A STEPWISE REFINEMENT APPROACH TO MULTIMEDIA PRESENTATION DESIGNS

Timothy K. Shih, Chin-Hwa Kuo, C. M. Chung
H. C. Keh, Y. H. Wang, and D. R. Jiang
Dept. of Computer Science and
Information Engineering
Tamkang University
Tamsui, Taiwan 251, R.O.C.
email: TSHIH@CS.TKU.EDU.TW

Wen C. Pai
Dept. of Info. Management
Kuang Wu Institute of
Technology and Commerce
Peitou, Taipei
Taiwan, R.O.C.

ABSTRACT

We propose a stepwise refinement mechanism for the analysis and design of multimedia presentations. The mechanism is based on a revised Data Flow/Control Flow Diagram and an improved Petri net. A number of new diagram components are introduced in the multimedia DFD/CFD. The Petri net is also extended to allow interactions and conditions. Unlike mapping a traditional data flow/control flow diagram to a structured chart, the refinement of a multimedia DFD/CFD results in a number of multimedia Petri nets which represent the schedule and navigation of a presentation. The layout of the presentation can be designed by using our graphical user interface of an implemented presentation system. The proposed mechanism also allows a presentation to change itself. Thus a dynamic multimedia presentation can learn from an audience and act according to the audience's individual behavior. Algorithms of the Petri net engine are also described. Using our system, a presentation designer is able to analyze his/her presentation script and design the presentation in a systematic manner.

Key words: Multimedia Presentation, Stepwise Refinement, Data Flow Diagram, Structured Analysis, Petri Net, Software Engineering

1 INTRODUCTION

One of the most important contributions of multimedia technologies is the diversification of multimedia presentations. In the community of multimedia computing, however, there is little or no discussion of the need of a good presentation development methodology. We have surveyed a number of presentation design tools. We found that, not many authoring systems focus on the concept of stepwise refinement. In this paper, we discuss a new authoring system for this purpose. The proposed system is based on data flow and control flow diagrams, with many enhanced notations especially useful for multimedia presentation designs. Presentation designers can use our system to quickly develop a prototype presentation, and refine it step by step toward a final presentation.

A multimedia DFD/CFD of multiple levels is to help a presentation designer to analyze the script structure of a presentation. However, it is not powerful enough to define the precise schedule or layout of a presentation. Incorporated with an interactive multimedia Petri net diagramming mechanism, the last level of a presentation window is refined to a Petri net, which describes the temporal behavior of a presentation window. Our interactive multimedia Petri net is a variation of timed Petri net, with the addition of *User Transitions* and *Sync Arc*. Since a multimedia presentation is interactive, we introduce the above two objects for participant dependent synchronization.

The proposed mechanism also allows a presentation to change itself. Thus a dynamic multimedia presentation can learn from the audience and act according to the audience's individual behavior. The proposed notations and the methodology are implemented in a prototype software run on Microsoft Windows 95/3.1. In order to run a multimedia presentation, we have developed a Multimedia Abstract Machine (MAM) which is based on the timed Petri net model. Using our system, a presentation designer is able to analyze his/her presentation script and design the presentation in a systematic manner.

Before we design our system, we have surveyed both academic researches and industrial software packages. Sony Corporation developed a hypermedia prototype system (SAL) [3] for multimedia authoring, which is based on a link and node model used in most authoring systems. The Layered Multimedia Data Model (LMDM) [4] allows the reuse of presentation templates which is important for improving the efficiency of multimedia presentation designs. LMDM has a number of strengths including the support of a general model of media synchronization, limited system dependencies, and the generalization of a traditional animation model. The work dis-

0-7803-4053-1/97/$10.00 © 1997 IEEE

Authorized licensed use limited to: Tamkang University. Downloaded on March 23,2010 at 20:58:28 EDT from IEEE Xplore. Restrictions apply.

cussed in [1] proposes an architecture and data model for integrated multimedia presentations. The architecture provides a homogeneous strategy to access, process, and exchange multimedia documents generated by different authoring and presentation systems. Diamond [9] is a multimedia message system built on a distributed architecture for creating, editing, transmitting, and managing multimedia documents. Multimedia documents are stored in folders in a distributed database. An open hypermedia system is discussed in [2]. The system supports heterogeneous multimedia data types and allows new types to be added. A platform independent multimedia presentation composition system is discussed in [10]. In the system, a script based approach is used to model the object-oriented presentations as well as user interactions.

