ASSEMBLY OF SIMULATION NETWORKS USING DESIGNS, PLANS, AND METHODS

By I. D. Tommelein,[1] Associate Member, ASCE, R. I. Carr,[2] Fellow, ASCE, and A. M. Odeh[3]

ABSTRACT: An object-oriented and interactive computer system is presented that realistically models construction processes by matching resource properties with design component properties and operation durations. This system, named CIPROS, uses a modular representation to create discrete-event simulation networks, and to aid in relating simulation output back to the design and construction plan of a facility to be built. CIPROS users must identify and describe attributes of components to be constructed, based on the facility's design drawings and specifications, and they must develop a critical path method (CPM) plan. They must also select a construction method to perform each activity by retrieving the appropriate elemental simulation network from a library of networks that represent such methods. CIPROS then pieces together the networks based on sequential relationships from the plan and property values input from the drawing and specification data. The latter initialize the simulation network resources that make up the constructed facility. To complete the simulation network, users must specify the construction resources that are available to perform the work and which may be shared by activities. CIPROS comprises a fully operational discrete-event simulation engine that is called once a network is completed. Besides producing statistical reports that are instrumental in assessing the quality of the construction plan, CIPROS can also be used to check the degree of facility completion as the simulation progresses.

INTRODUCTION

Construction processes can be characterized by the properties of resources involved in their execution, and interactions between those resources. The extent and duration of such interaction typically carry a degree of uncertainty. Advance planning therefore warrants the use of simulation, which is a powerful method to imitate the behavior of a real-world system over time by modeling repetitive processes in which durations of operations are stochastic and many resources interact (Law and Kelton 1991). This generates statistical data to provide users with insight into the system's resource application, interactions, and constraints. Though discrete-event simulation has been around for many years and is well-suited to model construction operations, this technique has not gained widespread use in industry, in part because existing implementations did not represent many of the relevant characteristics of project components or construction resources, and because it is tedious to collect and assemble all required input data and to construct simulation networks.

Modeling construction processes is hampered by the wide variety of components that make up each facility. Although one-of-a-kind designs comprise many standard types of components (such as footings, beams, and columns, which represent *classes* of components), each individual component (termed a class *instance*) has property values (including its actual dimensions, material, and location in the facility; the number of reinforcing bars needed to build it; the amount of concrete to be placed in it; etc.) that represent the values of *attributes* of the instance. Such attributes are abstracted in most construction management models, e.g., when the critical path method (CPM) is used to create a construction plan, though they largely define what construction resources a contractor must bring to the site and how long individual activities will take. A realistic construction process model must therefore use this detailed but otherwise disparate construction management data, as is provided in design drawings, specifications, precedences among activities, available contractor resources, and typical construction methods. This can be done by linking disparate data sources to one another, across their levels of abstraction, to form a comprehensive unit. It is then necessary to create a discrete-event simulation engine that can use individual component characteristics without unduly complicating the discrete-event model formulation task.

The aim of the research presented here was to realistically model construction processes by matching resource properties with design component properties and operation durations. This is done by tying the discrete-event simulation model to data extracted from other construction management models, namely design drawings, specifications, construction methods, and CPM project plans. Such an integration was not possible until the advent and widespread use of object-oriented programming, which makes it easy to represent concepts such as classes, instances, and attribute values, like those mentioned earlier. Now that sophisticated development tools are available on personal computers, the technology has become affordable to construction researchers and practitioners alike.

RELATED WORK

Process Planning

Cyclone (Halpin and Riggs 1992) is probably the best-known discrete-event simulation system used in construction. It has powerful modeling capabilities that are based on a very small, comprehensive set of primitives. In the last decade, several stand-along, personal-computer-based versions with Cyclone capabilities have been developed, including Insight (Kalk 1980), MicroCyclone (Lluch and Halpin 1982; Halpin 1989), and UM-Cyclone (Ioannou 1989a; 1989b).

To advance the state of the art of construction simulation, some researchers have focused on modeling specific characteristics of the construction domain. Carr (1979) modeled projects of related activities sharing resources and weather in MUD. Morua Padilla and Carr (1991) extended MUD to include costs and allocation of resources in DYNASTRAT. Bernold (1989) addressed non-steady construction processes. Some have extended the expressiveness of primitives to model resource characterization and tracking [RESQUE by Chang (1986)] as well as flexible calendars [COOPS by Liu (1991)]. Others have integrated simulation with videotape data collection (Paulson et al. 1987), added a user interface to enhance data entry (AbouRizk and Halpin 1992), and provided interactive graphics to facilitate model definition (Paulson 1978; Riggs 1987; Liu 1991; Huang and Halpin 1993).

