Proceedings of International Conference on Distributed
Systems, Software Engineering and Database Systems

Multimedia Database Modeling with Z Notations *

Timothy K. Shih, Huan-Chao Keh and Chao T. Fang-Tsou
Department of Computer Science and Information Engineering
Tamkang University
Tamsui, Taiwan
R.O.C.
email: TSHIH@CS.TKU.EDU.TW
fax: Intl. (02) 620 9749

Abstract

The advantages of using a formal specification are in its precise definition of object properties and its declarative description of operations of a system. The Z notation is used widely in the literatures of formal system designs and the researches of language semantics. In line with the rapid growing of multimedia computation researches, we use the Z notation to design a multimedia database supports the reuse of multimedia presentations. In this paper, a formal specification of the database is presented. The database consists of two layers: the frame layer and the resource layer. A frame group in the first layer is the basic presentation unit to be reused. A resource group in the second layer is a collection of resources to be used by a frame, or a frame group. In the hierarchy, we use four type of links to group frames and/or resources. The database is built on the top of a commercial object-oriented database server and run under a windowing system.

Key words: Multimedia Database, Object Reuse, Multimedia Presentation, *Z* Notation, Formal Specification

1 Introduction

As multimedia personal computers widely available with reasonable prices, multimedia PCs built with sound card, CD ROM, and high resolution graphics display unit are used broadly by business persons, engineers, and others. Due to the impressive sound effect, graphics animation, and video play, the importance and business opportunity of multimedia presentations are realized by researchers and commercial software developers. Presentation design software are also available for affordable prices. However, most presentations generated by these systems communicate with the addressee in a single direction manner. That is, the presentation software does not listen to the listeners' response. For this reason, we have developed an intelligent multimedia presentation design system [8, 9] to overcome the shortage. Our research is to investigate presentation design techniques and to develop a system that helps multimedia presentation designers to deliver intelligent multimedia applications as CD ROM titles. Our research project focuses on the following issues:

- Canonical representation of knowledge
- Intelligent presentation specification language
- Addressee characteristics specification and learning
- Multimedia resources DBMS with reuse controls

Presentation intelligence is represented by a canonical rule-based format. These knowledge not only include the addressee's background (i.e., common sense of the person who watches the presentation), but allow human reactions to be learned by the presentation program. A database management system is also designed for the CD ROM title designers to organize and store multimedia resource information. This database is associated with a presentation reuse control model. An intelligent specification language is designed. The language provides facilities for hypermedia access and rule-based statements for knowledge representation. Our system supports personalization. Not only the graphical user interface of the generated presentation can be fully customized, but the underlying knowledge of the addressee can be easily updated. The system also provides a learning subsystem to be included in the generated title which allows an addressee's interaction to be asserted into the knowledge base.

This paper focuses on the discussion of an object-oriented multimedia presentation/resource database. Section 2 is a survey of related works. A short subsection reviews some notations of *Z* is also given. Section 3 presents the hierarchy of our database and defines the domain of database objects. Some important properties of multimedia resources and presentations are

*This project is supported by the NSC grant # NSC 85-2213-E-032-011 of the Republic of China.

also addressed. We discuss operations of the database in section 4. Finally, a short conclusion and the discussion of our future works are given in section 5.

2 Related Works

A number of researchers developed domain specific presentations using artificial intelligence techniques. For example, COMET (COordinated Multimedia Explanation Testbed) [12, 13] uses a knowledge base and AI techniques to generate coordinated, interactive explanations with text and graphics that illustrates how to repair a military radio receiver-transmitter. WIP [1, 2] is able to generate knowledge-based presentations that explain to a user how to use an espresso machine. The work described in [4] integrates knowledge representation systems and a propositional logic theorem prover to create text and map based illustrations showing the situations and plans of a Navy's fleet. APT (A Presentation Tool) [16, 17] automatically generates graphical presentations of relational information. A Piano tutor described in [10, 11] is able to use coordinated media, video, voice, and graphics display, to teach beginners how to play the piano.

Other articles [18, 5] address relations between multimedia and *multiple modalities*. A *modal* is the way information is presented, such as a natural language statement, one piece of picture, or a table. A medium is the device which presentation is delivered, such as sound, text, or video. The approach described in [5] suggests that the mapping should be constructed between characteristics of media and characteristics of information. The research discussed in [5] also proposes that presentation knowledge can be classified naturally into four major groups: characteristics of media, characteristics of information, goal of the presenter, and the background of addressees. Issues related to solving the synchronization problems of temporal multimedia resources can be found in [20, 6]. A multimedia collaborative design environment for scientific and engineering applications can be found in [3]. The work described in [23] is a learning environment for the second year French students to learn about French culture. This multimedia application uses maps and visual icons as well as video files to show locations that can be visited in a city.

