

Enriching Multimedia Content Description for Broadcast Environments: From A Unified Metadata Model to A New Generation of Authoring Tool

Wilfried JOUVE

Under the supervision of Laure Berti-Equille and Boris Rousseau

IRISA Research Laboratory, INRIA Rennes

Abstract

This master's thesis introduces a novel approach for authoring a diversity of multimedia resources (audio, video, text, images, etc). I designed an authoring tool (the M-Tool) supporting a metadata model unifying MPEG-21 and TV-Anytime descriptions. The M-Tool aims to answer numerous challenges resulting from the use of metadata in various broadcast scenarios. In addition to basic functionalities (edition and enrichment of audiovisual contents), innovative functionalities of the M-Tool are presented. These include the M-Tool search engine which enables querying TV-Anytime and MPEG-21 structures. This new generation of metadata authoring tools is designed and currently used for scenarios of TV and News broadcasting and video on demand broadcasting in the framework of the European project ENTHRONE.

Keywords: Metadata, Multimedia Authoring, MPEG-21, TV-Anytime, XML, Audiovisual Resources, QoS, Search Engine

Description de Contenus Multimédia dans les Réseaux de Diffusion: D'un Modèle de Métadonnées Unifié à un Outil d'Édition de Nouvelle Génération

Wilfried JOUVE

Sous la direction de Laure Berti-Equille et Boris Rousseau

Laboratoire de Recherche de l'IRISA, INRIA Rennes

Résumé

Ce mémoire présente une nouvelle approche pour l'annotation de contenus multimédia très variées (audio, vidéo, texte, images, etc). J'ai conçu un prototype (le M-Tool) qui exploite le modèle de métadonnées développé dans le cadre du projet Européen ENTHRONE. Ce modèle utilise les standards MPEG-21 et TV-Anytime. Le M-Tool répond à de nombreux besoins liés à l'utilisation des métadonnées dans les réseaux de distribution de contenus multimédia. Les fonctionnalités du M-Tool sont exposées à travers les descriptions des différents modules : Les modules dédiées pour le Tv-Anytime et le MPEG-21 et le moteur de recherche. Cette nouvelle génération d'outils est conçue et actuellement utilisée dans le projet Européen ENTHRONE qui rassemblent des scénarios aussi variés que la diffusion de la télévision, de news et de vidéos à la demande.

Mots clés: Métadonnées, outil d'édition, MPEG-21, TV-Anytime, XML, contenus audiovisuels, qualité de Service, moteur de recherche

Remerciements

Je remercie Boris Rousseau pour avoir été à mes cotés et pour son aide tout au long de mes 6 mois de stage. Je remercie Laure Berti-Equille pour ses conseils et pour la confiance qu'elle m'a accordée. Je remercie également tous les membres de l'équipe TEXMEX et en particulier Patrick Gros pour m'avoir accueilli dans leur équipe. Je remercie les membres et stagiaires de l'IRISA sans qui l'atmosphère n'aurait pas été aussi agréable durant ce stage (En particulier Brigitte Fauvet, Fabien Allard, Cédric Dufouil, Julien Lancia, Mickael Le Baillif, Cédric Motsch et les autres). Enfin, je tiens à remercier Toufik Ahmed pour m'avoir aidé à obtenir ce stage.

Table of contents

LIST OF FIGURES.....	5
ABBREVIATIONS	6
1. INTRODUCTION	7
2. CONTEXT.....	10
2.1. <i>Motivation</i>	10
2.2. <i>Project Management</i>	12
2.3. <i>My Contribution</i>	13
3. RELATED WORK	15
3.1. <i>Describing Multimedia Contents with Metadata Standards</i>	15
3.1.1. MPEG-21	15
3.1.2. MPEG-7	16
3.1.3. TV-Anytime	16
3.1.4. Hybrid approaches	18
3.2. <i>Multimedia Authoring Tools</i>	19
4. FROM THE UNIFIED METADATA MODEL TO THE DESIGN OF THE M-TOOL	21
4.1. <i>A Unified Model for Content-Descriptive Metadata</i>	21
4.2. <i>Building complex multimedia documents</i>	23
5. M-TOOL DESIGN AND IMPLEMENTATION	25
5.1. <i>Overall Architecture and Functionalities</i>	25
5.2. <i>M-Tool for TV-Anytime</i>	29
5.2.1. M-Tool GUI Configuration.....	29
5.2.2. TVA Editor	30
5.3. <i>M-Tool for MPEG-21</i>	26
5.3.1. Creating MPEG-21 Metadata.....	26
5.3.2. Communication Interface.....	28
5.3.3. Interactions with the M-Tool for TVA	28
5.4. <i>M-Tool API</i>	32
5.5. <i>M-Tool Search Engine</i>	32
7. CONCLUSION	36
REFERENCES.....	38
ANNEX A: TV-ANYTIME ELEMENTS	41
ANNEX B: SCENARIOS IN ENTHRONE	43
ANNEX C: TECHNICAL DESCRIPTION OF THE M-TOOL	44
C.1. <i>Form generator</i>	44
How to describe the form?	44
C.2. <i>Saving</i>	46
C.3. <i>Building DID structures</i>	49
The drawing panel.....	49
C.3. <i>Right click on icons</i>	50
C.4. <i>Double click on icons</i>	50
C.5. <i>Creating new elements</i>	51
New Schedule events or program information.....	51
New DIA and REL.....	51

List of figures

FIGURE 1: CONTENT-DESCRIPTIVE METADATA MANAGEMENT ARCHITECTURE	11
FIGURE 2: IMS/TVMS INTERACTION	12
FIGURE 3: MPEG-21 DIGITAL ITEM DECLARATION STRUCTURE	16
FIGURE 4: EXAMPLE OF <i>PROGRAM INFORMATION</i>	18
FIGURE 5: EXAMPLE OF <i>SCHEDULE EVENTS</i>	18
FIGURE 6: METADATA ORCHESTRATION IN THE ENTHRONE PROJECT	22
FIGURE 7: DESCRIBING COMPLEX MULTIMEDIA CONTENTS WITH MPEG-21	23
FIGURE 8: MAPPING FROM TV-ANYTIME TO MPEG-21	24
FIGURE 9: M-TOOL TVA CONFIGURATION	30
FIGURE 10: M-TOOL TVA EDITOR.....	31
FIGURE 11: M-TOOL DID EDITOR.....	27
FIGURE 12: M-TOOL COMMUNICATION INTERFACE.....	28
FIGURE 13: THE M-TOOL SEARCH ENGINE	33
FIGURE 14: M-TOOL SEARCH ENGINE RESULTS PAGE	34
FIGURE 15: CONFIGURATION OF THE M-TOOL SEARCH ENGINE	35

Abbreviations

<i>AQoS</i>	<i>Adaptation Quality of Service</i>
<i>CP</i>	<i>Content Provider</i>
<i>CS</i>	<i>Classification Scheme</i>
<i>DI</i>	<i>Digital Item</i>
<i>DIA</i>	<i>Digital Item Adaptation</i>
<i>DID</i>	<i>Digital Item Declaration</i>
<i>DII</i>	<i>Digital Item Identification</i>
<i>EPG</i>	<i>Electronic Program Guide</i>
<i>GUI</i>	<i>Graphical User Interface</i>
<i>IMS</i>	<i>Integrated Management Supervisor</i>
<i>IPMP</i>	<i>Intellectual Property Management and Protection</i>
<i>IST</i>	<i>Information Society Technologies</i>
<i>MD</i>	<i>Metadata</i>
<i>MDDB</i>	<i>Metadata Database</i>
<i>PVR</i>	<i>Personal Video Recorder</i>
<i>QoS</i>	<i>Quality of Service</i>
<i>REL</i>	<i>Rights Expression Language</i>
<i>SOAP</i>	<i>Simple Object Access Protocol</i>
<i>TVA</i>	<i>TV-Anytime</i>
<i>TVM</i>	<i>TV/Multimedia Processors</i>
<i>UED</i>	<i>Usage Environment Description</i>
<i>URI</i>	<i>Uniform Resource Identifier</i>
<i>URL</i>	<i>Uniform Resource Locator</i>
<i>XML</i>	<i>eXtensible Markup Language</i>
<i>XSL</i>	<i>eXtensible Stylesheet Language</i>

1. Introduction

The number of available digital contents is increasing over Internet and broadcast networks. Such a quantity of documents requires new ways to handle them. Besides, new services need tools to describe and organize these documents in an efficient and extensible way. These issues make metadata an interesting subject of research. Metadata, literally data about data, is information used to describe a wide array of multimedia contents from audiovisual contents to web pages. Metadata has become mandatory to cope with the increasing and huge amount of digital sources across the Internet or in firms/communities specialized in the storage of numerous documents such as INA (Institut National de l'Audiovisuel) [INA] or data warehouses. Metadata enables organizing these documents in logic clusters. A cluster is a group of documents sharing some common features. The clustering is a very present subject in the research community [JPL00]. This makes information retrieval and data-mining much more efficient. These clusters are built on account of an indexation which gathers documents by their high level information (also called semantic information), low level information and segmentation information. Low-level information describes the inner structure of documents (e.g. colors, shapes) whereas high-level information describes the content itself in a semantic way (e.g. title, synopsis). Low-level information cannot suit all kinds of digital contents and all kinds of search mode. For instance, a user searching images which contains a bird could find bothering to provide another image with a bird whereas input the single word "bird" would be much more convenient [FSN95]. So, high level information and manual inputs are essential for searching. The temporal segmentation consists in cutting a time-based document (e.g. audiovisual resources) in smaller parts called shots or chapters; this operation can be repeated to create a hierarchy of shots. The temporal segmentation is very useful for advanced search; the result of the search will be all the more accurate since audiovisual contents will be segmented in many shots. Each shot is then described with its own content-descriptive metadata. Metadata can be created directly with a text editor but it would be very daunting. Annotation refers to the act of adding metadata to a particular resource. Annotation is often performed on account of an authoring tool. There are manual, semi-automatic and automatic annotations. Semi automatic and automatic annotations mainly concern low-level information and temporal segmentation. High level information by nature is rather given by a manual input.

The overwhelming majority of metadata specifications uses the eXtensible Markup Language (XML) [XML]. The XML language permits to build hierarchical structures suitable for most types of documents and so facilitates advanced search. XML is often used with XML schemas [XMLS]. An XML schema is an XML language expressing rules to create XML documents. Most metadata standards are built on this language, among them MPEG-7, MPEG-21 and TV-Anytime. The primary goal of metadata is to manage the huge number of digital sources,

thus facilitating the search. However, metadata is employed for many other functions. Metadata is useful to give information about multimedia resources. For instance, information can be displayed on the client screen while playing an audiovisual content, as in the Télévision Numérique Terrestre (TNT); this information could be a TV-guide with the list of scheduled programs. Interactive interfaces are also built on account of metadata; these interfaces consist in gathering multiple mixed multimedia contents inside the same structure. Thus, metadata defines the spatial and temporal layout as well as hyperlinks. Examples of such interactive interfaces lie in DVD movies where images, texts, audio tracks, audiovisual sequences are combined. Another example of such interactivity could be when a character appears in a movie, a textual description of this character is displayed and a hyperlink is proposed to see a biography of this actor. Metadata is also used for content protection which restricts content usage to a particular user or group of users; it is particularly useful for service providers in a pay-per-view scenario. Finally, adaptation can be performed by using metadata; In this case metadata describes terminal capabilities, network characteristics and the way data has to be modified.

The emergence of broadcast networks for audiovisual contents (e.g. UMTS, EDGE or DVB) is inseparable from the outbreak of new services. Most of these services are directly dependant of extra information supplied by metadata. With the growing development of these networks, it is high time to propose tools that exploit the potential of these new technologies. Two trends are mainly identified in the digital broadcasting of audiovisual contents: On the one hand, the digital TV provides users with scheduled programs broadcasted according to a steady scheduling; on the other hand, the Video on Demand (VoD) provides users with a choice of programs available over a certain period of time. In parallel to these two ways of broadcasting, material means to receive digital-typed services are swiftly increasing namely with set-top boxes. A set-top box is an electronic device connected to an external signal source such as satellite dish, a cable or a DSL connection. It receives and converts digital signals to a content displayable on a client terminal. This content is as various as provided services could be: simple audiovisual contents, interactive multimedia documents including games and advanced DVD-like menu, internet web pages, videoconferencing, IP telephony and Video On Demand. These devices are often coupled with a personal video recorder (PVR) which is a consumer electronics device recording displayed contents (above all television) to a hard disk in digital format. This enables advanced features such as time shifting, pausing live TV, instant replay of interesting scenes, and skipping advertising. Digital television currently uses Electronic Program Guides (EPG) which is a set of technologies, often using metadata, that provides set-top boxes with basic information about current or/and upcoming multimedia contents. This EPG metadata is directly displayed on-screen for informative purpose. However, current EPGs are pretty limited and cannot answer new various scenarios. There is a crying need of metadata to deal with these mixed scenarios. As a result, we designed a new metadata model to create metadata able to cope with this evolution.

I developed an authoring tool, the M-Tool, which supports this metadata model. Section 2 gives the context and motivation for performing such works. Section 3 shows similar works and

put forward two standards used in the framework of the M-Tool, MPEG-21 and TV-Anytime (TVA). Section 4 explains our approach and highlights our metadata model. The M-Tool is described in Section 5. This description includes the M-Tool for TVA, the M-Tool for MPEG-21 and the M-Tool search engine. Finally, last sections describe the project management and future work.

2. Context

2.1. Motivation

This work was performed in the TEXMEX research team part of the INRIA Rennes research unit (IRISA) during my 6-month internship. The TEXMEX team is specialized in the efficient exploitation of multimedia documents. This includes indexation, data-mining, metadata management among others. This team is involved in several projects, among them the ENTHRONE European project where our goal was to define and build the content-descriptive metadata management module.

Among numerous initiatives that have recently emerged in the area of multimedia delivery and applications, ENTHRONE is an IST project which considers the provision of an integrated management based on the end-to-end Quality of Service (QoS) over heterogeneous networks and terminals. The ENTHRONE project proposes an integrated management solution which covers an entire audiovisual service distribution chain, including content generation and protection, distribution across networks and reception at user terminals. The ENTHRONE project investigates and develops an integrated solution to manage the functionality of various entities in the digital information distribution chain, from content/service generation to user terminals.

