

A System Architecture for Flexible, Knowledge-Based, Multimedia CBT-Applications*

FREIMUT BODENDORF
KLAUS LANGER

University of Erlangen-Nuernberg, Department of Information Systems, Lange Gasse 20,
90403 Nuernberg, Germany

One objective of the system architecture described in this paper is to increase the reusability of multimedia resources. For modern CBT applications, designing and creating multimedia resources such as text, graphics, pictures, animations, audio and video sequences requires a considerable amount of time and effort. The project covers the development of methods and tools for authors. These tools enable authors to re-configure existing modules for different educational settings, e.g. teaching basics, brush-up and training, and user classes, e.g. subject experts and novices. Besides reusability, increasing flexibility to adapt to each user's preferences, motivation and experiences is another objective. The architecture is based on a model combining elements of hypermedia with those of semantic networks. Hypermedia concepts use and link small multimedia teaching modules, whereas semantic networks provide an interpretable description of these modules. For each user, a specific course is dynamically created at run-time, using a run-time controller developed within the project. This controller contains a fuzzy logic component for representing pedagogical knowledge by fuzzy rules and artificial neural networks. The latter is included to represent experiences and decisions of former users (other learners or even authors) for their successors. In order to adapt learning paths and styles to individual users' needs (not only for user classes), a user model with static and dynamic data is integrated. For considering a broad variety of learning styles, learner control is kept variable within the range of free navigation and strict system guidance.

INTRODUCTION

RAPID growth of information and qualification demands have made learning a life-long process, especially in a workplace determined by technological issues. New educational needs are covered by introducing computer supported methods (CBT). During the last decade, CBT has been improved by two different methodological directions. First, in the field of intelligent tutoring systems (ITS) results and methods of artificial intelligence are used. ITS try to represent subject knowledge (in most cases domain dependent) to simulate an ideal human teacher's performance, especially concentrating on sophisticated diagnosing techniques. Second, so-called hypersystems are forcing explorative learning. Hypersystems are based more and more on multimedia elements (hypermedia systems). Technological progress and dropping prices for multimedia equipment promote this development, backed by stimulating findings of pedagogical media research. Compared with conventional text-based ones, the expenditure for multimedia courses is rising considerably.

Increasing reusability of multimedia resources for CBT applications and providing user individual flexibility for courses were the primary objectives

of the research described by this paper. The system architecture that has been developed integrates a multimedia database system and hybrid knowledge-based methods (fuzzy logic, semantic networks, artificial neural networks), trying to merge conceptual elements of ITS and hypermedia systems.

OVERVIEW AND OBJECTIVES

Within the project, a concept for a system architecture and appropriate tools for multimedia CBT applications are developed. From a methodological point of view, maximum domain independence is intended. The architecture provides authors and students with services for different areas of application, e.g. operator training and basics of operating systems. Figure 1 presents an overview of the system, both from the author's and student's point of view. As seen from the student, multimedia objects constitute the smallest elements of information. They are created and modified by authors using standard tools (graphics packages, audio and video editors, etc.).

Following [1], they are already complex multimedia objects, being aggregations of so-called elementary media objects. However, elementary media objects consist of just one distinct media

* Accepted 23 May 1996.