The related works addressed above are mostly academic researches. On the other hand, we also looked at some commercial products related to multimedia authoring or presentation designs:

1. Authorware Professional by Macromedia, Inc.
2. Multimedia Viewer by Microsoft
3. Multimedia Toolbook by Asymetrix Corporation
4. Hypermedia System by ITRI (Taiwan)
5. Action! by Macromedia, Inc.
6. Audio Visual Connection by IBM
7. Astound by Gold Disk Inc.
8. Director by Macromedia, Inc.

Authorware uses an event control flow diagram allowing the presenter to specify presentation objects and controls, which can be decomposed into several levels in a hierarchy structure. The system also provides a simple script language for calculation and data manipulation. Other systems (i.e., 2, 3, and 6 above) also provide script languages and API (application program interface) functions. Hypermedia System, Action!, and Director use a time line table allowing actions or objects to be dropped in a particular time slot. Most systems allow users to cut and paste presentation objects or actions via button click and drawing. Multimedia Viewer also provides a set of medium editing tools. Presentation objects produced by these tools can be linked together by a script language supporting functions, data structures, and commands. None of the above systems, however, allows the interactive sequences provided by a user to be learned by a presentation.

To support the creation of a good multimedia presentation, many articles suggest that a multimedia database supporting fast indexing and synchronization is essential [7, 19, 21, 25]. The discussion of issues in multimedia database management systems can be found in [19]. A distributed database supporting the development of multimedia applications is found in [7]. A mechanism for formal specification and modeling of multimedia object composition is found in [15]. The work discussed in [15] also consider the temporal properties of multimedia resources. A database system for video objects is discussed in [14]. A content-based querying mechanism for retrieving images is given in [25]. Layered multimedia data modeling [22] suggests a good mechanism to manage multimedia data.

2.1 A Brief Review of Z

Z notation [24] is a language for expressing formal specifications of systems. It is based on typed set theory, coupled with a structuring mechanism (i.e., the schema calculus is one of its key features). A schema introduces a named collection of variables and relationships among variables that are specified by axiom definitions. Schemas are used to describe both the static aspects of a system (e.g., the structure of a program) and the dynamic aspects (e.g., the execution). Schemas can be generic thus polymorphic functions can be defined. Every variable introduced in a *Z* specification is given a type. These types can be given set names or can be constructed by type constructors (e.g., tuple, schema product, or the power set constructors). Free type definitions[1] add nothing to the power of the *Z* lan-

[1] Free type definitions are discussed in [24] as a short mechanism to introduce new types in Z.

guage but ease the definition of recursive objects. Free type definitions can be translated into the other Z constructs. An abbreviation definition using symbol "==" introduces a new global constant. The identifier on the left becomes a global constant, and its value is given by the expression on the right. An axiomatic description introduces one or more global variables and optionally specifies a constraint on their values. In the discussion follows, we will introduce concepts and some syntax of Z notation while necessary.

The reason we take a formal specification approach is that formal specifications use mathematical notation to precisely describe **what** properties a system need to have, without concerning too much about **how** these properties are implemented. This approach avoids our discussion of the system from being too tedious. Z notation is a formal specification language widely used in Europe. The notation we use here follows the standard given in [24]. We find that, by using the mathematical tool-kit provided in [24], we can easily describe the formal specification of our multimedia database.

3 The Database Object Domain

We begin our discussion of database modeling with the attributes of database objects, followed by the hierarchy of the database. In the discussion, following most of the specifications in Z notation founded in the literature, a domain has its name all in capital, a schema name starts with a capital letter followed by lower case letters, and variables or functions are all in lower case.

The hierarchy of the proposed multimedia database consists of two layers: the *frame object layer* and the *resource object layer*. The two types of objects in our database are frames and resources. A frame is denoted by a box while a resource is represented by a circle with its associated properties given in an attached rounded box. The actual data of a resource is stored in the commercial multimedia file format on the hard disk[2].