In efforts to incorporate existing construction data and knowledge into the modeling environment, discrete-event simulation has been integrated

---

[1] Asst. Prof., Civ. and Envir. Engrg. Dept., Univ. of Michigan, Ann Arbor, MI 48109-2125.

[2] Prof., Civ. and Envir. Engrg. Dept., Univ. of Michigan, Ann Arbor, MI.

[3] Asst. Prof., Civ. Engrg. Dept., Jordan Univ. of Sci. and Technol., Irbid, Jordan.

Note. Discussion open until May 1, 1995. To extend the closing date one month, a written request must be filed with the ASCE Manager of Journals. The manuscript for this paper was submitted for review and possible publication on August 16, 1993. This paper is part of the ***Journal of Construction Engineering and Management***, Vol. 120, No. 4, December, 1994. ©ASCE, ISSN 0733-9364/94/0004-0796/$2.00 + $.25 per page. Paper No. 6788.

with more traditional construction planning tools (Dabbas and Halpin 1982) and with more expressive knowledge-based environments (Shannon et al. 1985; Reddy 1987; Lehmann 1988; Touran 1990). Such integration is desirable when one sets out to realistically model all variables that play a role in project execution and management decision-making. Left undone, however, was the creation of flexible discrete-event simulation models that are truly capable of taking into account individual resource characteristics without making the creation of the simulation model too complicated.

Construction Planning and Scheduling

Research involving the use of artificial intelligence to automate planning and scheduling set out to meet a similar objective. This has resulted in layered knowledge-based systems that integrate the generation of computer-aided design (CAD) drawings with construction planning, scheduling, and estimating [PLANEX by Zozaya et al. (1989) and BUILDER by Cherneff et al. (1991)]. The effort of the research presented here is much along the same line, except that it extends the integration by including discrete-event simulation as a way to represent stochastic activity durations, model resource availability and interactions, and to provide statistical feedback, thus allowing plan construction and construction method selection to be validated. More importantly, it is by means of this integration that realistic characteristics of individual resources are made available for use in the discrete-event simulation model. In addition, a key aspect of easing this integration is to develop an extended notation that tersely captures all model characteristics. Such a notation is also presented in this paper. The implementation of this extended integrated model is named CIPROS, which is a pseudoacronym for "construction integrated project and process planning simulation" system.

OBSERVATIONS SUPPORTING MODEL FORMULATION

The methodology developed to realistically model construction processes was based upon the following observations. The characteristics of resources in a simulation network are determined either by the architect-engineers who design the facility to be built, or by the contractors who do the construction. Accordingly, one can distinguish two types of resources that occur in network queues, namely, product components and construction resources. *Product components* relate to design element classes or instances thereof, and their attributes and values. They make up the completed facility. Architect-engineers communicate this information to the contractor by means of plans and specifications. *Construction resources* relate to temporary equipment and materials that are used throughout construction but that do not make up the completed facility. These are determined by contractors who perform the work. By tapping the information embedded in designs and specifications and in data describing a contractor's capabilities and choices, a simulation system can determine the characteristics of individual components and resources that flow through the discrete-event simulation network.

Specifying discrete-event simulation networks consists of identifying process operations and associated resources. Often, such operations and resources appear in groupings that represent alternative construction methods to perform activities, which are selected during estimating. The interaction between such grouped resources is fairly standard, and it can therefore be captured by means of elemental discrete-event simulation networks. Many variables describing elemental networks can be initialized with default values or values retrieved from other construction-management models.

To ease network construction, one can view an activity-level plan as a skeletal simulation network, where each activity needs to be fleshed out by choosing a construction method to perform it. When methods are predefined as elemental simulation-networks, it is easy to select alternative ways to conduct the work and to sequence methods based on activity-level precedences. Because plans directly relate to construction drawings and specifications, this relationship can also serve to initialize the simulation networks.

The process-level detail of methods also makes it possible to represent resources that are shared among activities and thus to characterize correlated activities. Simulation then provides data beyond what can be generated by PERT-like models. It becomes a means for validating method selection and studying the effect of sharing resources among activities.