This paper is organized as the following. A short discussion of MAM is given in section 2. The proposed multimedia data flow/control flow diagram is discussed in section 3. Section 4 proposes a new direction of multimedia presentation designs. Unlike other systems, which generates static presentations, our system allows dynamic multimedia presentations. The implementation of our system is discussed in section 5. The graphical user interface of our system is given in section 6. A short conclusion summarizes our contributions is also presented in section 7.

2 A MULTIMEDIA ABSTRACT MACHINE

The main theme of our system is a timed Petri Net based Multimedia Abstract Machine (MAM). MAM is a software architecture and system for multimedia documentation demonstrations and interactions. We realized that most multimedia information processes consist of a sequence of atomic steps, such as opening a sound channel, transfering a block of video data from a buffer to the graphic memory page, closing a MIDI sequencer, etc. These atomic steps, from the perspective of multimedia computing, are non-separable items. This concept is relatively similar to assembly language instructions, which are atomic steps of a procedural program. We have the MAM instruction set defined [5]. Some instructions control multimedia hardware devices while others support user interactions. A multimedia presentation designed by using our multimedia DFD/CFD and Petri net design tool has a number of Petri nets and navigation messages. These Petri nets are represented in a MAM assembly program, which is produced by an internal program translator. Therefore, the designed presentation is runnable on the MAM.

The development of MAM has multiple purposes. We have developed a multimedia script language. The language serves as an intermediate representations of various formats of multimedia presentations which we have developed [8, 6]. These representations are compiled to the MAM instructions. Thus, it is possible to link multimedia presentations produced by different tools to make an integrated presentation. On the other side of the MAM is a multimedia DBMS [7], which we integrated to the environment. The objective of this MDBMS, from the perspective of MAM, is to assist a presentation designer to quickly locate the multimedia resources required in the demonstration.

Since MAM is an integrated multimedia programming environment with a relatively large scope, in this paper, we focus on the discussion of the multimedia DFD/CFD and Petri net design tool. Other components in the environment are omitted.

3 MULTIMEDIA DFD/CFD

In this section, we propose a revised DFD/CFD mechanism for multimedia presentation designs. The design of a multimedia presentation script has a similar concept to writing a software specification. The proposed multimedia data flow/control flow diagram is based on DFD and CFD, with the extension of some new objects:

- **Multimedia Resource**: similar to the data store in a DFD, a multimedia resource is denoted by a pair of thick parallel lines.
- **State Variable**: besides resources, an interactive presentation may contain state variables store presentation data, such as the audience's name. A state variable is represented by a pair of thin parallel lines. State variables not only keep information, but also control *dynamic presentations* (to be discussed in section 4).
- **Resource Data**: similar to the data link in a DFD, resources are passed to a presentation program by a link with a regular arrow.
- **Navigation Message (NM)**: similar to the control link in a CFD, a navigation message is passed by a link with a light-weighted arrow.
- **Dynamic Mutation**: to support dynamic multimedia presentations, the dynamic mutation link is introduced, which is represented by a curved arrow. There are four types of mutations in a multimedia DFD/CFD:
  - State Variable Change
  - Layout Change
  - Resource Change
  - Navigation Change

  These links mutate the content and appearance of a multimedia presentation. When we discuss the stepwise refinement of a presentation, we will define these mutations in detail.

Authorized licensed use limited to: Tamkang University. Downloaded on March 23,2010 at 20:58:28 EDT from IEEE Xplore. Restrictions apply.

- **External Entity (EE)**: similar to one in DFD/CFD, an external entity could represent a user, a hardware device, or another system which passes data/controls to the multimedia presentation. An EE is denoted by a box surrounded by thick lines.
- **Presentation Window (PWin)**: similar to a process in a DFD, a presentation window is denoted by a circle.

