DOCUMENT RESUME

ED 388 292 IR 017 441

AUTHOR	Schloss, Gerhard A.; Wynblatt, Michael J.
TITLE	Using a Layered Multimedia Model To Build Educational Applications.
PUB DATE	94
NOTE	7p.; In: Educational Multimedia and Hypermedia, 1994. Proceedings of ED-MEDIA 94--World Conference on Educational Multimedia and Hypermedia (Vancouver, British Columbia, Canada, June 25-30, 1994); see IR 017 359.
PUB TYPE	Reports - Descriptive (141) -- Speeches/Conference Papers (150)
EDRS PRICE	MF01/PC01 Plus Postage.
DESCRIPTORS	*Computer Assisted Instruction; Computer Interfaces; *Computer Software Development; Databases; Design Requirements; Intelligent Tutoring Systems; Interaction; Models; *Multimedia Materials; Programming; Teaching Styles; User Needs (Information)

ABSTRACT

As new users from the non-technical community are attracted to multimedia (MM) computing, it becomes necessary to design a convenient and easy-to-understand logical structure that facilitates data transfer from the MM data repository to MM applications. A recently introduced layered formalism, which describes MM applications in terms of the data that they employ, provides a structure that interfaces MM data with MM authoring. Such a formalism together with a database of MM data objects is a convenient development environment for the creation of a variety of reusable and portable educational multimedia. The Layered Multimedia Dat Model (LMDM) consists of four layers: the data definition layer, the data manipulation layer, the data presentation layer, and the control layer. As many MM compositions have interactive components, the LMDM supports the capability to express the effects of user-interaction. The layered structure of the LMDM allows the developer to pick and choose, by tailoring his or her own development system through personalized selection of tools for each of the layers. Moreover, the extensive browsing and score-keeping features greatly expand the number of different teaching styles and teaching strategies that can be implemented using LMDM. Tutorials can be developed in LMDM, that not only use the student's score to determine areas that need review, but other tutoring programs can check to see what material has been mastered and what requires more study. (AEF)

ED 388 292

Using a Layered Multimedia Model to Build Educational Applications *

GERHARD A. SCHLOSS MICHAEL J. WYNBLATT
Computer Science Department
SUNY at Stony Brook, Stony Brook, NY 11794
email: {schloss, wynblatt}@cs.sunysb.edu

Abstract
Currently, educational multimedia (MM) products are prepared in an ad hoc fashion, i.e. the different media objects, when needed, are produced by the MM author. In the future, as more digitised MM data becomes available, building multimedia compositions will likely involve searching through databases of existing images, sounds, and video clips. Where today a MM application is typically associated with a specific hardware platform, future MM developers will author MM products which are reusable across a variety of platforms. A recently introduced layered formalism, which describes MM applications in terms of the data that they employ, provides a structure that interfaces MM data with MM authoring. Such a formalism together with a database of MM data objects, is a convenient development environment for creation of a variety of reusable and portable educational multimedia.

1 Introduction and Background

Two problems faced by the majority of today's multimedia (MM) users point to the fact that future MM application development is likely to become a data-driven process, relying heavily on shared MM data repositories as the main resource for MM authoring. First, multimedia data is storage intensive. Second, the production of high quality audio, video and animation is an expensive and time consuming process, which few potential MM authors will have the skills or resources to master. Hence, the concept of several users sharing access to large MM databases will alleviate both of these problems by distributing storage, and production or purchase costs among several individuals or institutions.

Of all the new digital technologies, multimedia computing seems to be the one with the greatest potential to create interest and attract new users from Social Sciences, the Arts and Humanities. Their involvement and contributions can significantly increase the penetration of computing into the traditionally technology-averse parts our society. However, fewer people in the non-technical community are skilled in programming, and their most important interaction with multimedia computing will likely occur at the interface level [Ward, P. S. and Arshad, F. N., 1991]. Hence the importance of designing a convenient and easy-to-understand logical structure, one that facilitates data transfer from the MM data repository to MM applications.

