Structuring Interactive TV Documents

Rudinei Goularte
Department of Computing - ICMC
University of São Paulo at São Carlos
P.O. BOX 668, 13560-970 Brazil
+55 16 273 8161
rudinei@icmc.usp.br

Edson dos Santos Moreira
Department of Computing - ICMC
University of São Paulo at São Carlos
P.O. BOX 668, 13560-970 Brazil
+55 16 273 9699
edson@icmc.usp.br

Maria da Graça C. Pimentel
Department of Computing - ICMC
University of São Paulo at São Carlos
P.O. BOX 668, 13560-970 Brazil
+55 16 273 9657
mgp@icmc.usp.br

ABSTRACT
Interactive video technology is meant to support user-interaction with video *in scene* objects associated with navigation in video segments and access to text-based metadata. Interactive TV is one of the most important applications of this area, which has required the development of standards, techniques and tools, such as MPEG-4 and MPEG-7, to create, to describe, to deliver and to present interactive content.

In this scenario, the structure and organization of documents containing multimedia metadata play an important role. However, the Interactive TV documents structuring and organization has not been properly explored during the development of advanced Interactive TV services.

This work presents a model to structure and to organize documents describing Interactive TV programs and its related media objects, as well as the links between them. This model gives support to represent contextual information, and makes possible to use relevant metadata information in order to implement advanced services like object-based searches, in-movie (scenes, frames, in-frame regions) navigation, and personalization. To demonstrate the functionalities of our model, we have developed an application which uses an Interactive TV program's documents descriptions to present information about in-frame video objects.

Categories and Subject Descriptors
H.5.4 [**Information Interfaces and Presentation (I.7)**]: Multimedia Information Systems – *video, navigation;* I.7.2 [**Document and Text Processing (H.5)**]: Document preparation – *hypertext/hypermedia, markup languages*.

General Terms
Documentation, Design, Human Factors, Standardization, Languages.

Keywords
Media descriptions, Interactive TV, metadata, XLink, MPEG-7.

DocEng'03, November 20–22, 2003, Grenoble, France.

1. INTRODUCTION
The dissemination of interactive video technology has led to a large number of applications and forms of interaction, allowing for interaction with *in scene* objects; complex searches based on objects, complementing text-based searches; and automatic reaction in response to user interaction. Research into the area of interactive video has led to the development of standards, techniques, and tools to create, deliver and present interactive content, such as MPEG-4 [6, 19, 25], and to provide metadata for this content, such as MPEG-7 [20, 24]. These standards, as well as the strong interest shown by the communication and entertainment industries, is motivating the development of Interactive TV (I-TV) as one of the main interactive video applications.

The I-TV major benefit for users is the capacity of the video delivery chain to process the digital information to build new interaction paradigms. In this scenario, the structure and organization of documents containing multimedia metadata play an important role. In particular, it is expected the construction of personalized programs, specifically designed to fit the needs of each user. In an ideal situation, every user would expect to have a TV program at a time of his choice, with a duration that corresponds to the time that this user has available, and with content that specifically matches this user's interests [23]. In addition, programme and sub-programme selection features and advanced object-based interactivity are expected [10, 29].

On the other hand, as computing becomes ever more pervasive and ubiquitous, users expect to be able to interact with services and applications anywhere, anytime, regardless of the device in use. The ubiquitous computing, through its context-aware computing sub-area, explores human-computer communication during interactions by taking advantage of inherent contextual information to provide better services to the user [2]. Moreover, the contextual information can be used by applications to allow for advanced I-TV services, matching the content with users' preferences and/or devices' capabilities.

One of these services is the provision of personalized I-TV programs supporting object-based interactivity. This kind of service needs both multimedia objects (like video) to be composed by sub-objects, and documents describing the program and the components objects of the programs. These descriptions include: structural composition, media features (like size and coding method), index information (like author and creation date), links and relationships between objects and programs, and contextual information (like the position of the objects on the screen, the location where a scene took place, or who is in the scene) [10, 23, 27, 29, 30].

The composition of media objects using sub-objects is fully supported by the MPEG-4 standard. This standard specifies ways to compose interactive multimedia presentations arranging objects into a scene. These ways are based on the MPEG-4 native textual BIFS [25] (low level way) or based on SMIL [34] (high level way). The SMIL-based approach is translated to BIFS. Besides that, the MPEG-4 provides advanced standardized methods for encoding, compression, delivery and content interactivity of multimedia presentations [25]. These methods are not found in standards like SMIL, Flash [22] or QuickTime [4].

In spite of the power for composition and segmentation of objects provided by the MPEG-4, the models to represent I-TV programs and related media objects found in the literature [8, 9, 10, 18, 23, 28, 32] present: strict hierarchical relationships between media objects, low levels of metadata granularity, do not allow for programs and media objects descriptions to be separate and do not describe objects beyond the frame level. Besides that, few investigations have focused on integrating context-aware computing and interactive video.

In this paper we present a model to structure and to organize documents describing I-TV programs and its components media objects, as well as the links between them. This model is more flexible than others found in I-TV related works, giving support to represent contextual information, and making possible the implementation of I-TV services like automatic generation of summaries and indexes, object-based searches, in movie (scenes, frames, in frame regions) navigation, and personalization. Towards the development of a complete I-TV system, in this work we explore the model's documents organization building an application which uses the metadata descriptions to support object-based interactivity. The application uses the descriptions to present information about video objects, in-frame objects in special.

This paper is organized as follows: section 2 discusses related work, presenting the differences of our approach. Section 3 presents an application, developed using our model, which allows interact with in-frame video objects. The model to create structured descriptions of media objects and I-TV programs is detailed in section 4, explaining the proposed schemas, how they describe and segment media objects, and how they represent contextual information. Section 5 explains how media objects and programs are linked, and section 6 presents conclusions and future work.