A multimedia presentation contains a number of presentation windows. Each presentation window contains a layout definition, a collection of multimedia resources (sharable among presentation windows), some state variables, and navigation control messages. A Presentation Window must be refined. Stepwise refinement of a presentation allows the presentation to be specified in different levels of details. The system allows the user to use multimedia DFDs/CFDs, with the help of a drag and drop mechanism, to design a multimedia presentation. In section 6, the graphical user interface is discussed. The system also provides a connection to a multimedia database [7]. The multimedia database resource browser allows the user to select multimedia resources needed for a presentation.

A multimedia DFD/CFD of multiple levels is to help a presentation designer to analyze the script structure of a presentation. However, it is not powerful enough to define the precise schedule or layout of a presentation. Incorporated with an interactive multimedia Petri net diagramming mechanism, the last level of a presentation window is refined to a multimedia Petri net, which describes the temporal behavior of a presentation window. Our interactive multimedia Petri net is a variation of timed Petri net, with the addition of *User Transitions* and *Sync Arcs*. Since a multimedia presentation is interactive, it is natural for us to introduce the above two types of objects for participant dependent synchronization.

A timed Petri net is a bipartite graph with two types of nodes: the transition nodes and the place nodes. A transition controls synchronization and a place holds a token and a time duration. A transition is fired only after each place adjacent to the transition releases the token. A place holds a multimedia resource to be played for the time duration. Transitions and places are connected by *sync arcs* in our revised Petri net. We add *user transitions* and *user arcs* to the timed Petri net. A user transition receives a navigation message from the user before it is fired. A user transition is directly connected to some transitions. The activation of the user transition interrupts the demonstration of an arbitrary presentation window (usually, another window) and causes the activations of the connected transitions simultaneously.

Using these user transitions and navigation messages, the revised Petri net is able to accept user interactions. The last level of a multimedia presentation refinement has a number of interactive multimedia petri nets. The followings are components of the Petri nets:

- **Transition**: is for synchronization control. Transitions are shown as vertical dark bars.
- **User Transition**: accepts a message and causes the activation of connected transitions. User transitions are shown as vertical light bars.
- **Place**: is for playing a multimedia resource. Places are shown as ellipses.
- **User Arc**: connects from a user transition to a transition. User arcs are shown as curved arrowheaded arcs.
- **Sync Arc**: connects from a place to a transition, or from a transition to a place. Sync arcs are shown as straight arrowheaded lines.

In the multimedia presentation design system, we allow six type of resources: sound, MIDI, video, animation, picture, and text resources. Some of these resource types are static (i.e., picture and text) while others are dynamic. Dynamic multimedia resources come with a resource duration. Within the time period, repeated steps are performed to load data, such as video frames, from disk to the video memory page. We defined four subnets for the six type of resources. These subnets consists of atomic execution steps, such as preparing a sound card, or writing data to sound buffer. Each of these atomic step is corresponding to an assembly instruction of the MAM. Some of the subnets (e.g., the video subnet) use transitions for synchronization control. All subnets of dynamic resources have an iterative counter (IC) calculating the duration of the resource demonstration.

3.1 Verification of Refinement

It is the responsibility of our system to check the so called flow balance of a DFD/CFD. That is, at each refinement, the number of incoming and outgoing data/control links of a bubble (i.e., a process unit or a PWin) has to match those of the next level DFD/CFD, which is a refinement of the bubble. However, in our multimedia DFD/CFD, we have three kind of links: Resource Data, Navigation Message, and Dynamic Mutation. Balance need to be maintained for each kind of link. The last level of the refinement between a presentation window and a Petri net needs to maintain two types of balances: the navigation message balance and the multimedia resource/state variable balance. That is, the incoming and outgoing messages of a presentation window have to match those of the refined Petri net. Similarly, the multimedia resources and state variables used have to be the same. The system also needs to verify the existence of multimedia resources used. If there is a Resource Data link, the corresponding resource must be declared in the Multimedia Resource data store. These multimedia resources are used in the Place object of the Petri nets.

We have discussed the global view of our diagramming mechanism. However, a multimedia presenta-

Authorized licensed use limited to: Tamkang University. Downloaded on March 23,2010 at 20:58:28 EDT from IEEE Xplore. Restrictions apply.

tion in our system is dynamic and mutable. In the following section, we propose other diagramming techniques to achieve our goal – allowing dynamic multimedia presentations.