The frame object layer contains presentation frames. A frame, similar to a presentation window, holds a number of presentation items, such as video clips, music background, etc. These presentation items, however, are stored in the resource object layer. In the frame object layer, each frame is associated with a number of attributes:

- **name**: a unique name of the frame.
- **keyword**: one or more keywords are used as the description of the frame.
- **inheritance links**: pointers to other frames which inherit properties from the current frame.
- **usage links**: messages from the current frame to the destination frames, including possible parameters.
- **aggregation links**: pointers to resources which are used in the current frame.
- **presentation knowledge**: logic facts, rules, and a query used in the frame when the frame is open [9]. Note that presentation properties can be represented as logic facts.
- **frame layouts**: screen coordinates of resources.

[2] For the current implementation, our database still relies on the file system. However, we are seeking for techniques that allow fast indexing of multimedia resources for the future versions of our multimedia database.

Now, we present the formal specification of our multimedia database. Firstly, we need to define the domain of objects. In Z notation, *basic types* are declared in a pair of square parentheses. Basic types are given with no detail specification of how they are represented. As given in the following declaration, messages, frame layouts, and predicates (used in our presentation inference system) are given as basic types:

$$[MSG, LAYOUT, P]$$

Similarly, the detail formats of multimedia resource binary data, ASCII strings, or integers are declared as basic types:

$$[MRDATA, ASCII, INTEGER]$$

Using these basic types, we use abbreviation definitions in Z to express the domain of other objects. Frame names ($FNAME$) and keywords ($KEYWORD$) are ASCII strings. The knowledge set of a presentation consists of three types of objects: a set of rules (KR), a set of facts (KF), and a set of queries (KQ). Note that, in Z, $\mathbb{P}\, X$ denotes a set of objects in domain X. And, $\mathbb{P}_1\, Y$ represents a non-empty set of objects in domain Y. We treat the knowledge set of a frame as a number of facts, rules, and queries, which are similar to the ones used in the logic programming language Prolog:

$$\begin{aligned} &FNAME == ASCII \\ &KEYWORD == ASCII \\ &KNOWLEDGE == \mathbb{P}\, KR \cup \mathbb{P}\, KF \cup \mathbb{P}\, KQ \\ &KR == P \times \mathbb{P}_1\, P \\ &KF == P \\ &KQ == P \end{aligned}$$

In the hierarchy, objects are connected by links. Four types of links are used, as discussed in the following definitions.

Definition: An *inheritance link* represents a property inheritance between two frames. ■

Inheritance links are used in the process of knowledge collection of an activated frame before the logical inference of the frame proceeds [8, 9]. A message, with or without parameters, shows the usage relation between two frames.

Definition: A *usage link* is a link which represents a message passed between two frames. ■

Definition: An *aggregation link* indicates that a frame is using a resource. ■

An aggregation link connects frames and resources in the two layers of the database. An association may exist in between two resources. For example, an animation resource is associated with a MIDI (Musical Instrument Digital Interface) resource which is used as the background music.

Definition: An *association link* is used between two resources which are corelated. ■

Some of the links are divided into two categories: internal and external, as to support the presentation reuse process (to be discussed in section 4). Thus, a link type ($LTYPE$) is defined by a *free type definition*. A free type definition introduces a new data type in Z. It is useful especially in recursive type definitions. However, the $LTYPE$ is a type contains only constants: *internal* and *external*. No recursive specification is used. Denoted by four abbreviation definitions, the four kind of links are defined:

$$LTYPE ::= internal \mid external$$
$$IL == FNAME \times FNAME \times LTYPE$$
$$UL == FNAME \times FNAME \times MSG \times LTYPE$$
$$AGL == FNAME \times RNAME$$
$$ASL == RNAME \times RNAME \times LTYPE$$

Finally, a frame is defined by an abbreviation again:

$$FRAME == FNAME \times \mathbb{P}_1\, KEYWORD \times \mathbb{P}\, IL \times \mathbb{P}\, UL \times \mathbb{P}\, AGL \times KNOWLEDGE \times LAYOUT$$

To create a high quality multimedia presentation, not only a good presentation design environment is essential, but good multimedia resources are the key. Multimedia resources are recorded or captured via camera, tape recorder, or video camera, converted to their digital formats, and saved on the disk. These resource files can be reused in different presentations. A resource is associated with a number of attributes. We consider the following attributes for objects in the resource object layer of our database:

- **name**: a unique name of the resource.
- **keyword**: one or more keywords are used as the description of a multimedia resource. For instance, name of the city is a keyword of the bitmapped picture of Paris.
- **usage**: how the resource is used (e.g., background, navigation, or focus).
- **medium**: what multimedia device is used to carry out this resource (e.g., sound, video, MPEG, or picture).
- **model**: how the resource is presented (e.g., table, map, chart, or spoken language).
- **temporal endurance**: how long does the resource last in a presentation (e.g., 20 seconds or permanent).
- **synchronization tolerance**: how does a participant feel about the synchronization delay of a resource. For instance, a user usually expects the immediate response after pushing a button for the next page of text. But, the user might be able to tolerate for a video play being delayed for two seconds.
- **detectability**: how does the resource attract a listener (e.g., high, medium, or low).
- **startup delay**: the duration between a message is issued and the corresponding resource is presented, especially when the resource is on a remote computer connected via network.
- **hardware limitation**: what kind of hardware is essential for carrying out the resource (e.g., MPC level 1, level 2, level 3, or other limitations).
- **version**: the version of this resource file.
- **date/time**: the date and time this resource file is created.
- **resolution**: the resolution of this resource file, specified by $X \times Y$ screen units, or 8-bit/16-bit sound.
- **start/end time**: for non-permanent resources, the starting cycle and the ending cycle of the piece of video, sound, or other resources that can be used, especially used as a presentation resource. A cycle can be a second, one-tenth of a second, or a frame number of a video/animation.
- **resource descriptor**: a logical descriptor to a physical resource data segment on the disk.
- **association links**: pointers to other resources who have the coexistence relation with the current resource.

These attributes are either declared as an abbreviation, or defined as a free type. Note that, the free type, $DEGREE$, contains three constants indicate the intensities of acceptance with respect to an attribute. For instance, a synchronization tolerance could be *small*, *medium*, or *large*:

$DEGREE ::= small \mid medium \mid large$
$RNAME == ASCII$
$USAGE ==$
$\quad background \mid navigation \mid focus \mid illustration \mid \ldots$
$MEDIUM ::=$
$\quad sound \mid video \mid animation \mid text \mid picture \mid midi$
$MODEL == table \mid map \mid chart \mid spoken\_language \mid \ldots$
$TENDURANCE == INTEGER \cup \{ \infty \}$
$STOLERANCE == DEGREE$
$DETECTABILITY == DEGREE$
$SDELAY == INTEGER$
$HLIMITATION == ASCII$
$VERSION == INTEGER$
$DATETIME == INTEGER \times INTEGER$
$RESOLUTION ==$
$\quad (INTEGER \times INTEGER) \cup INTEGER$
$SETIME == INTEGER \times INTEGER$
$RDESCRIPTOR == ASCII$

Thus, the attributes of a multimedia resource is a Cartesian product of the above types. And, a multimedia resource is defined below:

$$\begin{aligned} RATTR == {} & RNAME \times \mathbb{P}_1\, KEYWORD \times USAGE \times \\ & MEDIUM \times MODEL \times TENDURANCE \times \\ & STOLERANCE \times DETECTABILITY \times \\ & SDELAY \times \mathbb{P}_1\, HLIMITATION \times \\ & VERSION \times DATETIME \times \\ & RESOLUTION \times SETIME \times \\ & RDESCRIPTOR \times \mathbb{P}\, ASL \end{aligned}$$

$$RESOURCE == RATTR \times MRDATA$$

The above attributes in the two object layers are used in database queries to retrieve suitable objects for a multimedia presentation. Presentation knowledge and frame layouts are provided by the user via our graphical user interface [9, 8]. In section 4, we present semantics of the reuse process in our database.

4 Semantics of Object Reuse

Based on some early experiences of designing multimedia presentations, we found that most multimedia authoring systems either provide a very primitive reuse function, or support no reuse mechanism at all. In the development of our multimedia presentation system, object reuse becomes one of the important goal for us to achieve. In this section, following the object domains and the hierarchy discussed in section 3, we present the functions support object reuse in our database.

The reuse of objects in our system is based on the concept of *object groups*. An object group is a collection of objects which serves as the basic unit in a presentation. For example, a piece of presentation showing the history of computers consists of several frames associated with a number of resources used in the frames is defined as a reusable object. This piece of presentation can be reused in several computer related presentations. Two types of groups are defined in the database: the *frame groups* and the *resource groups*. A frame group or a resource group could become a reusable object stored in the database, if the group is declared by the user via our graphical user interface. After the declaration, an object group becomes an *object class*. These *frame object class* and *resource object class* are reused when instantiated.

