Integrated Multimedia Authoring and Description Framework

Luís F. Teixeira[1,2] **and Luís Corte-Real**[1,2]

[1] Faculdade de Engenharia da Universidade do Porto
Rua Dr. Roberto Frias, s/n – 4200-465 Porto, Portugal
[2] INESC Porto
Campus da FEUP, Rua Dr. Roberto Frias, 378 – 4200-465 Porto, Portugal

Abstract Digital information has reached all types of expression and, above all, it acquired important properties: portability, mobility and ubiquity. Moreover, with multimedia content being produced at a fast pace, the question of how one can effectively search and process the growing offer of content is immediately raised. We address these demands, proposing a modular, flexible and scalable framework that fully integrates content representation and description from early stages of authoring. A prototype, supporting the MPEG-4 and MPEG-7 standards, has been implemented to prove the concept and is being used as a test platform for multimedia content adaptation.

1 Introduction

Nowadays a scenario where multimedia content is provided by a server and accessed by a large set of devices (with different characteristics and access means) is easily imaginable. The content is itself composed by several primary elements, or objects, including video, static images, music, voice, text. Moreover, the content can be adapted according with network and device characteristics (bandwidth, resolution, capabilities) and the taste of the user. Some challenges are therefore raised. On one side, the end user expects mobility and ubiquity. On the other, the content creators demand content adaptability and reuse, as effortless as possible. Any information gathered during the authoring process can be very valuable to tackle these challenges. The most obvious gains are for the search and retrieval of multimedia content and content adaptation by providing important clues to the search and adaptation engines. The MPEG-4 and MPEG-7 standards play an important role in this scenario [7] [9].

The MPEG-4 standard (ISO/IEC 14496) allows the composition of interactive multimedia content consisting of both natural and synthetic media, which can be streamed within environments characterised by a high degree of variability and flexibility. Until 2001, the research in tools capable of manipulating MPEG-4 was nearly absent. Two early exceptions were MPEG-4 Toolbox [3], for the composition of 3D scenes, and MPEG-Pro [2], supporting 2D authoring of scenes directed by a timeline. However none of these tools enabled the association of metadata during the content creation process. The MPEG-7 standard (ISO/IEC 15938) provides technologies for the description of audiovisual data content in multimedia environments. Until recently, content description tools were mainly used for the information retrieval domain but efforts have been put to design systems that integrate multimedia content and content description. Tran-Thuong et al. [10] propose the use of description tools in multimedia authoring which provide access into the media structure for fine-grained composition. Bertini et al. [1] have developed a prototype system for annotation and adaptation of soccer sport videos, with adaptation based on objects and highlights. Both show that the integration of content, or *media-data* with content description, or *metadata* proves to be a valuable addition to multimedia creation and distribution processes. Nevertheless, both focus on specific applications which limits their use when flexibility and adaptability are in order.

Taking into account this background, we present the specification and implementation of a framework for multimedia authoring, which integrates all information during the creation process in a seamless way.

2 System Architecture and Implementation

To provide a flexible integration of media-data with metadata, the system architecture proposed is based in a *distributed architecture*. System users can access and share resources in a transparent, open and scalable way. It also allows its deployment in a wide variety of scenarios and environments. The system stores the content as multimedia presentations (MPs) and media objects (MOs).

Each MP, defined using a scene description format, may reference MOs, which are stored in their original format (e.g. JPEG or PNG for images, MP3 for audio and MPEG-2 for video). Alongside the stored content, additional information (metadata) is also stored, that describes the content and can be useful for re-authoring and re-purposing of contents.

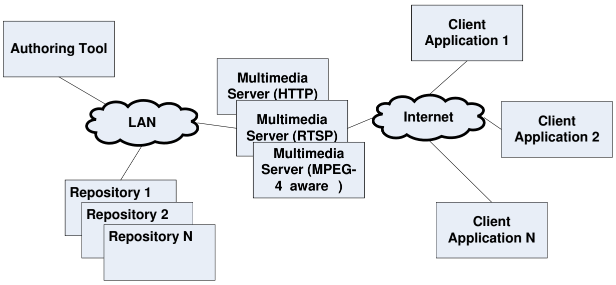


Fig. 1 Generic distributed system architecture.

