Proceedings of the 1994 Winter Simulation Conference
ed. J. D. Tew, S. Manivannan, D. A. Sadowski, and A. F. Seila

A DATABASE DESIGN FOR UNIFORM REPRESENTATION OF HYPERMEDIA AND MATHEMATICAL MODELS

Howard W. Beck
Agricultural Engineering Department
University of Florida
Gainesville, FL 32611, U.S.A
hwb@agen.ufl.edu

Paul Fishwick
Computer and Information Sciences
University of Florida
Gainesville, FL 32611, U.S.A
fishwick@cis.ufl.edu

Glen Smerage
Agricultural Engineering Department
University of Florida
Gainesville, FL 32611, U.S.A
smerage@agen.ufl.edu

ABSTRACT

Semantic data modeling techniques are used to provide a theoretical and applied basis for construction of hypermedia systems which incorporate mathematical models. Traditional hypermedia systems provide informal modeling tools and fail to represent adequately the concepts which they are intended to organize and present to users. As a result, such systems provide few services, and they must be constructed and applied with considerable manual labor. The components of the hypermedia system, including text, drawings, photographs, video, equations, plots, conceptual models, symbolic manipulations, and simulation, can be described as objects in a database. By applying sound data modeling techniques, behavior, structure, and association among concepts are formally described. Data manipulation techniques including query processing, automatic classification, and knowledge discovery can also be applied. A user interface is generated automatically once the underlying structure of concepts in a particular domain is captured in the database. This work is applied to organizing and presenting materials for a graduate level course in biological systems analysis.

1 INTRODUCTION

Traditional hypermedia systems offer crude data modeling capabilities. As a result they cannot provide many of the services typical of database management systems. For example, search is limited to manual browsing, keyword searching or fulltext searching. It is not possible to search over the *concepts* in the database, a severe limitation in large hypermedia systems where user disorientation is a common problem.

For example, if one examines HTML used in World-Wide Web (Berners-Lee et al. 1992) as a model of hypermedia, it is clear that very few services typical of database management systems are offered. The underlying data structure is a set of HTML files, and the only organization to these files consists of their placement in UNIX directories. Updates to such a system are very difficult (consider the effects of a change in the location of an important Home page), and the integrity of the database (such as assuring that the target of a hyperlink exists) is difficult to enforce. Users may add information in an ad hoc fashion using arbitrary links. Tags defined for HTML describe the structure of a document well (although currently they are not as complete as tagging systems such as LaTex (Lamport 1986) or standard word processors) but they are not designed for describing what the document is about.

One solution is to use a formal data modeling language to describe the concepts in the database. By formal data modeling language is meant that the primitive constructs of the language are well defined. This leads to identification of precisely what operations can be applied to data structures created to conform with the data modeling language. Semantic data modeling, widely used in database management, can be applied to data typically used in hypermedia systems to provide a more formal basis for data definition. Once a formal representation is in place, operations can be applied to the data structures. This could not only result in improved searching techniques, but other services could be provided. Hypermedia is generated as a *projection* from data structures to screen presentation. Hypermedia is an interface device. An interface is generated automatically once the underlying structure of information is captured in data structures. A user interacts with the system by manipulating objects on the

screen as in conventional hypermedia systems, but underneath lies a formal representation of concepts which can be utilized in a variety of ways.

The same approach can be used to incorporate modeling into hypermedia systems. An improved data model for representing knowledge in a hypermedia system can be extended to mathematical and other modeling entities. This can be considered a knowledge-based approach in which detailed, declarative representations of concepts involving mathematical modeling are built. Manipulations of these representations lead to higher reasoning capabilities such as model design and construction, derivation of mathematical proofs, and problem solving. Knowledge about how to build mathematical models can be incorporated to cover the complete process from conceptual modeling to simulation. Through a hypermedia interface, users can explore each mathematical object (be it a conceptual object, single variable in an equation, or a complex simulation) in brief or in great detail.

We will present an example of applying these concepts by constructing conceptual models as well as detailed representations of mathematical entities, showing operations which can be made on the resulting data structures, and showing how these internal structures can be projected onto a user interface to generate the hypermedia system. The theoretical issues addressed here are motivated by our work in developing a hypermedia system for teaching engineering in a graduate level course on biological systems analysis.

