Object-oriented modeling of multi media databases in UML

Amani Anwar Saad
Computer Science and Automatic control Dept., Faculty of Eng., Alexandria University
Alexandria 21544, Egypt

This paper presents an Object-Oriented data model for Multimedia Databases. The model supports building Hierarchical Video Models and Composite Multimedia Presentations. It attacks the problems of Time-based Media that are the most complex in modeling Multimedia data. The model is presented in the Unified Modeling Language UML that is the current standard for modeling Object-Oriented Systems. The abstract concepts of the model are represented by the Class Diagram and the System Functionality is illustrated by the Use Case Diagram. Also a Scenario for implementing a Use Case is illustrated by the Sequence Diagram. Finally, an example that illustrates how a Composite Multimedia Presentation can be modeled in the system is illustrated.

يقترح هذا البحث نموذج لتمثيل قواعد البيانات المتعددة الوسائط قائم على النمذجة الشيئية ويستخدم لغة النمذجة الموحدة القياسية UML. ويعالج هذا النموذج مشاكل الوسائط التي تعتمد على الزمن وهى الأكثر تعقيداً في التعامل معها. كما يتيح هذا النموذج تمثيل الفيديو بطريقة تدريجية مثل الشجرة أي بمستويات مختلفة من الدقة. كما يتيح النموذج تمثيل عروض مركبة من حيث الزمن والمكان للوسائط المتعددة. ويعد هذا النموذج هو أول نموذج قياسي قائم على النمذجة الشيئية لقواعد البيانات المتعددة الوسائط.

Keywords: Object-oriented modeling, Unified modeling language, Multimedia databases,Time-based media, Content-based retrieval

1. Introduction

A key characteristic of Multimedia Database Management Systems (MMDBMS) is the manipulation of different types of media; audio, video, text, images and graphics simultaneously [1]. Multimedia data should be retrievable by content and should also be integral components of the query process [2]. Moreover, a key characteristic of video data, which is the most complex media to deal with, is the associated spatial and temporal semantics. Thus, a data model for multimedia data should identify the media objects and their supporting temporal as well as spatial relationship [3]. Thus, in order to provide efficient management, a MMDBMS should integrate both content attributes of video/audio streams, as well as, their semantic structure. It should also describe the physical media objects (persons, cars, trees...etc) present in a video/audio clip and the associated verbal communication for each segment according to a hierarchical model for video/audio [3].

It should be clear that there is a qualitative difference between Time-based media and the forms of data traditionally stored in database systems. Time-based media, including audio, video, music and animation, involve notions of data flow, timing, temporal composition and synchronization. These notions are foreign to conventional data models and as a result, conventional data models are not well suited to Multimedia database systems in general [4].

A related problem is the support of Multimedia Presentations [5] that requires the design of presentation models for spatial composition and temporal composition [6].

In [7], a model for semantic content-based retrieval of MMDBMS was proposed. A conceptual level data model was designed and mapped into a relational design. This model tried to

overcome the problems of previous models by treating different types of media uniformly with respect to their contents, and by following a standard design methodology. However, the temporal characteristics of Time-based media were not taken into consideration. Also, Hierarchical Video Models and Composite Multimedia Presentations were not supported.

It is well established now [1] that the Object Oriented (OO) models is the best for representing such complex data as that present in a MMDBMS. It allows a natural way of thinking of these data as objects, and it facilitates modeling composition relationships through the concept of ***composite objects*** present in OODBMS. Also, the encapsulation concept is well suited to package each class operations (such as Media Type and a set of derivations on it) with its structure thus facilitating the use of pattern *recognition* and image processing operations to extract the contents of media components and build an integrated and extensible Multimedia system. Moreover, the ***inheritance*** concept is used to inherit the structure and operations of a super-type by a sub-type.

In this paper, an OO data model for multimedia databases that covers all the concepts in [7] and deals with the problems related with Time-based media is proposed. These problems may be summarized as; modeling temporal, and spatial relations among media components, as well as, the physical media objects they contain, modeling spatial and temporal composition relationships among media components supporting presentation models, and supporting Hierarchical Video Models.

The model is presented using the Unified Modeling Language UML [8] which is now the standard modeling language for OO systems proposed by the Object Management Group OMG [9]. Thus, the work in this paper is a promising step towards a standard Object Oriented model for the content-based retrieval from MMDBMS.

Previous OO Models for video data such as [10, 11], did not cover all the concepts covered in this model, did not follow a standard methodology or use a standard modeling language such as (UML). Moreover, the OVID model [10] was not purely Object-Oriented, as it was based on a schema-less database and its concept of *interval inclusion inheritance* was a special definition for inheritance and is not the same as the inheritance between objects in different classes in OO systems.

The rest of this paper is organized as follows; Section 2 presents some concepts related to Multimedia and specially Time-based Media. Section 3 presents the Unified Modeling Language UML that is used to present the proposed model. Section 4 presents the structural aspects of the proposed model that supports the concepts discussed in this paper. Section 5 continues the model presentation by illustrating the functionality of the model also by UML diagrams. Section 6 presents an example that shows how to model a Multimedia Presentation, and finally, section 7 presents a summary of the paper and its future work.

2. Multimedia and Time-based media concepts

In this section, some related definitions and multimedia concepts are presented. Most of these concepts are introduced for modeling Time-based media except for the Binary Large OBject (BLOB) which was introduced for the storage of very large objects as images and video or audio clips.

Definition 1: Binary Large OBject (BLOB) [4]. A BLOB is an attribute value that appears to applications as a sequence of bytes. The DBMS provides an interface by which applications can read and append data to BLOBs. The BLOB can be very large in size (i.e., many Megabytes), it may be a part of contiguous storage or fragmented which is a performance issue and not relevant to data modeling. However, while the storage of very large values is necessary for a MMDBMS, it is

not sufficient. The MMDBMS should also have some understanding about the internal structure of a BLOB, i.e., it must be able to interpret the data.

Definition 2: Interpretation [4]. An Interpretation I of a BLOB B , is a mapping from B to a set of Media Instances (media elements). For each Media Instance, I specifies the Media Instance descriptor and its placement in B. From the above definitions we can say that a BLOB may represent several Media Instances which may belong to several media types. For example, it may represent an interleaved sequence of video and audio clips.

