Consiglio Nazionale delle Ricerche

ISTITUTO DI ELABORAZIONE DELLA INFORMAZIONE

PISA

Modeling Multimedia objects for content-based retrieval

G. Amato, G. Mainetto, P. Savino, F.Rabitti

Nota Interna B4-09

Febbraio 1996

Modelling Multimedia Objects for Content-Based Retrieval

Giuseppe Amato[1], Gianni Mainetto[2], Pasquale Savino[1] and Pavel Zezula[1]

[1]IEI–CNR, Via S. Maria 46, Pisa
[2]CNUCE–CNR, Via S. Maria 36,Pisa
{amato,savino,zezula}@iei.pi.cnr.it
G.Mainetto@cnuce.cnr.it

Abstract:

This paper reports the work-in-progress on the definition of an object-oriented data model tailored for multimedia applications within the HERMES project. The wide diffusion of multimedia applications that use CD quality audio, video, high quality images, etc. and the initial availability of multimedia databases lead to the need of finding suitable solutions for the retrieval and the manipulation of multimedia data. In this paper we present a multimedia data model that addresses the aspects related to data presentation, manipulation and content-based retrieval. The core of the model consists of three parts: the Multimedia Description Model, which provides a structural view of raw multimedia data; the Multimedia Presentation Model, whose main feature is the possibility of describing the temporal and spatial relationships among different media objects; the Multimedia Interpretation Model which allows to associate semantic interpretation with the multimedia objects stored in the database.

This work has been partly funded by European Union under ESPRIT Long Term Research Project HERMES, No 9141

1. INTRODUCTION

In the sixties, both the necessity of managing the large amount of persistent data needed from business applications and the continuous improvements of disk technology led to the definition of simple data models and made realistic the development of Database Systems (DSs). In the eighties, the need of supplying CAD/CAM, VLSI, CASE applications with repositories storing complex structured data and the possibility of distributing the burden of the computation on LAN based client–server architectures determined the development of Object–Oriented data models and Object–Oriented DSs. Today, both the presence of repositories containing huge amounts of multimedia data such as raster images, text documents, video data, scientific data and the improvement of several technologies that include large capacity storage devices (e.g., CD–Roms, juke–box, disk–array) ask for the definition of a multimedia data model and design of architectures well–suited for Multi Media Database Systems (MMDSs). MMDSs are strongly required in many new application areas such as merchandising, education, journalism, television.

The provision of MMDSs entails several fundamental issues of a DS, ranging from access methods and operating system support up to efficient multimedia query languages capable of expressing spatial–temporal concepts [Berra 93], [Ghafoor 95]. In this article,

we focus our attention on the structuring of a multimedia data model and its most relevant features.

We present a *Multimedia Data Model* that addresses the aspects related to presentation, manipulation and content-based retrieval of multimedia data. The core of the model consists of three parts: a *Multimedia Description Model* (MDM), which provides a structural view of raw multimedia data; a *Multimedia Presentation Model* (MPM), whose main feature is the possibility of describing the temporal and spatial relationships among different structured multimedia data; the *Multimedia Interpretation Model* (MIM) that allows to associate semantic interpretation with the structured multimedia data. The three data models are in one–to–one relationship with the three subsystems of a MMDS that have to accomplish the storage, retrieval and delivery of multimedia data.

The three data models are distributed between two abstraction levels: the physical level and the logical level[1]. At our lowest physical level, we place MDM that allows to give a structure to the unstructured raw multimedia data (that is data as stored in the storage server). These data are large in number and in size because they are rich of complex implicit information such as those represented from the so-called *continuous media* (for example, audio and video and audio/video). The peculiarity of these data makes necessary the use of ad-hoc techniques when we deal with the design and implementation of a multimedia storage server, that subsystem of the MMDS subsystem that implements MDM. The ad–hoc techniques have to take into account real time storage and retrieval, high data transfer rate, large storage space and low-level synchronisation among different multimedia streams.