The system has four different entities, as shown in Figure 1. The *Repository* module stores the MPs and MOs, each having a unique identifier within the system, allowing an univocal identification. An interface to the user allows to manage MOs, MPs and the associated metadata. Typical functionalities include importing, exporting, deleting MPs and MOs and editing metadata. The *Authoring Tool* is responsible for the creation and modification of MPs. This manipulation consists in the composition of dynamic and interactive multimedia content. MOs stored in one of the repositories can be referenced by the presentation using its identifier. Although the Authoring Tool is shown in Figure 1 as single entity, it has in fact a many-to-many relation with other entities. The *Multimedia Server* is responsible for the content delivery. If the Multimedia Server is a file-based server (HTTP, FTP), a file, containing the MP and the composing objects, is created and sent to the client. On the other hand, if the server is stream-based, one possible option is to create an hinted file, containing information that aids the server in the streaming process. Another possibility is a *content-aware* server that initially sends only the MPEG-4 scene description. Each referenced object is then streamed only when requested by the client application, allowing remote interactivity: commands sent by the client application to the server can change some characteristic of the presentation. Finally, the *Client Application*, in any device can access, through any network, one of Multimedia Server and start receiving a MP. The user can then view and interact with the presentation.

2.1 Prototype implementation – edVO

Taking the system architecture shown in Figure 1 as a basis, the edVO prototype was developed. Currently, the MPEG-4 Systems [4] and MPEG-7 MDS [5] standards are used for the representation and description of multimedia content. The prototype consists of three different modules: `edVO.Composer`, `edVO.Server` and `edVO.Client`. Only the former two modules will be detailed here.

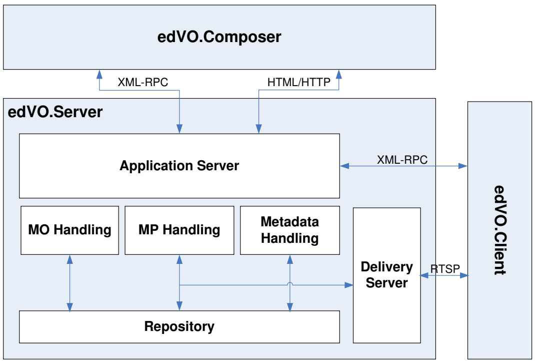
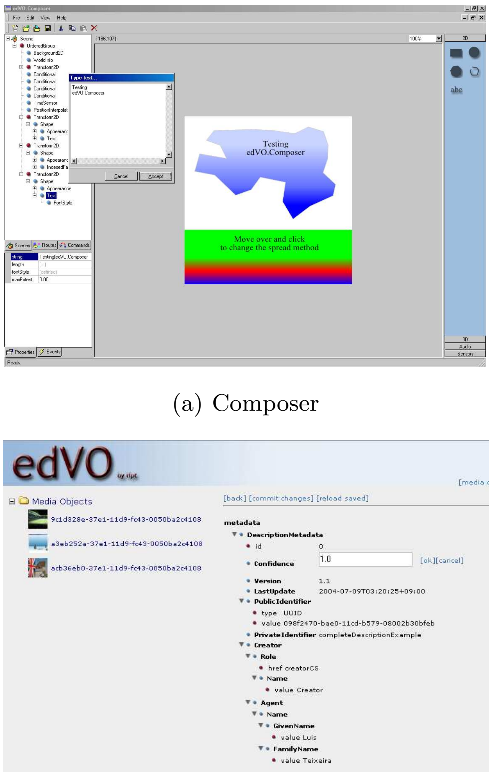


Fig. 2 Diagram showing module interaction in the edVO prototype.