Definition 3: Derivation. A Derivation D of a Media Instance MI_1 for a set of Media Instances MI is a mapping of the form $D(MI,Pd) \rightarrow MI_1$, where Pd is the set of parameters specific to D.

- MI_1 is said to be a derived media instance.
- Pd is called the derivation descriptor.
- Derivations may be classified into derivations changing the content, changing the timing, or changing the Media Type, where each media type has its possible set of derivations.

Definition 4: Composition [6]. Composition is the specification of temporal and/or spatial relationships between a group of Media Instances. In our model terminology, we shall call the result of a composition, a composite media instance. This composite Media Instance can model a part of a Multimedia presentation.

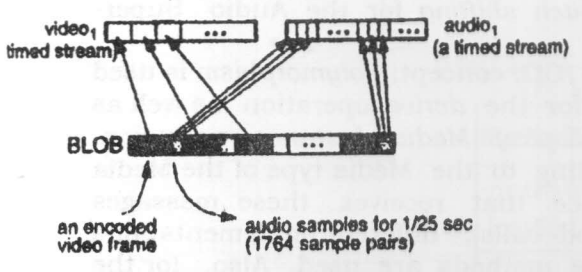

Fig. 1. A BLOB and its interpretation.

3. The Unified Modeling Language (UML)

The Unified Modeling Language (UML) [8] presented by the Object Management Group (OMG) [9] has become the standard language for modeling and documenting OO systems. It is a very expressive language, addressing all the views needed to develop and then deploy a wide range of software systems [8].

The UML distinguishes between the notions of diagram and model. A model contains all of the underlying elements of information about a system under consideration and does so independent of how these elements are usually presented. A diagram is a particular visualization of certain kinds of elements from the model and generally exposes only a subset of these elements detailed information. A given model element might exist on multiple diagrams, but there is but one definition of that element in the underlying model [8].

In this work, only a subset of the diagrams supported in UML is used. The *Class Diagram* is used to model the static aspects of the system. These are the important abstractions of the system (Classes) and how they relate to each other (Relationships).

The Use Case Diagram is used to model the functionality of the system. It describes what the system can do at a very high level. A Use Case is a general pattern or strategy of using the system that may represent many related yet distinctly different scenarios. Thus, each scenario represents a specific thread through a use case.

The Sequence diagram is an interaction diagram that is used to represent a scenario that is a dynamic aspect of the system.

In the rest of this paper, the proposed model is presented. First, the different classes of the system and their relationships are represented by the Class Diagram. Then the system functionality is illustrated by giving a list of the Use cases that are supported by this proposed

model. Finally, an example scenario is illustrated by a Sequence Diagram.

4. The proposed model

The following subsections present an Object-Oriented model for Multimedia databases that supports the abstract concepts that are mentioned in this paper. These concepts are divided into groups of classes and the classes of each group or two groups are finally presented by a Class diagram. A specific class may be present in more than one class diagram (group) but each Class Diagram illustrates different relationships between this specific class and others.

4.1. Modeling media instances, media types and super-types

4.1.1. The super-type class

The Super-type class is used to represent the basic multimedia data types that are; text, audio, video, image, and graphics [1]. Each basic media type is an instance of the Super -Type Class. The *Characteristic* attribute in this class may describe the common characteristics of all media types which are subclasses of a specific Super-type, e.g., the set of valid derivations on this Super-Type may be referenced here.

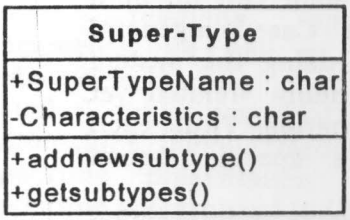

Fig. 2. Super-type class.

4.1.2. The media type class

A Media Type is used to represent multimedia data types which have a specific format and are subclasses of a certain basic type , e.g., MPEG-I as a sub-class of the Video Super-Type or .TIFF as a sub-class of the Image Super-Type. Every Media Type has a Media Descriptor and requires a special Display Routine.

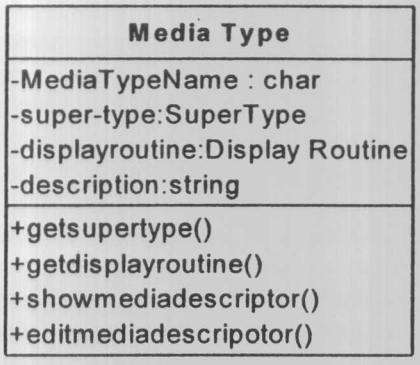

Fig. 3. Media type class.

4.1.3. The media instance class:

A Media Instance is the minimum unit of multimedia data of a certain Media Type that can be stored or retrieved [7]. A certain Media Instance is an instance of a certain Media Type.

Each Media Instance points to a set of *Features* which characterize it and a set of *Media Objects* that it contains. It has an optional attribute that is the BLOB, in case this Media Instance is stored as a part of a BLOB. The *Location* attribute points to an instance of Class Location that facilitates using different physical database designs for storing Media Instances or BLOBs.

The *derive* method is used to apply different derivations on Media Instances. Each Media Type has a set of derivations that can be applied on it. For example, *color separation* for the Image Super-type, *video transition* for the Video Super-type, and *pitch shifting* for the Audio Super-Type .

The OO concept; *Polymorphism* is used here, for the *derive* operation as well as the *display Media Instance* operation. According to the Media type of the Media Instance that receives these messages (method calls), different implementations for the methods are used. Also, for the same Media Type, different derivations are applied according to the operation code of the derivation.

4.1.4. The media descriptor class

The Media Descriptor Class is a Super-Type for different basic Media Type Descriptor classes; Video Descriptor, Audio Descriptor, Image Descriptor, etc...Each specific Media Type (e.g., TIFF for images), has its Media Descriptor represented as an instance of the Image Descriptor Class. Thus, all image formats registered in the system have the same structure (set of attributes that describe them) but each has different values for these attributes.

4.1.5. The display routine class

The Display Routine Class gathers the objects of type Display Routine. Each instance of this class gives the display routine name and its location for a certain Media Type.