2. RELATED WORK

Most of the earlier approaches on describing multimedia content (as in Benitez et al. [7], Dublin Core Metadata Initiative [11] and Lagoze & Hunter [21]) were covered by the MPEG group in the form of an international open ISO standard – the MPEG-7. This standard is a toolbox of generic description structures with associated semantics which is meant to be instantiated by any multimedia document that requires the use of metadata. While having such schemas standardized is useful to maximize interoperability, the scopes and possible application domains are so broad that it is unrealistic to provide an exhaustive set. Consequently, MPEG-7 has been made extensible by means of its Description Definition Language (DDL) [26], so that application developers can extend the standard description schemas to meet their specific requirements. Similarly, it is very unlikely that a single application will need the whole set of powerful and sometimes complex tools specified in MPEG-7. As a result, a practical way to use the standard, which is used here, is to make a selection of the description structures needed by the application and to validate them against specific target functional requirements.

In spite of the standard's power, the direct use of the MPEG-7 descriptors and description schemas can generate unnecessary very complex description documents, called just descriptions. Moreover, the support for contextual descriptions is limited, providing poorly structured free textual annotations. Our proposal creates a high level wrapper, with contextual descriptions support, in order to structure and to organize media objects descriptions. This wrapper abstracts all media types into objects, and the particular features of each object are described using very carefully selected MPEG-7 descriptors and description schemas, reducing the descriptions complexity. The wrapper and the MPEG-7 descriptors and description schemas are integrated using the MPEG-7 DDL, making our proposal MPEG-7 compliant.

The problem on how to produce specific documents for the I-TV area was addressed by a number of works [8, 9, 10, 17, 23, 28]. Most of these works fail in to provide an adequate combination of metadata and structural representations for both programs and media objects. An important exception is the work being developed by the TV-Anytime Forum international body [33], which aims to standardize Interactive TV. The TV-Anytime approach uses MPEG-7 standard to describe media objects. However, the TV-Anytime specifications do not provide structured context support, and do not provide the means to produce documents describing media objects and programs in an independent way, making difficult to reuse these documents. Another problem is the way in which TV-Anytime uses the MPEG-7 standard to describe media: it's not possible to describe in-frame objects [33]. In contrast, our approach allows both program and media description separation and description of in-frame objects.

Another problem, related to I-TV documents, is how to link programs. The TV-Anytime's solution creates an URI [33] that points to a single big object representing the program (a video, most of the time). The linking is made putting this URI into the program's description and into program guides [33]. In this approach the program segmentation into sequences, scenes and frames is made just in a logical way, through the MPEG-7 program's media descriptions. This makes difficult to reuse program's segments and related descriptions.

The combination of the MPEG-4 and MPEG-7 standards allow the physical segmentation of programs' media with interactivity and delivery advantages. However, until the time this paper was being written, the committees had not standardized a way to link an MPEG-4 object to its related MPEG-7 description yet.

Our approach segments each media into a set of MPEG-4 objects with related MPEG-7 descriptions. The programs are composed by pointing to the desired segments, easing the segment's reuse. In order to link programs, objects and descriptions we have developed a system which uses the linking information present in the model's instances documents. This linking information is based on the XLink W3C recommendation [37].

3. THE EXAMPLE APPLICATION

The application streams an I-TV program from a server to a client player with MPEG-4 support (we are using Real One player with EnvivioTV MPEG-4 plug-in [14]). The program is composed by an MPEG-4 video and annotations. The video is segmented into a number of MPEG-4 video objects (scenes, frames and in-frame objects). The annotations are documents (instantiated from our model) describing the program and each one of its objects.

While the program is playing, users are allowed to interact with the video's objects by clicking on them. MPEG-4 allows two types of responses for mouse events (and for events in general): pre-defined actions like changing the object's size, or a new programmatic action developed through the MPEG-J API [25].

The MPEG-J API allows Java programs, named MPEGLets, to be delivered with MPEG-4 presentations and interact with them. MPEGLets are encoded as MPEG-4 elementary streams and encapsulated, with other MPEG-4 I-TV program's objects, into an MPEG-4 file. This file, containing the I-TV program and the MPEGLet, will be sent to the user's player. At the user's player the MPEGLet will be decoded and executed at the client's Java Virtual Machine. More information about MPEG-4 encoding process and MPEG-J API can be found at [5, 6, 12, 19, 25].

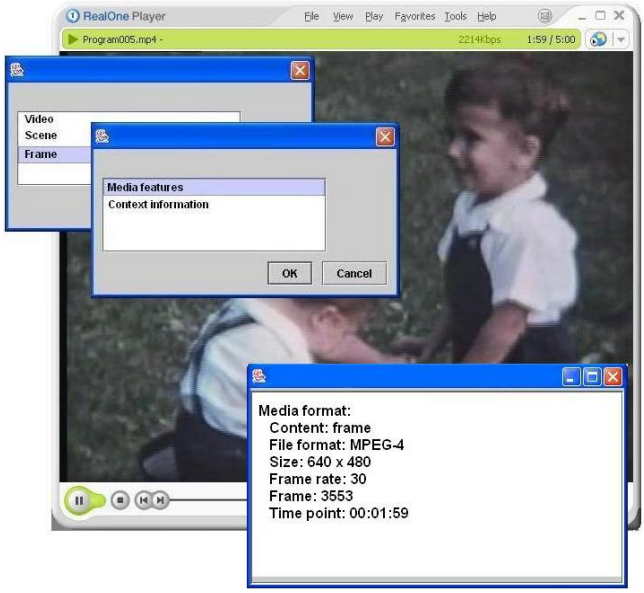

Figure 1. Accessing document's media information by interacting with video objects.

