

3-18-2002

Multimedia presentation authoring and browsing in multimedia database systems

Xiaoming Chen

Florida International University

Follow this and additional works at: <http://digitalcommons.fiu.edu/etd>



Part of the [Computer Sciences Commons](#)

Recommended Citation

Chen, Xiaoming, "Multimedia presentation authoring and browsing in multimedia database systems" (2002). *FIU Electronic Theses and Dissertations*. 2146.

<http://digitalcommons.fiu.edu/etd/2146>

This work is brought to you for free and open access by the University Graduate School at FIU Digital Commons. It has been accepted for inclusion in FIU Electronic Theses and Dissertations by an authorized administrator of FIU Digital Commons. For more information, please contact dcc@fiu.edu.

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

MULTIMEDIA PRESENTATION AUTHORING AND BROWSING IN
MULTIMEDIA DATABASE SYSTEMS

A thesis submitted in partial fulfillment of the

requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

by

Xiaoming Chen

2002

DEDICATION

I dedicate this thesis to my husband Danny and my parents. Without their understanding, support, and all of love, the completion of this work would not have been possible.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my major professor, Dr. Shu-Ching Chen, whose academic guide, stimulating suggestions and encouragement helped me in all the time of research for and writing of this thesis. I wish to thank Dr. Masoud Milani for taking the time to review my thesis and for his constructive comments. I would like to thank Dr. Nagarajan Prabakar for the efforts to evaluate my thesis and his valuable instruction.

I want to thank my colleagues Jeff strickrott, Lin Luo and Chengcui Zhang for their help, interest and valuable hints in my research work. Also, I would like to take this opportunity to thank cordially everyone who helped me throughout the years at Florida International University.

Especially, I would like to give my special thanks to my husband Danny whose patient love enabled me to complete this work.

ABSTRACT OF THE THESIS
MULTIMEDIA PRESENTATION AUTHORIZING AND BROWSING IN
MULTIMEDIA DATABASE SYSTEMS

by

Xiaoming Chen

Florida International University, 2002

Miami, Florida

Professor Shu-Ching Chen, Major Professor

The purpose of this study was to design and implement multimedia presentation authoring and browsing in multimedia database systems, based on a two-tier client/server architecture.

The major approaches included: (i) Build a client/server multimedia presentation authoring and browsing environment; (ii) Allow query-by-image image retrieval and domain-based browsing through the User Datagram Protocol (UDP) between the Java socket at the client and the Unix socket at the server; (iii) Allow the multimedia presentation authoring by unifying the temporal, spatial and spatio-temporal relations among various media items into Multimedia Augmented Transition Network (MATN) model; (iv) Support the user interaction during the authoring process; (v) Visualize the presentation based on the created MATN models.

As a result, this methodology provided users a flexible environment for heterogeneous media data retrieval and multimedia presentation authoring in multimedia database systems.

TABLE OF CONTENTS

CHAPTER	PAGE
1. Introduction.....	1
1.1 Characteristics of multimedia data.....	3
1.1.1 Temporal and spatial dimensionality	4
1.1.2 Temporal and spatial issues	5
1.2 Scope of the thesis.....	7
1.2.1 Multimedia data querying and browsing	8
1.2.2 Multimedia presentation modeling	9
1.2.3 Visualization of the modeling	12
1.3 Structure of the thesis.....	12
2. Literature Review.....	14
2.1 Multimedia Querying and Browsing.....	15
2.2 Modeling Multimedia Presentation.....	17
2.2.1 Timeline-based Model	17
2.2.2 Flowchart-based model	20
2.2.3 Structure-based model.....	23
2.2.4 Language-based model	27
2.2.5 ATN-based model	29
2.3 Conclusion	31
3. System Architecture.....	33
3.1 The architecture of the client	34
3.1.1 The Browser.....	35
3.1.2 The Editor	39
3.1.3 The player	44
3.2 The architecture of the server.....	45
3.3 Communication path.....	46
3.4 Conclusion	46
4. Multimedia Querying and Browsing.....	48
4.1 Requirements for information retrieval.....	48
4.2 Management of the information retrieval	50
4.2.1 Querying.....	52
4.2.2 Browsing	57
4.3 Communication between the server and the client	63
4.4 Conclusion	69
5. Multimedia presentation authoring	70
5.1 General process of multimedia presentation authoring.....	70
5.2 MATN	71

5.2.1 Graphic view and grammar.....	72
5.2.2 Time flow in MATN.....	73
5.2.3 Data structure of MATN.....	75
5.3 The editor interface for presentation composition.....	78
5.3.1 Media items management.....	80
5.3.2 Model editing.....	82
5.3.3 User interaction.....	87
5.4 Multimedia presentation visualization.....	89
5.5 Conclusion.....	91
6. Conclusions and Future Work.....	92
List of References.....	95

LIST OF TABLES

TABLE	PAGE
Table 4.1 Pseudo code for directory tree	61
Table 4.2 Pseudo code for file browsing.....	62
Table 4.3 Pseudo code for Java socket	66
Table 4.4 The packet header constructure	66
Table 4.5 Pseudo code for datagram packet in the client side	67
Table 5.1 Attributes of objects <i>State</i>	77
Table 5.2 The attributes of object <i>Arc</i>	77

LIST OF FIGURES

FIGURE	PAGE
Figure 1.1 Media items in the three-dimensional space.....	5
Figure 1.2 The Allen time relations	6
Figure 2.1 A timeline-based model.....	18
Figure 2.2 Director.....	19
Figure 2.3 Flowchart-based model.....	20
Figure 2.4 A view of Authorware	22
Figure 2.5 Structure-based model	24
Figure 2.6 MAD (Movie Authoring and Design) script view.....	26
Figure2.7 Language-based model	28
Figure 2.8 ATN with user interaction and sub-network	30
Figure 3.1 Overall view of the system architecture	33
Figure 3.2 Three layers in the client side	34
Figure 3.3 The browser interface	36
Figure 3.4 The former interface of the browser	36
Figure 3.5 Browse-by-image.....	38
Figure 3.6 Browse data catalog Hurricane.....	38
Figure 3.7 Download category of hurricane from server.....	39
Figure 3.8 The screen view of the editor	40
Figure 3.9 The manipulation states of the editor	41
Figure 3.10 Edit a MATN model.....	42

Figure 3.11 State property dialog.....	42
Figure 3.12 Arc property.....	43
Figure 3.13 User selection	43
Figure 3.14 The screen view of the player.....	44
Figure 4.1 The iterative process of information retrieval	50
Figure 4.2 The first version of query module	52
Figure 4.2 The new version of browser interface in client side.....	55
Figure 4.3 Sending a image query	56
Figure 4.4 The query results with the target image in front.....	56
Figure 4.5 Find the selected file from the browsing window for Hurricane.....	57
Figure 4.6 The hierarchical structure of database in server site.....	59
Figure 4.7 Choice box for subject selection.....	60
Figure 4.8 The screen view of the archive tree for subject “Hurricane”.....	61
Figure 4.9 The screen view when category downloading finished.....	62
Figure 4.10 The system output of the server when request file transmission.....	63
Figure 4.11 The communication routine between the server and the client	66
Figure 4.12 Sending and receiving in client site.....	68
Figure 5.1 MATN graphic form and grammar	73
Figure 5.2 The time flow in a MATN.....	74
Figure 5.3 The objects in Matn package.....	76
Figure 5.4 The screen view of the editor	79
Figure 5.5 Dialog for selecting media item from the directory tree	81
Figure 5.6 Dialog for selected items list	81

Figure5.7 Draw model after confirm the selection	81
Figure 5.8 Dialog for creating branches.....	82
Figure 5.9 Branches composition.....	83
Figure 5.10 Complete branches composition.....	84
Figure 5.11 Insertion of a new state	84
Figure 5.12 The process of deleting media item from a label.....	86
Figure 5.13 The process of deleting an arc directly	87
Figure 5.14 Adjust starting item and the duration	88
Figure 5.15 Interaction by selecting different starting and ending state	88
Figure 5.16 Interaction by selecting a branch	89
Figure 5.17 The screen view of the player.....	90

1. Introduction

The media used for recording information are no longer bound to their carrier technology, such as paper, films or audiotapes, but become unified in a digital environment. This allows the different media to become part of a larger whole. Meanwhile, information conveying is no longer supported by means of only traditional media types, such as text and images, but also by media types, such as video and audio. The usage of multiple media types makes information presentations more attractive and improves the value of the information presented [Susanne]. Some media types such as text and image are time-independent, whereas the values of other media types such as video or audio change over time, and therefore these media types are time-dependant. This makes the management and delivery of multimedia more complicated.

However, it is very difficult for traditional database systems to allow users compose multimedia presentations in a heterogeneous multimedia environment [Chens2000a]. Therefore, to synchronize various multimedia data from distributed sources for multimedia presentations derives more and more attentions in this fast-growing information age. Among many newly developed applications, computer systems have been used to present information in increasingly extensive and appealing forms. An important example is the creation of a flexible computer-aided environment for multimedia presentations.

Looking at the different types of multimedia presentation applications, such as Video-On-Demand(VOD), computer games, and video conferences, etc., there are different characteristics with regard to the challenging problems. For example, VOD

applications show low complexity with respect to presentation functionality as only videos are retrieved from the server and continuous data are delivered uni-directionally to the clients. Computer-based training, learning programs, and electronic encyclopedias are currently able to include time-independent and time-dependant media elements, synchronization among elements, and referencing among presentations, but require complex content modeling. Therefore, it becomes major challenge for a multimedia database system to provide a data model, which isolates user applications from the details of the management and structure of the multimedia database system [Chens1997]. A novel model called Multimedia Augmented Transition Network (MATN) [Chens2000a] was thus created on this demand. Its goal is to model multimedia presentations, the temporal and/or spatial relations of different media streams, and multimedia browsing.

However, a multimedia environment should not only display media streams to users but also allow two-way communication between users and the multimedia system [Chens2000b]. The resulting multimedia database system then consists of database clients to query, browse and present multimedia data with user interactions, and a database server for the efficient management of and access to multimedia data.

The goal of this thesis is to design and implement the client end of a distributed multimedia database system with a set of user-friendly, easy-to-use and easy-to-extend Graphical User Interface (GUI). It proposes to retrieve multimedia information by querying and browsing for multimedia presentation authoring, process the composition of MATN to author multimedia presentation, and visualize the created MATN model at runtime.

1.1 Characteristics of multimedia data

A media item contains the data that is presented to the reader and as such is the basis of a presentation. We define the media item as an amount of data that can be retrieved as one object from a store of data objects, which is not necessarily a small amount. Media items can be of different media types. For the purpose of combining different media types within a single presentation, we define the characteristics of the four basic media types: text, image, audio and video.

- Text is an ordered linear sequence of two-dimensional characters, for example, English language text, Chinese characters.
- An image is static two-dimensional, visual representation, for example, a real world image such as a drawing or a photo, or symbolic representation such as a graph.
- Audio is continuous audible medium, for example, a piece of music, speech or sound-effects.
- Video is a continuous sequence of moving images, for example, continuous real-world images, animation or any sequence of still images that is intended to be perceived as a unity.

A number of media types, such as text and vector graphics, can themselves be structured. Text is a special case in that the internal structure of the media item can be expressed using the components of it, for example, HTML [Ragg97] or Postscript [Adobe90].

A media item may consist of a single medium, such as those just described, or a composite medium such as interleaved video and audio.

1.1.1 Temporal and spatial dimensionality

There are categories of media – continuous and non-continuous, or time dependent and time-independent. Continuous media have a temporal dimension intrinsic to the media item itself, such as audio and video. Non-continuous media have no intrinsic temporal dimension, such as text and image.

Figure 1.1 illustrates the four basic types of media items in a three-dimensional space. Their spatio-temporal dimensions can be classified as below [Pun98]:

- Text and image require two spatial dimensions for display, Figure 1.1(a). Since they are non-continuous data, there is no measurement in a temporal domain.
- Video requires two spatial dimensions to display the spatial data plus third dimension to display temporal domain, Figure 1.1(b). Video can be regarded as a sequence of images, where each image is displayed at a particular time.
- Audio is a continuous medium and has no spatial dimensions, Figure 1.1(c), thus it only requires time dimension to display its temporal domain.

When a media item is incorporated into a multimedia presentation, it is displayed on the screen for some duration or played through an audio device. Thus, it is necessary to obtain knowledge of its temporal and spatial dimensions in order to incorporate a media item within a multimedia presentation. [ISO97].

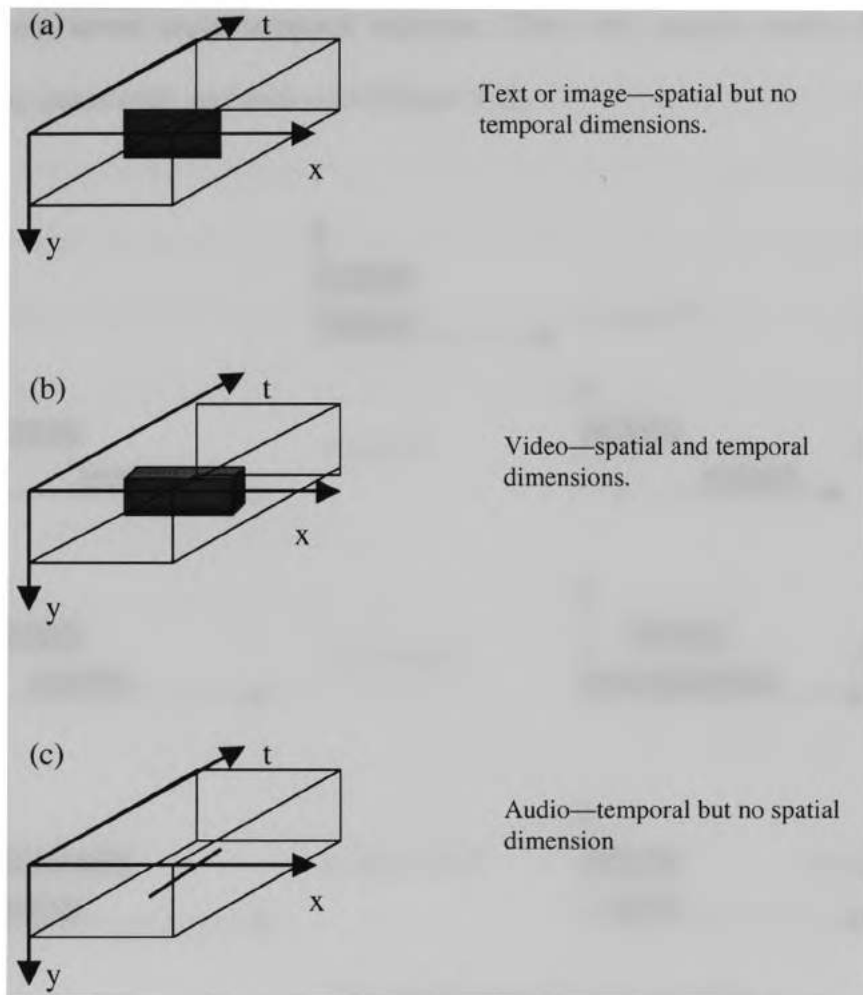


Figure 1.1 Media items in the three-dimensional space [HaBu97]

1.1.2 Temporal and spatial issues

Temporal layout is the determining characteristic of multimedia, defining when media items are displayed within the presentation. Temporal relations among media items can be defined based on the time interval. The famous Allen-time-relations [Allen83]

represent seven basic temporal relations. They are: *equals*, *meets*, *before*, *overlaps*, *during*, *starts with*, and *ends with* (Figure 1.2).

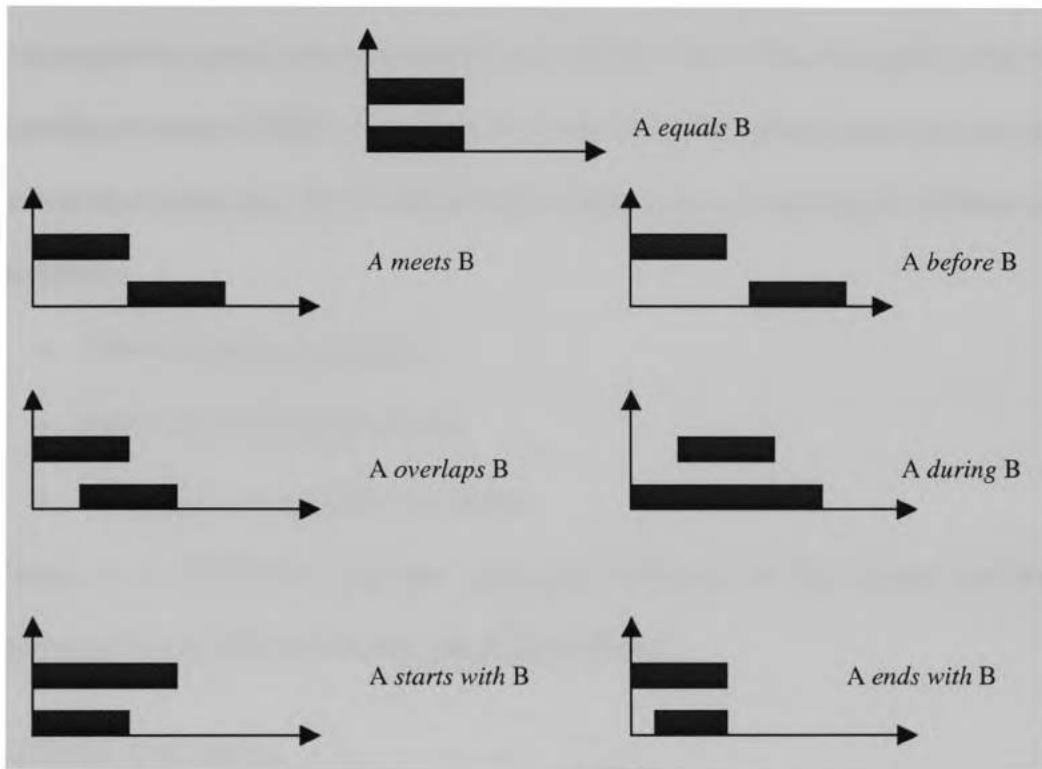


Figure 1.2 The Allen time relations

Allen's relations make the assumption that the durations of the media items are known beforehand. If this is not the case, then a conditional action can be specified. For example, two media items are playing and when the first of them stops playing, which is unknown beforehand, the other one also stops [Erf193]. Bordegoni [Bord92] gives a further categorization of conditions as being deterministic or non-deterministic, and simple or complex conditions.

Spatial layout is another important characteristic of multimedia, defining where media items are displayed within the presentation. Spatial relations can be coded using various knowledge-based techniques [Chens2000a]. Rectangular coordinates can be used to represent the spatial relation between two objects. One of the examples is the minimal bounding rectangle (MBR) concept in an R-tree [Gutt94], where each semantic object is covered by a rectangle. [ChYA88] identified three types of topological relations between the MBRs:

- Nonoverlapping rectangles;
- Partly overlapping rectangles;
- Completely overlapping rectangles.

Chang et al. [ChYA88] proposed orthogonal relations for the second and the third alternatives to find the relation objects [Chens2000a].

1.2 Scope of the thesis

Multimedia database management system (MDBMS) is built to organize, store, manage, and retrieve multimedia data in multimedia database systems. In order to embrace the heterogeneous nature of multimedia data and their characteristics, an efficient multimedia database management system should have the capability to accomplish the following tasks [Chens2001b]:

- Handling time-dependency and synchronized presentation of multimedia data;
- Querying data including both media data and textual data represented in different formats;

- Retrieving the query result and developing a presentation of that result in terms of audio-visual media; and
- Delivering this presentation supported by various Quality-of-Service.

This makes MDBMS differ from any traditional database management system (DBMS), in which only textual and numerical data is stored and managed, and synchronization among media is not a crucial issue.

1.2.1 Multimedia data querying and browsing

To mechanize multimedia data querying and browsing is one of the important tasks in a multimedia database management system. Traditional database systems can rely on textual and numerical indexing to query and retrieve data, but because of the complex nature of multimedia data, multimedia database systems require not only traditional query techniques but also other special functions to realize information retrieval. Due to multiple matching may occur, a flexible browsing interface is necessary for a more accurate query result.

Content-based retrieval (CBR) is a challenging method to achieve data in MDBMS. Many content-based image and video search systems have been developed for various applications. Enormous progress has been made in developing powerful tools that allow users to specify image queries by giving examples, drawing sketches, selecting visual features (e.g. color, histogram, and texture), arranging spatial structures of features, and combination of keywords and the approaches above. However, in order to better carry out CBR, there still a great deal of work to be done in multimedia database

indexing and searching. The connection between querying client and database server is one of the central issues to be improved and developed.

This proposed system is designed to support CBR and query by image. On the client side, a query-by-image module carries out image querying, image processing, and image displaying. A query interface is provided to allow users to interact with the system. On the server side, a novel unsupervised segmentation method, called simultaneous partition and class parameter estimation (SPCPE) algorithm, is implemented to carry out image segmentation and image object tracking to support CBR. The communication between the server and the client is based on UDP protocol due to real-time constraint from time-dependant type data.