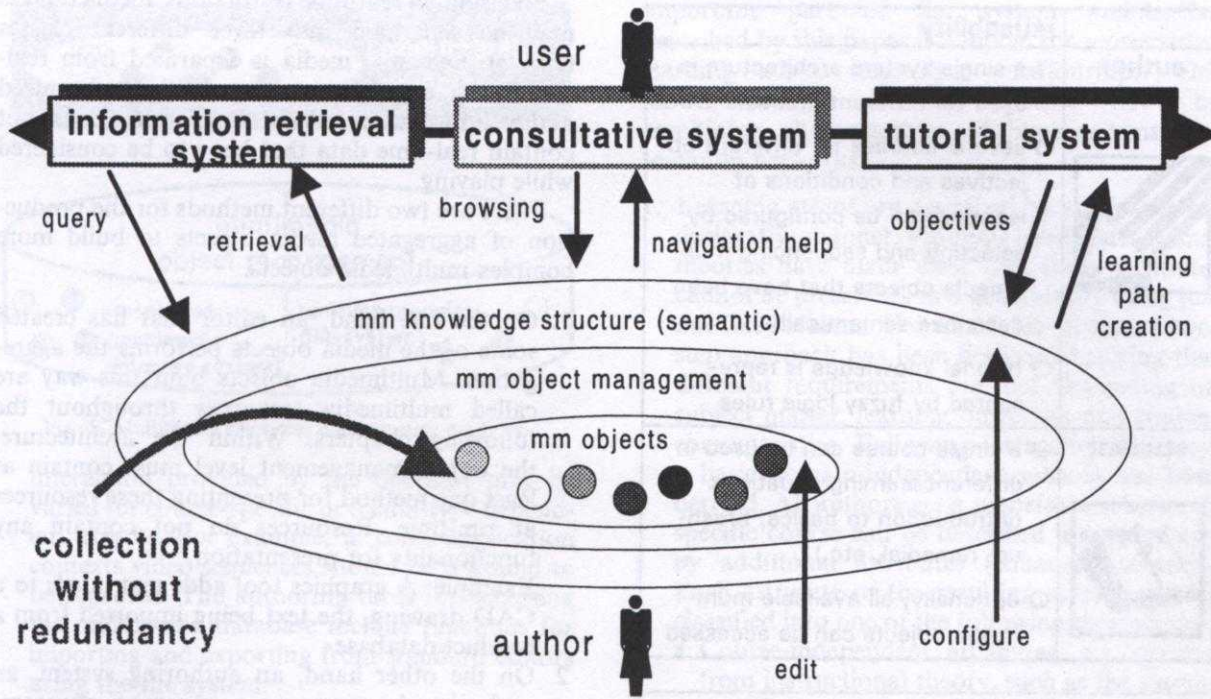


Fig. 1. Overview of the system.

type (text, graphic, picture, animation, audio, video). Multimedia objects are built by combining media objects along local or time dimensions, e.g. labelling graphics with text including speech annotations. Thus, a distinct media type is lost. During the early stages of a CBT project, authors should take non-redundant multimedia objects into consideration. A set of multimedia objects with minimal redundancy promises best conditions for high-level reusability.

The multimedia object management level (see Fig. 1) provides a standardized interface for functional operators like presentation, modification and retrieval. This is necessary because of the decision to use existing heterogeneous editors and authoring tools for each single multimedia object. This also ensures designing an architecture which integrates effective available tools.

On the third level, multimedia knowledge structures provide a semantic description of all accessible multimedia objects. This semantic information enables the run-time controller to create an appropriate learning path. A sequence of multimedia objects is built that fits best in a given situation of learning for a specific student.

From the student's point of view, the architecture offers the possibility to vary the level of learner control, i.e. presenting either a pure tutorial system, a consultative system or an information retrieval system (see Fig. 1).

Tutorial system

The student specifies preferences (subject matter, learning style, media selection) and the run-time controller builds a suitable learning path, selecting and presenting appropriate multimedia objects in sequence. The system determines

dynamically which multimedia object is to be presented next.

Consultative system

The student's preferences are used for selecting a partial set of appropriate multimedia objects for the next learning step. These objects are presented to the student for free selection (navigational help).

Information retrieval system

In this mode, the student may select one of all available multimedia objects. In contrast with the consultative mode, the system does not pre-select a subset.

Regarding authors' and students' point of view, both reusability and flexibility as primary objectives are outlined in more detail in Fig. 2. The primary objectives are interdependent. Reusability increases flexibility, because existing multimedia objects of one course are selected and configured to form a new one for different learning conditions. Vice versa, flexibility supports reusability, since adapting to an individual user's needs appeals to a larger community of potential students.