Before an object group is stored as an object class in the database, some links will be discarded while some links will be maintained. For the frames in a frame group, the usage (or inheritance) links are divided into two parts: the *internal usage (or inheritance) links* and the *external usage (or inheritance) links*. An internal usage (or inheritance) link has its source and destination frames both belong to the same frame group. An external usage (or inheritance) link, on the other hand, has its destination frame outside the frame group. When a frame group becomes a frame object class, our system keeps the internal inheritance links, the internal usage links, and the aggregation links. But the system will discard all external links of the frame group. The similar concept is applied to a resource group. When a resource group becomes a resource object class, our system keeps the internal association links and discards all external association links.

The process of instantiation requires the allocation of memory or disk storage for the new instantiated object group, and the declaration of external links to other objects in the presentation. For a frame group instantiation, the external usage links and the external inheritance links will be restored by the user. For a resource group instantiation, the external association links need to be given. An instantiation process creates a new object group with new links connected to other objects in the presentation. But, the instantiation process does not duplicate information that can be shared among instances, such as frame layouts or the actual resource data stored on the disk.

In the following Z abbreviation definitions, we declare resource groups and resource object classes as nonempty sets of resources. Frame groups and frame object classes are treated as non-empty sets of frames. A multimedia database is thus declared as the Cartesian product of resource object class set and frame object class set. A multimedia presentation, on the other hand, is declared as the Cartesian product of resource group set and frame group set:

$$RGROUP == \mathbb{P}_1\ RESOURCE$$
$$RCLASS == RGROUP$$
$$FGROUP == \mathbb{P}_1\ FRAME$$
$$FCLASS == FGROUP$$
$$MDATABASE == \mathbb{P}_1\ RCLASS \times \mathbb{P}_1\ FCLASS$$
$$PRESENTATION == \mathbb{P}_1\ RGROUP \times \mathbb{P}_1\ FGROUP$$

The reuse process of database objects are introduced by four axiomatic descriptions of Z . An axiomatic description introduces global variables, with possible restrictions on their values. The global function *make_resource_class* takes as input a resource group and produces a resource object class, as specified in the following axiom:

$$make\_resource\_class : RGROUP \rightarrow RCLASS$$

$$
\begin{array}{l}
\forall\, rg : RGROUP \bullet \forall\, rc : RCLASS \bullet \\
make\_resource\_class(rg) = rc \Leftrightarrow \\
(\forall\, r, r_1 : RESOURCE \bullet r \in rg \wedge r_1 \in rc \wedge \\
r = \langle ra, mrdata \rangle \wedge r_1 = \langle ra_1, mrdata \rangle \wedge \\
ra = \langle ..., asl\_set \rangle \wedge ra_1 = \langle ..., asl\_set_1 \rangle \wedge \\
(\forall\, asl, asl_1 : ASL \bullet asl \in asl\_set \wedge asl_1 \in asl\_set_1 \bullet \\
asl = \langle rn_1, rn_2, internal \rangle \Rightarrow asl \in asl\_set_1 \vee \\
asl = \langle rn_1, rn_2, external \rangle \Rightarrow < rn_1, \perp, external \rangle \\
\in asl\_set_1))
\end{array}
$$

Note that, the $\perp$ sign represents an unknown value, which means the destination resources of all association links are discarded. The global function *instantiate_resource_group* takes an opposite operation. It instantiates a resource group from a resource object class. In the instantiation, the destination resources are restored by the user:

$$instantiate\_resource\_group : RCLASS \rightarrow RGROUP$$

$$
\begin{array}{l}
\forall\, rc : RCLASS \bullet \forall\, rg : RGROUP \bullet \\
instantiate\_resource\_group(rc) = rg \Leftrightarrow \\
(\forall\, r, r_1 : RESOURCE \bullet r \in rc \wedge r_1 \in rg \wedge \\
r = \langle ra, mrdata \rangle \wedge r_1 = \langle ra_1, mrdata \rangle \wedge \\
ra = \langle ..., asl\_set \rangle \wedge ra_1 = \langle ..., asl\_set_1 \rangle \wedge \\
(\forall\, asl, asl_1 : ASL \bullet asl \in asl\_set \wedge asl_1 \in asl\_set_1 \bullet \\
(\exists\, given\_resource\_name : RNAME \bullet \\
asl = \langle rn_1, rn_2, internal \rangle \Rightarrow asl \in asl\_set_1 \vee \\
asl = \langle rn_1, \perp, external \rangle \Rightarrow \\
\langle rn_1, given\_resource\_name, external \rangle \in asl\_set_1)))
\end{array}
$$