2 DATABASE

We developed a semantic data model (Beck et al. 1994) to use as the underlying knowledge representation and data management facility for storing and organizing all information in the hypermedia system. The associations among concepts which are expressed informally by a general-purpose hyperlink in conventional hypermedia systems are captured formally by the abstractions in the semantic data model. The data model supports standard database abstractions such as generalization, classes/instances, part/whole, association, and object identity. Data objects consist of attributes having values which can be simple or composite.

Our data model does not contain procedural attachments or methods. This is in order to facilitate a number of inferencing operations related to query processing and object manipulation including automatic classification of objects (determining where in a class taxonomy a new class or instance belongs) and conceptual clustering (automatically generating a new class by induction over sets of instances). Such inferences cannot be applied to procedural representations.

Though the data model is quite simple, we have applied it successfully to a large hypermedia system covering the broad domain of agriculture. The existing hypermedia system includes a large collection of documents and images. For example, we describe text covering a particular topic using a data object such as:

```
Definition of Derivative
  ISA: Glossary Term
  Attributes
    Body: {...text...,FIGURE 2.3}
    Discussion: <REF OID 1001>
    Existence: <REF OID 1002>
    Higher Derivatives: <REF OID 1003>
```

for a text fragment which provides a glossary entry defining the derivative. The body includes a brief textual definition as well as a figure (FIGURE 2.3 is a pointer to an object stored in another region of the database dealing with figures). Other objects associated with this definition are related through the other attributes. This object can be projected onto the user interface for display in the hypermedia system. But it also serves as a basis for defining concepts, in this case the definition of derivative, which can be used by a query processor which searches over concepts.

There is growing interest in using databases to facilitate the modeling process. The key to integrating databases and simulation is to use database data structures (data objects) to represent models at all levels, from the abstract conceptual models to the detailed simulation processes such as solving a differential equation (e.g. Hitz et al. 1993, Lenard 1993). There may be practical if not theoretical limitations to achieving this goal. But not achieving this goal would force a compartmentalized solution in which database facilities may be good for storing general descriptions of models, but then the real model must be described by additional data structures outside the database, and a separate package must be used to analyze or run the models. The disadvantage of such an approach is that reasoning processes cannot be applied at all levels. If a database management system does not have the capabilities to analyze models and their components, then it cannot adequately classify, compare, contrast, or retrieve models, the very services we wanted the database to perform.

Below we illustrate several ways in which various aspects of modeling can be described by data structures in a database. We are attempting to achieve a fully integrated solution, not a compartmentalized approach.

3 CONCEPTUAL MODELS

We will use an example on temperature-dependent development of organisms. Many organisms, including plants and insects, grow at rates which are determined uniquely by temperature. The following problem presents some experimental observations, and then inquires about the dynamic relationships between time, temperature, and development rates.

> Problem: Laboratory experiments on celery plants at constant temperature indicate that petioles were initiated such that their number N as a function of time t and temperature T was given by:
>
> $$N(t,T) = \begin{cases} (5/40)t & T = 15^\circ C \\ (10/40)t & T = 20^\circ C \\ (15/40)t & T = 25^\circ C \\ (20/40)t & T = 30^\circ C \end{cases}$$
>
> Determine a model of development of a celery plant based on these experimental observations which will predict the number of petioles as a function of time for the following temperature function:
>
> $$T(t) = 10 + (\pi/4)\sin(\pi/40)t$$

We begin with a conceptual model (Figure 1). Different ways to model development rate are described by objects at various levels of abstraction. Database objects provide a way to examine a system and its components at a conceptual level, without regard for the details of how those components are implemented. In the figure it is seen that Temperature-Dependent Development Rate (TDDR) is a subclass of development rate, which is a subclass of rate. Rate, in the most abstract sense, is a change of some quantity with respects to time. Inverse links capture reciprocal relationships between rate and quantity arising from differentiation and integration. Development rate is, more specifically, a change in a developmental quantity (age, size of organism). Temperature-Dependent Development introduces the influence of temperature on rate.