Such a digital information distribution chain requires mechanisms to efficiently manage, exchange and adapt distributed audiovisual resources. This is best achieved through the use of various metadata from content-descriptive metadata to QoS adaptation metadata. Metadata must describe the data sufficiently well as to be used as a surrogate for the data when taking decisions regarding description, adaptation and use of resources. In the MPEG-21 framework [ISOM21] adopted by the ENTHRONE project, metadata is considered as a transversal keystone of the ENTHRONE project. Metadata is a global resource for achieving interoperable and transparent access to multimedia resources from any type of client terminal or network and for providing an implementation to the Universal Multimedia Access (UMA) concept. Metadata describing Digital Items (DI), network, users, services, terminals, and subsystems configuration is then advantageously used in a distributed way throughout the whole architecture of the project [ENTD03][ENTD15]. Furthermore, metadata is concerned with a wide range of issues such as VoD and live broadcast scenarios and should answer different requirements.

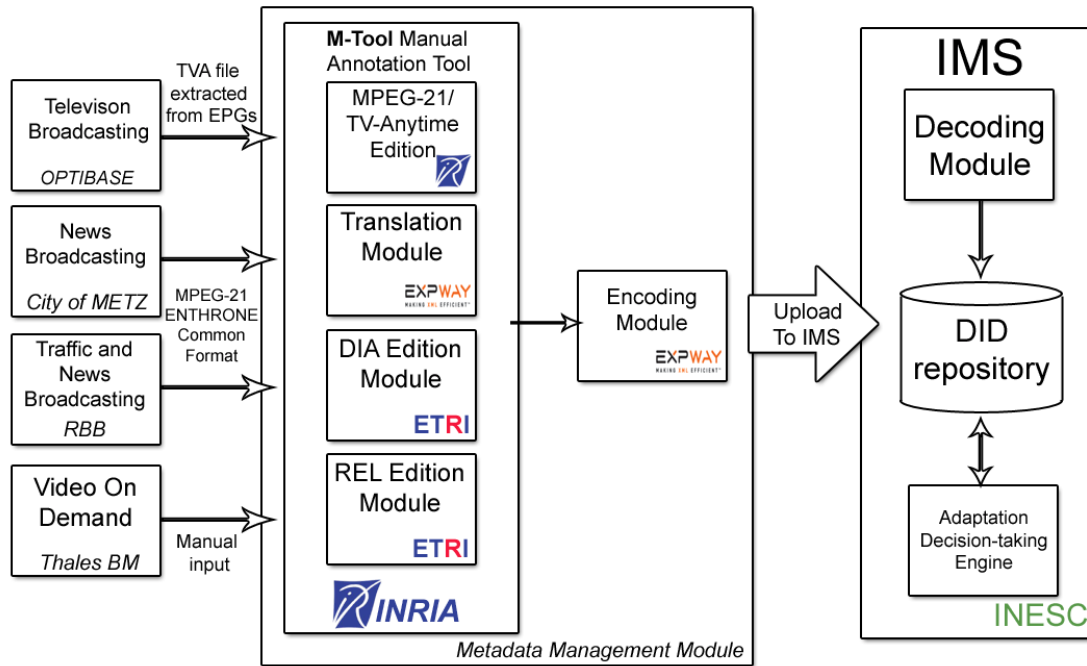


Figure 1: Content-Descriptive Metadata Management Architecture

Content providers in the ENTHRONE project have different needs as they do not provide similar services: Free TV broadcast, Video on Demand, Information services on public transportation, Pay-TV, distance learning... So, we had to design metadata model able to support this wide range of services. In the Figure 1, four content providers are represented: OPTIBASE TVM (TV broadcast), City of Metz (city news broadcast), Rundfunk Berlin-Brandenburg (traffic and news broadcast) and Thales Broadcast Multimedia (Video On Demand). In the ENTHRONE project, the M-Tool is designed to create and manage metadata information relevant to a particular resource at the server-side. So, content providers manage their metadata for their own resources. After metadata creation or update, the metadata management module sends this data to the Integrated Management Supervisor (IMS) using SOAP. ENTHRONE's Integrated Management Supervisor (IMS) is a service oriented architecture, following a distributed approach. The IMS is responsible for maintaining a metadata database (MDDb) that allows uploading and updating MPEG-21 DID. Besides, the IMS is the architecture which decides the type of adaptation to perform according to several facts including network conditions and client terminal features. The IMS configures TVMs (TV/Multimedia Processors) to adapt the stream as shown in Figure 2. This configuration is performed on account of metadata sent by the metadata management module. Any basic activity related to content treatment (encoding, filtering, multiplexing, remultiplexing, etc.) is provided by its associated TVM Processor. The TVM Processor takes as inputs, digital TV and/or Multimedia streams and processes them, according to the user specification and constraints, managed and controlled by the IMS. Digital streams include all types of data (Audio, Video, Private Data, IP Data, ...) broadcasted in a network (TV

Satellite, TV Cable, TV Terrestrial, Telecom, IP, ...). The IMS is acting between the user requests and the content descriptive information.

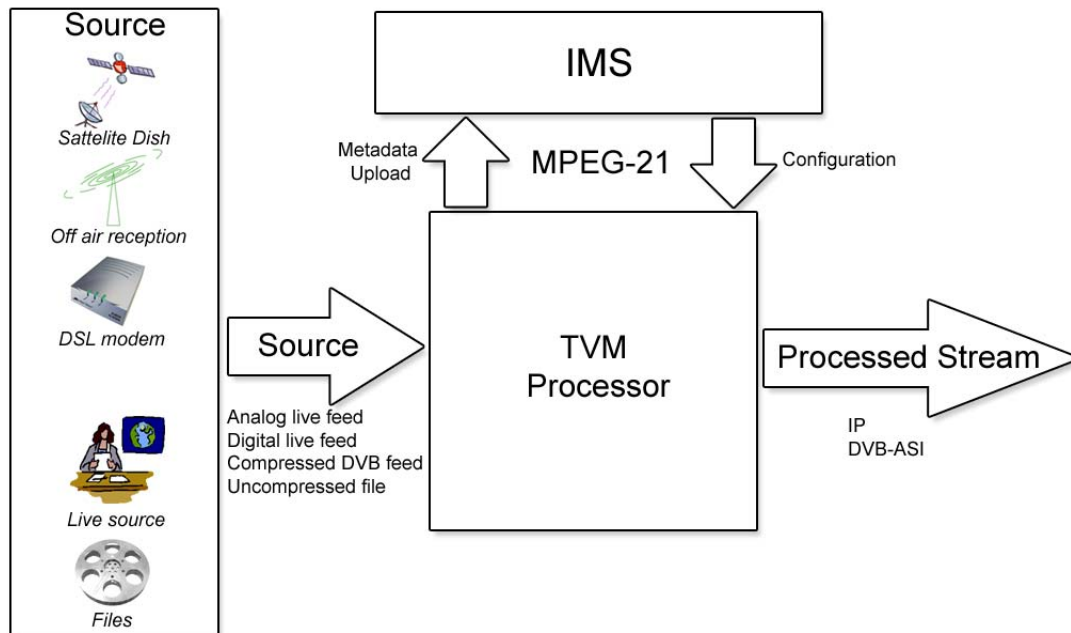


Figure 2: IMS/TVMs interaction

2.2. Project Management

The project consists of several work packages, each of them dealing with a particular aspect of the project. I mainly work on the Work Package 4 named “Content Generation”, its main goal being the development and implementation of the TVMs (TV and Multimedia Processors) for content generation and its associated metadata. Work packages are organized as follows:

- Work package 1: Project Management
- Work package 2: Overall System Requirements and Architecture
- Work package 3: Integrated Management Supervisor (IMS)
- Work package 4: Content Generation
- Work package 5: Content Protection
- Work package 6: Network Infrastructures
- Work package 7: End User Terminal Management
- Work package 8: System Integration, Test and Evaluation
- Work package 9: Standardization, Dissemination and Exploitation
- Work Package 10: Training Activities
- Work Package 11: Demonstration Activities

Deliverables are written to be submitted to the European Commission. We (INRIA) have leaded the writing of the Deliverable 15 (D15) [ENTD15]. This document represents the fifteenth

deliverable under WP4 entitled 'Metadata Authoring Tool TVM Processor' which aims at providing a detailed description of the architecture, management tools and applications developed to handle metadata flows throughout the ENTHRONE system. Besides, management reports were written to introduce our work and to report on the issues or deviations from the plan. These reports are internal and permit to facilitate the coordination between all teams.

Several meetings are organized to solve particular issues. The 28th of April, a meeting in Postdam, Germany was organized to discuss metadata related issues regarding flows and interfaces. The Athens meeting between the 9th and 11th of May dealt with the hearing preparation (for the European Commission) and demonstrators implementation. The Berlin meetings, scheduled on the 21st of July, focused on the demonstrator implementation.

2.3. My Contribution

I designed the M-Tool for TV-Anytime and the M-Tool for MPEG-21. I also created the first version of the M-Tool Search Engine which includes all but the configuration module. I did technical choices such as the C# language for the M-Tool and ASP.NET/IIS server for the M-Tool Search Engine. The choice of using C# was done because it is well-documented and numerous APIs are available. Moreover, the C# language is very easy to write and maintain since there is almost no consideration about memory allocations on contrary to C++. I also participated to the definition of the metadata model used in the ENTHRONE project.

I submitted a paper entitled "Enriching Multimedia Content Description for Broadcast Environments: From A Unified Metadata Model to A New Generation of Authoring Tool" [RJB05] to the IEEE ISM2005 conference, Irvine, California, USA, December 2005. This paper was written with Boris Rousseau (Engineer, IRISA) and Laure Berti-Equille (Associate professor, IRISA).

I submitted a short paper entitled "Stratégies de gestion et d'utilisation des métadonnées dans les réseaux de diffusion de contenus multimedia" [DJ05] to the MAJECSTIC conference, Rennes, France, November 2005. This paper was accepted and will be in the proceedings of this conference.

I wrote some parts of the Deliverable 15 entitled "Metadata Authoring Tool TVM Processor" (Communication interface, DID Editor, M-Tool GUI configuration, TVA Editor, M-Tool interactions between TVA and MPEG-21). I also wrote parts of management reports in the framework of the ENTHRONE project.

I wrote a report entitled "MPEG-21 and TV-Anytime in ENTHRONE" (June 2005) introduced to the ENTHRONE partners. I also wrote smaller reports to show the advancement and future work of the M-Tool ("Check In report" February 2005, "How to take into account all elements of the TV-Anytime specification" Mars 2005, "Modules overview" April 2005).

I created an internal website to introduce the work performed on the M-Tool. It contains screenshots, videos and descriptions of the M-Tool architecture. Partners were required to discover the M-Tool and to provide feedback and opinion about this work. As our partners will

have to use the M-Tool, we have to get as much feedback as possible to adapt the tool to their needs and to correct possible bugs.

Finally, I created HTML documentation and technical reports to help developing the M-Tool in the future.

In the next section, I present a state of the art of metadata standards for multimedia resources and different authoring approaches found in the literature and related standards.

3. Related Work

3.1. Describing Multimedia Contents with Metadata Standards

MPEG-21, MPEG-7 and TV-Anytime are established standards for defining multimedia contents. They answer to various needs and thus, they are complementary on several levels. MPEG-21 enables a hierarchical representation of heterogeneous contents and provides tools to protect and adapt these contents to the network and to client terminals.

3.1.1. MPEG-21

MPEG-21 [ENTSoA] aims at defining an open framework for the delivery and consumption of multimedia contents in heterogeneous conditions. In other words, the goal of MPEG-21 is to define a metadata model to support users to exchange, access, consume and manipulate Digital Items in an efficient, transparent and interoperable way. MPEG-21 [ISOM21] is based on two essentials concepts: the definition of a fundamental unit of distribution and transaction (the Digital Item or DI) and the concept of users interacting with Digital Items. The MPEG-21 specification is flexible and enables higher level functionality and interoperability by allowing the connection of the several parts of MPEG-21, the inclusion of other description schemes, etc. The DID [MPEG21-DID] represents a complete separation of metadata from its associated media resource. The DID specifications encompass the following features:

- DII: The Digital Item Identifier (DII) uniquely identifies Digital Items and parts thereof.
- Container: A container is a structure that allows grouping several items to form logical packages for easier transport and organization.
- Item: Items are declarative representations of Digital Items and are groupings of components that are bound to relevant descriptors.
- Component: A component is the binding of a resource to all of its relevant descriptors. Components can be considered as building blocks of items. Each component contains a resource.
- Descriptor: A descriptor associates information with the enclosing element. This information may be plain text or xml statement.
- Others: Condition and choice can also be expressed in MPEG-21.

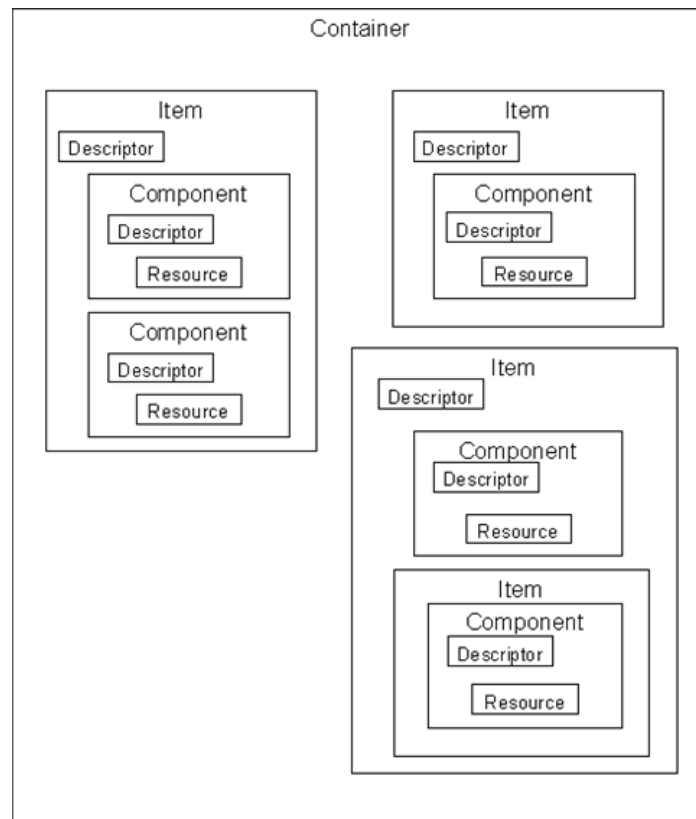


Figure 3: MPEG-21 Digital Item Declaration Structure

3.1.2. MPEG-7

MPEG-7 [MSS02] supplies a set of tools to describe a wide range of multimedia contents. Although MPEG-7 [ISOM7] covers a wide range of abstraction level, the MPEG-7 strength is inherent in the description of low level information such as texture, color, movement and position information. Many low-level features can be automatically extracted from the content (e.g., color histogram). As a result, it makes easier the management of huge amount of multimedia sources by proposing powerful tools for indexing and searching and by permitting automatic annotation.

