Proceedings of the 1987 Winter Simulation Conference
A. Thesen, H. Grant, W. David Kelton (eds.)

AUTOMATIC MODEL SYNTHESIS: USING AUTOMATIC PROGRAMMING AND EXPERT SYSTEMS TECHNIQUES TOWARD SIMULATION MODELING

Karen J. Murray *
Bell Communications Research
444 Hoes Lane - RRC
Piscataway, NJ 08854

Sallie V. Sheppard
Department of Computer Science
Texas A&M University
College Station, TX 77843

ABSTRACT

A Knowledge-Based Model Construction (KBMC) system is described which has been developed to automate the model construction phase of the simulation life-cycle. The system utilizes a knowledge-based approach to automatic programming to build a simulation model and extends the knowledge-based approach to include model specification acquisition. The system's underlying rule base, implemented in the production system paradigm of OPS83, incorporates several types of knowledge. Domain knowledge is used in conjunction with simulation modeling knowledge to facilitate a structured interactive dialog for the acquisition of a complete model specification from a user. Modeling knowledge and target language (SIMAN) knowledge are then used to automatically construct an executable discrete simulation model from this specification. This paper presents an overview of the KBMC system and focuses on various issues involved in the conceptualization and implementation of such a system.

KEYWORDS: artificial intelligence in simulation, knowledge-based model construction, discrete simulation modeling

INTRODUCTION

Simulation modeling is still considered an "art" and somewhat of an intuitive process despite the fact that its roots lie in the "sciences" of computer science, mathematics, and statistics. The continuing surge of interest in artificial intelligence (AI) and expert systems has led a growing number of simulation researchers to suggest the potential for the use of AI techniques in automating various aspects of the simulation life cycle. The work described here is one example of the application of such techniques toward model synthesis or simulation modeling. Automatic programming techniques have been consolidated with a rule-based production system paradigm to allow the elicitation of a model specification from a user and the automatic construction of the corresponding model.

A taxonomy of the various approaches to simulation modeling is presented in Figure 1. The traditional or manual approaches involve a simulationist who builds a simulation model in either a general purpose programming language or a special purpose simulation language. The three categories of automated approaches to simulation modeling involve the use of program generators, automatic programming systems, and simulation systems or environments, where each successive category represents an increase in system sophistication and a decrease in the effort and simulation expertise required of the user.

Many of the current simulation research efforts are focusing on the development of highly sophisticated simulation systems or environments. ROSS [Klahr and Faught 1980a; Klahr et al. 1980b; McArthur and Klahr 1982] and KBS [Baskaran and Reddy 1984; Fox 1981; Fox and Reddy 1982; Fox et al. 1984; McRoberts et al. 1985; Reddy and Fox 1982a; Reddy and Fox 1982b; Reddy et al. 1983; Reddy et al. 1985; Reddy et al. 1986] represent completed examples of such systems, while the Conceptual Modeling approach of Lehmann, et al. [Lehmann et al. 1986] and the Simulation Environment System by Adelsberger, et al. [Adelsberger et al. 1986; Umphress 1987] are in the early stages of long-term development.

Although the knowledge-based model construction (KBMC) system described here focuses on the domain of simulation modeling, it also closely parallels suggestions made by software engineering researchers for a revolutionary approach to software production that incorporates AI techniques, that is, an automated software production paradigm. This work has been motivated by automatic programming research, most notably that of Barstow in the development of PECOS [Barstow 1977a; Barstow 1977b; Barstow 1979a; Barstow 1979b; Barstow 1979c; Kant and Barstow 1981] and his ongoing research in the area of automatic programming [Barstow 1983; Barstow 1985a; Barstow 1985b].

The KBMC system utilizes a knowledge-based approach to

```
                         Simulation Modeling
                         /                 \
          Traditional Approaches            Automated Approaches
               /        \                  /          |          \
       general        special       program      automatic      AI-based
       purpose        purpose       generators   programming    simulation
       programming    simulation                 systems        environments
       languages      languages
```

Figure 1. Taxonomy of approaches to simulation modeling.

automatic programming to automatically construct an executable simulation model and extends this knowledge-based approach to include specification acquisition. Thus the need for intricate specification procedures on the part of the user is greatly reduced. In principle, the structured interactive dialog of the KBMC system is similar to a detailed questionnaire, but the knowledge-based approach eliminates the need to present all possible questions to the user.

The knowledge-based approach to model construction requires the capture of simulation expertise to allow the automatic production of a complete simulation model in a manner that is transparent to the user. The work presented in the following sections can best be summarized by Barstow's statement that "the essence of [the knowledge-based approach to automatic programming] involves the identification of concepts and decisions involved in the programming process and their codification into individual rules" [Barstow 1977b p.388]. The following section presents an overview of the KBMC system and its design. The remainder of this paper presents a discussion of the issues involved with conceptualization (the identification of concepts) and implementation (the codification of knowledge into rules) for the KBMC system.