Both of the objectives are pursued to minimize the expenditure for modern, multimedia CBT applications. Considering the state-of-the-art in CBT development, most courses, with some exceptions in academic research projects, need 'reprogramming' to be adapted to different learning objectives, subject matter and users.

MULTIMEDIA OBJECTS FOR BUILDING A KNOWLEDGE BASE

Since the architecture is intended to form an open system, tools for creating and editing media





reusability	
author 	<input type="radio"/> a single system architecture is used for different domains <input type="radio"/> several courses for different objectives and conditions of learning can be configured by selecting and sequencing multimedia objects that have been described semantically <input type="radio"/> tutorial knowledge is represented by fuzzy logic rules
student 	<input type="radio"/> a single course can be used in different learning situations (introduction to basics, brush-up, remedial, etc.) <input type="radio"/> optionally, all available multimedia objects can be accessed
flexibility	
author 	<input type="radio"/> users are grouped into classes and characterised by attributes <input type="radio"/> authors specify different learning paths for different user classes <input type="radio"/> individual user characteristics are used by predicates of fuzzy logic rules
student 	<input type="radio"/> learning paths are created dynamically for individual users <input type="radio"/> user preferences are taken into consideration (learning style, media selection) <input type="radio"/> decisions of former users about the learning path contribute to the creation of the current student's path by involving artificial neural networks

Fig. 2. Objectives of the system.

objects that are compatible with the architecture's GUI have been integrated. Many of these tools provide sophisticated functions for each specific media type and, moreover, are often widely distributed. From an author's point of view, media objects can be divided into several categories:

- (formatted) text
- graphics (based on vectors)
- pictures (based on pixels)
- animations
- audio sequences
- video sequences

Focusing on real-time restrictions, media objects may be separated into three different classes (Fig. 3). Dynamic media is separated from real-time media, e.g. video sequences with synchronized audio information, since the former does not contain real-time data that have to be considered while playing.

There are two different methods for the production of aggregated media objects to build more complex multimedia objects.

1. On the one hand, an editor that has created some of the media objects performs the aggregation. Multimedia objects built this way are called multimedia resources throughout the following chapters. Within the architecture, the object management level must contain at least one method for presenting these resources at run-time. Resources do not contain any functionality for presentation.
Example: A graphics tool adds text labels to a CAD drawing, the text being imported from a product database.
2. On the other hand, an authoring system, an authoring language or a common programming language is used for implementing small 'code' modules, which present several media objects, thus aggregating these objects indirectly. The modules may also provide well-known features of CBT applications (analysis of student answers, elements of simulation and microworlds, etc.). In terms of operating systems, multimedia objects built this way are tasks. In the following, they are called run-time modules, since they are executable, either directly or via interpreters.

One of the most important tasks of the object management is to provide an operator for 'presentation' of both multimedia resources and run-time modules. This operator is used by a run-time controller, which is a central component of the architecture. According to the differences between multimedia resources and run-time modules (Fig. 4), the multimedia object management is based on two mechanisms:

1. Resources are managed by a multimedia database system to provide retrieval, version control and consistency ensuring services for authors. For presenting resources, a media-type independent operator is implemented. The

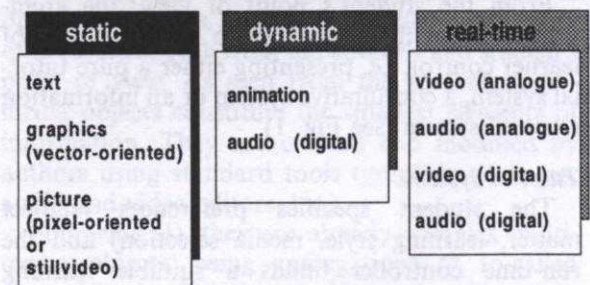


Fig. 3. Time-related categories of media type.