3.1.3. TV-Anytime

In contrast, the TV-Anytime specification [TVA1, TVA2] provides very little low level information but provides a large panel of high level information or semantic information such as title, synopsis, genre, awards, credits, release information etc. Since TV-Anytime is initially dedicated to TV services, it handles the segmentation and the repeat broadcast of multimedia resources by the way of program. It is also interesting to note that TV-Anytime uses a sub-set of the MPEG-7 standard namely for low-level information (e.g. audio and video coding). . The TV-Anytime enables audio and video search, capture and playback of content. It also enables segmentation and indexing of that content. The TV-Anytime specification mainly consists of the following features:

- Content-descriptive Metadata can be divided in two category: the semantic description and the low-level description (Audio/Video attributes)
- Segmentation Metadata refers to the ability to create, access and manipulate temporal intervals of a particular audiovisual stream. This feature is useful for time shifted viewing.
- Consumer Metadata enables describing consumer related information such as action history and user preferences.

TV-Anytime features enable the search, selection, acquisition and rightful use of content on local and/or remote personal storage systems from both broadcast and online services. The TV-Anytime basic element is the *program*. A *program* represents the description of a part of an audiovisual resource. Each program has a semantic description and/or a low-level description both belonging to the *Program Information* element. Besides, a reference identifier called the *CRID* (Content Reference Identifier) enables linking content-descriptive metadata of a single program with its *Schedule Events* or *Program Locations*. Each *Schedule Event* matches with a broadcast moment and/or a part of the multimedia resource. Thus, by adding pointers inside the schedule event, a multimedia resource is easily cut in multiple programs. These pointers define the beginning and the end of the program inside the resource. They are useful for pre-recorded content to segment the resource making easier the navigation and enabling advanced search. A case in point is TV broadcast; some programs could be repeated such as the weather forecast. As all weather forecasts can be described in a same way, there is no need to create several programs in the TV-Anytime specification. A single program could be used to supply the description of several weather forecast programs as described in Figure 4.

```
<ProgramInformationTable>
  <ProgramInformation programId="crid://test_irisa/0">
    <BasicDescription>
      <Title><![CDATA[Weather Broadcast]]></Title>
      <Synopsis><![CDATA[the weather of tomorrow]]></Synopsis>
      <PromotionalInformation>Buy our Froggy – the weather forecaster</PromotionalInformation>
      <Keyword type="main">weather</Keyword>
      <ParentalGuidance>
        <mpeg7:ParentalRating href="urn:tva:metadata:cs:ContentAlertCS:2002:6.0">
          <mpeg7:Name>Alert not required</mpeg7:Name>
        </mpeg7:ParentalRating>
      </ParentalGuidance>
      <Language type="original">fr</Language>
      <CreditsList>
        <CreditsItem role="urn::mpeg:mpeg7:cs:RoleCS:2001:PERFORMER">
          <PersonName>
            <mpeg7:GivenName>Emile</mpeg7:GivenName>
            <mpeg7:FamilyName>Dupont</mpeg7:FamilyName>
            <mpeg7:Title>M.</mpeg7:Title>
          </PersonName>
        </CreditsItem>
      </CreditsList>
    </BasicDescription>
    <AVAttributes>
      <AudioAttributes>
        <NumOfChannels>2</NumOfChannels>
      </AudioAttributes>
      <VideoAttributes>
```

```

        <HorizontalSize>360</HorizontalSize>
        <VerticalSize>200</VerticalSize>
    </VideoAttributes>
</AVAttributes>
</ProgramInformation>
</ProgramInformationTable>

```

Figure 4: Example of *Program Information*

A schedule event is used for each broadcast of this program. Multiple schedules could also be used to specify several locations inside the multimedia resource. Figure 5 illustrates these two cases; the two schedule events have the same *CRID* and so belong to the same program (shown in Figure 4). The *ProgramURL* element contains the URL of the multimedia resource and two pointers indicating the beginning and the end of this program in seconds (programURL;beginning;end). The *PublishedStartTime* and *PublishedEndTime* elements give the broadcast date and time. They could also be useful for Video on Demand to specify periods of availability of a particular resource.

```

<ProgramLocationTable>
  <Schedule serviceIDRef="defaultService">
    <ScheduleEvent>
      <Program crid="crid://test_irisa/0"/>
      <ProgramURL>16-11-2005.mpg;1556;2005</ProgramURL>
      <PublishedStartTime>2005-11-16T10:00:00+00:00</PublishedStartTime>
      <PublishedEndTime>2005-11-16T10:05:00+00:00</PublishedEndTime>
    </ScheduleEvent>
    <ScheduleEvent>
      <Program crid="crid://test_irisa/0"/>
      <ProgramURL>16-11-2005.mpg;4560;5000</ProgramURL>
      <PublishedStartTime>2005-11-16T14:00:00+00:00</PublishedStartTime>
      <PublishedEndTime>2005-11-16T14:05:00+00:00</PublishedEndTime>
    </ScheduleEvent>
  </Schedule>
</ProgramLocationTable>

```

Figure 5: Example of *Schedule Events*

3.1.4. Hybrid approaches

As every metadata specification answers to needs matching specific requirements, they cannot perfectly meet all requirements. Therefore, hybrid approaches have been designed to adapt to particular situations. These hybrid approaches associate complementary features of multiple standards to gain flexibility. Indeed, specific requirements imply refining these standards. Durand et al. [DKL05] encapsulate MPEG-21 inside TV-Anytime to realize a service description for scalable interactive TV. The *Madeus* [JLR98] model associates the MPEG-7 with a SMIL-like mechanism to create complex multimedia documents. In addition to interactivity, it deals with the adaptation of documents to the current context including user capabilities and preferences, physical location, network and system resources. Other standards are especially designed to create interactive multimedia presentation such as the *MHEG standard* (Multimedia and Hypermedia Information Coding Experts Group) [ME95]. This standard is not based on XML but on SGML (Standard Generalized Markup Language) and ASN.1 (Abstract Syntax Notation One).

We choose a hybrid approach for the metadata model used in the M-Tool. This choice is justified in Section 4

3.2. Multimedia Authoring Tools

Multimedia authoring tools are designed to achieve multiple purposes which can be amounted to browsing and visualizing, analyzing, segmenting, annotating and integrating. The common purpose of most authoring tools is the capacity of browsing and visualizing the metadata together with the multimedia content. Some authoring tools enable analyzing multimedia contents to provide users with information making easier the temporal segmentation. From these analyses, semiautomatic or automatic scene boundary detection can also be performed as in the *Video Retrieval and Sequencing System* (VRSS) [CR95] which semi automatically segments the video for later retrieval. Temporal segmentation can also be done manually by specifying the beginning and the end of each segment as in media production applications such as *Adobe Premiere* [ADO05]. In addition to segmentation information, authoring tools enable adding semantic information to the content description. Finally, the integration authoring tools consist in creating structured multimedia contents by establishing temporal synchronization, spatial layout and hyperlinks to bind multiple contents in advanced multimedia presentations. Most authoring tools combine several of these purposes, often fostering one aspect. Most of these tools focus on the temporal segmentation aspect disregarding the semantic annotation one. Thus, the IBM's MPEG-7 authoring tool called *VideoAnnex* [IBM02] supplies automatic temporal segmentation of audiovisual contents but only enables the semantic annotation of four elements: events, static scenes, key objects and keywords. In the same way, the *family video archive* [AGL03] proposes advanced browsing and temporal segmentation features however only three kinds of semantic annotation are available: date, freeform text, and a metadata tag. Numerous other examples could illustrate that most current authoring tools only work at a low syntactic level. On the contrary, the semantic annotation aspect constitutes an important issue in the M-Tool.

Advanced browsing features have been developed to perform searching. Some researches focus on video feature based search; the *ViMeta-Vu* [TCJ01] authoring tool put forward a query system based on automatic segmentation and analysis of low-level information. They argue that text annotation is too subjective to build reliable and accurate query methods whereas visual features are much more objective and so well-adapted for query methods. On the contrary, the authoring tool *Silver* [MCS01] uses semantic information from the *Informedia Digital Video Library* [HS95] to perform searching. *Informedia* provides titles and textual transcripts of audio tracks for each segment of audiovisual contents. However, *Silver* only allows users to search words in a single free form box in a Google-like fashion and the non-structural textual transcript is not adapted for advanced search. On account of extensive content-descriptive information created with the M-Tool, we plan to design efficient advanced search strategies.

By providing a full and comprehensive description solution for multimedia contents, the MPEG-7 standard has emerged as the primary choice for mono-media approach in numerous

authoring tools [RSK02, IBM02, TR03]. Conversely, there is almost no authoring tool relying on TV-Anytime. I only notice a single tool from *ETRI* (Electronics and Telecommunications Research Institute).

In contrast to the mono-media approach, interactive hypermedia authoring tools endeavor to integrate several multimedia contents inside a same structure. *Enikos* [ENI05] designed a MPEG-21 authoring tool, called *DIEditor*, allowing users to link multiple resources inside a MPEG-21 structure. However, this *DIEditor* does not enable temporal and spatial segmentation; moreover, resources descriptions are restricted to a freeform text. The *Mdefi* authoring tool [TR03] combines two models: the MPEG-7 standard and a SMIL-like multimedia model. MPEG-7 is used to describe audiovisual resources and the SMIL-like model to integrate these resources inside a complex multimedia presentation. Owing to the specificity of our requirements in the framework of the ENTHRONE European project, no current solution could answer to our needs (see the various scenarios given in Annex B). As a consequence, our approach was to combine and unify the TV-Anytime and the MPEG-21 standards. MPEG-21 provides content protection, network adaptation, client terminal adaptation and a structure to link several multimedia contents while TV-Anytime provides content-descriptive metadata and temporal segmentation. This approach is detailed in the next section.

4. From the Unified Metadata Model to the Design of the M-Tool

In this section, I describe the metadata model used in this project and I show its flexibility through the representation of complex multimedia documents.

4.1. A Unified Model for Content-Descriptive Metadata

TV-Anytime and MPEG-21 are complementary and cover the requirements specified by our partners (RBB, City of Metz, France Telecom, Thales Broadcast and Multimedia, Optibase):

- Free TV broadcast
- iTV-MHP Service
- On demand video services to mobile devices with QoS over cooperating DVB-T / GPRS
- Broadcast TV service to mobile devices over DVB-T
- information services on public transportation
- Individual on demand information
- Video on Demand
- Multimedia content download
- Pay-TV
- Distance learning

Table B.1 in Annex B gives more details about the ENTHRONE scenarios.

Although initially dedicated to TV services, the TV-Anytime specification enables the description of all these services and provides powerful tools to perform advanced search on a huge amount of heterogeneous contents. Our approach is that TV-Anytime alone is not sufficient as it neither defines transport mechanism nor the hierarchical structure to represent a mix of multimedia contents. The use of MPEG-21 adds the flexibility mandatory to adapt to these various services and situations. MPEG-21 enables the hierarchical representation of multimedia contents which is useful to create advanced and interactive multimedia contents. Every resource is described using the TV-Anytime standard and then, resources are gathered and synchronized inside a MPEG-21 structure that includes: 1) network quality of service and adaptation metadata and 2) right protection metadata.

1) MPEG-21 Digital Item Adaptation (DIA) metadata [ENTSoA] describes how multimedia contents should be adapted to network characteristics and client terminals. DIA metadata consists of adaptation QoS (aQoS) and Usage Environment Description (DIA UED) metadata. The aQoS provides the required information to select optimal adaptation parameters. The description of terminal capabilities given by DIA UED metadata is primarily required to satisfy consumption and processing constraints of a particular terminal. DIA UED also specifies network

characteristics in terms of network capabilities and conditions, including available bandwidth, delay and error characteristics.

2) MPEG-21 Right Expression Language (REL) [ENTSoA] specifies whether a given group of people can perform a given right upon a given resource under a given condition. MPEG-21 Intellectual Property Management and Protection (IPMP) manages rights and intellectual property of a specific resource.

The metadata description schema is illustrated in Figure 6; TV-Anytime metadata (1) is encapsulated inside the MPEG-21 structure and each MPEG-21 descriptor contains REL, DIA or TV-Anytime information. Metadata is seen as a global resource that will be used in a distributed way throughout the architecture of the project and will feed or be generated by different component of the ENTHRONE system. Content-descriptive metadata describes a resource (e.g. title, synopsis, etc.) in addition to dedicated fields for content providers' specific needs and with potentially all the fields proposed by TV-Anytime. Right protection metadata (2) manages rights and intellectual property of the specific resource (REL and IPMP). Network QoS metadata (3) is information necessary to adapt content delivery with network and QoS parameters (DIA). User metadata (4) is information on the end user such as preferences, interest, etc. Terminal metadata (5) summarizes terminal capabilities such as display resolution. Content providers (CPs) and subsystems configuration allow content providers and ENTHRONE subsystems metadata (6) to be configured with parameters and commands.

This model is well adapted to enrich content-descriptive metadata and to integrate annotated resources inside the MPEG-21 structure for various broadcast scenarios.