MODELING PROCEDURE

Creating a discrete-event simulation model using CIPROS includes seven consecutive steps that use CIPROS's knowledge-based representation and simulation primitives. These items, shown in parentheses, are explained later in the paper.

Step 1—define project design and specifications: The CIPROS user specifies a plan to build a facility by entering design drawing and specification data in object-oriented format. (These data will fit within CIPROS's Project Taxonomy and Component Class Hierarchy.)

Step 2—create activity-level plan and relate activities to the design components: The user specifies the activities needed to build this design by entering the activities by number, name, and scope, and by showing their precedence relationships.

Step 3—choose construction method for each activity in the plan: For each activity, the user then selects a construction method. (Methods are defined by process charts and operations functions.)

Step 4—initialize product components: CIPROS uses the design and specification data to initialize the product components that relate to activities in the plan. (They will initialize CInQues in the method that was chosen to perform the activity.)

Step 5—identify construction resources: The user provides the type and quantity of construction resources to be used in each method. (This is done by linking the RInQues from each chart to elements in CIPROS's Resource Class Hierarchy, so that the appropriate attributes can be inherited and used in simulation.)

Step 6—custom-tailor simulation network: CIPROS pieces the elemental simulation networks that describe construction methods together in the sequential order defined by the construction plan. Generally, each process has its own operations and queues. However, if operations of different processes share the same queues of resources, the user must define these and assign priorities.

Step 7—simulate and interpret results: The construction plan is simulated for a user-defined number of replications.

EXAMPLE APPLICATION

Consider the construction of a simple concrete structure that has:

- Two types of footings (Footing-1 and Footing-2)
- Two types of columns (Column-1 and Column-2)
- Two types of walls (Interior-Wall-1 and Exterior-Wall-1)
- One type of slab (Slab-1)
- A single type of roofing (Roof-1)

The individual components have additional characteristics. For example this project has eight components of type Footing-1 and four of Footing-2. Each footing of type Footing-1 is made of Concrete-4, includes five elements of Rebar-1 and seven of Rebar-2, and is 0.6 m long by 0.3 m wide by 0.2 m high. Each footing of type Footing-2 is also made of Concrete-4, but includes eight elements of Rebar-1 and 10 of Rebar-2, and is 0.4 m by 0.4 m by 0.2 m. Columns, walls, slab, and roof would have their own characteristics.

Estimating rules determine that 0.054 m^3 of soil must be excavated for each footing of type Footing-1 and 0.048 m^3 for those of Footing-2 (a rule of thumb might be to take 1.5 times the volume of the footing). Similarly, 0.0378 m^3 of concrete will fill each footing of type Footing-1 and 0.0336 m^3 each of type Footing-2 (5% more than the calculated volume of each element). Also, two forms of type Form-1 and two of type Form-2 will shape each Footing-1, while four of Form-3 will shape Footing-2.

To model the placement of rebar and concrete using a traditional simulation model, a user must construct a process diagram with several resource queues, because elements in a queue are considered to be identical for modeling purposes. If Rebar-1 and Rebar-2 are not identical, one queue should hold Rebar-1 and another one Rebar-2, despite the fact that all pieces of rebar may have to go through similar processing steps such as handling and placing. A single queue might be used to hold the resource Concrete-4 with a total amount of 0.4368 m^3, but some counter mechanism must be crafted to guarantee that the exact fraction thereof is placed in each Footing-1 and the other fraction in each Footing-2. Duplication of queues that hold similar components, and the concoction of a counter mechanism that uses a combination of the few available primitives has been avoided in CIPROS.

CIPROS KNOWLEDGE BASES

CIPROS uses an object-oriented representation to distinguish between individual components at simulation run time. The CIPROS discrete-event simulator handles distinctions on an as-needed basis, according to which class resources belong to and what sets individuals apart in each class. This greatly simplifies model formulation, realism, and reusability. To achieve these results, simulation input must be specified in knowledge bases that contain project design data, individual component data, and construction resources.

Project Taxonomy

The aforementioned example can be cast in a hierarchy that describes the design elements in the project, which is termed the project taxonomy [Fig. 1(*a*)]. Some attribute values, such as unit dimensions of a component, must be input explicitly. Other attribute values, such as volume of concrete per component, can be specified as a formula (sometimes called a method or a demon) that computes this volume based on the component's dimensions.