KBMC SYSTEM DESIGN

The design of the KBMC system is based on three research areas: simulation modeling, automatic programming, and expert systems. An automatic programming approach to simulation modeling has been implemented using the expert system building tool OPS83 [Forgy 1984; Forgy 1985]. OPS83 provides a rule-based production system paradigm that allows the incorporation of knowledge in the form of rules and an inference mechanism that activates or *fires* rules based on goals and information available within the system. Thus knowledge engineering has been used to develop the underlying KBMC rule base which uses program-transformation and goal-directed search for the automatic production of executable simulation models in the special purpose simulation language SIMAN for the domain of queueing systems. The basic concepts used by a simulationist for the construction of SIMAN models for a restricted subset of queueing systems have been identified and implemented in the representation of OPS83. A structured interactive dialog allows user specification of the desired model and a knowledge base containing specification extraction and model construction rules guides the dialog and constructs the complete model.

Design Overview

The traditional approach to simulation modeling, shown in Figure 2, involves a simulationist who evaluates the system to be simulated to obtain pertinent detailed information about the system and manually produces a program. This software is a computer-executable model of the system, implemented in a general purpose programming language or a special purpose simulation language.

Both the evaluation and programming tasks, performed manually by a simulationist in the traditional approach can be automated. Since a simulationist utilizes simulation modeling knowledge for both the specification acquisition and model construction tasks, this knowledge can be captured as part of a rule base. Thus, a knowledge-based automatic programming approach allows this combined automation of both the specification acquisition and the model construction processes.

An overview of the KBMC system which implements this knowledge-based approach to specification acquisition and simulation model construction is presented in Figure 3. Three distinct kinds of knowledge are used in the specification and construction of a simulation model. These are domain knowledge, in this case, knowledge of queueing systems; general simulation modeling knowledge; and target language knowledge, specifically knowledge of the simulation language SIMAN. This knowledge is incorporated into the underlying rule base of the KBMC system in the form of condition–action rules.

In general, domain knowledge and modeling knowledge are combined to form *extraction* rules, and modeling and target language knowledge are combined to form *construction* rules. The extraction rules guide the interactive session with the user, which takes the form of question-and-answer dialog or menu selection, in order to obtain an internal specification of the desired model. The construction rules then transform the internal specification into an executable simulation model in the target language.

The traditional approach to simulation model construction as shown in Figure 2 requires that a simulationist perform the evaluation and programming tasks to glean enough pertinent information about the system to be simulated and code a computer-executable model of that system. In contrast, the knowledge-based automatic programming approach shown in Figure 3 captures much of the expertise of a simulationist in a rule base. This approach and the resulting rule base then facilitate the automation of both specification acquisition and model construction, thus allowing a user with minimal knowledge of simulation techniques and who lacks programming language expertise to construct executable simulation models.

Design Requirements

The knowledge-based automatic programming approach to simulation model construction shown in Figure 3 is based on the capture of domain, modeling and target language knowledge. The design requirements of the KBMC system included four major elements that are intertwined with regard to their use of these three types of knowledge. The requirements for the KBMC system design:

- a specification method
- an internal representation
- a model construction method
- a rule base formed from the codification and implementation of knowledge using an expert system building tool

A brief introduction to each of these requirements is provided

System to be Simulated $\xrightarrow{\textit{evaluation}}$ Detailed Specification $\xrightarrow{\textit{programming}}$ Simulation Model

Figure 2. Traditional approach to simulation model construction.

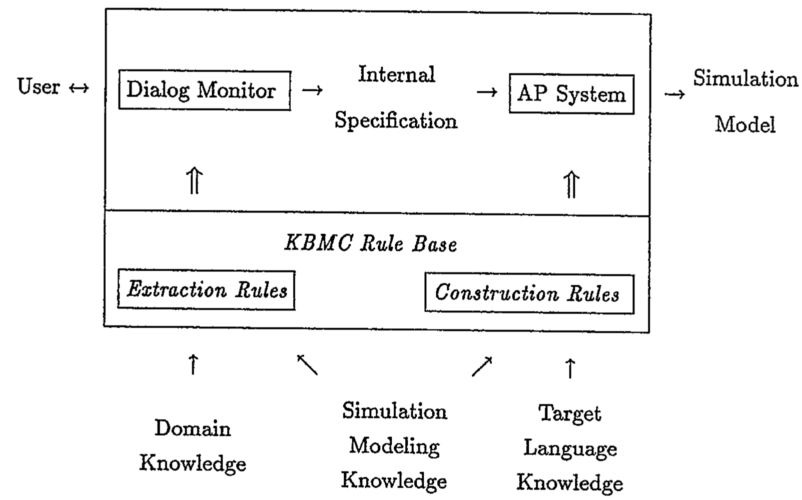

Figure 3. KBMC system overview.

in the remainder of this section.