We use MPEG-J API to implement methods to: handle mouse events on the I-TV program's objects, and, to access the documents describing the selected object in response to mouse events. A mouse right-click pops-up a window which allows the selection of the media's structural level: video, scene or frame. After selecting a level, another window pops-up. This new window allows choosing the type of description: media features or context information. In the example illustrated in Figure 1 the user have selected "frame level" and "media features", and the frame's media features were presented to him in the window at the bottom of Figure 1.

Figure 2. Contextual information about an in-frame object.

A mouse left-click pops-up a window containing contextual information about the selected in-frame abject. For example, Figure 1 shows two kids playing: Peter at the left and John at the right. A mouse-click on John's image will retrieve the contextual information annotated with the MPEG-4 object representing John, as illustrated in Figure 2.

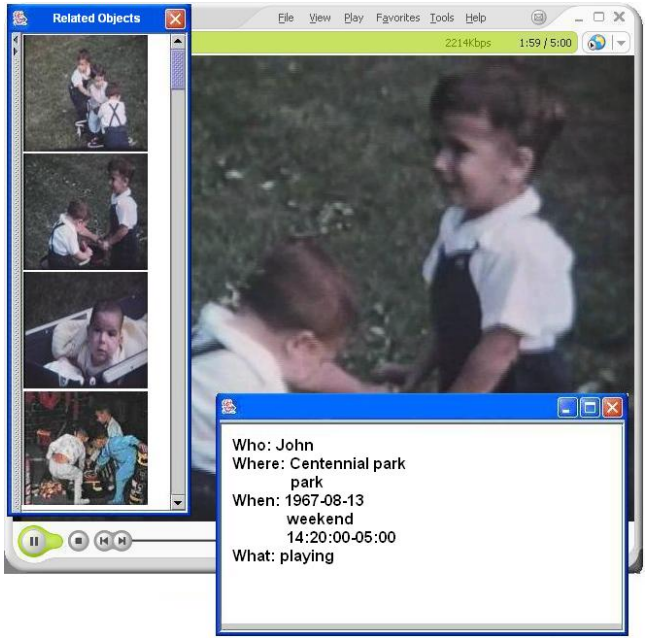

Figure 3. Service suggesting related objects.

All documents describing programs, objects and linking information are instantiated from the model's schemas (section 4) and, in this way, are in an XML [36] format. We have used the Xerces API [3, 35] in order to build an application which processes these documents and retrieves relevant information. The processing of linking information is made using Fujitsu XLink Processor [15]. The application is called from inside an MPEGLet when the user clicks on an MPEG-4 object, presenting information inside an appropriate window.

As all documents describing the program and its related objects are sent to the client to be locally manipulated, it is possible to generate dynamic indexes pointing to any set of program's objects, in addition to the object being presented. In the example illustrated in Figure 3 an index to scenes related with the selected in-frame object (John) was generated and associated with key-

frames (the list of pictures at the left). A mouse-click on one of these key-frames will start the playing of the associated scene. In order to generate this index, the application just traverses the documents analyzing the objects' descriptions looking for matches. In this example the application will look in "Who" and "What" clauses of the contextual descriptions related with John (like: "John", "John playing", "Peter and John", "Baby watching John and Peter"). After selecting the objects, the application just gets the related information to build a link to the specific point in time where each scene takes place. The traversing from a document to another is facilitated by the links between programs and objects and between objects (section 4).

The advantage of this kind of structuring for searching and navigation is that searches about the program being played are executed locally. However, the search approach is not limited to local processing neither to the program being presented. In fact, the last key-frame in Figure's 3 list is related to a remote program, located at the server. In this application we have limited the search results to three local scenes and one remote scene.

This kind of services is useful to support automatic personalization. In these services, applications must choose the best content to be presented based on the user's preferences or based on the users' device constraints. In the next section we present the schemas compounding our model, which give necessary support to build the application with object-based interactive and context-aware features.

4. STRUCTURED DESCRIPTION FOR MEDIA OBJECTS

4.1 Context Namespace

One of the challenges in the area of context awareness is how to represent context. Abowd et al. [2] argue that, without good representations for context, application developers are left to develop ad-hoc and limited schemas for storing and manipulating this key information.

Previous work have categorized context in the five dimensions: *who, where, when* and *what* [1]. This categorization is a guideline helping application designers to decide what information is context [2]. In the capture and access ubiquitous computing sub-area some authors include the *how* dimension, representing the methods used to capture and to access contextual information [31]. However, developers still have to analyze the application requirements and decide how to model and represent the context information for each dimension. The requirements for contextual information differ from one to another, making it very difficult to cover all the possibilities of context in a self-contained document schema. Similar problem was addressed by the MPEG-7 standard in the multimedia metadata representation. So, similarly, a solution to the context representation is to provide an extensible library of contextual elements in the form of a XML Schema namespace [27]. Developers can use the elements present in the namespace as they are, or, developers can use the library to build their own specific elements.

Table 1 lists some examples of the elements' types present in our library. Each type corresponds to one element. In the same way that complex types were built using simple types, developers can use simple or complex types in order to build their elements. For example, a type representing the users' history could be: userHistoryType={userIDType, fullNameType, userActionType, dateType, simpleTimeType}. An exhaustive list of the types and detailed explanations about each element are outside the scope of this paper.

Table 1. Examples of types present in the library.

Dimensions	Complex built-in types	Simple built-in types
Who		userIDType, roleType, simpleNameType, … .
	fullNameType	givenNameType, middleNameType, familyNameType.
	personal InformationType	ageType, heightType, weightType, eyesType, hairType, birthDateType.
Where		locationIDType, locationNameType, locationDescriptionType, … .
	indoor LocationType	floorType, roomType, corridorType, gateType, exitType.
	postal AddressType	streetType, numberType, complementType, stateType, countryType, zipCodeType.
	electronic LocationType	urlType, e-mailType, phoneType, icqType.
When		occasionType, dateType, simpleTimeType, durationType, … .
What		userActionType, userActivityType, … .