Recently, we proposed and described a new modeling approach to MM data management and MM application development that follows the *layered architecture* paradigm [Schloss, G. A. and Wynblatt, M. J., 1993]. There are two major reasons that prompted the development of the LMDM model. First, many characteristics of MM data make it uniquely different from the traditional alpha-numeric data. In particular, MM data may have semantics that are only relevant with regard to its presentation. For instance, a clearly sarcastic remark in a sound bite, may not convey its true meaning in text representation.

Second, the existing data models generally do not support *multimedia development* because their prime focus is the data itself. In traditional data models, presentation is usually irrelevant. Indeed, a record with a person's address may be displayed in a large box or a small one, in the Helvetica font or the Times; the meaning of the address is still the same. Compare that with a color photograph which conveys some message that would be obscured by a monochrome display. Hence, our objectives in developing the LMDM were that

*This work was partially supported by the National Science Foundation under Grant No. CDA-9214942.

the new model: (a) be application-independent; (b) allow conceptual separation between the data itself its presentation; (c) provide an interface to a shared MM database; and (d) support a wide variety of MM applications.

At a time when multiple-platform computing environments are becoming the rule rather than the exception, MM applications are among the least portable [Woelk, D. et al., 1986]. This is due in part to their dependence on specific hardware for presentation. LMDM's layered architecture allows description of complex data structures within applications, independent of the particular hardware specifications.

We provide below a concise description of the LMDM, and then discuss its potential as a toolkit for developing flexible, adaptable, reusable and portable educational materials, which utilize interactive multimedia compositions.

2 The Layered Multimedia Data Model (LMDM)

LMDM follows the layered modeling paradigm. Such an approach, in addition clearly separating data management, presentation management, and application development, has several other advantages. First, hardware- and OS-specific concerns can be isolated in particular layers, resulting in a more general model. Second, the implementation of the functionality of different layers can progress independently, so that the advances in one area can quickly be added to existing strengths in others. Finally, a layered paradigm is a clear favorite with those who perceive multimedia as a new communication and information delivery system. A layered MM model was previously proposed in [Klas, W. et al., 1990], though the choice of layers was somewhat different.

The LMDM model consists of four layers, see Figure 1. It places particular emphasis on expressiveness for continuous media data objects like video, animation, and audio, allowing them to be sequenced, synchronized or related temporally in other ways. We provide below brief definitions of the four layers, their functionality, as well as of the basic terminology used in the LMDM.

2.1 Data Definition Layer

The Data Definition Layer (DDL) allows data specification of a *MM data object*, either explicitly or through some reference to the data, and it provides this abstraction to the higher layers. A data object consists of a definition statement which specifies the data itself, and also a data type which describes properties of the data. The DDL provides: (1) a language in which the definition statement may be made formally; (2) a set of types which can be used to describe the retrieved data; (3) a tagging mechanism, through which tags containing semantic information may be attached to data objects to improve data accessibility; and (4) an accounting or charge-back mechanism, through which MM data objects containing information subject to copyright or royalty agreements may be accessed.

Data types convey semantic information. In addition to data types such as audio clip or text passage, more specific data types like `music` or `movie` can be defined. In order to facilitate description of such data types, the DDL supports subtyping and attribute inheritance. New media can be introduced by creating a new data type to represent the media and providing operators to act on objects of the new data type. The DDL also provides an *object scripting* mechanism which allows filtering, enhancing, altering, and comparing data objects, as well as grouping data objects into arrays, without modifying the data itself.

2.2 Data Manipulation Layer

The Data Manipulation Layer (DML) allows data to be combined into more complex constructs called *MM events*. MM events are MM objects, or groups of MM objects, which share the same abstract event clock, or event time reference. The DML provides operators and an algebra in which these transformations can be described formally. It contains a symbolic language for formal description of complex manipulations. Hence, simple MM events can be combined to form complex temporal structures.

Data objects that share a time reference can be related to each other through temporal relationships [Hoepner, P., 1992], which can be used for sequencing (before, after) or synchronization (starts, finishes). Synchronization points may be described, to allow arbitrary strictness of synchronization. Objects which share a time reference are said to be *temporally bound* to each other.

Control Layer	$\Longrightarrow$	MM Composition
Data Presentation Layer	$\Longrightarrow$	MM Presentation
Data Manipulation Layer	$\Longrightarrow$	MM Event
Data Definition Layer	$\Longrightarrow$	MM Object

Figure 1: The Layered Multimedia Data Model (LMDM).