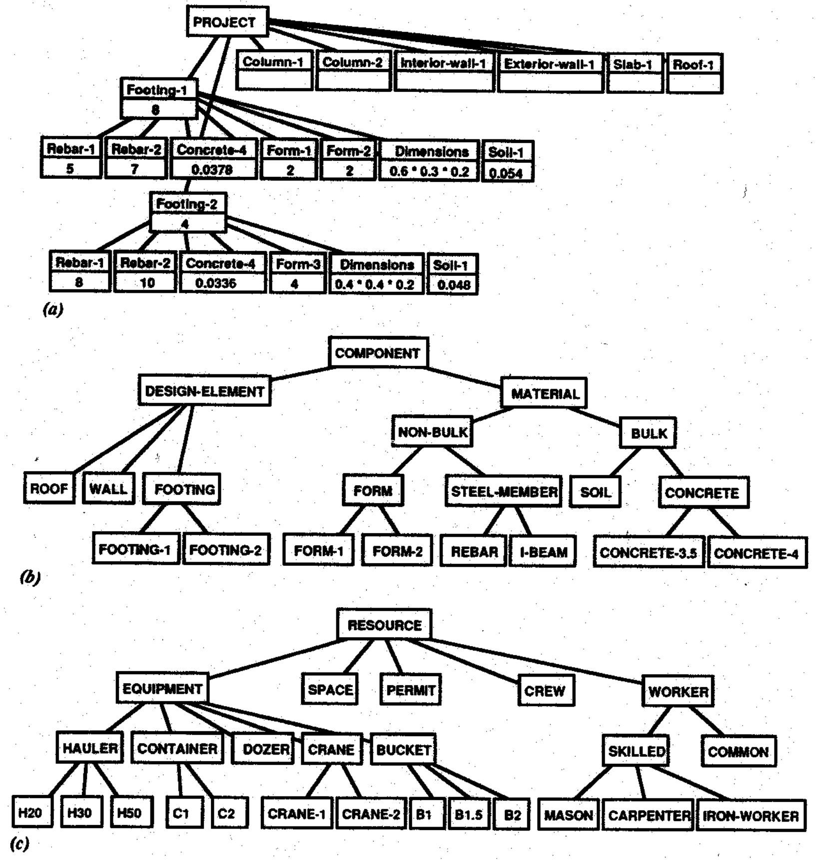


FIG. 1. Hierarchical Knowledge Bases: (*a*) Project Taxonomy; (*b*) Component Class Hierarchy; and (c) Resource Class Hierarchy

Class Hierarchy of Product Components

For each component, the user must specify the CIPROS generic component class hierarchy to which the component belongs. For example, Footing-1 and Footing-2 belong to the class Footing [Fig. 1(*b*)]. As will be clarified later, this specialization is necessary for CIPROS to apply its knowledge of construction methods for building generic components, to those specific project components (these components will initialize CInQues).

Class Hierarchy of Construction Resources

CIPROS also contains a class hierarchy of contractor resources that can be used in construction activities [Fig. 1(*c*)]. They are classified as labor; equipment; or materials in support of construction, such as carpenters, cranes, and form work. Labor refers to individual workers or to crews composed of workers with the same or different skills. Equipment includes both the machines themselves and attachments for them because attachments often determine the machine's functionality. Many equipment attributes are found in performance handbooks. The user can specify the contractor resources to be used for construction, to which CIPROS will then apply its knowledge of construction methods using those specific construction resources (they will initialize RInQues).

Activity-Level Construction Plan

The construction of a CIPROS simulation network is based on fleshing out the activity-level construction plan. A traditional way to plan construction work is to break a project down with a manageable level of detail into activities. A common convention is that activities pertain to a certain type of work and involve the continuous application of selected resources. Activities are sequenced with the traditional finish-to-start relationships to form the project plan.

Fig. 2 represents a possible plan to construct part of the example project. The space required for each footing will be excavated. Then, formwork and reinforcing will be placed. Following this, each footing will be cast. Of course, the time it takes to perform any of these activities for any individual footing will depend on that footing's attribute values. For example, it will take longer to place rebar in footings of type Footing-2 than in those of type Footing-1 because more pieces of rebar are involved.