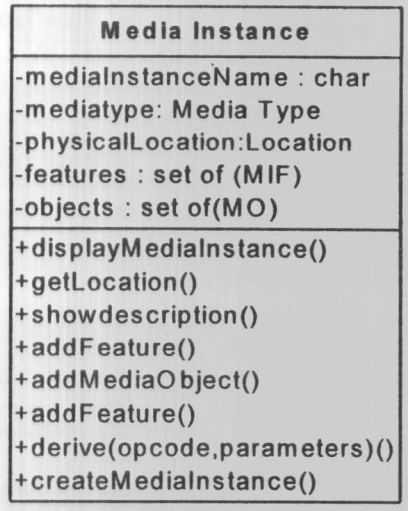

Fig. 4. Media instance class.

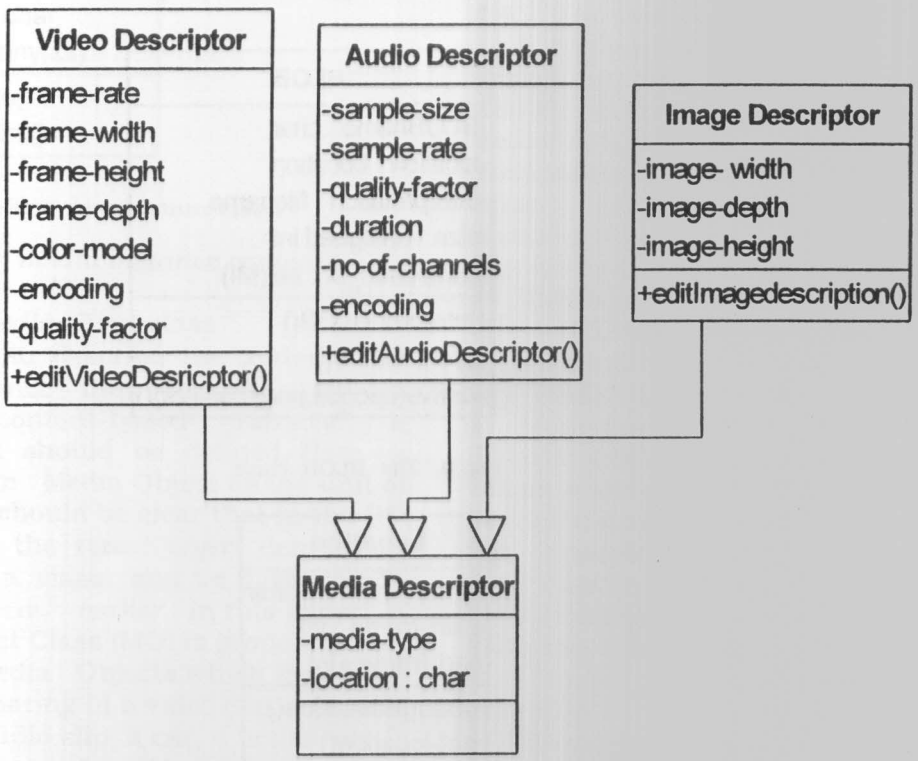

Fig. 5. The media descriptor class and its subclasses.

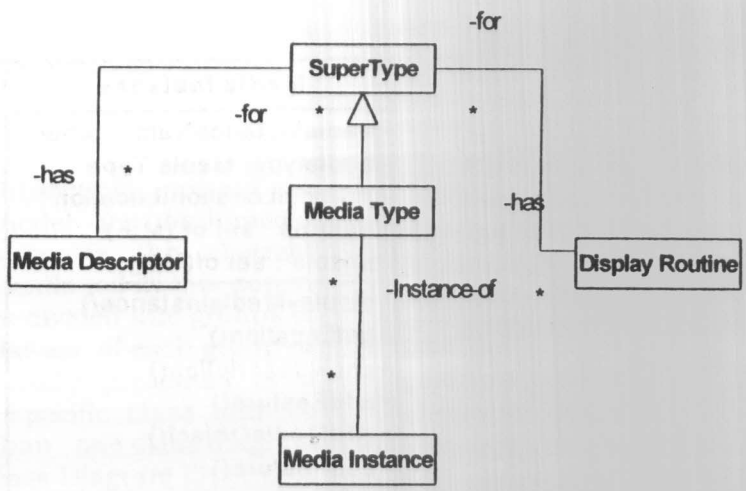

Fig. 6. Class diagram for classes in group 1.

Display Routine
-display-routine-name : char -display-routine-location: Location -media-type : MediaType
+display-media-instance() +edit-location()

Fig. 7. The display routine class.

4.2. *Modeling BLOBs and composite media instances*

4.2.1. *The BLOB class*

According to the definition of a BLOB presented early in this paper, a BLOB consists of a composition of Media Instances that are not necessarily of the same type. Each BLOB holds information about (points to) its interpretation which maps it into several Media Instances. It has also a reverse reference as a set-valued attribute that holds the object identifiers of all its Media Instances. This reverse reference may be used to facilitate some queries processing.

4.2.2. *The location class*

The Location Class is used to facilitate different physical database designs, i.e., by changing its attributes by the DBA, he can describe the location of a Media Instance or a BLOB in different ways. In this model, we use three attributes to identify a location, which are the *Stream Name* that is the same as OS file name, *Start frame* and *End Frame*. This format is the one suggested in [12].

BLOB
-BLOBname : char -location : Location -interpretation : filename -size : unsigned int -composed_of : set(MI)
+interpret-BLOB() +createBLOB() +derive(opcode,paramaeters)()

Fig. 8. The BLOB class.

Location
-stream-name : char -start-frame : int -end-frame : int
+change-location() +get-location()

Fig. 9 The location class.

4.3. Modeling media instance features

Media Instance Features are meta data about Media Instances which identify them externally and are relevant to user queries, such as ; date, author, etc.. They have nothing to do with the Media Instance content.

4.3.1. The media instance feature class

The Media Instance features are modeled through the Media Instance Feature (MIF) Class that has an aggregation relationship with the Media Instance Class. That is each instance of the class Media Instance, points to a set of instances (objects in OO terminology) in the MIF class using its composite set-valued attribute *Features*. The attributes of an object in Class MIF are the *feature name* and *Feature value*.