Retrieving information by queries reduces the number of false hits, i.e. precision is improved. However, queries are limited to individual parts of a collection or filtering. For this reason, browsing is a good way to access data in a wide range if users don't have a specific idea of what they are looking for.

Browsing under structured schema organize multimedia data to be associated with the content itself so as to allow efficient searching for multimedia material of user's interest. Several pre-defined paths are provided to access information that is organized into units of storage. These environments are user-friendly, provide nice interfaces and require no prior particular system expertise from the user.

1.2.2 Multimedia presentation modeling

An appropriate semantic data model is an important issue in multimedia database management system. Along with the rich requirements for great significance and benefits

in both practical and academic aspects, the major challenge of such semantic data model is to efficiently synchronize the various data types for high performance multimedia presentations. As it mentions in [Chens2000a], the models must be devised to support the specification of temporal constraints for multimedia data and the satisfaction of these constraints must be at runtime. It also should be a good programming data structure for implementation to control multimedia playback. Meanwhile, in order to meet complex characteristics of multimedia data, this kind of model should allow users to specify the temporal and spatial requirements at the time of modeling.

There are many semantic models have been developed to model the temporal and/or spatial relations among multimedia objects. These semantic model can be identified several major categories by the underlying paradigm.

- Timeline-based models

A timeline-based model is the most common temporal model. In the traditional timeline model, all media streams are aligned on a single time axis [Blakowskis1991]. An enhanced timeline model expands the traditional timeline model to include temporal inequalities between media streams [Hirzallas1995].

- Flowchart-based model

A flowchart-based model gives the author a visual representation of the commands describing a presentation. Authoring with a flowchart model is similar to programming the presentation in a procedural way, but with an interface improved by graphic icons for visualizing the actions that take place, thus it is also called graphic model.

- Structure-based model

A structure-based model supports the explicit representation and manipulation of the structure of a presentation. The structure groups media items included in the presentation into “sub-presentations” which can be manipulated as one entity, and thus can in turn be grouped.

- Language-based models

A language-based model [Ates’s1996] provides the author with a programming language, where positions and timings of individual media items, and other events, can be specified. Authoring the presentation is programming. The major advantage of this approach is that it makes implementation straightforward.

- Augmented Transition Network (ATN) models

An ATN [Woods1970] model consists of a set of nodes and labeled arcs. Transitions are adopted to capture the temporal relationships. It was originally developed by Woods for natural language understanding systems and question answering systems. In [Chens1997a], the authors proposed to use ATN as a semantic model to model multimedia presentations, multimedia database searches, the temporal and/or spatial relations of different media streams and semantic objects, and multimedia browsing.

In our system, an extended model, called Multimedia Augmented Transition Network (MATN) is used for better satisfaction of modeling multimedia presentations and two-way communication. MATN is a semantic model based on Wood’s ATN [Chens1997a] [Chens1997b]. It basically follows all the principle rules of ATN. Under extensive studies of modeling techniques, MATN turns out to be the more appropriate model to serve as the basis for the design of multimedia presentation authoring. It fulfills

our system's requirement by many ways: For example, it can be implemented with foreseeable effort, provide a structure that allows editing and can be easily extended with additional elements that specify interaction and synchronization. Hence, it has not only practical need but also academic value to implement a flexible interface to realize MATN's multimedia presentation modeling features.

1.2.3 Visualization of the modeling

In order to fully visualize multimedia data from documented model, the logical solution is to separate all media types into its own media stream and let the client application synchronize them into a single cohesive presentation. To achieve this goal, [Lo2002] proposes two approaches, which include Java Media Framework (JMF) [JMF] and Synchronized Multimedia Integration Language (SMIL) [W3C]. JMF is a package of application programming interface (API), designed to incorporate audio, video, and other time-based media into Java applications and applets; SMIL is a markup language (like HTML) and is designed to be easy to learn and deploy on Web sites. SMIL was created specifically to solve the problems of coordinating the display of a variety of media (multimedia) on Web sites. By using a single time line for all of the media on a page, their display can be properly time-coordinated and synchronized.

1.3 Structure of the thesis

Chapter 2 reviews related work for information retrieval and different semantic models used in different multimedia presentation authoring systems. Chapter 3 presents the overall system architecture and the detail structure in the client side. Chapter 4 introduces information retrieval by querying and browsing and the interface implemented. Chapter 5

discusses multimedia presentation authoring using MATN model and the interface implemented. Chapter 6 concludes the thesis and suggests future work.

2. Literature Review

Information management systems are undergoing a revolution in this information era. Information was captured initially as images before developing to a symbolic system of text. Although humankind has recorded information for tens of thousands of years [Clottes], time-based media became common-place only one hundred years ago through the development of technologies such as film and gramophone records. Around fifty years ago, the introduction of computer technology had an impact as a new storage medium, but had little consequence for the media types used in digital encoding of documents. Instead, the computer provided a convenient means for creating and storing familiar media. Initial support was for textual documents, followed by image support. Computing power is now sufficient that support can also be provided for time-based media data, such as film and audio.

While the nature of media data is getting complex, the relationship among heterogeneous multimedia data becomes more complicated. One of the inherent characteristics of multimedia data is the heavy time-dependence so that they are usually related by temporal relations, which have to be maintained during their play-out [Chens2000a]. Traditional database systems lack the ability to manage heterogeneous multimedia properly. For this reason, problems related to querying, browsing and presenting multimedia information has been largely investigated in the literature.

In this chapter, we review some related issues in multimedia database systems research field. We briefly describe some recently developed work for multimedia

querying and browsing in section 2.1, then discuss different semantic models used to author multimedia presentation in section 2.2. A conclusion is given in section 2.3.

2.1 Multimedia Querying and Browsing

As multimedia data become more and more diversified, the technologies for information retrieval is growing, and does not suffer any more technical constraints that have characterized the early stage of the World Wide Web. Browsing and querying are two interactive and exploratory processes for accessing large amounts of data, from the point of information retrieval. They have drawn a number of concepts from fields of databases and information retrieval, but have added their own models, requirements and techniques. Meanwhile, multimedia information adds another dimension to the problem, when information is globally conveyed by different media which are archived and delivered separately, and must be coordinated and synchronized [Chia97].

[MTW95] describes an integrated query and navigation model built upon techniques from declarative query languages and navigational paradigms. The system implemented provides facilities to help non-expert user in query writing activity. Navigation hierarchies are used to present to the user summary information instead of a simple listing of query results. However, the authors never consider timing relations between objects, while they manage to hide querying of heterogeneous data on distributed repositories with a unified user interface.

Delaunay^{MM} [CL97] is a framework for querying and presenting multimedia data stored in distributed data repositories. The user specifies a multimedia presentation spatial layout by arranging graphical icons. Then, each icon is assigned to a query, thus

combining data selection with presentation. Delaunay^{MM} uses ad hoc querying capabilities to search each type of media item in distributed database and on the Web. Also in this proposed paper, the authors did not address any solution for the specification of temporal synchronization among the objects.

In [HiRu96] [HiRu97], the authors present temporal visual query language (TVQL). It is a multimedia visual query language for temporal analysis of video data. They consider a video of a teacher's lesson in classroom and annotate the video to identify interesting events, such as a student question or the teacher talk. TVQL enables user to browse for temporal relationships between two objects subsets. For example the user can search for which student speaks frequently after a teacher talk. The authors do not approach complex presentations with heterogeneous synchronized objects, but only a single video stream at a time.

In [Chia97], a model presents capabilities of fully integrate browsing and querying of hypermedia data. The model gives particular emphasis to structured information and combines two components: the hypermedia component and the information retrieval component. The hypermedia component contains information about both structure and content of a document. This integrated system allows users to compose queries that contain both kinds of information in their formulation. The information retrieval component contains information about the model and the resolution of the queries. Problems related to indexing structured information are also discussed and elements of a strategy for indexing hierarchical structures are presented. Although [Chia97] deals with composite documents, this paper does not consider time relationships among atomic objects of a structured document.

2.2 Modeling Multimedia Presentation

The ability to record information brings with it the ability to query and browse information. Since querying and browsing have their instinct limits to fully capture the various characteristics of multimedia data and their relations whenever synchronization is needed between multimedia data, the ability to present multimedia draws more and more demand on the field of multimedia database system.

However, the construction of a coherent multimedia presentation composed from its constituent parts is a non-trivial task. While the nature of media data is getting complex, the relationship among heterogeneous multimedia data becomes more complicated because of their heavy time-dependence characteristic. In order to synchronize the various data types for sophisticated multimedia presentations, the multimedia data system designers introduce some abstract semantic models to serve as bases for this task. According to the underlying paradigms, the existing models can be classified into some different categories. These various models provide distinct approaches for multimedia authoring systems.

2.2.1 Timeline-based Model

A timeline-based model shows the constituent media items placed along a time axis [Blakowskis1991], possible on different tracks, as shown in Figure 2.1. It gives an overview of which objects are playing when during the presentation. Timeline based authoring systems allow the specification of the beginning and end times of display of a media item in relation to a time axis. Manipulation is of individual objects, so that if the start time or duration of a media item is changed then this change is made independently

of any other objects placed on the timeline. The destinations of choice points are given in terms of a new position on the timeline.

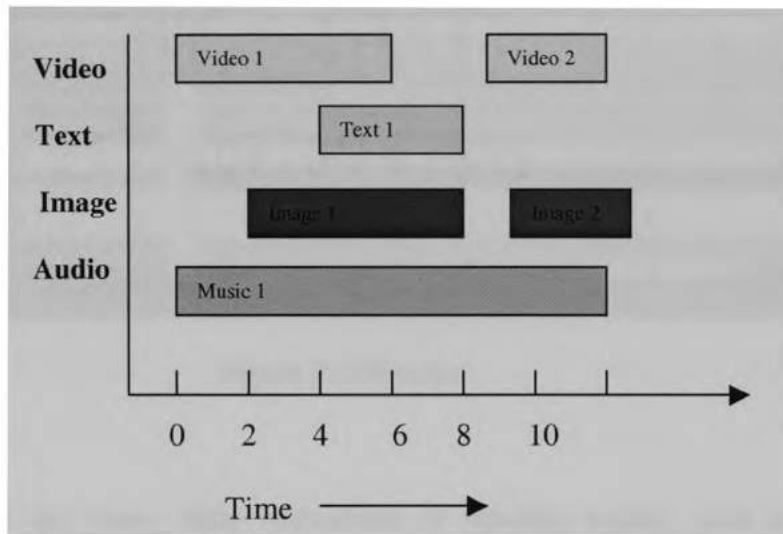


Figure 2.1 A timeline-based model

One example of timeline-based authoring systems is Director [Macr97] (Figure 2.2), which is a commercial system designed for creating animation-based presentations. Graphics, text, audio and video media items can be placed on a timeline. The timeline is divided into discrete time intervals, called frames. The timeline also has a number of associated tracks, where any media item can be placed on any track. A media item has a position in each frame, and the author can describe a path for the media item to follow through a series of frames. Sections of the timeline can be cut, copied and pasted.

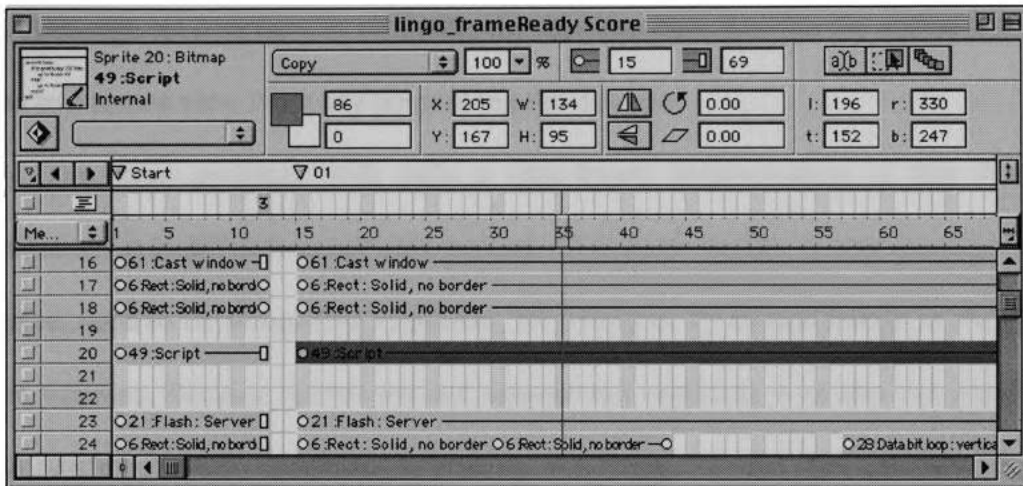


Figure 2.2 Director

There are many other utilizations of timeline model, such as Dreamweaver [DMW], Macromedia [Macr97], Flash [Flash], etc. Timeline-based authoring centers on presentation over time. It uses a time-axis as the main method of organizing the temporal positioning of media items in the presentation. It is visualized as a line with marked-off time intervals. The advantage of this approach is that the start times and durations of the media items in the presentation are displayed explicitly, and in principle can also be manipulated directly. The timeline can also be used to show the values of properties of the media items that vary over time. A further advantage of the timeline is that synchronization constraints. These constraints can be between media items or between a media item and the timeline.

Spatial layout is specified by assigning a position on the screen to the media item. This can be done by positioning the item where it should appear or by specifying the x and y positions over time. Neither method is specific to the timeline model. It is difficult to get an overview of the position of the object compared with other objects and over

time. Although no overview is available, the timeline does provide the author with easy access to a screen view from any point along the timeline.

The main problem with only a time-based representation is that for long presentations it is difficult to navigate around. Also scene breaks, or any overview of the narrative, need to be recognized implicitly, for example, by abrupt change in the objects on the timeline. Because scenes are not represented explicitly it is also not possible to create synchronization constraints in relation to a scene. Control flow can be added to a presentation as an object on the timeline but its effect cannot be visualized using the timeline.

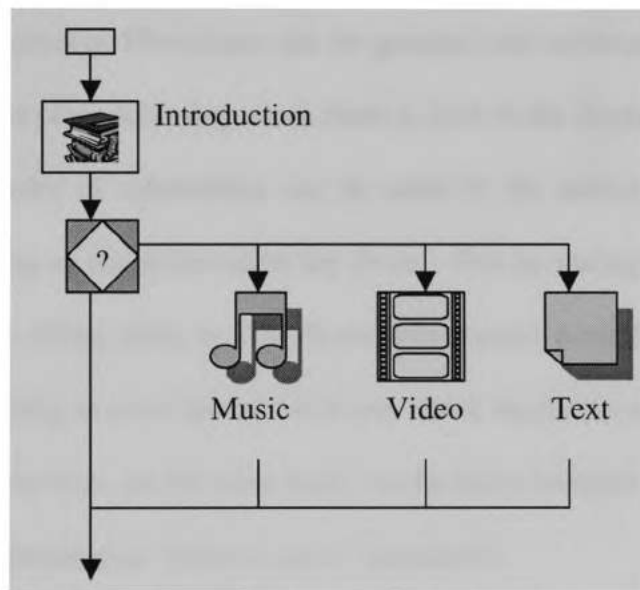


Figure 2.3 Flowchart-based model

2.2.2 Flowchart-based model

A flowchart-based model gives the author a visual representation of the commands describing a presentation. Authoring with a flowchart model is similar to programming the presentation in a procedural way, but with an interface improved by graphic icons for

visualizing the actions that take place, thus it is also called graphic model. Figure 2.3 The narrative of the presentation can be reflected in the routines and subroutines used. The order of displaying or removing objects and other events is shown, but time is not represented explicitly. The destinations of choice points are given in terms of jumping to a new procedure.

Authorware [Bufo94] [BuHe93] [Mac97] (Figure 2.4), is a flowchart-based commercial authoring system for creating interacting multimedia presentations for computer based training and kiosk applications. To create a presentation, icons representing actions are selected and incorporated into a flowchart defining the sequence of events in the presentation. Flowcharts can be grouped into subroutines and nested to arbitrary levels. This is often necessary, since there is limit to the display area for any one flowchart. The hierarchy of subroutines can be used by the author as an outline, or storyboard, for working on the presentation top down – first by stating the sections in the presentation and then filling them in. The flowcharts remain procedural however, and there is no way of getting an overview of when and which media items will be played on the screen when. Interactions, on the other hand, can be fairly complex and go far beyond links, which are implemented as “jump to there” commands.

In a flowchart-based model, control is the emphasized view: events are executed in turn, determined by the surrounding control structure. The advantage of the flowchart paradigm is that it incorporates more powerful interaction commands. For example, standard multiple-choice question formats are often provided. This supports the creation of links to a different section from each answer.

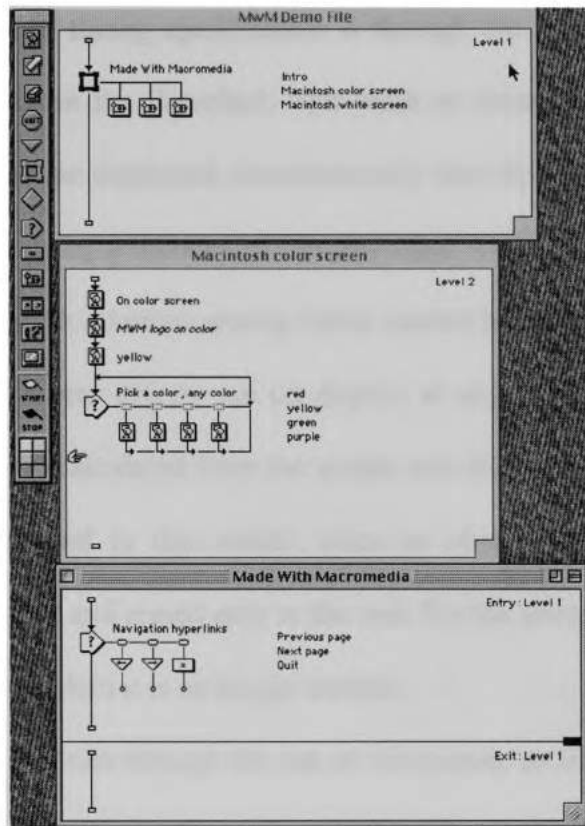


Figure 2.4 A view of Authorware

Some form of abstraction is also provided in flowchart-based model, but this is a grouping of commands in the form of nested flowcharts instead of relations among groups of events. This allows the narrative structure of the presentation to be reflected by the different levels of flowcharts. Although this is a useful view of the presentation's structure, it is difficult to find which items are displayed on the screen in the middle of a flowchart. For example, background images may have been displayed before the flowchart was executed. This means that a sub-scene cannot be played independently – since the state of the presentation is known only by playing the presentation.

The only form of timing specification is through the use of “display item” and “erase item” commands in the flowchart. This leads to three disadvantages. First, if a number of items are to be displayed simultaneously then this cannot be specified, but only approximated by using a number of “display item” commands one after the other. Secondly, synchronization relations among items cannot be specified. Thirdly, it is not clear from the script, where objects are on display at any particular time. This could, however, in principle be calculated from the scripts and displayed in a different view. A timing overview is needed in this model, since an object may be displayed at the beginning of a long script and erased only at the end. For the same reason, an author may forget to erase an object when it is no longer needed.

Just as timing is given through the use of commands, so is the spatial information for an object. Where the other objects are placed on the screen is only to be found by looking through the flowchart. Any overview of the objects in time, or their relations with respect to other objects is thus not available. Another example system, called Eventor [ENKY94], introduced the spatial synchronizer to help with this problem. Links are specified via commands associated with hotspots, which define which playing objects should be erased and which new objects should be displayed.

2.2.3 Structure-based model

Structure-based authoring systems support the explicit representation and manipulation of the structure of a presentation. Figure 2.5. The structure groups media items included in the presentation into “sub-presentations” which can be manipulated as one entity, and thus can in turn be grouped. Although in principle the same object can belong to one or

more groups, in current authoring systems this is not the case. The destinations of choice points in a presentation, that is where the reader is able to select to go to other parts of the presentation, are given in terms of the structure. The structuring may group the media items indirectly, where, for example, higher-level concepts are associated with each other and each concept is associated with one or more (groups of) media items.