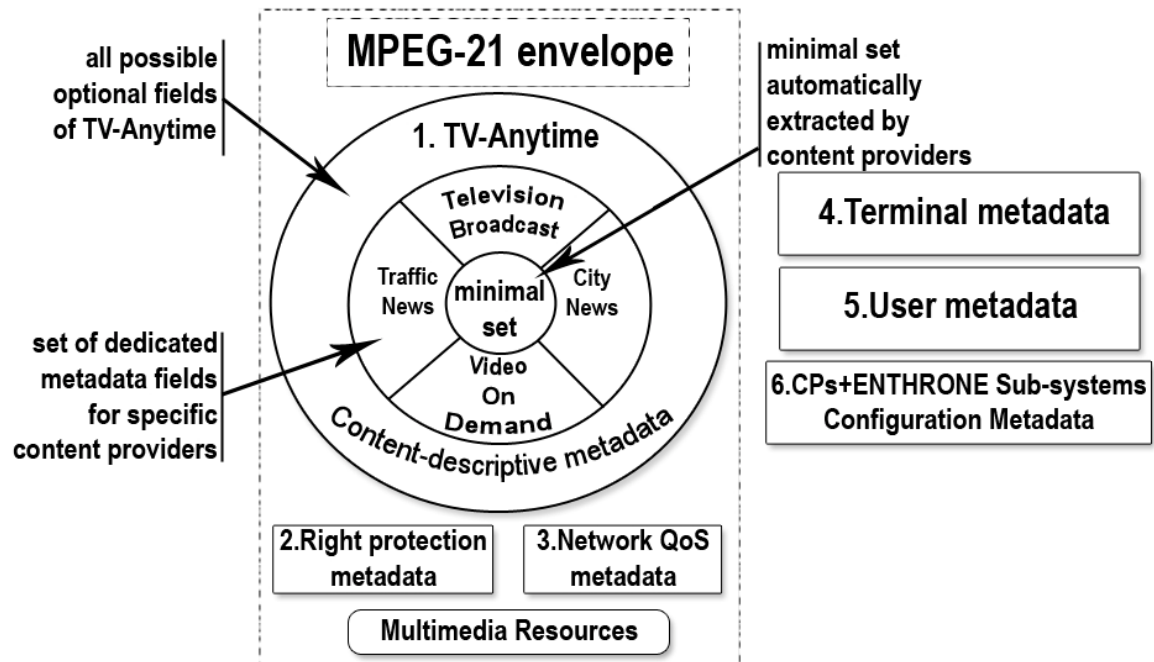


Figure 6: Metadata orchestration in the ENTHRONE project

4.2. Building complex multimedia documents

A complex multimedia document consists of several mixed resources. A case in point is a web page which consists of texts, images, audio and video contents, flash and many other elements. In Figure 7, a complex multimedia content (a web page) consisting of a text and an audiovisual content is translated to the MPEG-21 structure. Figure 7 shows one way to perform this translation. In this example, two *items* are built, one for the text and the other for the audiovisual content. An *item* is a group of synchronized *components* which describe a single resource. An *item* could consist of a video resource synchronized with an audio resource or only of a text resource as in our example. The movie *item* contains two *components* matching the audio and the video component of the movie. *Descriptors* are included in every element of the MPEG-21 structure. They apply to all hierarchically lower elements. For instance, the web page *container descriptor* contains an overall description of the web page. Within the text component, several programs could be defined. In this case, a program would be a paragraph for instance (e.g. introduction, development or conclusion) and the pointer would be the line number. The resource could be a text file. Within the video *item*, there are two *components*, one for the video and one for the audio. The video *item* descriptor contains an overall description of the video.



Figure 7: Describing complex multimedia contents with MPEG-21

The creation of a final MPEG-21 file is mainly performed in two steps:

- Annotation of every resource (that will imply the creation of one TV-Anytime file for each resource)
- Creation of the MPEG-21 structure

Figure 8 gives a first illustration of our two tools, the M-Tool for TV-Anytime and for MPEG-21; in a first stage, each resource is described on account of the M-Tool for TV-Anytime

and in a second stage, the M-Tool for MPEG-21 gathers these descriptions within a single DID. These steps could also be performed in a more simultaneous way since some modules of the M-Tool for TV-Anytime are integrated in the M-Tool for MPEG-21. As the assembly of the MPEG-21 structure could be done in different ways, I had to provide users with tools facilitating this assembly while letting them enough freedom to build their own structure. Automatic and semi-automatically mapping are also considered.

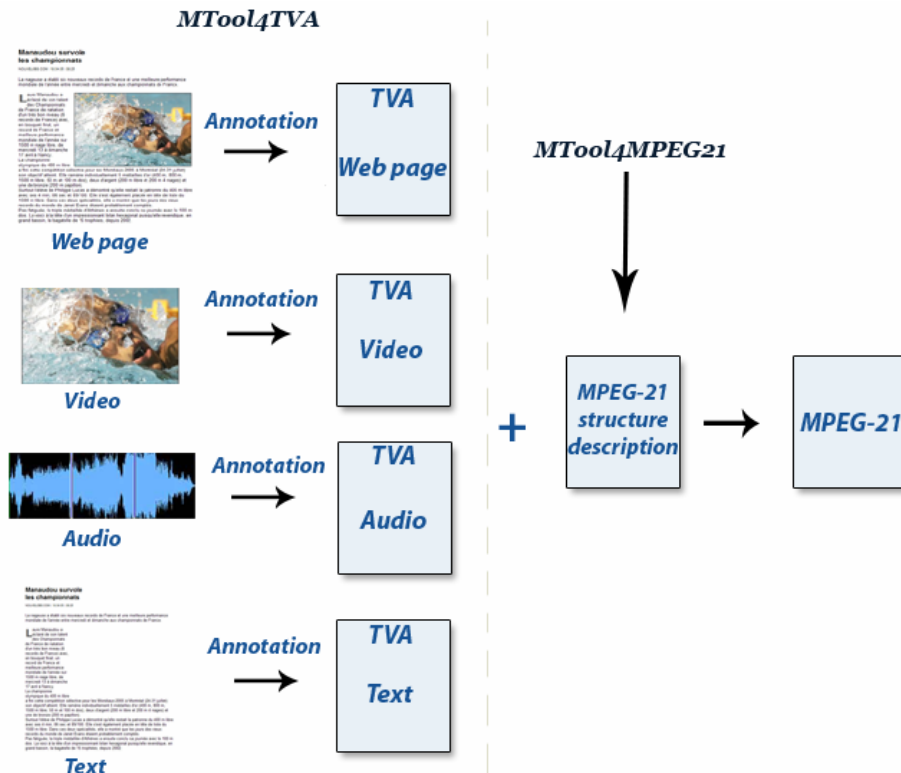


Figure 8: Mapping from TV-Anytime to MPEG-21

In the next section, I describe the main functionalities of our metadata authoring tool divided in two main applications that can be either independently used or combined to create a MPEG-21 file containing TV-Anytime (TVA) information: the first application consists in authoring TVA metadata and the second consists of authoring MPEG-21 digital items. These applications are designed to be the core of the metadata management module.

5. M-Tool Design and Implementation

The ENTHRONE Metadata Tool (M-Tool) is a prototype that offers the functionalities required to create or manage metadata information relevant to a particular resource. Its aim is to provide the end-user and content provider with tools that enable annotating, browsing, maintaining, monitoring and querying ENTHRONE content-descriptive metadata. The motivation for developing the M-Tool is to address a wide range of issues regarding metadata database-backed management and digital resources manipulation under the MPEG-21 framework. It is designed to help users in annotating and authoring multimedia resources. In other words, this application allows users to create objects' relationships to each other in order to enrich multimedia resources with metadata both formatted and stored in XML or in a relational database. This type of service is useful for applications that present a mixture of textual, graphical, and audio data. The M-Tool is a flexible tool adapting to various conditions of use: the M-Tool is a modular authoring tool (the user chooses displayed areas and their sizes) and configurable (the user chooses editable information on account of a profile edition module).

The M-Tool has been developed with Visual Studio .Net 2003 and consists of four projects:

- M-Tool for MPEG-21
- M-Tool for TVA
- M-Tool Search Engine
- M-Tool deployment project

The M-Tool deployment project generates a Microsoft Installation package (.msi). Such project facilitates the application's installation by automatically adapting the M-Tool to the current computer environment. In this way, the M-Tool only requires Microsoft Windows XP operating system and the .Net framework version 1.1 to be installed.

5.1. Overall Architecture and Functionalities

The M-Tool is a prototype composed of two applications providing functionalities to respectively handle MPEG-21 and TV-Anytime metadata. Basically, the M-Tool gets manually or automatically extracted metadata from various CPs of the ENTHRONE system. Once executed, the metadata document is analyzed and visualized using the relevant (MPEG-21 or TV-Anytime) Graphical User Interface (GUI). Users annotate, convert and encode this metadata and then, the updated file is uploaded to the IMS in the DID format. It mainly consists of two modules: M-Tool for MPEG-21 and M-Tool for TV-Anytime. These two modules are complementary inside the ENTHRONE framework. They share common functionalities such as open, create, modify files in their respective formats. The M-Tool for MPEG-21 is the central tool. It deals with communication functionalities to receive files from content providers and send them to the IMS. It also calls TV-Anytime modules to integrate TV-Anytime content-descriptive

metadata inside DIDs. Being the central tool, it enables running specialized tools (for TV-Anytime, REL or DIA) to edit a particular descriptor within a MPEG-21 structure or to create a new TV-Anytime, REL or DIA file.

5.2. M-Tool for MPEG-21

The M-Tool for MPEG-21 serves as the keystone to the prototype multimedia authoring tool as it centralizes all downloads of manually or automatically extracted metadata from various content providers. Once downloaded, these metadata files are locally stored and its list is displayed to the user. The user is then allowed to select one or more metadata document which is analyzed and visualized using the relevant (MPEG-21 or TV-Anytime) graphical user interface (GUI). Users have the possibility to create a new MPEG-21 document, edit, delete, convert or send this metadata document to a specific (local or external) metadata database.

5.2.1. Creating MPEG-21 Metadata

The M-Tool for MPEG-21 is a DID editor and generator. The generation of the DID structure is achieved thanks to a graphical representation of the MPEG-21 structure (Figure 9, area 1). The user can drag and drop relevant icons (Table 1) from the GUI. On this event, the M-Tool detects the type of the dropped component. These elements consist of two kinds; MPEG-21 structure elements (Container, Item and Component) and descriptor elements (TVA schedule event, TVA program information, REL and DIA metadata and Free Text). A DID is edited or generated by inserting/deleting or modifying these elements inside MPEG-21 descriptor.






<i>Icon</i>	<i>Element Name</i>	<i>Called tool</i>
	TV-Anytime program information	M-Tool for TV-Anytime (Program Information module)
	TV-Anytime Schedule Event	M-Tool for TV-Anytime (Program Location module)
	DIA and REL	ETRI REL generator ETRI DIA generator
	Free Text	M-Tool for MPEG-21 (Free Text module)
	Structure elements: Container, Item, Components	M-Tool for MPEG-21 (Graphical Editor module)

Table 1 : Dragged and dropped elements

Basic information about descriptors is provided by the graphical representation: the type of descriptor (DIA, REL, Program Information, and Schedule Event), the type of program information (general information or only Audio/Video attributes), the type of program (root or sub program) and the *CRID*. The interface allows users to perform actions on each descriptor. These actions are displayed by right clicking on descriptors inside the graphical representation. Then, the appropriate tool is called to edit the selected descriptor. TVA program information and TVA schedule event are edited by a restricted part of the M-Tool for TVA.

The tree view (3rd area in Figure 9) is complementary with the DID representation; it gives a more detailed view of all descriptors. By selecting a descriptor in the top tree view, the content of this descriptor is displayed in the bottom tree view. The DID structure can also be directly edited by clicking inside the top tree view and selecting an option in the context menu. The TVA integration module (4th area in Figure 9) enables integrating TV-Anytime elements (Program Information or Schedule Event) from a TV-Anytime file inside the DID structure on account of a drag-and-drop method.

The Free Text element is used to add either plain text or any xml language inside a MPEG-21 descriptor. This element is proposed to make the M-Tool a universal MPEG-21 authoring tool. In this way, this tool can be used out of the scope of the ENTHRONE project and could be useful for further extensions of the ENTHRONE project.

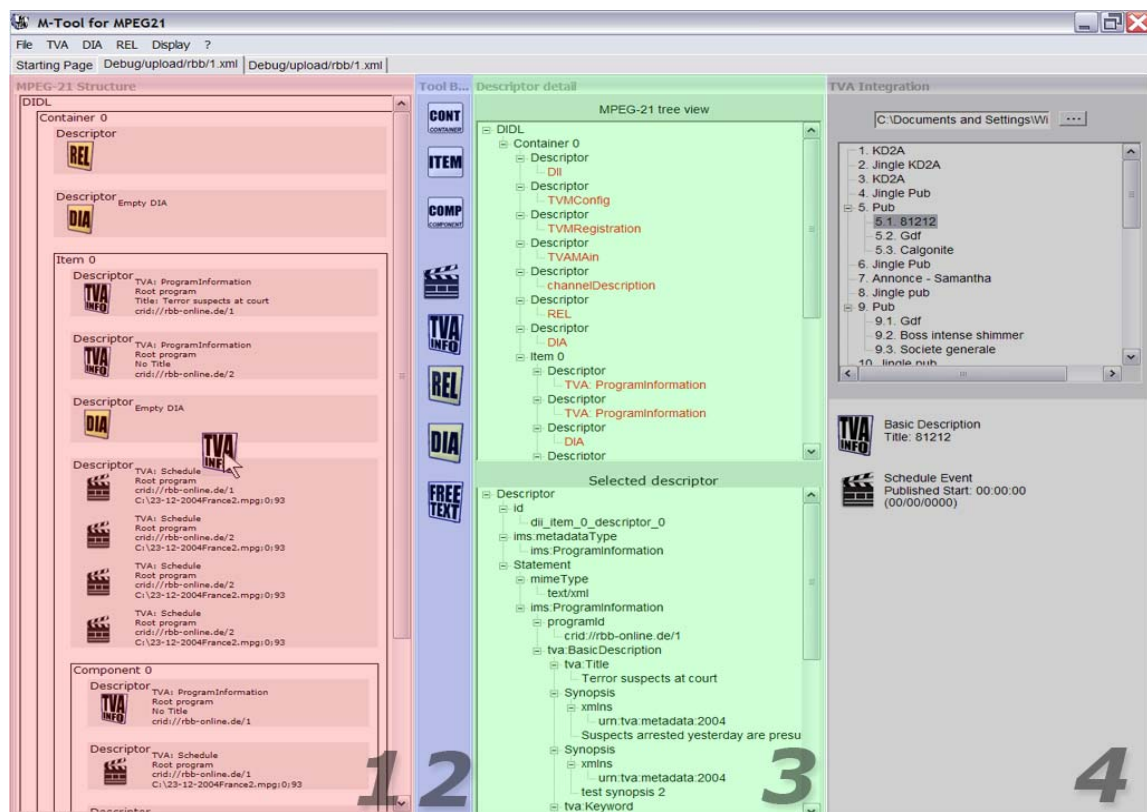


Figure 9: M-Tool DID Editor

5.2.2. Communication Interface

The M-Tool for MPEG-21 allows downloading metadata documents from various input sources based on the interfaces provided by Content Providers. Once downloaded, these files are locally stored. A list of these files is displayed and the user interacts with them in several ways:

- “Edit” action runs the adapted modules according the type of files (MPEG-21 or TV-Anytime).
- “Delete” action deletes the file stored locally.
- “Convert” action will convert the file either to MPEG-21 or TV-Anytime.
- “Send to IMS” action will send the selected file(s) to the IMS.