At a higher logical level, multimedia applications need to interact with a layer of the MMDS that efficiently provides at least the following two features: (1) access to complex structured multimedia data such as books, periodicals, images, video clips, slide shows and computer–generated music; (2) interactivity, for browsing both in the database of multimedia data and in a single complex multimedia data item. For the sake of clearness, we distinguish two logical data models: MPM and MIM. The presence of two logical data models, built on the top of the same physical one, originally comes from the complex nature of multimedia data: the usual simple problem represented from the delivery of accessed data here becomes a complex problem that needs an ad–hoc *presentation* model. A presentation[2] of multimedia data is more complex than what is allowed in an ordinary database, because the end–users can interact with the presentation process and the data can have different dimensions and durations. The intrinsic complexity of presentation is put in evidence from the presence a "presentation programmer", usually called *author*, that produces *interactive presentations*. To construct an effective presentation, an author needs to find out, in a huge amount of useless data, those multimedia data that are better suited for the presentation. Thus he needs to issue queries against a world that gives meaning to multimedia data. The author is a user of the world that represents an *interpretation* of multimedia data. Similarly to presentation, an interpretation of multimedia data is usually much more complex than in traditional databases because the semantic content of stored multimedia data is often uncertain or can have multiple interpretations.

The description data model offers the following features:

1. description of raw multimedia data;
2. structuring by means of complex object, disjoint union, and bulk data constructors.

---

[1] Notice that physical and logical levels have been introduced to distinguish different levels of abstraction in the model. The physical level of the model should not be confused with the physical level of a DS.

[2] In the rest of the paper the term "presentation" will refer to the displaying of the content of a multimedia object. The presentation of a video is its playing; the presentation of an image is its show on the screen, etc.

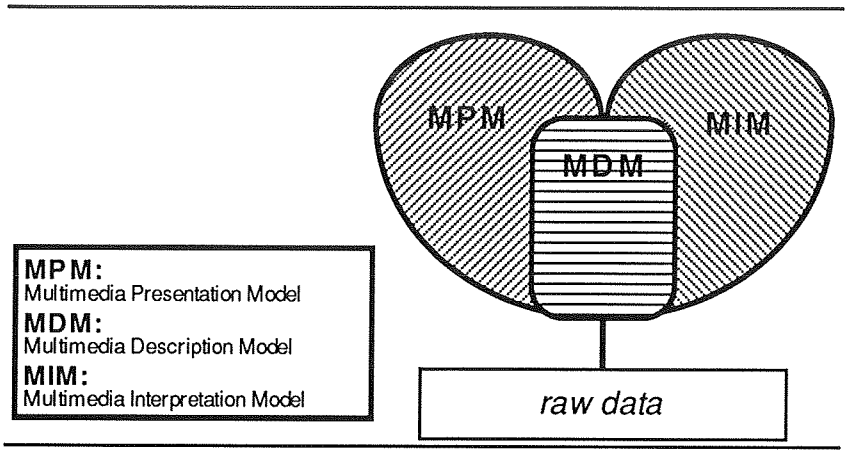

Figure 1 - Relationships among the models of a MMDS

The most important features of the presentation data model are:

3. object–oriented and event driven paradigms for dealing with media objects;
4. reusability of media objects;
5. specification of spatial and temporal constraints;
6. support for interactive presentations.

Finally, as far as the interpretation data model is concerned, the following aspects are particularly relevant:

7. representation of the multimedia data content in a form that the computer can manipulate;
8. temporal and spatial knowledge;
9. representation of background knowledge not directly contained in multimedia data items;
10. multiple interpretations of the multimedia data;
11. similarity based retrieval.

The rest of the paper is organised as follows. Section 2 contains an overview of the three models composing the Multimedia Representation Model. It also describes relationships between different models and illustrates how data of these models are connected with the multimedia raw data actually stored in the database. Sections 3, 4, and 5 introduce the three data models, respectively. Section 6 presents the conclusions and outlines our future work.

2. STRUCTURING THE MULTIMEDIA DATA MODEL

As mentioned in the introduction, the *Multimedia Data Model* is composed of three data models: the *Multimedia Description Model* (MDM), which provides a structural view of raw multimedia data; a *Multimedia Presentation Model* (MPM), which specifies how objects of the MDM have to be delivered respecting their temporal and spatial relationships; and the *Multimedia Interpretation Model* (MIM), which allows to specify the semantic interpretation of multimedia objects. MDM objects will be usually defined according to the needs of users of the other two models, the MPM and the MIM. Let us consider, for example, a video sequence and an audio sequence and let us suppose that synchronisation between them is necessary. Synchronisation is expressed by the MPM that refers to objects defined in the MDM. The MDM maintains a direct connection with the raw data objects. **Figure 1** sketches such view.