In the same manner, frame object classes and frame groups are defined and instantiated. However, different type of links are discarded and restored:

$$make\_frame\_class : FGROUP \rightarrow FCLASS$$

$$
\begin{array}{l}
\forall\, fg : FGROUP \bullet \forall\, fc : FCLASS \bullet \\
make\_frame\_class(fg) = fc \Leftrightarrow \\
(\forall\, f, f_1 : FRAME \bullet f \in fg \wedge f_1 \in fc \wedge \\
f = \langle fname, keywords, il\_set, ui\_set, \\
\quad agl\_set, k, layout \rangle \wedge \\
f_1 = \langle fname, keywords, il\_set_1, ui\_set_1, \\
\quad agl\_set, k, layout \rangle \wedge \\
(\forall\, il, il_1 : IL \bullet il \in il\_set \wedge il_1 \in il\_set_1 \bullet \\
il = \langle fn_1, fn_2, internal \rangle \Rightarrow il \in il\_set_1 \vee \\
il = \langle fn_1, fn_2, external \rangle \Rightarrow \langle fn_1, \perp, external \rangle \\
\quad \in il\_set_1) \wedge \\
(\forall\, ul, ul_1 : UL \bullet ul \in ul\_set \wedge ul_1 \in ul\_set_1 \bullet \\
ul = \langle fn_1, fn_2, m, internal \rangle \Rightarrow ul \in ul\_set_1 \vee \\
ul = \langle fn_1, fn_2, m, external \rangle \Rightarrow \langle fn_1, \perp, \perp, external \rangle \\
\quad \in ul\_set_1))
\end{array}
$$

$$instantiate\_frame\_group : FCLASS \rightarrow FGROUP$$

$$
\begin{array}{l}
\forall\, fc : FCLASS \bullet \forall\, fg : FGROUP \bullet \\
instantiate\_frame\_group(fc) = fg \Leftrightarrow \\
(\forall\, f, f_1 : FRAME \bullet f \in fc \wedge f_1 \in fg \wedge \\
f = \langle fname, keywords, il\_set, ui\_set, \\
\quad agl\_set, k, layout \rangle \wedge \\
f_1 = \langle fname, keywords, il\_set_1, ui\_set_1, \\
\quad agl\_set, k, layout \rangle \wedge \\
(\forall\, il, il_1 : IL \bullet il \in il\_set \wedge il_1 \in il\_set_1 \bullet \\
(\exists\, given\_frame\_name : FNAME \bullet \\
il = \langle fn_1, fn_2, internal \rangle \Rightarrow il \in il\_set_1 \vee \\
il = \langle fn_1, \perp, external \rangle \Rightarrow \\
\quad \langle fn_1, given\_frame\_name, external \rangle \\
\quad \in il\_set_1)) \wedge \\
(\forall\, ul, ul_1 : UL \bullet \\
ul \in ul\_set \wedge ul_1 \in ul\_set_1 \bullet \\
(\exists\, given\_frame\_name : FNAME \bullet \\
\exists\, given\_message : MSG \bullet \\
ul = \langle fn_1, fn_2, m, internal \rangle \Rightarrow ul \in ul\_set_1 \vee \\
ul = \langle fn_1, \perp, \perp, external \rangle \Rightarrow \\
\langle fn_1, given\_frame\_name, given\_message, external \rangle \\
\quad \in ul\_set_1)))
\end{array}
$$

Now, we use the following schemas to define the construction and reuse process of presentations. As shown in between two horizontal lines leading by a name, schema *ConstructPresentation* holds a multimedia presentation *mmp* in domain *PRESENTATION* and a multimedia database *mdb*, where the presentation extracts presentation resources. The two instantiation functions *instantiate_resource_group* and *instantiate_frame_group* are used in creating the presentation. Similarly, schema *ReusePresentation* presents the reuse process:

ConstructPresentation

$$mmp : PRESENTATION$$
$$mdb : MDATABASE$$

$$\begin{aligned}
&(mmp = \langle rgroup_set, fgroup_set\rangle \land \\
&mdb = \langle rclass_set, fclass_set\rangle) \Rightarrow \\
&(\forall\, rgroup : RGROUP \bullet \forall\, fgroup : FGROUP \bullet \\
&\forall\, rclass : RCLASS \bullet \forall\, fclass : FCLASS \bullet \\
&(rgroup \in rgroup_set \land fgroup \in fgroup_set \land \\
&rclass \in rclass_set \land fclass \in fclass_set) \Rightarrow \\
&(rgroup = instantiate_resource_group(rclass) \land \\
&fgroup = instantiate_frame_group(fclass)))
\end{aligned}$$