4.2 The MediaObject Schema

The *MediaObject* Description Schema describes objects such as video, audio, images and animations, and establishes the link between the described object and the program that contains the object. In the definition of the Description Schema, the hierarchy-based structure proposed by Benitez et al. [7] was used to organize objects. We have structured the schema to support the *Context Namespace* described in section 4.1.

Moreover, to support a linking structure between objects and descriptions we use the XLink standard [37], as detailed later in section 5.

Figure 4 shows a graphical representation of the *MediaObject* Description Schema using the UML notation. The diamond symbol represents a composition relationship. The range associated with each element represents frequency in the

composition relationship (e.g., *"0..*"* means zero or more and *"1..*"* means one or more).

The *MediaObject* Description Schema, represented in Figure 4 by the <*MediaObject*> element, consists of an <*ObjectSet*> element, zero or more <*ObjectHierarchy*> elements, and zero or more <*ContextHierarchy*> elements. The purpose of this Description Schema is to represent a given media as a set of objects, which are interconnected via both hierarchical and non-hierarchical relationships, from now on referred to simply as relationships. The hierarchies allow for the specification of multiple abstraction levels supporting the indexation and visualization of objects based on media and/or context characteristics. The explicit use of hierarchy in a Description Schema has the following advantages [7]: it is a more efficient retrieval structure than graphs; a hierarchy is the most natural way to define object composition; MPEG-4 objects are built hierarchically.

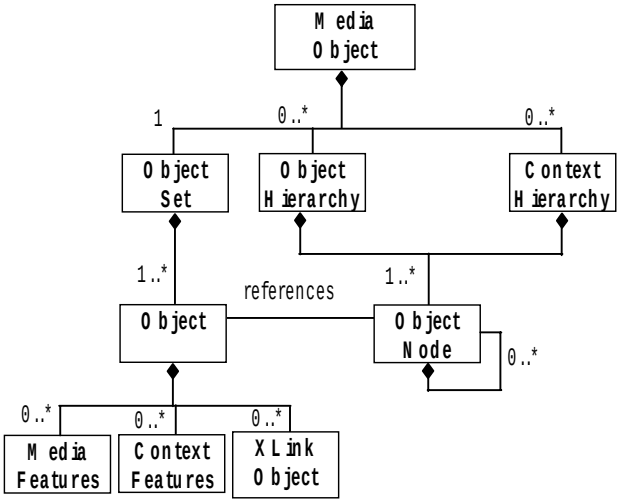

Figure 4. UML Representation of the *MediaObject* schema.

4.2.1 Object Composition

An object can be composed of (or segmented by) many other objects. For example, a video can consist of a set of scenes, which can be composed of a set of frames. In order to represent this composition, all the objects that are part of the media object to be represented (a video, for example) and are encapsulated in the <*ObjectSet*> element, and hierarchically structured and referenced into an *ObjectHierarchy* element. This hierarchical organization arranges the representation of the segmentation of the media objects.

The media objects are represented by the *Object* element and can have *Global*, *Segment* or *Local* scope, meaning, for example, a video or an image, a set of video frames, and a region inside a video frame or image, respectively.

Each <*Object*> element has the following attributes, illustrated in Figure 5:

- A unique identifier (id).
- An object scope identifier (scope).
- Object type (type, which can be video, audio, image, etc.).

```
<ObjectSet>
 <!-- whole video-->
 <Object type="VIDEO" scope="GLOBAL" id="ID001">…</Object>
 <!-- scene 1-->
 <Object type="VIDEO" scope="SEGMENT"
                          id="ID002">…</Object>
 <!--  video frame -->
 <Object type="VIDEO" scope="SEGMENT"
                          id="ID003">…</Object>
 <!-- John-->
 <Object type="VIDEO" scope="LOCAL" id="ID004">…</Object>
 <!-- Peter-->
 <Object type="VIDEO" scope="LOCAL" id="ID005">…</Object>
 <!-- scene 2-->
 <Object type="VIDEO" scope="SEGMENT"
                          id="ID006">…</Object>
       …
</ObjectSet>
```

Figure 5. Segmentation of a media object.

Figure 6 shows how the <*ObjectHierarchy*> element is used in order to organize the <*ObjectSet*> element: each object hierarchy is a tree of <*ObjectNode*> elements that reference the object of the <*ObjectSet*> element.

```
<MediaObject id="MO010">
      <ObjectSet> … </ObjectSet>
      <ObjectHierarchy>
          <ObjectNode id="node000" ObjectRef="ID001">
             <ObjectNode id="node001" ObjectRef="ID002"/>
             <ObjectNode id="node002" ObjectRef="ID003">
                <ObjectNode id="node003" ObjectRef="ID004"/>
                <ObjectNode id="node004" ObjectRef="ID005"/>
             </ObjectNode>
             <ObjectNode id="node005" ObjectRef="ID006"/>
                 ...
          </ObjectNode>
      </ObjectHierarchy>
      …
</MediaObject>
```

Figure 6. Objects' Hierarchy.

Figure 6 also demonstrates how a hierarchy allows specifying a relation of containment from the child nodes to its parent node. For instance, the *node002* node (parent node) is composed of the *node003* and *node004* nodes (child nodes). These nodes make references (*ObjectRef*), respectively, to the *ID003*, *ID004* and *ID005* objects and represent the compositional relationship between the video frame and the *"John"* and *"Peter"* objects in figure 7.

Figure 7 shows the video frame represented in figure 5 as the <*Object*> element with the id="ID003" attribute. Figure 7, which illustrates how an <*Object*> element is composed, shows two people, John and Peter, talking while John is at Peter's right. This frame (Object A) is composed of two other objects: B, representing John; and C, representing Peter.