At the lowest level of representation, a *multimedia data* is any unstructured piece of information stored in the multimedia database. It can be acquired either from the real

world interfaces or from other existing multimedia databases. For example, a video sequence can be acquired through a video camera, an image can be digitised through a scanner, and so on. These are the "real" pieces of multimedia data and in the following we will call them *raw objects* (RO). These data do not contain any specification about their content but a portion of them contains information about their physical encoding and the remaining data is an unstructured linear stream of bytes.

A database can contain different kinds of ROs: Text, Audio, Video, Raster Image, Graphical Image, Audio/Video. Each RO is represented by a triple **(POID, Obj-attributes, Obj-constraints)** where:

- **POID** is the physical object identifier that uniquely identifies the object.
- **Obj-attributes** is a set of attributes that specify the characteristics of the object. These attributes are media dependent. For example, in a video object the Obj-attributes specify the coding format (for example MPEG-1 video), the duration, the creation date and time, etc.
- **Obj-default-constraints** is a set of parameters that specifies constraints for obtaining a normal presentation with the best quality. For example, in a video object the Obj-default-constraints may specify that the actual frame rate for playing the video is 20 frames/sec and the resolution is 640x480 and so on.

The *Multimedia Description Model* specifies the structure and the composition of all objects that the MMDS manages. In the Multimedia Description Model, the unstructured content of a RO can be conveniently structured by representing portions of it as *basic objects*, and then assembling such basic objects into a *complex object*. For example, when the presentation of a museum has to be faced, the author of the presentation can use images of paintings, texts representing critical reviews, parts of a video clip describing the techniques used for creations of paintings, parts of the image representing some important details and audio comments. This complex multimedia object has to be represented in the MMDS as well as basic objects representing portions of *raw data*. Objects of the Multimedia Description Model are those that can be retrieved, manipulated and delivered.

To express queries based on objects' content, the *multimedia interpretation model* should allow to express the semantic content of multimedia objects. The content of multimedia object is obtained by extracting features from multimedia streams and by associating them with a complex semantic description. Features are obtained using objects specified by the description model and queries are performed by using these features and their semantic description as arguments.

The third category of operations performed on the multimedia objects concerns object manipulation. This includes all operations for the presentation of objects but also operations that give support to the creation of specific presentations. This aspect is addressed in the *multimedia presentation model*.

3. MULTIMEDIA DESCRIPTION MODEL

Usually, we can view a RO as a collection of long, unstructured sequences of bytes, called BLOBs (binary large objects). Because of the large size of BLOBs, database systems offer special support for reading, inserting, deleting, and modifying BLOB data. BLOBs contain a lot of significant conceptual entities but they are all put together in an unstructured way that makes impossible to deal directly with them.

The Multimedia Description Model allows to organise raw data entities in *basic objects*. Each basic object has an identity *obj_id*, a type and a value, represented as a coding that specifies which raw data items and which portion of them are represented by the basic object. A basic object corresponds to a portion of the entire RO. This feature

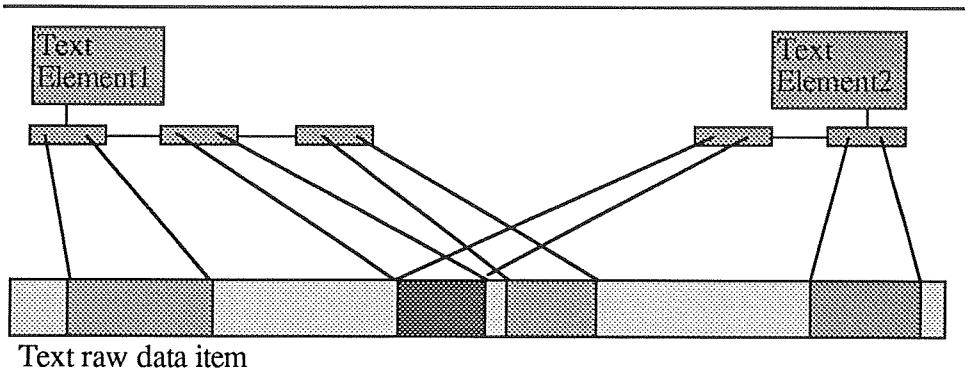

Figure 2 - Example of basic objects of Text Element type and relationships with a raw data item of Text type

allows to access the component of the RO in a structured way. Notice that the identity of a basic object is different from the identity of the component raw data item.