ReusePresentation

$$mmp : PRESENTATION$$
$$mdb : MDATABASE$$

$$\begin{aligned}
&(mmp = \langle rgroup_set, fgroup_set\rangle \land \\
&mdb = \langle rclass_set, fclass_set\rangle) \Rightarrow \\
&(\exists\, rgroup : RGROUP \bullet \exists\, fgroup : FGROUP \bullet \\
&\exists\, rclass : RCLASS \bullet \exists\, fclass : FCLASS \bullet \\
&(rgroup \in rgroup_set \land fgroup \in fgroup_set \land \\
&rclass \in rclass_set \land fclass \in fclass_set) \Rightarrow \\
&(rclass = make_resource_class(rgroup) \land \\
&fclass = make_frame_class(fgroup)))
\end{aligned}$$

The management of object reuse is assisted by our presentation reuse control interface and resource browser. The user is able to use our system to organize their presentations and resources systematically.

5 Conclusions

An object-oriented multimedia database is introduced. The database system is to support the design of intelligent multimedia presentations constructed by using our presentation system. A presentation consists of a number of frame groups. The generalization of a frame group, named a frame object class, serves as the basic reusable unit of a multimedia presentation. Similarly, the generalization of a resource group, or an individual resource as a group, can be reused. This reuse mechanism, supported by a graphical user interface, allows a user to organize his/her multimedia resources and presentation pieces easily. The intelligent multimedia presentation system as well as its supporting database run under the MS Windows 95[3]. The database hierarchy consists of two layers: the frame object layer and the resource object layer. Another article [22] also proposes this layered approach for multimedia data modeling. However, the reuse of multimedia objects are not fully discussed. Our approach focuses not only on the hierarchy of the database, but the mechanism of reusing multimedia objects.

We are looking at the semantic modeling of multimedia presentations. As indicated in [15], the temporal properties of multimedia resources and presentations need to be considered in the semantic model. We are using interval temporal logic to express the temporal properties of multimedia resources and solving the application level synchronization problems. Another area of our future work is to investigate database indexing techniques for the fast retrieval of multimedia data.

The contributions of this paper are, firstly, we propose a database hierarchy uses an object-oriented approach. A formal specification of the database is then defined. This database is implemented on the top of the ObjectPro/ODB database management system to support our intelligent multimedia presentation designs. Finally, the reuse mechanism allows the user of our system to organize and reuse their presentation easily. We believe that, the proposed mechanism provides a hint to the development of multimedia database in the literature of multimedia computing.

[3] Our future version of database will run under the Windows NT server

References

[1] E. Andre, et. al., "WIP: The Automatic Synthesis of Multimodal Presentations," In Intelligent Multimedia Interfaces, edited by Mark T. Maybury, American Association for Artificial Intelligence, pp 75–93, 1993.

[2] E. Andre and T. Rist, "The Design of Illustrated Documents as a Planning Task," In Intelligent Multimedia Interfaces, edited by Mark T. Maybury, American Association for Artificial Intelligence, pp 94–116, 1993.

[3] V. Anupam and C. L. Bajaj, "Shastra: Multimedia Collaborative Design Environment," IEEE Multimedia, summer, 1994.

[4] Y. Arens, et. al., "Presentation Design Using an Integrated Knowledge Base," In Intelligent User Interfaces, edited by J. W. Sullivan and S. W. Tyler, ACM Press, pp 241–258, 1991.

[5] Y. Arens, et. al., "On the Knowledge Underlying Multimedia Presentations," In Intelligent Multimedia Interfaces, edited by Mark T. Maybury, American Association for Artificial Intelligence, pp 280–306, 1993.

[6] Herng-Yow Chen, et. al., "A Novel Audio/Video Synchronization Model and Its Application in Multimedia Authoring System," In Proceeding of the 1994 HD-MEDIA Technical and Applications Workshop, October 6-8, Taipei, Taiwan, 1994.

[7] Te-Chih Chen, Wei-Po Lin Chin-An Wu, and Chih-Shen Shen, "A Client-Server Database Enviornment for Supporting Multimedia Applications," in proceedings of the 18th IEEE annual international computer software and application conference (COMPSAC'94), Taipei, Taiwan, pp 215–220, 1994.