The `edVO.Composer` is the module responsible for the creation of multimedia content, using the MPEG-4 description framework to compose objects in time and in space. It was based in previous work on user interfaces for multimedia authoring [3][2]. Figure 3(a) shows the GUI being used to create a presentation. The user is allowed to browse through the MOs stored in the server and to compose them in a MPEG-4 multimedia presentation. Composition is performed with the usual actions of visual manipulation of objects – grab, drag, move, rotate, etc. Object properties can also be changed individually with a property editor. Dynamic and interactive behaviour is associated with individual objects using its properties and the MPEG-4's routing mechanism. Scripting of the authoring process is also being developed. The connection with the server is established using XML-RPC as the communication protocol. XML-RPC is a remote procedure calling specification that uses HTTP as transport protocol, and XML as the encoding format. XML-RPC is designed to be as simple as possible allowing at the same time the transmission and processing of complex data structures. The choice for XML-RPC as structured communication protocol was mainly due to its lightweight implementation, as opposed to, for example, CORBA.

The `edVO.Server` module is responsible for the storage of MPs and MOs and provides the mechanisms to deliver content to the client. As expected, the entities defined as Repository and Multimedia Server in Figure 1 are instantiated by this module. It is partly implemented in Python and uses Zope as the application engine. Commands and requests from other modules are sent to the server using XML-RPC or HTML (cf. Figure 2) and are handled by Python scripts. Each MO is stored in its original encoding format and has associ-



Testing
edVO.Composer

string	Testing\edVO.Composer
length	[...]
fontStyle	(defined)
maxExtent	0.00

(a) Composer

9c1d328e-37e1-11d9-fc43-0050ba2c4108
a3eb252a-37e1-11d9-fc43-0050ba2c4108
acb36eb0-37e1-11d9-fc43-0050ba2c4108

```
metadata
  DescriptionMetadata
    id              0
    Confidence      1.0
    Version         1.1
    LastUpdate      2004-07-09T03:20:25+09:00
    PublicIdentifier
      type  UUID
      value 098f2470-bae0-11cd-b579-08002b30bfeb
    PrivateIdentifier completeDescriptionExample
    Creator
      Role
        href creatorCS
        Name
          value Creator
      Agent
        Name
          GivenName
            value Luis
          FamilyName
            value Teixeira
```

(b) Server

Fig. 3 User Interfaces in edVO. While the composer is a stand-alone application, the server is a web application.

ated metadata. Media-specific parsers are used to extract structural metadata when a MO is imported to the server and can be added as plugins. Variations to a MO can also be added and are used to support content adaptation. For example, a MO containing a video can be alternatively represented by another video with less resolution or by an image containing just a keyframe. All these versions are stored alongside the original content as shown in Figure 4.

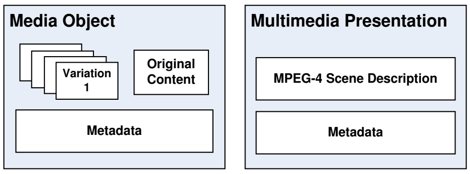
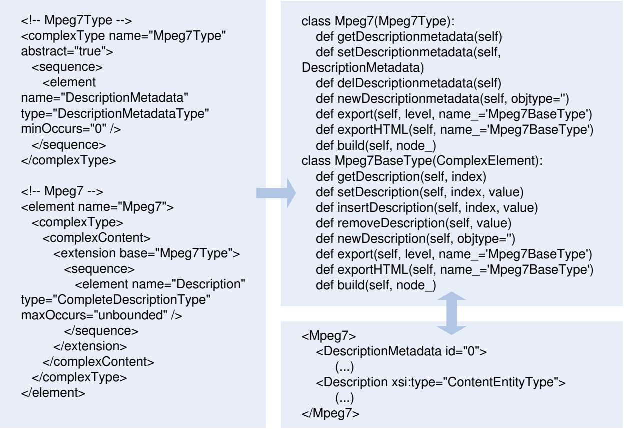


Fig. 4 MO and MP storage in server.