There is a great freedom of organising basic objects. In **Figure 2** is shown, how two basic objects of type *Text Elements* may share the same raw data item. Some suggestion of structuring time-based media can be found in [Gibbs 94].

The types of basic objects are the followings:

- **Text Elements** - A Text Element is any subpart of a text raw data item, that is a subsequence of a long sequence of (ASCII) characters. Text Elements can represent words, a set of keywords, sentences, paragraphs, and so on.
- **Image Elements** - An Image Element is a region of a raw image. If the image is represented in raster format, then the region is an area of any form in the image. On the contrary, if the image has a vector representation, then the Image Element is any group of graphical elements; in real situations they can represent objects contained in the image, such as a tree or a house [IEEE-ToSE 88], [Kunii 89] .
- **Video shots** - We consider a video as composed of a sequence of frames. A video shot is any subsequence of frames in the video. Usually, a video shot corresponds to a continuos sequence of frames from a video camera; however, the model does not impose any limitation on the choice of the frames composing the video shot.
- **Audio Shots** - A raw data item of type audio is composed of a long sequence of audio samples. An audio shot is any subsequence of audio samples.

The model also supports all conventional primitive data types: **string, character, real, integer,** etc. Naturally, a value of a primitive type is a basic object; so, in this way, all values in the MMDS can be considered as basic.

Now that every value is a basic object of a single media type, the MDM can provide the structure of multiple media objects by means of three classes of type constructors corresponding to three different abstraction mechanisms of the data model: *complex object* constructor, *disjoint union* constructor and *bulk object* constructors. All these object constructors can be applied orthogonally each other, the number of times that is necessary until a satisfactory description of the database of raw data is reached.

Complex objects are usually multiple media objects, created by aggregating single media objects (i.e. basic objects) and/or other complex objects. The abstraction mechanism that is used is the aggregation. It takes as input a non–empty set of pair <label,obj_id> and returns an aggregate object made of a number of attributes equal to the length of the sequence and containing as values the set of the object identifiers (that is the

values reachable by means of the object identifiers). Notice that the labels are all different one from each other.

Complex objects are labelled heterogeneous cartesian products. They can be used to aggregate multiple media and to associate them with a number of attributes. An example of a complex object could be an object composed of a video, an audio associated to the video and some text giving a textual description of the video. Attributes can represent technical information concerning the multimedia object, so a movie shot may contain information about the lens type, opening and closing shot information, such as fade and dissolve.

A disjoint union object is necessary for representing a single data item that can belong to different media. For instance, if we want to describe a person we can decide to associate her with a picture or a movie containing her (it depends on our disposability). The data item containing this information can be defined as disjoint union of images and movies. The disjoint unions are complementary to complex objects: a disjoint union is a labelled heterogeneous sum so a value of this type may contain a value of just one of the types specified in the sum. The disjoint union is different from the mathematical union of sets in that each argument of the union retains an inspectable label, indicating the original set the value is coming from. Thus testing the label of a disjoint union is like testing its type out of a finite set of possibilities.

An important component of a data model is bulk types (sets, sequences, bags, directed acyclic graphs, etc.) [Atkinson 89]. By means of bulk objects it is possible to represent collections of complex objects that are perceived to have some regular structure. They allow: to abstract over the size of collections they represent; the description of important regularities in the data; and to define efficient operations over these collections via exploitation of their regularity (e.g., optimised queries). Furthermore, they are useful for representing one–to–many relationships among complex objects. It is not clear yet how many and what bulk object constructors are particularly well–suited for MMDS, neither it is clear if efficiency makes it possible to use the *map* single bulk constructor as a type–quark that allow all the other bulk types to be build on the top of it as proposed in [Atkinson 91]: we just know that an investigation of bulk objects in MMDS has to be performed.

4. THE MULTIMEDIA PRESENTATION MODEL

The aim of the multimedia presentation model is to define a structure to organise objects of the description model stating how multimedia objects should be presented. In this section we address issues concerning: (1) Synchronisation Constraints and (2) Spatial Constraints on multimedia objects. In a presentation each media stream has to be synchronised with other streams and has to be presented in a portion of the screen with a certain layout.