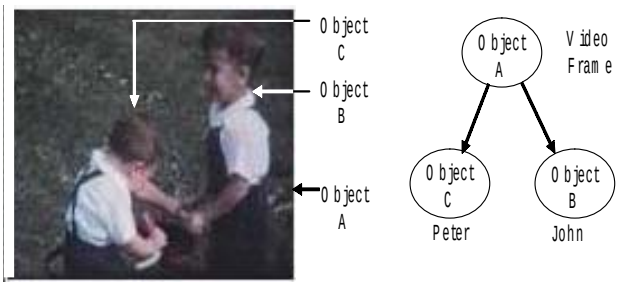

Figure 7. Objects compounding a video frame.

4.2.2 Media and Contextual Descriptions

Each object corresponds to an <*Object*> element, as illustrated in Figure 5. Each of these elements can contain contextual and media descriptions, which correspond to the <*ContextFeatures*> and <*MediaFeatures*> elements. The example depicted in Figure 7 contains three objects: object A, representing the video frame, and objects B and C, representing regions inside the video frame. Semantically, these objects represent an image of two children playing in a park. A description of the physical characteristics of the video frame (Object A) is given in Figure 8.

```
<ObjectSet> …
   <!-- Video frame-->
   <Object type=”VIDEO” scope=”SEGMENT” id=”ID003”>
      <MediaFeatures>
            <MediaFormat>
               <Content>frame</Content>
               <FileFormat>MPEG-4</FileFormat>
               <Size>640 x 480</Size>
               <FrameRate variable=”false”>30</FrameRate >
               <Frame>3553</Frame>
               <MediaTimePoint>00:01:59</MediaTimePoint>
            </MediaFormat>
       </MediaFeatures>
      <!-- contextual features of the video frame-->
       <ContextFeatures>... </ContextFeatures>
       <XLinkObject>... </ XLinkObject >
   </Object>
       …
</ObjectSet>
```

Figure 8. Description of the media characteristics of an object.

The contextual characteristics of object A can be described similarly by nesting a <*ContextFeature*> element inside the <*Object*> element. However, a more interesting example of contextual description, which uses the <*ContextHierarchy*> element to describe the contextual characteristics of Object B (John), is given next.

The <*MediaFeatures*> element consists of a set of MPEG-7 descriptors and description schemas for audio and video (*MediaFormatType*); image (*ImageType*); and 3D objects (*StillRegion3D*). These descriptors and description schemas provide flexibility to describe regions inside an image or a video frame (as in Figure 7). The example in Figure 8 uses <*MediaFormat*> (instantiated from *MediaFormatType*) to describe the video frame in Figure 7. Like most MPEG-7 description schemas, the *MediaFormatType* is complex. However, since most elements in that description schema are optional, there is room to make customized descriptions. The <*MediaFeatures*> element, employed here to describe a segment type object, can be used to describe global objects and even local objects (such as “John” and “Peter” in Figure 7).

```
<ObjectSet> …
      <!-- John’s description -->
      <Object type=”CONTEXT_FEATURES” id=”ID0015”/>
         <ContextFeatures>
            <Context>
               <Who> <Name> <GivenName> John <GivenName>
                        </Name>
                        <Role> Actor </Role>
               </Who>
               <Where><LocationName>Centennial park
                        </LocationName>
                        <OutDoorLocation> <Place> Park
                                 </Place> </OutDoorLocation>
               </Where>
               <When> <Date>1967-08-13</Date>
                        <Occasion> Weekend </Occasion>
                        <Time>14:20:00-05:00</Time>
               </When>
               <What>Playing</What>
            </Context>
         </ContextFeatures>
      </Object> …
</ObjectSet>
```

Figure 9. Description of the contextual characteristics of a local object.

Figure 9 shows the <*ContextFeatures*> element used as a wrap for the element defined in the context namespace, <*Context*>, in the example. This element uses the dimensions *What* and *When* provided by the namespace. The dimensions *Who* and *Where* were personalized using elements from the Context namespace.

```
<MediaObject id=”MO010”>  …
      <ContextHierarchy>
         <!-- John -->
         <ObjectNode id=”node025” ObjectRef=”ID004”>
            <!-- John’s description -->
            <ObjectNode id=”node026” ObjectRef=”ID015”/>
         </ObjectNode>
      </ContextHierarchy>
         …
</MediaObject>
```

Figure 10. Association between an object and its description.

An interesting example of contextual characteristics description is the description of the “John” object (B object in Figure 7). Contextual information, such as identity and location, can be described using the <*ContextFeature*> element inside the <*Object*> element representing the object. An alternative way is to create an <*Object*> node that contains the description (Figure 9), and to associate this object with the “John” object through the <*ContextHierarchy*> element (Figure 10).

Figure 10 shows the node “node26” referencing John’s description. This node is a child of “node25”, which references the object “John”, thus establishing a hierarchical relationship whereby “John’s description” is contained in “John”.

5. LINKING OBJECTS

An <*XLinkObject*> element is one of the <*Object*>'s components, as shown in Figure 4, which supports an underlying link structure associated with object descriptors. In other words, the aim is to allow for relationships between objects and descriptions.

As depicted in Figure 11, an <*XLinkObject*> element uses the XLink standard [37] to describe the link between objects in a *MediaObject* Description Schema, to establish relationships between the objects in a *MediaObject* Description Schema and to establish links between media objects and programs.

```
…
<Object type="VIDEO" scope="LOCAL" id="ID004">
      ...
  <XLinkObject xmlns:xlink="http://www.w3.org/2000/10/xlink-ns"
          xlink:type="extended xlink:title="Objects' relationships">
        <Locator
          xlink:type="locator"
          xlink:href="http://www.acme.org/ITV/MO010.xml#
                                          XPointer_expression"
          xlink:role=" http://www.acme.org/object"
          xlink:label="ID004"> <!-- John -->
        </Locator>
        <Locator
          xlink:type="locator"
          xlink:href="http://www.acme.org/ITV/MO010.xml#
                                          XPointer_expression"
          xlink:role=" http://www.acme.org/object"
          xlink:label="ID005"> <!-- Peter -->
        </Locator>
        <Relation
          xlink:type="arc"
          xlink:from="ID004" <!-- John -->
          xlink:to="ID005" <!-- Peter -->
          xlink:arcrole="http://www.acme.org/relations/Right_Of">
        </Relation>
        …
      </XLinkObject>
</Object>
…
```

Figure 11. Relationship between two objects.