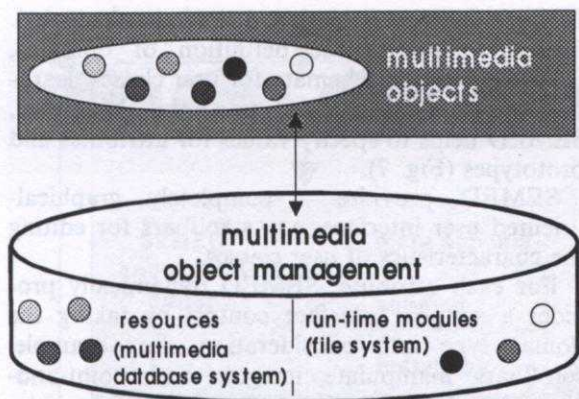


Fig. 4. Multimedia resources and run-time modules.

interaction provided by the operator may be varied for course-specific or media-type dependent aspects. For example, in some application contexts video sequences should be rewound to be repeated. The authoring tools for accessing the multimedia database include functions for importing and exporting from standard editors using the file system.

2. Run-time modules have to be stored by the file system and should not be managed by the database system. They normally make extensive use of multimedia resources inside the 'code', e.g. graphics, but are restricted to accessing these resources via the file system. Therefore the run-time modules would not profit from being managed by a database system.

Summarizing, the term multimedia object covers both multimedia resources and run-time modules. Generally, multimedia objects have a physical representation and can be executed. They present contents of the course and interact with the student where appropriate, forming the material on which courses are built. Learning paths are created by sequentially activating multimedia objects by the run-time controller. For a dynamic user-individual design of learning paths, the run-time controller must have access to 'knowledge' about the objects' contents, media design, pedagogical functions, etc. These semantics of the multimedia objects can only be described within the context of each course. For this purpose, a level of declarative elements of knowledge is introduced in the next chapter.

CREATING A KNOWLEDGE BASE WITH SEMANTIC DESCRIPTIONS OF MULTIMEDIA OBJECTS

From a pedagogical point of view, minimal self-contained elements of knowledge are called *learning atoms*. They should be as small as possible, in order to ensure high flexibility. At the same time they must be large enough to form a complete but minimal step for the student's learning process. Learning atoms are results of the conceptual design and refinement of CBT applications. An

important part of the system architecture described by this paper is a model for representing learning atoms and their relationships. This model includes user-class specific views by combining characteristics of hypersystems and semantic networks.

1. Learning atoms are described by attributes in a declarative manner. Findings from instructional theories have made clear that these attributes cannot be prescribed in a general way using just one fixed schema. For a variable schema, a two-step approach has been designed ensuring that different requirements are met, depending on subject matter, learning objectives and student characteristics. Following instructional theory, a basic domain-independent schema has been derived. All authors get a predefined schema. A specific course can be described in more detail by additional attributes (enhanced schema). Each attribute of the resulting schema can be classified into one of the following three classes:
 - Course-independent attributes, e.g. derived from instructional theory, such as the learner type (holistic versus serialistic).
 - Course-dependent attributes, e.g. 'painter's style' in a museum information system, or 'name of the operating system' in a course about basics of several operating systems.
 - System (internal) attributes, e.g. parameters for the number and functionality of control buttons for the presentation of multimedia objects ('stop', 'start' etc.).
2. The basic schema may optionally be enhanced by course-dependent attributes. Therefore an explicit meta schema has been set up to have a 'directory' of all available attributes within the context of a specific course. This meta schema also informs about available domains and related attributes. In order to increase reusability, the course-independent basic schema is used by all courses. In order to provide an evaluative example, findings from Merrill's CDT theory [2] have been transformed into a formal schema [3].
3. Relationships between several learning atoms like 'detail-of' are represented by complex typed links, comparable to hypersystems. For links, an attribute schema can be defined in analogy to learning atoms.
4. All courses of some reasonable complexity show hierarchical content structures, e.g. chapters, lessons, learning steps [4]. These structures are represented by aggregating learning atoms. Following hyper-terminology, resulting aggregations are also nodes of a hyper-network. Aggregations may be contained in another aggregation (Fig. 5). A limit for hierarchical levels has not been fixed. However, CBT designers recommend a maximum between 3 and 5 levels [4, 5].
5. A prototype concept has been integrated to support authors in assigning values to all