The scope of an activity refers to the project components to which the activity applies and is defined by an ordered list of component class(es), each with its number of units. These components must exist in the component hierarchy. In Fig. 2, the user has placed all of Footing-1 and Footing-2 in the same activities. Each activity must thus complete all of Footing-1 and Footing-2 before a following activity can start (this is the finish-to-start precedence relationship).

After determining the topology of the activity network, the user must select a construction method for each activity, select the construction resources needed by each method, and assign activity durations. CIPROS will use its previously defined hierarchy of construction resources to retrieve the attributes that are relevant to each process.

Library of Construction Techniques and Methods

Construction methods form the link between project and process planning. Methods are fairly standard, because most construction work is routine. For example, concrete delivery, transport, and placement operations are innate to the process of placing concrete. The sequencing of these operations in the process is basically the same for footings, first-level columns, or second-level columns, regardless of the type and quantity of concrete in each footing or column, though their durations may vary. The actual quantities can be determined at run time, and duration distributions can be selected when concrete placement is being simulated.

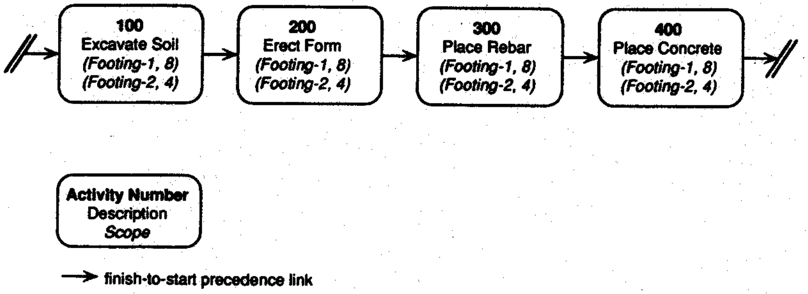


FIG. 2. Activity-Level Construction Plan

Accordingly, methods are predefined in a CIPROS knowledge base by means of elemental discrete-event simulation networks, comprising a process chart and operation functions. Processes have input and output queues, with inputs and outputs that are initially specified only at an abstract class level. This is the key to realistic process modeling in CIPROS: these classes will be instantiated at run time, based on information drawn from the actual design drawings and specifications and the construction resources provided by the contractor, in the order given by the construction plan. The operation functions will be applied to the inputs of the network and their evaluation might be based on those inputs.

Standard processes are classified in a knowledge base of *construction techniques* such as concrete placing, rebar placing, and form erecting. For each technique, the knowledge base includes processes representing different *construction methods* that specify which and how construction resources will be used to do the work. For example, the concrete placing technique can be applied by means of one of several methods, such as crane-and-bucket, concrete-pump, or buggy placement. An example crane-and-bucket method is shown later, after the CIPROS graphical notation has been explained.

PROCESS CHARTS

A process chart is a network of modeling elements (operations, queues, and arcs) that describes entity states and flows in a process. The generic term *entity* is used here to denote a unit such as a product component or a construction resource that flows in simple or compound form through the simulation network. A *set* is defined as an assembly of one or more entities in a strict hierarchical structure. That is, an entity may have more than one child but can have only one parent. For example, a hauler and a steel member each represent a set. A hauler loaded with three steel members is a set, in which the steel members are children of the hauler entity. The set is referenced by the set header entity, here, the hauler. The CIPROS simulator tracks the set's structure and the characteristics of each entity in the set throughout simulation.

The modeling elements that make up the process chart cannot themselves depict all relevant information. Operation functions that are separate from the process chart are used to define entity tasks and interactions and to control entity flow in the process.

CIPROS Modeling Elements

The CIPROS modeling elements carry the same names as Cyclone's, but their functionality is extended (Fig. 3).

"Normal Operation" (Normal) represents an unconstrained operation that does not require entities from more than one immediate source. A Normal is equivalent to a logical "or" in a network path, because it is activated upon arrival of any entity routed to the Normal. A Normal is represented graphically by a rectangle.