When a user requests to download a TV-Anytime or MPEG-21 document from content providers, the M-Tool invokes available remote servers to check whether TV-Anytime metadata could be retrieved. If so, the TV-Anytime metadata document is downloaded (HTTP Get or SOAP get request).

Finally the DID document is sent to the IMS via a SOAP interface through the openSession, uploadDID and closeSession methods.

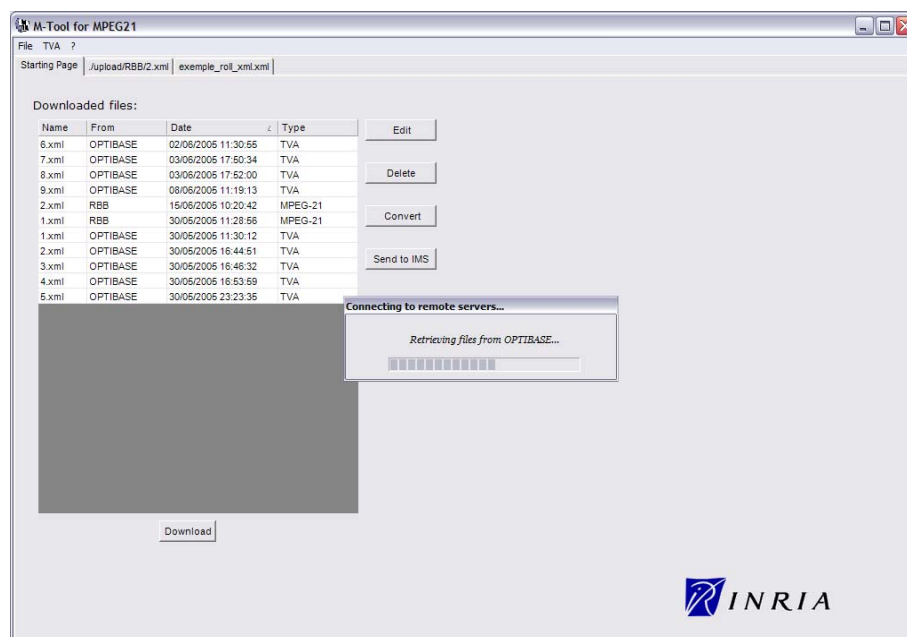


Figure 10: M-Tool communication interface

5.2.3. Interactions with the M-Tool for TVA

The first way to call the M-Tool for TVA consists in editing a particular descriptor (program information or schedule event) inside the DID graphical representation of the M-Tool for MPEG-21. Then, a form matching the user profile is displayed and program information can be directly enriched. The fourth area in Figure 9 enables loading a TV-Anytime file; then each program information and location can be dragged and dropped inside the DID graphical representation,

performing the integration of this TV-Anytime file. The third way (not yet implemented) will allow users to generate a MPEG-21 file from multiple TV-Anytime files in a semi and fully automatic way. Once TV-Anytime files were edited in the full version of the M-Tool for TVA, an interface will enable the linkage of these files in a unique DID structure. The M-Tool for TVA can also be directly called by selecting “Run M-Tool for TVA” in the M-Tool for MPEG 21 menu.

5.3. M-Tool for TV-Anytime

The M-Tool for TV-Anytime has functionalities to load, annotate, enrich or create a new TV-Anytime document while keeping any edition compliant with the schema and in synchronization with the audiovisual resource. It could be executed as a standalone application and is composed of two main modules: the configuration module and the TV-Anytime editor itself.

5.3.1. M-Tool GUI Configuration

The configuration module (Figure 11) allows user to select relevant TVA fields depending on their requirements for annotation. The M-Tool actually enables users to choose between over 90 TVA fields (e.g. genre, parental guidance, audiovisual attributes, etc) to adapt the annotation areas to the selected elements (Figure 12 area 1). Upon selection of one of the main elements (package) not only are the mandatory fields selected by default, but also a link (“Details”) permits personalizing selection of its optional sub-elements. Information of the number of elements per item is shown e.g. [1/5] meaning one out of five sub elements have been selected.

The selection is recorded as a profile, loaded automatically for future use of the M-Tool. Modifications to the current profile are possible through an option in the menu. Additionally, this functionality is flexible enough to allow easy addition of new elements to this set of TVA fields.

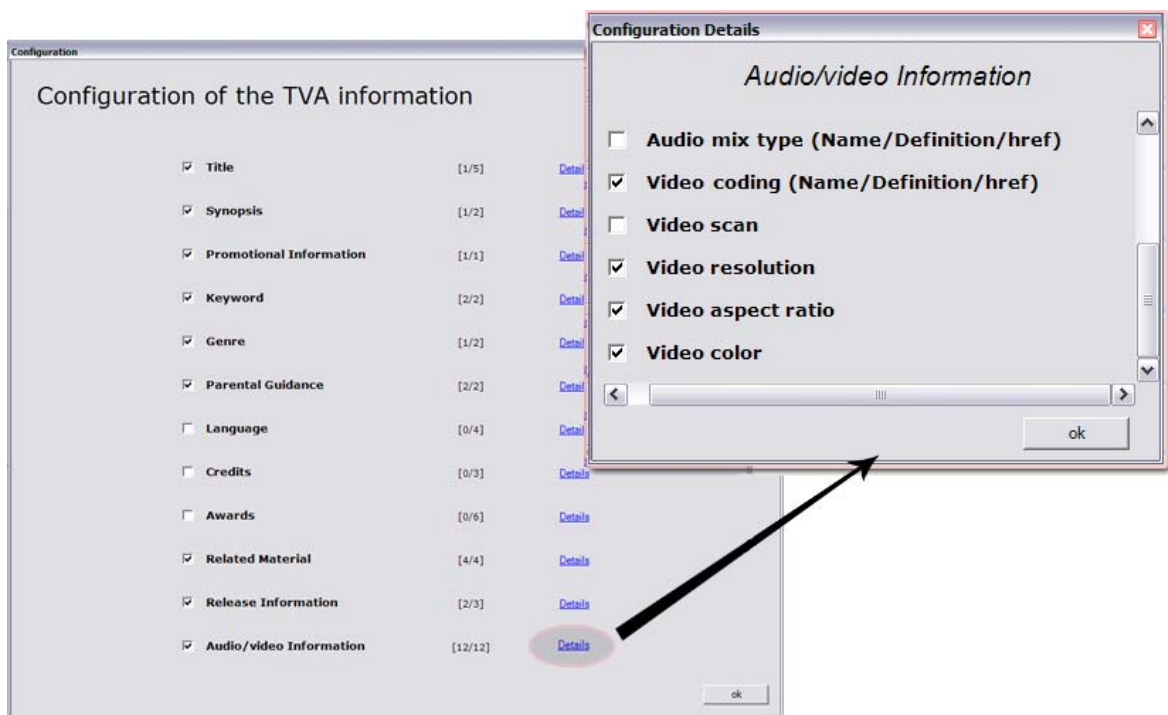


Figure 11: M-Tool TVA Configuration

5.3.2. TVA Editor

The Annotator is a form consisting of several editable fields as requested by the user with the configuration module and taken from the profile. The annotator is divided into two sections: *Program Information* (area 1) and *Program Location* (area 2) of the current program as shown in the above figure. As for the TV-Anytime specification, *Program Information* describes the resource in terms of title, synopsis, keywords, etc whereas *Schedule Event* represents a single instance of a program schedule or availability. The same program can be shown several times within the same day; as a result the same description can point to several schedules. The annotator supplies functionalities to edit, delete and add programs, *Schedule Events* or subprograms.

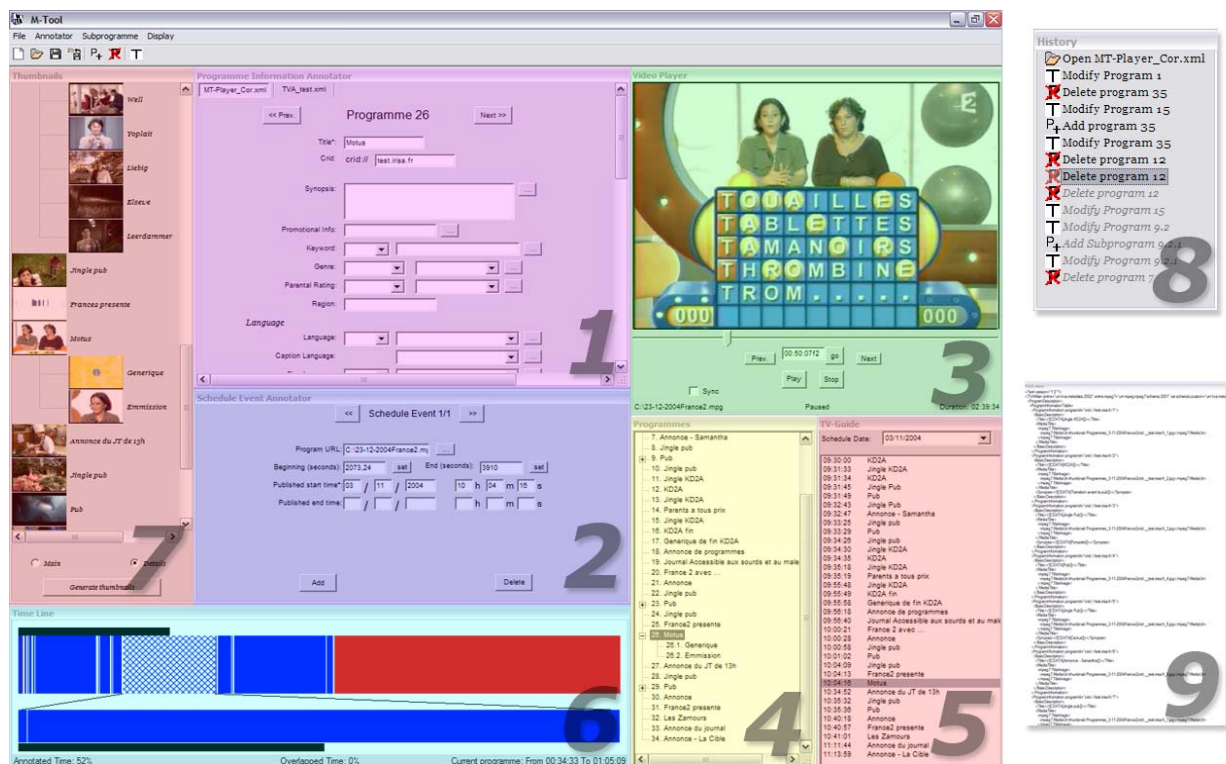


Figure 12: M-Tool TVA Editor

The functionalities for browsing, visualization and edition are composed of 7 sections:

- The media player (area 3) allows users to browse inside the audiovisual content with relevant action buttons and a track bar. This player is synchronized with all other modules to make sure that the associated metadata is displayed at the same time.
- The creation of subprograms (area 4) enables to separate a program of an audiovisual content in several sections like in DVD movies. Subprograms are represented by a tree view. Each node consists of the number of the program (e.g. 1. or 4.5.2.1.) plus the title of the program. This module enables browsing and addition of other subprograms.
- A TV-guide visualization (area 5) was designed to sort programs by schedule date. For each date, it shows programs titles and schedule times contained in the current XML file. Selecting one accesses the program directly.
- The time line (area 6) shows the different programs in a timeline that also allows in root programs and the first level of sub-programs. The first level, the highest and widest one, displays root programs (1., 2., 3., ...) and the second level, below the previous one, displays the first level of sub-programs of the current root program (1.1., 1.2., 1.3., ...). A color caption shows annotated region (in blue), non-annotated region (in red), overlapping programs (in violet and blue patchwork), current program or sub-program (in white and blue patchwork). When the mouse is over the

mosaic, information about the program is displayed (name, duration, bounds, number of *Schedule Events*...).

- A thumbnails view of each program and sub-programs is displayed in the area 7. Browsing is permitted through the selection of a thumbnail.
- The text mode edition allows users to edit a TV-Anytime file in a text editor in the same way as in the XmlSpy software.

The M-Tool for TV-Anytime allows edition of multiple files at the same time. There are 2 default display modes according to the fact the audiovisual content is available or not. Besides, the interface is easily modifiable (hiding, moving, maximizing modules...). Thus, modules and their sizes adapt to the user and the current TV-Anytime file. The History module (area 8) keeps track of events generated by the user on particular areas of the GUI in order to provide advanced undo/redo functions. This module displays the list of actions performed by the user on the current file; then, the user can click on any actions to go back to the selected action. This facilitates accurate annotation while bearing in mind that human errors can easily occur.

5.4. M-Tool API

The M-Tool API is the core of our authoring tool. This API has to be flexible and efficient enough for storing and displaying an extensible quantity of information. As a result, three levels of backups are considered in the M-Tool framework. The first level consists of *ArrayLists* (*ArrayList* class in C# or Java) of programs; the second level is the *XmlDocument* object and the third level is the physical level, the file. The second and third levels can be considered as a single level since they are always synchronized (e.g. when saving the second level, the third level is saved too). So the first level is called the *cache* and the last level, the file. This structure was chosen for its flexibility (namely for the form generation) and the enhancement of speed which follows; indeed, the cache adapts its size to the profile and selected TV-Anytime fields. On the contrary, the *XmlDocument* object is a more complex structure where all fields are loaded. The M-Tool API is enough flexible to be used for other standards since most of the fields are easily redefined. For further details, a part of the M-Tool API is described in the ANNEX C.

5.5. M-Tool Search Engine

The M-Tool search engine aims to perform complex queries on MPEG-21 files on account of a full text search and Xquery [XQU05]. Although the M-Tool Search Engine is at its very first stage of development, some functionalities have already been implemented. This includes querying TV-Anytime files and configuring the search method.