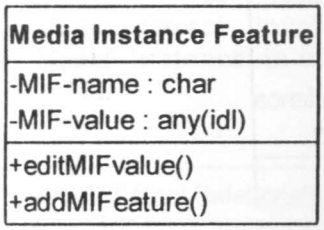

Fig. 10. The media instance feature class.

4.4. Modeling media instance contents

4.4.1. The media object class

In order to describe the contents of a Media Instance that can be used for semantic content-based retrieval, a content unit should be defined. Here we use the term Media Object as the unit of content. It should be clear that in the OO terminology, the term *object* denotes an instance of a class and we have already used this term earlier in this paper. A Media Object Class (MO) is proposed here to model Media Objects which may be; a person appearing in a video clip, a person singing an audio clip, a car, a house, etc.

An instance of the Media Object Class has simple attributes denoting its *name* and *description*, and a composite attribute *Features* which is set-valued and points to a set of Instance Independent Features of the Media Object. An optional composite attribute *appears-in* may be added which is a reverse reference to the Media Instances that this Media object appears in order to facilitate some queries processing.

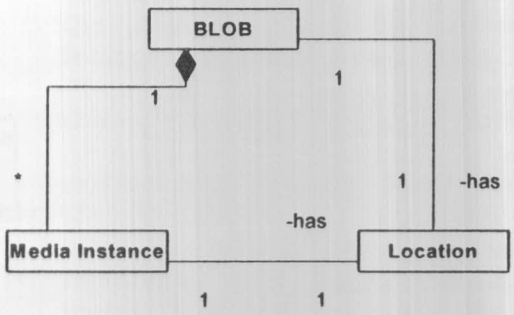

Fig. 11. Class diagram for classes in group 2.

4.4.2. The media object feature classes

A Media Object Feature may be any related data that may be used to describe , identify and retrieve the Media Object. A Media Object Feature (MOF) may be Instance Independent (IIMOF) , i.e. , its value is independent of the Media Instance that the Media Object appears in. For example, the Media object "Mr. X" has its IIMOF name ="name" with its IIMOF value = "Mr.X" independent of the appearance of the Media Object Mr.X in any Media Instance. On the other hand, a MOF may be instance dependent (IDMOF), i.e., these MOF take different values depending on the Media Instance in which the Media Object appears in.
For example, the media object "Mr.X" may have an IDMOF whose name is "Jacket color" with a value "red' in Media Instance image I and with another value say "blue" in another Media Instance say video clip J.

These two types of MOF are modeled by the two classes IIMOF and IDMOF.

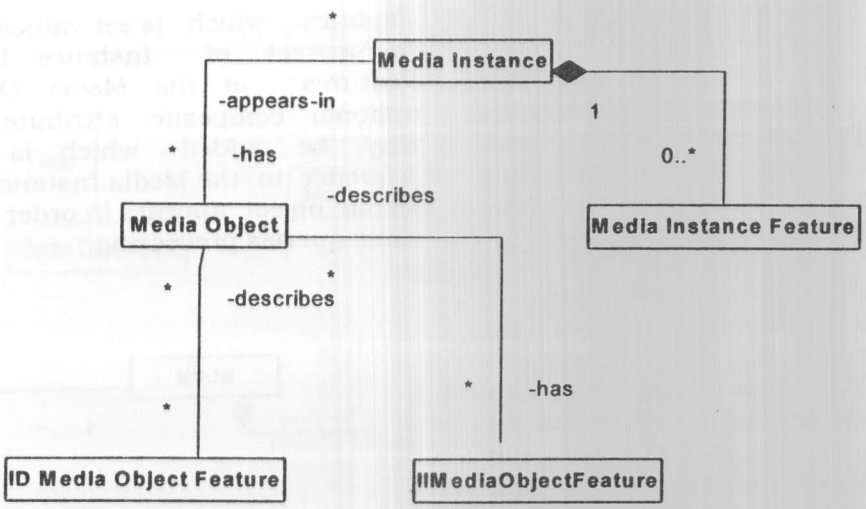

Fig. 12. Class diagram for classes in groups 3 and 4.

Media Object
-MO-name : char -description : String -features: setof(IIMOF) -appears-in: setof(MI)
+edit-description() +displayIIfeatures() +getcorrespondingMIs() +does-object-appear-in()

Fig. 13. The media object class.

Instance Independent MOF
-MOFname : String -MOF-value : any(idl) -media-object:MO
+is-MOF-in-MO() +get-MO-with- MOFvalue() +get-MO-with-MOF-name()

Fig. 14. The II media object feature class.

4.5. Modeling relationships among media objects

Different relationships among media objects contained in Media Instances may be used to retrieve specific Media Instances from multimedia databases.

Instance Dependent MOF
-MOF-name : String -MOF-value : any(idl) -media-instance: Media Instance -media-object: MediaObject
+get-MOF-value-in-MI()

Fig.15. The ID media object feature class.

These relationships may be classified into;

- *Logical*, such as; *married-to, father-of, etc.*
- *Spatial*, such as; *to-the-left-of, above, etc.*
- *Spatial Composition*, such as; *is-part-of, contained-in, etc...* These relationships can also be classified according to their dependency on the Media Instances in which they occur as follow:
- *Instance Independent Media Object Relationship (IIMOR)*, These are relationships among Media Objects existing in the MMDB and do not depend on the Media Instance in which these objects appear, i.e., the IIMORs preserve among these objects in whatever Media Instance. These relationships are modeled by the IIMOR

Class. They can be used to represent any Logical relationship.

- *Instance Dependent Media Object Relationship (IDMOR),* These are relationships among Media Objects that depend on the Media Instance in which the involved Media Objects appear. Spatial and Spatial Composition relationships are usually in this group. They are modeled by the IDMOR Class.

4.5.1. Modeling logical relationships between media objects

The IIMOR Class is used to model logical relationships. Each instance of this class refers to the involved Media Objects.

