EXPERT SIMULATION SYSTEM BASED ON A RELATIONAL DATABASE

Robert E. Shannon
Department of Industrial Engineering
Texas A&M University
College Station, Texas 77843

Martha A. Centeno
Department of Industrial Engineering
FAMU/FSU College of Engineering
Tallahassee, Florida 32300

ABSTRACT

The presentation will describe the design and early implementation of a life-cycle oriented simulation environment based on a commercial relational database. The modules and database that make up the environment will be discussed. This paper presents the rationale for such a system and the current state-of-the-art.

1. TRENDS IN LANGUAGE DEVELOPMENT

As the years have passed, new simulation languages have been developed and old ones enhanced. Language developers have focused their attention on three objectives: (1) reduced model development time, (2) improved accuracy, and (3) improved communication. There is, in fact, a revolution taking place and an explosion of creative activity in simulation modeling and language development. This revolution has been brought on to a large degree by developments outside the simulation community. Among these has been the advent of the micro-computer, graphics, and the technologies of artificial intelligence and expert systems.

The ultimate goal of any programming language is to close the gap between what users conceptualize as a representation of a system and how they actually express that relationship in some executable form. The current phase is one in which the development of simulation software is in a significant transition period. The emphasis is upon ease of use and providing an integrated simulation environment rather than simply more powerful languages. Henriksen [1984] says, "An integrated simulation environment is a collection of software tools for designing, writing, and validating models; writing and verifying simulation programs (implementing models); preparing model input data; analyzing model output data; and designing and carrying out experiments with models." He further points out that an important difference between a truly integrated approach and that promoted by traditional simulation languages or specialized tools is feedback. The simulation environment should include a mechanism whereby results of the experiment phase are feed back to the model construction phase. It is through this feedback mechanism that task closure is achieved without unduly interrupting the simulationist's train of thought.

The current language development activity is motivated by two problems as well as the opportunities presented by the confluence of developments in several areas of technology. It is clear to all practitioners that doing simulation modeling studies tends to be a low productivity activity. Even expert practitioners are repeatedly surprised by how much effort is required to accomplish a useful result. The traditional modeling life-cycle is very labor intensive and time consuming. The computer is used mainly for execution of the model, while the construction of the model, design of the experiments to be run, and analysis of the results are shouldered by the modeler. Furthermore, since most of the "intelligent" functions are performed by hand, a great deal of knowledge is discarded at the completion of each project. Most models are treated as "throw away" items as is the analysis required to solve the problem [Zeigler 1984; Oren 1986].

Giddings [1984] has argued that in order for significant changes in the man-power intensive nature of model development to be realized, the development process itself must be redefined. Attempts must be made to "conserve" effort by incorporating knowledge and expertise into the software. With this approach, as much work as possible will be shifted from the human to the machine, thus freeing the modeler for handling decisions that truly require the "human touch."
heretofore has been the user interface. Much of the work in the AI field on automated reasoning has dealt with the inference engine and how to use the facts and rules contained in the knowledge base to generate a solution. The development of these two concepts, generalization and aggregation, provide the means to treat the relational database as an object-oriented database.

2. EXPERT SYSTEMS

Beginning with the mid 1980's AI and Expert Systems technology gained a lot of attention from the simulation community because they seemed to offer many advantages over more traditional simulation approaches. The potential of expert systems and AI in simulation has been explored by Shannon [1984, 1986, 1989], Shannon et al. [1985], Reddy [1987], and O'Keefe [1986] among others. Several languages based upon AI methodologies have been developed. Adelsberger and Neumann [1985] explored the use of PROLOG as a simulation language. ROSS developed at the RAND Corporation [Klahr 1984] and KBS developed at Carnegie-Mellon [Reddy et al. 1985], are examples of simulation languages based upon AI concepts.

