > X_{T_s}(t) \text{ or } Y(i,j) > Y_{T_s}(t) \text{ or } Z(i,j) > Z_{T_s}(t), & p(i,j) = 0 \\ otherwise, & p(i,j) = 255 \end{cases}$$

go to **Step 4**.

}

where t is the iteration order for T_s ; moreover $X_{T_s}(0) = 0, Y_{T_s}(0) = 0, Z_{T_s}(0) = 0$.

- **Step 7:** Average the XYZ values for all remaining pixels (with $p(i,j) = 255$); these values correspond to the illuminant tri-stimulus value (again, $remain_p$ intuitively means the number of all pixels remained, which have $p(i,j) = 255$). Compute the scene illuminant chromaticity coordinates (x_p, y_p) in the CIE-xy diagram.

$$\begin{aligned}
 X_s &= \frac{1}{\text{remain_p}} \sum_{i=0}^{\text{row}-1} \sum_{j=0}^{\text{col}-1} \left(X(i, j) \cdot p(i, j) / 255 \right), \\
 Y_s &= \frac{1}{\text{remain_p}} \sum_{i=0}^{\text{row}-1} \sum_{j=0}^{\text{col}-1} \left(Y(i, j) \cdot p(i, j) / 255 \right), \\
 Z_s &= \frac{1}{\text{remain_p}} \sum_{i=0}^{\text{row}-1} \sum_{j=0}^{\text{col}-1} \left(Z(i, j) \cdot p(i, j) / 255 \right).
 \end{aligned}
 \quad
 \begin{aligned}
 x_s &= \frac{X_s}{X_s + Y_s + Z_s}, \\
 y_s &= \frac{Y_s}{X_s + Y_s + Z_s}.
 \end{aligned}$$

- **Step 8:** The output of the previous step corresponds to the image illuminant chromaticity. This step converts the illuminant chromaticity (x_s, y_s) into a color temperature (T) as described in [126] and making use of Table 6.3:

- (1) Compute the illuminant chromaticity coordinates (u_s, v_s) in the CIE-uv diagram from (x_s, y_s)

$$\begin{aligned}
 u_s &= \frac{4x_s}{-2x_s + 12y_s + 3}, \\
 v_s &= \frac{6y_s}{-2x_s + 12y_s + 3}.
 \end{aligned}$$

- (2) Find two adjacent iso-temperature lines⁷ from (u_s, v_s) and obtain the distance between those lines: if (u_s, v_s) is located between the i^{th} and $(i+1)^{\text{th}}$ iso-temperature lines then $d_i / d_{i+1} < 0$.

$$d_i = \frac{(v_s - v_i) - t_i(u_s - u_i)}{(1 + t_i^2)^{1/2}}$$

where (u_i, v_i) and t_i are the chromaticity coordinates and the slope for representing the i^{th} iso-temperature line (see Table 6.3); d_i is the distance between (u_s, v_s) and the i^{th} iso-temperature line.

- (3) Compute the correlated color temperature using the following equation:

$$T_c = \left[\frac{1}{T_i} + \frac{d_i}{d_i - d_{i+1}} \left(\frac{1}{T_{i+1}} - \frac{1}{T_i} \right) \right]^{-1}$$

where T_i is the color temperature for the crosspoint between the i^{th} iso-temperature line and the daylight locus⁸ ($T=6,500$ K). The color temperatures under 1,667 K and over 25,000 K are set to 1,667 K and 25,000 K, respectively, due to algorithmic limitations.

The method presented in this section was adopted by MPEG-7 for the definition of image color temperature.

⁷ Iso-temperature lines are standardized lines that pass through the blackbody locus and are used to correlate a given color distant from the blackbody locus with a color of the blackbody locus.

⁸ This is the point in the blackbody locus curve, which corresponds to the day light color ($T=6,500$ K).

Table 6.3 – Iso-temperature lines according to Wiseck & Stiles [126].

<i>I</i>	Reciprocal Megakelvin	Temperature <i>T</i> (K)	<i>u_i</i>	<i>v_i</i>	<i>t_i</i>
1	0	Infinity	0.18006	0.26352	-0.24341
2	10	100,000	0.18066	0.26589	-0.25479
3	20	50,000	0.18133	0.26846	-0.26876
4	30	33,333	0.18208	0.27119	-0.28539
5	40	25,000	0.18293	0.27407	-0.30470
6	50	20,000	0.18388	0.27709	-0.32675
7	60	16,667	0.18494	0.28021	-0.35156
8	70	14,286	0.18611	0.28342	-0.37915
9	80	12,500	0.18740	0.28668	-0.40955
10	90	11,111	0.18880	0.28997	-0.44278
11	100	10,000	0.19032	0.29326	-0.47888
12	125	8,000	0.19462	0.30141	-0.58204
13	150	6,667	0.19962	0.30921	-0.70471
14	175	5,714	0.20525	0.31647	-0.84901
15	200	5,000	0.21142	0.32312	-1.0182
16	225	4,444	0.21807	0.32909	-1.2168
17	250	4,000	0.22511	0.33439	-1.4512
18	275	3,636	0.23247	0.33904	-1.7298
19	300	3,333	0.24010	0.34308	-2.0637
20	325	3,077	0.24702	0.34655	-2.4681
21	350	2,857	0.25591	0.34951	-2.9641
22	375	2,677	0.26400	0.35200	-3.5814
23	400	2,500	0.27218	0.35407	-4.3633
24	425	2,353	0.28039	0.35577	-5.3762
25	450	2,222	0.28863	0.35714	-6.7262
26	475	2,105	0.29685	0.35823	-8.5955
27	500	2,000	0.30505	0.35907	-11.324
28	525	1,905	0.31320	0.35968	-15.628
29	550	1,818	0.32129	0.36011	-23.325
30	575	1,739	0.32931	0.36038	-40.770
31	600	1,667	0.33724	0.36051	-116.45

6.3.1.4 Color Temperature Transformation

The most direct application of the MPEG-7 color temperature descriptor is for the adaptation of images according to the user preferences in terms of color temperature. Chromatic adaptation can be considered as a dynamic mechanism of the human visual system to optimize the visual response to a particular viewing condition. Dark and light adaptations are the changes in visual sensitivity when the level of illumination is decreased or increased, respectively. Chromatic adaptation is the ability of the human visual system to discount the color of the illumination and to approximately preserve the appearance of an object. Chromatic adaptation can be observed by examining a white object under different types of

illumination, such as daylight and incandescent lamp. Daylight is “bluer”: it contains far more short-wavelength energy than incandescent. However, the white object retains its white appearance under both light sources, as long as the viewer is adapted to the light source.

To faithfully reproduce the appearance of image colors (or to adapt to certain user preferences), a color temperature processing system needs to apply a transform to the image that converts the input colors captured under the input illumination to the corresponding output colors under the relevant output illumination.

This can be achieved using a chromatic adaptation transform in the XYZ domain. However, in the implemented UMA application, the user provides a color temperature to express his/her color temperature preference, which means that both the image color temperature and the user preferred color temperature must be converted into the appropriate XYZ values (chromaticity values). In order to do that, the method presented in [126] (already used to calculate the color temperature from the color coordinates) was used.

Therefore the color temperature adaptation is reduced to a simple chromatic adaptation transform. Basically, applying a chromatic adaptation transform to the pixels of an image under one adapting light source ($X_i'Y_i'Z_i'$, the image illuminant chromaticity) will result in the corresponding image pixels under another light source ($X_u''Y_u''Z_u''$, the user preferred illuminant chromaticity).

There are several chromatic adaptation transforms described in the literature based on the following model applied to each pixel:

$$\begin{bmatrix} X''(i, j) \\ Y''(i, j) \\ Z''(i, j) \end{bmatrix} = [M_{BFD}]^{-1} \cdot \begin{bmatrix} R_u''/R_i' & 0 & 0 \\ 0 & G_u''/G_i' & 0 \\ 0 & 0 & B_u''/B_i' \end{bmatrix} \cdot [M_{BFD}] \cdot \begin{bmatrix} X'(i, j) \\ Y'(i, j) \\ Z'(i, j) \end{bmatrix}$$

$$\begin{bmatrix} X''(i, j) \\ Y''(i, j) \\ Z''(i, j) \end{bmatrix} = [M_A] \cdot \begin{bmatrix} X'(i, j) \\ Y'(i, j) \\ Z'(i, j) \end{bmatrix}$$

$$M_A = [M_{BFD}]^{-1} \cdot \begin{bmatrix} R_u''/R_i' & 0 & 0 \\ 0 & G_u''/G_i' & 0 \\ 0 & 0 & B_u''/B_i' \end{bmatrix} \cdot [M_{BFD}]$$

where the matrix \mathbf{M}_{BFD} corresponds to the Bradford chromatic transform [125]; however, any other chromatic transform can be used. The quantities $R_i'G_i'B_i'$ and $R_u''G_u''B_u''$ are computed, respectively, from the image illuminant chromaticity, $X_i'Y_i'Z_i'$, and from the preferred illuminant, $X_u''Y_u''Z_u''$, by multiplying the corresponding vectors by \mathbf{M}_{BFD} :

$$\begin{bmatrix} R'_i \\ G'_i \\ B'_i \end{bmatrix} = \begin{bmatrix} M_{BFD} \end{bmatrix} \begin{bmatrix} X'_i/Y'_i \\ Y'_i/Y'_i \\ Z'_i/Y'_i \end{bmatrix} \quad \begin{bmatrix} R''_u \\ G''_u \\ B''_u \end{bmatrix} = \begin{bmatrix} M_{BFD} \end{bmatrix} \begin{bmatrix} X''_u/Y''_u \\ Y''_u/Y''_u \\ Z''_u/Y''_u \end{bmatrix}$$

The Bradford chromatic transform may be used to convert an image from one illuminant chromaticity to another. This transform is given by:

$$\begin{bmatrix} M_{BFD} \end{bmatrix} = \begin{bmatrix} 0.8951 & 0.2664 & -0.1614 \\ -0.7502 & 1.7135 & 0.0367 \\ 0.0389 & -0.0685 & 1.0296 \end{bmatrix} \quad \begin{bmatrix} M_{BFD} \end{bmatrix}^{-1} = \begin{bmatrix} 0.9870 & -0.1471 & 0.1600 \\ 0.4323 & 0.5184 & 0.0493 \\ -0.0085 & 0.0400 & 0.9685 \end{bmatrix}$$

Figure 6.15 illustrates the overall color temperature adaptation process. After gamma-correcting the image, the pretended adaptation matrix, M_A , (based on the image illuminant chromaticity and the user preferred illuminant chromaticity) is computed and the image finally adapted.

The color temperature adaptation algorithm here described was implemented in the UMA Platform.

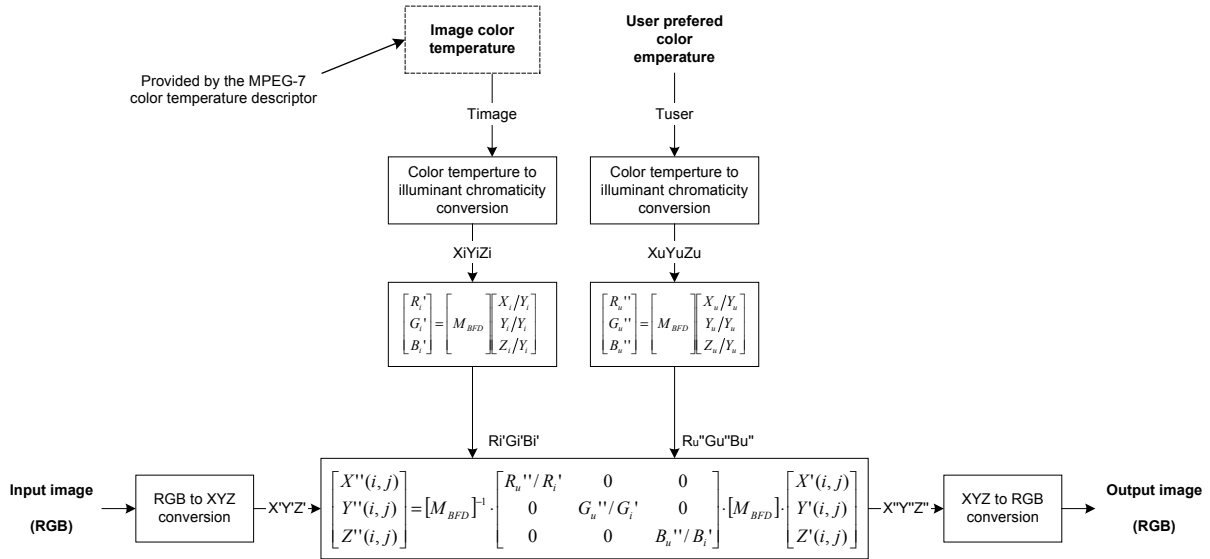


Figure 6.15 – Steps to transform the image color temperature.

6.3.1.5 Color Temperature Adaptation Results

The tests for the color temperature customization used the MPEG-7 color temperature extraction method presented in the previous section. Using the UMA browser to access the content, the user could select the pretended color temperature by example or value; it is the task of the UMA Platform to transform the available image to the color temperature corresponding to the expressed preference. Qualitative results corresponding to this process can be observed in Figure 6.16, Figure 6.17, Figure 6.18 and Figure 6.19.

This is the adaptation process from those implemented that requires more memory: besides the buffer for the source image, a buffer of the same size for the output image is also required. Moreover, the CPU costs

for this customization task are also quite high as shown in Table 6.4 (the processing time is in the order of thousands of milliseconds per image).



Figure 6.16 – Color temperature adaptation: a) 3,497 K; b) 5,033 K (source); c) 6,990 K.