4.5.2. Modeling spatial and spatial composition relationships among media objects

The IDMOR class is used to model Spatial and Spatial Composition relationships. Each instance has attributes that refer to the relation name, Media Object involved, as well as, the Media Instance in which this relationship holds.

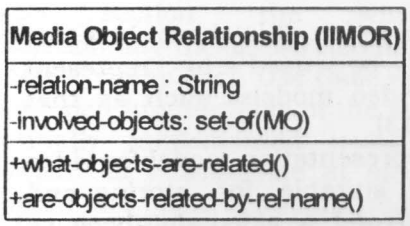

Fig. 16. The II media object relationship class.

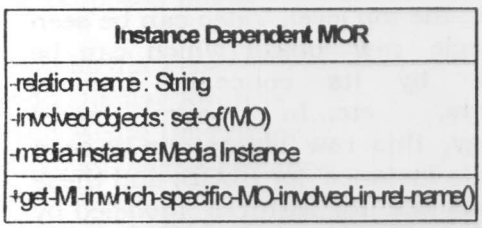

Fig. 17. The ID media object relationship class.

4.6. Modeling relationships among media instances

An essential feature of multimedia is the mixing of sound and imagery (for example, films). Looking at multimedia in general, one finds that complex multimedia structures are built up from simpler, perhaps "single media" components which we call here Media Instances.

These Media Instances are combined using various "Composition" mechanisms that establish relationships between them. These relationships can be classified into:

- "Spatial Relationships ", for example, the relative positioning of the Media Instances during a presentation.
- "Temporal Relationships", for example, the relative timing during a presentation. Several temporal relationships among Media Instances were defined in [6]. These relationships may be binary involving two Media Instances such as; A overlaps B, A meets B, A starts with B, etc...or may be n-array relationships involving n Media Instances.

4.6.1. The media instance relationship class

This class is used to model the Media Instance relationships in general. The main attributes are the relation name and the set of involved Media Instances.

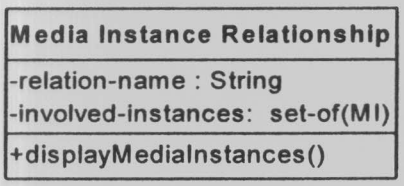

Fig. 18. The media instance relationship class.

4.7. Modeling activities, events and roles

An Activity type is an action that may occur in a Media Instance, for example, a wedding party in a video clip, a speech in an audio clip, etc...

An Event is a certain instance of an activity type, for example, the wedding party of Mr. A and Miss B specifically.

Each Activity type has a set of Roles associated with it, and hence for each event, a set of Media Objects takes these Roles. For example, the wedding activity type has the roles; Bride *and Groom.* At

the special event of the wedding of Mr.A and Miss B, Mr.A takes the role of the groom and Miss B takes the role of the Bride.

In the proposed OO model, the following four classes are used to model these concepts; The Activity Class, The Event Class, The Roles class, and The Role Object Class.

4.7.1. The activity class

Each activity instance of this class has the attributes; *activity name, description,* and *roles* which is a set-valued attribute that points to a set of instances (objects) in class Roles. A Role may exist in many activity types but we do not maintain the reverse reference here as we do not need it in any queries.

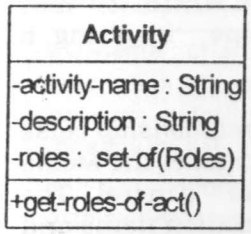

Fig. 19. The activity class.

4.7.2. The event class

Each instance (object) in the Event Class has the attributes; *media instance* denoting the Media Instance in which the event occurs, the *activity* which refers to the activity type of this event, and a set-valued attribute *heroes* which refers to a set of instances of Class Role-object that represent the Media Objects taking these Roles.

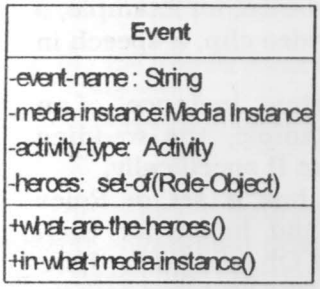

Fig. 20. The event class.

4.7.3. The role object class

Each instance of this class has two attributes, the *Role* and the *Media Object* taking this role.

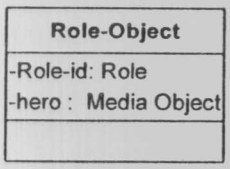

Fig. 21. The role-object class

4.7.4. The roles class

Each instance of this class has the two attributes *role name* and its *description.*

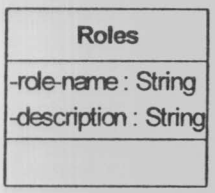

Fig. 22. The roles class.

4.8. Representing hierarchical video models

In this subsection, it is shown how this model can be used to represent hierarchical video models, such as that presented in [13].

In [13], a representation model for video data that is suitable for storing and retrieving (in/from) a video database or MMDBMS is presented. In this model, video can be represented in multiple levels. At the top level, Video can be seen as a single raw object which can be described by its concept, director, participants, etc...In our model terminology, this raw object can be seen as a Media Instance (or BLOB) and these descriptions are its features modeled by the MIF Class. However, this global description is not sufficient. Hence, in [13], three other levels of the hierarchy are presented and are illustrated by a Class diagram in fig 25. The Episode level video structuring provides the facility of accessing internal events. Furthermore, an episode is composed of several video

clips /shots. A video clip/shot refers to a collection of contiguous video frames depicting the same action in time and space.

In the proposed model, any level of this hierarchy can be represented as a BLOB that contains several Media Instances of lower levels in the hierarchy. A BLOB is actually representing a composite Media Instance that may contain several Media Instances of the same or different media types.

5. Modeling the MMDBS functionality by UML

The OO model for a MMDBMS proposed in this paper facilitates the integration of the operations (methods in OO terminology) performed on the instances of different classes (objects in OO terminology) with the objects themselves using the concept of Encapsulation.