The **specification method** selected for use in the KBMC system is that of a structured interactive dialog where extraction rules using domain and modeling knowledge guide the dialog and act as the dialog monitor. A model specification is *extracted* as the user answers questions posed by the system and makes selections from menus presented. The user is expected to be able to identify the major components of the system to be modeled, such as the mobile and fixed aspects of the system, and the nature of waiting lines or queues, as well as the information desired from the model. The user must also have access to various analysis information such as the expected time between arrivals and the expected service time. In effect, the user must describe *what* the model is to do in response to queries driven by the rules which guide the user interaction. Once the extraction process is underway, the order and content of questions posed and menus presented by the system are dependent on information previously provided by the user and on system "knowledge" of what information is required under the circumstances.

The **internal representation** to facilitate the model specification forms a central portion of the overall system. An Internal Model Specification (IMS) mechanism was developed to furnish a target-language independent set of generic data structures to allow a full specification of the desired queueing system simulation model. The IMS must permit specification of the model such that it provides a precise description of *what* the model is to do rather than *how* it is to do it.

Model construction requires a transformation from the IMS structures to the constructs of the target language SIMAN. This phase must be completely transparent to the user as the system uses construction rules, largely composed of modeling and target language knowledge, to guide the model construction process. As with the specification process, the order of rule usage for model construction is dependent on the information contained in the IMS structures and on goals established by previous rules.

The development of a **rule base** required the incorporation of each of the above components with the three types of knowledge shown in Figure 3 in the form of extraction and construction rules. Extraction rules are condition-action rules that guide the system interaction with the user and thus "extract" the specification of a model from the user via interactive dialog and menu selection. Extraction rules, formed by formalizing a combination of domain knowledge and simulation modeling knowledge, dynamically allocate and fill in appropriate IMS structures in order to provide a specification of the desired model. Construction rules that build the model, similarly formed from a combination of modeling and target language knowledge, transform the information in the IMS into a SIMAN model and experiment frame.

The design and development of the KBMC system to meet these four requirements was divided into conceptualization and implementation phases. Conceptualization encompasses the first three requirements, namely the design of an internal representation and the development of specification and construction methods. Several issues involved in conceptualization are discussed in the next section and the implementation phase which includes rule base development is detailed in the following section.

CONCEPTUALIZATION

The conceptualization phase of KBMC system development included the identification of the basic concepts used by a simulationist for the construction of simulation models. Due to the complexity of the general modeling task and the large number of languages, both special purpose simulation languages and general purpose programming languages available to a simulationist, it was first necessary to define and constrain the simulation modeling problem domain and to select a target language. The problem domain is taken to be a restricted subset of queueing system problems and the target language is the special purpose simulation language SIMAN. Conceptualization then involved the identification of concepts and relationships used by a simulationist when constructing models in the target language for the restricted problem domain. The rationale for the restricted domain and target language selection are presented below. Also included are overviews of the types of knowledge needed for automatic simulation model construction and the design of the internal representation.

Restricted Queueing System Domain

The domain of queueing systems was chosen because of its general applicability to real world situations and hence its popularity among simulationists. A wide variety of real world systems can be viewed as queueing systems which can be described by their arrival processes, the discipline and configuration of their queues, and their service mechanisms.

The domain of the KBMC system is defined and constrained to be a restricted subset of queueing systems that includes those in which an infinite arrival population behaves according to a single arrival process. This means that "customers" continue to "arrive" at one point in the system for the duration of time the system is to be simulated. The domain is limited to open systems, meaning that the transient objects of the system arrive, recieve one or several services, and eventually depart from the system. Systems within the restricted problem domain include those with multiple customer types, multiple servers and multiple types of service arranged in series or in parallel, and systems that contain branching customer paths or internal feed-back loops.

Target Language – SIMAN

SIMAN is a combined discrete-continuous SIMulation ANalysis language for modeling general systems [Pegden 1982]. This language was chosen as the target language of the KBMC system because it represents a progressive approach to simulation and modeling in that it stresses a fundamental distinction between the system model and the experimental frame. The SIMAN model describes the characteristics of the real system and the experimental frame describes the conditions under which the model is to be executed to produce the desired output information. Thus, the unique separation feature facilitates experimentation with a model since the simulationist need only vary parameters specified in the experimental frame while the model remains unchanged. While experimentation is an important aspect of simulation, the work described here focuses on the automatic construction of models rather than on experimentation with the models once they have been built.

Types of Knowledge

As illustrated in Figure 3, the three types of knowledge needed for automatic simulation model construction are domain, modeling and target language knowledge. **Domain knowledge** for the KBMC system is knowledge of the restricted subset of queueing systems. The major components of a queueing system are the arrivals, perhaps called customers, and the services or servers. The queueing system is described by its arrival process, its service mechanism and its queue discipline.

The arrival process is normally specified by the interarrival time, i.e. the time between arrivals, and the number of units that arrive at any one instant. The service mechanism is described by the time required to complete a service and the number of units serviced at one time. Both the interarrival time and the service time are usually specified by a probability distribution. When a customer arrives and finds that all servers are busy, the customer must wait until a server is available and thus a queue forms. The queue discipline describes the order in which the waiting customers are eventually served. Common queue disciplines are first-in-first-out (FIFO) and last-in-first-out (LIFO).