A search engine was developed to query TV-Anytime files on account of the DotLucene API [LUC05]. DotLucene is a Full Text Search Engine. A primary function to create an efficient search engine is to index beforehand all stored documents. This indexation is done by DotLucene. However, DotLucene supplies no function to extract information from XML files. So, another API is used to fill this function: The Xml Path Language (Xpath) [XPA99] is a language for

addressing parts of an XML document and it is useful for the parsing function of the M-Tool Search Engine. The results of the indexation is then stored in databases for later retrieval.

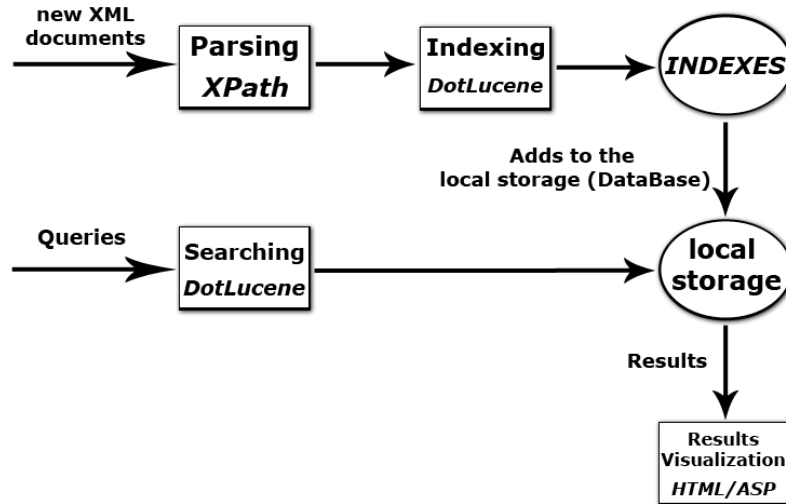


Figure 13: The M-Tool Search Engine

The search function is also performed by the DotLucene API which gives tools to search inside indexes built from files: DotLucene supplies advanced query functions such as wildcard, fuzzy, proximity and range searches. It also permits to affect weighted coefficients to boost a term; it is particularly interesting considering Xml structures since some markups can be considered as more relevant than others. For instance, the *Title* element can be considered as more important than the *PromotionalInformation* one. This kind of comparison is obviously subjective; As a result, the M-Tool search engine includes a module to allow users to personalize these weighted coefficients. This module could be used by the administrator or by client-side users.

After indexation and search functionalities, the search visualization is also an important aspect to take into account in search engines. The M-Tool search engine presents search results as in most internet search engines; Search results are displayed one after the other, each of them including:

- The title
- A short description of the context around the searched words
- A ranking mark (from 100 to 0%)
- A thumbnail of the multimedia content if available
- A link to the found program in the multimedia content and to a XML representation of the results (ProgramInformation and ProgramLocation)

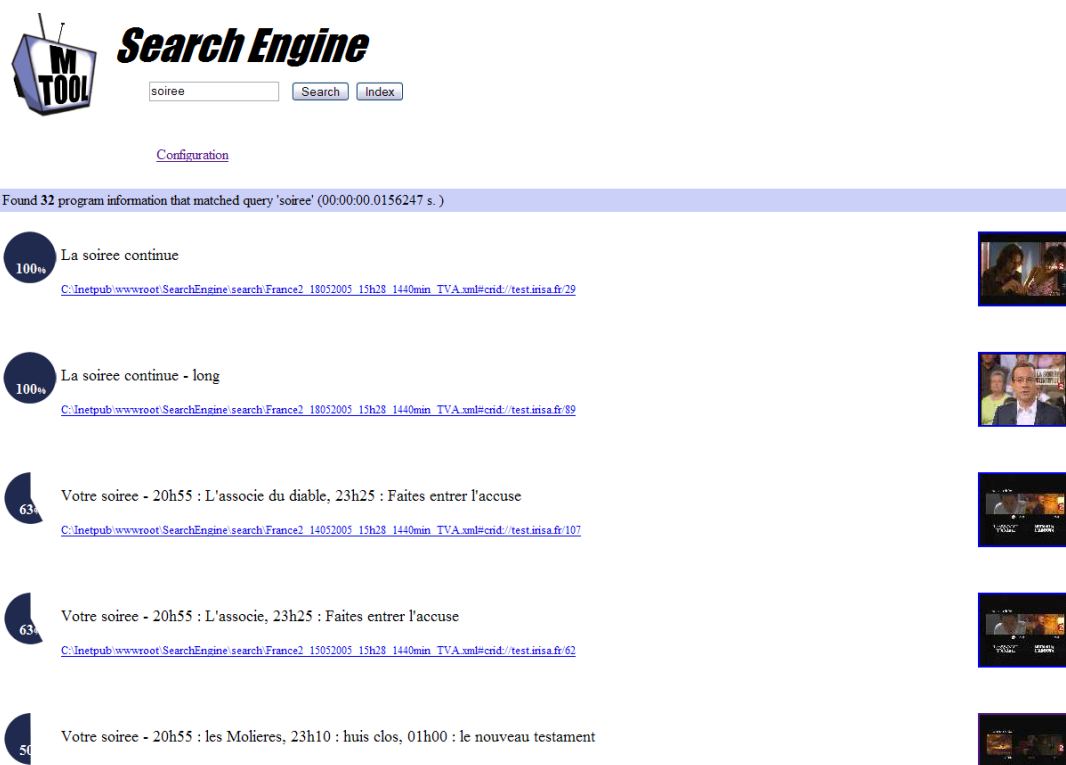



Figure 14: M-Tool Search Engine results page

A configuration module is under development (Figure 15). It implies choosing elements to search among and linking these elements to weighted coefficients (or boost coefficients). For example, a user could choose to search in *Title* and *Keyword* elements with 5 and 1 as weighted coefficients. These coefficients are useful to compute the final mark and thus to build a ranking of all results. The final mark consists of intermediate marks given to each element where the searched words were found. The intermediate mark depends on the context of words and the number of instances inside the element; for instance, if the searched word is *bed*, the movie *Bed of Roses* will get a better mark than the movie *The Bedroom Window* owing to its context.

This Search Engine will be used for querying multimedia documents inside the TELEMEX multimedia platform which gathers a corpus of audiovisual contents taken out from the France 2 TV channel. This corpus is currently used for testing purpose.



Search Engine

- ☒ Title
- ☐ Synopsis
- ☐ Keyword
- ☐ ShortTitle
- ☐ PromotionalInformation
- ☐ Genre
- ☐ ParentalGuidance
- ☐ Language
- ☐ CreditsList
- ☐ AwardsList
- ☐ RelatedMaterial
- ☐ ReleaseInformation

1

▼

1

2

3

Submit

Figure 15: Configuration of the M-Tool Search Engine

7. Conclusion

In this master's thesis, I have introduced an innovative approach that unifies MPEG-21 and TV-Anytime descriptions for modeling in a unified way content-descriptive metadata associated to audiovisual resources. I also described functionalities of the M-Tool authoring tool that serves as a proof of concept for our vision of metadata management and orchestration for TV, News and video on demand broadcasting scenarios in the framework of the European ENTHRONE project. The M-Tool will be improved to meet requirements of future developments in the ENTHRONE project. Besides, the M-Tool priority will be more oriented to the metadata management and querying.

Our metadata model enables gathering mixed resources inside the same structure. Even if building interactive interfaces is far from being a top priority in the ENTHRONE project, there is very little to add to our model to design a structure flexible enough to create these kinds of interfaces. As a result, adding spatial layout and hyperlinks information can be considered to make our model able to create these interfaces.

The M-Tool media player currently allows users to annotate audiovisual contents on account of the Windows Media Player. The M-Tool was foreseen to annotate a wider array of multimedia documents from web page to interactive audiovisual contents; so, the media player will also have to play other kinds of multimedia contents (images, text...) in addition to audio and video. The M-Tool main objective is to provide Content Providers with a tool that facilitates annotation of their resources. It could either be used to create metadata from scratch (manually) or enrich automatically extracted metadata. For this reason, the M-Tool prototype should be located on the Content Providers side which can create or update their metadata while resources are first being broadcasted. As a result, a major future work will be to integrate our work inside the ENTHRONE project. This includes the integration of interfaces to communicate with the IMS and TVMs. The M-Tool Search Engine will be enhanced with mixed search coupling exact keywords search and more complex Xqueries on content-descriptive information. It will also use low-level features on videos, images and multimedia documents (e.g., MPEG-7 image global descriptors such as ColorLayout, ScalableColor for color, HomogeneousTexture, EdgeHistogram for texture and RegionShape for shape; local image descriptors such as interest points; segmentation and face recognition tools for video, etc.). Besides, keeping track of user's previous searches is also an important way forward. User profile-based personalization for metadata annotation and multimedia information retrieval will be considered: based on previous work in the e-learning community [RBM04] a user profile-based personalization model of the annotation and search processes will be proposed by tracing, retrieving and inferring information from the users and their preferences in a non-intrusive way.

In parallel with the development of the M-Tool, metadata management will also be explored. Fusion of annotations and XML metadata synchronization will be studied: Merging

several XML trees of metadata files and several versions of annotations is a very useful enhanced functionality for annotators; this may allow integration, re-utilization and propagation of some descriptive parts of existing available annotations. Furthermore, metadata monitoring and control will be considered: because metadata and annotations will be transmitted along with the content over a variety of physical media using appropriate protocols, these metadata should be valid, up-to-date and consistent with their related media content. Controlling, refreshing and synchronizing metadata files in conformance with their content include detection and elimination of XML metadata (sub-)item (or component) duplicates, and checking freshness and consistency of both metadata associated to distributed digital resources.

The TV-Anytime metadata specification considers Consumer Metadata as a mean to track user actions and usage history to generate user preferences by automatic analysis. It defines:

- UsageHistory: the audiovisual content consumption history for a user as a list of actions performed by the user during an observation period
- UserActionHistory: provides an ordered log of specific user actions such as “Play”, “Pause”, “Fast Forward”, etc.
- UserPreferences: facilitates description of user preference to consume multimedia material. Correspondence between user preferences and resource description facilitates accurate and efficient personalization of content access and consumption.

All these functionalities given by TV-Anytime will be included in the M-Tool framework. Furthermore, TV-Anytime Phase 2 [TVA-2] should be analyzed in-depth to check how it could be integrated in future versions of the M-Tool. Phase 2 will specify open standards that build on the foundations of Phase One specifications and will include areas such as targeting, redistribution and new content types. Phase 2 of the TV-Anytime standard will address how content types other than audio and video (e.g., games, enhanced TV, web pages, music files, graphics, data and many other applications) can be transferred and stored throughout home networks. For this, the standard will permit the secure management of content within and among compliant devices and it will permit the privacy and security of data associated with user identity and interaction.

References

- [ADO05] Adobe Systems Inc., Adobe Premiere Pro 1.5. <http://www.adobe.com/>. Last accessed on the 13th of July 2005.
- [AGL03] G. D. Abowd, M. Gauger and A. Lachenmann, “The Family Video Archive: An Annotation and Browsing Environment for home movies”, Proceedings of the ACM conference on Multimedia Information Retrieval, November 2003.
- [CR95] T.S. Chua and L.Q. Ruan. “A video retrieval and sequencing system”. Proceedings of the ACM Transactions on Information Systems, pages 373-407, October 1995.
- [DJ05] Cédric Dufouil, Wilfried Jouve, “Stratégies de gestion et d’utilisation des métadonnées dans les réseaux de diffusion de contenus multimédia”, in Proceedings of the MajecSTIC’05 conference, Rennes, France, November 2005.
- [DKL05] G. Durand, G. Kazai, M. Lalmas, U. Rauschenbach and P. Wolf, “A metadata model supporting scalable interactive TV services”, Proceedings of the IEEE conference on Multi-Media Modeling, January 2005.
- [ENI05] Enikos DI Creator and Browser, <http://www.enikos.com/home.shtml>. Last accessed on the 13th of July 2005.
- [ENTD03] ENTHRONE deliverable D03 (WP2) – “Metadata definition and specification”, May 2004, INRIA.
- [ENTD15] ENTHRONE deliverable D15 (WP4) – “Metadata Authoring Tool TVM Processors”, June 2005, INRIA.
- [ENTSoA] ENTHRONE WP3 tutorial, “MPEG-21: A State-of-the-art Survey”, March 2005, INESC
- [FSN95] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content: the QBIC system. IEEE Computer, 28:23–32, 1995.
- [HS95] A. G. Hauptmann, and M. Smith, “Text speech and vision for video segmentation: The informedia project”, Proceedings of the AAAI Symposium on Computational Models for Integrating Language and Vision, 1995.
- [IBM02] IBM research. “VideoAnnEx – IBM MPEG-7 Annotation Tool”, 2002.
- [INA] Home page of “L’institut audiovisuel de l’audiovisuel”, <http://www.ina.fr/>
- [ISOM21] ISO/IEC PDTR 21000-1, “Information technology — Multimedia framework (MPEG-21) – Part 1: Vision, Technologies and Strategy”, December 2004.

- [ISOM212] ISO/IEC PDTR 21000-2, “Information technology – Multimedia framework (MPEG-21) – Part 2: Digital Item Declaration”, June 2005.
- [ISOM7] ISO MPEG-7, part 5 – Multimedia Description Schemes. ISO/IEC JTC1/SC29/WG11/N4242, 2001.
- [JLR98] M. Jourdan, N. Layada, C. Roisin, L. Sabry-Ismail, and L. Tardif. “Madeus: an Authoring Environment for Interactive Multimedia Documents”, Proceedings of the ACM conference on Multimedia, 1998.
- [JPL00] J. Platt, AutoAlbum: Clustering Digital Photographs Using Probabilistic Model Merging, IEEE Workshop on Content-Based Access of Image and Video Libraries 2000.
- [LUC05] “DotLucene – The Open Source Search Engine for .NET”, <http://www.dotlucene.net/>, Last accessed on August 2005.
- [MCS01] B. Myers, J. P. Casares, S. Stevens, L. Dabbish, D. Yocum and Albert Corbett, “A Multi-View Intelligent Editor for Digital Video Libraries”, Proceedings of the ACM/IEEE-CS joint conference on Digital libraries, June 2001.
- [ME95] T. Meyer-Boudnik and W. Effelsberg: “MHEG – An Interchange Format for Interactive Multimedia Presentations”, IEEE Multimedia Magazine, Spring 1995.
- [MSS02] B. S. Manjunath, P. Salembier and T. Sikora, “Introduction to MPEG-7: Multimedia Content Description Interface”, ISBN: 0-471-48678-7, Wiley, April 2002.
- [RBM04] B. Rousseau, P. Browne, P. Malone and M. Ófoghlu, “User Profiling for Content Personalisation in Information Retrieval”, Proceedings of the ACM Symposium on Applied Computing, March 2004.
- [RJB05] B. Rousseau, W. Jouve, L. Berti-Équille, “Enriching Multimedia Content Description for Broadcast Environments: From A Unified Metadata Model to A New Generation of Authoring Tool” submitted to the IEEE ISM2005 conference, Irvine, California, USA, December 2005.
- [RSK02] J. Ryu, Y. Sohn, and M. Kim, “MPEG-7 Metadata Authoring Tool”, Proceedings of the 10th ACM International Conference on Multimedia, pages 267-270, December 2002.
- [TCJ01] J. E. Tandianus, A. Chandra and J. S. Jin, “Video Cataloguing and Browsing”, Proceedings of the workshop on Visual Information Processing, 2001.
- [TR03] T. Tran-Thuong, C. Roisin, “Multimedia Modeling Using MPEG-7 for Authoring Multimedia Integration”, Proceedings of the ACM conference on Multimedia Information Retrieval, November 2003.