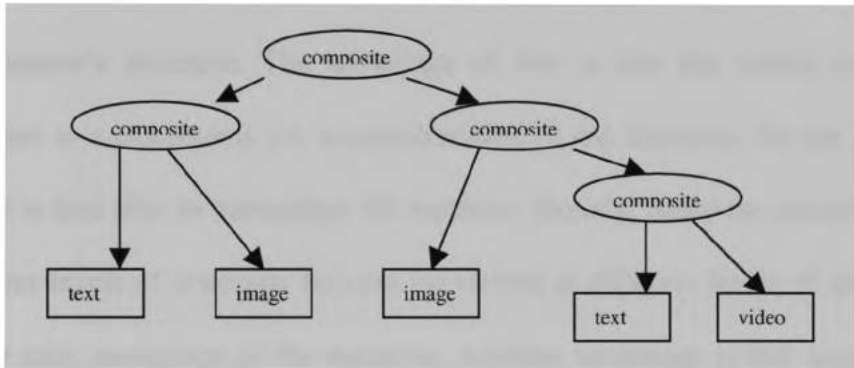


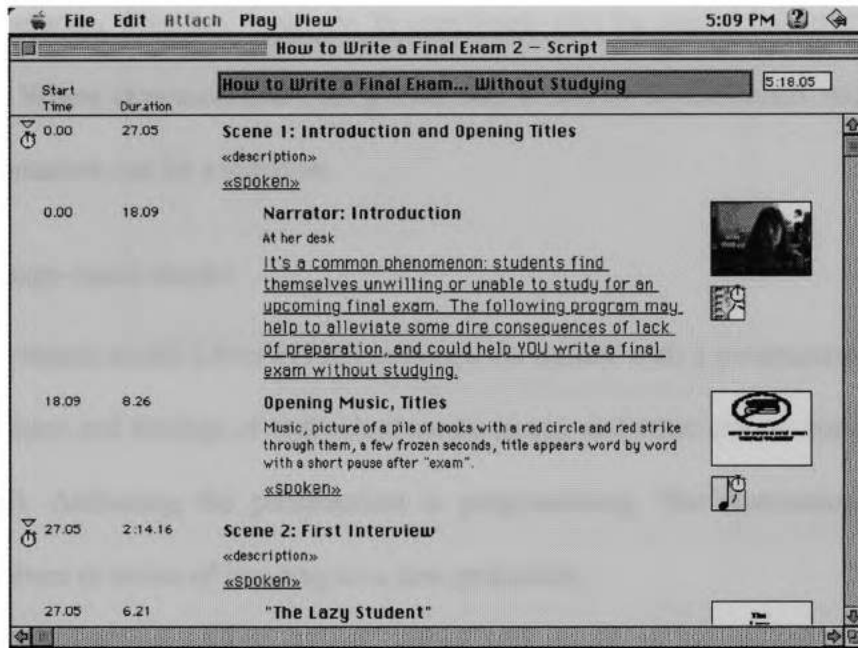
Figure 2.5 Structure-based model

MAD (Movie Authoring and Design) [BRFS96] is a structure-based authoring system that facilitates the process of creating dynamic visual presentations. Figure 2.6. MAD aids this process by simultaneously allowing easy structure creation or modification of motion pictures and visualization of the result of those modifications. It decomposes a multimedia presentation as a nested hierarchy. This hierarchy is able to represent “acts”, “scenes” and “shots”, although these divisions are not imposed on the author. In a manner similar to a text outliner, the different levels of the hierarchy can be hidden or revealed and the position of items can be moved within the hierarchy. The start time of each item is calculated from the start times and durations of preceding items and sub-items in the hierarchy. The duration of an item can be calculated on the basis of the

media item for video and audio, or can be specified by the author. The author also has control over playback of sections of the presentation by playing complete items or skipping forward to following items. MAD lacks any control of synchronization among items so that a single item with its associated parts can be played, but other items cannot start before it has finished. It is unclear to what extent there is control of spatial layout.

Structure-based systems allow the explicit specification and manipulation of a presentation's structure. The advantage of this is that the author is able to use the structure as a storyboard, i.e. a representation of the narrative, for the presentation. The author is thus able to manipulate the narrative directly. Since the presentation consists of different levels of structure, this can be viewed at different levels of detail, allowing the author easy navigation of the narrative. Another advantage is that since the structure is able to indicate an ordering, it can be used for deriving the timing for the presentation, as demonstrated in MAD. The timing can thus be visualized and edited, at least to some extent, in the structure-based view. In another structure-based authoring system, MET++ [Acke94], it has the structure displayed along a timeline.

Synchronization constraints can, at least in principle, be defined between media items, between a media item and a scene (or other structure), or between two structures. While the fact that a timing relation exists could be shown in a structure-based view, it requires a time-based view to show the actual influence of the constraints, of some complexity, can be defined, but since the view is not time-based the resultant timing is not visualized.



Hierarchical structure of items is indicated by indentation of the text. The text has 3 fields - title, screen directions, and narration or dialogue.

Figure 2.6 MAD (Movie Authoring and Design) script view

Spatial layout is defined by assigning a position on the screen to the media item. This can be done by positioning the item where it should appear or, as in MET++, by specifying the x and y positions over time. These two methods are not specific to the structure-based model, and in both cases it is difficult to get an overview of the position of the object relative to other objects or over time.

Links can be created among structures, allowing source and destination contexts to be specified along with the value from which the hotspot is derived.

The disadvantage of the structure-based model is that extra authoring effort has to be expended to create the initial structure.

A purely structural view of the presentation gives no understanding of the timing of the presentation. This can, however, be combined with the timing information, as done in MET++. Where structural and timing information cannot be combined, multiple views of the presentation can be a solution.

2.2.4 Language-based model

A language-based model [Ates's1996] provides the author with a programming language where positions and timings of individual media items, and other events, can be specified (Figure 2.7). Authoring the presentation is programming. The destinations of choice points are given in terms of jumping to a new procedure.

[Ogama90] designed a system, called Videobook, to incorporate a time-based media-composite data sequence with the hypertext node and link concept, allowing the construction of composite multimedia nodes. The system presents media items and hotspots according to a script specifying their presentation parameters – timing and layout (Figure 2.8). The script is visualized as three-dimensional display showing the layout of each object along a timeline. The system thus provides a low-level scripting language for the author to specify a presentation, which is then given a higher-level visualization along a timeline.

```
set win=main-win
set cursor =wait
clear win
put background "pastel.pic"
put text "heading1.txt" at 10,0
put picture"gable.pic" at 20,0
put text "contents.txt" at 20,10
set cursor = active
put video"sun.pic" at 30,0
```

Figure2.7 Language-based model

Some authors have taken the scripts form of specification even further [HeKo95]. Here, the authors view a presentation as a sequence of command streams, where each stream consists of an ordered collection of commands, each of which is assigned its own execution time. If a command stream starts to fall behind, then commands can be skipped to allow the stream to catch up. The command stream contains sufficient information to allow it to be played not only backwards, but also in both directions at a higher speed.

The major advantage of language-based models is that they can directly lead to an implementation [Chens2000a]. They have inherent characteristics in terms of flexibility and power of expression.

However, this is a tedious, low-level method for specifying a multimedia presentation without any other support. Language-based models lack tools for viewing the procedure calls in any structured way. This in turn leads to more likely program structure errors. Even if the narrative structure of the presentation has been reflected in the script structure, it remains difficult to manipulate at a higher level. Timing information for the presentation is embedded in the lines of code, or explicitly specified

in lines of code as in [HeKo95]. Spatial layout information is also given via the lines of code. Since the structure, timing and spatial layout is presented in the code. It is possible to derive a time-based visualization.

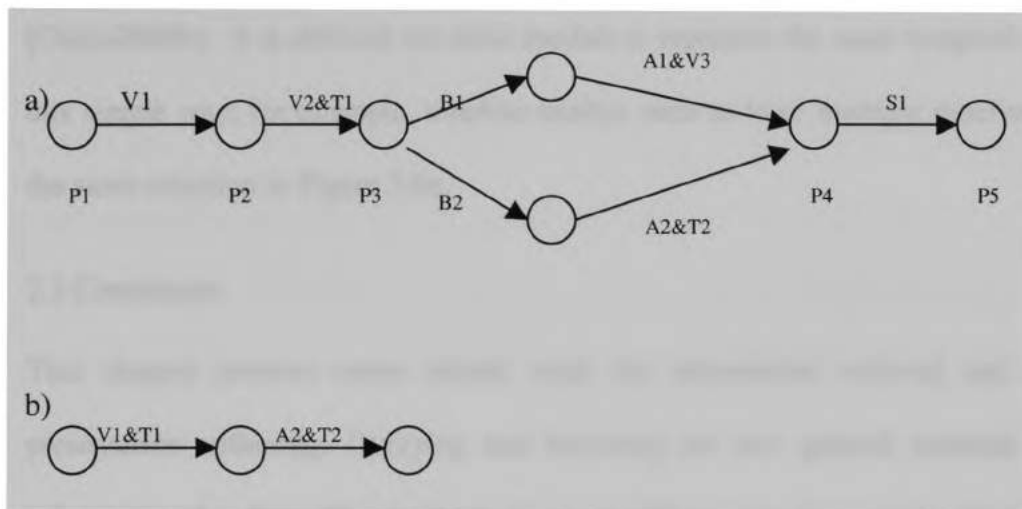
2.2.5 ATN-based model

The augmented transition network (ATN) [Woods70] [Woods73] has been developed for natural language understanding systems and question answering systems for both text and speech. Recently, the use of ATN as a semantic model [Chens2001a] [Chens1997] has been proposed to model multimedia presentation, multimedia database searching, the temporal and/or spatial relations of various media streams and semantic objects, and multimedia browsing.

An ATN represents the temporal relationship among multimedia data by a finite set of nodes and arcs. Nodes represent different states of transition, and arcs indicate the time flows of the presentation. Each arc comes with a label that defines the media items to be represented in the time interval. This time interval is specified by two states connected by the arc. In other word, the input of the label can cause a transition from the state at the tail of the arc to the state at its head [Chens2000b].

ATN-based models allow sub-networks and multiple arcs from one state. The advantages of these are that the author is able to use the multiple arcs to represent multiple selections for user interaction, and model detailed information of media streams in the sub-networks. Any changes in any one of the sub-networks will be automatically reflected in the presentation that includes these sub-networks. This feature allows

designers to exploit existing presentation sequences in their archives, making ATN a powerful tool for creating a new presentation.



- a) The main network with user selections: B1 and B2. S1 indicates there is a sub-network
- b) The detail information in sub-network S1.

Figure 2.8 ATN with user interaction and sub-network

Figure 2.8a illustrates the time flow, user interaction and an embedded sub-network in ATN. Assume P2 is the start state, arc label V2&T1 represents a video and a text are displayed concurrently. When V2 and T1 finish, the control reaches the next state p3. This is a state that allows user interaction. B1 and B2 indicate there are two selections in this state. If B1 or B2 is selected, the corresponding arc will lead to different state to display either A1&V3 or A2&T2. After displaying either state, there is a merge, i.e. either of them leads to state p4, where goes to a sub-network S1. Figure2.8b is the detail information of S1. Here, a single state actually contains three-state information. By this way, an existing presentation can be efficiently embedded in the current ATN.

As Figure 2.8 shows, an ATN model can author complex situation in a single framework. This is the most significant characteristic of an ATN. The advantage is that the other presentation structure is independent of the current presentation structure [Chens2000b]. It is difficult for other models to represent the same temporal structure in this simple way; for example, timeline models need to have multiple timelines to model the same situation in Figure 2.8a.

2.3 Conclusion

This chapter reviews some related work for information retrieval and multimedia presentation authoring. Querying and browsing are two general methods to retrieve information, but have their intrinsic limits to fully capture the various characteristics of multimedia data and their relations whenever synchronization is needed between multimedia data. Semantic models make a step further to well present multimedia presentation.

Each model we have described emphasizes a different aspect in multimedia presentation authoring environment. The timeline-based model emphasizes the temporal aspects; the structure-based model emphasizes the narrative structure of the presentation; the flowchart-based and language-based models emphasize the execution order of displaying and removing objects at the runtime; An ATN-based model has tried to provide more declarative nature to present the temporal relations among multimedia data, and allow more flexible user interactions by offer sub-networks and user selections.

The models themselves are not mutually exclusive, but reflect a difference in emphasis. It is possible that one approach cannot provide the ideal solution to an author's task and more often a combination is appropriate.

3. System Architecture

This system is a distributed multimedia database system, which is constructed upon two-tier client/server architecture. The client presents a set of easy to use Graphical User Interfaces (GUIs) to allow users to retrieve information by querying and browsing, and author the information presentation by composing MATN models. It is written in Java, and currently supported by Window platform. The server services as a multimedia database, implemented by C++, and supported by both Window and Unix platforms. The connection between the client and the server is through datagram sockets with User Datagram Protocol (UDP). Both sides set up their own communication channels and wrap the data information in packets to deliver. The overall view of the system architecture is given in Figure 3.1.

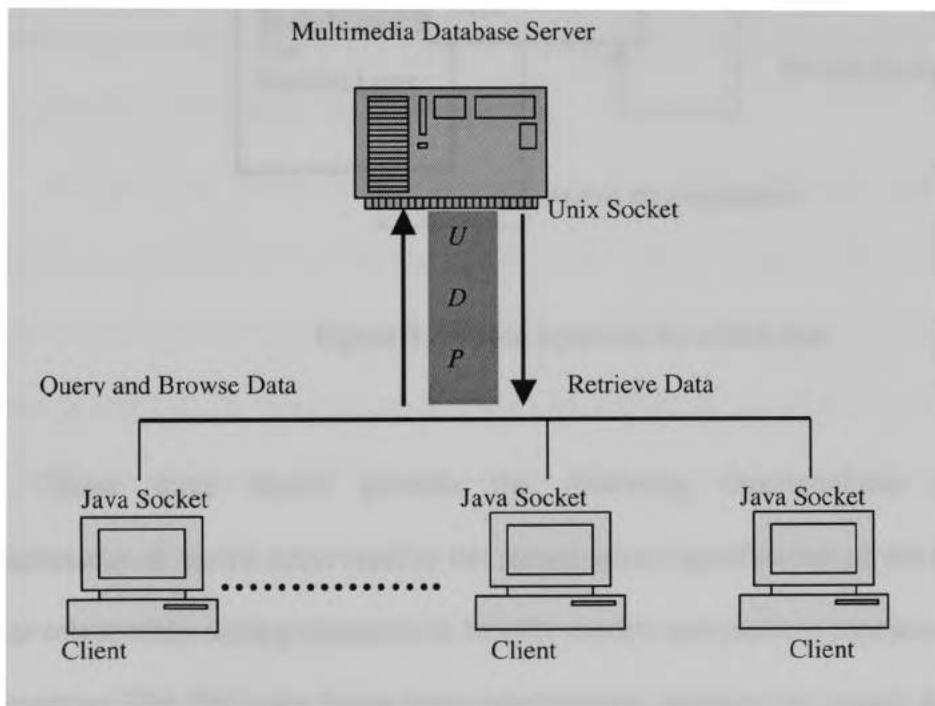


Figure 3.1 Overall view of the system architecture

3.1 The architecture of the client

The client end of the system is an integral presentation framework. This framework consists of three layers: the data layer, presentation composition layer, and the runtime layer. Figure 3.2 illustrates the three layers and their relations.

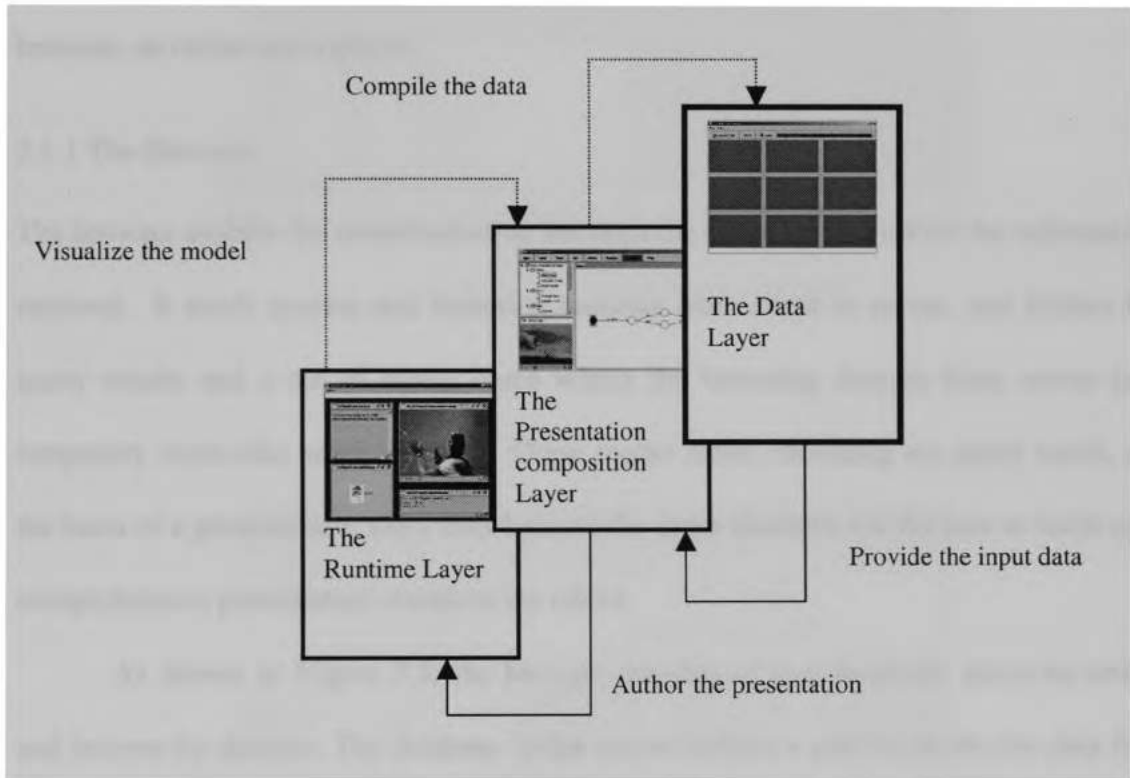


Figure 3.2 Three layers in the client side

These three layers provide the following functionalities respectively: characteristics of media items used in the presentation; specification of the temporal and spatial relationship among elements of MATN model; and runtime characteristics of the presentation. The first two layers have relationships between the media items and the model elements, and the second two layers have relationships between the elements and

their final presentation. The data layer retrieves information from the server and provides the retrieved information as input data to the presentation layer. The presentation composition layer provides a set of tools for user to author presentation based on MATN, which constructs the presentation structure. The runtime layer visualizes the documented presentation. Three interfaces are created to accomplish these three different tasks: a browser, an editor and a player.

3.1.1 The Browser

The browser enables the communication between the client and the server for information retrieval. It sends queries and browsing requests from client to server, and fetches the query results and a set of media items within the browsing domain from server to a temporary cache-like space in client. These media items, including the query result, are the basis of a presentation. They thus become the input elements for the user to build up a comprehensive presentation model in the editor.

As shown in Figure 3.3, the browser consists of two modules: query-by-image and browse-by-domain. The database in the server defines a path to divide the data files into several categories based on their characteristics of contents, for example: hurricane, texture, soccer, etc. In order to retrieve the data efficiently, a choice box is presented, allowing the user to select the category at the first step for browsing and querying. The browsing and querying modules are activated once the user selects a category. The browser is ready to browse by clicking *Browse Files* button. Meanwhile, at most nine images are displayed for query-by-image process and image-based browsing as well.



Figure 3.3 The browser interface

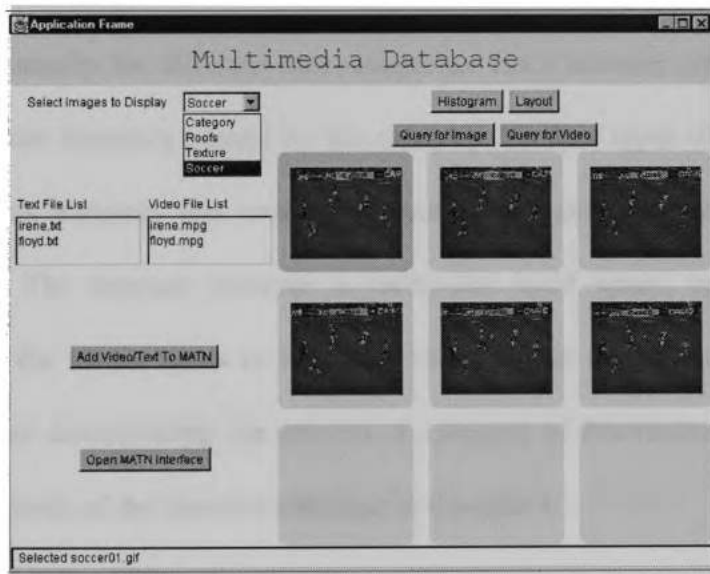


Figure 3.4 The former interface of the browser

The query module was first implemented in [Chenq2001]. Its original interface is shown in Figure 3.4. Since richer functionality is required in this system, we include

query module into the browser with some enhancements. Besides the different look and feel of the interface, the browser now supports more functions and better manipulation than the original one did.

The query module brings out at most nine images simultaneously. One image can be selected at a time as a query image. The background color of the image will be changed if it is selected. Because of the different characteristic of different type of multimedia data, to retrieve them may require different methods. This query module is thus borrowed to support a browsing function in the way that users browse the image file more specifically. Figure 3.5 shows that nine images have been browsed under the category of hurricane.