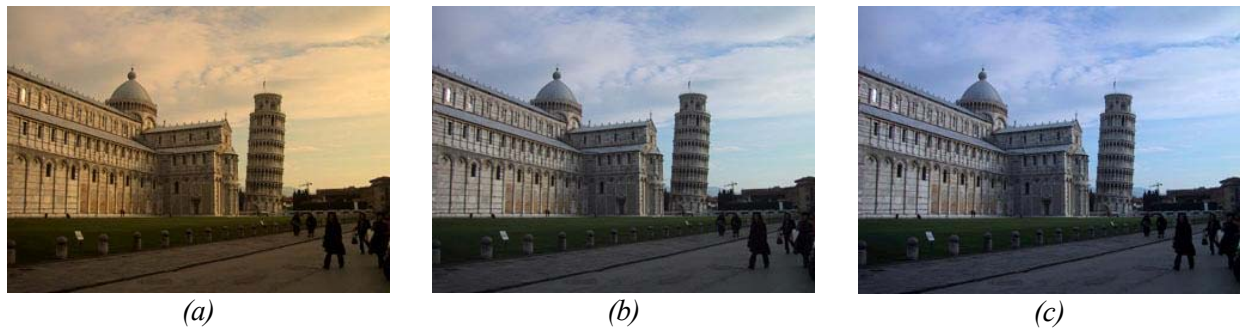


Figure 6.17 – Color temperature adaptation: a) 4,135 K; b) 7,924 K (source); c) 11,711 K.

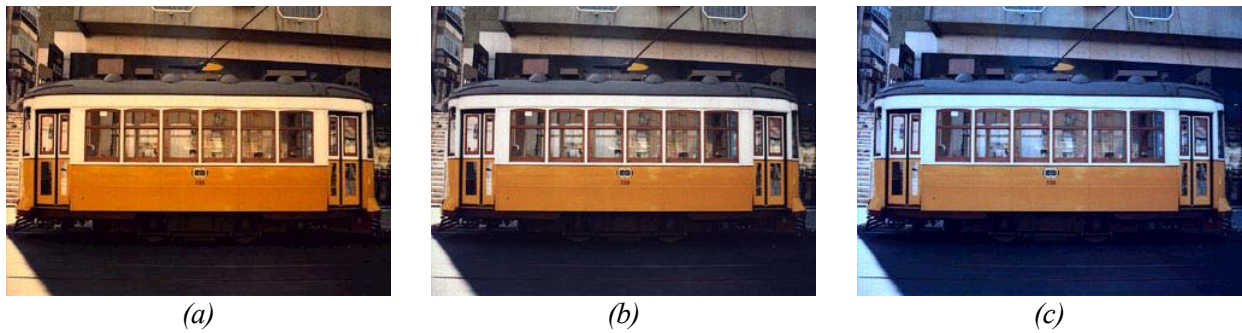


Figure 6.18 – Color temperature adaptation: a) 2,627 K; b) 3,718 K (source); c) 8,640 K.



Figure 6.19 – Color temperature adaptation: a) 2,697 K; b) 4,452 K (source); c) 6,292 K.

Table 6.4 – Color temperature image customization measures.

Image characteristics	Average customization time	Uncompressed image size
1600x1200 pixels; 24 bits	41,410 ms	5,625 kBytes
1024x768 pixels; 16 bits	14,851 ms	1,536 kBytes
640x480 pixels; 8 bits	5,808 ms	300 kBytes
320x240 pixels; 8 bits	1,552 ms	75 kBytes

It is trivial to note from the color temperature adaptation algorithm that the target color temperature does not affect the amount of transform calculations. Therefore, since the source image color temperature and the target color temperature are irrelevant for the amount of calculations, the measures taken used an arbitrary target color temperature.

The results presented in Figure 6.16, Figure 6.17, Figure 6.18 and Figure 6.19 provide in fact different perceived illuminations. The user preferred color temperature was used to transform the source images in order to offer the same color temperature sensation to the user.

The algorithm was used only with images, however it may also be applied to video content. The color temperature transformation in video will cause a greater delay because it is an operation must more demanding than the video adaptations that were implemented.

6.4 Video Customization Module

The video customization module implements a set of adaptation methods to process video content, as presented in Table 6.5. The video adaptation algorithms have been developed using a MPEG-1/-2 encoder and decoder: both were modified and implemented in separated DLLs to be accessed by any program. Both encoder/decoder receive and send all the data to memory, and were changed to work in a “step-by-step” way: Each adaptation operation must be configured by an appropriate method (`setParameter()`) which sets the parameters for the related adaptation. After all the adaptation operations have been set, a final method (`customize()`) is called to execute all the configured adaptation operations.

The video adaptation algorithms use the MPEG-1/-2 decoder to access the frames in the uncompressed domain and then adapt them. After the frames have been adapted, they are passed to the MPEG-1/-2 encoder to be again encoded.

The video content customization algorithms implemented in the uncompressed domain concern color component reduction, spatial resolution reduction, and bit rate reduction (through the encoding parameters, e.g., the quantization step). The video customization module is prepared to perform several types of video customization methods; however, only some of them have been implemented due to time limitations (see Table 6.5).

The customization method implemented to reduce the size of the video bitstream sets the new bit rate in the encoder configuration. The rest of the adaptation operations were implemented using spatial domain transformations:

- **Bits per pixel reduction:** this operation is achieved by simply re-quantifying each color component. The number of bits per pixel is assumed to be equal for all components.
- **Color space transformation:** the supported color spaces are gray and color; for color displays, the video suffers no change, but for gray displays the color information is suppressed.
- **Spatial resolution reduction:** in order to reduce the spatial resolution, a Gaussian filter is applied to the source image in order to obtain the target image; this method corresponds to a low-pass filtering, which was rather minimal for the case implemented since a window size of 3 pixels was used.
- **Region of interest cropping:** the region of interest functionality corresponds to a simple cropping of the source image around the region of interest.

Table 6.5 – Video adaptation operations supported by the video customization module.

Customization operation	Description	Status
Color depth reduction	Reduces the number of bits per pixel.	Implemented
Color space transformation	Reduces the color space (conversion from color space to gray scale)	Implemented
Spatial resolution reduction	Reduces the video spatial resolution.	Implemented
Temporal resolution reduction	Reduces the video frame rate.	Not implemented
Set the target bit rate	Reduces the bit rate consumed by the video (increase compression)	Implemented
Spatial resolution reduction using a Region of interest	Reduces the spatial resolution having in account a region of interest.	Implemented
Format conversion	Converts the video to another format.	Not implemented.
Replace by variation	Selects another variation.	Not implemented.
Operation-8 to 15	Free.	Not implemented.

As for the image customization module, the adaptation method parameters are activated and set with the `setParameter()` method; after every adaptation action has been set, the `customize()` method is used to execute the content customization.

Adaptation methods in the compressed domain would be faster and consume much less memory; due to time limitations, no investment could be made on implementing compressed domain adaptation algorithms in the context of this thesis.

6.5 Summary

This Chapter shows that the Content Action Decision / Content Customization modules enable different adaptation algorithms to be implemented maintaining always the same underlying system core. The customization modules act like plug-ins to the UMA Platform that the Content Action Decision module can use in its decisions, while the Content Customization module runs the content processing methods available on each customization module.

In the image customization module, algorithms to adapt the images based on the terminal display characteristics were implemented, e.g. spatial resolution. These algorithms processed the content in the uncompressed domain. In the image customization module, an algorithm to adapt the image illuminant color temperature was implemented. This adaptation algorithm provided the system a user preference

based customization using an MPEG-7 low-level descriptor which will soon be included in the standard; this is today a rather rare capability since most of the MPEG-7 enabled applications still only use text based descriptors. Besides the adaptation algorithm, the color temperature extraction algorithm was also implemented in the MPEG-7 description tool.

In the video customization module, algorithms to adapt MPEG-1 video content to the terminal display characteristics were implemented. These algorithms processed the content in the uncompressed domain. Even with these rather inefficient adaptation methods, it was possible to have the system processing and streaming content simultaneously.

Chapter 7

User Interfaces and Test Scenarios

In this thesis three applications have been implemented: the UMA browser, the UMA Platform and the MPEG-7 description tool. In the first part of this Chapter, the graphical interfaces of the three applications and the testing environment will be presented. In the second part of the Chapter, the adopted test scenarios and the corresponding adaptation results will be presented and discussed.

7.1 User Interfaces

This section presents the user interfaces for the implemented applications.

7.1.1 UMA Browser

The UMA browser is a Web browser application, which allows the user to create and manage his user environment descriptions (MPEG-21 DIUED) through appropriate menus. In Figure 7.1 the graphical interface of the UMA browser is presented; the figure shows the menu, which allows creating and changing DIUED and is already showing some content that has been retrieved.

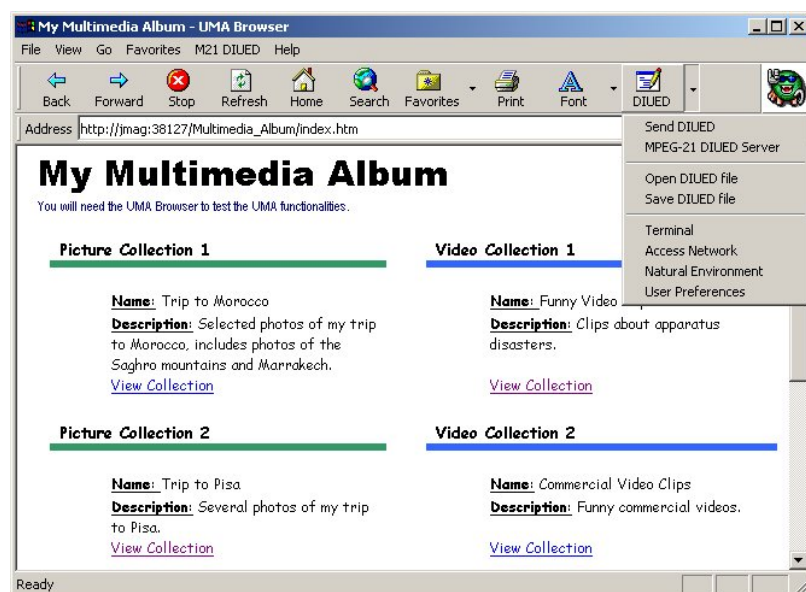


Figure 7.1 – UMA browser interface after retrieving the “My Multimedia Album” content.

The UMA browser is similar to the Microsoft Internet Explorer: in the address toolbar the user may insert the URL to navigate. The toolbar buttons’ functions are:

- **Back:** jumping to the previous Web page of the list of accessed Web pages;
- **Forward:** jumping to the next Web page of the list of accessed Web pages;
- **Stop:** stopping the retrieval within the current Web page;
- **Refresh:** reloading the current Web page;
- **Home:** the UMA browser navigates to its home URL;
- **Search:** accessing a Web search application;
- **Favourites:** giving access to the users bookmarks;
- **Print:** printing the current Web page;
- **Font:** enabling the user to change the default font size;
- **DIUED:** giving access to a menu of options to edit/create user environment descriptions.

The DIUED options available in the UMA browser are the following:

- **Send DIUED:** allows sending the user environment description to a DIUED server via an HTTP POST command.
- **MPEG-21 DIUED server:** allows modifying the address of the server where the DIUED will be sent via an HTTP POST command.
- **Open DIUED:** allows opening a DIUED file from the disk, e.g. to change some descriptors.
- **Save DIUED:** allows saving the current user environment characteristics into a DIUED file.
- **Terminal:** allows editing the DIUED terminal capabilities; the currently available characteristics are the display hardware characteristics (the audio hardware characteristics and the media decoding software are currently ignored by the UMA Platform).
- **Access Network:** allows editing the DIUED access network capabilities; the available characteristics are those presented in Chapter 4.
- **Natural Environment:** allows to edit the DIUED natural environment features; the natural environment descriptors available are location, speed, temperature and altitude.
- **User Preferences:** allows editing the DIUED user preferences; the only user preference currently available is the preference related to image color temperature; the user may specify his/her preference by example or with an exact value.

The UMA browser application has the following functionality: the user may create and change his/her DIUED as he/she wishes through the several menu options. When the user selects the Send DIUED option, the UMA browser sends the DIUED to the MPEG-21 DIUED server with an HTTP POST

command. The DIUED is sent to the selected MPEG-21 DIUED server, which keeps a database of DIUEDs indexed using the IP address. Figure 7.2 illustrates this procedure.

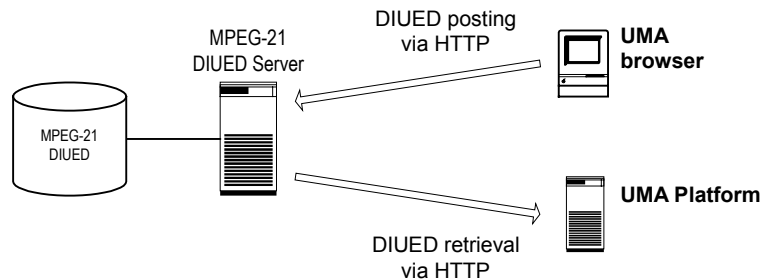


Figure 7.2 – UMA browser posting a DIUED.

This method has the disadvantage that it is not possible to use different DIUEDs for the same device simultaneously (e.g. to have different DIUEDs for different UMA browsers in the same desktop PC), this means that it is not possible to have several browsers active for the same terminal using different DIUEDs.

7.1.2 UMA Platform

The UMA Platform architecture and processing have been presented in Chapters 5 and 6. The interface to control its operation and monitor its status is shown in Figure 7.3.

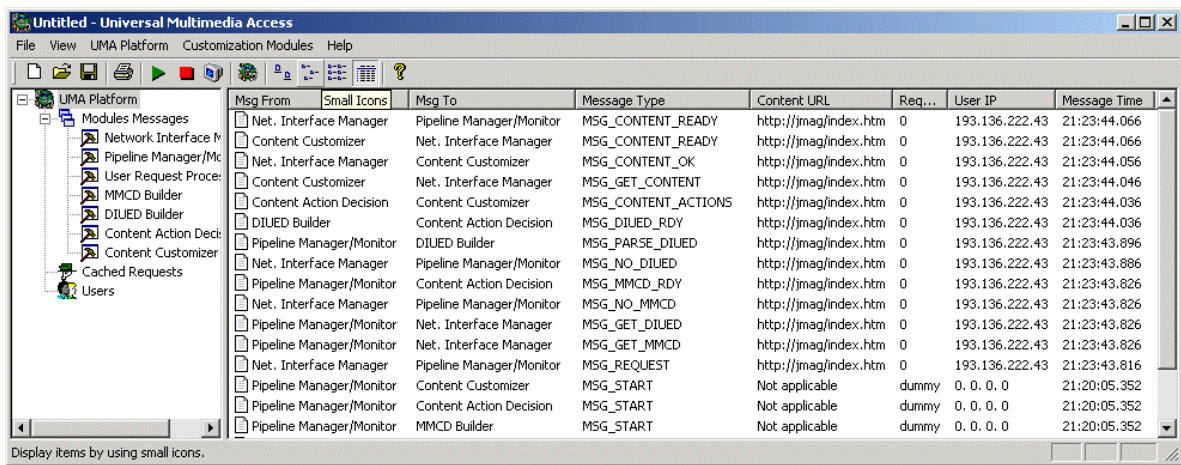










Figure 7.3 – UMA Platform Graphical User Interface.