- [TVA1] European Telecommunications Standards Institute, “ETSI TS 102 822-3-1 v1.2.1; Broadcast and On-line Services: Search, select and rightful use of content on personal storage systems (“TV-Anytime Phase 1”); Part 3: Metadata. Sub-part 1: Metadata Schemas”, September 2004.
- [TVA2] European Telecommunications Standards Institute, “ETSI TS 102 822-3-3; Search, select, and rightful use of content on personal storage systems (“TVAnytime Phase 2”); Part 3: Metadata; Sub-part 3: Extended Metadata Schema”, May 2005
- [XML] eXtensible Markup Language (XML), <http://www.w3.org/XML/>
- [XMLS] XML Schema, <http://www.w3.org/XML/Schema>
- [XPA99] XML Path Language (XPath) Version 1.0, <http://www.w3.org/TR/xpath>
- [XQU05] XQuery 1.0: An XML Query Language, <http://www.w3.org/TR/xquery/>

ANNEX A: TV-Anytime elements

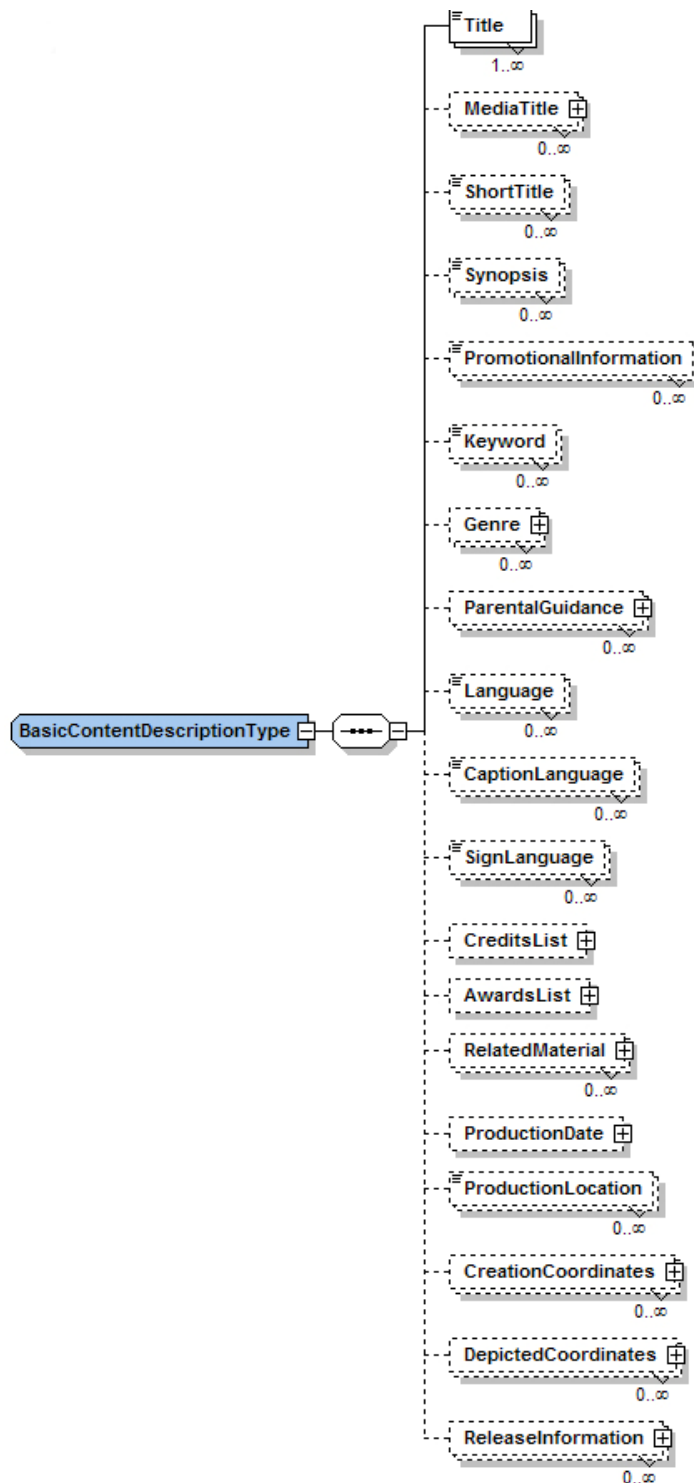


Figure A.1: Semantic Description in TV-Anytime

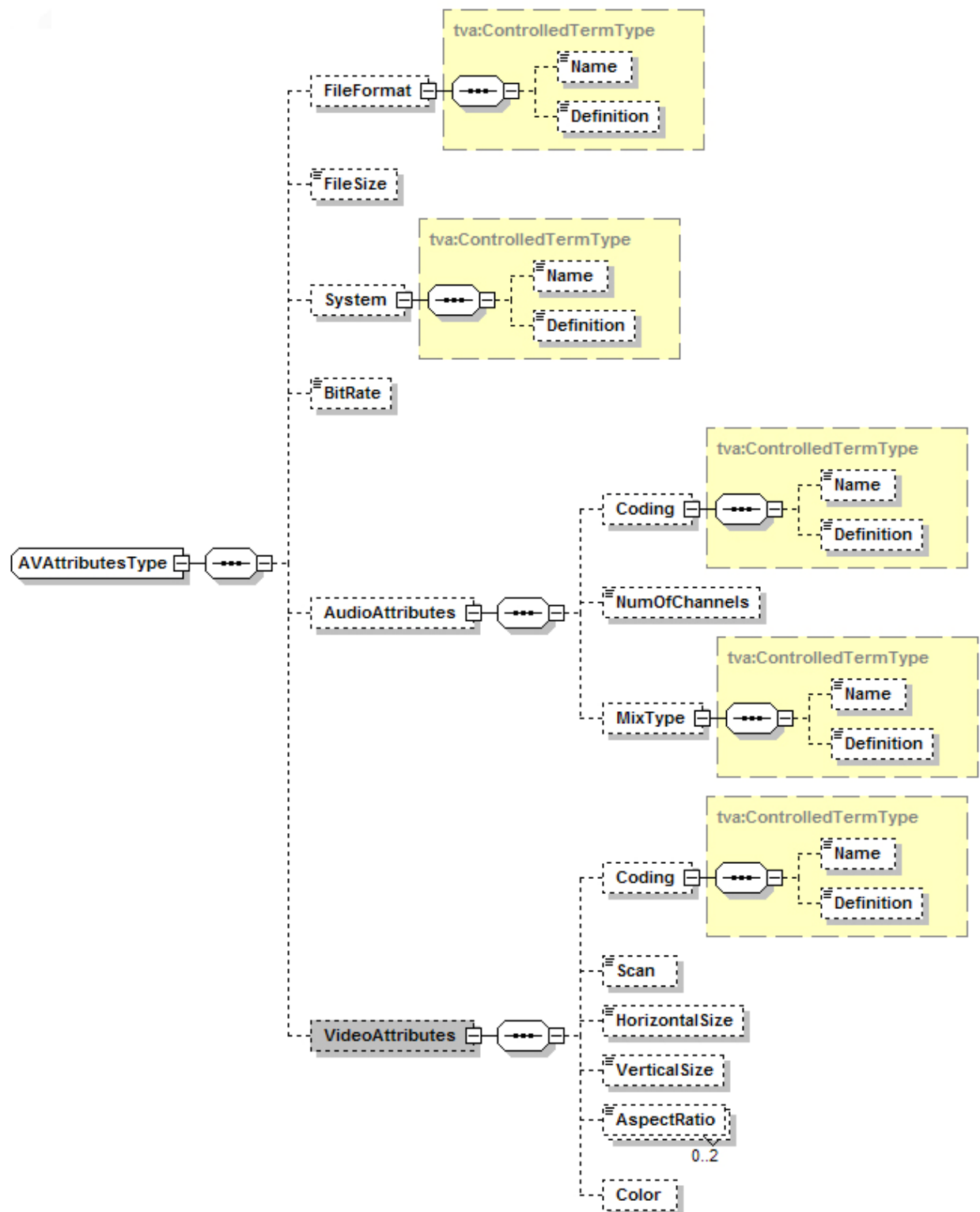


Figure A.2: Audio/Video Attributes in TV-Anytime

ANNEX B: Scenarios in ENTHRONE

<i>ENTHRONE Applications</i>	<i>Metadata format</i>	<i>Content decoding</i>	<i>Terminal Platforms</i>
Free TV Broadcast	MPEG-21 DID TV-Anytime	MPEG-2 over DVB-T	Television set, set top box, mobile DVB-T device.
iTV-MHP Service	MPEG-21 DID TV-Anytime	MPEG-2 over DVB-T	Television set, set top box with PSTN connection over ADSL. Mobile devices with PSTN connection over GPRS/UMTS. Support DVB MHP-compliant services.
On demand video services to mobile devices with QoS over cooperating DVB-T / GPRS	QoS metadata MPEG-21 DID	MPEG-2 over DVB-T	Mobile devices with PSTN connection over GPRS/UMTS.
Broadcast TV service to mobile devices over DVB-T	QoS metadata MPEG-21 DID MPEG-21 DI	MPEG-2 over DVB-T	Mobile devices with PSTN connection over GPRS/UMTS.
information services on public transportation	MPEG-21 DID and DI Metadata for enhanced information (GPS, time)	MPEG-2 over DVB-T	Public PC terminal (a.k.a. Collective Info-Mobile Terminals).
Individual on demand information	MPEG-21 DID and DI Metadata for enhanced information (GPS, time)	MPEG-2 over DVB-T	PDA/smart phone with WLAN (802.11x) connectivity.
Video on Demand	MPEG-21 DID and DI HTML	MPEG-2/4 over DVB-T/S, UMTS/GPRS or IP.	Private PC/STB terminal with broadband PSTN connectivity (ADSL, cable-modem, optical fiber). PDA or mobile phones with UMTS/GPRS connectivity.
Multimedia content download	MPEG-21 DID and DI. HTML browser IP protection, licenses needs	MPEG-2/4 A/V decoders, MPEG-21 DI and DID parsers and encoders. MPEG-7 parser and encoder.	Fixed (PC/STB) or mobile (PDA or mobile phones) terminals either with or without broadband PSTN connectivity (ADSL, cable-modem, optical fiber).
Pay-TV	MPEG-21 DID and DI HTML TV-Anytime	MPEG-2/4 over DVB-T/S or UMTS/GPRS or IP.	Fixed (PC/STB) or mobile (PDA or mobile phones) terminals either with or without broadband PSTN connectivity (ADSL, cable-modem, optical fiber).
Distance learning	MPEG descriptors and Digital Item or other format that are supported by the infrastructure (FTP, chat, whiteboard).	MPEG-2/4 over DVB-T, UMTS/GPRS or IP.	Fixed (PC/STB) or mobile (PDA or mobile phones) terminals either with or without broadband PSTN connectivity (ADSL, cable-modem, optical fiber).

Table B.1

ANNEX C: Technical description of the M-Tool

In this annex, my goal is not to give an exhaustive description of the M-Tool but to focus on main features of the M-Tool API.

C.1. Form generator

How to describe the form?

I have considered the TV-Anytime specification as consisting of groups of fields. By group of fields, I mean TV-Anytime elements having the same root element.

The following example is a group of fields (the root element is Genre):

BasicDescription~Genre~href

BasicDescription~Genre~Name

BasicDescription~Genre~Definition

In the framework of the M-Tool, root elements are the first children of *BasicDescription* (*Title*, *Synopsis*, *Genre*, *CreditsItemList*...) or the *AVAttributes* which is also considered as a root element.

These groups of fields are the corner stone of the form generator. They are displayed on account of two classes, *Appearance* and *Design*. There is a single *Appearance* class for one group of fields.











 cs	The group uses a classification scheme?
 designList	List of Design object building the group of fields
 height	height of the label
 instancesLevel	Instantiation levels of this group
 instancesName	Name of the instantiation levels; for presentation only; synchronized with the ArrayList instancesLevel.
 name	name of the label
 title	Is it a title?
 varName	Name of each field (ex: <i>BasicDescription~Genre~Name</i>)
 width	width of the label
 x	x position of the label

Table C.1: Public instance fields of Appearance

The Appearance class

The *Appearance* class describes the label used to identify the group of fields in the form (e.g. *Genre* for the *Genre* group of fields). It describes the level(s) of instantiation. An instantiation is the fact that some TV-Anytime elements can repeat itself several times. A case in point is the *Genre* element; there could be several *Genre* element inside the same *ProgramInformation*. So, in this case, we would put 1 in the *instancesLevel* ArrayList since the *Genre* element is the second element in the strings *BasicDescription~Genre~href*, *BasicDescription~Genre~Name* and *BasicDescription~Genre~Definition*. Some group of fields can have several levels of instantiation such as the *Credits* group of fields where *CreditsItem* and *Character* can be both repeated.

The *varName* ArrayList contains all available fields' paths. These paths follow the pattern: *BasicDescription~Genre~href*, *BasicDescription~Genre~Name* or *BasicDescription~Genre~Definition*.

The *Appearance* class can also describe a title in the form (e.g. *Credits*). In this case, there is obviously no group of fields attached to this class.

This class describes each field by a *Design* object so as to display these fields in the form.