Figure 11 describes the link between the objects "John" and "Peter" depicted in Figure 7, and the relationship between these two objects: John is at Peter's right:

- The <*Locator*> elements are locators in the context of the XLink standard and represent "John" and "Peter" objects.
- The <*Locator*> elements are associated with the objects through the object identifiers (IDs) as xlink labels (e.g., xlink:label="ID004").
- The xlink locator type uses an XPointer expression [38] (**XPointer_expression**, in Figure 11) to locate a specific portion of text inside a valid document (http://www.acme.org/ITV/**MO010.xml**, in Figure 11), where the document is a *MediaObject* Description Schema instance.
- The <*Relation*> element describes an xlink arc from the object ID004 ('John") to the object ID005 ("Peter"). The relationship between these two objects is described by the xlink *arcrole* attribute, indicating that object ID004 LetfOf object ID005, i.e., that John is on Peter's right.

A variety of relationships such as temporal, spatial, directional, topological, etc. can thus be established between objects in this way.

The link between an <*Object*> element and its media and context descriptions is done pointing a <*Locator*> element to related <*Object*>'s <*MediaFeatures*> and <*ContextFeatures*> elements.

The link between an MPEG-4 object and its descriptions is done as follows: the application presented in the section 3 gets the MPEG-4 object's ID during an interaction using MPEG-J API methods. During the scene composition phase, this ID must be assigned as the same ID of the related <*Object*> element in the descriptions. Based on this ID, the application can locate the correct <*Object*> element and its associated <*XLinkObject*> element. From this point, the parser loaded with the MPEGlet can process the xlinks and retrieve information, as illustrated in Figures 1, 2 and 3.

```
<ObjectSet> …
   <!-- Video frame-->
   <Object type="VIDEO" scope="SEGMENT" id="ID003">
      <MediaFeatures URL="./MF_ID003.xml"/>
      <ContextFeatures URL="./CF_ID003.xml"/>
      <XLinkObject URL="./XL_ID003.xml"/>
   </Object> …
</ObjectSet>
```

Figure 12. Elements pointing to separated description files.

It is important to observe that this approach allows the complete separation of link, media, context and structural descriptions. The given examples show all descriptions inside the same *MediaObject* instance XML file. However, it is possible to have each type of description in separated files instead of have them embedded into an <*Object*> element.

```
<?xml version="1.0" encoding="UTF-8"?>
<Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001"  … >
  <DescriptionUnit xsi:type="XLinkObjectType"
                                    ObjectRef="ID003">
    <XLinkObject xmlns:xlink="http://www.w3.org/2000/10/xlink-ns"
              xlink:type="extended> …
        <Locator
          xlink:type="locator"
          xlink:href="http://www.acme.org/ITV/MF_ID003.xml"
          xlink:role=" http://www.acme.org/description"
          xlink:label="MF_ID003">
        </Locator>
          …
    </XLinkObject>
  </DescriptionUnit>
</Mpeg7>
```

Figure 13. XL_ID003.xml file.

The <*MediaFeatures*>, <*ContextFeatures*> and <*XLinkObject*> elements, children of an <*Object*> element, have an optional URL attribute. This attribute points to separated XML *MediaObject* instances files, containing only media features, contextual or linking information (Figure 12). In this case, the <*Locator*> elements must point to these XML files and each one of these files has references to its related <*Object*> (Figure 13). The advantage of the separation is to facilitate the reutilization of descriptions. In this way, MPEG-4 objects can be reused to compose new objects and its related descriptions do not need to be edited in order to describe the new object's components.

The same approach is used to link an <*Object*> with its related I-TV program description, using a <*Locator*> element pointing to an external file. Figure 14 illustrates a case in which a link is established between the object identified as "MO010" ("from" attribute) and the program identified as "Program005" ("to" attribute).

```
<MediaObject id="MO010">
  ...
  <XLinkObject xmlns:xlink="http://www.w3.org/2000/10/xlink-ns"
          xlink:type="extended  xlink:title="Objects' relationships">
    <Locator
      xlink:type="locator"
      xlink:href="http://www.acme.org/ITV/MO010.xml"
      xlink:role=" http://www.acme.org/object"
      xlink:label="MO010"> <!-- The Object -->
    </Locator>
    <Locator
      xlink:type="locator"
      xlink:href="http://www.acme.org/ITV/Program005.xml"
      xlink:role=" http://www.acme.org/program"
      xlink:label="Program005"> <!-- The program -->
    </Locator>
    <Relation
      xlink:type="arc"
      xlink:from="MO010" <!-- Object -->
      xlink:to="Program005" <!-- Program -->
      xlink:arcrole="http://www.acme.org/relations/LinkObj2Prog">
    </Relation>
    …
  </XLinkObject>
</MediaObject>
```

Figure 14. Example of linking between an object and a program.

The *ITV_Program* Description Schema describes I-TV programs such as News, Sports, Series, etc. Figure 15 shows a graphic UML representation of the *ITV_Program* Description Schema using the same previous UML notation. The elements *Copyright*, *Classification*, and *Description* are imported from the TV-Anytime namespace. These elements describe the program's characteristics, such as the classification of the program (movie, news, social life, entertainment, etc.), which group this program is part of, the program's schedule, credits, reviews, genre, title, synopsis, languages, etc. A complete discussion of each of these elements and their sub-elements are outside the scope of this paper. Our aim here is to show how program descriptions are linked with media object descriptions. The linking is done through xLink, using the *ObjectsList* element.