The browsing module is activated by clicking *Browse Files* button. Figure 3.6 shows a browser frame for the category of *Hurricane*. There can be multiple browsers working simultaneously for different data categories. Each browser contains a directory tree, which has root directory named by the category. As four types of multimedia data are concerned, this directory tree consists of four default sub-directories: video, audio, text and image. The browser reserves a cache-like local space, called “Temp”, to temporally store the media items in the client side for fast access purpose. Figure 3.7 shows the browser downloading the content of category of *Hurricane* from the server. We describe the detail of the browser interface in Chapter 4.



Figure 3.5 Browse-by-image

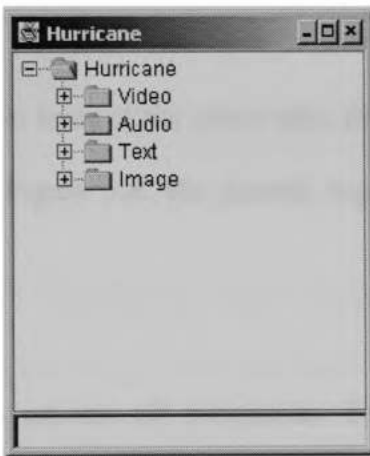


Figure 3.6 Browse data catalog Hurricane

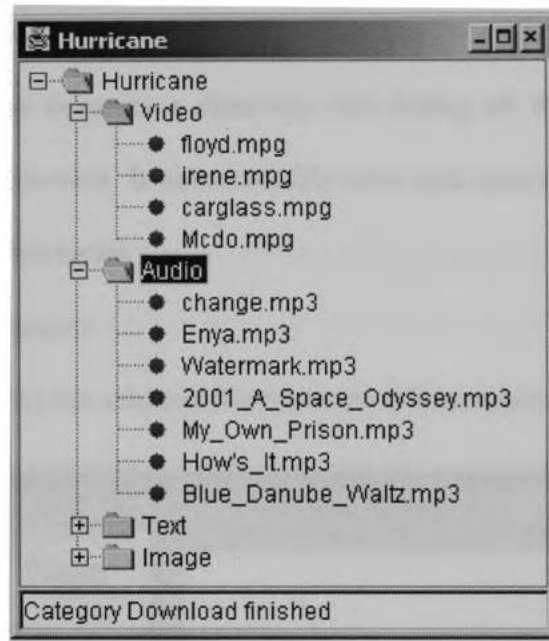


Figure 3.7 Download category of hurricane from server

3.1.2 The Editor

In the presentation composition layer of the client side, there is a user-friendly interface, called editor. As shown in Figure 3.8, the general organization of the editor screen consists of four parts:

- The menu bar

It contains buttons to activate all commands. Some buttons, such as *Branch*, *Preview* and *Play*, will turn red when the corresponding commands are selected.

- The editing window

It draws the graphic MATN model to author the multimedia presentation, easily manipulated by mouse click.

- The file window

The file window contains a directory tree listing all the files that users have selected from the browser. It automatically sorts and stores the files by their types under proper sub-directories.

- The preview window

It previews the selected file for users to better selection purpose.

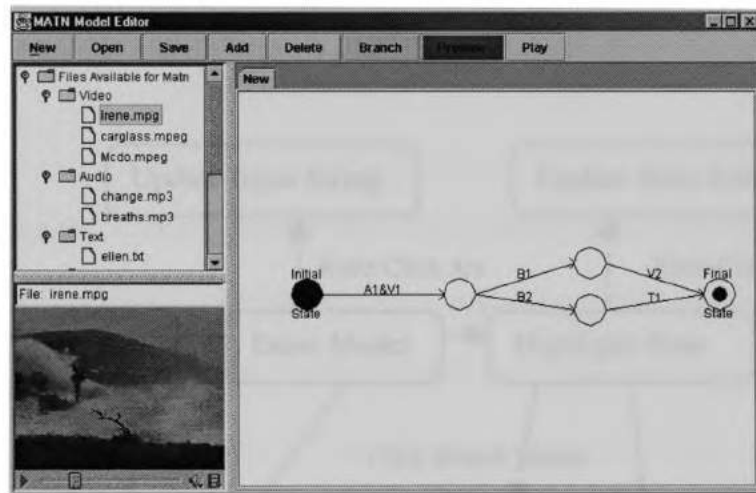


Figure 3.8 The screen view of the editor

The editor interface is brought out when users click *MATN* button from the browser interface. Users can always go back and forth between the two interfaces to select their desired input data for the MATN construction, and connect player interface by clicking *Play* button. Figure3.9 is the flow chart for the manipulation of the editor.

When the user selects a media item from the file window, there are two possible situations can be triggered. If the *Preview* button is pressed and turning red, then the preview window will display this media item for preview purpose. In Figure 3.8, the preview window is playing media item “irene.mpg”. Only one media file can be

previewed at a time. In other way, if *Preview* button is not pressed, the selected item is ready for composing the model. Users can right click mouse, then a choice dialogue pops up for selecting or deselecting. To select an item to compose a model, users click *Select* and press *Add* button in the menu bar, another dialog frame shows the file name of the item that has been selected. Multiple files can be selected at a time. Once users confirm the selection, the editor window draws a proper MATN model automatically. Figure 3.10 presents a three-state model with media items input on the arc.

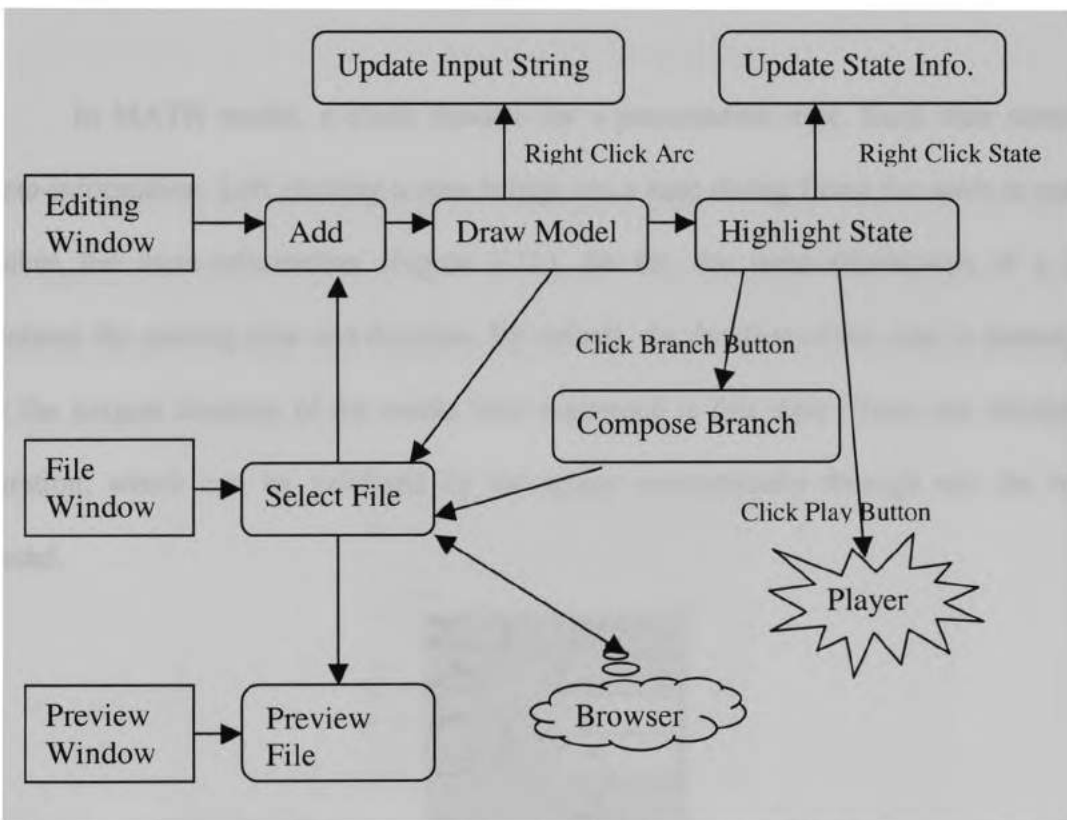


Figure 3.9 The manipulation states of the editor

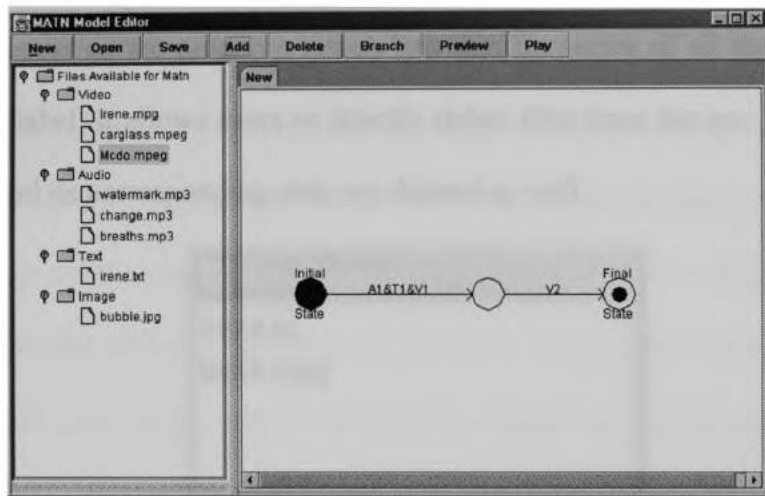


Figure 3.10 Edit a MATN model

In MATN model, a circle denotes for a presentation state. Each state stores its meta-information. Left clicking a state brings out a state dialog frame for users to read or update the meta-information (Figure 3.11). So far, the meta-information of a state contains the starting time and duration. By default, the duration of the state is determined by the longest duration of the media item registered in this state. Users can change the duration, which can be validated by the editor automatically through out the whole model.

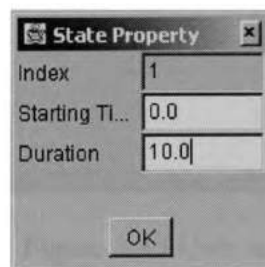


Figure 3.11 State property dialog

The input data, retrieved from the file window, is stored in the label above the arc. There is also a dialog frame linked to each arc (Figure 3.12). User can left-click a

selected arc to access an arc property, which lists the file names of all the media items composed in the label. It allows users to directly delete files from the arc. If all files are deleted, the arc and its corresponding state are deleted as well.

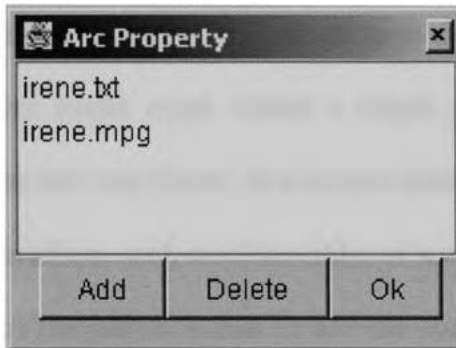


Figure 3.12 Arc property

Another significant element in the MATN model is user selections. It supports user interactions during the runtime of multimedia presentation. By composing multiple branches from one state, users are allowed to select any one of the branches to continue the presentation (Figure 3.13).

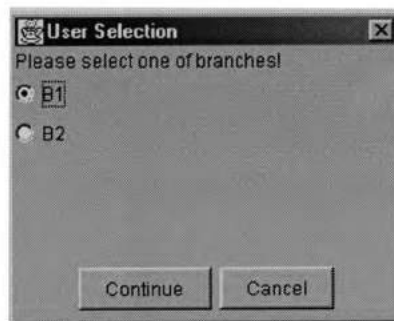


Figure 3.13 User selection

3.1.3 The player

The player interface (Figure 3.14) visualizes the created MATN model for presentation model. It is triggered by pressing *Play* button on the menu bar. Selecting a starting state and an ending state in the model, any part of the presentation can be displayed. For the purpose of combining different media types within a single presentation, the player comprises four different players into one frame. In a certain state of the presentation, the label may consist of a single medium, such as video only, or a composite medium, such as interleaved video and audio. The player is able to activate corresponding player units to satisfy different requirements in various situations.



Figure 3.14 The screen view of the player

3.2 The architecture of the server

The multimedia database server manages the data in the database and controls concurrent access of the multimedia clients. An important aspect of the server environment is how to design and build a high-performance multimedia database server for content-based retrieval of multimedia data.

In order to search for more accurate information by content-based retrieval, three tasks must be done. First is to segment the foreground from the background of an image to extract image features, second is to identify the objects in an image and capture the temporal and spatial relations of semantic objects within the video frames [Chens2000c], and the third task is to search query results based on the target image whose features have been extracted and stored beforehand.

The new approach adopted is an enhanced unsupervised segmentation method and the algorithm is a Simultaneous Partition and Class Parameter Estimation (SPCPE) algorithm [Sistas1997] [Chens2000d]. The algorithm is implemented to extract the features of image data and capture the temporal and spatial relations of semantic objects. But this algorithm is not able to identify the objects that overlap within video frames. There are four steps to fulfill image segmentation and object tracking:

- Start with an arbitrary partition and computing class parameters;
- Estimate a new partition by class parameters and data;
- Iteratively refine partition and class parameters until no change; and
- Obtain the bounding box of each semantic object.

To deliver the query results to the client, the server first reads the image into binary files, and then wraps them into packets to send to the client.

3.3 Communication path

Java socket and Unix socket are used in this system to provide a communication path between Windows client and Unix server with connectionless User Datagram Protocol (UDP) transport mechanism.

When the system is running, the Unix server creates a socket, binds it to a local address and a predefined port, and waits for the requests from clients. When a client sends the request to the port, the server creates a process to service the client. Then the new server process talks with the client, using UDP. Once a UDP socket has been created and bound to a local source port, it is now capable of being used for sending and receiving datagrams.

On the client side, Java encapsulates the concept of a UDP socket with the class *DatagramSocket*, and the concept of a datagram with the class *DatagramPacket*. A *DatagramPacket* consists of a fixed-length array of bytes together with an IP address and port. When one sends a *DatagramPacket*, its array of bytes is sent to the socket with the specified IP address and port (if it exists). When one receives a *DatagramPacket*, the data is copied into its array of bytes, and the IP address and port of the sender are copied into the IP address and port of the *DatagramPacket*.

3.4 Conclusion

This chapter gives an overview of the structure of the whole system. The system so far is a two-tier client/server application and subject to be further enhanced. The Java client consists of three layers to carry out information retrieval, multimedia presentation authoring based on MATN model, and presentation runtime using JMF. There are three

major graphical user interfaces interacting with each other: a browser, an editor, and a player. The Unix server implements SPCPE algorithm to realize content-based retrieval and object tracking. The communication between the client and the server goes through datagram sockets under UDP. Java socket and Unix socket are used in the client and the server side respectively.

4. Multimedia Querying and Browsing

Querying and browsing the multimedia database is an important research field, which has drawn a number of concepts from databases and information retrieval fields, but has added its own requirements and techniques. As multimedia is entering all fields of communication, information presentation is becoming more and more rich. It requires not only good quality of media data, but also effective information retrieval according to complex schema. Our system's data layer in the client side was created in this demand.

In this chapter, we present the requirements for information retrieval in section 4.1. Section 4.2 discusses the management of information retrieval, including how information can be retrieved as to provide the input data for further presentation authoring, and the interface design and implementation. Section 4.2 describes the communication between the server and the client with the emphasis on the client side. A conclusion is given in section 4.4.

4.1 Requirements for information retrieval

The result of a query on a multimedia database is a set of media items, which respond to the query criteria. Each media item can be a time-independent or non-continuous medium, like a text page or an image, but also can be a time-dependent or continuous medium, like an audio or video file. This is the first glance at the information retrieval. If retrieved items are also part of a composite multimedia document, such as a complex presentation, then it is not a good idea to display them alone. For example, a slice of a lesson without the accompanying audio track, which is recorded separately, is scarcely useful.

A well-organized multimedia presentation is usually composed of several media items that are to be coherently harmonized in order to be meaningful [AuCe]. Such multimedia data sources need to be effectively and efficiently searched for information of interest to users, or filtered to receive only information that satisfies user's preferences. For this purpose, some peculiarities are to be presented.

- Information is contained in media items of different types; i.e., querying can retrieve information from different media at the same time.

Multimedia databases encompass multiple representations of data in order to answer queries on different media types. Although this becomes a commonplace in today's multimedia database system, it is still not a small concern, and doesn't mean that all technical aspects are solved. For instance, the language (textual or pictorial) used to formulate the query belongs to a specific syntactic domain against which data representation of different types must be matched.

- The user is not interested in any single media item but in part of presentation, in which at least one of the media items satisfy the query; in other word, the true query result is the presentation, and the retrieved item is only a component of a part of the presentation.

This defines the static organization of a presentation, which relates media items together.

- A coherent and understandable segment can be browsed and returned to the user; i.e., the query result extent can be identified according to a context, which takes into account the presentation structure.

This relates to the temporal aspects of the presentation, which describe how the different segments evolve.

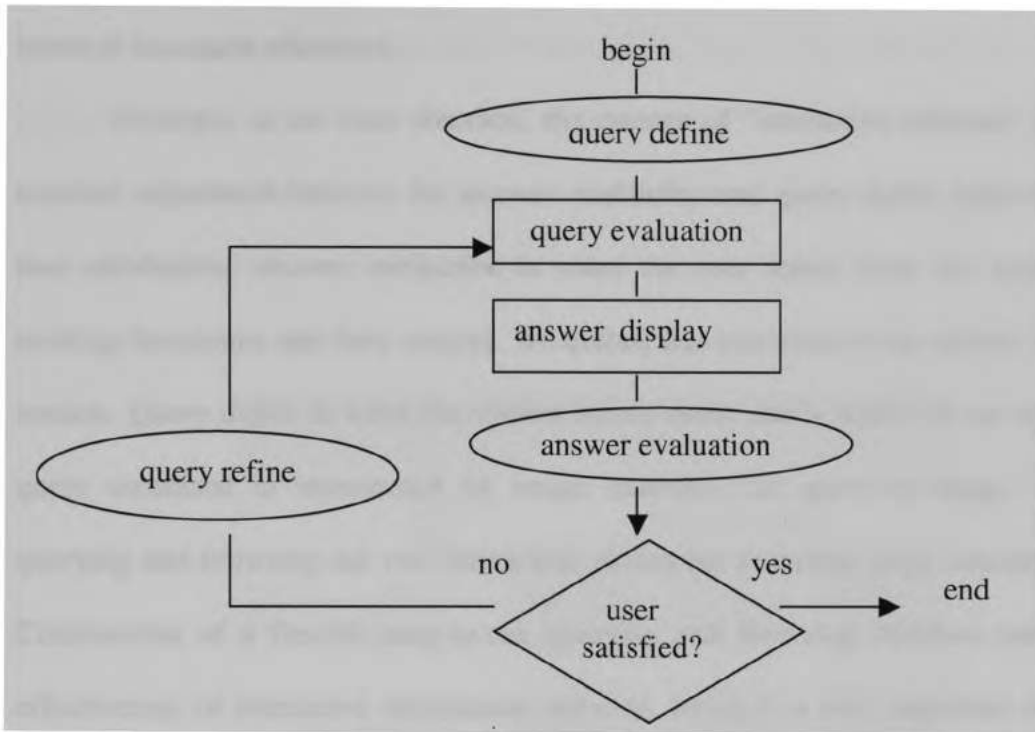


Figure 4.1 The iterative process of information retrieval [Cruz97]

4.2 Management of the information retrieval

Information retrieval is essentially an interactive task [AuCe]. It is almost impossible to solve an information retrieval problem in a single step of issuing a query and observing the response of the corresponding system. Most probably, an interactive process will be involved. Figure 4.1 is a simplified schema illustrates how the user and system's tasks are intertwined in the process of information retrieval, where tasks in oval and square boxes correspond respectively to user and system's tasks [Cruz97]. Depending on the user's

expertise, this often results in the long interaction process of a retrieval session delimited in Figure 4.1 by the *begin* and *end* symbols. The steps include defining a query command (an image example in our system), issuing it to the system, and evaluating its answer in terms of document relevance.

However, as we have observed, the essence of “interactive retrieval” lies in the constant adjustment between the *answer evaluation* and *query define* tasks to achieve user satisfaction. *Answer evaluation* is when the user learns from the system about existing documents and their content. We embed this operation in our answer evaluation session. *Query define* is when the system learns about user’s needs. In our system, the query command is represented by image example, i.e. query-by-image. Therefore, querying and browsing are two interaction modes for accessing large amounts of data. Construction of a flexible easy-to-use querying and browsing interface can improve effectiveness of interactive information retrieval, which is a very important step in our system to compose a sophisticated multimedia presentation.

Based on the requirements mentioned on section 4.1 and process illustrated by Figure 4.1, we design a novel browser interface for users to accomplish information retrieval in the client side. The server side, which maintains the multimedia database, is designed to support content-based retrieval and archive browsing.