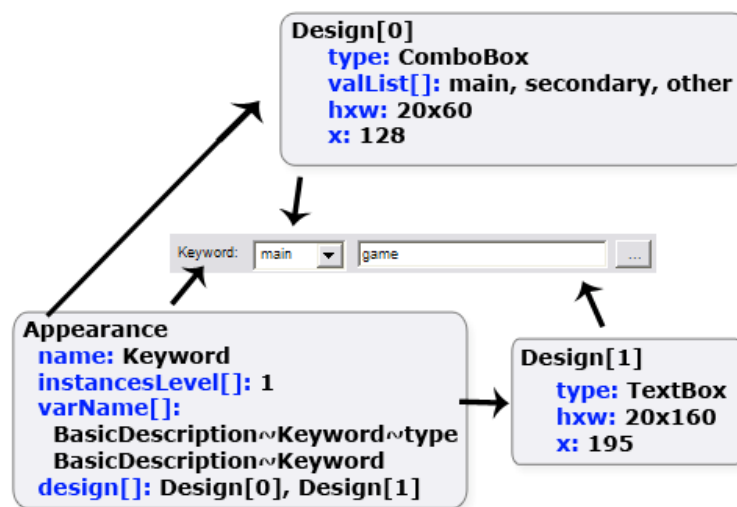


Figure C.1: What do the Appearance and Design classes describe?

The Design class

A *Design* object describes a single field. This field has a type (*textbox*, *richtextbox* or *combobox*), a size (*height*, *width*) and a position (*x*,*y*). The *y* position being never used, all fields belonging to the same group are in the same line. If the type is *ComboBox*, the group of fields can be described by the Classification Schemes or not. If the group of fields uses some classification schemes (=cs *ComboBoxes*), it has the following pattern: the *Appearance* class has 3 *varName*

(*href*, *Name* and *Definition*) and 2 *Designs* (for the 2 *ComboBoxes*). A case in point is the *Keyword* group of fields (see above).

The *valList[]* contains values to fill the *ComboBox* (e.g. *main*, *secondary* and *other*), only for non *cs* *combo boxes*. The *csList* contains the values of each member of this combo box or the path of the classification schemes for *cs* combo box.









 <i>csList</i>	Classification schemas to load or value to record in the <i>comboBox</i> if non <i>cs</i> <i>combobox</i>
 <i>height</i>	height and width of the box
 <i>name</i>	name of the box (useless)
 <i>type</i>	type of the box (<i>textbox</i> , <i>combobox</i> , <i>richtextbox</i>)
 <i>valList</i>	When the type is <i>combo box</i> and it is not a <i>CS</i> <i>combo box</i> , it contains values to fill this <i>combo box</i>
 <i>width</i>	height and width of the box
 <i>x</i>	<i>x</i> position of the box
 <i>y</i>	<i>y</i> position of the box

Table C.2: Public instance fields of *Design*

The profile

A group of fields is not static. It depends on the selected elements in the profile. For instance, the *Genre* group of fields could contain *BasicDescription~Genre~href*, *BasicDescription~Genre~Name* and *BasicDescription~Genre~Definition*. OR *BasicDescription~Genre~href*, *BasicDescription~Genre~Name*, *BasicDescription~Genre~Definition*. and *BasicDescription~Genre~Type*.

When loading the profile, *Appearance* classes are built and gathered in a single *ArrayList*.

C.2. Saving

There are 3 levels of saves. The first level consists of *ArrayLists* of programs; the second level is the *XmlDocument* and the third level is the physical level, the file. The second and third levels can be considered as a single level since they are always synchronized (e.g. when saving the second level, the third level is saved too). So the first level is called the *cache* and the last level, the file. This structure was thought for its flexibility (namely for the form generation) and the enhancement of speed which follows.

The main ArrayList *XmlTVADoc.listPrograms* contains all programs hierarchically sorted. A program is described with 2 classes: *ProgramInformation* and *ProgramLocation*. A *ProgramInformation* object mainly consists of:

- A list of fields' names (e.g. *BasicDescription~Genre~Name*), *varName*
- A list of fields' values synchronized with *varName*, *varValue*
- A list of sub programs (which are also *ProgramInformation* objects), *subPrograms*
- A list of *ProgramLocation* objects, *programLocation*











 add	Does this program have to be added?
 crid	The programId or crid attribute
 new_crid	If the user changes the crid, we must keep the old value (crid) to browse inside the XmlDocument.
 programLocation	List of ScheduleEvent for this program
 save	Does this program have to be saved?
 subPrograms	It contains all subprogrammes with a depth of 1
 title	The title element
 varInstance	It contains level(s) of instantiation
 varName	It contains names of variables.
 varValue	It contains values of variables.

Table C.3: Private instance fields of ProgramInformation

A *ProgramLocation* object contains all values of a *schedule event*.

The following example gives the details of *varName[]* and *varValue[]* for the *CreditsList* group of fields. The *CreditsList* group of fields has two levels of instantiation: the *CreditsItem* element (position 1) and *Character* element (position 2). So, *varInstance* will contain values, 1 and 2.

```

<CreditsList>
  <CreditsItem role="CreditsItem_1_role">
    <PersonName>
      <mpeg7:GivenName>CreditsItem_1_PersonName_1_GivenName</mpeg7:GivenName>
      <mpeg7:FamilyName>CreditsItem_1_PersonName_1_FamilyName</mpeg7:FamilyName>
      <mpeg7:Title>CreditsItem_1_PersonName_1_Title</mpeg7:Title>
    </PersonName>
    <Character>
      <mpeg7:GivenName>CreditsItem_1_Character_1_GivenName</mpeg7:GivenName>
      <mpeg7:FamilyName>CreditsItem_1_Character_1_FamilyName</mpeg7:FamilyName>
      <mpeg7:Title>CreditsItem_1_Character_1_Title</mpeg7:Title>
    </Character>
  </CreditsItem>
</CreditsList>

```

```

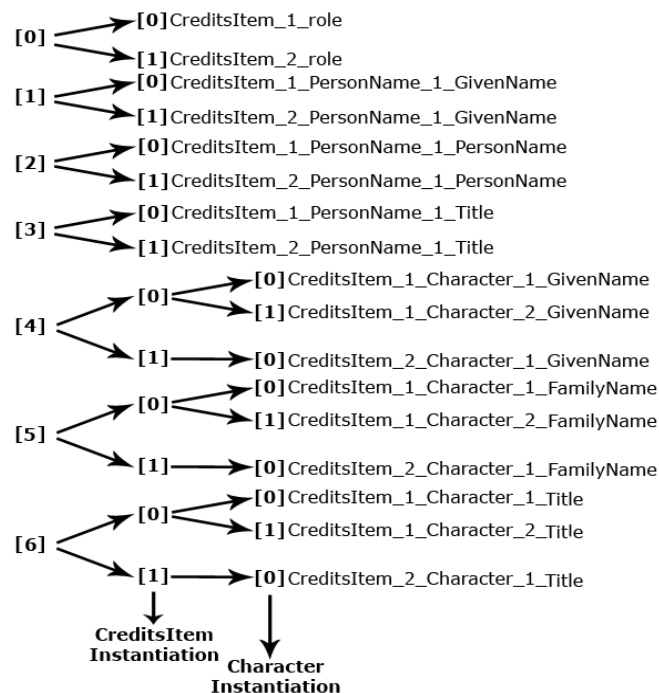
<Character>
  <mpeg7:GivenName>CreditsItem_1_Character_2_GivenName</mpeg7:GivenName>
  <mpeg7:FamilyName>CreditsItem_1_Character_2_FamilyName</mpeg7:FamilyName>
  <mpeg7:Title>CreditsItem_1_Character_2_Title</mpeg7:Title>
</Character>
</CreditsItem>
<CreditsItem role="CreditsItem_2_role">
  <PersonName>
    <mpeg7:GivenName>CreditsItem_2_PersonName_1_GivenName</mpeg7:GivenName>
    <mpeg7:FamilyName>CreditsItem_2_PersonName_1_FamilyName</mpeg7:FamilyName>
    <mpeg7:Title>CreditsItem_2_PersonName_1_Title</mpeg7:Title>
  </PersonName>
  <Character>
    <mpeg7:GivenName>CreditsItem_2_Character_1_GivenName</mpeg7:GivenName>
    <mpeg7:FamilyName>CreditsItem_2_Character_1_FamilyName</mpeg7:FamilyName>
    <mpeg7:Title>CreditsItem_2_Character_1_Title</mpeg7:Title>
  </Character>
</CreditsItem>
</CreditsList>

```

varName[]

- [0] "BasicDescription~CreditsList~CreditsItem~role"
- [1] "BasicDescription~CreditsList~CreditsItem~PersonName~mpeg7:GivenName"
- [2] "BasicDescription~CreditsList~CreditsItem~PersonName~mpeg7:FamilyName"
- [3] "BasicDescription~CreditsList~CreditsItem~PersonName~mpeg7:Title"
- [4] "BasicDescription~CreditsList~CreditsItem~Character~mpeg7:GivenName"
- [5] "BasicDescription~CreditsList~CreditsItem~Character~mpeg7:FamilyName"
- [6] "BasicDescription~CreditsList~CreditsItem~Character~mpeg7:Title"

varValue[] (Storage of values)



C.3. Building DID structures

The drag and drop is performed from Picture Boxes of the *ToolBox* to the *drawingPanel* of the *MPEG21Structure* class. Three events are used to simulate this drag and drop action. First, the *MouseDown* event is used when a user keeps pushing the left button of the mouse on one of *ToolBox* icons (*Picture Boxes*). It activates another event, the *MouseMove* event which sticks a *Picture Box* to the mouse pointer. It is useful to notice that *Picture Boxes* used for moving, are different from those used in the *ToolBox*; these moving *Picture Boxes* belong to the *MPEG21Edition* class. When the user releases the left button, the *MouseUp* event is raised and stops the *MouseMove* event. Then, it calls a method of the *MPEG21Structure* to determine where the icon was dropped.

The drawing panel

The drawing is built on account of the *TreeView*. To avoid repeating the same action several times, I proceed as follows: First the MPEG-21 *XmlDocument* is parsed in the *MPEG21TreeView* class by selecting relevant elements. Then the *MPEG21Structure* draws the *TreeView* in the panel (so it uses information from this *TreeView*).

In a panel, it is impossible to differentiate a rectangle from another, all belonging to the same structure. So, when drawing the structure of a MPEG-21 file, the rectangles positions/names (for DIDL, Containers, Items, Components and Descriptors) and the icons positions/names (for all descriptors) are recorded. This information is stored in 4 *ArrayLists*:

- *rectanglePathList* (string => "Container 0")
- *rectanglePositionList* (int[4] => x, y, width, height)
- *imageNameList* (string => "1. REL")
- *imagePositionList* (int[4] => x, y, width, height)

Rectangles positions and names are used when dropping an element inside the panel. On account of these positions/names, we can check if this action is allowed and then insert the new element in the right place. This information is also useful to delete structure elements (Container/Item/Component). When right clicking on the panel, a menu is displayed, notifying the element that can be deleted.

Icons positions/names are used to display a menu for each descriptor; when right clicking on an icon, a menu proposes to edit or delete the descriptor.

It also important to notice that *imageNameList* and *imagePositionList* contain a list of *ArrayLists*. The first index matches with the number of descriptor and the second index matches with the number of the icon inside the descriptor (for the moment only Schedule descriptor can have multiple *ScheduleEvents* events).

rectanglePathList	{Count=14}
[0]	"DIDL"
[1]	"Container 0"
[2]	"REL"
[3]	"DIA"
[4]	"Item 0"
[5]	"TVA: ProgramInformation"
[6]	"TVA: ProgramInformation"
[7]	"DIA"
[8]	"TVA: Schedule"
[9]	"Component 0"
[10]	"TVA: ProgramInformation"
[11]	"TVA: Schedule"
[12]	"Free Text"
[13]	"DIA"

Figure C.2: rectanglePathList example

imageNameList	{Count=10}
[0]	{Count=1}
[0]	"1. REL"
[1]	{Count=1}
[0]	"2. DIA"
[2]	{Count=1}
[0]	"3. TVA: ProgramInformation"
[3]	{Count=1}
[0]	"4. TVA: ProgramInformation"
[4]	{Count=1}
[0]	"5. DIA"
[5]	{Count=4}
[0]	"6.1. TVA: Schedule"
[1]	"6.2. TVA: Schedule"
[2]	"6.3. TVA: Schedule"
[3]	"6.4. TVA: Schedule"
[6]	{Count=1}
[0]	"7. TVA: ProgramInformation"
[7]	{Count=1}
[0]	"8.1. TVA: Schedule"
[8]	{Count=1}
[0]	"9. Free Text"
[9]	{Count=1}
[0]	"10. DIA"

Figure C.3: imageNameList example

C.3. Right click on icons

MPEG21Edition.drawingPanel_MouseDown: It determines if it is a right click

MPEG21Structure.DisplayContextMenu: It displays the ContextMenu by retrieving icon information, especially the descriptor ID and the name (e.g. 6.1. ScheduleEvent or 2. REL)

MPEG21Structure.editMenuItem_Click: If "edit" is chosen, it runs the appropriate module to edit the selected descriptor.

I use the descriptor ID to find the right descriptor. The descriptor ID is the number of the descriptor among editable descriptors (DIA/REL/ProgramInformation/ScheduleEvent/FreeText)

C.4. Double click on icons

It uses the same mechanism as the "right click" action.

C.5. Creating new elements

New Schedule events or program information

When creating new schedule event or program information, the program must get some information to determine the CRID. A first form is proposed to the user to choose (NewProgramInformation and NewScheduleEventForm)

- For new program information
 - If it is a new program (so new crid => previous crid + 1)
 - If it is a sub program (e.g. an existing crid + “;0”)
 - If it is a sequel of an existing program (so same crid), Useful for AVAttributes separated form BasicDescription
 - For new schedule event
- If it is attached to an existing program (so same crid)
- If it is a sub program attached to a root program (e.g. an existing crid + “;0”)

A second form (ProgramInformationListForm) could be proposed to choose the ProgramInformation to link with the new element.

New DIA and REL

When creating a new DIA element, a form proposes to load an existing DIA file inside the specified place. Then an open dialog box is displayed to find the file to load. The form also proposes to create the DIA file; if the user chooses this option, the ETRI DIA editor is loaded to create a new DIA file. Then, the user must choose the first option to load this new DIA file.