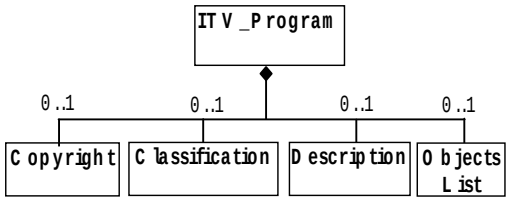

Figure 15. UML Representation of the *ITVProgram* schema.

The <*ObjectsList*> element is the set of all links, linking the program to each of its media objects. As shown in Figure 16, the first <*Locator*> element is representing the program. The subsequent <*Locator*> elements are representing program's objects. The <*Arc*> elements are arcs in the xlink context, establishing links between the programs and its component objects.

```
<ITV_Program ProgramID="Program005" version="0001"
                                    xml:lang="en" publisher="" …>
  ...
  <ObjectsList xmlns:xlink="http://www.w3.org/2000/10/xlink-ns"
        xlink:type="extended  xlink:title="Links to objects">
    <Locator
      xlink:type="locator"
      xlink:href="http://www.acme.org/ITV/Program005.xml"
      xlink:label="Program005">
    </Locator>
    <Locator
      xlink:type="locator"
      xlink:href="http://www.acme.org/ITV/MO010.xml"
      xlink:label="MO010">
    </Locator>
    ….
    <Arc xlink:type="arc"
      xlink:from="Program005"
      link:to="MO010" …>
    </Arc>
    ...
  </ObjectsList>
</ITV_Program>
```

Figure 16. Link from a Program to Objects.

6. CONCLUSIONS

This paper has presented a flexible way for linking and describing I-TV programs and media objects, providing support for context awareness. Structured descriptions of media objects are made through our proposed description schema, which uses MPEG-7 Descriptors and Description Schemas selectively. This is done by creating abstractions through the MPEG-7 Description Definition Language, which encapsulates selected MPEG-7 Descriptors and Description Schemas. In addition, the schema specifies a single description schema for any kind of media; provides support for structured contextual information; allows for the separation of media structure and description from the programs structure and description; and provides object segmentation beyond the scene level, allowing, for example, for descriptions about areas inside a video frame.

The support for contextual description is given through the *context library*, which is a namespace of context elements. The elements of the library are imported by our *MediaObject* schema, allowing for descriptions of semantic information about objects such as video, video scenes, video frames and areas inside a frame. This is useful, for instance, in searches for related material (suggestion of relevant material), previewing content before access, and in context-aware systems that use this kind of information to make adaptations [2, 16, 31].

Our model allows relationships to be described between media objects, in addition to hierarchical relationships inherent to segmented material. This kind of description is done through schema elements that are based on the XLink W3C recommendation. The links are enriched with semantic information about the content they indicate and about the relationships (spatial, temporal, directional, etc.) between indicated objects. This information helps in the selection of material of interest, facilitating the retrieval of context information about the content indicated by the links.

In ongoing work, we are implementing a context namespace for the system. We expect to use contextual information from the system, in addition to contextual information on the user, in order to automatically adapt services. These services, in the Interactive TV scenario, involve the personalization of both content and interface.

Acknowledgements

The authors would like to thank the Brazilian Funding Agencies CNPq and FAPESP. Rudinei Goularte was supported by FAPESP (98/15360-0) during his PhD. Maria Pimentel and Edson Moreira are supported by FAPESP and CNPq, in the context of the InCA-SERVE Project in Brazil -- jointly with Gregory Abowd who is supported by NSF in the U.S. The authors also would like to thank Gregory Abowd in providing the videos used in the implementation.

7. REFERENCES

[1] Abowd, G. D.; Dey, A. K.; Brown, P. J.; Davies, N.; Smith, M.; Steggles, P.: Towards a better understanding of context and context-awareness, In Gellersen, H. (ed.): Handheld and Ubiquitous Computing, Lecture Notes in Computer Science, Springer-Verlag, v. 1707, p. 304-307, 1999.

[2] Abowd, G. D.; Mynatt, E. D.; Rodden, T.: The Human Experience. Pervasive Computing. p. 48-57, January-February, 2002.

[3] Apache Software Foundation. Xerces Java Parser Readme. http://xml.apache.org/xerces-j/index.html.

[4] Apple Computers, Inc: QuickTime File Format. http://developer.apple.com/techpubs/quicktime/qtdevdocs/.

[5] Avaro, O.; Eleftheriadis, A.; Herpel, C.; Rajan, G.; Ward, L.: MPEG-4 Systems: Overview. Signal Processing: Image Communication, v. 15, n. 4-5, p. 281-298, 2000.

[6] Battista, S.; Casalino, F.; Lande, C.: MPEG-4: A Multimedia Standard for the Third Millennium, part 1, IEEE MultiMedia, v. 6, n.4, p. 74–83, 1999.

[7] Benitez, A.B.; Paek, S .; Chang, S.; Puri, A.; Huang, Q.; Smith, J. R.; Li, C.; Bergman, L. D.; Judice, C. N.: Object-based multimedia content description schemes and applications for MPEG-7. Signal Processing: Image Communication, v.16, n 1-2, p. 235-269, 2000.

[8] Boll, S.; Klas, W.; Wandel, J.: A Cross-Media Adaptation Strategy for Multimedia Presentations. In proceedings of 7th ACM International Conference on Multimedia, p. 37-46, 1999.

[9] Chorianopoulos, K.; Lekakos, G.; Spinellis, D.: Intelligent User Interfaces in the Living Room**:** Usability Design for Personalized Television Applications. In proceedings of the ACM 2003 International Conference on Intelligent User Interfaces, p. 230-232, 2003.