4 DYNAMIC PRESENTATION

Usually, when a multimedia presentation is designed, the usage of resources, the layout, and the navigation sequences are all fixed until the presentation is re-designed again. We want to improve this approach by allowing dynamic replacement of resources, layouts, and controls of the presentation. This makes the presentation generator have the ability of computing the presentation representation at the run-time. Possible dynamic events are:

- asserting/retracting of information
- changing resources
- changing layouts
- changing navigation controls

On the top of the revised timed Petri net, we further add the following components:

- **Selection**: selects one of the outgoing navigation messages. A selection could be a push button or a key of the presentation window. A selection holds an internal representation of which outgoing navigation message will be sent upon the activation of the selection. A selection is represented as a circle labeled with an "S" in the Petri net diagram.
- **Assignment**: assigns a value to a state variable. An assignment is connected to a state variable. When an assignment received a navigation message (with a parameter holding a value), the assignment set the state variable to the value. An assignment is denoted by a box labeled with the "Var ¡= Val" sign.
- **Condition**: decides whether to proceed with a change. A condition is a pair of state variable and value. The value is checked against the value of the state variable. If they match each other, the associated outgoing change is fired. A condition is represented by a box with the "Var ?= val" sign.

The Petri nets of a presentation may contain a number of selections. Usually, a selection is associated with only a navigation message. However, if the selection is tight to a condition, the selection may have more than one outgoing messages. Which message to send out is decided by the condition's value. We allow an assignment statement to change the value of a state variable. State variables are used in a presentation to hold internal states. These variables will be saved on the disk when the presentation terminates (upon the user's decision), and will be loaded when the presentation starts again. A condition decides whether to proceed with a change or the propagation of a navigation message. A condition may change the execution flow of a Petri net.

The above are semantics of the newly introduced components of our Petri net. In section 6, we will discuss a graphical user interface which allows designers to specify the above components. Since each Petri net place is associated with a multimedia resource, in order to allow the designer to select and reuse a multimedia resource, we have developed a multimedia resource database. A resource contains a number of attributes. In the following subsection, we discuss these attributes. In the section of graphical user interface, we will show a multimedia resource browser.

5 IMPLEMENTATION OF THE SYSTEM

We use Microsoft *VisualC++* (VC) and *VisualBasic* (VB) to implement our system. The graphical user interface (see section 6) is implemented in VB and the presentation running engine is implemented in VC with the supports from Media Control Interface (MCI) functions of Windows 95. In this section, we present algorithms of the presentation engine.

To run a presentation, its schedule must be decided first. We want to construct a *transition graph* from the Petri net. Each place is associated with a number which represents the duration of presenting a multimedia resource. Each transition has some incoming and outgoing arcs. We want to construct a transition graph such that the weighted edges represent places with durations, and nodes represent transitions. From this transition graph, the schedule of presentation can be computed. However, the activation of a Petri net is due to the receiving of a navigation message. Therefore, a transition graph constructed w.r.t. the navigation message contains only those transitions (or nodes) reachable from the navigation message.

From the transition graph, the algorithm initials the node pointed by the message to zero. And, the nodes of the graph are sorted according to their topological order. Finally, the node time of each node is set to the maximum estimated time of all incoming edges of the node. An estimated time is computed from the node time of the source node and the duration of a multimedia resource.

The presentation engine is a concurrent program, which is simulated by the multithread/multiprocess facility of Windows 95. The first process plays resources in the sorted transition list. The second process take cares of interrupt events, such as the receiving of another navigation message.

Authorized licensed use limited to: Tamkang University. Downloaded on March 23,2010 at 20:58:28 EDT from IEEE Xplore. Restrictions apply.