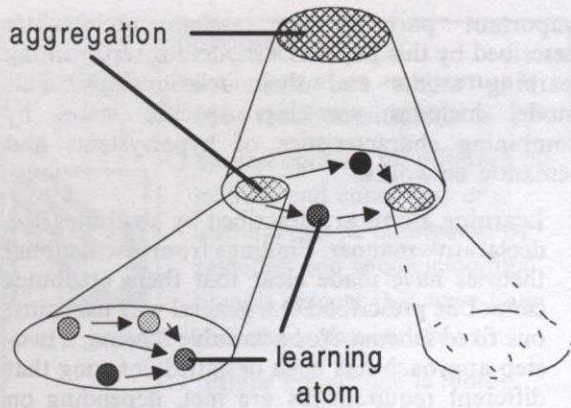


Fig. 5. Learning atoms and aggregations.

attributes of learning atoms, aggregations and links. If a learning atom has not been assigned a value, it is automatically given its prototype's value.

- Links between learning atoms are created by authors, one set of links for each user class. These *static* links represent the author's pedagogical design and recommend connections to the run-time controller. This way, for each user class an own course is configured. However, the run-time controller is not restricted to these recommendations but is free to activate every existing learning atom as a successor to the present one (dynamic creation of *virtual* links).

In order to create and edit the multimedia-related knowledge base (Fig. 6), an authoring tool called 'SEMED' (SEMantic network

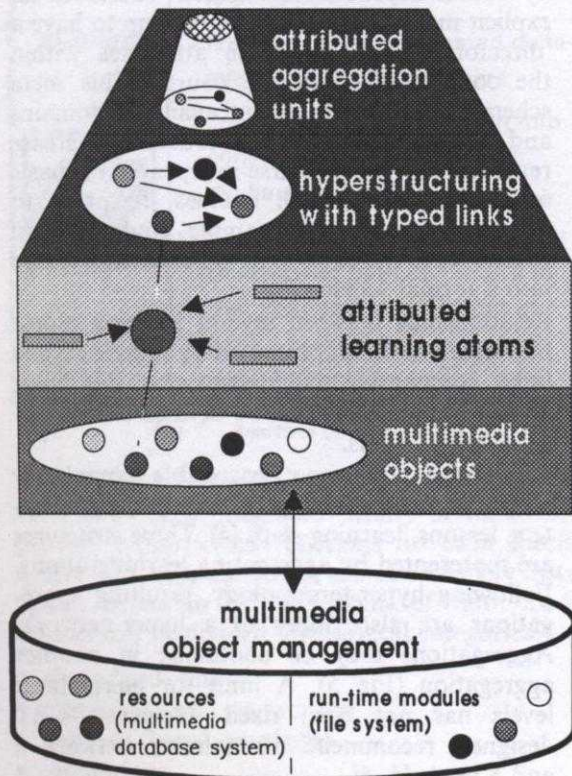


Fig. 6. Knowledge base: semantic network and hyper paradigm.

EDitor) has been designed and implemented. SEMED supports the definition of domains, course-dependent schemata for user classes, learning atoms, aggregations and links. Moreover, SEMED helps to specify values for attributes and prototypes (Fig. 7).

SEMED provides a completely graphical-oriented user interface, e.g. scrollbars for editing the characteristics of user classes.

For each attribute, SEMED dynamically provides a suitable interface control by taking the domain-type into consideration. For example, scrollbars manipulate integers and point-and-click lists support enumeration domains. An important task, also performed with SEMED, is setting up the relation between defined learning atoms and existing multimedia objects. Dialogs are used that are structured to reflect the separation between multimedia resources and run-time modules, offering appropriate views for both kinds of objects.

DYNAMIC BUILDING OF LESSONS

The run-time controller is responsible for creating learning paths dynamically, regarding students' preferences and characteristics.