So far, we have describe these processes at a very abstract level. Specifying exactly how temperature affects development rate occurs at lower levels. In the example, observing the experimental data suggests a linear temperature development model (LTDM).
The LTDM may be described qualitatively with several variations. The simplest (Model 1) assumes that there is a threshold temperature below which the organism cannot develop and above which rate increases linearly with temperature. Another refinement (Model 2) states that there is an upper temperature threshold, above which development no longer increases. Another (Model 3) specifies that development decreases at temperatures above this upper threshold. All of these approaches apply to various theories of development as well as developmental characteristics of particular organisms.

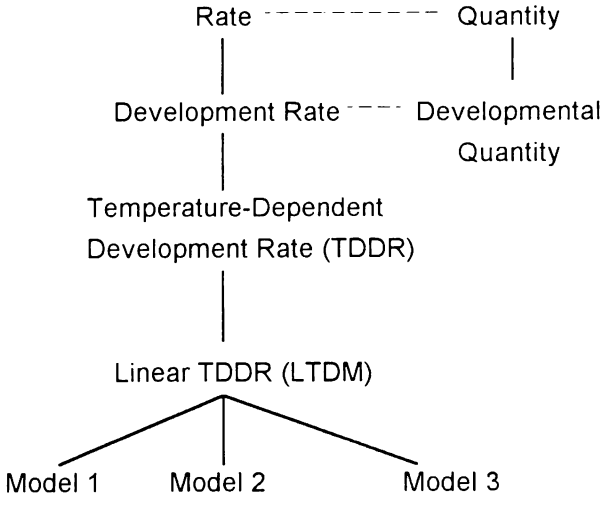

Figure 1. Conceptual Models Arranged in a Database Class Taxonomy (Solid lines are ISA relationships; dashed are associational)

A purpose of conceptual modeling is to introduce these various options and to use a database class taxonomy to show the interrelationships of the models at various levels of abstractions. Modelers can select particular objects for use in constructing a model (we select the LTDM Model 1 for the current example). Modelers can also begin by choosing a model at a very general level and then later descend the taxonomy making more specific selections during model refinement (Fishwick 1992).

The internal details of an object's behavior will now be explored in more detail. The hypermedia aspects of conceptual modeling are merely to build models by selecting and arranging objects on the screen. Users can browse through details of the model by examining particular objects.

4 BEHAVIOR

Behavior of objects can be described in declarative terms, as in analytical models, or in procedural terms, by using methods. When methods are used, the only way to study behavior is through simulation. Declarative representations are symbolic and can be manipulated through a variety of automated inferencing operations. Since the example above can be treated using declarative techniques, these will now be described.

4.1 Equation Representation

The conceptual model for temperature fluctuation would include time-varying functions of various kinds. In particular, a periodic function appropriately describes the diurnal rise and fall of temperature. An idealized model for daily temperature can be described declaratively by the following equation:

$$T(t) = 10 + 5\pi\sin 2\pi t$$

Representing an equation symbolically means using data structures to describe both its syntax (the relationships among the elements in the equation) and semantics (the meaning of elements formed as a result of those relationships). The equation displayed above was generated by creating a sequence of characters using a word processor. Though the syntax and semantics of this equation are apparent to you the reader, they are not explicitly coded in the data structure which is only a sequence of characters (even more problematic would be an equation coded as a bit-map graphic as must be done in many hypermedia systems which cannot display equations using math symbols).

One symbolic representation of this equation, an s-expression, could take the form:

(= (func T (var t)) (+ 10 (x 5 π (sin (x 2 π (var t))))))

Here the syntax of the equation is made explicit by nested parenthesis. The semantics of the equation is defined by predicates (=, func, var, +, ×, sin) having one or more arguments.

We use s-expressions here to simplify the notation. In the database we use data objects containing attributes with values rather than s-expressions to represent equations. For example, the class "+" would contain instances with attributes for each term of the sum. The values of these attributes would be the expressions to add.