Thus for each class of objects proposed in this model, I have already presented the set of essential methods that can be sent as messages to objects of this class. In this section, the whole system functionality is illustrated as a set of Use Cases where each Use Case represents an intended functionality of the system and is actually implemented using a Scenario which is a sequence of messages calls which are sent between objects that are possibly of different classes. The following list represents the supported use cases in the proposed model:

- Add Media Type
- Add Display Routine
- Add Media Descriptor
- Display Media Instance
- Add Media Instance Feature
- Extract / Add Media Objects in Media Instances
- Derive from Media Instance
- Store / Register Media Instance in the system
- Add Media Instance Relationship
- Add Sub-type for Super-type
- Store BLOB in the system
- Interpret BLOB (get corresponding Media Instances)
- Edit BLOB Location
- Edit Media Instance Location
- Display Media Descriptor
- Add Media Instance Feature
- Display Media Instance Features
- Retrieve Media Instance by Media Instance Feature value
- Get Media Instances where a Media Object appears
- Add/Edit Instance Independent Media Object Feature
- Does Media Object appear in a specific Media Instance?
- Is Media Object Feature name in a specific Media Instance?
- Get Media Instance with a specific IDMOF
- What is a specific IDMOF value in a specific Media Instance?
- Are the two objects MO1 and MO2 related by relation name?
- Get a Media Instance in which MO_1 and MO_2 are related by a certain Media Object Relation name
- Instances MI_1 and MI_2 related by a certain Media Instance Relation?
- What are the Are Media Roles of a specific activity type?
- What Media Objects took Roles in Event1?
- What events occurred in Media Instance1?

5.1. Use Case Diagram

A Use Case Diagram is a rectangular box in which each use case in the system or a subsystem is represented by an oval. A dotted line between two use cases implies that one is dependent on the other.

An *Actor* is an external entity that may initiate a use case (e.g., the DBA) or is the recipient of the system usage (e.g., the User, a display device, etc..).

The Use Case Diagram in fig 26 is given as an example but it represents only a small subset of the list of the supported

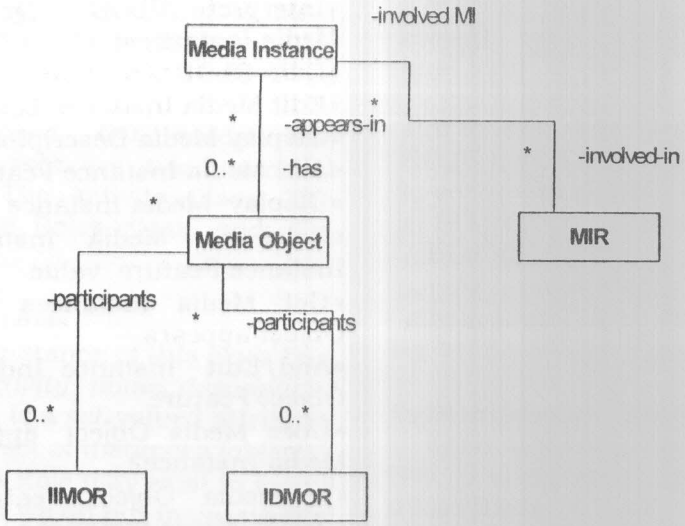

Fig. 23. Class diagram for classes in groups 5 and 6.

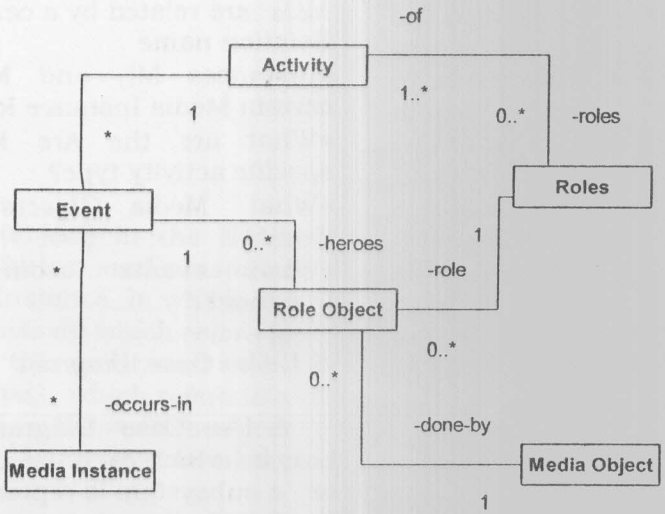

Fig. 24. Class diagram for classes in group 7.

Use cases in the model proposed in this paper.

5.2. Sequence diagram

A Scenario is a particular path through the system functionality. A Use Case may represent many related yet distinctly different scenarios. A Scenario may be represented in UML by one of the Interaction Diagrams supported in it, that is the Sequence Diagram.

In this subsection, an example scenario is illustrated by its sequence diagram in fig 27. This example is for the Display Media Instance use case. The

rectangles above represent classes in the system or actors that are external to the system. A horizontal arrow shows a message call from an object to another (in the classes represented in the corresponding rectangles). Time flows from the top of the diagram downwards.

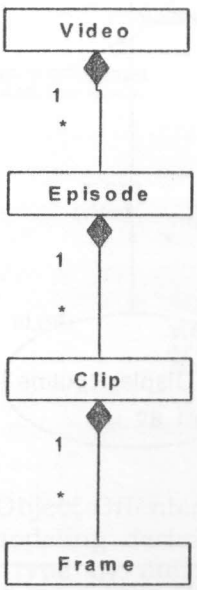

Fig. 25. Class diagram of video hierarchy.

6. Creating a composite multimedia presentation out of a BLOB

In this section, an example is given to show how a Multimedia Composite Instance is created from a BLOB and used to prepare a Multimedia Presentation. This problem could not be solved by our previous model [7] or previous models to it presented in [14].

Assume BLOB1 is the BLOB shown in Fig 28 that is composed of an interleaved sequence of video and audio clips; V1, A1, V2, A2, V3, A3, V4, and A4. This Figure is a modified version from the figure presented in [4] to illustrate the time based concepts which were introduced there.

The steps shown in fig 28 that are used in order to prepare a Composite Multimedia Presentation, can be very easily implemented in a system that uses the model proposed in this paper, using a set of *messages* between objects (instances of classes). This shows the benefits of the Object Oriented model that integrates the operations with the multimedia data itself in an OODBMS.

The following set of messages (method calls) implement the steps in fig 28.