Learning paths are created by sequentially activating multimedia objects, using the run-time controller mentioned above (see Fig. 8). For a dynamic, user-individual design of learning paths, the run-time controller must have access to 'knowledge' about the objects' contents (e.g. painter = 'Rubens'), media design (e.g. portion of video = 90%), pedagogical functions (e.g. 'introduction') [2], etc. These semantics of the multimedia objects are stored in the semantic data model outlined above.

To summarize, the following information is provided within the system architecture.

- Available learning atoms, aggregations and their characteristics are described by a semantic network outlined above.
- Typed links describe the relationship between learning atoms, given by the author from a pedagogical point of view.
- Using SEMED, the author creates a user profile and defines corresponding user classes (static user model).
- To achieve high flexibility, the user selects a level of learner control within the range of 'strictly guided', 'consultative' and 'free navigation'.

The knowledge represented by the semantic model is used as input for a fuzzy controller as well as for an artificial neural network (ANN) with a back-propagation topology (see Fig. 8).

Example of a fuzzy rule:

```
IF      (user_class.prior_knowledge =
        'very_small')
THEN   (learning_atom.difficulty = 'very_low')
```

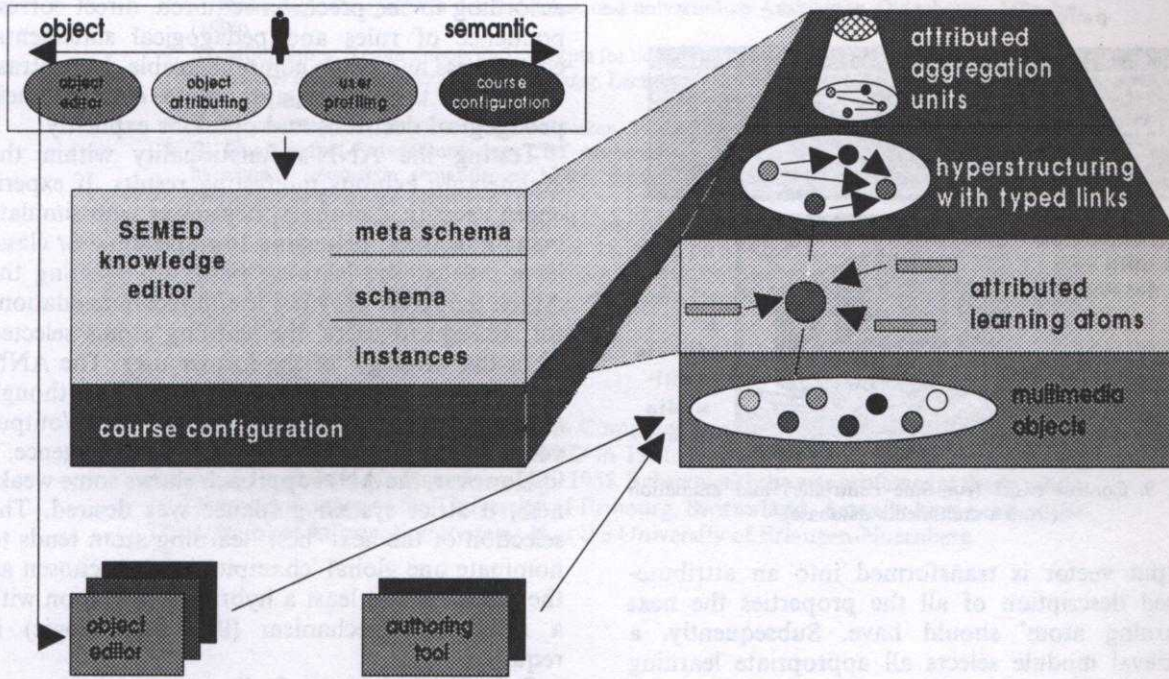


Fig. 7. Authoring tool SEMED.

The *training* data for the ANN are learning paths of former users. For successful training, these have to be experienced learners or authors, following a well-defined strategy that can be

'discovered' by the ANN. While *processing*, the attribute values of the current object instances are fed into the network.