The two windows in the UMA Platform GUI (tree on the left and list on the right) provide a good overview on the status of the UMA Platform. The tree view provides a list of items regarding the UMA Platform: the internal modules; the processed requests; and the users. By selecting each item on the tree, all information related to that item is displayed on the right side. The information available for each item is related to the messages that are exchanged between the UMA Platform modules (presented in Chapter 5). The items available on the tree allow to:

- View the status of each module;
- View the message flow among all modules;
- View the message flow corresponding to a single module;

- Access the message log⁹ file containing the messages exchanged among the UMA Platform modules while processing the received user requests;
- View a list of the requests that were received from the UMA browser (or any other HTTP client);
- View the DIUED descriptions of the users that are using the UMA Platform.

The commands available through the application toolbar and the menus are:

- The  button creates a new log file for recording the messages exchanged among the internal modules, which can be used to analyze the time consumed by each operation in the UMA Engine.
- The  opens log files in the notepad so that the user may see the content of the file.
- The  button saves the current message log into a file.
- The  prints the current message logs.
- The  button starts the UMA Engine.
- The  button stops the UMA Engine without deleting the caching tables.
- The  button performs a reset to the UMA Engine (the information in the caching tables is lost).
- The  button allows configuring the UMA Platform configuration (as described next).

The save button is used to save the log of the UMA Platform internal messages. This log file contains the results of the behavior of the system, which is used to extract several measures as will be observed in section 7.4.

To control the operation of the UMA Platform, it is possible to Start/Stop or restart the UMA Engine and to edit the UMA Platform configuration. Using the option `UMA Platform Configuration` in the `UMA Platform` menu (or in the corresponding toolbar button), the administrator has access to the dialog box of Figure 7.4, which allows him/her to change the settings of the UMA Platform.

The UMA Platform configuration dialog box offers the following options:

- `UMA Engine Operation Mode`:
 - **Bypass:** the UMA Platform will act as a ‘tunnel’ for the content; the content is transmitted to the user with no adaptation action;
 - **Normal:** the content customization process becomes active, through the decision and adaptation algorithms;

⁹ The message log is a record of every messages exchanged among the UMA Platform internal modules.

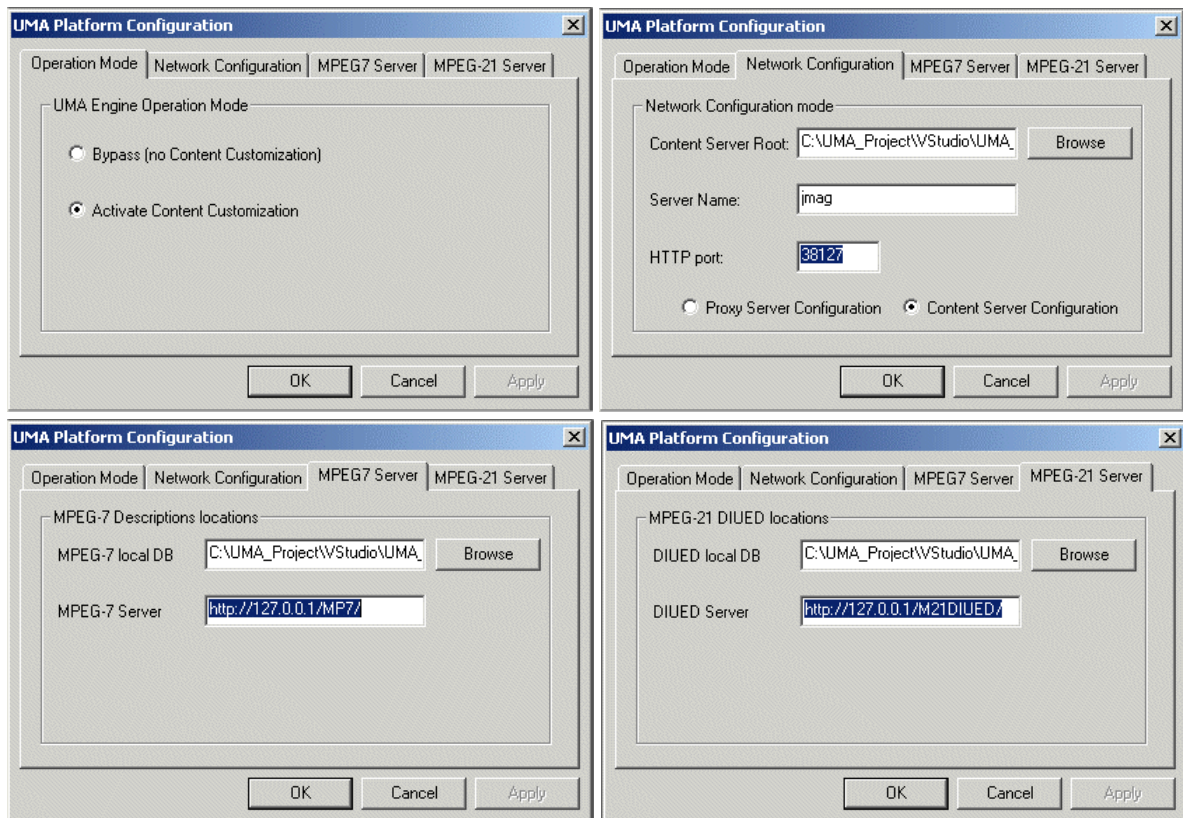


Figure 7.4 – UMA Platform configuration options.

- Network Configuration Mode:
 - **Content server root:** specifies the local directory from where the content is retrieved when using the content server configuration.
 - **Server name:** specifies the server name.
 - **HTTP Port:** specifies the HTTP port where the UMA Platform will be listening for HTTP commands.
 - **Content server configuration:** allows to configure the UMA Platform as a content server; the UMA Platform will be listening to user requests in the specified port and the content will be fetched from the directory specified.
 - **Proxy server configuration:** allows to configure the UMA Platform as a proxy server; the UMA Platform will be listening to user requests in the specified HTTP port.
- MPEG-7 Descriptions Location:
 - **MPEG-7 description server:** specifies the server URL where the UMA Platform will retrieve the MPEG-7 descriptions.
 - **MPEG-7 Local DB:** specifies the local directory where the UMA Platform will retrieve the MPEG-7 descriptions.

– MPEG-21 DIUED Location:

- **DIUED server:** specifies the server URL where the UMA Platform will retrieve the DIUED descriptions.
- **DIUED Local DB:** specifies the local directory where the UMA Platform will retrieve the DIUED descriptions.

The Customization Modules menu allows selecting the customization modules that are active; for example, video customizations may be disabled.

7.1.3 MPEG-7 Description Tool

The MPEG-7 description tool is used to analyze multimedia content extracting the features that are stored in the MPEG-7 description format. The MPEG-7 descriptions can be stored in the local disk or in a remote MPEG-7 description server. The MPEG-7 description tool uses an HTTP POST command to send the content descriptions to the server, as illustrated in Figure 7.5.

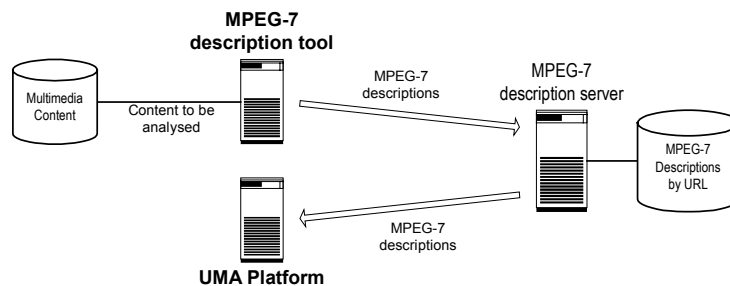


Figure 7.5 – MPEG-7 description tool posting a description to the MPEG-7 description server.

This tool allows to instantiate several MPEG-7 descriptors: the MediaInformationDS is always generated, while the color temperature is extracted only when the corresponding check box is selected. The dialog box for this application is presented in Figure 7.6.

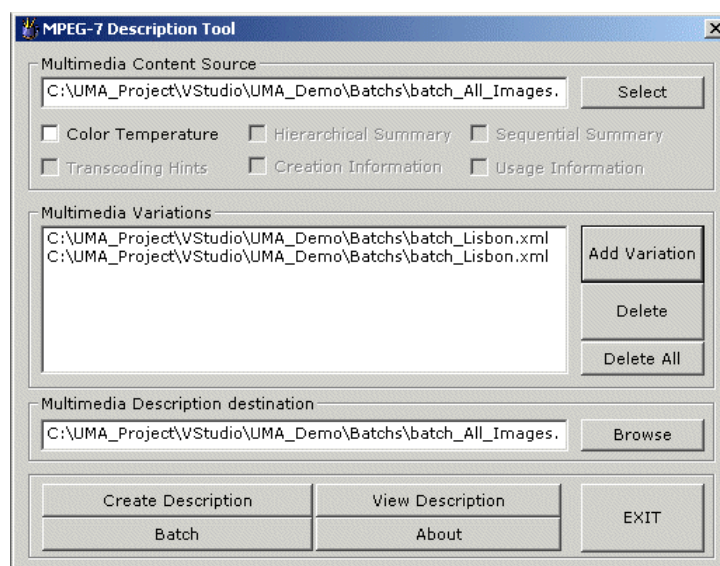


Figure 7.6 – MPEG-7 description tool dialog box.

The options available in the MPEG-7 description tool are:

- Multimedia Content Source:
 - **Select:** to select the content to be analyzed and described.
 - **Color Temperature:** activates the color temperature extraction.
- Multimedia Variations:
 - **Add Variation:** to add a file as a content variation to the list of variations for a certain piece of content.
 - **Delete:** to delete the selected variation from the list of variations for a certain piece of content.
 - **Delete All:** to delete all files from the list of variations.
- Multimedia Description Destination:
 - **Browse:** to define the file name and location of the description to be created.
- **Batch:** to select a file containing a list of files which content must be described.
- **Create description:** to start the content analysis to create the description.
- **View description:** to open a window to display the MPEG-7 description in textual format.
- **EXIT:** to exit the application.

The MPEG-7 description tool should implement all MPEG-7 descriptors and description schemes that are useful for content customization. The color temperature is currently the only descriptor that can be configured to be extracted or not for the descriptions created.

7.2 UMA Test Bed

Through the user interfaces (presented in the previous sections) of the implemented applications, the test bed and tests scenarios could be configured and created. Two UMA test beds corresponding to the two network configurations implemented were used to evaluate the UMA System: content server and proxy server.

7.2.1 Content Server Test Bed

The UMA System test bed in content server configuration is illustrated in Figure 7.7, which presents how the network elements are interconnected. The content server configuration settings were setup as follows through the appropriate GUIs:

- The UMA Platform is operating as a proxy server in port 38127.
- The UMA browser has programmed the UMA Platform as its proxy.

- When the DIUED is changed in the UMA browser, the post of the DIUED goes to the UMA Platform that is working as an MPEG-21 DIUED server.
- In this test bed, the MPEG-7 description tool reads the content from the local disk and writes the MPEG-7 descriptions directly in the MPEG-7 database of the Web server.
- The content files and the description files are under the Web server root directory.
- The descriptions files are in the directory configured in the UMA Platform (see section 7.1.2).

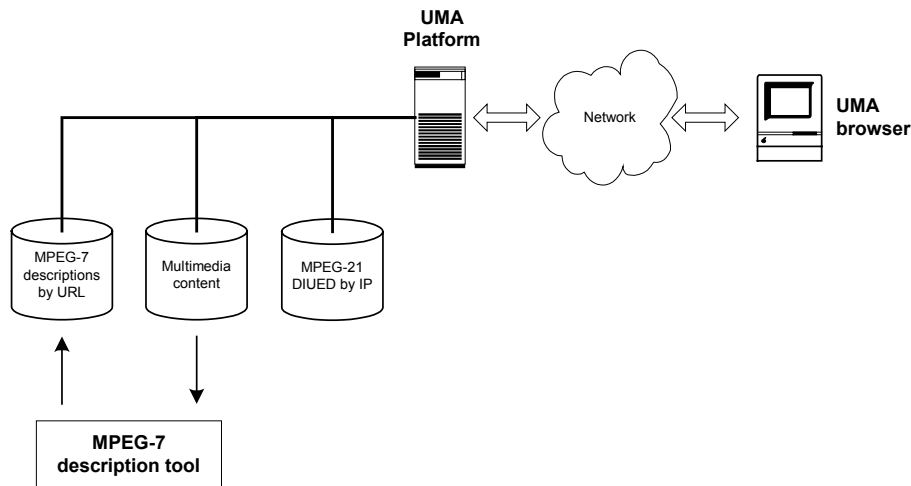


Figure 7.7 – UMA System test bed in content server configuration.

7.2.2 Proxy Server Test Bed

The UMA System test bed in proxy server configuration is illustrated in Figure 7.8 which presents how the network elements are interconnected. The configuration settings are as follows:

- The UMA Platform is operating as a proxy server in port 38127.
- The UMA browser has programmed the UMA Platform as its proxy.
- The UMA browser has programmed the UMA Platform as its MPEG-21 DIUED Server.
- In this test –bed, the MPEG-7 description tool reads the content from the local disk and writes the MPEG-7 descriptions directly in the MPEG-7 database of the Web server.
- The Web server is an Apache HTTP server.
- The databases are files under the Web server root directory.