It is easy to generate a display for this equation by projecting the symbolic form to the more traditional mathematical equation format. Generation rules applied to each predicate describe how the predicate is to be displayed. These rules are applied recursively to the arguments of the predicate. Thus, if they so choose, modelers can work with the system entirely by using standard mathematical notations and need not be concerned with the underlying symbolic form.

4.2 Equation Manipulation

A symbolic representation can be manipulated through the application of rules. For example, some well known rules for computing derivatives are:

Derivative of a sum:

(= (d/dx (+ a b)) (+ (d/dx a) (d/dx b)))

Derivative of a product:

(= (d/dx (x a b))(+ (x (d/dx a) b)(x a (d/dx b))))

Derivative of an exponential:

(= (d/dx (exp a))(x (d/dx a)(exp a))

A processor can be written which applies these rules to compute derivative. Note that both the equations and the rules for transforming the equations can be stored as database objects, leading to a uniform treatment of both model descriptions and their analysis.

4.3 Calculations

Computations are made by evaluating each predicate, and passing up the result to higher predicates. For example, computing T(0):

(= (func T (var t)) (+ 10 (x 5 π (sin (x 2 π (var t))))))
(= (func T 0) (+ 10 (x 5 π (sin (x 2 π 0)))))
(= (func T 0) (+ 10 (x 5 π (sin 0))))
(= (func T 0) (+ 10 (x 5 π 0)))
(= (func T 0) (+ 10 0))
(= (func T 0) 10)

Behavior of each predicate must also be described specifying how the arguments are to be combined. It is simple to apply operators such as +, ×, or sine, and it may not seem necessary to represent this behavior declaratively. Nevertheless, it is conceivable that even

the mathematical basis for these primitive operations could be described in the database.

4.4 Problem Solving Heuristics

In order to solve the problem stated in the example, a general framework or approach to problem solving is needed. One way to capture problem solving strategies is through the use of heuristics. In solving the problem in the example, a number of key concepts must be applied which are not explicitly given in the statement of the problem. They are essentially the "know how" required to solve the problem. Once these key concepts are in place, routine application of analytical techniques can be used to finish. The key concepts are:

A. For events occurring as a linear functions of time, the interval between unit events can be computed by solving:

$$N(t+\Delta t) - N(t) = 1$$

for Δt.

B. A linear model for temperature dependent development rate is described by the relationship shown in Figure 2.

C. Degree days are computed by the integration of a time function of temperature above a threshold (T_o).

D. Integration over initiation rate gives number.

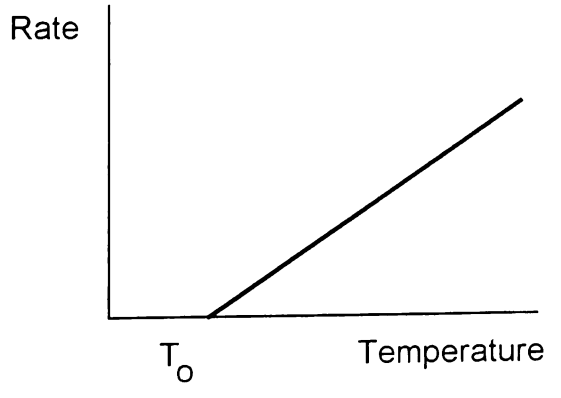

Figure 2. Development Rate Increases Linearly with Temperature Above a Threshold T_o

4.5 Symbol Definition

The symbols appearing in mathematical equations are references to complex entities. It is possible to expand upon the meaning of each symbol by building a data object for it. This is especially useful in a hypermedia system when a user wishes to obtain additional information about a variable appearing in an equation. Thus, since we know that N appearing in the equations above stands for the number of petioles on a celery plant, we can build a description for N:

N
ISA: Developmental Quantity, Plant Part
ATTRIBUTES
Value: REAL
Organ: Petiole
Part Of: Celery Plant
Photo: IMAGE CH032

Here N, which is a developmental quantity and a plant part, has as its value a real number (instances of N would contain a particular numeric value) . The value for "Part Of" is the name of an instance which describes the celery plant. Photo is a reference to an image which is a photograph of a petiole.