The model should allow to specify how the user can interact with the system during a presentation and should allow the reuse of media objects. The presentation model should also provide support for the multimedia authoring.

4.1 Expressiveness for authoring

Typical approaches used in the definition of multimedia presentations, such as for instance *time line*, associate absolute timing information with the streams to be played [Semich 92]. Each stream is placed on an axis representing the time flow; the synchronisation with the other streams is given implicitly. Unfortunately this simple approach lacks expressiveness and it does not provide powerful constructs to edit multimedia objects. In particular it makes hard the task of modifying an existing

presentation since it requires the redefinition of many space and temporal constraints already specified. Let us suppose, for example, that an author wants to substitute a video sequence in a presentation. If the new video sequence has a different duration then all the remaining multimedia objects have to be moved from their position on the time axis.

Another disadvantage of the use of absolute timing is that precise timing information could not be available during the authoring process. Sometimes the delays due to overload of some device, or delay due to the interaction with the user can change synchronisation previously defined.

When an author builds a presentation he should be relieved of the burden of defining what happens in each instant of the presentation. An author should think in high level terms instead of defining what happens in the presentation at each instant and in each position of the screen. He should be interested in the overall information to use and in the overall layout of the presentation and not in the low level information describing details about synchronisation and timing.

The authoring process requires to structure information represented by multimedia objects [Hamakawa 93]: if these structures are not explicitly stored but just the final stream is maintained, it could not be possible to perform further editing or to reuse them in new applications. For example the creation of TV news may require the composition of different video objects, each one with its own audio comment; if the final result were an unstructured linear stream then it would be impossible to edit efficiently the stream and to reuse part of it in another presentation.

It could also be useful to provide a programming language for defining presentations. Some presentations may need repetitions of portions of streams or the presentation flow may behave differently depending on the interaction with end users. The use of a programming language that manipulates multimedia objects allows to automate repetitive tasks that have to be performed explicitly by the authors elsewhere. For instance it is possible to specify the following statements: "repeat this sequence twice" or "go to the next sequence" or "move this window on the screen".

There are some structured multimedia models that use powerful sets of operators to manage different types of multimedia data but they do not give powerful constructs to deal with single media. For example the models HyTime [Newcomb 91], CMIF [Hardman 93] and MHEG [Meyer-Boudnik 95] allow a structured composition of multimedia presentation and the specification of synchronisation constraints between media items but video data are still viewed as unstructured linear streams of data [Amato 95].

4.2 Structured Organisation

A presentation is made of a set of composite objects that are scheduled to be played during the time of the presentation. The component objects of a composite object have spatial and temporal constraints relative to the composite object. Updating the temporal and spatial constraints of the root object of a composition hierarchy leads to the recomputation of the absolute constraints of the component objects. The notion of composite objects is useful in order to group objects that should behave in the same way regarding the computation of temporal and spatial constraints.

A specific presentation can either use complete objects of the description model or parts of them: the author may decide to display only a temporal portion of the object; he can decide to use only a spatial portion of the object; the object can be displayed at a slow, normal, or fast speed. Each media can be synchronised with the execution of different media; the synchronisation can either be static or dynamic because of user interaction or when delays, due to characteristics of hardware devices, occur. In the proposed model, multimedia objects can stretch and shrink in the temporal dimension as well as in the

spatial one. The temporal synchronisation is obtained by managing events and actions using an event driven paradigm.

The first step consists in associating basic objects of the description model with space and temporal constraints, a presentation layout, and a list of triggers. Triggers specify actions to be executed when certain events occurr while the object is presented. Since constraints defined on these objects are relative to the constraint associated with the composite object containing them, we decided to name this category of objects *Virtual objects*.

- Virtual objects (*Vobject*) are built starting from the basic objects of the description model. They are associated with a layout, some spatial constraints, some temporal constraints, and a set of triggers:

  Vobject ::= (BOID, Layout, SPosition, Tposition, Triggerlist)

- Composite virtual objects (*CVobject*) are obtained by grouping existing complex objects of the description model:

  CVobject ::= (COID, Sposition, Tposition, Triggerlist)

  - *Layout* is media dependent and is not considered in this article. For instance, it can define font type and font size for a text, frame rate for a video, resolution and colour palette for an image.