Information about the variations, like quality parameters defined by MPEG-7, are stored in the metadata. Each MP is, on the other hand, stored as textual MPEG-4 scene description using XMT-A. A subset of MPEG-7 MDS tools is used for the description of both MOs and MPs. This subset includes the tools defined by the MPEG-7 Core Description Profile (CDP) and the Variation description tools, which are not part of the CDP.

The handling of content description is processed in several steps. Firstly, the descriptive elements are defined through a XML schema. A programmatic data model in Python is then automatically generated that fully complies with the schema. The data model consists of classes and methods that programmatically handle metadata files in XML. Figure 5 shows an example of how an XML schema is coded in the Python data model.



```
<!-- Mpeg7Type -->
<complexType name="Mpeg7Type"
abstract="true">
  <sequence>
    <element
name="DescriptionMetadata"
type="DescriptionMetadataType"
minOccurs="0" />
  </sequence>
</complexType>

<!-- Mpeg7 -->
<element name="Mpeg7">
  <complexType>
    <complexContent>
      <extension base="Mpeg7Type">
        <sequence>
          <element name="Description"
type="CompleteDescriptionType"
maxOccurs="unbounded" />
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</element>
```

```
class Mpeg7(Mpeg7Type):
   def getDescriptionmetadata(self)
   def setDescriptionmetadata(self,
DescriptionMetadata)
   def delDescriptionmetadata(self)
   def newDescriptionmetadata(self, objtype='')
   def export(self, level, name_='Mpeg7BaseType')
   def exportHTML(self, name_='Mpeg7BaseType')
   def build(self, node_)
class Mpeg7BaseType(ComplexElement):
   def getDescription(self, index)
   def setDescription(self, index, value)
   def insertDescription(self, index, value)
   def removeDescription(self, value)
   def newDescription(self, objtype='')
   def export(self, level, name_='Mpeg7BaseType')
   def exportHTML(self, name_='Mpeg7BaseType')
   def build(self, node_)
```

```
<Mpeg7>
   <DescriptionMetadata id="0">
      (...)
   <Description xsi:type="ContentEntityType">
      (...)
</Mpeg7>
```

Fig. 5 Relations between XML Schema, Python data model and XML metadata.

When the metadata is read, each corresponding class is automatically populated. Modifications using get/set/del methods can then be performed and stored. These classes also generate an HTML/JavaScript User Interface (UI) that allows users to manipulate metadata (exportHTML method). Figure 3(b) shows this UI in action. The main advantages of this method are: flexibility and technology-independence. Any change in the descriptive scheme is easily integrated in the already working system. Providing that no backward-incompatible change is performed, a simple data model rebuild suffices.

Currently, edVO uses mainly semantic information that is inserted by the authors. But, besides semantic information, more low-level description of content can easily be used, providing fine-tune controls to the search and adaptation engines. This represents an important difference when compared to the previous works [1] [10].

3 Usage scenario: adaptation

It is known that the burden of content adaptation can be lightened when additional information is available alongside the multimedia content [8]. A simple terminal and network driven adaptation scenario was implemented in the edVO system proving its flexibility. When a presentation is requested, the edVO system generates dynamically an MP4 file ready to be delivered that complies to the restrictions set by the current context of usage – network restrictions, device capabilities. The resulting MP4 file is based on the presentation scene description and the referenced media objects as depicted in Figure 6. Currently, the system only follows three options regarding each object: preserve the object as is, choose one of the variations associated with the object or remove the object and its references in the presentation. Other possible operations include transcoding or transmoding the media object content, using specific modules, and can be added as plugins to the system.

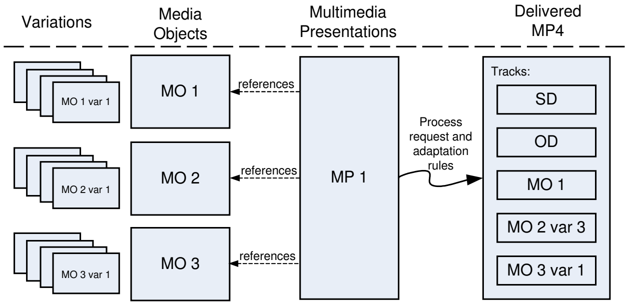


Fig. 6 Content adaptation process.