There are two categories of **modeling knowledge**, the general requirements for any model and the design decisions needed for the construction of a specific model. All discrete simulation models have several common characteristics. Models are composed of permanent entities that represent the fixed components or services provided, and temporary entities that represent the mobile components of the real system being simulated. All models have some mechanism for the creation of temporary entities, for handling queues, and for causing delays while services are administered. Each model requires some means of ending model execution, perhaps after a prespecified length of simulation time or number of arrivals. Finally, models have goals since they are built to produce information which might lead to an increased understanding of the system being simulated.

The model design decisions include the identification of permanent and temporary entities based on the components of the real system, and the determination of where to collect statistics in order to satisfy the goals of the model. The simulationist must also select appropriate constructs of the target language to implement the model.

Target language knowledge includes an understanding of the semantics of the target language, SIMAN, to allow the proper selection of constructs for model implementation. Knowledge of the language syntax is also necessary to produce an executable model. Since SIMAN requires the separation of the model and experimental frame, the target language knowledge must also include the distinction between model logic and experimental parameters.

Representation Methods

The design of internal representation methods to facilitate the model specification formed a major portion of the KBMC system development. The Internal Model Specification (IMS) furnishes a set of generic data structures used to represent the queueing system to be simulated and permits a full specification of the simulation model. The IMS allows specification of both the static and dynamic components of a real queueing system and thus provides a precise description of *what* the model is to do rather than *how* it is to do it. The static components include descriptions of the temporary and permanent entities, the necessary distribution information, model goals and model termination characteristics. The dynamic components are those that describe the flow and interrelationships of entities as they progress through the model.

The IMS is the heart of the knowledge-based model construction system. The system utilizes queueing and modeling knowledge to transform information obtained via interaction with the user into a specification, a set of filled-in IMS data structures. The system then applies modeling and target language knowledge to transform information retained in the IMS into the constructs and syntax of the target language to produce an executable simulation model. These transformations constitute the extraction and construction processes as illustrated in Figure 4.

$$\text{user} \xrightarrow{\text{extraction}} \text{IMS} \xrightarrow{\text{construction}} \text{model}$$

Figure 4. Transformations to and from IMS structures.

The internal model specification is built by extraction rules that dynamically allocate and fill in the appropriate individual data structures. For instance, if a server in a queueing system provides two types of service, two generic service_information structures, as shown in Figure 5, are created. Each structure is then filled in with the detailed information for one type of service. Consider for example, a bank in which one or more tellers process both general and business accounts. Since the time required for a teller to service a customer depends on the type of service needed by that customer, two service_information structures would be instantiated in the internal model specification and might look like those in Figure 6. This instantiation of generic IMS structures easily describes the static components of the queueing system and works quite nicely in the OPS83 paradigm of working memory.

```
service_information
   name               – name of the service
   perm_entity_id     – name of the permanent entity
                          providing this service
   queue_id           – name of the queue
                          preceeding this service
   distribution_id    – pointer to a distribution structure
                          for service_time information
```

Figure 5. Generic IMS structure for service specification.

```
service_information              service_information
   general_account_service          business_account_service
   teller                           teller
   teller_queue                     teller_queue
   general_service_time             business_service_time
```

a. general accounts   **b.** business accounts

Figure 6. Service_information structures for two service types.

The dynamic aspects of the queueing system, namely the flow and interactions of system components, describe the model logic and form the basis for the simulation model that is to be created. IMS flow information structures provide the link between each temporary entity and the services it needs, and thus implicitly depict the interactions between the temporary and permanent entities of the system. This flow information as obtained via extraction rules is used to form a basic model. Basic model structures, a subset of IMS structures, describe the activities in which a temporary entity engages and the permanent entities that are needed for each activity. The primary activities for a queueing system are arrive, service, branch and depart.

IMPLEMENTATION

The implementation phase of the KBMC system first required a formalization of the concepts and knowledge identified during conceptualization into general English-like or pseudocode rules to guide the specification extraction and the model construction processes. Once this was accomplished, the expert system building tool OPS83 was selected for implementation of the system.

The actual implementation involved encoding knowledge and the English-like rules in the rule-based paradigm of OPS83, and the enhancement and refinement of the rule base. The selection of OPS83 and the formalization and rule base development for the KBMC system are discussed in the remainder of this section.

Formalization

The three types of knowledge discussed earlier are incorporated into the underlying rule base for the KBMC system as extraction and construction rules. Extraction rules are condition-action rules that guide the system interaction with the user and thus "extract" the specification of a model from the user via interactive dialog and menu selection. Extraction rules are formed by formalizing a combination of domain knowledge and simulation modeling knowledge. For example, consider the following facts about queueing systems and simulation modeling.

queueing knowledge

- the arrival process is described by time between arrivals
- time between arrivals may be given by a random distribution

modeling knowledge

- temporary entities represent the mobile components of the real system
- the arrival of temporary entities is simulated by a creation mechanism
- a normal random distribution requires parameters for mean and standard deviation

These few facts might lead to the following specification extraction rules.