- interpret-BLOB(BLOB1)-> BLOB2, BLOB3

that is used to interpret BLOB1 and produce a video stream BLOB2 and an audio stream BLOB3.

- derive(select-from, BLOB2, V2,V3,V4)-> BLOB4

produces from BLOB2 another video stream BLOB4 that contains V2, V3, and V4.

- derive(select-from, BLOB3, A2,A3,A4)-> BLOB5

produces from BLOB3 another audio stream BLOB5 that contains A2,A3, and A4.

- derive(video-transition, BLOB4, transition parameters) -> BLOB6

applies the *video transition* derivation on BLOB4 producing another video stream BLOB6.

Now in order to model a Multimedia Presentation that is a *temporal composition* between the audio stream BLOB5 and the video stream BLOB6, we may use an instance of the Media Instance Relationship Class which is the relation; *starts- with* that represents a binary temporal relation [6].

Starts-with (BLOB5,BLOB6)

7. Summary and future work

In this work, an Object-Oriented model for Multimedia Databases is proposed. The model supports building Hierarchical Video models, Composition models for Multimedia Presentations, and attacks the special problems of Time-based Media.

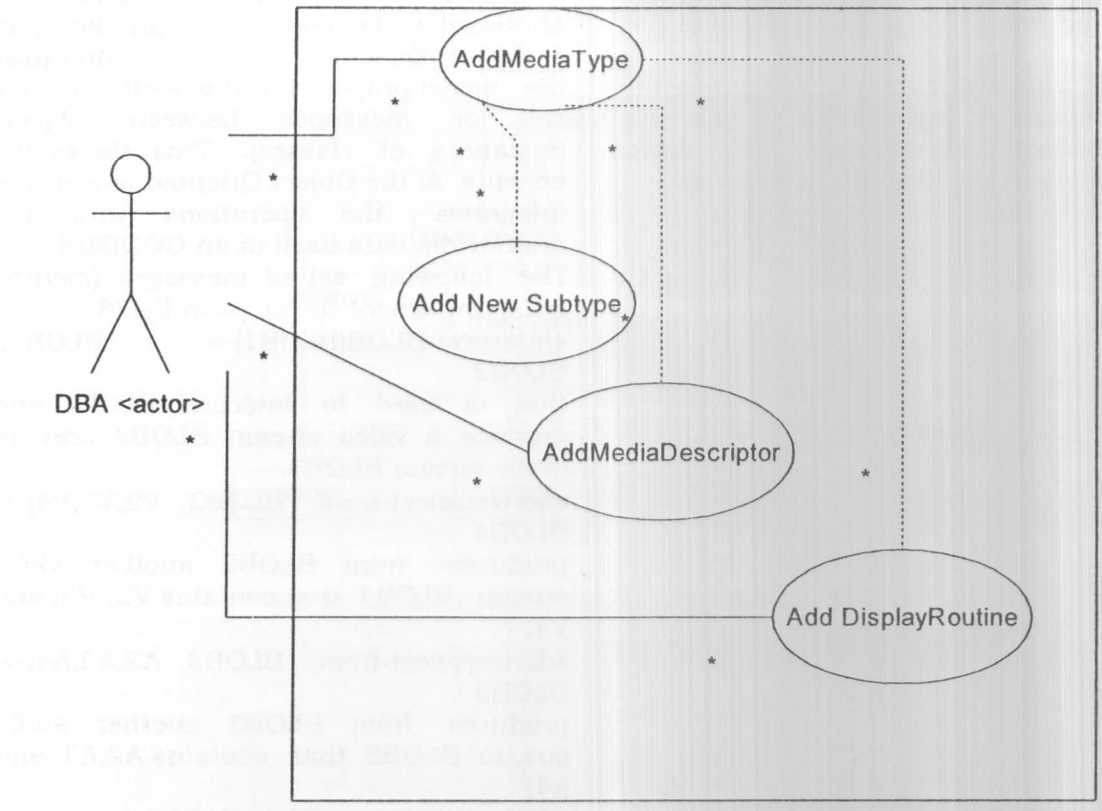

Fig. 26. A Use case diagram example.

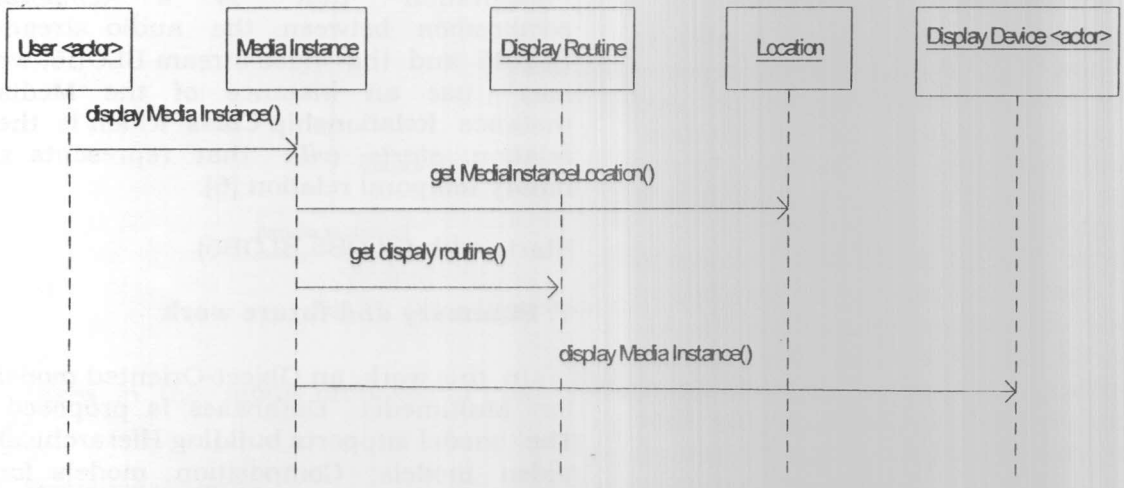

Fig. 27. The sequence diagram for display media instance use case.