"Combination Operation" (Combi) represents an operation constrained by a combination of entities from more than one immediate predecessor. A Combi is equivalent to a logical "and" in a network path, because it operates on a user-defined combination of entities of different types called

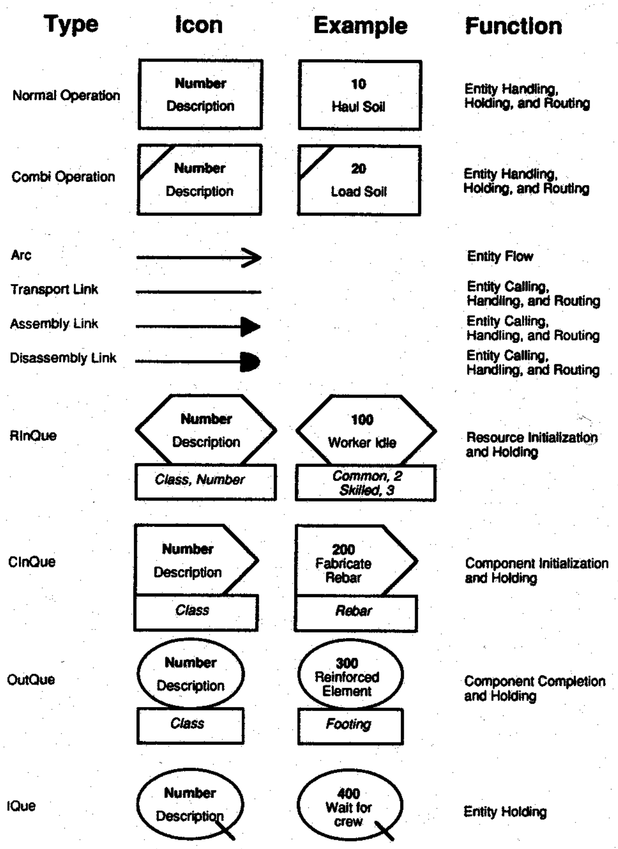


FIG. 3. CIPROS Modeling Elements and Functions

the Combi ingredients. A Combi can only be activated when all its ingredients are available. It is represented graphically by a rectangle with a diagonal line at the top left corner. In CIPROS, Combis have operation functions. These pertain to determining which number of entities must be withdrawn from a preceding Queue, how long an entity should be captured by the Combi, etc. These operation functions are defined with the links that connect a Queue to the Combi. They are described with the handling tasks, later in this paper.

"Arc" represents the flow path of entities released from an operation (i.e. Combi or Normal). An Arc is represented graphically by a line with an arrowhead to show the direction of flow.

"Link" represents entity flows from Queues and depicts entity requirements and entity tasks that will be applied to the entities joining the Combis that follow. Links are classified based on these entity tasks.

- "Transport Link" is represented by a line.
- "Assembly Link" is represented by a line with a black arrowhead.
- "Disassembly Link" is represented by a line with a rounded head.

These tasks and associated operation functions are described in more detail later.

"Queue" represents an idle state of entities or sets. Process delays occur when entities at one Queue wait for entities to arrive at other Queues to start their common succeeding Combi operation. Besides representing a wait state, Queues also serve two other functions. First, Queues are used to match construction resources to product components at the start of the simulation process. Second, Queues are used to track the process status. This enables CIPROS to schedule succeeding processes. To serve these functions, Queues are classified as follows.

"Resource Input Queue" (RInQue) represents a Queue of entities, named input resources, such as the construction resources a contractor brings to the site (e.g., a crane or form work). These entities set the production rate of Combis and Activities but do not get consumed in the process. An RInQue is graphically represented by a hexagon with a rectangle under it. The construction resources are defined by ranked pairs of their class(es) and their number, as shown in the rectangle. Resources of different classes are initialized in the queue in the order of their listing.

"Component Input Queue" (CInQue) represents a Queue of entities, named input components, such as the design elements specified by the architects or engineers (e.g., concrete and rebar to be placed in a beam). These entities generally get handled, assembled, or converted in the process. A CInQue is graphically represented by a pointed box with a rectangle under it. The input components of a CInQue are defined by their type, as shown in the rectangle. Only one type of component can be defined in each CInQue.

"Component Output Queue" (OutQue) represents a Queue of entities, named output components (e.g., a beam that has been cast). These entities result from performing the process. An OutQue is graphically represented by an oval with a rectangle under it. The output components of an OutQue are defined by their type, as shown in the rectangle. Only one type of component can be defined in an OutQue. An OutQue provides a counter that updates the process completion status whenever a component of the defined type joins this Queue.