- If the name of the temporary entity is unknown, ask the user for the name of the objects that move through the queueing system.
- If the mobile components of the system are customers, the temporary entities are called customers.
- If nothing is known about the arrival of customers, ask the user to select the appropriate distribution to describe the time between customer arrivals.
- If the arrival distribution is normal, ask the user for the mean and standard deviation.

Construction rules that build the model are similarly formed from a combination of modeling and SIMAN knowledge. Consider these additional facts about simulation modeling and SIMAN.

modeling knowledge

- a possible model goal might be to produce information about the average time-in-system for temporary entities
- time-in-system information requires the use of a temporary entity attribute to retain the time that the entity arrives

SIMAN knowledge

- a SIMAN model CREATE block creates temporary entities
- a temporary entity attribute may be given a value by using a SIMAN MARK modifier
- the MARK modifier is given as MARK(n) where n represents the temporary entity attribute number

These new facts, taken with those above, can be formalized into the following model construction rules.

- If the arrival of temporary entities is to be modeled, a SIMAN model CREATE block is needed.
- If time-in-system is needed, add a MARK modifier to the CREATE block.

These brief examples illustrate the combination of three kinds of knowledge to form extraction and construction rules which make up the underlying rule base for the KBMC system. The extraction rules produce an internal specification of the desired model and the construction rules transform that specification into a complete simulation model.

Tool Selection

The formalization stage yielded English-like rules that were then implemented using an expert system building tool. Hayes-Roth, et al. [Hayes-Roth et al. 1983] suggest that the selection of an expert system building tool should be based on the nature of the task at hand. The KBMC system is a constructive system, i.e., one in which something is built from parts, in contrast to a diagnostic system that deduces parts from the whole as in the early medical diagnostic expert systems. A forward chaining inference mechanism is desirable for such a task, meaning that the system would work from facts that describe an initial state to a final state representing conclusions. For the extraction process the initial state is that of having no information about the system to be simulated, that is, no specification, and the final state is a complete specification. Likewise for the construction process, the initial state is that of having a specification and no SIMAN model, and the final state is a SIMAN model that satisfies the specification requirements. The design of the IMS structures and the internal representation of the model specification requires that the chosen tool allow some form of dynamic allocation of the IMS structures. Also, some capabilities for traditional procedural programming were deemed useful to easily handle such tasks as a conversion of time units for consistent distribution parameters.

Inherent throughout the conceptualization phase of system development and formalization of knowledge was the use of the rule-based paradigm. The condition-action form of rules allows a data-driven approach in which the actions previously completed have some influence on the actions to follow. A rule-based methodology also permited a goal-oriented approach that provides an overall structure for the extraction and construction processes.

OPS83, an expert system building tool developed by Charles Forgy at Production Systems Technologies in Pittsburgh, PA, was selected for the implementation of the KBMC system as it meets the criteria outlined above. It is an expert system language that provides a forward chaining inference mechanism and supports both rule-based and imperative programming. Furthermore, the OPS83 paradigm has a proven track record in a constructive problem domain with the highly successful work of John McDermott and the R1 system, now known as XCON, which configures VAX-11/780 equipment based on specifications of user requirements [Kraft 1984; McDermott 1982].

The three components of OPS83 are *working memory* in which *elements* are used to represent the current problem state, a rule base that is known as *production memory*, and an inference mechanism that invokes a *recognize-act cycle* to select and execute appropriate rules. Both goals and data are represented as elements that can be added, changed, or deleted from working memory. Thus working memory provides the capabilities for dynamic allocation of IMS structures and could be used to "build" a simulation model in a constructive problem domain.

OPS83 runs on a variety of hardware including a VAX-11/750, available for use in this research. Since SIMAN simulation programs also run on VAX equipment, an added benefit of an OPS83-based KBMC system is that programs produced could easily be compiled and executed on the same machine.

Rule Base Development

The underlying rule base of extraction and construction rules dictates the capabilities of the KBMC system in that the knowledge incorporated into the rule base must be adequate to allow the specification and construction of complete and correct models. Initially a kernel rule base was developed in the representation of OPS83. This "kernel" had sufficient knowledge to handle the extraction of information from the user and the construction of a complete model for a simple queueing system containing a single entity type and a single server.

Several simple models were produced by the system, compiled using the SIMAN model and experimental frame processors to check for syntactic correctness, evaluated for semantic correctness by checking for the correlation with the real system description and with hand-constructed models, and executed to show that model goals are satisfied. As the KBMC implementation proceeded, this kernel was tested and enhanced to include additional capabilities.

Once the capabilities of the kernel rule base reached a sufficient level of expertise for simple systems, the system was incrementally enhanced by adding supplementary rules to support additional cases typical of the problem domain. The characteristics of the test cases used throughout various aspects of the KBMC design and development are summarized in Figure 7. Note that the characteristics for each case presented in the figure include only those characteristics not previously listed for cases presented earlier in the figure. Complete system descriptions and the models produced for each of these cases are included in [Murray 1986].