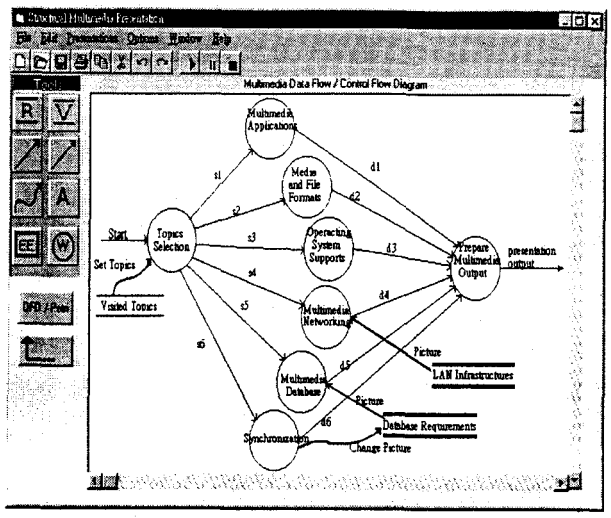

Figure 1: A Structured Multimedia Presentation Design Tool

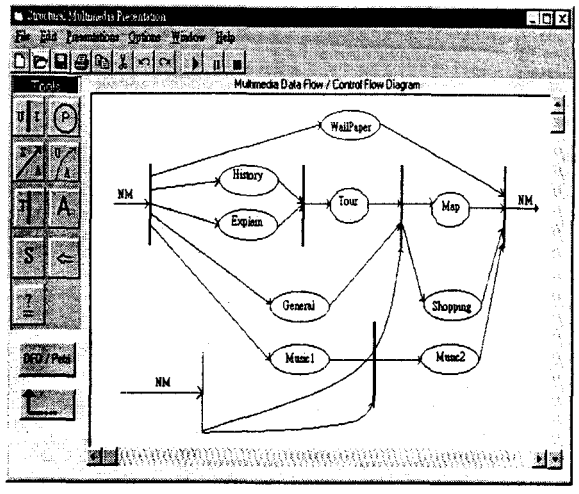

Figure 2: A Multimedia Petri Net Tool

Figure 3: The Layout Design Window

6 THE USER INTERFACE

In this section, we present the graphical user interface of our proposed multimedia presentation system. Figure 1 shows the main window, which has a tool box containing some ICONs. The ICONs in the left column allows the designer to create: Multimedia Resource, Resource Data link, Dynamic Mutation, and External Entity respectively. The ICONs in the right column are to create: State Variable, Navigation Message, text label, and Presentation Window. We also have two push buttons below the tool box allowing the user to choose a DFD/CFD or Petri net diagram, and to navigate to an upper level of the diagram. A presentation designer has to use a drag and drop mechanism, similar to that of a standard windowing system, to create, insert, or delete diagram components. Clicking on a Presentation Window circle brings up the next level refinement of that window. Clicking on other objects results in other design windows. We have text and selection windows allowing the editing of individual type of diagram components. These windows contain text boxes and push buttons which allow the user to enter specific information of each component.

In figure 2, the tool box on the left of the main window is changed for Petri net diagram designs. ICONs in the left column are to create: User Transition, Sync Arc, Transition, Selection, and Condition. ICONs in the right columns are for: Place, User arc, text label, and Assignment. The mechanism to create the diagram is similar to those in figure 1. We also have a navigation message window. One or more messages with optional parameters are entered by the user.

The layout design window shown in figure 3 is a little complicate. The ruler below a number of place names indicate that the presentation is divided into five states. In a multimedia presentation, the start or the end point of a multimedia resource makes a state change. The change of state usually changes the presentation layout. While in a state, the layout is fixed. Clicking on a transition or user transition brings up the presentation layout starting from the transition. For example, clicking on the first transition in figure 2 results in the layout design window in figure 3. Presentation states are deduced from the propagation starting from a transition. Each presentation state is associated with a number of places (above the ruler). Clicking on a section of the ruler (representing the state) allows the designer to alter the location or size of objects of the state. However, Sound and MIDI objects does not have layout. They are displayed as ICONs below the layout area. Each state in the ruler is given a number indicate the number of executing cycles of the state.

In figure 4, we show a multimedia resource browser.

Authorized licensed use limited to: Tamkang University. Downloaded on March 23,2010 at 20:58:28 EDT from IEEE Xplore. Restrictions apply.