  - *SPosition* is the relative spatial position of the object. It contains co-ordinates of the portion of the screen where specific object will be displayed and a spatial offset that specifies which portion of the object has to be displayed:

    SPosition ::= (Window, Soffset)

    where *Window* is represented by specifying the co-ordinates of the upper left corner and the lower right corner relative to the window of the object containing it or to the screen if it is a root object. The *spatial* offset (*Soffset*) specifies which portion of the object has to be displayed. It contains the distance of the upper left corner and of the lower right corner of the area to be displayed from the original object.

  - *Tposition* is the temporal position and defines the duration and the temporal period of the object to be displayed:

    Tposition ::= (Duration,Toffset)

Objects have associated a list of *triggers*, specifying which action has to be executed when a specific *event.* occurs.

Triggerlist ::= Trigger | Trigger, Triggerlist

Trigger ::= start(Time,Actionlist)| {start event}
end(Time,Actionlist)| {end event}
click(Time,AsyncActionlist) {user interaction}

Start and *End* events are synchronous while *Click* ones are asynchronous. When an asynchronous event occurs, just a subset of the specified actions (*AsyncActionlist*) can be issued.

Actionlist ::= Action | Action, Actionlist

AsyncActionlist ::= AsyncAction | AsyncAction, AsyncActionlist

Example of synchronous actions are *"start an object after some event"* or *"stop an object after some event"*. An example of a generic action is *"start an object in a way such that it ends before or after some event"*.

AsyncAction ::= StartAfter(time,TVobject)| StopAfter(time,TVobject)

Action ::= EndBefore(time,TVobject) |
EndAfter(time,TVobject)|
StartBefore(time,TVobject)|
StopBefore(time,TVobject)|
SyncAction

5. MULTIMEDIA INTERPRETATION MODEL

Multimedia data are representations of reality which is rich in information. This information cannot be easily managed by a computer since it is generally represented as a meaningless linear stream of bytes. For instance, a person can recognise in a movie two persons that are walking hand by hand in a park and he can also understand what they are saying. On the contrary, a computer views the movies as sequences of bytes and it is able to perform on it only simple operations such as display the video on the screen or play the sound track on the sound board. The Multimedia Interpretation Model allows to represent the features of multimedia objects and to combine these features to derive the concepts and the relationships among them. The model allows also to represent the knowledge of *spatial* and *temporal* relationships among the objects and it makes use of background knowledge in order to perform the interpretation of the multimedia objects. The Multimedia Interpretation Model will be the main support for the execution of *similarity based retrieval.*

However, our aim is not the definition of a new model for the interpretation of multimedia objects, while it is to enable the use of different existing models. This approach allows to "adapt" the model to the specific application needs and it enables to incorporate new models that could be defined in the future. Actually we are investigating the possibility of integrating the model defined in the ESPRIT BRA Project FERMI [Meghini 95].

In the following the main characteristics of the Multimedia Interpretation Model are described.

5.1 Similarity based retrieval

There are basically two modes for searching a multimedia database:

browsing, where users have foggy ides of what they are looking for and are interested in sample objects which might be used for retrieval;

content-based retrieval, where a requirement is specified and (a relevance-based) retrieval of objects satisfying the query is expected.

An important group of queries in multimedia environments useful to support browsing and content-based retrieval forms *similarity queries* [Wu 95], [Petrakis 93], [Petrakis 96], [Cardenas 93], [Day 95], [Dimitrova 94]. Generally, a similarity-based retrieval is needed when:

- exact comparison is not possible, it is too restrictive or it may even lead to empty results—the data is vague and/or the user is not able to express queries in a precise way;
- ranking of retrieved objects is needed so that the set of retrieved objects can be restricted and/or qualifying objects shown to the user in decreasing order of relevance.