2.3 Data Presentation Layer

The Data Presentation Layer (DPL) provides a description of how data is to be communicated to the user. The DPL uses MM events as its building blocks, adding information about layout, output format, presentation dependencies between events, user interface elements such as windows or icons, as well as default values for playback parameters such as colormap and speaker volume. The resulting *MM presentation* is a set of instructions for communicating MM data to a viewer.

The DPL maintains a library of specifications for the various playback devices, spatial layouts of the data on the screen, windows or other display methods, colormaps, text fonts and styles, as well as descriptions of presentation dependencies between frequently used MM event templates. The DPL also includes the logic necessary to describe the interaction of simultaneously occurring events, e.g. opacity and color of object overlays (alpha channel), and volume adjustment of multiple sound tracks.

In the same way that a MM event may have multiple presentations associated with it, a single MM presentation may contain several different events, or may be reused with a different set of events. This allows data to be presented in an alternative manner if certain hardware or presentation parameters change. Moreover, generic presentations may be developed which can present several similarly structured events in the same way.

2.4 Control Layer

A *MM composition* is defined as one or more MM presentations which are grouped with a common control structure and user-interface. Hence, the Control Layer (CL) describes how such compositions are built from one or more presentations. It contains ordering instructions for the various MM presentations, a navigation mechanism through which the user interacts with each presentation, and conditions for starting and stopping the different presentations that make up a MM composition. The CL determines whether these presentations are logically affiliated, and how they interact with each other and the user.

The CL provides a language which describes what signals the MM composition can accept from the user or from I/O devices, and what actions must be taken upon receiving these signals. These actions might include stopping or starting a presentation, or changing the presentation mode in one or more of the presentations. In addition to describing the reaction to signals, the CL language provides constructs for 'hard-coded' sequencing and looping of presentations, as well as for conditional presentation based upon the values of any available input.

3 LMDM and Interactive Learning

In [Pea, R. D., 1991], a clear distinction is made between the *chained multimedia* [sic] and *integrated multimedia* in education, where the former is attributed to the traditional audio-visual aids, e.g. interruptible slide shows or educational films. We feel that a more computer-oriented terminology may be in order. Hence, we distinguish between: (a) *computer-controlled multimedia* (CCMM); and (b) *computer-integrated multimedia* (CIMM). With CCMM, a MM presentation or experiment may be centrally controlled by a computer; however, the participating media need not be digitized. An example of a system that operates in the CCMM mode is a MM classroom [Skill Dynamics, 1993], where a mix of analog and digital devices can be interconnected and slaved to a computer. Thus, in CCMM the range of possibilities in both data integration as well as user interaction, are strictly limited. In contrast, CIMM implies fully-digitized media, with a complete integration of the computer in *all* aspects of media production, orchestration, storage, retrieval, manipulation, and delivery to the end-user. Hence, the CIMM opens unlimited possibilities to both the

MM materials developer and MM consumer, as any imaginable mix of data, as well as non-linear access and viewing, can now be supported. Since large-scale CIMM systems are not yet technologically feasible, a *MM workstation* equipped with media-editing and MM authoring software is the current surrogate for the future CIMM system.

The specifications that led to the development of the LMDM model, are clearly intended as a framework that would efficiently describe and support computer-integrated multimedia. As many MM compositions have interactive components, the LMDM supports the capability to express the effects of user-interaction. Educational MM applications often require support for sophisticated hypermedia-style browsing and navigation techniques [Ward, P. S. and Arshad, F. N., 1991], which are necessary for self-paced access to learning materials. In LMDM, the users can mark their current position within the application for future return (a technique similar to a *bookmark*), or may designate a collection of bookmarks within the application as a walk-through path that can be stored and repeated at will. This second technique is often called a *trail* [Lipton, R., 1992]. Finally, the support for persistent storage provided within the LMDM makes tools for instructional chores such as testing, quizzing, and score keeping, easy to create and maintain.