The UMA Platform uses the MPEG-7 descriptions by URL and the MPEG-21 DIUEDs by IP address. The URL to access the MPEG-7 descriptions is composed by the URL of the server (e.g. “http://m7server/M7MMCD/”) combined with the URL of the content which description is looked for (e.g. “GET http://m7server/M7MMCD/**www.yahoo.com/images/image1.jpgx.mp7**”).

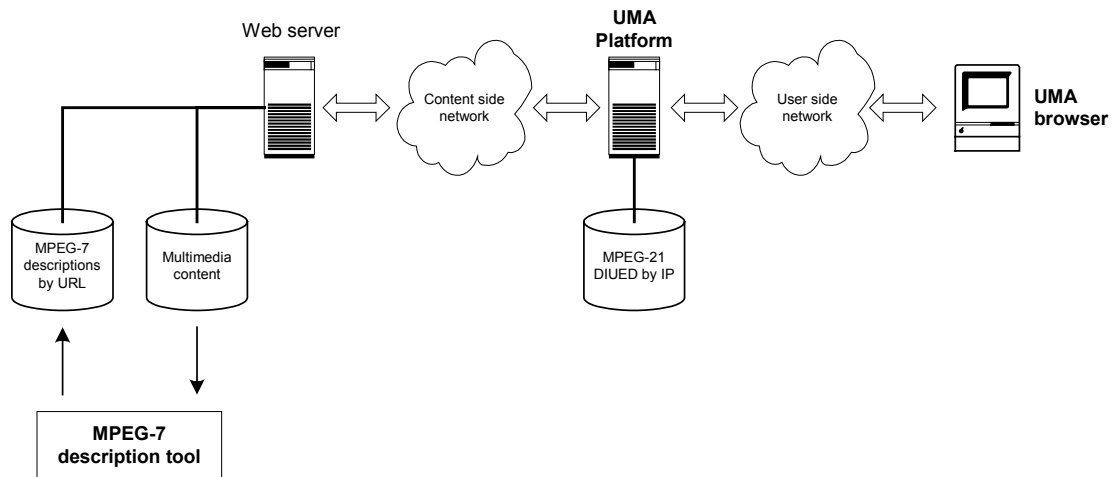


Figure 7.8 – UMA System test-bed in proxy server configuration.

This configuration is one of the most interesting since much of today's available content in Internet can only be accessed from mobile devices using this network configuration. Therefore, this configuration received most of the efforts. Since this configuration raised some problems with video streaming, the solution adopted was to download the video, and only after, perform the adaptation and transmission to the user simultaneously. In some cases the system would have to be receiving, adapting and streaming at the same time.

7.3 UMA Test Scenarios

In order to test both the UMA System configurations, a Web site containing heavy multimedia content was designed. It is called “My Multimedia Album” and it is basically a collection of pictures and videos that can be accessed using the UMA browser or any other HTTP client.

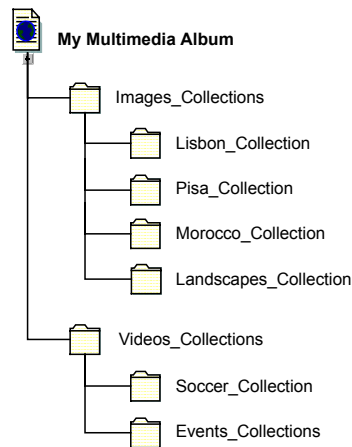


Figure 7.9 – “My Multimedia Album” content structure.

The content at the content server is structured as illustrated in Figure 7.9:

- The multimedia album is composed by four image collections and two video collections.
- Each collection has four images or videos.

- Each image exists in four different variations (see section 7.3.1).
- Each video exists in three different variations (see section 7.3.1).

The content and user environment descriptors used to access and adapt the content are presented in the following sections. In Figure 7.1 the “My Multimedia Album” entrance page is shown in the UMA browser interface. Figure 7.10 illustrates the pages of a picture collection and of a video collection. After sending the DIUED defined or selected (corresponding to a certain type of user environment) to the UMA Platform, the “My Multimedia Album” can be accessed by the UMA browser simulating the environment in question. Beside simulated user environments also real equipment will be used to access the “My Multimedia Album”.

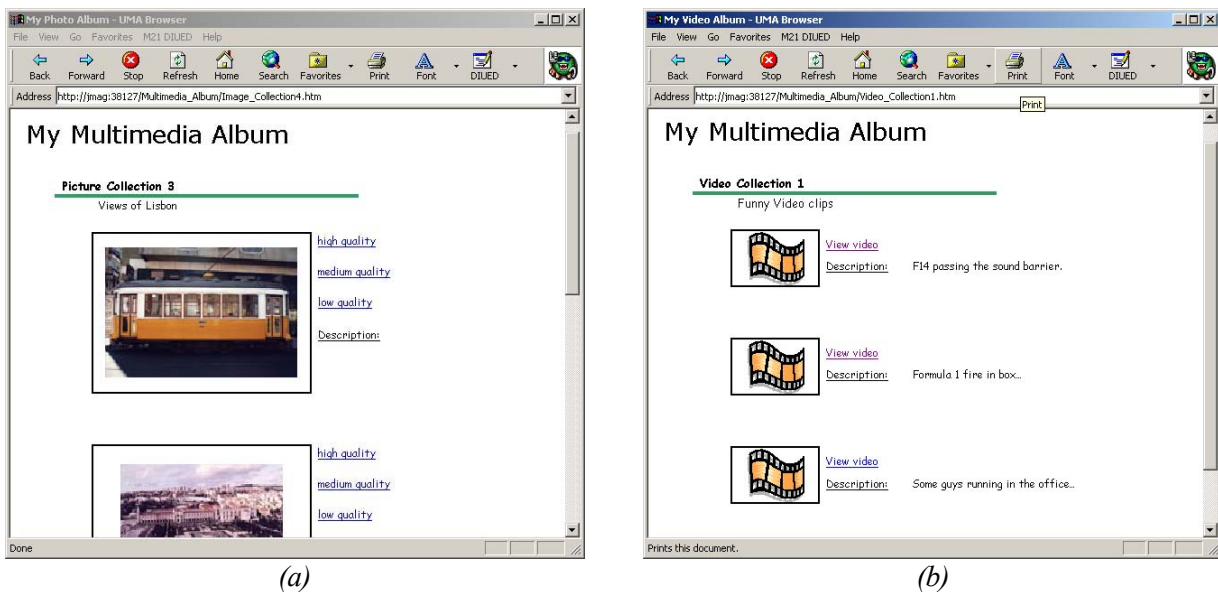


Figure 7.10 – “My Multimedia Album” content: (a) Lisbon pictures collection; (b) video collection.

7.3.1 Multimedia Content and MPEG-7 Descriptions

Several images and videos with the corresponding MPEG-7 descriptions were used for testing the overall UMA System. As shown in Table 7.1, four categories of images were selected in order to evaluate the performance of the system under different conditions. The goal was to evaluate the influence of large content (high spatial resolution and data size) in the content customization process under different adaptation conditions.

For video, three categories were selected, ranging from CIF resolution coded at 256 kbit/s to QCIF resolution coded at 64 kbit/s (see Table 7.1). Once again the existence of variations enabled to test the content customization process under different adaptation conditions.

Table 7.1 – Characteristics of the content tested.

Content characteristics	Image Category A	Image Category B	Image Category C	Image Category D	Video Category A	Video Category B	Video Category C
Width	1600	1024	640	320	352	240	176
Height	1200	768	480	240	288	192	144
Bits per pixel	24 bits	16 bits	8 bits	8 bits	24 bits	24 bits	24 bits
Format	JPEG	JPEG	JPEG	JPEG	MPEG-1	MPEG-1	MPEG-1

Bit rate	-	-	-	-	256 kbit/s	128 kbit/s	64 kbit/s
Duration	-	-	-	-	≅ 10 sec	≅ 10 sec	≅ 10 sec
Frame rate	-	-	-	-	25 fps	15 fps	10 fps
Number of pieces	16	16	16	16	6	6	6

In [5], a statistical analysis has been made to characterize the multimedia content populating the Internet. The used images and video resolution have no professional characteristics (e.g. Kodak Photo CD 4096 x 6144 pixels, or video resolutions above CIF) and the video durations could easily be the result of a summarization algorithm. This study was considered when selecting the content characteristics adopted for the tests. Besides typical Web content, content adequate for future mobile multimedia applications was also considered (e.g. QCIF videos, and low resolution images).

7.3.2 DIUED Scenarios

The diversity of content characteristics considered, required an equivalent diversity in terms of user environment in order to make a good testing of the UMA System. Consequently, four user environment scenarios were selected to test the content customization process. The display size and the network bandwidth features were selected as the major varying user environment characteristics so that QCIF/CIF content could be matched against different displays and bandwidths.

Table 7.2 presents the DIUED characteristics, which were used to simulate WAP terminals, Pocket PCs and Hand Held PCs.

Table 7.2 – User environment tested scenarios.

User environment characteristics	Pocket PC 1	Pocket PC 2	WAP terminal 1	WAP terminal 2
Terminal	Compaq IPAQ	Simulated	Simulated	Siemens ME45
Width	240	240	176	66
Height	320	320	144	101
Color Domain	Color	Gray	Color	Gray
Bits per pixel	16	8	8	1
Bit rate	64 kbit/s	64 kbit/s	64 kbit/s	64 kbit/s

For the Pocket PC 1 and WAP terminal 2 scenarios, real terminals were used to access the content. These real terminals could not be used to test video customizations because they do not send any information regarding the available network bandwidth. Thus, for video tests, the UMA browser simulating the corresponding user environment characteristics was always used. The Pocket PC2 and WAP terminal 1 scenarios were tested using the UMA browser simulating the required user environment features.

Even though the terminal would connect to the UMA Platform using different connections, the UMA Platform considers always the bandwidth information present in the DIUED.

7.4 UMA Results

Great parts of the results to be presented are qualitative: in many cases the content is just converted in such a way that the terminal may consume it, in other cases the content is tailored to the user environment in order to improve the user experience. Nevertheless, some measures have been made, such as the customization time measured as the time required to adapt the content (to compare different customization conditions) and the customized content size (to compare the gain relatively to the original source in terms of required bandwidth). Even though both network configurations were used (content and proxy servers), the measures presented only refer to the customization process and the network connection between the UMA Platform and the terminal and thus no difference exists between the two network configurations.

7.4.1 Pocket PC 1 Scenario

This test scenario was evaluated using a Compaq IPAQ Pocket PC terminal. The terminal was connected to the UMA Platform through a UMTS link at 64 kbit/s¹⁰. Since the Pocket PC terminal has no UMTS capabilities, it was connected to a UMTS terminal, which performed the connection to the UMTS network, as illustrated by Figure 7.11. The UMA Platform was installed in the UMTS IP network as a content server.



Figure 7.11 – Compaq IPAQ terminal connected to a UMTS terminal.

This Compaq IPAQ Pocket PC terminal is capable of receiving HTML content and so the “My Multimedia Album” content could be accessed. Figure 7.13 presents the “My Multimedia Album” site being accessed by the PocketPC terminal.

¹⁰ A Siemens UMTS laboratory was used for this test.

The HTTP request for content made by terminals that use the Pocket PC operating system is presented in Figure 7.12. This figure shows that the terminal has 16 bits per pixel, is color capable (UA-color header), and has a screen resolution of 240 per 320 pixels (UA-pixels header).

```
GET /index.html HTTP/1.1
Accept: */*
UA-OS: Windows CE (POCKET PC) - Version 3.0
UA-color: color16
UA-pixels: 240x320
UA-CPU: ARM SA1110
UA-Language: JavaScript
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/2.0 (compatible; MSIE 3.02; Windows CE; 240x320)
Host: 141.29.152.134:38127
Connection: Keep-Alive
```

Figure 7.12 – Compaq IPAQ Pocket PC terminal HTTP request.

Because the Pocket PC does not send any network characteristics, the real terminal was used only to test image customization; for video customization, UMA browser simulating the relevant user environment features (including network bandwidth) was used.

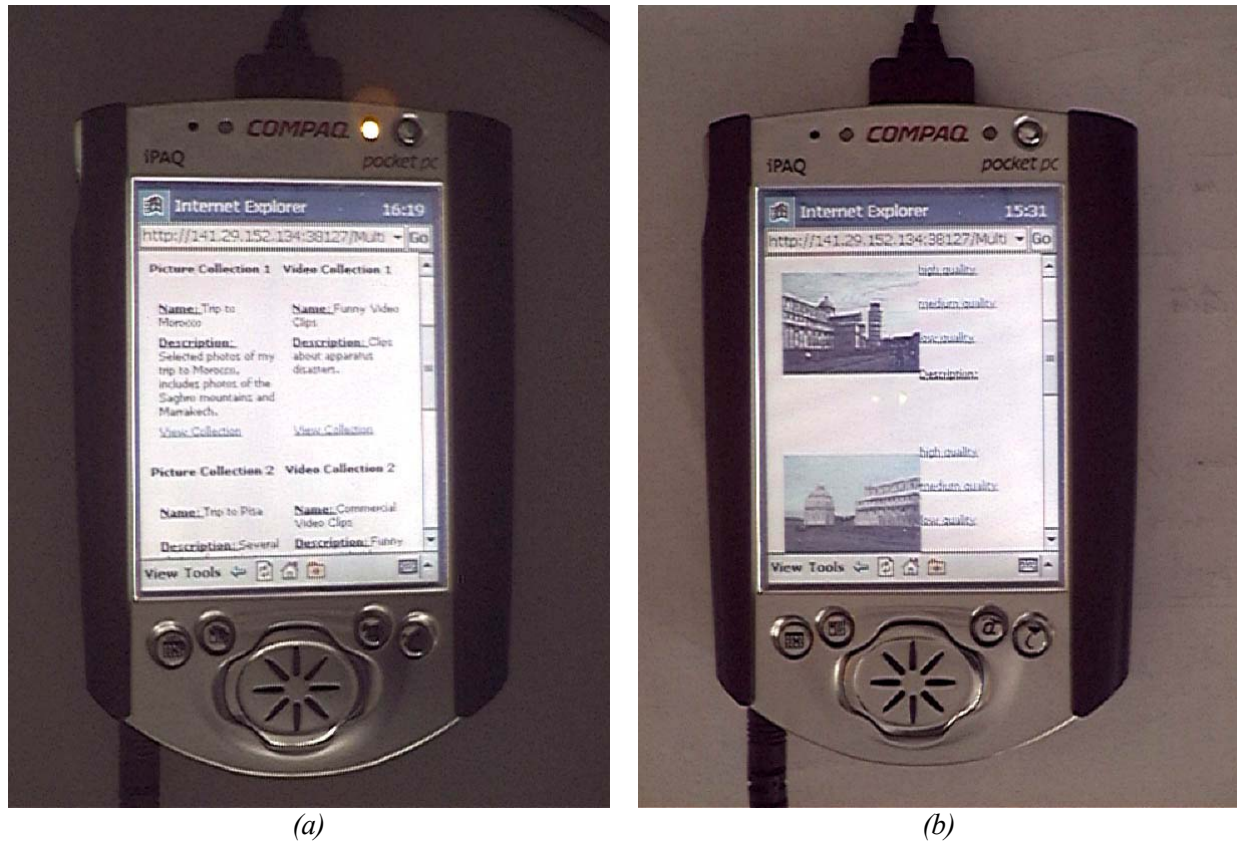


Figure 7.13 – “My Multimedia Album” accessed in a Pocket PC terminal.