The adaptation is performed by an *adaptation engine*. To accomplish its task some rules need to be associated with the presentation, namely how objects are selected to be replaced by an alternate variation or removed – *adaptation rules*. Hence, the adaptation engine takes into account three sources of information to fulfill a request as shown in Figure 7 – the presentation scene description, including the media objects referenced by it, the constraints determined by the context and a set of adaptation rules.

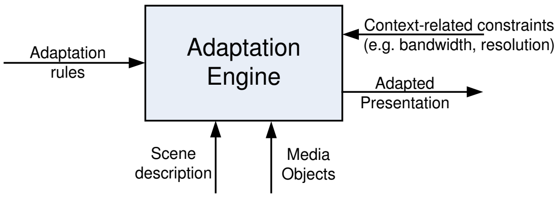


Fig. 7 Adaptation engine.

The first source of information is provided by the client and the second is stored in the edVO system. The adaptation rules, the third source, can be defined by a set of adaptation descriptors associated to each presentation. However, the MPEG-7 standard does not define such adaptation descriptors. On the other hand, the MPEG-21 standard proposes a set of descriptors regarding Digital Item Adaptation (DIA) [6] that can be used for that purpose, with some modifications. However, only user preferences are considered. For that reason, adaptation rules defined by authors can be tested using an additional simple non-standard descriptive scheme that allows authors to define, for example, modality conversion restrictions and presentation priorities. Regarging the context-related constraints, the system uses some of the standard descriptors proposed by MPEG-21 DIA – namely, Terminal capabilities and Network characteristics. Other adaptation scenarions can also be implemented, adding the appropriate descriptors.

4 Conclusions

In the proposed authoring system Both representation and description information are collected and aggregated during the creation process and provided to the distribution and consumption chains. The association of description with multimedia authoring will allow in the future the emergence of a semantic system. Semantic authoring allows content to be adapted, regarding available resources (specially important in heterogeneous systems with different network topologies and terminal characteristics). This work is a further step toward this goal.

Acknowledgement: This work had the financial support of Fundação para a Ciência e a Tecnologia/POSI (Portuguese Science and Technology Foundation).

References

1. M. Bertini, R. Cucchiara, A. D. Bimbo, and A. Prati. An integrated framework for semantic annotation and transcoding. *Multimedia Tools and Applications*, 26(3):345–363, August 2005.
2. S. Boughoufalah, J.-C. Dufourd, and F. Bouilhaguet. MPEG-Pro, an authoring system for MPEG-4. In *Proceedings of The 2000 IEEE International Symposium on Circuits and Systems*, pages 465–468, May 2000.
3. P. Daras, I. Kompatsiris, T. Raptis, and M. Strintzist. An MPEG-4 tool for composing 3D scenes. *IEEE Multimedia*, 11(2):58–71, April-June 2004.
4. ISO/IEC. Information technology – Coding of Audio-Visual Objects – Part 1: Systems, 2001.
5. ISO/IEC. Information technology – Multimedia content descr. interface – Part 5: Mult. descr. schemes, 2003.
6. ISO/IEC. Information technology – Multimedia framework – Part 7: Digital Item Adaptation, 2004.
7. R. Koenen. MPEG-4 multimedia for our time. *IEEE Spectrum*, 36(2):26–33, February 1999.
8. J. Magalhaes and F. Pereira. Using MPEG standards for multimedia customization. *Signal Processing: Image Communication*, (19):437–456, 2004.
9. J. Martnez, R. Koenen, and F. Pereira. MPEG-7: the generic multimedia content description standard, part 1. *IEEE Multimedia*, 9(2):78–87, April-June 2002.
10. T. Tran-Thuong and C. Roisin. Multimedia modeling using MPEG-7 for authoring multimedia integration. In *Proc. of ACM SIGMM Int. Workshop on Multimedia Information Retrieval*, pages 171–178, November 2003.