The original version of browser interface was implemented in [Chenq2001]. Figure 4.2. It carried out image querying, image processing, and image displaying. This interface was created with intent to process image query by content-based retrieval (CBR). Although it was named as “Multimedia database”, its querying and browsing operations are limited in image data, which is only a small part of multimedia data. Since

our final goal is to provide a flexible environment to support multimedia presentation authoring, we adopt its query-by-image operation for the retrieval of image data, and fulfill its usage by allowing users submit the query result to be the input data of the presentation, which is the major task of our system's client end. Meanwhile, in order to retrieve multiple types of media data, such as video, audio and text, we improve the answer evaluation session by enabling archive browsing. Therefore the new version of the browser interface has richer functionality.

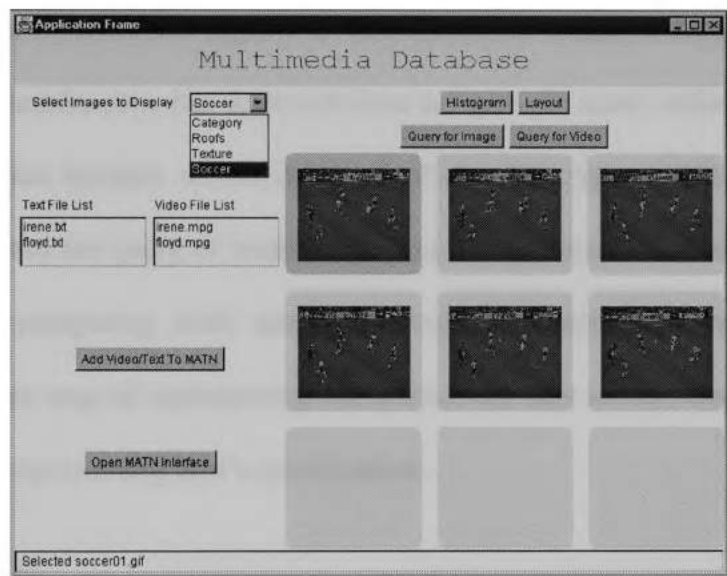


Figure 4.2 The first version of query module

4.2.1 Querying

Querying is a popular research issue in multimedia database system. Classical Information Retrieval systems (IRS) provide powerful and effective content-based retrieval (CBR) processes. These environments require some expertise from the users

who have to master the index language, which describes the content of the stored information, and the query languages of the systems. This case often corresponds to comparatively poor user interface. However, “Query-By-Example” (QBE) [Zloof81], which is one of many methods of CBR, simplifies this problem and is easily presented in a convenient user interface.

Just as what QBE literally stands for, QBE proposed a way to allow users to make a query by giving an example. Traditional database systems usually present a query by keywords, which are in the form of text or numerical value. In this way, the query result is retrieved by relational algebra or descriptions of simple comparison of attribute values. For multimedia data, this approach dose not gain the same achievement as it does for traditional data because of the diversity of the data types. QBE provide a more promising approach for query of multimedia data. [Yoshitaka 99] mentioned advantages of using QBE comparing with using keywords in query representations. Besides providing a better way of representing the queries for non-textual data, it provides an intuitive way of representing user’s specification.

QBE in server side

In order to respond to client side’s query, our database server provides a set of methods and an algorithm to extract an image’s features. To process a query sent from the client, the server first identifies the features in the query pattern, matches them against features stored in the database, and then returns the searching results.

[Chens2000a] described several steps required in the server environment to realize CBR. First is the feature extraction process, including coarse level extraction and

finer level extraction; Second is feature value normalization, which is to ensure scale independence; Last is feature classification, which is to index multimedia objects.

Segmentation is one of important aspects of image feature extraction. [Chens2000a] also introduced the segmentation of an image to divide the image into smaller parts or regions. During an image segmentation process, an input image is partitioned into regions and each region meets some homogeneity criterion such as the intensity values. The generated regions are called classes, which ideally should correspond to objects.

Meanwhile, key objects also need to be identified for future analysis of the spatial relationship of objects in video data. The server adopts an enhanced unsupervised segmentation method and Simultaneous Partition and Class Parameter Estimation (SPCPE) algorithm [Sistas99] [Chens2000d]. This algorithm takes the problem of video frame segmentation as a joint estimation of the partition and class parameter variables, and can be implemented to identify objects and their corresponding spatial relations.

QBE in client side

With the reliable communication with the server, a user-friendly GUI can be implemented to support query-by-example in the client side. Figure 4.2 is the screen view of the new version of browser interface in client side, named FIU Multimedia Browser. We discuss the network construction between server and client in later section.

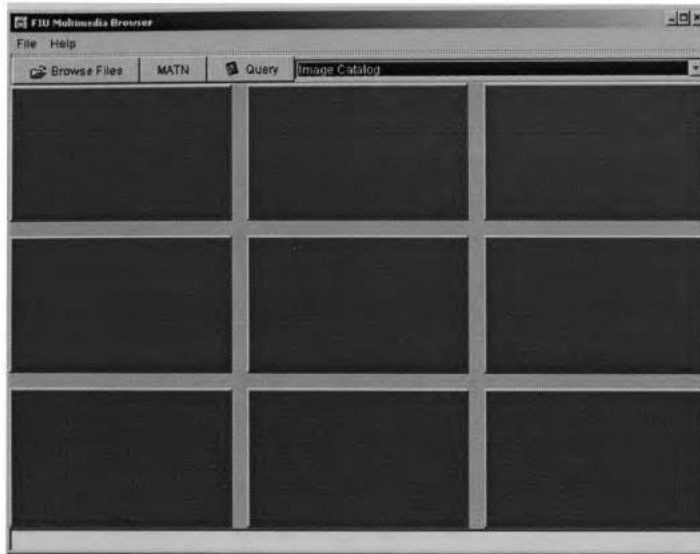


Figure 4.2 The new version of browser interface in client side

The client provides some sample images by default according to various data categories. In Figure 4.3, there are nine sample images displayed after users select a subject domain from the category box. To request a desired image that is similar to any one of these images, simply click a target image (the red highlighted image has been selected in Figure 4.3) and click *Query* button, then at most nine images, which have the similar feature as the query image, are retrieved from the server. Figure 4.4 shows the query result. The process behind these graphics is done as the following:

- The client initiates the query by sending this image's id to the server;
- The server retrieves the features of the query image, and searches the images that have similar features to the query image;
- The server returns the result in the order of descending similarity, following the query image at the first position;
- Repeat above steps until a desired image retrieved.



Browser interface displays nine hurricane images after select Hurricane from the category. The middle red highlighted image has been selected.

Figure 4.3 Sending a image query



Figure 4.4 The query results with the target image in front

Once users find a matching image, they can select this image to be a part of the presentation. The small text window at the lowest part of the interface will specify the file name of the selected image. For example, after query and find the desired image item from category *Hurricane*, to input this item into later presentation, open file browser window for *Hurricane*, find the file name of the selected item, then double click it to download the file from the server (Figure 4.5). This is also a part of browsing function. We discuss this detail in section 4.2.2. The new modified interface thus can grab media information and send it to editor interface for later presentation authoring.

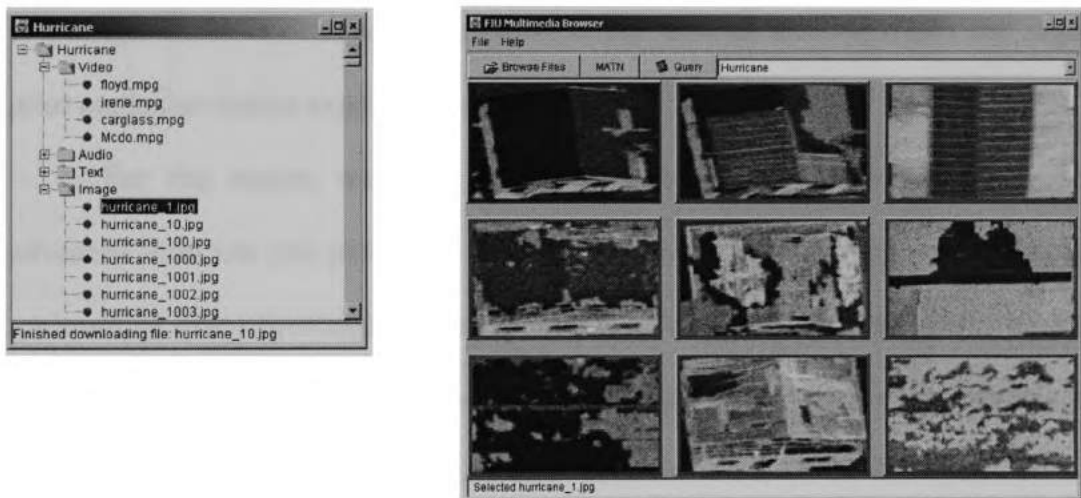


Figure 4.5 Find the selected file from the browsing window for Hurricane

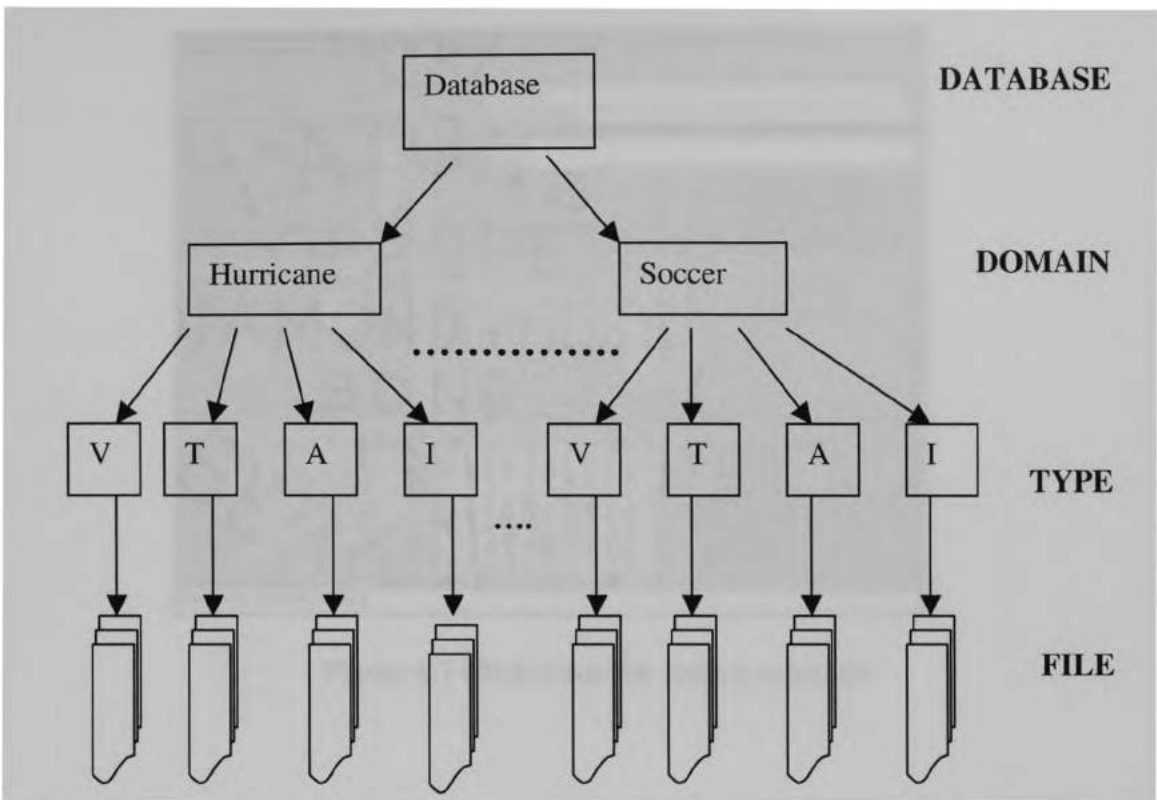
4.2.2 Browsing

Retrieving information by queries reduces the number of false hits, i.e. precision is improved. However, queries are limited to individual parts of a collection or filtering. Browsing is a good way to access documents if users don't have a specific idea of what

they're looking for. As we mentioned in section 4.1, a retrieved media item may not only be a separate media data, but also be a part of a composite multimedia document, such as a complex presentation, which cannot be displayed or played alone. For this reason, we need to retrieve also any other items belonging to the same domain. In the other word, the retrieved data should be associated with the content itself to allow efficient searching for multimedia material of user's interest.

Multimedia database systems have widely demonstrated their ability for organizing, storing and retrieving multimedia information. Users of such systems can browse across several pre-defined paths to access information that is organized into units of storage. These environments are user-friendly, provide nice interfaces and require no prior particular system expertise from the user.

For this reason, we offer two level of browsing, simple and advanced. The advanced form lets you perform a finer-grained search by browsing a set of image files retrieved by query-by-image, as we mentioned in section 4.2.1. The simple browsing is under a hierarchy schema, which allows users to get an overview of resources covering broader or narrower topics as they search different levels of the hierarchy. This offers users the opportunity to reach a subject they are interested in, and select documents that interest them from the list they are presented with.



V: Video
 T: Text
 A: Audio
 I: Image

Figure 4.6 The hierarchical structure of database in server site

In the database server, the media information is stored in a pre-defined hierarchical structure (Figure 4.6), which is constructed by the domain of the media content and the media type. Figure 4.7 shows the choice box for subject selection.



Figure 4.7 Choice box for subject selection

To perform browsing by subject, select a subject from the choice box “Subject category” in right-top of the interface (Figure 4.7). Once the user select a subject, for example, *Hurricane*, a browser frame can be brought out after clicking *Browse files* button from left top of the menu bar. There is a hierarchical tree that holds the archive pertaining to that subject, i.e. *Hurricane* in this case. Every subject’s archive tree usually lists four types of media data as their second level classification, such as video, audio, image and text, as shown in Figure 4.8. A part of construction of this archive tree is given in Table 4.1:

```

//subject of the tree
rootNode = new DefaultMutableTreeNode();

//create a tree specifying whether any node
can have children
treeModel = new DefaultTreeModel();

//register interface to manipulate the tree
treeModel.addTreeModelListener();
tree = new JTree();
tree.addTreeSelectionListener();
tree.addMouseListener();
tree.enableEditing

//the default subdirectory for four data
types
subdirectory_for_video = "Video";
subdirectory_for_audio = "Audio";
subdirectory_for_text = "Text";
subdirectory_for_image = "Image";

add_subdirectory(subdirectory_for_video);
add_subdirectory(subdirectory_for_audio);
add_subdirectory(subdirectory_for_text);
add_subdirectory(subdirectory_for_image);

```

Table 4.1 Pseudo code for directory tree

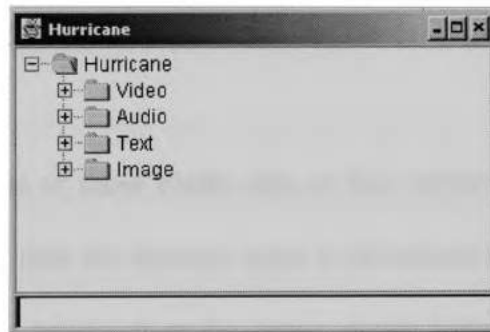


Figure 4.8 The screen view of the archive tree for subject “Hurricane”

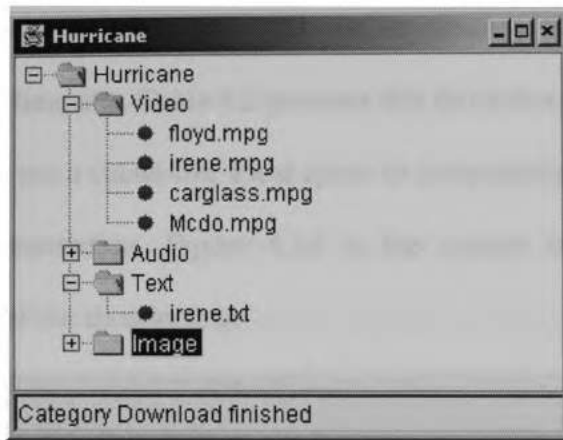


Figure 4.9 The screen view when category downloading finished

```

object = selected node from the tree;
category = the object's root parent;
if (double click)
  if(object is root)
    show text "Downloading file list for
    "object" ...";
    download file list for "object";

  else
    show text "Downloading file...";
    downloadfile "object" from domain
    "category";

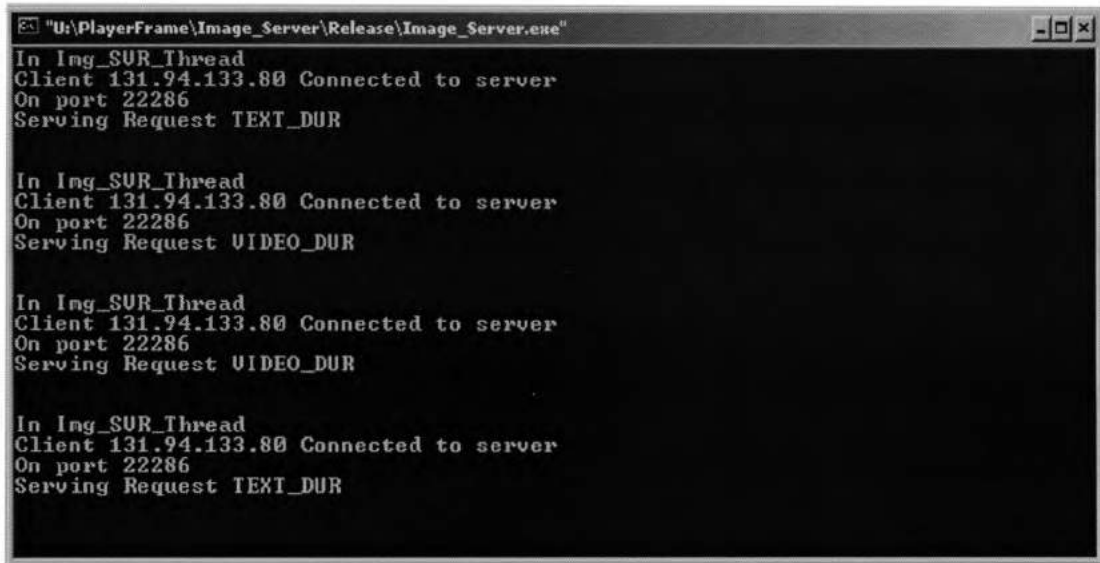
```

Table 4.2 Pseudo code for file browsing

To access any type of these media data in this subject domain, simply click the folder named by the type, then the browser starts to download the list of all available data of the selected type in this domain from the server. At the bottom of the frame, a message indicates the download status. Figure 4.9 shows the list of the available files listing under the video folder after “Category download finished”. At this time being, only the list of these files has been transferred from the server. To get an interesting file from this list,

double click the file name from the tree. This double clicking initiates server to transfer the entire file into the client site. Table 4.2 presents this procedure in pseudo code.

The client reserves a cache-like local space to temporarily store the selected files for later real-time presentation. Figure 4.10 is the system output from the server indicating the process of the transferring.



```
"U:\PlayerFrame\Image_Server\Release\Image_Server.exe"
In Img_SUR_Thread
Client 131.94.133.80 Connected to server
On port 22286
Serving Request TEXT_DUR

In Img_SUR_Thread
Client 131.94.133.80 Connected to server
On port 22286
Serving Request VIDEO_DUR

In Img_SUR_Thread
Client 131.94.133.80 Connected to server
On port 22286
Serving Request VIDEO_DUR

In Img_SUR_Thread
Client 131.94.133.80 Connected to server
On port 22286
Serving Request TEXT_DUR
```

Figure 4.10 The system output of the server when request file transmission

4.3 Communication between the server and the client

Due to the heterogeneity of multimedia database systems, two major communication problems between the server and the client need to be solved in our system. One is how to achieve real-time retrieval of video and audio data. The other one is how to allow the client to access hierarchical structured database in the server.

In our system, the communication between the server and the client is based on Internet Protocol (IP). There are two transfer protocols available in IP family, UDP (User Datagram Protocol) and TCP (Transmission Control Protocol). The primary difference

between TCP and UDP is that UDP does not necessarily provide reliable data transmission. Traditional networks are used to use TCP to provide reliable end-to-end service by using error recovery. However, when the goal of a program is to transmit as much information as quickly as possible, any given piece of the data is relatively unimportant. The transmission protocol cannot accept retransmission, since that might introduce unacceptable delay. TCP is not practical in this case because error recovery mechanism will increase network latency [RB01]. UDP is the simplest possible transport protocol, which extends the host-to-host delivery service of the underlying network into a process-to-process communication service [PeDa00]. Besides no error recovery, UDP has the following aspects that are also in our concern:

- No connection establishment. TCP uses a three-way handshake before it starts to transfer data. UDP just blasts away without any formal preliminaries. Thus UDP does not introduce any delay to establish a connection.
- No connection state. TCP maintains connection state in the end system in order to provide reliable data transfer service and congestion control. UDP does not maintain connection state and does not support congestion control. For this reason, a server devoted to a particular application can typically support many more active clients when the application runs over UDP rather than TCP.
- Small segment header overhead. The TCP segment has 20 bytes of header overhead in every segment, whereas UDP only has 8 bytes of overhead.
- Unregulated send rate. TCP has a congestion control mechanism that throttles the sender when one or more links between sender and receiver becomes

excessively congested. This throttling can have a severe impact on real-time data retrieval, which can tolerate some packet loss but require a minimum send rate. On the other hand, the speed at which UDP sends data is only constrained by the rate at which the application generates data, the capabilities of the source (CPU, clock rate, etc.) and the access bandwidth to the network. We should keep in mind, however, that the receiving host does not necessarily receive all the data - when the network is congested, a significant fraction of the UDP-transmitted data could be lost due to router buffer overflow. Thus, the receive rate is limited by network congestion even if the sending rate is not constrained.