7.4.1.1 Image Customization Results

The UMA Platform uses the information present in the HTTP request to customize images. Table 7.3 presents the measures that were made using the UMA Platform internal message logs. The rows in the table have the following meaning:

- **Performed customizations:** content customization actions performed (decided by the Content Action Decision module and sent to the Content Customization module).

- **Customization time:** average time to execute the adaptation in the Content Customization module.
- **Customized compressed size:** adapted content average size in bytes that is transmitted to the terminal.
- **Source compressed size:** average size of the source content that the UMA Platform has to process.
- **Source PCM size:** amount of data in terms of bytes that the adaptation method will have to process.

It can be observed that the memory resources and the CPU time consumed by these customizations are quite large, knowing that the CPU has other tasks to execute such as responding to other user requests. The uncompressed domain content processing is the cause of the high computational cost.

Table 7.3 – Image customization measures for the Pocket PC 1 scenario.

Image	Category A	Category B	Category C	Category D
Performed customization	Spatial resolution reduction; bits per pixel reduction;	Spatial resolution reduction;	Spatial resolution reduction;	Spatial resolution reduction;
Customization time	621 ms	411 ms	300 ms	240 ms
Customized compressed size	10.4 kBytes	10.4 kBytes	10.1 kBytes	10.1 kBytes
Source compressed size	414 kBytes	121 kBytes	58 kBytes	11 kBytes
Source PCM size	5,625 kBytes	1,536 kBytes	300 kBytes	75 kBytes

Figure 7.14 illustrates an image that was delivered by the UMA Platform without being customized to the terminal display (spatial resolution) characteristics. Figure 7.15 illustrates the same image delivered by the UMA Platform but this time customized to the terminal display spatial resolution.



Figure 7.14 – Image not customized to the Pocket PC display resolution.



Figure 7.15 – Image customized to the Pocket PC display resolution.

7.4.1.2 Video Customization Results

The video tests were performed with the UMA browser simulating a device with capabilities similar to the Compaq iPAQ. The UMA browser is connected through a LAN to the UMA Platform. The terminal was not used because it does not send information regarding bandwidth. Table 7.4 presents the measures that were made using the UMA Platform logs. The meaning of each row is:

- **Customized bit rate:** bit rate used by MPEG-1 encoder to encode the adapted video.
- **Customized PCM frame size:** size of each frame that the encoder will encode; this frame is obtained by processing the source frame.
- **Source bit rate:** this the bit rate with which the source content is encoded.
- **Source PCM frame size:** size of each frame that the decoder gives for adaptation by the Content Customization module.

The customization time has such a long value for two reasons: first the customization is performed in the uncompressed domain, and second the decoder/encoder is not an optimized one.

Table 7.4 – Video customization measures for the Pocket PC 1 scenario.

Video	Category A	Category B	Category C
Performed customizations	Spatial resolution reduction; bit rate reduction;	Bit rate reduction;	-

Customization time	22,292 ms	20,810 ms	-
Customized bit rate	64 kbit/s	64 kbit/s	64 kbit/s
Customized PCM frame size	24.75 kBytes	24.75 kBytes	24.75 kBytes
Source bit rate	256 kbit/s	128 kbit/s	64 kbit/s
Source PCM frame size	297 kBytes	135 kBytes	74.25 kBytes

7.4.2 Pocket PC 2 Scenario

The majority of PDAs is grayscale, since only more recently this type of terminals became color capable. The use of grayscale PDAs has the advantage of a longer autonomy. These UMA System tests used the UMA browser to simulate a terminal with such characteristics. The UMA browser is always connected through a LAN connection to the UMA Platform. The DIUED characteristics for this scenario were presented in Table 7.2.

7.4.2.1 Image Customization Results

Table 7.5 presents the measures that were made using the UMA Platform logs. The same comments apply here as for the previous case. The major difference between this case and the previous one is the fact that the adapted content has a smaller size because it is grayscale and not color anymore.

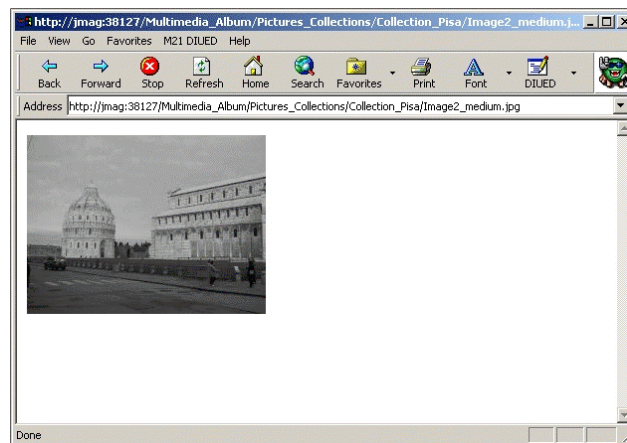


Figure 7.16 – Image customized to the Pocket PC 2 display.

Table 7.5 – Image customization measures for the Pocket PC 2 scenario.

Image	Category A	Category B	Category C	Category D
Performed customizations	Spatial resolution reduction; color space transformation; bits per pixel reduction;	Spatial resolution reduction; color space transformation; bits per pixel reduction;	Spatial resolution reduction; color space transformation;	Spatial resolution reduction; color space transformation;
Customization time	521 ms	310 ms	221 ms	161 ms
Customized compressed size	9,954 Bytes	9,938 Bytes	9,990 Bytes	9,718 Bytes
Source compressed size	414 kBytes	121 kBytes	58 kBytes	11 kBytes
Source PCM size	5,625 kBytes	1,536 kBytes	300 kBytes	75 kBytes

7.4.2.2 Video Customization Results

Table 7.6 presents the measures that were made for video adaptations using the UMA Platform logs. The major difference between this case and the Pocket PC scenario presented before is the fact that the encoder only processes now the luminance information and thus a much smaller amount of data. This reduction of data to be processed by the MPEG-1 encoder leads to lower customization times as can be seen in Table 7.6.

Table 7.6 – Video customization measures for the Pocket PC 2 scenario.

Video	Category A	Category B	Category C
Performed customizations	Spatial resolution reduction; color space transformation; bit rate reduction;	Color space transformation; bit rate reduction;	Color space transformation;
Customization time	21,571 ms	20,099 ms	11,247 ms
Customized bit rate	64 kbit/s	64 kbit/s	64 kbit/s
Source bit rate	256 kbit/s	128 kbit/s	64 kbit/s
Customized PCM frame size	24.75 kBytes	24.75 kBytes	24.75 kBytes
Source PCM frame size	297 kBytes	135 kBytes	74.25 kBytes

The customization time has such a long value for the reasons that were already presented: decoder/encoder optimization, and uncompressed domain content processing. It is noticeable that the removal of color information improves the customization time.

7.4.3 WAP Terminal 1 Scenario

The UMA browser was used to simulate the WAP Terminal 1 scenario with the DIUED characteristics indicated in Table 7.2. There are already today some WAP terminals that have color capabilities and thus this scenario simulates a terminal with such capabilities.

7.4.3.1 Image Customization Results

Table 7.7 presents the measures that were made using the UMA Platform logs. As expected, the total customization time is lower than for the Pocket PC 2 scenario. The smaller customization time is due to the much smaller target image size (from 240x320 to 176x144).

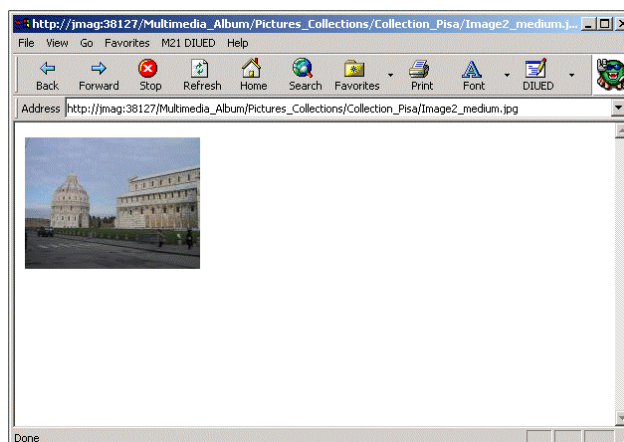


Figure 7.17 – Image customized to the WAP terminal 1 display.

Table 7.7 – Image customization measures for the WAP terminal 1 scenario.

Image	Category A	Category B	Category C	Category D
Performed customizations	Spatial resolution reduction; color space transformation; bits per pixel reduction;	Spatial resolution reduction; bits per pixel reduction;	Spatial resolution reduction;	Spatial resolution reduction;
Customization time	501 ms	300 ms	201 ms	151 ms
Customized compressed size	9,760 Bytes	9,794 Bytes	9,718 Bytes	9,990 Bytes
Source compressed size	414 kBytes	121 kBytes	58 kBytes	11 kBytes
Source PCM size	5,625 kBytes	1,536 kBytes	300 kBytes	75 kBytes

7.4.3.2 Video Customization Results

Table 7.8 presents the measures that were made in terms of video adaptation using the UMA Platform logs. The major difference between this case and the previous one is the use of QCIF resolution for the terminal display. This fact significantly affects the overall customization time because for this resolution the encoder has a much less data to process and thus reduces the used time as can be observed in Table 7.8.

Table 7.8 – Video customization measures for the WAP terminal 1 scenario.

Video	Category A	Category B	Category C
Performed customizations	Spatial resolution reduction; bits rate reduction;	Spatial resolution reduction; bits rate reduction;	Spatial resolution reduction;
Customization time	12,709 ms	11,977 ms	11,296 ms
Customized bit rate	64 kbit/s	64 kbit/s	64 kbit/s
Source bit rate	256 kbit/s	128 kbit/s	64 kbit/s
Customized PCM frame size	24.75 kBytes	24.75 kBytes	24.75 kBytes
Source PCM frame size	297 kBytes	135 kBytes	74.25 kBytes

7.4.4 WAP Terminal 2 Scenario

This user environment corresponds to the Siemens ME45 GPRS terminal with WAP 1.1 capabilities. This scenario was tested using a real terminal and not a UMA browser simulation. The terminal connects to the UMA Platform through an ISP¹¹.

Because the UMA Platform is not capable of adapting HTML content to WML (Wireless Markup Language used in WAP terminals), the content was created both in HTML and WML in order to cover WAP terminals. Figure 7.18 illustrates the navigation in the “My Multimedia Album” content using the Siemens ME 45 terminal. As can be observed in Figure 7.18c, the pictures thumbnails are not available in this version (because the display itself is already too small), and thus only a link is available.

¹¹ TMN (Telecomunicações Móveis Nacionais) was used as the Internet Service Provider to connect the WAP terminal.

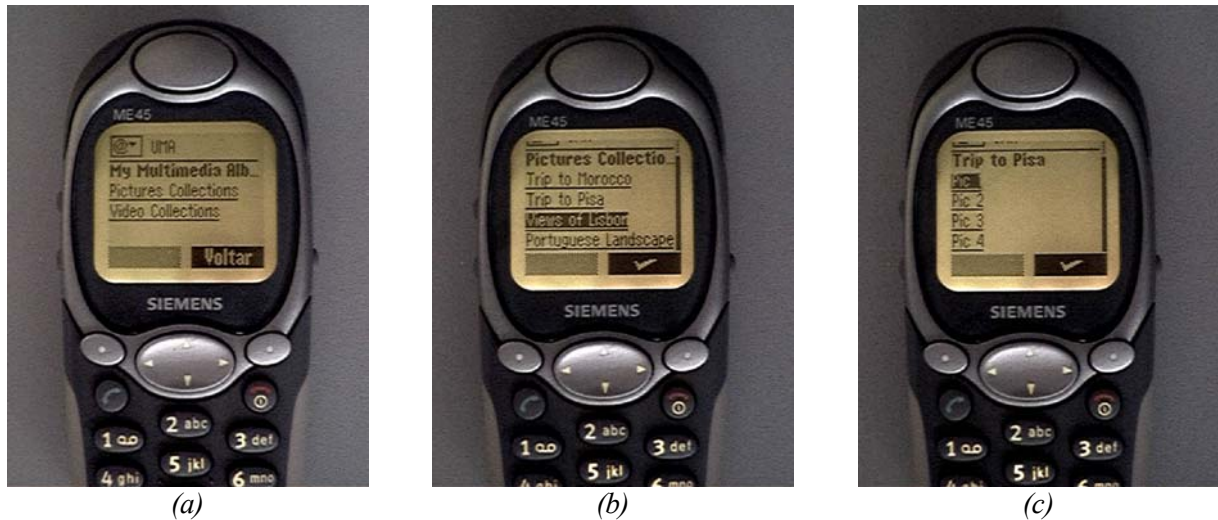


Figure 7.18 – “My Multimedia Album” in WAP terminals: a) the main page; c) the page with the pictures collection; c) the Pisa pictures collection.