The types of similarity queries fall into three categories:

- *similarity selection* retrieves either x most similar objects (i.e. x-nearest neighbours) or all objects with a threshold level t of similarity. In fact, the queries are constrained by two parameters at the same time. The first parameter, x, specifies the maximum number of objects returned by a query, and the second parameter, t, specifies the minimum threshold similarity in order for an object to be returned by the query. If the threshold is set to its smallest possible level of similarity, then the query becomes a standard x most similar query. If x is set to its maximum value, then the query returns all objects with the given threshold level of similarity;
- *similarity join* is a generalisation of the traditional join operation. In general, it finds all pairs of objects that are similar. The similarity constrains are specified in exactly the same way as described for the similarity selection above;
- *similarity sampling* finds a representative object sample that is similar either to a reference object or relative to the complete contents of the database. A general query must specify the reference object, some measure of the spread of the sampling (usually based on the measure of similarity) and the number of samples desired.

5.2 Features and concepts

The data interpretation model is based on features (observations, measures, abstractions, etc.) of primary multimedia data interpreted by specific concepts according to the needs of users. It can be seen as a two-level hierarchy. The first level, the *feature-level*, works on simple (or complex) single (or multi) media data by recognising, extracting, and materialising useful features, which are needed for application dependent interpretations on the second level of the interpretation model, called the *concept-level.*

Both of the mappings—from the multimedia data to the feature-level and from the feature-level to the concept-level—are of the *many-to-many* type. For example, let us consider a picture of a group of people in front of a church. One can be interested in recognising different people while some other users might be interested in specific

features of the church architecture. At the same time, the same features can be observed in different pictures.

Similarly, multispectral images obtained from earth satellite observations can be interpreted according to different concepts. They can be used for weather-forecast analysis, but geologists can use them for exploring the earth while searching for mineral resources.

The particular difficulties of extracting features in multimedia data environments can mainly be attributed to the following properties:

- features are subjective, thus multiple interpretations are needed—even a specific application may require to deal with data with different perceptions;
- features are difficult to describe and their identifications often require advanced and time-consuming extraction techniques;
- features typically represent only abstractions (or approximations) of real entities;
- similarity queries are typical due to the imprecision in features and uncertainty in query specification.

Interpretation of features on the concept-level is usually even more difficult because the concept interpretations are:

- application and domain dependent;
- specific interpretation models, often based on sophisticated AI techniques, are applied;
- strict performance constrains are typically required and that asks for efficient implementation techniques to apply.

5.3 Temporal and spatial knowledge

Multimedia objects are characterised by having many properties not related to the entities contained in them, instead they depend on the relationship existing among the entities and on the actions that the entities perform. For instance the same person can *be walking* in a sequence of a video and can *be sitting* in an other one; a person can *appear on the left of* the desk; a person can exit from the room *before* the bomb explodes.

It could be useful to interrogate the multimedia database using these facts so the interpretation model has to provide a representation for spatial relationship, temporal relationship and events. To obtain a complete interpretation of a multimedia document the feature layer has to provide a measure also for these aspects and the concept level has to provide techniques to perform spatial and temporal reasoning.

5.4 Background Knowledge

The information contained in a multimedia document represents only a partial information of the real world. For instance, in a picture in which two children appear, every person can recognise that they are a boy and a girl, inferring this from their appearance. Nobody, unless he knows them, can infer that they are brothers and that their names are Anne and Mark. This kind of knowledge cannot be generally inferred using only the information contained in the multimedia document but it should be explicitly inserted. We call this kind of knowledge background knowledge.

Background knowledge can contain also temporal and spatial knowledge. For instance a person could search for the picture of Mark taken after that his sister was born, even thought that there are no multimedia data documenting that Anne was born.

6. CONCLUSIONS

This paper reports the work-in-progress on the definition of an object-oriented multimedia representation model performed within the BRA ESPRIT Project HERMES. The proposed multimedia model tries to addresses all aspects related to data presentation, manipulation, and content-based retrieval of future MMDBs. It is composed of three mutually related models, Multimedia Description Model, Multimedia Presentation Model, Multimedia Interpretation Model. The main objective of the presented work is to create a framework which would allow integration of different implementation approaches and partial modelling efforts.

We consider our future work to evolve along the following two main directions:

- A formal definition of the three proposed models and in the integration of specific models for semantic object representation. In particular, as already mentioned, the FERMI model will be considered and the connections and relationships with the Description model and the Presentation model studied.
- The implications of these models with the storage and access of multimedia objects will be studied. Indeed, real application environments will require the storage of many trillions of bytes of data, requiring the use of a storage hierarchy consisting of different layers. This implies that data placement is crucial for effective manipulation of the data and for an efficient retrieval.