[10] Cosmas, E.; Parker, Y.; Schoonjas, P.; Vaduva, A.; Dosch, C.; Schäfer, R.; Erk, A.; Mies, R.; Bais, M.; Schäfer, R.; Stammnitz, P.; Selinger, T.; Klunsoyr, G.; Pedersen, L.; Bauer, S.; Engelsberg, A.; Klock, B.; Evensen, G.: CustomTV with MPEG-4 and MPEG-7. Colloquium on interactive television, IEE, Savoy-Place, London, UK, 1999.

[11] Dublin Core Metadata Initiative. http://dublincore.org.

[12] Ebrahimi, T.; Horne, C.: MPEG-4 Natural Video Coding - an Overview, Signal Processing, Image Communication, v. 15, n 4-5, p. 365–385, 2000.

[13] Envivio MPEG-4 Encoding Station Software – Overview. http://www.envivio.com/products/ees.html.

[14] EnvivioTV MPEG-4 Player – Overview. http://www.envivio.com/products/etv/index.html

[15] Fujitsu XLink Processor. http://www.labs.fujitsu.com/free/xlip/en/index.html.

[16] Harrison, B. L.; Fishkin, K. P.; Gujar, A.; Mochon, C.; Want R.: Squeeze me, hold me, tilt me! An exploration of manipulative user interfaces. In proceedings of ACM Conference on Human Factors in Computing Systems (CHI 98), p. 17-24, 1998.

[17] Hjelsvold, R.; Vdaygiri, S.; Léauté, Y.: Web-based Personalization and Management of Interactive Video. In proceedings of 10th International World Wide Web Conferece, p. 129-139, 2001.

[18] Hu, M.J.; Ye, J.: MD2L: content description of multimedia documents for efficient process and search/retrieval. In proceedings of IEEE Forum on Research and Technology Advances in Digital Libraries, p. 200-213, 1999.

[19] ISO/IEC JTC1/SC29/WG11 N4668: Overview of the MPEG-4 Standard, 2001: http://mpeg.telecomitalialab.com/standards/mpeg-4/mpeg-4.htm.

[20] ISO/IEC JTC1/SC29/WG11 N4509: Overview of the MPEG-7 Standard, 2001: http://mpeg.telecomitalialab.com/standards/mpeg-7/mpeg-7.htm.

[21] Lagoze, C.; Hunter, J.: The ABC Ontology and Model (v3.0). Journal of Digital Information, v. 2 , issue 2, November, 2001.

[22] Macromedia, Inc.: Flash MX. http://www.macromedia.com/software/flash/. 2003.

[23] Merialdo, B.; Lee, K. T.; Luparello, D.: Automatic Construction of Personalized TV News Programs. In proceedings of 7th ACM International Conference on Multimedia, p. 323-331, 1999.

[24] Nack, F.; Lindsay, A.T.: Everything you wanted to know about MPEG-7: Part 1, IEEE MultiMedia, v. 6, n. 3, p. 65–77, 1999.

[25] Pereira, F., Ebrahimi, T. (eds.): The MPEG-4 Book. Prentice Hall PTR. 2002.

[26] Salembier, P., Smith, J. R.: MPEG-7 Multimedia Description Schemes. IEEE Transactions on Circuits and Systems for Video Technology, v. 11, n. 6, p. 748-759, June, 2001.

[27] Santos Jr., J.B. dos; Goularte, R.; Moreira, E. S.; Faria, G. B.: "The Modeling of Structured Context-Aware Interactive Environments". Transactions of the SDPS Journal of Integrated Design and Process Science, v. 5, n. 4, p. 77-93, December de 2001.

[28] Smyth. B.; Wilson, D.; O' Sullivan, D.: Improving the Quality of the Personalized Electronic Programme Guide. In proceedings of the 2nd Workshop of Personalization in Future TV at the 2nd International Conference on Adaptive Hypermedia and Adaptive Web Based Systems, p. 48-51, 2002.

[29] Srivastava, H. O.: Interactive TV Technology and Markets. Artech House, 2002.

[30] Stotts, D., Smith, J.: Semi-Automated Hyperlink Markup for Archived Video. In proceeding of ACM Hypertext 2002 (College Park, Maryland, USA), p. 105-106, June, 2002.

[31] Truong, K. N.; Abowd, G. D.; Brotherton, J. A. Who, What, When, Where, How: Design Issues of Capture & Access Applications. Lecture Notes on Comptuer Science 2201 - Ubicomp 2001: Ubiquitous Computing, p. 209-224. Atlanta, GA, USA. September de 2001.

[32] Tsinaraki, C.; Papadomanolakis, S.; Christodoulakis, S.: Towards a two-layered video metadata model. In proceedings of 12th International Workshop on Database and Expert Systems Applications, p. 937-941, 2001.

[33] TV-Anytime Forum: Specification Series: S3: Metadata (Normative),2001: ftp://tva:tva@ftp.bbc.co.uk/pub/Specifications/SP003v11.zip

[34] World Wide Web Consortium (W3C). Synchronized Multimedia Integration Language (SMIL 2.0). http://www.w3.org/TR/smil20/.

[35] World Wide Web Consortium (W3C): Document Object Model (DOM) Level 3 Core Specification. http://www.w3.org/TR/2002/WD-DOM-Level-3-Core-20020409/.

[36] World Wide Web Consortium (W3C): XML Schema Part 0: Primer: http://www.w3.org/TR/xmlschema-0/.

[37] World Wide Web Consortium (W3C): XML Linking Language (XLink) Version 1.0: http://www.w3.org/TR/xlink/

[38] World Wide Web Consortium (W3C): XML Pointer Language (XPointer) Version 1.0: http://www.w3.org/TR/xptr/.