Because the real WAP terminal does not have video capabilities the link to the video collections leads to a page that warns the user that his/her terminal is not adequate for that content. Therefore, for testing the video customizations the UMA browser simulated a terminal with similar characteristics to this one.

As a result of the tests with real equipment, it was possible to check the information that the used terminal sends with its request. The HTTP request of this GPRS WAP terminal is presented in Figure 7.19. The request shows that the terminal is not color capable (x-up-devcap-iscolor header), uses 1 bit per pixel (x-up-devcap-screendepth header) and has a screen resolution of 66x101 pixels (x-up-devcap-screenpixels header).

```
GET /index.html HTTP/1.1
Accept-Language: en
Content-Type: application/x-www-form-urlencoded
Accept-Charset: ISO-8859-1, UTF-8, *
x-up-subno: jm_YBRP083368
x-upfax-accepts: none
x-up-devcap-charset: ISO-8859-1
x-up-devcap-smartdialing: 1
x-up-devcap-screendepth: 1
x-up-devcap-iscolor: 0
x-up-devcap-immed-alert: 1
x-up-devcap-max-pdu: 2984
x-up-devcap-numsoftkeys: 2
x-up-devcap-screenpixels: 101,66
x-up-devcap-msize: 8,13
Accept: */*, application/vnd.wap.wml, application/vnd.wap.wmlc,
application/vnd.wap.wmlc;type=1108, application/vnd.wap.wmlscriptc, text/plain, text/vnd.wap.wml,
text/vnd.wap.wmlscript, text/x-hdml, text/x-hdml;version=2.0, text/x-hdml;version=3.0, text/x-
hdml;version=3.1, text/x-wap.wml
User-Agent: SIE-S45/4.0 UP/5.0.1.2 (GUI) UP.Browser/5.0.1.2 (GUI)-XXXX UP.Link/5.0.HTTP-DIRECT
If-Modified-Since: Sat, 30 Mar 2002 17:09:55 GMT; length=16347
Host: 141.29.152.134:38127
```

Figure 7.19 – Siemens ME 45 WAP terminal HTTP request.

7.4.4.1 Image Customization Results

Figure 7.20 illustrates the adaptation of a picture processed with dithering (using a simple threshold to re-quantize the image gives an unusable result). The results for that image are quite good considering the terminal in question. However, the same error-dithering adaptation method for the Figure 7.21 image gives a poor result, making the image completely unperceivable. There are some cases where the image contours provide better results than processing with dithering (e.g. in a soccer photo). However, the UMA

Platform has no way of knowing which adaptation method gives better results for a specific image. This may be a reason to use an MPEG-7 descriptor that instructs UMA applications on the best types of adaptations to perform.

Table 7.9 presents the measures that were made using the UMA Platform logs. As expected, the average customization time is lower than for the WAP terminal 2 scenario, since in this case the color information has been removed. The difference from one image category customization time reduces as in the previous cases.

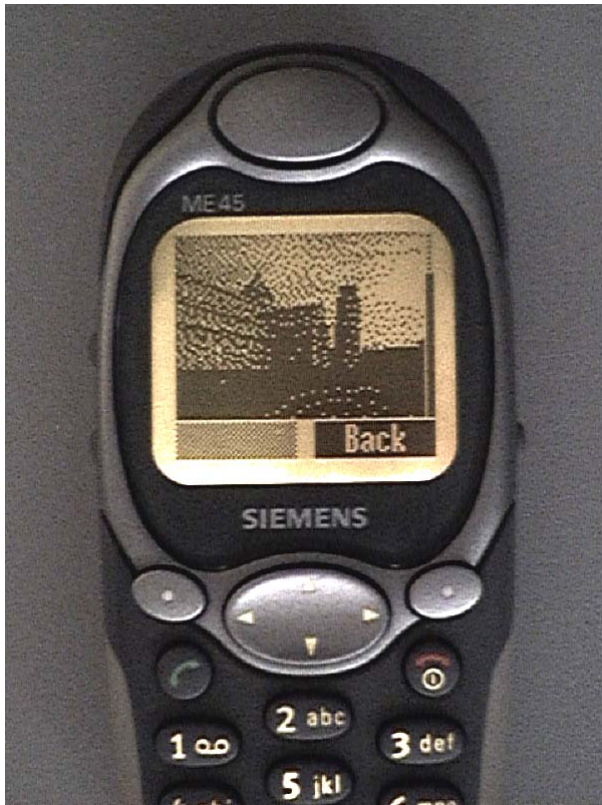


Figure 7.20 – Good result of an image adaptation.

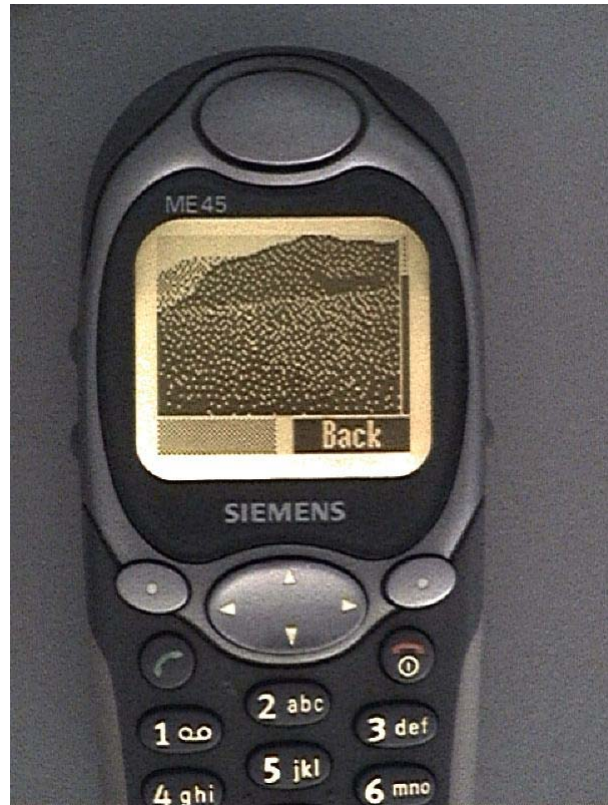


Figure 7.21 – Poor result of an image adaptation.

Table 7.9 – Image customization measures for the WAP terminal 2 scenario.

Image	Category A	Category B	Category C	Category D
Performed customizations	Spatial resolution reduction; color space transformation; bits per pixel reduction;	Spatial resolution reduction; color space transformation; bits per pixel reduction;	Spatial resolution reduction; color space transformation; bits per pixel reduction;	Spatial resolution reduction; color space transformation; bits per pixel reduction;
Customization time	391 ms	180 ms	90 ms	50 ms
Customized compressed size	1,010 Bytes	915 Bytes	854 Bytes	980 Bytes
Source compressed size	414 kBytes	121 kBytes	58 kBytes	11 kBytes
Source PCM size	5,625 kBytes	1,536 kBytes	300 kBytes	75 kBytes

7.4.4.2 Video Customization Results

The video tests were not performed with the Siemens ME45 GPRS terminal because it does not have video capabilities. Instead, the UMA browser was used to simulate a terminal with **similar** display and bandwidth characteristics but now able to decode the received video stream: it was assumed that the display was grayscale. Notice that the used GPRS terminal has a maximum network bandwidth of 24 kbit/s; however, because the MPEG-1 encoder could not work with such a low bit rate the value of 64 kbit/s was used. In order to test such a low bit rate, an H.263 or MPEG-4 Video encoder should be used. Table 7.10 presents the measures that were made for this scenario using the UMA Platform logs.



Figure 7.22 – Visualization of a video customized to the WAP terminal 2.

Table 7.10 – Video customization measures for the WAP terminal 2 scenario.

Video	Category A	Category B	Category C
Performed customizations	Spatial resolution reduction; color space transformation; bit rate reduction;	Spatial resolution reduction; color space transformation; bit rate reduction;	Color space transformation;
Customization time	11,536 ms	11,957 ms	11,217 ms
Customized bit rate	64 kbit/s	64 kbit/s	64 kbit/s
Source bit rate	256 kbit/s	128 kbit/s	64 kbit/s
Customized PCM frame size	24.75 kBytes	24.75 kBytes	24.75 kBytes
Source PCM frame size	297 kBytes	135 kBytes	74.25 kBytes

In this case, the customization time is much smaller than for the first WAP terminal. Once again the reason is that the color information has been removed. The customization time is rather long for two reasons: first the customization is performed in the uncompressed domain and second the decoder/encoder is not an optimized one.

7.5 Summary

The three applications visual interfaces were presented in the first part of this Chapter: the UMA browser, the UMA Platform and the MPEG-7 description tool. In the following parts the used test bed and the test scenarios were presented.

The UMA System was tested under different access conditions, a WAP terminal with a GPRS connection, a Pocket PC terminal with a UMTS connection and the UMA browser with a LAN connection were used to access the customized content. The diversity of test scenarios in which the UMA System was tested confirmed that the adaptation processes are the main cause of the server load, since in any situation the time consumed by content processing was always rather high. More precisely, the uncompressed domain content processing is a major responsible for the poor adaptation performance. Therefore, content processing algorithm optimizations are essential to improve the performance.

When the content source had a high quality and the terminal limited capabilities, the server CPU was rapidly used up for a long period (several milliseconds) by the content processing. In order to improve these situations, where a great gap exists between the available content characteristics and the pretended content characteristics variation, strategic pre-encoded content variations with lower quality are used to ease the customization efforts. This process allowed the UMA Platform to jump from one image category to another image category that has the content with characteristics closer to the terminal capabilities. In order to perform the measures presented in this Chapter, the source content was always used.

The UMA Platform could in fact serve a very diverse range of clients with different multimedia content. Only for WML and HTML the customization was not made since an HTML customization module was not implemented or studied in this thesis. The tests with the Pocket PC, WAP terminal and UMA browser (desktop PC) showed that the same content could be delivered to different terminals and thus reducing the maintenance and storage costs of having one content variation for each terminal type.

There is also a very important test that was not made but for sure the current system would provide some improvements: the terminal battery autonomy. Since the adapted content requires a reduced amount of resources since it is somehow ‘simplified’, the resources consumed in the terminal are smaller.

Chapter 8

Conclusions and Future Work

“The pervasive computing age is dawning. Over the next decades, ... trillions of devices will be always connected. The promise, of course, is anytime, anywhere, any-device access to everyone and everything, ...”

*Jim Spohrer and Mitch Stein
in IEEE Multimedia, April-June 2001.*

Mobile multimedia applications will become a new trend in telecommunications in the next years. The importance of this trend is visible by the recent creation of two new journals in the context of the IEEE Computer Society and IEEE Communications Society: the IEEE Pervasive Computing and the IEEE Transactions on Mobile Computing.

The Universal Multimedia Access challenge presented in Chapter 1 raised some technical problems that were the starting point for this thesis. A framework has been implemented where the Universal Multimedia Access principles, concepts and some proposed techniques and solutions have been tested. For that purpose, a multimedia content collection, the “My Multimedia Album”, was used to test the entire UMA System in a Web scenario.

8.1 Conclusions

The Universal Multimedia Access definition given in Chapter 1 states:

The objective of UMA technology is to make available different presentations of the same information, more or less complex e.g. in terms of media types, suiting different terminals, networks and user preferences.

Based on this definition a UMA System has been designed and implemented. A top-down approach (from a high level approach to the UMA problem, down to the low level of the adaptation algorithms) enabled to develop a rather complete framework to test multimedia adaptation technology. This thesis provides three types of answers, each one addressing the UMA problem at a different level:

- **UMA System:** the end-to-end problem was analyzed at the highest level and a complete framework was designed and tested.
- **UMA Platform:** this is the network element that performs the content customization; a content customization engine and the interfaces required for the UMA Platform were developed.

- **UMA Engine:** the lowest level answer corresponds to the content customization engine that implements the multimedia content adaptation/selection algorithms.

The content customization engine is divided into four parts, all central to this thesis:

- 1) the multimedia content description tool;
- 2) the user environment description tool;
- 3) the customization decision algorithms; and
- 4) the multimedia adaptation algorithms.

Concerning **multimedia content description**, several solutions were considered; the one that best fulfills the UMA requirements was presented in Chapter 3: the MPEG-7 standard. While implementing the required MPEG-7 tools, some missing elements were detected. These elements were proposed by the author to the MPEG Committee in July 2001 [116]. This document proposed improvements and extensions to the MPEG-7 standard that were accepted and used to improve the standard. From the experience acquired with the use of the MPEG-7 standard it was concluded that:

- MPEG-7 provides powerful description tools for audiovisual content notably targeting UMA applications (e.g. transcoding hints tools).
- MPEG-7 does not provide a solution for the description of certain types of composite content (only considers content that is composed by audio and/or visual streams, e.g. content similar to MPEG-4). Therefore, MPEG-7 is not yet powerful enough for UMA applications when the adaptation of other types of composite content such as HTML or WML.

While investigating the possible solutions for the **user environment description** problem, the most important conclusion was the lack of a generic user environment (or user context) description framework. When testing the UMA System with real terminals, the missing technology was always the user environment description. Usually almost no information is available regarding the user environment when accessing content: WAP terminals provide a reasonable description of the user environment; PocketPC terminals provide only information about its screen; and for other terminals almost no information is available. The conclusion is that there is no generic and powerful user environment description framework that covers a large range of terminals and contexts.

This problem was presented to the MPEG Committee, which decided to create a new part of the MPEG-21 standard targeting the provision of a generic user environment or usage context description framework. This MPEG-21 part is called Digital Item Adaptation (DIA). One of DIA objectives is the creation of a Digital Item Usage Environment Description (DIUED) framework, which should grant any terminal the ability of describing its environment in a standardized way so that any application can provide its service in a personalized way, thus enabling “context-aware” applications.

The contributions made to the MPEG Committee, in the context of this thesis, regarding the user environment framework were:

- Study of the most relevant technologies to address the user environment description problem [123]; following this study, MPEG created an Ad-Hoc group to further discuss and analyze the problem.