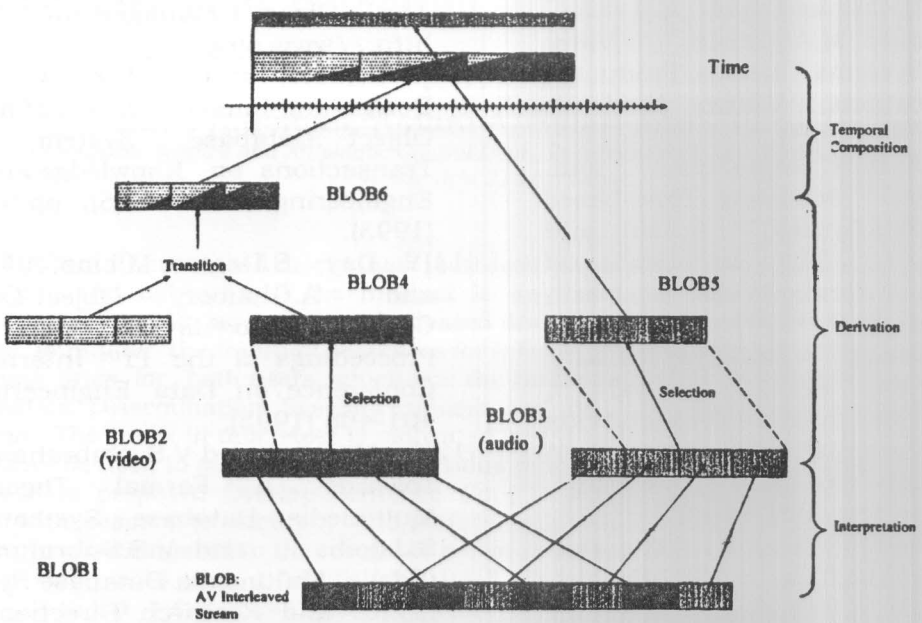

Fig. 28. Creating a composite multimedia instance from a BLOB.

The Object-Oriented paradigm facilitates modeling derivations for each Multimedia type by making use of the concept of Encapsulation. The model supports different physical database designs for storing BLOBs and supports storing the Interpretation of each BLOB in the Multimedia Database. Thus the model supports the integration of the Multimedia data with all the needed analysis and interpretation routines for all the supported media types in the system to make a complete and automated MMDBMS that does not rely on the user to interpret or annotate the audio/video clips or images, etc.

The Unified Modeling Language UML is used to represent the model as it is now the standard language for modeling OO systems. The Class Diagram is used to represent the several classes in the model, as well as, the relationships between them. The functionality of the system is shown as a list of Use Cases and a subset of the Use Case diagram is presented. Moreover, an example scenario is illustrated to show how to implement one of these use cases.

Finally, an example for creating a Composite Multimedia Presentation from a BLOB, using the system classes (abstract concepts) and the associated operations provided as methods of each class, is illustrated.

Therefore, this model presents the first step towards a standard OO model for Multimedia data that solves the most complex problems of Time-based Media.
In our future work, we plan to concentrate on the problem of querying the multimedia database management system and studying the relationship between query processing and implementing such scenarios that model the system functionality. Also, possible system architecture will be proposed.

References

[1]S. Khoshafian, and A.B. Baker, Multimedia and Imaging Databases, Morgan Kauffman Publishing, San Fransisco (1996).

[2]W.I.Grosky, R.Jain, and R.Mehrotra, The Handbook of Multimedia Information Management, Prentice Hall PTR (1997).

[3]H. Jiang, A.K. Elmagarmid, A.A.Helal, A.Joshi and M.A.Ahmed, Video Database Systems: Issues, Products , and Applications, Kluwer Academic Publishers, NY (1997).

[4]S. Gibbs, C.Breiteneder, and D.Tsichritzis, Modeling Time-based Media, in: W.I.Grosky, R.Jain, and R.Mehrotra, The__Handbook of Multimedia Information Management, Prentice Hall PTR (1997).

[5]T. Lee, L.Sheng , T. Bozkaya, N.Blkir, Z. Ozsoyglu, and G.Ozsoyglu , Querying Multimedia Presentations Based on contents", IEEE Transaction on Knowledge and Data Engineering , Vol.11 (3), pp.361-385 (1999).

[6]R.Hamakawa, and A.Atarshi, Composition Models, in: W.I.Grosky, R.Jain, and R.Mehrotra, (Eds) The Handbook of Multimedia Information Management, Prentice Hall PTR (1997).

[7]M.S. Abougabal, Amani.A.Saad, M.A.Ahmed, and M.A.Mohamrrum, Modeling and Querying of Semantic Content-based Multimedia Databases, Proceedings of the 6th International Workshop on Multimedia Information Systems (MIS'00), Chicago, October 26-28, pp.134-145(2000).

[8]G.Booch, J.Rumbaugh, and I. Jacobson, The Unified Modeling Language User Guide, Addison-Wesley Publishing Company (1999).

[9]The Object Management Group. http://www.omg.org/.

[10]E.Oomoto, and K.Tanaka, OVID: Design and Implementation of a Video-Object Database System, IEEE Transactions on Knowledge and Data Engineering, Vol. 5 (5), pp.622-643 (1993).

[11]Y .Day, S.Dagtas, M.Lino, A.Khofar, and A.Ghafoor, Object-Oriented Conceptual Modeling of Video Data, Proceedings of the 11th International Conference on Data Engineering, pp. 401-408 (1995).

[12]S. Marcus and V.S. Subrahmanian, Towards a Formal Theory of Multimedia Database Systems, in: S.Jajodia , and V.S.Subrahmanian, (Eds) Multimedia Database Systems: Issues and Research Directions, NY, Springer-Verlag, pp.1-35 (1995).

[13]N.V.Patel, and I.Sethi, Video Segmentation for Video Data Management" in: W.I.Grosky, R.Jain, and R.Mehrotra , (Eds) The Handbook of Multimedia Information Management, Prentice Hall PTR (1997).

[14]A.Yoshitaka, and T. Ichikawa, A Survey on content-Based Retrieval for Multimedia Databases, IEEE Transactions on Knowledge and Data Engineering , Vol. 11(1), pp. 81-93 (1999).

Received June 7, 2001
Accepted July 17, 2001