Based on the UDP protocol, we develop the communication between the server, which uses Unix socket implemented in C++, and the client, which uses Java socket implemented in Java. Figure 4.11 illustrates the communication routine between the server and the client. When the system is running, the Unix server creates a socket, binds it to a local address and a predefined port, and waits for the requests from clients. When a client sends the request to the port, the server creates a process to service the client. Then the new server process talks with the client. Once a UDP socket has been created and bound to a local source port, it is now capable of being used for sending and receiving datagrams. The functions for sending and receiving datagrams are *sendto* and *recvfrom*.

When an IP address and a port number are given, the Java socket for the client can be created (Table 4.3).

```

udpSocket = new DatagramSocket();
addr = InetAddress.getByName(ipAddr);
port = port_num;

```

Table 4.3 Pseudo code for Java socket

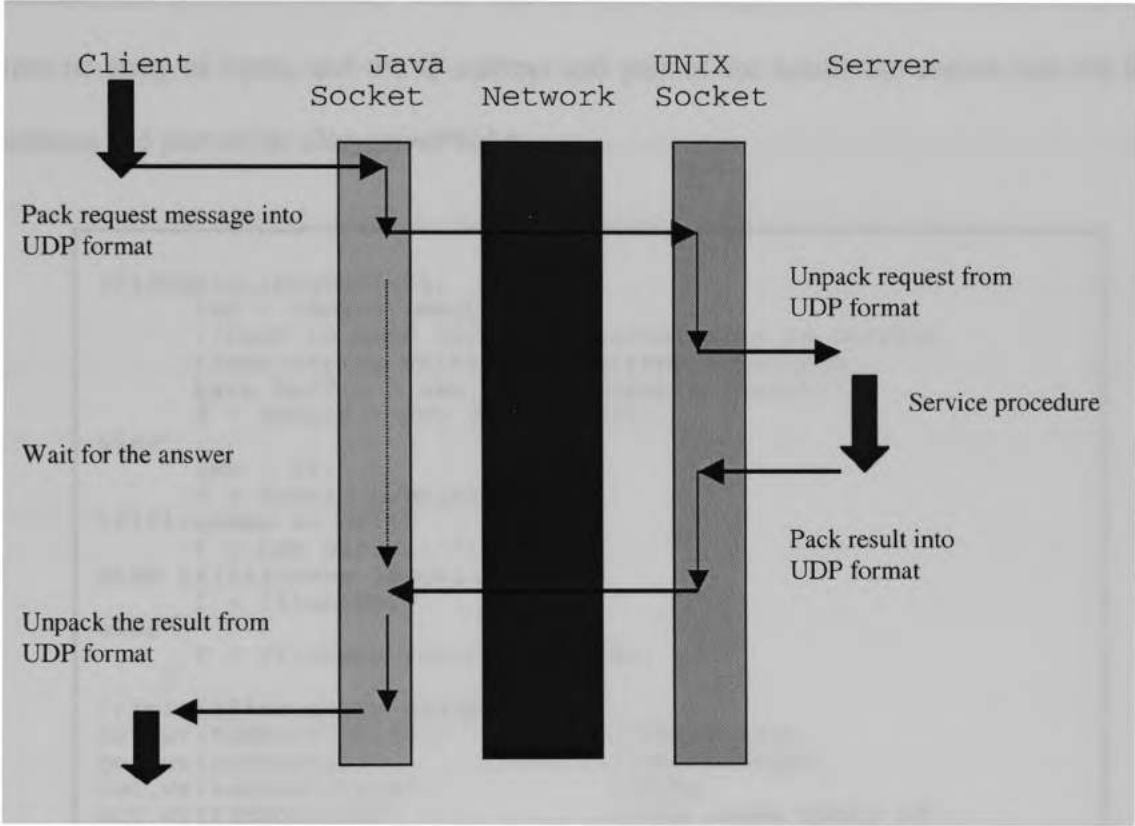


Figure 4.11 The communication routine between the server and the client

size	seqNo	type	num_frags	flag	data_size	domain	filename
------	-------	------	-----------	------	-----------	--------	----------

Table 4.4 The packet header constructure

Java encapsulates the concept of a UDP socket with the class *DatagramSocket*, and the concept of a datagram with the class *DatagramPacket*. A *DatagramPacket* consists of a fixed-length array of bytes together with an IP address and port. When one sends a *DatagramPacket*, its array of bytes is sent to the socket with the specified IP address and port (if it exists). When one receives a *DatagramPacket*, the data is copied into its array of bytes, and the IP address and port of the sender are copied into the IP address and port of the *DatagramPacket*.

```
if(domain.length() $<$ 24)
    len = domain.length();
    //hack to make sure that packet size is correct
    //and string written to correct locations
    byte buff[] = new byte[24-domain.length()];
    d = domain + new String(buff);
else
    len = 24;
    d = domain.substring(0,24);
if(filename == null)
    f = new String("");
else if(filename.length() $<$ 24)
    f = filename;
else
    f = filename.substring(0,24);

//initialize query message
out.writeShort(QSize);           //short size
out.writeShort(0);               //short seqno;
out.writeShort(type);           //type
out.writeShort(id);             //the image query id
out.writeShort(9);              //number to query
out.writeShort(0);              //pad
out.writeShort(len);            //short d_size; size of domain
out.writeShort(f.length());     //short fn_size; size of filename
out.writeBytes(d);              //byte domain[24];
out.writeBytes(f);              //byte filename[24];
```

Table 4.5 Pseudo code for datagram packet in the client side

The UDP packets consist of two parts: header and data body. The data body contains byte-streams of different types of media data. The header is given in Table 4.4,

which consists of eight fields of information. *size* is the size of the frame; *seqNo* is the sequence number of the packet, *type* indicates the data type of the transferring file; *num_frags* is the number of packets for an image or file which size is bigger than *size*; *flag* is assigned to be 0 if the frame is the last one; *data_size*, *domain* and *filename* carry information about the file to be transferring: *data_size* is the total data size of the file, *domain* is typically needed for matching data hierarchy. The client starts to send query message by wrapping the packet. Table 4.5 describes the procedure that the client wraps the data into a packet. Notice that the domain length is limited by 24 bytes.

After the query message is wrapped, the output packet is ready by calling function *get_output_packet(InetAddress addr, int port)*. It returns a *DatagramPacket* by taking parameters *bytearray of output stream*, *size of output stream*, *addr*, and *port*. Then, to send the query, call *send(DatagramPacket)*.

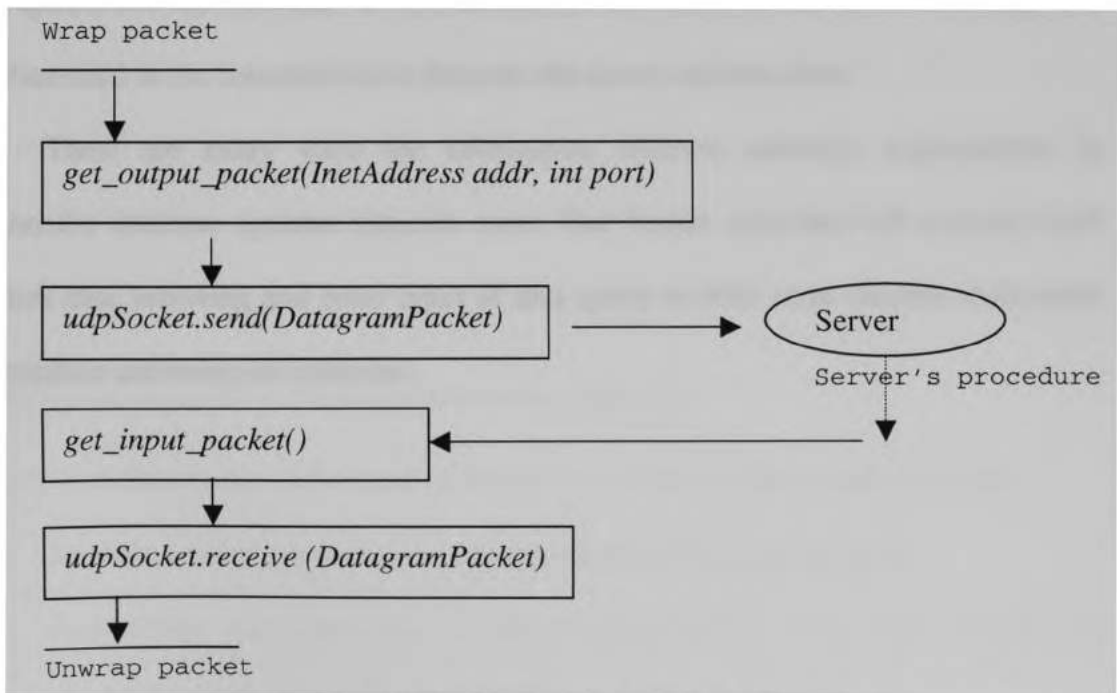


Figure 4.12 Sending and receiving in client site

To receive query results or files selected from browsing, an opposite procedure is started by calling function *get_input_packet()* , which returns the a *DatagramPacket* to hold a receiving packet. The system call *udpSocket.receive(DatagramPacket)* is used at this time being. At last, the client unwraps received packet, checks the packet header, and consumes the data in the data body of the packet. Figure 4.12 illustrates the whole sending and receiving procedure with a set of function calls in the client side.

4.4 Conclusion

In this chapter, we have discussed the requirements for information retrieval and how to manage information retrieval. In order to provide input data for multimedia presentation authoring, we have introduced our novel browser interface, which uses content-based-retrieval and archive browsing methods to retrieve user-selected data. Since our system is built upon a two-tier client/server architecture, we also illustrate the network strategy and interface used in the communication between the server and the client.

There are many ways for information retrieval currently experimented in multimedia database systems research areas. Our further activities will concern more efficient data browsing and more types of data query to offer more flexible multimedia presentation authoring environment.

5. Multimedia presentation authoring

Multimedia presentation authoring is one of the demands to be met in the multimedia database system. The problem to be examined is the capability of manipulating and presenting arbitrary combinations of different media types to one multimedia composition. This calls for models that support the description of such multimedia compositions. As we mentioned in Chapter 2, there are different means for the representation of multimedia compositions. In our system, Multimedia Augmented Transition Net (MATN) [Chens2001a] [Chens1997] is the model we use to capture the arrangement of different media objects with different temporal and spatial relations among them.

In this chapter, we briefly describe the general process of multimedia presentation authoring in section 5.1. Then we discuss MATN model, including its graphic view and grammar, time flow in the model, and data structure designed for editing a MATN model in section 5.2. We present the editor interface and its functionality for editing the model and user interaction in section 5.3, and introduce the visualization of multimedia presentation in Section 5.4. A conclusion is given in section 5.5.

5.1 General process of multimedia presentation authoring

Normally, constructing a multimedia presentation consists of three major processes:

- Creating and editing the media items comprising the presentations;
- Assembling the items into a coherent presentation, where this includes the specifying of the temporal and spatial layout of the items; and
- Specifying the interactions between the user and the presentation.

Our task concentrates primarily on designing an authoring interface that supports the last two of these – the assembly and interaction processes. We are less concerned with the first – the creation of individual media items – that requires the use of specialist data editors for the range of media types used.

In some respects, authoring multimedia presentation can be compared with word processing. Both activities require the collection /generation of source material and the placement of these sources within a presentation environment. A generic word processor allows an author to layout information for use on a printed page. Depending on the features supported by the formatter, authors may be able to vary the font and size of the text, they may be able to vary the spatial layout of the information on the page, and they may be able to incorporate higher-level structures, such as chapters and sections, in the document. In the same way, multimedia presentation authoring tools allow an author to integrate several types of information into a composite presentation. Unlike text, however, the temporal dimension often dominates the multimedia presentation authoring process because of the heavy time-dependant property of multimedia data. Here an editor is concerned that the individual shots that have been created are assembled into sequences which are in turn are grouped into scenes containing a single coherent thread of the story [RuDa89].

5.2 MATN

The construction of a coherent multimedia presentation composed from its constituent parts is a non-trivial task. While the nature of media data is getting complex, the relationship among heterogeneous multimedia data becomes more complicated. One of

the inherent characteristics of multimedia data is the heavy time-dependence in that they usually related by temporal relations, which have to be maintained during their playout [Chens2000a]. In order to synchronize the various data types for sophisticated multimedia presentations, we introduce a novel semantic model, called Multimedia Augmented Transition Net (MATN) [Chens1997a] [Chens1997b], as the underlying paradigm for construction.

5.2.1 Graphic view and grammar

MATN was extended from Augmented Transition Net (ATN) [Woods70], which originally created by Woods for natural language understanding systems and question answering systems for both text and speech. Based on the major rules and principles of ATN, [Chens1997a] [Chens1997b] proposed MATN to author multimedia presentation, multimedia database searching, the temporal and/or spatial relations of various media streams and semantic objects, and multimedia browsing. Our task concentrates on its ability of authoring multimedia presentation.

An ATN represents the temporal relationship among multimedia data by a finite set of nodes and arcs. Nodes represent different states of transition, and arcs indicate the time flows of the presentation. Each arc comes with a label that defines the media items to be represented in the time interval. This time interval is specified by two states connected by the arc. In other word, the input of the label can cause a transition from the state at the tail of the arc to the state at its head [Chens2000b].

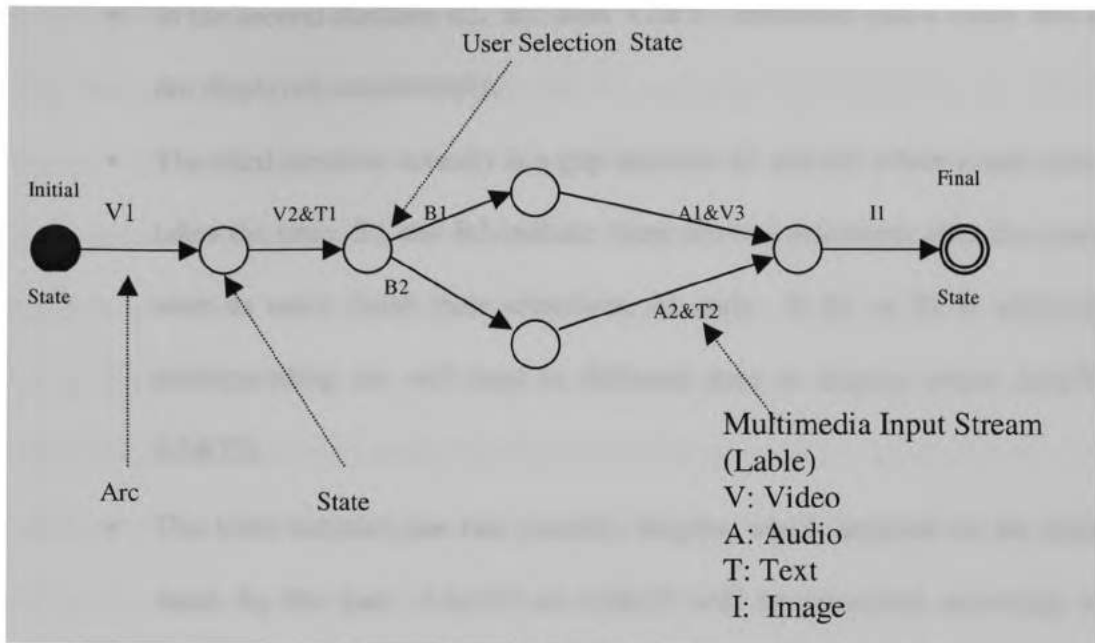


Figure 5.1 MATN graphic form and grammar

The composition of MATN is very similar to ATN. Its graphic form and grammar can be illustrated in Figure 5.1. Two notations L and D are used to define multimedia input strings and defined as follows:

- $L = \{V, A, T, I\}$ is the set whose members represent the media type, where V, A, T, I denote video, audio, text, and image, respectively.
- $D = \{0, 1, \dots, 9\}$ is the set consisting of the set of ten decimal digits.

5.2.2 Time flow in MATN

Figure 5.2 illustrates the time flow in the MATN. The sequential process from the initial state to the final state can be interpreted as the following:

- In the first duration d_1 , V1 is presented alone. Once V1 consumes its specified duration, the control goes to the next state;

- In the second duration d_2 , arc label $V2\&T1$ represents that a video and a text are displayed concurrently;
- The third duration actually is a gap between d_2 and d_4 , where a user selection takes the time. $B1$ and $B2$ indicate there are two selections after this state. As soon as users finish their selections, d_3 ends. If $B1$ or $B2$ is selected, the corresponding arc will lead to different state to display either $A1\&V3$ or $A2\&T2$;
- The fourth duration has two possible lengths, which depends on the selection made by the user. $A1\&V3$ or $A2\&T2$ will be presented according to the selection;
- After displaying either selection in d_4 , there is a merge, i.e. either of them leads to a common state, where goes to d_5 ;
- d_5 is the last duration which ends when $I1$ finishes, and leads to the final state, where it is also the end of the presentation.

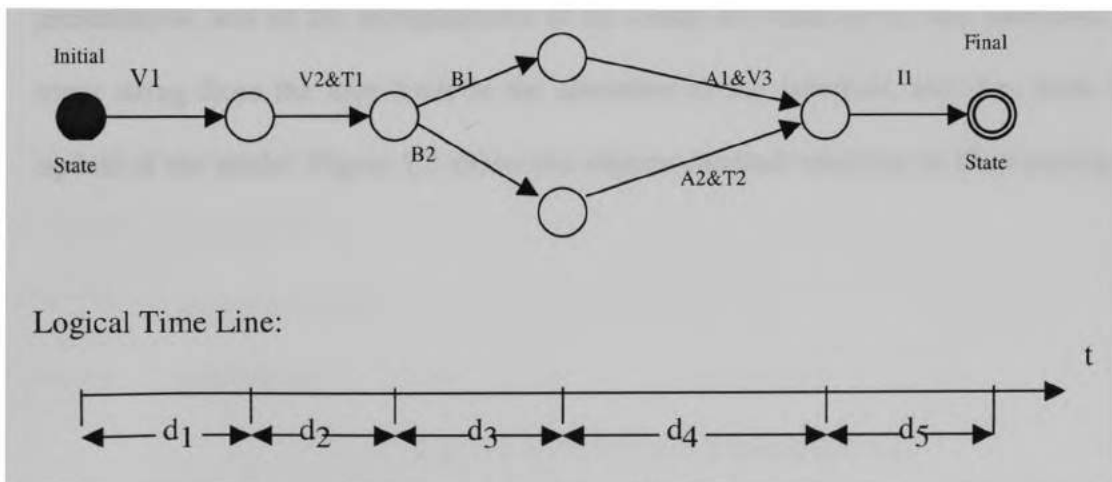


Figure 5.2 The time flow in a MATN

5.2.3 Data structure of MATN

The data structure underneath the MATN is constructed by linked list, in which the objects are arranged in a linear order. Linked list provides a simple, flexible representation for dynamic sets, supporting *search*, *delete* and *insert* operations with small runtime. In a typical implementation of a dynamic set, each element is represented by an object whose fields can be examined and manipulated if we have a pointer to the object [CLR90]. From the implementation's point of view, Java *LinkedList* class provides uniformly named methods to *get*, *remove* and *insert* an element at the beginning and end of the list. Operations that index into the list will traverse the list from the beginning or the end, whichever is closer to the specified index.

In this design of implementation, the core object *Matn* consists of three basic linked lists: *stateList*, *arcList*, and *labelList*. The *StateLists* and the *ArcList* are designed for the model to keep a reference of its states and arcs so that it is easy for the model to retrieve their information when needed. *labelList* contents all the components of the presentation, and all the manipulations of the model are based on it. Any alteration in the input string from the user leads to the alteration in the *labelList*, and then leads to the update of the model. Figure 5.3 shows the objects and their relations in *Matn* package.