Case		Characteristics
1	gas station	single temporary entity type single server, single service type exponential, normal distributions termination: total time stats: time in system
2	car wash	single server, multiple services in series multiple services use same distribution stats: waiting time service time
3	bank	multiple temporary entity types one line for multiple servers uniform distribution stats: server utilization termination: total clients
4	TV inspection	multiple server types stats: queue size branching looping
5	clinic	constant distribution
6	job shop	multiple branches and loops

Figure 7. Summary of cases.

The sequence of cases was specifically established to allow a progression from the simple to the more complex queueing system situations. Thus enhancements of the kernel rule base began with the addition of capabilities for a server who provides several kinds of service in series, as in a car wash, and continued to incorporate capabilities for the multiple branches and loops required in a job shop model. Additional enhancements were made to provide capabilities for producing comments to be included with the SIMAN model, and to build meaningful labels and identifiers that appear on the statistics report produced by executing the SIMAN model.

The rule base continued to grow throughout the progression of cases, but at a slowing rate. The overall finding is that as additional, more difficult cases were used to enhance the rule base capabilities, existing rules had to be tightened. This sometimes involved further restricting the condition sets of existing rules so that a rule could be fired only in appropriate situations. Other times it involved adding a rule similar to an existing rule with a slightly different set of conditions that invoked an entirely different set of actions.

CONCLUSION

The knowledge-based model construction system as described here represents an automatic programming approach to simulation modeling. The knowledge-based approach has been extended to include automation of specification acquisition. Implemented using the expert system building tool OPS83, sufficient knowledge has been captured and encoded in the form of rules to allow the specification and construction of a complete model for various test cases. The underlying knowledge base consists of extraction and construction rules that consolidate queueing system, simulation modeling, and SIMAN knowledge. Extraction rules guide the interactive dialog that allows user specification of a model. An internal model specification mechanism was developed to retain that specification. Construction rules produce the executable SIMAN model and accompanying experimental frame.

The design and development of the KBMC system proceeded in two phases. The first, conceptualization, involved the identification of concepts and relationships used by a simulationist for building queueing system models in the target language SIMAN. Conceptualization also necessitated the development of methods for obtaining from a user and retaining internally a model specification, and for constructing the desired model. The internal model specification or IMS was developed as a set of generic data structures that are instantiated as needed to permit a complete specification of a simulation model. The specification is obtained via an extraction process, meaning that an interactive dialog is conducted with the user and the specification is derived as the user answers questions and makes menu choices. Model construction is accomplished by goal- and data-directed transformation where the internal specification is transformed first into a basic model and then into SIMAN structures.

During the second phase, implementation, the concepts and relationships identified during conceptualization were first formalized into English rules in an if-then format. These rules, some based on queueing system and modeling knowledge, guide the extraction of information from a user to build a model specification. Other rules automatically construct and format the SIMAN model through the use of simulation modeling and SIMAN knowledge. The rules were implemented using the expert system building tool OPS83, selected due to its past success in a similar constructive domain and its suitable representation methodologies. Initially a kernel rule base was developed with the capabilities to extract a specification and automatically construct a model for a simple single server queueing system. The capabilities of the system were then extended by the addition of supplementary rules to support the specification and construction of models for more complex queueing system situations.

The remainder of this section summarizes some of the conclusions reached during the conceptualization and implementation phases of the KBMC system development as well as on the utility of the system.

Conceptualization

The KBMC system has illustrated the suitability of extending the knowledge-based approach to automatic programming to include specification acquisition via a structured interactive dialog. For a restricted queueing system domain, this approach to specification acquisition was found to have the following benefits:

- requires no simulation language or programming expertise from the user
- eliminates the need for a high-level specification language

The KBMC extraction rules successfully provide the facilities needed to allow a user to describe the queueing system to be simulated via interactive dialog and menu selection. This approach facilitates tailoring the system to its users in that the questions posed and menus presented by the extraction rules utilize terminology of the queueing system problem domain and information previously provided by the user. It has been concluded that the effectiveness of this knowledge-based approach to specification is subject to the following limitations:

- The domain must be clearly defined.
- Enough knowledge must be incorporated into the rule base to allow acquisition of a complete specification.

The extraction rules were successful in capturing queueing system and simulation modeling knowledge for a restricted domain. The specification extraction process allows the user to describe the system to be simulated and to provide all necessary information to enable the construction of a complete model.

The specification representation or IMS developed as an integral part of the KBMC system was shown to provide adequate capabilities and flexibility for storing the model specification. The IMS successfully provided a repository for specification information as it was acquired, and permitted acquisition of a complete specification to allow automatic construction of an executable model. The design of the IMS as a set of simple generic structures that are allocated and instantiated as needed proved to be an effective means of representing both the static and dynamic aspects of model specification information. The IMS structures were found to function well in the OPS83 paradigm that uses working memory as a global data base to represent the current problem state.