- Proposal of a generic framework to describe multimedia user environments [124]; following this proposal, MPEG asked for the creation of part 7 of the MPEG-21 standard: Digital Item Adaptation. At the MPEG March 2002 meeting, a Call for Proposals on DIA technology was issued; the answers to this Call will be evaluated in May 2002.

The **customization decision** algorithms developed in this thesis are rather simple since the customization was always made for a single media type, e.g. images or video. The decision algorithms considered always the display characteristics; for video, also the network characteristics were considered. The performed tests allowed to conclude that the implemented decision algorithms offer good results for the considered user environment characteristics. The adaptation decision is a part of the UMA Engine that does not consume much resources but greatly influences the performance of the overall system. More elaborated decision algorithms based on iterative decisions to consider the impact of one adaptation decision on other adaptation decisions would lead to more optimized decisions and thus better adaptation results. Furthermore, composite content requires more sophisticated algorithms, due to the possible dependence of the adaptation of one object on the adaptation of another object within the same piece of content.

As expected, the **adaptation algorithms** have a crucial role in the overall UMA System performance. The implemented algorithms required decompressing the content into memory to customize it and then recompress it again (uncompressed domain processing). Since the adaptation process consumes much CPU resources and it was not optimized, this block is in fact the bottleneck of the UMA Platform implemented. It is thus possible to conclude that the use of more efficient adaptation algorithms could significantly improve the performance of the overall system (e.g. compressed domain processing).

From the overall system, it is possible to conclude that the existence of an MPEG-7 description server (for the content descriptions), as well as a user environment description server, like the WAP UAProf server [50], are essential to facilitate the access the necessary features to adequately adapt the content. The network component implemented allowed to achieve a more complete UMA System and to get a better perspective on how UMA concepts must be deployed in a real scenario. Even though the UMA System was implemented using Internet technologies, it could easily be implemented with any other network technology. For example, it was concluded that the content server configuration provides technical advantages over the proxy configuration notably in terms of content processing because the content is locally available. However, the proxy configuration is by far the most interesting configuration for a real scenario since it allows reusing existing Web content and infrastructures.

The key factors for a good universal multimedia access system include user environment description, multimedia content description, the customization decision algorithms and the adaptation algorithms. UMA technology will not find an answer from a single research area; the answer will come together from all the research areas previously mentioned since they are all interdependent.

8.2 Future Work

The universal multimedia access research topic is far from being completely studied. This section lists some of the topics that may deserve further research in the context of future UMA developments:

1. **Optimization of adaptation algorithms:** development and implementation of more powerful adaptation algorithms, notably based on compressed domain content processing;
2. **Inclusion of summarization capabilities:** Table 3.1 shows a list of important content customization actions, notably transcoding, quality reduction and summarization; for certain environments and types of content, summarization brings an extra essential adaptation

dimension that was not explored in the UMA System developed for this thesis. A very relevant improvement to the UMA System developed would be the creation of content descriptions including summarization features targeting the generation of different types of summaries in the UMA customization module depending on the user environment characteristics;

3. **Consideration of new media types:** consideration in the UMA System of additional media types from the list presented in Chapter 1 (text, image, video, audio, structure elements and interactive elements) in terms of content adaptation:
 - a. Audio: visual content is already supported by the present UMA System (besides text); therefore the adaptation of audio content is essential since in many applications these two media types are used together (e.g. multimedia communications, movie scene);
 - b. Structural elements: the support of this media type implies the processing of content that can be spatially re-structured, such as HTML content;
4. **Implementation of additional MPEG-7 Ds and DSs:** inclusion in the UMA System of more MPEG-7 descriptors related to UMA functionalities such as SegmentRelationDS, MediaTranscodingD, SummarizationDS as presented in Chapter 3;
5. **Development of additional MPEG-7 Ds and DSs:** since MPEG-7 still has limited capabilities, e.g. in terms of composite content adaptation, the development of MPEG-7 descriptors able to improve UMA applications should be considered for MPEG-7 Version 2;
6. **Development of additional user environment descriptors:** because the list of user environment characteristics presented in this thesis can be improved in order to cover many more access conditions and user preferences, the definition of additional user environment descriptors shall be considered;
7. **Investigation on the benefits of scalable coding techniques:** scalable coding techniques have obvious advantages in terms of UMA applications since scalable content does not require much adaptation when the access conditions vary; this is due to the fact that the content itself provides different consumption layers that can be dynamically used according to quality that the resources allow for; the types of scalability, the granularity as well as the efficiency of scalable techniques are today a hot topic of research;
8. **Provision of real-time streaming capabilities:** the real-time streaming scenarios require that the UMA System be able to perform real-time adaptation while the content is being retrieved and simultaneously streamed. Moreover the Quality of Service must not be degraded by a poor performance of the UMA System; several buffering problems are expected time critical content processing is required.

The list above has an impact mostly on the MPEG-7 description tool and the UMA Platform. The UMA browser is essentially a test tool for the UMA Platform, and only the required user environment descriptors must be added to that application. However, the last topic of the previous list requires that the UMA browser implement a real-time streaming client.

In conclusion, universal multimedia access is a rather recent research topic that still keeps many interesting problematic aspects which need to be researched and solved.

References

- [1] R. Mohan, J. Smith, C.-S. Li, "Adapting Multimedia Internet Content for Universal Access," IEEE Transactions on Multimedia, March 1999.
- [2] Wei-Ying Ma, Ilja Bedner, Grace Chang, Allan Kuchinsky, and HongJiang Zhang "A framework for adaptive content delivery in heterogeneous network environments", Proc. of SPIE/ACM Conference on Multimedia Computing and Networking, San Jose, Feb. 2000.
- [3] Eija Kaasinen, Matti Aaltonen, Juha Kolari, Suvi Melakoski, Timo Laakko, "Two Approaches to Bringing Internet Services to WAP Devices," 9th International WWW Conference, May 2000.
- [4] Charilaos Christopoulos, Touradj Ebrahimi, V.V. Vinod, John R. Smith, Rakesh Mohan, and Chung-Sheng Li, "MPEG-7 Application: Universal Multimedia Access Through Content Repurposing and Media Conversion", ISO/IEC JTC1/SC29/WG11 MPEG99/M4433, March 99, Seoul, Korea.
- [5] Soamedv Acharya, "Techniques for Improving Multimedia Communication Over Wide Area Networks", PhD Thesis, Cornell University, 1999.
- [6] Moving Pictures Expert Group – MPEG
<http://mpeg.telecomitalialab.com/>
- [7] Digital Video Broadcasting – DVB
<http://www.dvb.org>
- [8] Society of Motion Picture Television Engineers – SMPTE
<http://www.smpte.org>
- [9] Dublin Core Metadata Initiative
<http://dublincore.org>
- [10] Apache HTTPD Server Project
<http://httpd.apache.org/>
- [11] Apache Xerces XML Parser
<http://xml.apache.org/>
- [12] ImageMagick
<http://www.imagemagick.org/>
- [13] MPEG-1 / MPEG-2 encoder and decoder software.
<http://www.mpeg.org/MPEG/MSSG/>
- [14] ASP – Active Server Pages
<http://msdn.microsoft.com/asp/>

- [15] Java Server Pages Technology
<http://java.sun.com/products/jsp/>
- [16] Argy Krikelis, “Mobile multimedia-shaping the Infoverse”, IEEE Concurrency, January–March 1999.
- [17] Argy Krikelis, “Location-dependent multimedia computing”, IEEE Concurrency, April–June 1999.
- [18] Argy Krikelis, “Challenges in delivering multimedia content in mobile environments”, IEEE Concurrency, July–September 1999.
- [19] Argy Krikelis, “Mobile multimedia considerations”, IEEE Concurrency, October–December 1999.
- [20] Argy Krikelis, “Considerations for a new generation of mobile multimedia communication systems”, IEEE Concurrency, April–June 2000.
- [21] Argy Krikelis, “Enhancing visual quality in mobile multimedia”, IEEE Concurrency, July–September 2000.

A.1 IETF references

- [22] Internet Engineering Task Force
<http://www.ietf.org>
- [23] HTTP T. Berners-Lee, R. Fielding, H. Frystyk, "Hypertext Transfer Protocol – HTTP/1.0", RFC 1945, W3C/MIT, UC Irvine, W3C/MIT, May 1996.
- [24] K. Holtman, A. Mutz, “Transparent Content Negotiation in HTTP”, RFC 2295, March 1998.
- [25] K. Holtman, A. Mutz, T. Hardie, “Media Feature Tag Registration Procedure”, RFC 2506, March 1999.
- [26] L. Masinter, D. Wing, A. Mutz, K. Holtman, “Media Feature for Display, Print, and Fax”, RFC2534, March 1999.
- [27] G. Klyne, “A syntax for describing media feature sets”, RFC2533, March 1999.
- [28] T. Howes, “The String Representation of LDAP Search Filters”, RFC2254, December 1997.
- [29] Berners-Lee, T., Masinter, L., and M. McCahill, Editors, "Uniform Resource Locators (URL)", RFC 1738, December 1994.
- [30] T. Berners-Lee, R. Fielding, L. Masinter, “Uniform Resource Identifiers (URI): Generic Syntax”, RFC 2396, August 1998.

A.2 W3C references: CC/PP, RDF and XML Schema

- [31] World Wide Web Consortium
<http://www.w3c.org>
- [32] W3C, “Extensible Markup Language”
<http://www.w3.org/TR/REC-xml>
- [33] W3C, “XML Schema Part 0: Primer”
<http://www.w3.org/TR/xmlschema-0/>
- [34] W3C, “XML Schema Part 1: Structure”
<http://www.w3.org/TR/xmlschema-1/>
- [35] W3C, “XML Schema Part 2: Datatypes”
<http://www.w3.org/TR/xmlschema-2/>
- [36] W3C, “Document Object Model”
<http://www.w3.org/DOM/DOMTR>
- [37] W3C, “Resource Description Framework”
<http://www.w3.org/TR/REC-rdf-syntax/>
- [38] W3C, “W3C Mobile Access”:
<http://www.w3c.org/Mobile>.
- [39] W3C, “CC/PP Architecture and Requirements”:
<http://www.w3.org/TR/2000/WD-CCPP-ra-20000721/>.
- [40] W3C, “CC/PP Structure and Vocabularies”, 15 March 2001:
<http://www.w3.org/TR/2001/WD-CCPP-struct-vocab-20010315/>.
- [41] Television and Web Activity
<http://www.w3c.org/TV/>
- [42] Device Independence Activity
<http://www.w3c.org/2001/di/>
- [43] W3C, “Namespaces in XML”:
<http://www.w3.org/TR/REC-xml-names/>.
- [44] W3C “External Annotation of Web Content for Transcoding”:
<http://www.w3.org/TR/annot>.
- [45] W3C XHTML, “HTML Home Page”:
<http://www.w3.org/MarkUp>.

A.3 WAP references

- [46] Wireless Application Protocol

<http://www.wapforum.org>

- [47] WAP Forum Ltd ,“WAP 2.0 Technical White Paper”, August 2001.
- [48] WAP Forum Ltd, “Wireless Application Protocol – Architecture Specification 2.0”, 12 July 2001.
- [49] WAP Forum Ltd ,“Wireless Application Environment Specification 2.0”, 29 July 2001.
- [50] WAP Forum Ltd ,“WAG – User Agent Profile Specification 2.0”, 30 May 2001.

A.4 3GPP references

- [51] 3GPP , "General UMTS Architecture ", TR 23.101.
- [52] 3GPP , "Network Architecture", TR 23.002.
- [53] 3GPP , "Core Networks Protocols", TR 23.108.
- [54] 3GPP , "UTRAN Overall Description ", TR 25.401.
- [55] 3GPP , "Real Time Multimedia", TR22.972.
- [56] 3GPP , "Mobile Multimedia Services", TR 22.960.
- [57] 3GPP , "Multimedia Messaging Service” TR 22.140.
- [58] 3GPP , "Modifications to H.324", TR 26.111.
- [59] 3GPP , "Technical Specification Group Services and System Aspects; Mobile Station Execution Environment; Service Description", TR 22.057.
- [60] 3GPP , "Technical Specification Group Terminals; Mobile Station Execution Environment; Functional Description", TR 23.057.
- [61] 3GPP , "Universal Mobile Telecommunications System (UMTS); Service aspects; Provision of Services in UMTS – The Virtual Home Environment", TR 22.170.
- [62] 3GPP , "Universal Mobile Telecommunications System (UMTS); Provision of Services in UMTS – The Virtual Home Environment: Stage 1”, TS 22.121.

A.5 Ericsson articles related to Mobile Multimedia

- [63] Universal Multimedia Access Demo
<http://itswww.epfl.ch/~uma/>
- [64] Mats Nilsson, “Third-generation radio access standards”, ”, Ericsson Review n°3 1999.
- [65] Torbjörn Nilsson, “Toward third-generation mobile multimedia communication”, Ericsson

Review n°3 1999.

- [66] Gyran Swedberg, "Ericsson's mobile location solution", Ericsson Review n°4 1999.
- [67] Christoffer Andersson and Patrik Svensson, "Mobile Internet – An industry-wide paradigm shift?", Ericsson Review n°4 1999.
- [68] Staffan Pehrson, "WAP – The catalyst of the Mobile Internet", Ericsson Review n°1 2000.
- [69] Joel Askelyf, Charilaos Christopoulos, Mathias Larsson Carlander and Fredrik Oijer, "Wireless image applications and next-generation imaging", Ericsson Review n°2 2001.
- [70] Tim Murphy, "The cdma2000 packet core network", Ericsson Review n°2 2001.