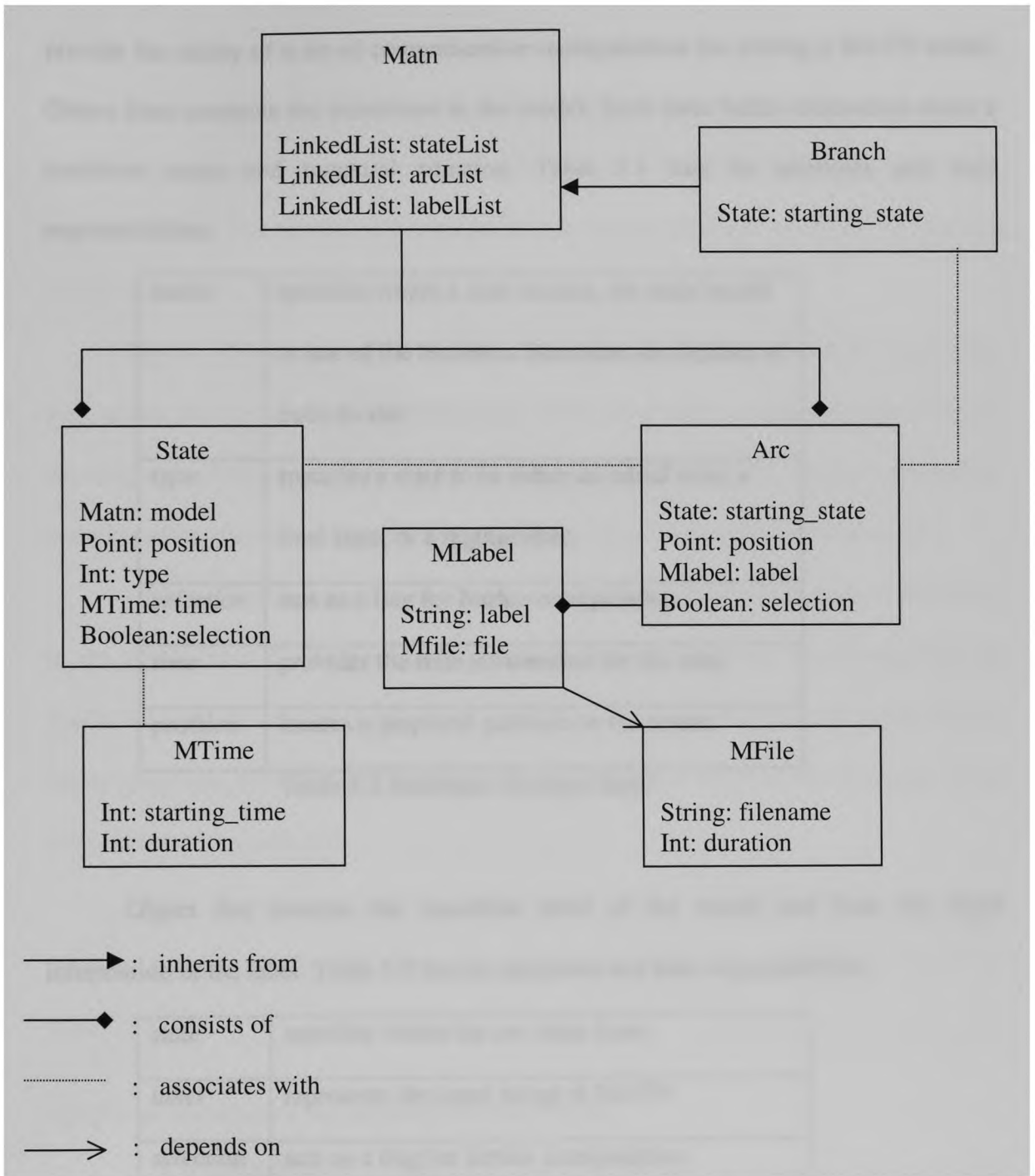


Figure 5.3 The objects in Matn package

Object *State* and *Arc* are the basic components of MATN model. Their attributes provide the ability of a set of comprehensive manipulations for editing a MATN model. Object *State* connects the transitions in the model. Each state holds information about a transition status and temporal situation. Table 5.1 lists its attributes and their responsibilities.

<i>model</i>	specifies where a state locates, the main model or one of the branches. Branches are children of main model
<i>type</i>	specifies a state to be either an initial state, a final state, or a regular state
<i>selection</i>	acts as a flag for further manipulation
<i>time</i>	provides the time information for the state
<i>position</i>	locates a graphical position on the screen

Table 5.1 Attributes of object *State*

Object *Arc* controls the transition trend of the model and links the input information of the label. Table 5.2 lists its attributes and their responsibilities.

<i>state</i>	specifies where the arc starts from
<i>label</i>	represents the input string of MATN
<i>selection</i>	acts as a flag for further manipulation
<i>position</i>	locates a graphical position on the screen

Table 5.2 The attributes of object *Arc*

Object *MLabel* is the element that forms *labelList*. It wraps a set of *MFile* objects into the format that MATN requires. It semantically indicates a set of media items to be displayed concurrently in certain duration. Java Vector class is used to contain this set of *MFile*. Its major operations include *getLabel()*, *setLabel(string)*, *getFiles()*, and *setFiles(vector)*. The searching operations in Java Vector class are referenced for *search*, *insert*, and *delete* a file from the label.

Object *MTime* consists of *starting time* and *duration*, which are the basic time information for a single medium presentation. It is referenced by object *State* from which the temporal information can be updated by the author of the model. Its major operations include *getStartTime()*, *setStartTime(time)*, *getDuration()* and *setDuration(duration)*.

Object *MFile* is designed to point to a media item and hold the meta-information of the media item. Currently, this meta-information only includes the filename of the media item and its original duration. Its operations are *getFileName()* and *getDuration()*. More operations are to be implemented when the server is ready to provide more meta-information of a media item.

Object *Branch* is a child of object *Matn*. Besides all attributes and operations inherited from its parent object *Matn*, it needs a *state* attribute to attach itself to a certain state.

5.3 The editor interface for presentation composition

As we mentioned before, the architecture of the client side in our system can be divided into three layers: the data layer, the presentation composition layer and the runtime layer. This chapter concentrates on the presentation composition layer, where a flexible editor

interface has been designed and implemented for presentation composition purpose.

Figure 5.4 is the screen view of the editor.

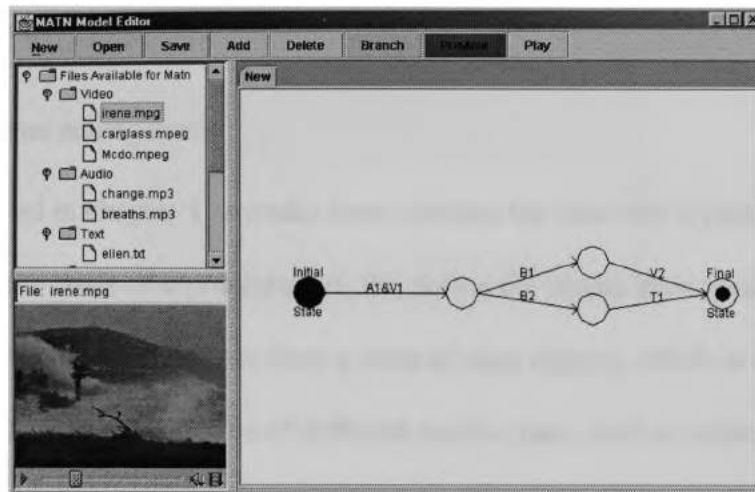


Figure 5.4 The screen view of the editor

As Figure 5.4 illustrates, the general organization of the editor screen consists of four parts:

- The menu bar

It contains buttons to activate all commands. Some buttons, such as branch, preview and play, will turn red when the corresponding command is selected.

- The editing window

It provides a set of editing tools to composite MATN model, easily manipulated by mouse click.

- The file window

The file window contains a directory tree listing all the media items that users have selected from the browser. It automatically sorts and stores the items by their data types under proper sub-directories.

- The preview window

It previews a single selected media item for user to better organize the presentation.

5.3.1 Media items management

As we introduced in chapter 1, a media item contains the data that is presented to the user and as such is the basis of a presentation. We define the media item as an amount of data that can be retrieved as one object from a store of data objects, which is not necessarily a small amount. Media items can be of different media types, such as video, audio, text and image. In order to have a clear view of all the media items that users have selected from the browser, a directory tree is imported based on Java's JTree structure. Its construction basically is the same as the directory tree in the browser interface. Users can always go back and forth between the browser and the editor interfaces to selected their desired media items for presentation composition.

There are two methods provided to add/delete a media item to/from the editing window. *select(media_item)* and *deselete(media_item)*. Figure 5.5 is the pop-up menu brought out by right clicking the mouse after the user select a media item from the tree. Click *Select*, the selected item will be added to a linked list, which is the data structure of a label in the MATN; Click *deselete*, the target item will be removed from the list if it has been selected. On the menu bar, clicking *Add* button brings out the linked list of the preselected items. Figure 5.6 is the dialog of the list. The selection is confirmed by clicking *Enter* button on the dialog, and a corresponding MATN model is drawn on the editing window (Figure 5.7).

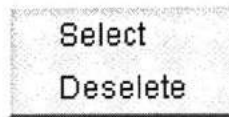


Figure 5.5 Dialog for selecting media item from the directory tree

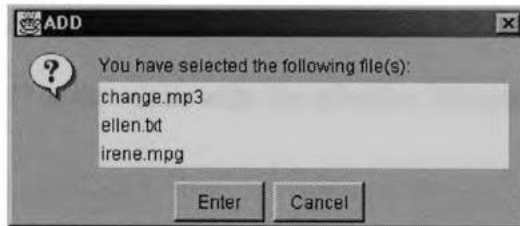


Figure 5.6 Dialog for selected items list

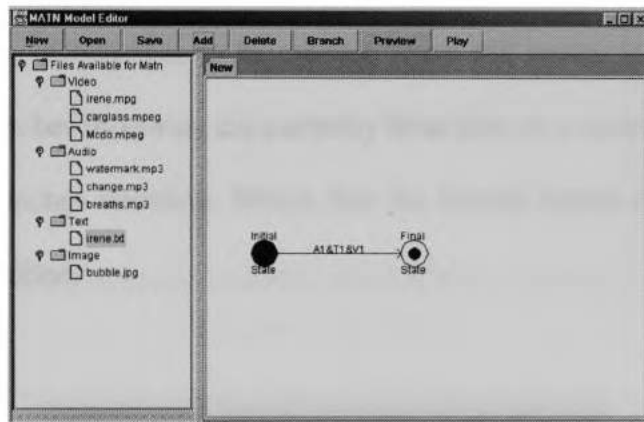


Figure5.7 The editor draws the model after the selection is confirmed

Meanwhile, in order to help users make more proper selection of media items, this interface also offer a preview function. To turn on the preview, click *Preview* button from the menu bar to bring out the preview window. In Figure 5.4, the left-bottom window is previewing a video media item, and the preview button is turning red during the process of preview.

5.3.2 Model editing

- Basic steps

After confirming the selection of the media items, the editor automatically draws/updates the model as required. As shown in Figure 5.7, a two-state model with an arc labeled “A1&T1&V1” is the result with the selection illustrated in Figure 5.6.

- Branches

To create the branches, click *Branch* button from the menu bar. This brings up dialog “*Create branch*”. In figure 5.8, the area “*Create Branch from State*” indicates the index of the state where the new branches start; the area “*Number of Adding Branches*” specifies the number of branches to be created. Once *OK* button is clicked, the editor draws the initial branches following the currently final state or a selected state. Figure 5.9 (a) shows a two-branches situation. Notice that the branch button is turning red when branches are being edited.

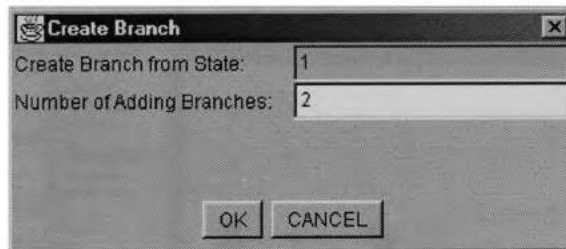
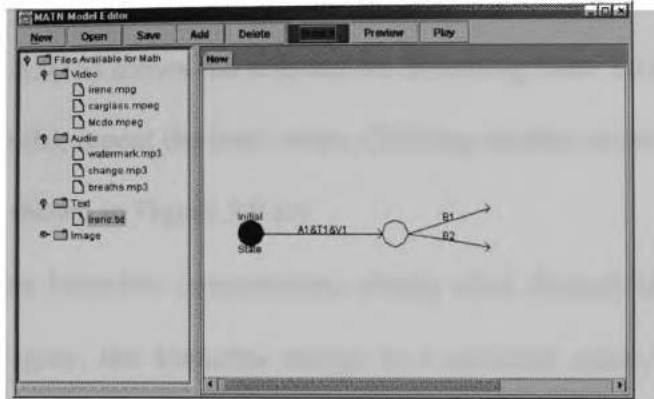


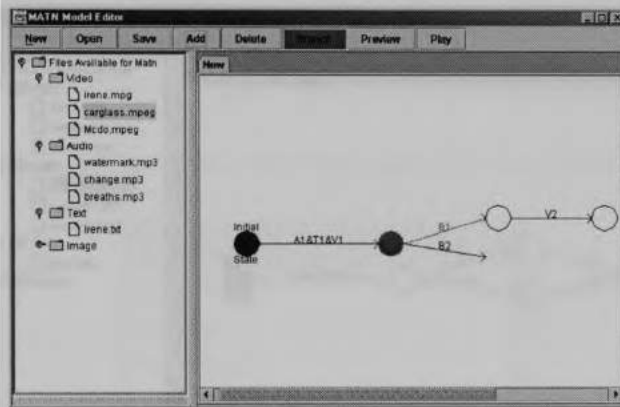
Figure 5.8 Dialog for creating branches

a)



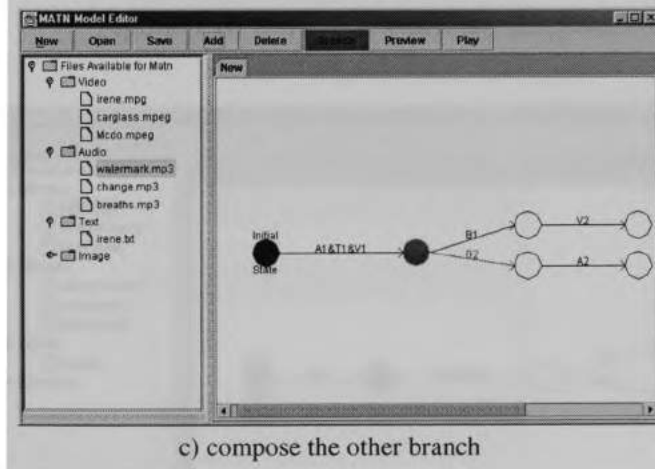
a)two branches has been initialized

b)



b)compose one branch

c)



c) compose the other branch

Figure 5.9 Branches composition

When branches are initiated, select one of arcs extending from the certain state by mouse click. Figure 5.9 (b) shows the selected arc is turning blue. To compose a selected branch as Figure 5.9 (b), repeat the basic steps. Clicking another arc will lead to compose the other branch, as shown in Figure 5.9 (c)

To complete branches composition, simply click *Branch* button again. While the button turns to gray, the branches merge to a common state. Figure 5.10 is the completing status of the branches.

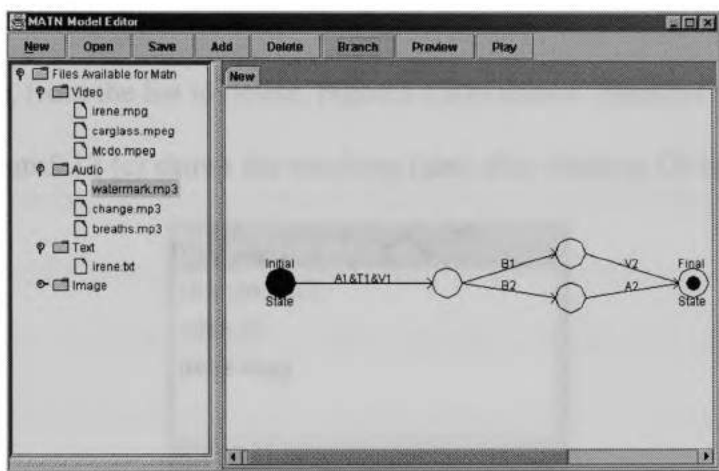


Figure 5.10 Complete branches composition

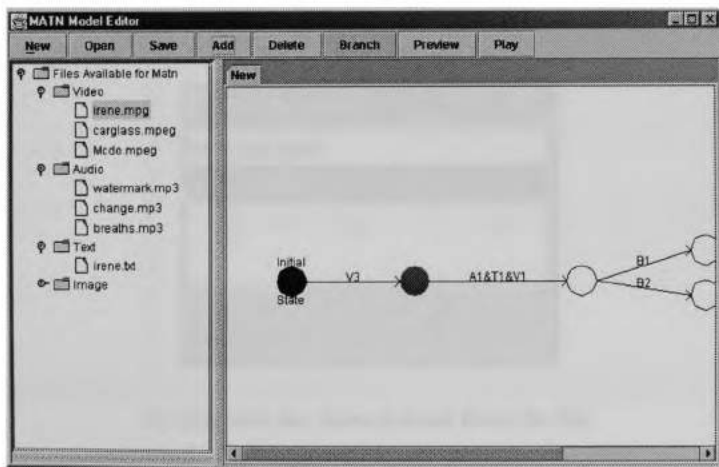


Figure 5.11 Insertion of a new state

- Insertion

To insert a new state, simply select a target state, then repeat the basic steps.

Figure 5.11 shows the result insertion before initial state. The second state is newly inserted state.

- Deletion

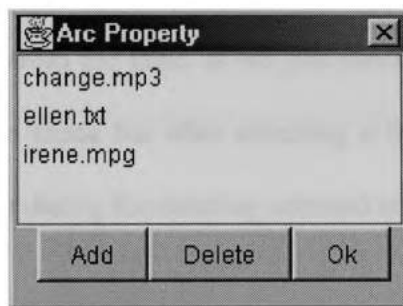
To delete a media item from an existing label, right click the arc with target label.

A dialog called arc property lists all the media items placed on this label. Figure 5.12 (a)

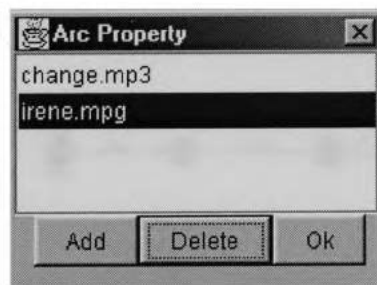
is arc property of the label "A1&T1&V1" in figure 5.11. Select a media item, i.e. a file

name in this case, from the list to delete. Figure 5.12(b) shows "ellen.txt" has been deleted

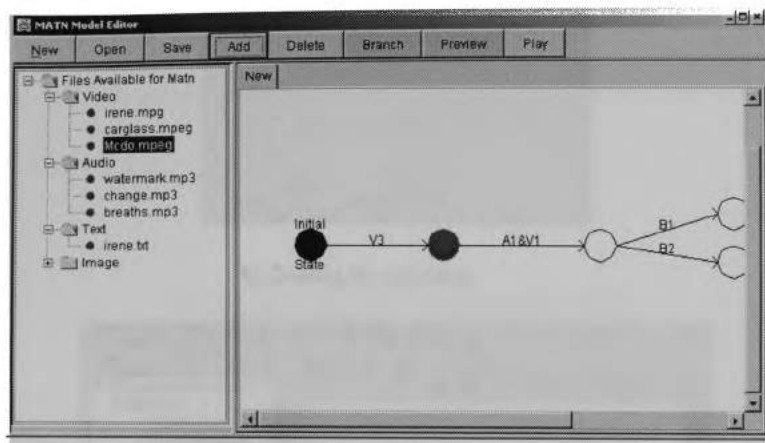
from the list. Figure 5.12 (c) shows the resulting label after clicking *Ok* button.



a) The list of the media items the label "A1&T1&V1"



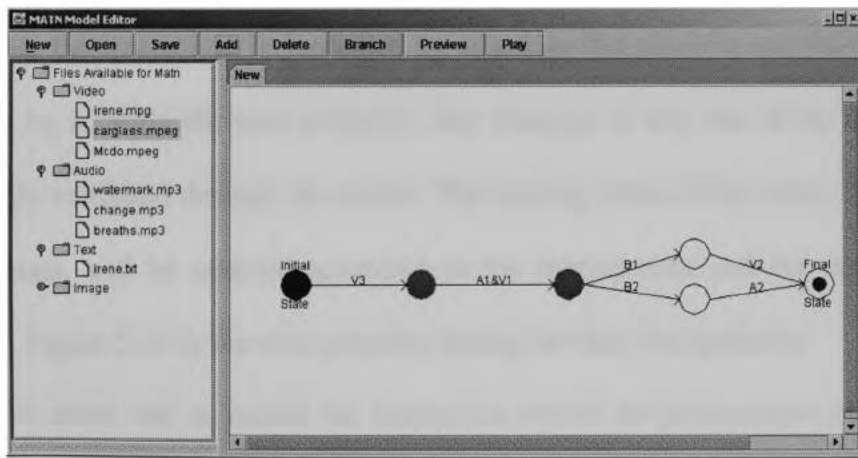
b) One item has been deleted from the list



c) The label is "A1&V1" now

Figure 5.12 The process of deleting media item from a label

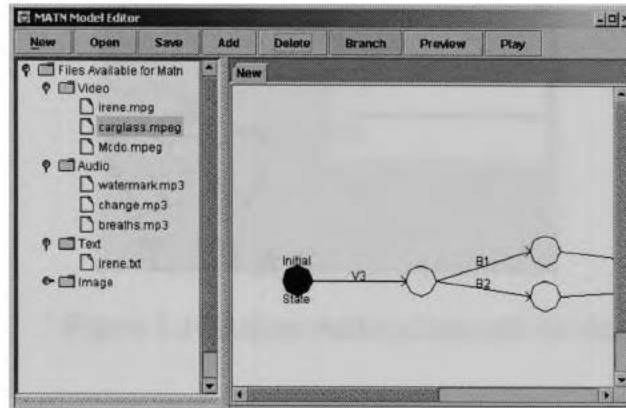
There are two ways to delete an arc and its corresponding state. One way is deleting all the media items from the label as we just mentioned above, the other way is using *Delete* button from the menu bar after selecting a from-state and to-state (Figure 5.13(a)). Figure 5.13(b) is the dialog for deleting selected states or the whole model.



a) Select two states to delete the arc between them



b) Dialog for deletion



c) The arc has been deleted

Figure 5.13 The process of deleting an arc directly

5.3.3 User interaction

The user interaction can be handled before or during the presentation is playing back. In the first case, the interaction is processed by adjusting the duration and the starting time of the state by refining the state property. Any changes in any one of the states will be automatically validated through the model. The starting times of the states, which follow the target state, will be updated according to the starting time and the duration of the target state. Figure 5.14 is the state property dialog for time manipulation.