The fuzzy controller's and/or ANN's resulting

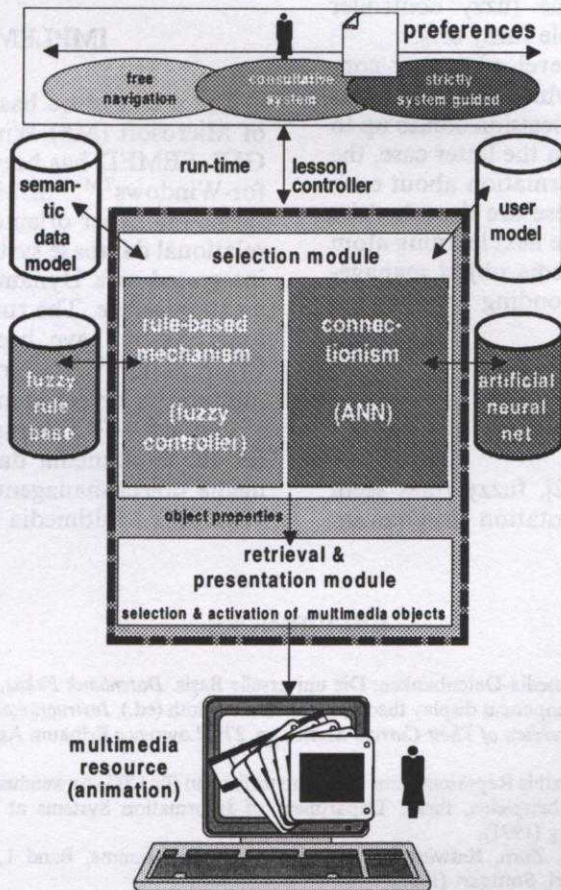


Fig. 8. Run-time controller, fuzzy controller and artificial neural network.

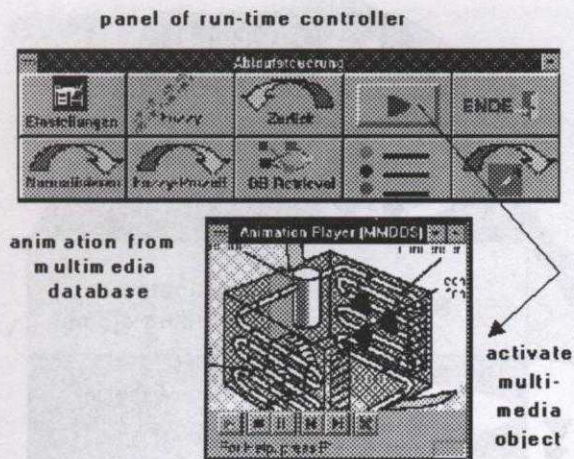


Fig. 9. Control panel (run-time controller) and animation (from a multimedia database).

output vector is transformed into an attribute-based description of all the properties the next 'learning atom' should have. Subsequently, a retrieval module selects all appropriate learning atoms to consult the user.

As an essential point, easy-to-use editors have been developed to assist CBT authors with little experience in assigning and transforming values of attribute-oriented data models into fuzzy sets and/or domains of ANNs [11]. For example, depending on the attribute's domain type, the ANN editor automatically calculates the number of needed input/output neurons and the fuzzy controller automatically generates suitable fuzzy sets.

According to the selected level of learner control, the system determines which learning atom will be presented next, or this decision comes up to the user selecting from a list. In the latter case, the user can be supported by information about each listed learning atom, since these are described by the semantic network. After the next learning atom has been selected, the multimedia object management activates the corresponding multimedia object (see Fig. 9).

EXPERIENCES

For pedagogical findings [2], fuzzy rules seem to be an adequate representation mechanism

according to the precision required, direct correspondence of rules and pedagogical statements, conciseness and being comprehensible. In contrast to ANNs, they allow authors to describe their pedagogical decisions and opinions explicitly.