A.6 IBM articles related to Universal Multimedia Access

- [71] IBM Transcoding Technology,
http://www.research.ibm.com/networked_data_systems/transcoding/
- [72] WebSphere
- [73] R. Han, P. Bhagwat, "Dynamic Adaptation in an Image Transcoding Proxy for Mobile Web Browsing", IEEE Personal Communications Magazine, Dec. 1998.
- [74] C.-S. Li, R. Mohan and J. R. Smith, "Multimedia Content Description in the InfoPyramid," IEEE Proc. Int. Conf. Acoust., Speech, Signal Processing (ICASSP), Seattle, WA, Special session on Signal Processing in Modern Multimedia Standards, May, 1998.
- [75] J. R. Smith, R. Mohan, C.-S. Li, "Content-based Transcoding of Images in the Internet," Proceedings of the International Conference on Image Processing (ICIP), 1998.
- [76] J. R. Smith, R. Mohan and C.-S. Li, "Transcoding Internet Content for Heterogeneous Client Devices," Proc. IEEE Inter. Symposium on Circuits and System. (ISCAS), Special session on Next Generation Internet, June, 1998.
- [77] J. R. Smith, "Digital Video Libraries and the Internet," IEEE Communications, special issue on the Next Generation Internet, January 1999.
- [78] J. R. Smith, "VideoZoom Spatio-temporal video browser," IEEE Trans. Multimedia, Vol. 1, No. 2, June 1999.
- [79] J. R. Smith, V. Castelli and C.-S. Li, "Adaptive Storage, Retrieval of Large Compressed Images," Proc. IS&T/SPIE Symposium on Electronic Imaging: Science, Technology – Storage & Retrieval for Image, Video Databases VII, San Jose, CA, January, 1999.
- [80] S. Paek and J. R. Smith, "Detecting Image Purpose in World-Wide Web Documents," Proc. IS&T/SPIE Symposium on Electronic Imaging: Science, Technology – Document Recognition, San Jose CA, January 1998.
- [81] R. Han, J. R. Smith, "Internet Transcoding for Universal Access," Multimedia

Communications Handbook, 1999.

- [82] J. Spohrer, M. Stein, “User Experience in the Pervasive Computing Age”, IEEE Multimedia, January-March 2000.

A.7 MPEG-7 documents

- [83] F.Pereira, R. Koenen, "MPEG-7: a standard for multimedia content description", Special Issue on Image and Video Databases, International Journal of Image and Graphics, vol.1, n°3, August 2001.
- [84] Rob Koenen, “MPEG-7 Context and Objectives”, ISO/IEC JTC1/SC29/WG11/N2861, July 1999.
- [85] F. Pereira, “MPEG-7 Requirements”, ISO/IEC JTC1/SC29/WG11/N3933, January 2000.
- [86] Claude Seyrat, Michael Wollborn, Ali Tabatabai, Olivier Avaro, “Text of ISO/IEC FCD 15938-1 Information Technology – Multimedia Content Description Interface – Part 1 System”, ISO/IEC JTC1/SC29/WG11 MPEG99/M4433, May 2001
- [87] Jane Hunter, Claude Seyrat, Cédric Thiénot, Ernest Wan, “Text of ISO/IEC FCD 15938-2 Information Technology – Multimedia Content Description Interface – Part 2 Definition Description Language (DDL)”, May 2001
- [88] Leszek Cieplinski, Munchurl Kim, Jens-Rainer Ohm, Mark Pickering, Akio Yamada, “Text of ISO/IEC FCD 15938-3 Information Technology – Multimedia Content Description Interface – Part 3 Visual”, ISO/IEC JTC1/SC29/WG11 MPEG99/M4433, May 2001
- [89] Audio Group, “Text of ISO/IEC FCD 15938-4 Information Technology – Multimedia Content Description Interface – Part 4 Audio”, ISO/IEC JTC1/SC29/WG11 MPEG99/M4433, May 2001
- [90] Peter van Beek, Ana B. Benitez, Joerg Heuer, Jose Martinez, Philippe Salembier, John Smith, Toby Walker (editors), “Text of ISO/IEC FCD 15938-5 Information technology – Multimedia content description interface: Multimedia description schemes”, ISO/IEC JTC1/SC29/WG11 MPEG99/M4433, May 2001.
- [91] Reference Software Group, “Text of ISO/IEC FCD 15938-6 Information Technology – Multimedia Content Description Interface – Part 6 Reference Software”, ISO/IEC JTC1/SC29/WG11 MPEG99/M4433, May 2001
- [92] Touradj Ebrahimi and Charilaos Christopoulos, “Can MPEG-7 be used beyond database applications?”, ISO/IEC JTC1/SC29/WG11 MPEG99/M3861, October 98.
- [93] Y. Abdeljaoued, T. Ebrahimi, Ch. Christopoulos,, I. Mas Ivars, J. Smith, ”A new algorithm for video summarization”, ISO/IEC JTC1/SC29/WG11 MPEG99/M4738, March 99.
- [94] John R. Smith, Chung-Sheng Li, Rakesh Mohan, Atul Puri, Charilaos Christopoulos, Ana B. Benitez, Paul Bocheck, Shih-Fu Chang, Touradj Ebrahim, V.V. Vinod , “MPEG-7 Content Description for Universal Multimedia Access”, ISO/IEC JTC1/SC29/WG11

- MPEG99/M4749, July 99.
- [95] John R. Smith, Chung-Sheng Li, Atul Puri, “Model Description Scheme for MPEG-7”, ISO/IEC JTC1/SC29/WG11 MPEG99/M4750, July 99.
 - [96] John R. Smith, Patrick Brigger, Chung-Sheng Li, Ana B. Benitez, Shih-Fu Chang, “Conceptual Modeling of MPEG-7 Description Schemes”, ISO/IEC JTC1/SC29/WG11 MPEG99/M4775, July 99.
 - [97] John R. Smith, Chung-Sheng Li, Ana B. Benitez, Paul Bocheck, Shih-Fu Chang, Charilaos Christopoulos, Sanghoon Sull, “Validation Experiment for MPEG-7 Description Schemes related to Universal Multimedia Access (UMA)”, ISO/IEC JTC1/SC29/WG11 MPEG99/M5086, September 99.
 - [98] Y. Abdeljaoued, T. Ebrahimi, Ch. Christopoulos, I. Mas Ivars, “Video Summarization for Universal Multimedia Access Applications”, ISO/IEC JTC1/SC29/WG11 MPEG99/M5105, September 99.
 - [99] Ana B. Benitez, Paul Bocheck, Shih-Fu Chang, John R. Smith, Rick Han, Chung-Sheng Li, Joe Chicharo, Andrew Perkis, Sanghoon Sull, ”Work Plan for Validating the Media Transcoding Hint DS for Universal Multimedia Access”, ISO/IEC JTC1/SC29/WG11 MPEG99/M5433, November 99.
 - [100] John R. Smith, Sanghoon Sull, Charilaos Christopoulos, Yousri Abdeljaoued, Ana B. Benitez, “Report on Validation Experiments for Universal Multimedia Access (UMA) – Part 1”, ISO/IEC JTC1/SC29/WG11 MPEG99/M5437, November 99.
 - [101] John R. Smith, Chung-Sheng Li, Sanghoon Sull, Ana B. Benitez, “Work Plan for Validating the Segment Hint DS for Universal Multimedia Access (UMA)”, ISO/IEC JTC1/SC29/WG11 MPEG99/M5438, November 99.
 - [102] John R. Smith, Chung-Sheng Li, Ana B. Benitez, “Work Plan for Validation of Space and Frequency View Description Schemes”, ISO/IEC JTC1/SC29/WG11 MPEG99/M5439, November 99.
 - [103] Y. Abdeljaoued, T. Ebrahimi, Ch. Christopoulos, Aljoscha Smolic, “Validation Experiment for MosaicDS”, ISO/IEC JTC1/SC29/WG11 MPEG99/M5504, November 99.
 - [104] Yousri Abdeljaoued, Touradj Ebrahimi, Charilaos Christopoulos “A proposal for the modification of the Sequential Summary DS”, ISO/IEC JTC1/SC29/WG11 MPEG99/M5566, November 99.
 - [105] A. Smolic, J.-R. Ohm, C. Christopoulos, H. Walin, Y. Abdeljaoued, T. Ebrahimi, “Results of CE on MosaicDS – Proposal for a DDL Specification”, ISO/IEC JTC1/SC29/WG11 MPEG00/M5697, March 2000.
 - [106] Hekan Wallin, Charilaos Christopoulos, A. Smolic, Y. Abdeljaoued, T. Ebrahimi, “Robust Mosaic construction algorithm”, ISO/IEC JTC1/SC29/WG11 MPEG00/M5698, March 2000.
 - [107] John R. Smith, Ana B. Benitez, Joerg Heuer, Osamu Hori, “Report on Core Experiment on the Space and Frequency View DS”, ISO/IEC JTC1/SC29/WG11 MPEG00/M5734, March

2000.

- [108] I. Sezan, K. Yoon, K. Emura, S. Sull, A. Perkis, C. Christopoulos, Y. Abdeljaoued, “Report on Results of Core Experiments on User Preference DS”, ISO/IEC JTC1/SC29/WG11 MPEG00/M5784, March 2000.
- [109] Ana B. Benitez, John R. Smith, Joe Chicharo, Andrew Perkis, Sanghoon Sull, Charilaos Christopoulos, Teruhiko Suzuki, “Report on Core Experiment on Media Transcoding Hint DS”, ISO/IEC JTC1/SC29/WG11 MPEG00/M5803, March 2000.
- [110] Peter Kuhn, Teruhiko Suzuki, Anthony Vetro, John R. Smith, Ana B. Benitez, Charilaos Christopoulos, “Report on the Transcoding Hint DS Core Experiment”, ISO/IEC JTC1/SC29/WG11 MPEG00/M6002, May 2000.
- [111] Dulce Ponceleon,, Peter van Beek, Ibrahim Sezan,, Osamu Hori ,Masaru Sugano,, Nevenka Dimitrova , Gulrukh Ahanger,, C. Christopoulos ,, Y. Abdeljoued, “Report on the CE on Summarization Description Schemes”, ISO/IEC JTC1/SC29/WG11 MPEG00/M6006, May 2000.
- [112] Anthony Vetro, Peter Kuhn, Teruhiko Suzuki, John R. Smith, Ana B. Benitez, Charilaos Christopoulos, “CE Report on Media Transcoding Hint DS”, ISO/IEC JTC1/SC29/WG11 MPEG00/M6168, July 2000.
- [113] Sanghoon Sull, Keansub Lee, “Improving the Media Transcoding Hint DS by adding an attribute for spatial resolution , “reduction”, ISO/IEC JTC1/SC29/WG11 MPEG00/M6267, July 2000.
- [114] Jiuhuai Lu, Michel Rynderman, John R. Smith, “Report of Core Experiment on MediaQuality DS”, ISO/IEC JTC1/SC29/WG11 MPEG00/M6354, July 2000.
- [115] John R. Smith, Uma Srinivasan, “Report of the AHG on Conceptual Modeling”, ISO/IEC JTC1/SC29/WG11 MPEG00/M6619, October 2000.
- [116] João Magalhães, Fernando Pereira, “Extension of MediaProfileDS”, ISO/IEC JTC1/SC29/WG11 MPEG00/M7310, July 2001.

A.8 MPEG-21 documents

- [117] Keith Hill et all (editors), “MPEG-21 Part 1: Vision, Technologies and Strategy”, ISO/IEC JTC1/SC29/WG11, N4333.
- [118] Vaughn Iverson, Young-Won Song, Rik Van de Walle, Mark Rowe, Doim Chang, Ernesto Santos, Todd Schwartz, “MPEG-21 Part 2: Digital Item Declaration CD”, ISO/IEC JTC 1/SC 29/WG 11, N4248 July 2001.
- [119] Niels Rump, Young-Won Song, “MPEG-21 Part 3: Digital Item Identification and Description WD 2.0”, ISO/IEC JTC 1/SC 29/WG 11, N4249, July 2001.
- [120] ISO/IEC JTC1/SC29/WG11, “MPEG-21 Part 6: Digital Item Adaptation”, to be published.

- [121] Anthony Vectro (editor), “MPEG-21 Requirements on Digital Item Adaptation”, ISO/IEC JTC 1/SC 29/WG 11, N4515, December 2001.
- [122] Requiements Group, “Preliminary Call for Digital Item Adaptation”, ISO/IEC JTC 1/SC 29/WG 11, N4514, December 2001.
- [123] João Magalhães, Fernando Pereira, “User Environment Characterization for UMA Applications”, ISO/IEC JTC 1/SC 29/WG 11, M6744, January 2001.
- [124] João Magalhães, Fernando Pereira, “Enhancing user interaction in UMA applications”, ISO/IEC JTC 1/SC 29/WG 11, M7312, July 2001.

A.9 Color Temperature

- [125] S. J. Sangwine, R. E. N. Horne, “The Colour Image Processing Handbook”, Chapman & Hall, 1998.
- [126] Gunter Wyszecki, W. S. Stiles, “Color Science : Concepts and Methods, Quantitative Data and Formulae”, Wiley Series in Pure and Applied Optics, 1982.
- [127] Matthew J. Ochs, “Development of a Linear Version of the CIECAM97s Color Appearance Model”,
<http://www.cis.rit.edu/research/thesis/bs/2000/ochs/thesis.htm>
- [128] A.P.Petrov, Chang-Yeong Kim, Yang-Seok Seo, In-So Kweon, “Perceived illuminant measured,” Color Research and Application, Vol.23, No.3, pp.159-168, 1998.
- [129] J.Y.Kim, D.S.Park, C.Y.Kim, Y.S.Seo, and Y.H.Ha, "New algorithm for detecting illuminant chromaticity from color images", Proceedings of SPIE (Electronic Imaging 2000) Vol. 3963, 2000.
- [130] Youngsik Huh, Du Sik Park, Sang Kyun Kim, Yang Lim Choi, “A Proposal for Display Preference Descriptions in MPEG-7 MDS”, ISO/IEC JTC1/SC29/WG11 M7263, July 2001, Sydney, Australia.
- [131] Sang Kyun Kim, Du Sik Park, “Color temperature descriptor for display preference”, ISO/IEC JTC1/SC29/WG11 M7264, July 2001, Sydney, Australia.
- [132] Sang Kyun Kim, Du Sik Park, “Report of VC-6 on MPEG-7 color temperature browsing descriptors”, ISO/IEC JTC1/SC29/WG11 M7265, July 2001, Sydney, Australia.
- [133] Steven A. Shafer, “Using color to separate reflection components”, COLOR Research and Application, 10(4):210–218, 1985.

REFERENCES