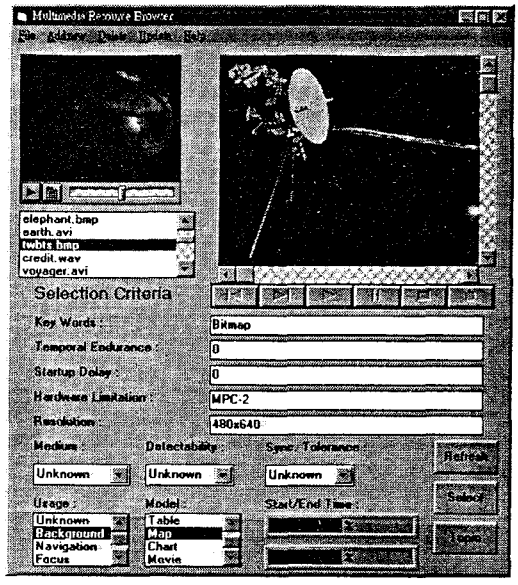

Figure 4: The Multimedia Resource Browser

The browser is to help the designer to select resources needed in a presentation [7].

7 CONCLUSIONS

We spent about six months in the design of our mechanism and system interface. The development of algorithms and implementation took another six months before the system was tested for about three months. The preliminary experiences show that, the proposed multimedia DFD/CFD is easy to learn since data flow diagrams have been used by many engineers and managers for two decades. The concept is easily adapted. However, using Petri net to design multimedia presentations is not as easy to learn due to the new diagram components introduced.

We have some contributions in this paper. Firstly, we revised structured analysis/design methodology and data flow/control flow diagrams for the use of multimedia presentation designs. Next, we proposed an interactive multimedia Petri net for dynamic multimedia presentations. The new Petri net allows not only interactions but also selections and conditions. Finally, a presentation system was developed on MS Windows 95. to justify our approach. With this system, we hope to bring the spirit of structured analysis/design methodology to the design of multimedia presentations. The system can be used for general-purposed presentations as well as education software, demonstrations, and others.

References

[1] H. Khalfallah and A. Karmouch. "An architecture and a data model for integrated multimedia documents and presentational applications". *Multimedia Systems, Springer-Verlag*, 3:238–250, 1995.

[2] T. Kirsta and W. Hubner. "An Open Hypermedia System for Multimedia Applications". In *Eurographic Seminars, Tutorials and Perspectives in Computer Graphics, L. Kjelldahl(ED.) Multimedia Systems,Interaction and Applications Chapter 17*, chapter 17. 1991.

[3] A. Lundeberg, T. Yamamoto, and T. Usuki. "SAL, A Hypermedia Prototype System". In *Eurographic Seminars, Tutorials and Perspectives in Computer Graphics, L. Kjelldahl(ED.) Multimedia Systems,Interaction and Applications Chapter 10*, chapter 10. 1991.

[4] G. A. Schloss and M. J. Wynblatt. "Presentation Layer Primitives for the Layered Multimedia Data Model". In *Proceedings of the IEEE 1995 International Conference on Multimedia Computing and Systems, May 15-18, Washington DC*, pages 231–238, 1995.

[5] T. K. Shih. "Multimedia Abstract Machine". In *Proceedings of the Third Joint Conference on Information Science*, 1997.

[6] T. K. Shih and R. E. Davis. "IMMPS: A Multimedia Presentation Design System". *IEEE Multimedia*, Summer, 1997.

[7] T. K. Shih, C.-H. Kuo, and K.-S. An. "An Object-Oriented Database for Intelligent Multimedia Presentations". In *Proceedings of the IEEE International Conference on System, Man, and Cybernetics Information, Intelligence and Systems Conference*, 1996.

[8] T. K. Shih, C.-H. Kuo, and H.-J. Lin. "Visual Multimedia Presentation Designs". In *Proceedings of the 1996 Pacific Workshop on Distributed Multimedia Systems*, pages 306 – 315, 1996.

[9] R. H. Thomas and e. a. Harry C. Forsdick. "Diamond : A Multimedia Message System Built on a Distributed Architecture". *IEEE Computer*, December:65–78, 1985.

[10] M. Vazirgiannis and C. Mourlas. "An Object-Oriented Model for Interactive Multimedia Presentations". *The Computer Journal*, 36(1), 1993.

Authorized licensed use limited to: Tamkang University. Downloaded on March 23,2010 at 20:58:28 EDT from IEEE Xplore. Restrictions apply.