Testing the ANN's functionality within the architecture exhibits interesting results. If experienced users (e.g. authors themselves who simulate being a 'learner' belonging to a specific user class) have created the learning paths for training the ANN, the ANN provides useful recommendations for subsequent users: the learning atoms selected fit in the 'strategy' of the former user. The ANN works like an adaptive, auto-learning filter, though in all cases with more complex input/output vectors the training has not shown convergence.

However, the ANN approach shows some weakness, if strict system guidance was desired. The selection of the next 'best' learning atom tends to nominate one global 'champion' that is chosen all the time. Here at least a hybrid combination with a rule-based mechanism (like fuzzy logic) is required.

Summarizing, these findings are encouraging, because involving an ANN enables an author to configure user-specific CBT courses by just walking along an appropriate path of learning atoms. The strategy is automatically learned by the ANN and can be applied to subsequent users.

IMPLEMENTATION

The architecture has been implemented on top of Microsoft (MS) Windows™ as the underlying GUI. SEMED has been built using Turbo-Pascal-for-Windows™. SEMED automatically transforms the hyper-oriented semantic network into a relational database system. This database system is integrated as a Dynamic Link Library (DLL) via a call interface. The run-time controller and fuzzy logic system have been implemented using MS C/M S SDK™. The artificial neural network and multimedia object management are based on MS Visual C++™. Gupta's SQL Server™ is used for the multimedia database system. The multimedia object management makes extensive use of Window's Multimedia Control Interface (M.C.I.).

REFERENCES

1. D. Merten, Multimedia-Datenbanken: Die universelle Basis. *Datenbank Fokus*, pp. 20 (1994).
2. M. D. Merrill, Component display theory, in C. M. Reigeluth (ed.), *Instructional Design—Theories and Models: an Overview of Their Current Status*, pp. 279. Lawrence Erlbaum Associates, Hillsdale, NJ (1983).
3. S. Gabelmann, Flexible Repräsentation von Lehrstoffwissen für CBT-Anwendungen Methodisches Konzept mit Fallbeispielen, thesis, Department of Information Systems at the University of Erlangen-Nürnberg (1991).
4. E. Gabele and B. Zürn, Entwicklung interaktiver Lernprogramme, Band 1, Grundlagen und Leitfaden. Poeschel, Stuttgart (1993).
5. Marconi Simulation, Author's Manual for the Mandarin Instructional System. Marconi, Edinburgh (1992).

6. F. Bodendorf, Computer in der fachlichen und universitären Ausbildung. Oldenbourg, München (1990).
7. M. Mühlhäuser, Requirements and concepts for networked multimedia courseware engineering, in: H. Maurer (ed.), *Computer Assisted Learning*, Lecture Notes in Computer Science, no. 360, pp. 400. Springer, Berlin (1989).
8. K. Meyer-Wegener, Multimedia-Datenbanken. Teubner, Stuttgart (1991).
9. W. Günselmann, Entwicklung eines CBT-Autorenwerkzeugs zur Lehrstoffbeschreibung mit Hilfe hypermediaorientierter semantischer Netze, thesis, Department of Information Systems at the University of Erlangen-Nürnberg (1993).
10. W. D. Milheim, Theoretical bases for the use of learner control: three different perspectives. *Journal of Computer-Based Instruction*, vol. 18, no. 3, pp. 99 (1991).
11. O. Hofmann, Entwicklung einer lernfähigen konnektionistischen Komponente zur Ergänzung einer Ablaufsteuerung für CBT-Applikationen, thesis, Department of Information Systems at the University of Erlangen-Nürnberg (1993).

Freimut Bodendorf took an MS degree in Computer Science in 1978 from the University of Erlangen-Nuernberg and obtained a PhD in 1981. He was professor at the Postgraduate School of Engineering in Nuernberg until 1988. Subsequently he was professor at the Institute of Computer Science at the University of Fribourg, Switzerland. Actually he is head of the Department of Information Systems II at the University of Erlangen-Nuernberg.