Use of AI techniques for simulation environments can be found in Ketcham and Hahen [1986] who proposed an integrated network of expert systems to handle the decision making process in simulation studies. Zeigler [1987] has taken advantage of the compatibility between object oriented programming paradigms and discrete simulation formalisms to develop an environment for model construction. Guariso et al. [1989] designed and prototyped a knowledge based system that interactively generates simulation models.

Despite their potential, however, AI-based systems have been regarded by many as an expensive, sometimes inefficient solution. More recently, a number of researchers has taken the approach of developing intelligent front ends and back ends around existing commercial languages [e.g., Brazier and Shannon 1987; Haddock and Davis 1985; Murray 1986; among others]. Conceptually, merging the best of these two fields would yield a knowledge-based simulation environment in which a simulation model is synthesized by accepting a description of the model and consulting the appropriate knowledge base. This implies that generic descriptions of the objects to describe the system are stored in long-term fashion within the computer for use whenever needed. The latter has brought a third player into the game: databases.

3. DATABASES

The use of databases, in conjunction with simulation modeling, began just recently. TESS [Standridge et al. 1985; Grant and Starks 1988] integrates simulation, data management, graphics, and animation on top of a network database. TESS represents a new generation of software that integrates model building, model execution, analysis and presentation of results. Ketcham [1986, 1987, Ketcham et al. 1989] developed MBS which has undergone several refinements and has been renamed from MBS to IBIS. IBIS is based upon a hierarchical database which makes it a little hard to use. Each application program must know exactly how the data is stored, plus the user must define the schema of each record type needed to describe the system to be modeled. In addition, most corporate data base management systems today are relational rather than hierarchical.

The concept of the relational database model, as proposed by Codd [1970], is simple, yet robust. A relational database is a database that is perceived by its users as a collection of tables (and nothing else but tables). [Date 1986]. The relational model does not imply nor does it address specific physical approaches for data storage and retrieval. Merging simulation and the relational model was explored by Yancey [1987], who investigated the idea of a relational database management system centered environment to properly collect and manipulate data generated by simulation runs.

Ghoshal [1988] took Ketcham's [1986] schema definitions and converted them to relational-based schema using the format given by Smith and Smith [1977]. Ghoshal showed that you could superimpose a hierarchical structure upon a relational database by applying the concepts of "aggregation" which refers to an abstraction in which a relationship between objects is regarded as a higher level relational database. The set of similar objects is regarded as a generic object to pure relational tables.

The concepts of generalization and aggregation in databases are the equivalent of classes and objects in simulation and AI. Aggregation is easily achieved through the normalization process. These two concepts, generalization and aggregation, provide the means to treat normalized databases as objects representing different levels of abstraction.

4. CONCLUSIONS

Simulation, AI, and the relational data model are just beginning to be studied as a triplet. Centeno [1990] describes an integrated simulation environment that is under development which is based on ORACLE, a commercial relational database management system. The Integrated Simulation Modeling Environment (ISME) uses the relational database as the only means to represent the objects in a simulation model, the relationships between them and some elements of the user interface.

The flexibility, robustness, and elegance of the relational model have caused it to be widely used in industry and it is available on the entire gamut of computer hardware. Furthermore, not only does the use of the relational model allow the development of a combined ordinary database and simulation system, but it offers a suitable framework for the construction of a knowledge base for a KBS.

REFERENCES


Centeno, M.A. (1990), "Design of an Integrated Simulation Modeling Environment Using a Relational Database Framework," Doctoral Dissertation, Industrial Engineering Department, Texas A&M University, College Station, TX.


Ketcham, M.G. (1986), "Computer Simulation as a Decision Support Tool," Doctoral Dissertation, Industrial Engineering Department, Texas A&M University, College Station, TX.


Ketcham, M.G. (1987), "IBIS: Information-Based Integrated Simulation: The Design of a Simulation Model Base for Manufactur-
Expert Simulation System Based on a Relational Database

Murray, K.J. (1986), "Knowledge-Based Model Construction System: An Automatic Programming Approach to Simulation Modeling," Ph.D. Dissertation, Department of Computer Science, Texas A&M University, College Station, TX.