[8] Chi-Ming Chung, Timothy K. Shih, Jiung-Yao Huang, Ying-Hong Wang, and Tsu-Feng Kuo, "An Object-Oriented Approach and System for Intelligent Multimedia Presentation Designs," In proceedings of the ICMCS'95 conference, pp 278–281, 1995.

[9] Timothy K. Shih, "An Artificial Intelligent Approach to Multimedia Authoring," In proceedings of the second IASTED/ISMM international conference on distributed multimedia systems and applications, Stanford, California, August 7-9, pp 71–74, 1995.

[10] R. B. Dannenberg, et. al., "A Computer Based Multimedia Tutor for Beginning Piano Students," In Interface 19(2-3), pp 155–173, 1990.

[11] R. B. Dannenberg and R. L. Joseph, "Human Computer Interaction in the Piano Tutor," In Multimedia Interface Design, edited by M. M. Blattner and R. B. Dannenberg, ACM Press, pp 65–78, 1992.

[12] S. K. Feiner and K. R. McKeown, "Automating the Generation of Coordinated Multimedia Explanations," In Intelligent Multimedia Interfaces, edited by Mark T. Maybury, American Association for Artificial Intelligence, pp 117–138, 1993.

[13] S. K. Feiner, et. al., "Towards Coordinated Temporal Multimedia Presentations," In Intelligent Multimedia Interfaces, edited by Mark T. Maybury, American Association for Artificial Intelligence, pp 139–147, 1993.

[14] Keh-Feng Lin, Chueh-Wei Chang, and Suh-Yin Lee, "Design of an Interactive Video Database," in proceedings of the 1994 HD-Media Tech. and Application workshop, Taipei, Taiwan, pp PO2-17–PO2-22.

[15] Thomas D. C. Little and Arif Ghafoor, "Synchronization and Storage Models for Multimedia Objects," IEEE Journal on Selected Areas in Communications, Vol. 8, No. 3, April, pp 413–427, 1990.

[16] J. D. Mackinlay, "Automatic Design of Graphical Presentations," Ph. D. thesis, Stanford University, 1987.

[17] J. D. Mackinlay, "Search Architectures for the Automatic Design of Graphical Presentations," In Intelligent User Interfaces, edited by J. W. Sullivan and S. W. Tyler, ACM Press, pp 281–292, 1991.

[18] J. G. Neal and S. C. Shapiro, "Knowledge Based Multimedia Systems," In Multimedia Systems, edited by J. F. K. Buford, ACM Press, pp 403–438, 1994.

[19] Raymond Paul, M. Farrukh Khan, Ashfaq Khokhar, and Arif Ghafoor, "Issues in Database Management of Multimedia Information," in proceedings of the 18th IEEE annual international computer software and application conference (COMPSAC'94), Taipei, Taiwan, pp 209–214, 1994.

[20] B. Prabhakaran and S. V. Raghavan, "Synchronization models for multimedia presentation with user participation," Multimedia Systems, Vol. 2, pp 53–62, Springer-Verlag, 1994.

[21] J. A. Rody and A. Karmouch, "A Remote Presentation Agent for Multimedia Database," in Proceedings of the IEEE 1995 International Conference on Multimedia Computing and Systems, May 15-18, Washington DC, pp 223–230, 1995.

[22] Gerhard A. Schloss and Michael J. Wynblatt, "Presentation Layer Primitives for the Layered Multimedia Data Model," in Proceedings of the IEEE 1995 International Conference on Multimedia Computing and Systems, May 15-18, Washington DC, pp 231–238, 1995.

[23] E. Schlusselberg, "Dans le Quartier St. Gervais: An Exploratory Learning Environment," In Multimedia Computing: Case Studies from MIT Project Athena, eited by M. E. Hodges, and R. M. Sasnett, Addison-Wesley, pp 103– 115, 1993.

[24] J. Michael, Spivey, *The Z Notation: A Reference Manual.* International Series in Computer Science. Prentice Hall, 1989.

[25] Atsuo Yoshitaka, Setsuko Kishida, Masahito Hirakawa, and Tadao Ichikawa, "Knowledge-Assisted Content-Based Retrieval for Multimedia Database," IEEE Multimedia Magazine, pp 12–21, Winter 1994.

[26] ObjectPro/ODB reference manual, Institute for Information Industry, Taiwan, 1995.