Automatic programming can be successfully implemented using a rule-based approach. In the KBMC system, the use of construction rules to capture simulation modeling and SIMAN knowledge, and the utilization of goals to structure the construction of the model, enable the actual model construction phase to be entirely transparent to the user. A basic model is built from the dynamic aspects of the specification information. Then appropriate SIMAN structures are selected and filled in to implement the basic model with SIMAN model blocks. SIMAN experimental frame elements are built as needed, and suitable SIMAN model blocks and experimental frame elements are added to implement statistics collection.

Furthermore, the rule-based construction method used in the KBMC system appears to support target-language independence. The development of a basic model that represents the dynamic aspects of the desired model and is separate from the target language dependent representation of the model will facilitate future expansion of the KBMC system for other target languages. The SIMAN-specific construction rules can easily be replaced by construction rules for another target language. The system handles general modeling functions such as temporary entity attribute usage and the use of accumulator numbers for statistics and file numbers for queues. The system also shortens permanent entity names if necessary, builds labels for branching constructs and identifiers for reporting statistics. The majority of these functions would remain intact for other target languages.

Implementation

Specific conclusions from the implementation strategy employed for the KBMC system relate to the use of OPS83 and the development of the rule base.

This work has shown that an expert system building tool provides an effective means of implementing a knowledge-based approach to automatic program (or model) construction. The rule-based production system paradigm of OPS83 worked nicely. Working memory provided the necessary capabilities for dynamic allocation of specification (IMS) and target language structures in a constructive domain.

As indicated by Hayes-Roth [Hayes-Roth et al. 1983; Hayes-Roth 1984], the development of knowledge-based systems is an iterative process. The case study approach proved to facilitate this process for the development of the KBMC system rule base. The use of a simple case with a single server and single entity type was an excellent starting point for rule base development. The kernel rule base contained sufficient rules for this simple case. Thus, the kernel included various fundamental aspects of simulation modeling and provided early and encouraging results by allowing the specification and construction of an executable, albeit simple, model. As rules were added for the increasingly complex characteristics of the progression of cases, the following observations were made:

- Fewer rules were added with each iteration.
- The addition of new capabilities frequently required further restriction of existing rules.
- Previously added capabilities needed to be retested.

Experiences with the KBMC system have established the utility of a knowledge-based approach to automated simulation model construction. The KBMC system is useful to both simulationists and nonsimulationists. The system provides fast results while freeing the simulationist from the details of the target language. Some familiarity with the language SIMAN allows the simulationist to easily vary experimental frame parameters and thus facilitates experimentation with the model produced by the KBMC system.

Operations research or statistics experience might prove useful to a nonsimulationist user of the KBMC system since the user must be able to identify the transient and fixed components of the system to be simulated and to provide information on interarrival and service time distributions. The system, however, frees this user from syntactic and semantic concerns of the target language as well as from all understanding of the mechanisms needed to carry out the simulation.

The KBMC system would allow simulation students to concentrate on learning system analysis techniques rather than syntactic and semantic details of a particular language. The capabilities of the knowledge-based approach for the automatic construction of executable simulation models facilitates rapid prototyping of simulation models.

REFERENCES

Heimo H. Adelsberger, Udo W. Pooch, Robert E. Shannon and Glen N. Williams. 1986. "Rule Based Object Oriented Simulation Systems." *Proceedings of the Conference on Intelligent Simulation Environments* (Paul A. Luker and Heimo H. Adelsberger, eds.) SCS Simulation Series 17:1 San Diego, CA (23-25 January) 107-112.

David R. Barstow. 1977. "A Knowledge Base Organization for Rules about Programming." *SIGART Newsletter* 63 (June) 18-22.

David R. Barstow. 1977. "A Knowledge-Based System for Automatic Program Construction." *Proceedings of the Fifth International Joint Conference on Artificial Intelligence* IJCAI. Cambridge, MA (22-25 August) 382-388.

David R. Barstow. 1979. "An Experiment in Knowledge-Based Automatic Programming." *Artificial Intelligence* 12:2 (August) 73-119.

David R. Barstow. 1979. "The Roles of Knowledge and Deduction in Program Synthesis." *Proceedings of the Sixth International Joint Conference on Artificial Intelligence* IJCAI. Tokyo, Japan (20-23 August) 37-43.

David R. Barstow. 1979. *Knowledge-Based Program Construction.* New York and Amsterdam: Elsevier North-Holland.

David R. Barstow. 1983. "A Perspective on Automatic Programming." *Schlumberger - Doll Research, Research Note, Project SYS-004* Schlumberger - Doll Research, Ridgefield, CT (14 November).

David R. Barstow. 1985. "On Convergence Toward a Database of Program Transformations." *ACM Transactions on Programming Languages and Systems* 7:1 (January) 1-9.

David R. Barstow. 1985. "Domain-Specific Automatic Programming." *IEEE Transactions on Software Engineering* SE-11:11 (November) 1321-1336.

Venkataseshan Baskaran and Y. V. Reddy. 1984. "An Introspective Environment for Knowledge Based Simulation." *Proceedings of the 1984 Winter Simulation Conference* (S. Sheppard, U. Pooch, D. Pegden, eds.) SCS. Dallas, TX (28-30 November) 645-651.