7. REFERENCES

[Amato 95]
Amato G. and C. Meghini, "Considerations on a model for video data", *IEI Tech. Rep. B4–42*, Dec. 1995.

[Atkinson 89]
Atkinson M. P. and O. P. Buneman, "Types and Persistence in Database Programming Languages", *ACM Computer Surveys*, Vol 19, N. 2, 1989, pp. 105–190.

[Atkinson 91]
Atkinson M. P., C. Lecluse, P. Philbrow and P. Richard, "Design issues in a map language", in *Bulk Types and Persistent Data*, P. Kanellakis and J. W. Schmidt (eds.), Morgan Kaufmann, San Francisco, CA, pp. 20–32.

[Berra 93]
Berra P. B., F. Golshani, R. Mehrotra and O.R. Liu-Sheng "Guest editors' introduction: Multimedia information systems", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 5, No. 4, Aug. 1993, pp. 545-550.

[Cardenas 93]
Cardenas A. F., I. T. Ieong, R. K. Taira, R. Barker and C. M. Breant, "The Knowledge-Based Object-Oriented PICQUERY$^+$ Language", *IEEE Transactions on Knowledge and Data Engineering*, 5(4), Aug. 1993, pp. 644-657

[Day 95]
Day Y. F., S. Dagtas, M. Iino, A. Khokhar and A. Ghafoor, "Object-Oriented Conceptual Modeling of Video Data", *Proc. of the 11th Int. Conf. on Data Engineering*, Taiwan, 1995, pp. 401-408.

[Dimitrova 94]
Dimitrova N. and F. Golshani, "RX for Semantics video Database Retrieval", Proc. of *ACM Multimedia '94*, 1994

[Ghafoor 95]
Ghafoor A., "Multimedia Database Management: Perspectives and Challenges", Proc. *Advances in Databases, 13th British National Conf. on Databases*, C. A. Goble and J. A. Keane (Eds.), UK, July 12-14, 1995. LNCS(940), pp. 12–23.

[Gibbs 94]
Gibbs S., B. Christian, D. Tsichritzis, "Data Modeling of Time-Based Media", Proc. of *ACM SIGMOD Conference on Management of Data 1994*, Minneapolis, Minnesota USA, 1994, pp.92-102

[Hamakawa 93]
Hamakawa R. and J. Rekimoto, "ObjectComposition and Playback Models for Handling Multimedia Data", Proc. of *ACM Multimedia 1993.*

[Hardman 93]
Hardman L., G. van Rossum and D. C. A. Bulterman, "Structured Multimedia Authoring", *Proc of ACM Multimedia 1993.*

[IEEE-ToSE 88]
Special Issue on Image Databases, *IEEE Trans. on Soft. Eng.*, May 1988.

[Kunii 89]
Kunii T. K. (ed.), *Visual Database Systems,* North-Holland, 1989.

[Meghini 95]
Meghini C., "A logic for information retrieval", FERMI, *Deliverable D2 of the ESPRIT Basic Research Action FERMI* (Project n. 8134), 1995.

[Meyer-Boudnik 95]
Meyer-Boudnik T. and W. Effelsberg, "MHEG Explained", *IEEE Multimedia*, Spring 1995.

[Newcomb 91]
Newcomb S., N. Kipp and V. Newcomb, "The "HyTime" - Hypermedia/Time-based document structuring language", *CACM* Vol 34, N. 11, 1991, pp. 67 - 83.

[Petrakis 93]
Petrakis E.G. and S. C. Orphanoudakis, "Methodology for the Representation, Indexing and Retrieval of Images by Content", *Image and Vision Computing*, 11(8), Oct. 1993, pp. 504-521.

[Petrakis 96]
Petrakis E.G. and C. Faloutsos, "Similarity Searching in Large Image Databases" *IEEE Transactions on Knowledge and Data Engineering*, 1996, to be published.

[Semich 92]
Semich J.W., "Multimedia Tools For Development Pros",*Datamation*, Aug. 1992.

[Wu 95]
Wu J. K., A. D. Narasimhalu, B. M. Mehtre, C. P. Lam and Y. J. Gao, "CORE: A Content-based Retrieval Engine for Multimedia Information Systems", *Multimedia Systems*, Springer Verlag, 3(1), Feb. 1995, pp. 25-41.