The layered structure of the LMDM allows the developer to pick and choose, by tailoring her own development system through personalized selection of tools for each of the LMDM layers. Moreover, the extensive browsing and other score-keeping features, which can be supported with relatively little effort within the LMDM, greatly expand the number of different teaching styles and teaching strategies that can be implemented utilizing LMDM. Likewise, the amount of feedback given to students during instruction is completely flexible, and it can be adjusted as a function of their progress or achievements. Thus, LMDM avoids some of the weaknesses observed in the older computer-based training (CBT) systems, such as HITS [Barker, J. and Tucker, R. N., 1990].

The LMDM allows fast prototyping of many different instructional paradigms. In the following example, we demonstrate how a reusable structure of a tutorial can be built with the help of LMDM. Space limitations prevent us from making the example too complicated.

4 Example: Building a Tutorial with LMDM

Suppose that the objective is to develop a tutorial designed to teach the difference between two things. It might be used to teach the difference between kinds of architecture, or styles of painting, or different makes of car, or any similar distinction. The method of instruction relies on a slide show paradigm accompanied by appropriate narration and text. The student is to acquire the knowledge provided by the tutorial, and be tested before she is allowed to proceed to the next subject. Hypothetical specification examples are provided for each layer.

Data Definition Layer:

At the DDL, we define a MM object called a *lesson*. A lesson is an aggregate [Woelk, D. et al., 1986] and conceptual [Klas, W. et al., 1990] object, which links all the pieces of data that are needed as part of a particular tutoring session. Lesson objects may be custom designed for particular tutorials, but their real power can be shown through *lesson templates*, which allow tutorials covering similar materials to be developed rapidly. A *template* of an object is defined as the meta-data that describes the object structure without explicitly stating the data (contents). When initialized (or, linked) with specific data, a template becomes an instance of the object.

Consider the following object template, called a `SlideShowLesson`. This template exists at the data definition layer, and describes the *types* of data involved in the lesson, without actually describing the data. In this particular example, the `SlideShowLesson` consists of two sets of photographs, and a series of narrations describing the photographs. It also contains information about the preferred colormap and resolution of the photos and the recording rate of the audio narrations. Finally, it contains a link to a *score* object, in which the results of the lesson will be kept. In this example, the score object is defined as generic text.

```
Template SlideShowLesson:
     contains
          photo1: array [1..M] of photos
          photo2: array [1..M] of photos
```

```
        narration: array [1..N] of audioclips
        score: text passage;
PreferredCmap is worse(photo1.preferredCmap, photo2.preferredCmap);
PreferredRes is worse(photo1.preferredRes, photo2.preferredRes);
PreferredRate is narration.RecordingRate
```

Data Manipulation Layer:

Based on this template defined in the DDL, a full composition can be built in the higher layers, without requiring precise knowledge of what the actual content of the photos and narrations will be. At the manipulation layer, photos and narrations have to be synchronized and sequenced.

In this example, narrations are synchronized with the first N pairs of photos. These narrated photos provide the explanatory section of the composition, in which the narrator explains a difference in style by referring to pairs of photos which exhibit that difference. After the explanatory section, there is series of unnarrated photos which are used as a quiz section. The manipulation layer also describes an operator Scoring, which accepts a string and updates the data in the score object.

```
Event SlideShow is
        Sequence((for I from 1 to N
                        TempEqual(narration[I],photo1[I],photo2[I]))
                  for I from N+1 to M
                        either UntilRestart(photo1[I]) or
                               UntilRestart(photo2[I])
                 )
Operator Scoring(answer:String, score:TextPassage)
        /* ... definition ... */
```

Data Presentation Layer:

The DPL describes the window in which the presentation will take place, and mentions that the preferred colormaps and resolutions will be used. The presentation has two modes, one for the explanatory section, which has a Next button, and one for the quiz section, which has Style1 and Style2 buttons. The functions of these buttons are specified at the control layer.

```
Presentation #1
        is SlideShow
        using  PlayAudio(narration)
                    with SlideShowLesson.PreferredRate
        and ShowImage(photo1) and ShowImage(photo2)
                    in 10"x10" SplitWindow
                    with SlideShowLesson.PreferredCmap
                         SlideShowLesson.PreferredRes
        Mode #1 has PushButton "Next" at (150,30);
        Mode #2 has PushButton "Style1" called photo1.stylename
                                        at (150,10)
                and PushButton "Style2" called photo2.stylename
                                        at (150,50);
```