The other way to handle the interaction before the presentation playback is by selecting starting state and ending state. The user can select any two states to be the starting state and ending state of the presentation, or select more states to be the different

starting state and ending state in order to skip some states in the model. In Figure 5.15, only V1 and V2 will be presented, while A1 and T1 are skipped because of the selection of the starting and ending state.

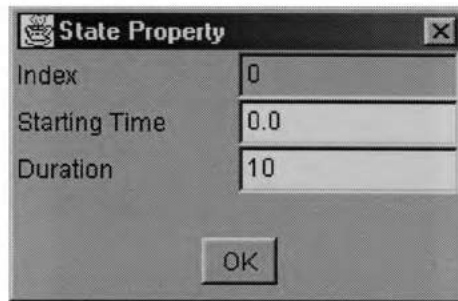


Figure 5.14 Adjust starting item and the duration

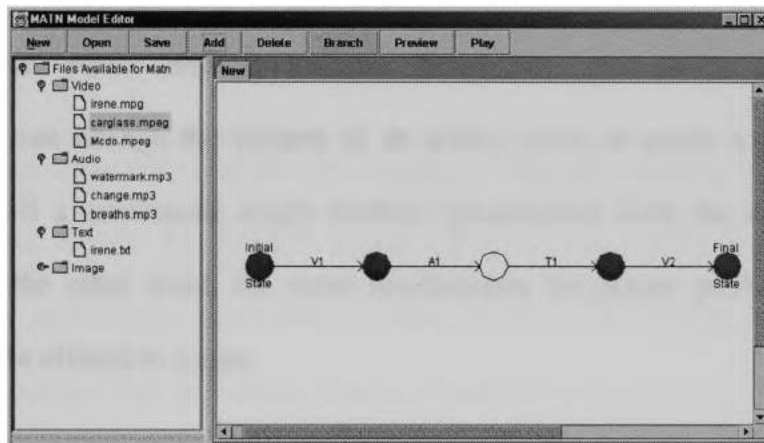


Figure 5.15 Interaction by selecting different starting and ending state

The interaction also can be handled during the presentation playback. As we have introduced that MATN model allows users to create multiple branches from a single state. Therefore, if there are branches between a starting state and an ending state, the presentation will be paused at the state that has multiple branches. At this time being, a

user selection dialog pops up to lead the users to make a selection. In Figure 5.16, B1 and B2 indicates there are two selections after this state. Once B1 or B2 is selected, the playback goes on, and the arc will lead the presentation to corresponding state to continue the presentation.

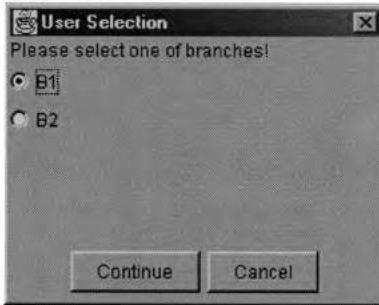


Figure 5.16 Interaction by selecting a branch

There are some other interactions during the presentation playback. For example, a user can change the volume of an audio, move or resize a picture on the screen, or turn off a continuous single medium presentation from the remainder of a presentation. In the other word, the more functionality the player provides, the more interactions can be offered to a user.

5.4 Multimedia presentation visualization

There is another interface called player implemented by [Lo2002]. Although this thesis concentrates on authoring support in a multimedia environment, it is necessary to briefly introduce the player, since it is part of the authoring environment to extend and the author needs to be able to check how the presentation will look to the end-user.

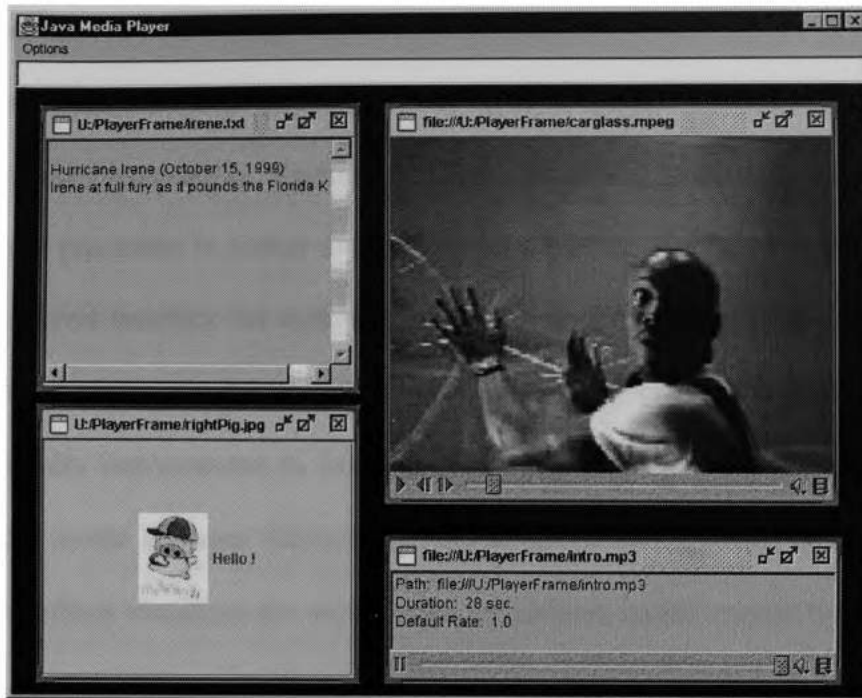


Figure 5.17 The screen view of the player

The player interface consists of four types of internal frames: text, image, video and audio internal frames. Figure 5.17 in the following page illustrates four types of internal frames in multimedia presentation model interface. The left-top and left-bottom internal frames indicate text and image frame, respectively. Java Swing components are used to construct and display the text and image multimedia data. The right-top and right-bottom internal frames indicate video and audio frame, respectively. Java Media Framework (JMF) media player is used to construct and display the video and audio multimedia data. Users are allowed to minimize, maximize, and close any one of the internal frames. The player also provides VCR-style presentation controls for continuous data. Those controls include stopping, starting, fast-forwarding, and rewinding the movie, etc.

5.5 Conclusion

Based on the semantic mode MATN, various types of media items can be synchronized in a multimedia presentation. Assembling media items and specifying user interactions are two major processes to author multimedia presentation. The editor introduced in this chapter is a novel interface for authoring a sophisticated multimedia presentation. Three linked lists and six major object classes build up the data structure of MATN model. A set of GUI tools implemented in Java programming language support flexible graphic editing of the model and user interactions. Invoking the runtime layer in the client side, the player interface visualizes the multimedia presentation model created by the user.

6. Conclusions and Future Work

Due to the growth of information conveying technologies, various types of media data have been developed in recent decades. Traditional database management systems are undergoing a great reformation in order to meet the demands on processing heterogeneous multimedia data. More and more methods involved with retrieving and presenting multimedia data have been largely investigated in multimedia database research area.

In this thesis we proposed the design and implementation of multimedia presentation authoring and browsing environment in the client side of a novel multimedia database system. Our contribution has been to retrieve information by querying and browsing, and unify the temporal relations among various media items into MATN model. An introduction of the basic characteristics of media items and the methods we used in this work was given in Chapter 1. Chapter 2 took a rich survey on the related research fields, both commercial and academic. We analyzed existing semantic models for multimedia presentation authoring purpose. Advantages and disadvantages among these semantic models were carefully described. The architecture of the whole system, which was finely constructed based on two-tier client/server architecture, was presented in Chapter 3. The client so far is a three-layer integrated presentation framework, which consists of a data layer, a presentation composition layer and a runtime layer. The constructive implementation concentrated on the data layer and the presentation composition layer, where the browser and the editor are the user-friendly interfaces for these two layers, respectively. Chapter 4 discussed how information could be retrieved

as to provide the input data for multimedia presentation authoring. The interface in the data layer was implemented and the communication path between the server and the client was defined. In Chapter 5, we focused on the presentation composition layer. The general process of multimedia presentation authoring was described. As a semantic model, MATN was interpreted to model the temporal relations of multimedia data. The editor interface was intensive implemented for editing MATN model and handling user interactions. While it responds to the data layer to compile the data into model, it also supports the runtime layer to visualize the model, and brings the documented presentation into life.

As the positive result of this research and construction, users can edit the retrieved information as the various input elements to MATN model, and thus attain the synchronizations between heterogeneous multimedia data. Moreover, user interactions are allowed during the multimedia presentation authoring process. This result has encouraged us to continue to construct finer client/server system. Based on the existing work and the limitation on the client so far, we make the following suggestions:

1. In order to retrieve richer information required by users, improve searching method to allow user retrieve more various data in more specific constraint. Update the browser interface to allow users to browse more meta-information of the media data.
2. In order to model detail information of a certain media stream, embed the sub-network functionality into MATN model editing. Update the interface to carry out more editing tools and more user interactions.

3. Improve the interactions between the presentation composition layer and the runtime layer in order to control the spatial layout of the presentation.
4. Currently the retrieved media items are saved locally in order to achieve real-time presentation. If the number of media items needed for a typical presentation increases, large storage space is required for the client side. This increases the difficulty of portability. Hence, the mechanism for the real-time delivery should support on-line presentations and browsing without occupying local memory space.

List of References

- [Adob90] Adobe Systems Incorporated (1990). Postscript Language Reference Manual, second edition. Addison Wesley.
- [Allen83] J. F. Allen (1983). Maintaining Knowledge about temporal Intervals. *Communications of the ACM*, 26(11), Nov, 832-843
- [Apers's] P.M.G Apers, H.M.Blanken and M.A.W.Houtsma. "Multimedia Databases in Perspective".
- [Ates's1996] A.F. Ates, M. Bilgic, S. Saito, and B. Sarikaya, "Using Timed CSP for Specification Verification and Simulation of Multimedia Synchronization," *IEEE J. Selected Areas in Comm.*, vol. 14, no.1, pp. 126-137, Jan. 1996.
- [Arens] Arens, Y., Hovy, E, and Vossers, M. "The knowledge underlying multimedia presentations". In M. Maybury, editor, *Intelligent Multimedia interfaces*, pp280-306. The MIT Press, 1993.
- [AuCe] Augusto Celentano and Ombretta Gaggi, "Querying and Browsing Multimedia Presentation"
<http://www.dsoi.unive.it/~auce>
- [Bufo94] J.F Koegel Buford(ed.) (1994). *Multimedia System*. Addison-Wesley, New York, New York.
- [Bord92] M. Bordegoni (1992) *Multimedia in Views*. CWI Report CS-R9263, December 1992. <http://www.cwi.nl/CWIREports/AA/CS-R9263.ps.z>
- [BuHe93] J.F Koegel and J.M.Heines (1993). *Improving Visual Programming Languages for Multimedia Authoring*, ED-MEDIA '93, World Conference on Educational Multimedia and Hypermedia, Charlottesville, Virginia, June, 286-293.
- [Blakowskis1991] G. Blakowski, J.Huebel, and U.Langrehr, "Tools for Specifying and Executing Synchronized Multimedia Presentations," *Proc. 2nd Int'l Workshop on Network and Operating System Support for Digital Audio and Video*, pp. 271-279,1991.
- [Chens2000a] Shu-Ching Chen, R. L. Kashyap, and Arif Ghafoor, "Semantic Models for Multimedia Database Searching and Browsing," Kluwer Academic Publishers, 2000.

- [Chens2000b] Shu-Ching Chen, Mei-Ling Shyu, and Naphtali Rishe, "Modeling Interactive Multimedia Presentation Systems Using Augmented Transition Networks," *First International Workshop on Intelligent Multimedia Computing and Networking (IMMCN'2000)*, pp. 643-646, February 27- March 3, 2000, Atlantic City, NJ, U.S.A.
- [Chens2000c] Shu-Ching Chen, Mei-Ling Shyu, Chengcui Zhang, and R. L. Kashyap, "Object Tracking and Augmented Transition Network for Video Indexing and Modeling," 12th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2000), pp. 428-435, November 13-15, 2000, Vancouver, British Columbia, Canada.
- [Chens2000d] Shu-Ching Chen, Srinivas Sista, Mei-Ling Shyu, and R. L. Kashyap, "An Indexing and Searching Structure for Multimedia Database Systems," IS&T/SPIE conference on Storage and Retrieval for Media Databases 2000, pp. 262-270, January 23-28, 2000, San Jose, CA, U.S.A.
- [Chens2001a] Shu-Ching Chen and R. L. Kashyap, "A Spatio-Temporal Semantic Model for Multimedia Presentations and Multimedia Database Systems," *IEEE Trans. On Knowledge and Data Engineering*, vol. 13, no. 4, pp. 607-622, July/August, 2001.
- [Chens2001b] Shu-Ching Chen, Mei-Ling Shyu, Xia Jin, Qiong Chen, Chengcui Zhang, and Jeff Strickrott, "A Flexible Image Retrieval and Multimedia Presentation Management System for Multimedia Databases," *ACM Multimedia 2001 conference*, pp. 601-602, September 30 - October 5, 2001, Ottawa, CANADA.
- [Chens1997] Shu-Ching Chen and R. L. Kashyap, "Temporal and Spatial Semantic Models for Multimedia Presentations," in *1997 International Symposium on Multimedia Information Process*, pp. 441-446, Dec. 11-13, 1997.
- [Chenq2001] Qiong Chen, Master thesis, "A Flexible Multimedia Query and Presentation Interface for Multimedia Database Systems", Florida International University, May 2001
- [Chia97] Y. Chiaramella. "Browsing and Querying: Two Complementary Approaches for Multimedia Information Retrieval". Proceeding of Hypertext – Information Trrieval – Multimedia '97, pages 9 – 26, Dortmund, WA, USA, 1997.

- [ChYA88] S. K. Chang, C. W. Yan, D.C. Dimitroff, and T.kArndt, "An Intelligent Image Database System," *IEEE Trans. On Software Engineering*, vol14, no.5, pp.681-688, May 1988.
- [CL97] I. F. Cruz, W. T. Lucas. "A Visual Approach to Multimedia Querying and Presentation". *Proceedings of the fifth ACM International Conference on Multimedia '97*, pages 109-120, Seattle, WA, USA November 9-13, 1997.
- [Clottes] J.Clottes. An Extraordinary Archaeological Find: A Decorated Paleolithic Cave in the Ardeche Region of France. <http://www.culture.fr/culture/arcnat/chauvet/en/gvpda-d.htm>
- [CLR90] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. "Introduction to Algorithms". The MIT Press, Cambridge, Massachusetts London, England
- [Cruz97] I.F. Cruz, W. T. Lucas, "A Visual Approach to Multimedia Querying and Presentation," *Proceedings of the Fifth ACM international Conference on Multimedia '97*, pages 109-120, Seattle, WA, USA November 9-13, 1997.
- [DMW] www.macromedia.com/software/dreamweaver
- [ENKY94] S.Eun, E.S,No, H.C,Kim, H. Yoon and S,R. Maeng(1994). MICROCOSM: An Open Model for Hypermedia With Dynamic Linking. In proceedings: ECHT '90(First European Conference on Hypertext), Nov, INRIA France, 298-311.
- [Erf193] R.Erfle(1993). Specification of Temporal Constraints in Multimedia Documents using HyTime. *Electronic Publishing*. 6(4), 397-411.
- [Flash] www.flash.com
- [Furuta] Furuta, R. "An Object-based Taxonomy for Abstract Structure in Document Models," *The Computer Journal*, Vol. 32, No. 6, 1989, pp494-504.
- [Gutt94] A. Guttman, "R-tree: A Dynamic Index Structure for Spatial Search," in *Proc. ACM SIGMOD*, pp. 47-57, June 1984.
- [HeKo95] J.L. Herlocker and J.A.Konstan(1995), "Commands as Media: Design and Implementation of a Command Stream," In *Proceedings: Multimedia '95*, San Francisco, CA, Nov, 155-165.

- [HiRu96] S. Hibino, E. A. Rundensteiner. "A Visual Multimedia Query Language for Temporal Analysis of Video Data" . Multimedia Database Systems: Design and Implementation Strategies, Kluwer Academic Publisher, ISBN 0-7923-9712-6, pages 123-159, 1996.
- [HiRu97] S. Hibino, E. A. Rundensteiner. "User Interface Evaluation of a Direct Manipulation Temporal Visual Query Language". Proceeding of the Fifth ACM International Conference on Multimedia '97, pages 99-107, Seattle, WA, USA November 9-13, 1997.
- [ISO97] Hytime. Hypermedia/Time-based structuring language. ISO/IEC 10744:1997
- [JMF] <http://java.sun.com/products/java-media/jmf/index.html>
- [Lo2002] Ling-Ling Lo, master thesis, "A Flexible Multimedia Presentation Environment Using JMF and SMIL", Florida International University.
- [Macr97] Macromedia. Authorware version4. director version 6. <http://www.macromedia.com>
- [Mckinlay] Mckinlay, J.D., Automomating the Design of Graphical Presentation of Relational Information, ACM Transaction on Graphics. 5, 2(April 1986), 1986, pp110-41.
- [MTW95] R. J. Miller, O. G. Tsatalos and J. H. Williams. "Integrating Hierarchical Navigation and Querying: A User Customizable Solution". In Electronic Proceedings of the ACM Working on Effective Abstractions in Multimedia. San Francisco, CA, 4 November 1995.
- [Ogawa90] Ogawa, Ryuichi, H. Harada and A. Kaneko. "Scenario based Hypermedia: A Model and a System," in A. Rizk, N. Streitz and J. Andre (eds.), *Hypertext: Concepts, Systems, and Applications*, Cambridge University Press, 1990.
- [PeDa00] Larry L. Peterson, Bruce S. Davie, "Computer networks: a systems approach" Morgan Kaufmann Publisher, 2000.
- [Pun98] Punpiti Piamsa-nga, et. al, "A Unified Model for Multimedia Retrieval by Content", ISCA 13th International Conference on Computers and Their Application (CATA98), Honolulu, Hawaii, March 25-28, 1998.

- [Ragg97] D. Raggett (1997). HTML 3.2 Reference Specification, W3C Recommendation 14-Jan-1997. <http://www.w3.org/TR/REC-html32>
- [RB01] R. Bourret (2001), "Data transfer strategies" <http://www.rpbourret.com/xml/DataTransfer.htm>.
- [Sistas 99] S. Sista and R.L. Kashyap, "Unsupervised Video Segmentation and Object Tracking," in *ICIP'99*, Janpan, 1999.
- [Woods70] Woods, W.A., "Transition Network Grammars for Natural Language Analysis", *Communications of the A.C.M.*, Vol 13, 1970, pp 591-606.
- [Woods73] Woods, W.A., "The Experimental Parsing System for Transition Network Grammars", in *Natural Language Processing*, R. Rustin (Ed.), New York, Algorithmic Press, 1973.
- [W3C] <http://www.w3.org/AudioVideo>
- [Yoshitaka 99] A. Yoshitaka and T. Ichikawa, "A Survey on Content-Based Retrieval for Multimedia Databases," *IEEE Transactions on Knowledge and Data Engineering*, vol. 11, no. 1, pp. 81-93, January/February 1999.
- [Zloof 81] M.M. Zloof, "QBE/OBE: A Language for Office and Business Automation," *Computer*, vol. 14, no. 5, pp. 13-22, 1981.