Charles L. Forgy. 1984. *The OPS83 Report – System Version 2.1.* Production Systems Technologies, Inc., Pittsburgh, PA (October).

Charles L. Forgy. 1985. *The OPS83 User's Manual.* Production Systems Technologies, Inc., Pittsburgh, PA (March).

Mark S. Fox. 1981. "The Intelligence Management System: An Overview." *Technical Report CMU-RI-TR-81-4* Carnegie-Mellon University, The Robotics Institute, Pittsburgh, PA (July).

Mark S. Fox and Y. V. Reddy. 1982. "Knowledge Representation in Organization Modeling and Simulation: Definition and Interpretation." *Proceedings of the Thirteenth Annual Pittsburgh Conference on Modeling and Simulation* (Vol 13) Pittsburgh, PA (April) 675-683.

Mark S. Fox, J. Mark Wright, and David Adam. 1984. "Experience with SRL: An Analysis of a Frame-based Knowledge-Representation." *Proceedings of the First International Workshop on Expert Database Systems* Kiawah Island, SC (24-27 October).

Frederick Hayes-Roth, Donald A. Waterman and Douglas B. Lenat (eds.). 1983. *Building Expert Systems.* Addison-Wesley, Reading, MA.

Frederick Hayes-Roth. 1984. "The Knowledge-Based Expert System: A Tutorial." *Computer* 17:9 (September) 11-28.

Elaine Kant and David R. Barstow. 1981. "The Refinement Paradigm: The Interaction of Coding and Efficiency Knowledge in Program Synthesis." *IEEE Transactions on Software Engineering* SE-7:5 (September) 458-471.

Philip Klahr and William S. Faught. 1980. "Knowledge-Based Simulation." *Proceedings of the First Annual National Conference on Artificial Intelligence* AAAI. Stanford, CA (18-21 August) 181-183.

Philip Klahr, William S. Faught, and Gary R. Martins. 1980. "Rule-Oriented Simulation." *Proceedings of the International Conference on Cybernetics and Society* IEEE. Boston, MA (8-10 October) 350-354.

Arnold Kraft. 1984. "XCON: An Expert Configuration System at Digital Equipment Corporation." *The AI Business* (Patrick H. Winston and Karen A. Prendergast, eds.) The MIT Press, Cambridge, MA 41-49.

A. Lehmann, B. Knödler, E. Kwee, and H. Szczerbicka. 1986. "Dialog-Oriented and Knowledge-Based Modeling in a Typical PC Environment." *Proceedings of the Conference on Intelligent Simulation Environments* (Paul A. Luker and Heimo H. Adelsberger, eds.) SCS Simulation Series 17:1 San Diego, CA (23-25 January) 133-138.

David McArthur and Philip Klahr. 1982. "The ROSS Language Manual." *Rand Report N-1854-AF* The Rand Corporation, Santa Monica, CA.

John McDermott. 1982. "R1: A Rule-Based Configurer of Computer Systems." *Artificial Intelligence* 19:1 (September) 39-88.

Malcolm McRoberts, Mark Fox, and Nizwer Husain. 1985. "Generating Model Abstraction Scenarios in KBS." *Proceedings of the 1985 Conference on Artificial Intelligence, Graphics, and Simulation* (Graham Birtwistle, ed.) SCS. San Diego, CA (January) 29-33.

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

C. Dennis Pegden. 1982. *Introduction to SIMAN.* Systems Modeling Corporation, State College, PA.

Y. V. Reddy and Mark S. Fox. 1982. "Knowledge Representation in Organization Modeling and Simulation: A Detailed Example." *Proceedings of the Thirteenth Annual Pittsburgh Conference on Modeling and Simulation* (Vol 13) Pittsburgh, PA (April) 685-691.

Y. V. Reddy and Mark S. Fox. 1982. "KBS (Knowledge Based Simulation): An Artificial Intelligence Approach to Flexible Simulation." *Technical Report CMU-RI-TR-82-1* Carnegie-Mellon University, Intelligent Systems Laboratory, The Robotics Institute, Pittsburgh, PA (14 September).

Y. V. Reddy, Mark S. Fox, Kevin Doyle, and John Arnold. 1983. "INET: A Knowledge Based Simulation Model of a Corporate Distribution System." *Proceedings of IEEE Conference on Trends and Applications 83* National Bureau of Standards, Gaithersburg, MD (May) 109-118.

Y. V. Reddy, Mark Fox, and Nizwer Husain. 1985. "Automating the Analysis of Simulations in KBS." *Proceedings of the 1985 Conference on Artificial Intelligence, Graphics, and Simulation* (Graham Birtwistle, ed.) SCS. San Diego, CA (January) 34-40.

Y. V. Reddy, Mark S. Fox, Nizwer Husain, Malcolm McRoberts. 1986. "The Knowledge-Based Simulation System." *IEEE Software* 3:2 (March) 26-37.

David Umphress. 1987. "Model Execution in a Goal-Oriented Discrete Event Simulation Environment." *Ph.D. Dissertation* Computer Science Department, Texas A&M University, College Station, TX (May).