Control Layer:

The control layer describes the control sequence. The presentation moves forward for the duration of an audio sequence, and then pauses until the Next button is pressed. After the first N slides, the mode is changed from Mode #1 to Mode #2. Now the student advances the presentation by selecting one of the two Style buttons to indicate which of the two styles she believes this photo represents. Her choice is recorded in the score object using the Scoring function provided at the manipulation layer. Note that no bookmarks or trails need to be defined, because in LMDM these services are globally available.

```
Signals
        Signal Next from Button "Next";
        Signal Style1 from Button "Style1";
        Signal Style2 from Button "Style2";
        Signal AudioEnd from Method PlayAudio;
```

```
Control Script
        for I from 1 to N
                DO Presentation#1(Mode#1) UNTIL AudioEnd;
                WAIT UNTIL Next;
        for I from N+1 to M
                DO Presentation#1(Mode#2) UNTIL Style1 or Style2;
                Scoring(LastSignal,score);
```

In this example, we have designed a complete interactive multimedia tutorial, without actually specifying the subject of the lessons. In order to use the `SlideShowLesson`, an instructor need simply provide two sets of photos and record a few narrations. The work of putting together the synchronization and control, as well as designing the screen layout and user-interface, is a one time cost, and can be re-used with many different tutorials.

Another advantage of this example is that the score data is persistent, and can be linked to the material of the lesson (the photos) at the definition layer. Not only can the current tutorial use the score to determine what areas need review, but other, independent, tutorials can draw upon this information as well. Other tutoring programs can check to see what material the student has mastered, and what requires more study. In this way, programs can be designed to tailor themselves to the needs of individual students. The key is that progress of the student is stored in a way which is independent of the program being used. Any program can check on the results of previous programs and use them to customize a session for the student.

5 Summary

The major advantage of the new *Layered Multimedia Data Model* (LMDM) is its ability to address within a single modeling framework two issues that make today's multimedia computing difficult: (1) interface between the data and application development; and (2) data sharing and application portability. A clean and efficient modeling paradigm, LMDM conveniently integrates MM data with MM authoring into a unified applications development environment. As a system that follows the computer-integrated multimedia (CIMM) mode, LMDM can be helpful in development of sophisticated, flexible, portable and reusable educational multimedia products. It provides extended capabilities for various interaction and teaching styles, scoring and record keeping, persistent navigation, for student performance evaluation, etc.

References

[Barker, J. and Tucker, R. N., 1990] Barker, J. and Tucker, R. N., editors (1990). *The Interactive Learning Revolution: Multimedia in Education and Training.* Kogan Page, Ltd. (UK).

[Hoepner, P., 1992] Hoepner, P. (1992). Synchronizing the presentation of multimedia objects. *Computer Communications*, pages 557–564.

[Klas, W. et al., 1990] Klas, W., Neuhold, E., and Schrefl, M. (1990). Using an object oriented approach to model multimedia data. *Computer Communications*, pages 204–216.

[Lipton, R., 1992] Lipton, R. (1992). *Multimedia Toolkit: Build Your Own Solutions with DocuSource*. Random House Electronic Publishing.

[Pea, R. D., 1991] Pea, R. D. (1991). Learning through multimedia. *IEEE Computer Graphics and Applications*, pages 58–66.

[Schloss, G. A. and Wynblatt, M. J., 1993] Schloss, G. A. and Wynblatt, M. J. (1993). A layered model for multimedia data. In *Proceedings of the 3-rd Int. Workshop on Information Technologies and Systems (WITS-93)*, pages 142–151.

[Skill Dynamics, 1993] Skill Dynamics (1993). Introducing the IBM interactive multimedia classroom. Videotape LV31-3567-00, Skill Dymnamics, An IBM Company.

[Ward, P. S. and Arshad, F. N., 1991] Ward, P. S. and Arshad, F. N. (1991). Interactive multimedia information systems for education. In *Proceedings of the 1-st Eurographics Workshop on Multimedia*, pages 337–341.

[Woelk, D. et al., 1986] Woelk, D., Kim, W., and Luther, W. (1986). An object-oriented approach to multimedia databases. In *Proceedings of the ACM-SIGMOD Conference on Management of Data*, pages 311–325.