5 CREATING A DATABASE

It has been shown how a variety of symbolic representations can be used to describe equations, describe rules for manipulating equations, perform calculations, describe rules of thumb needed for problem solving, and provide background knowledge about such things as variable appearing in an equation. Large numbers of these data structures would be needed in a system capable of reasoning about mathematical models. We use the database to systematically store all these structures.

Objects are created in a variety of ways. In addition to directly specifying objects through a high-level editor, we make extensive use of parsers to analyze input data and convert them to objects. One such parser is used to convert tagged word processing files into data objects describing the document structure. Another parser analyzes equations prepared using equation editors or using notations such as LaTex and creates a symbolic representation. The resulting data objects can be used in a variety of applications, including creation of hypermedia.

6 CREATING HYPERMEDIA

Once data structures have been created and stored in the database, rules specifying presentation style can be used to transform these internal representation to on-screen presentations. The user interface is generated automatically from the underlying data structures. Different mapping rules can produce different interface styles. Users interact with the system as they do with conventional hypermedia systems, by browsing though a network of concepts, mainly by pointing and selecting items appearing on the screen.

We want to contrast this approach with one which we might have taken using conventional hypermedia development techniques. For example, we could have constructed a network for World-Wide Web with nodes containing HTML-tagged text describing model components and links establishing relationships among these components. Users could explore the model by browsing through this web.

But such an approach would have been very static. The approach presented here achieves the same objective; a user can browse through descriptions of models at any level of detail, but the data structures also support design and analysis. The system can provide automatic assistance in model construction, and explanation aimed at the user's ability level. This is achieved by creating data structures which represent real models, not just text descriptions of models. Thus it is a unified approach, and hypermedia happens as a side-effect of building a good representation of the models and the modeling process.

REFERENCES

Beck, H. W., T. Anwar, and S. Navathe. 1994. A conceptual clustering algorithm for database schema design. *IEEE Transactions on Knowledge and Data Engineering*. 6(3):396-411.

Berners-Lee, T.J., R. Cailliau, J-F Groff, and B. Pollermann. 1992. World-Wide Web: The information universe. *Electronic Networking: Research, Applications and Policy* 2(1):52-58.

Fishwick, P.A. 1992. An integrated approach to system modelling using a synthesis of artificial intelligence, software engineering and simulation methodologies. ACM Transaction on Modelling and Computer Simulation, 2(4):307-330.

Lamport, L. A. 1986. LaTex: Document preparation system. Addison-Wesley Publishing Company, Inc. Reading, MA.

Hitz, M., H. Werthner, and T.I. Ören. Employing databases for large scale reuse of simulation models. *Proceedings of the 1993 Winter Simulation Conference*, G.W. Evans, M. Mollaghasemi, E.C. Russell, W.E. Biles (eds). pp. 544 -551.

Lenard, M. L. 1993. A prototype implementation of a model management system for discrete-event simulation models. *Proceedings of the 1993 Winter Simulation Conference*, G.W. Evans, M. Mollaghasemi, E.C. Russell, W.E. Biles (eds). pp. 560-568.

AUTHOR BIOGRAPHIES

HOWARD W. BECK is an Assistant Professor of Agricultural Engineering at the University of Florida. He has a MS in Electrical Engineering from the University of Illinois, and PhD in Computer Science from the University of Florida. His research involves development of intelligent information retrieval systems. He has developed several large scale hypermedia systems, and is conducting research on semantic data modeling applied to hypermedia, natural language processing, and machine learning in databases.

PAUL FISHWICK is an Associate Professor in the Department of Computer and Information Sciences at the University of Florida. He received the BS in Mathematics from the Pennsylvania State University, MS in Applied Science from the College of William and Mary, and PhD in Computer and Information Science from the University of Pennsylvania in 1986. His research interests are in computer simulation modeling and analysis methods for complex systems.

GLEN SMERAGE is an Associate Professor of Agricultural Engineering at the University of Florida. He received a PhD from Stanford University. His research and teaching are in systems analysis applied to biology, including population and physiological systems. Other recent efforts have been in applications of microcomputer multimedia to instruction of engineering and science.