"Intermediate Queue" (IQue) represents a Queue that is neither an input nor an output. In contrast with other Queues, which are strictly typed, an IQue can hold either resource or component entities. An IQue is graphically represented by an oval with a diagonal line across its lower right side that forms the letter "Q."

CIPROS process charts are small, comprehensive units that show how construction resources can be applied to perform a certain activity. In short, they describe *construction methods* as elemental simulation networks. Examples will be shown later. These charts depict the static structure of the processes. They are complemented by operation functions that define the dynamic nature of the processes.

Operation Functions

Operation functions specify entity interactions and they control entity flows during simulation. They allow the user to model different process strategies without altering the process network. Process strategies are generally related to entity use or entity routing. An example of a strategy is "use largest available hauler" in a loading operation. In the case of a Normal operation, these functions are attributes of the operation and thus apply to all entities joining the Normal. However, in the case of a Combi operation, where entities from different Queues are involved, the functions (except the operation duration) are attributes of the Link between the Combi and a preceding Queue. The operation functions in CIPROS are as follows.

"Operation Duration" defines the random variable representing the duration for each operation activation. This variable can depend on attributes of entities involved in the operation (e.g., the time to load a hauler depends on the hauler size). Durations are specified as probability distributions (e.g., normal, beta).

"Set Destination" defines the route to be followed by an entity or set after its handling in operations. A set can be routed to a process modeling element, to another set (i.e., it can be assembled), or it can be destroyed (i.e., it will no longer be tracked). A route to an element is depicted by an arc emanating from the operation, and it can be part of a probabilistic fork.

"Set Calling Priority" defines the preference by which an entity or set is drawn from a Queue. Examples are first in first out (FIFO), last in first out (LIFO), or a user-defined function of the set header attributes (e.g., take largest hauler first).

"Handling task" defines changes to a set structure in an operation. Three handling tasks can be modeled in CIPROS.

- "Transportation" refers to the movement of a set (e.g., a bundle of rebar) without altering its hierarchical structure. The task is a function of the calling number or quantity of sets to be drawn from the preceding Queue (e.g., take three at a time).
- "Assembly" refers to the aggregation of sets from different Queues into a new set (e.g., loading steel members to a trailer-truck). Sets to be assembled are those whose destination is the assembly base. The set to which these sets are assembled is called the base set. Sets can be assembled to the base set header (e.g., the truck) or to any of its nonbulk descendants (e.g., one of the trailers), called the assembly bases. The assembly task is a function of the type of base(s) (e.g., trailer size), and the type and number or quantity of sets to be assembled to the base(s) (e.g., steel member size). A base set is held in the operation until the defined number or quantity of sets are assembled or the capacity of the base(s) included in the set is exhausted. Because the number or quantity of sets that are assembled in each operation activation is defined by another task (i.e., transportation or disassembly), a base set may be held idle in a Combi operation waiting for subsequent activation(s).
- "Disassembly" refers to the segregation of a preassembled set (e.g., unloading steel members from a hauler). The set from which entities are disassembled is called the base set (here, the hauler). Entities can be disassembled from the base set header or any of its descendants, called the disassembly bases. The disassembly task is a function of the type of base(s), the type and number or quantity of sets to be disassembled from the base(s), and the disassembled set's destination (e.g., unload *X* elements of size *Y* from the hauler at bay *Z*). Entities can be disassembled in one or more operation activations, depending on the number or quantity of entities at hand. The base set will only be released from the operation when the defined number or quantity of entities are disassembled. Therefore, a base set may be held idle in a Combi operation while waiting for subsequent activation(s).

"Resource allocation and capturing" defines the type and maximum number of transported resources to be allocated or captured by an assembly or disassembly task. This allows parallel processing of multiple sets within the same operation. On one hand, captured resources are locked into the task and can only be freed upon task completion. On the other hand, allocated resources are freed and routed to their destination at the end of each operation activation.

Examples of how these operation functions can be used follow the description of the remainder of CIPROS's knowledge environment.

DISCRETE-EVENT SIMULATOR AND SIMULATION OUTPUT REPORTS

CIPROS simulates the project construction plan as follows.

- It identifies activities in the project plan that can start and their corresponding methods. If concurrent activities share resources, precedence goes to those with lower number.
- It identifies the product components and construction resources to initialize the CInQues and RInQues in each elemental process network representing the chosen methods.
- It simulates the processes. That is, CIPROS retrieves the appropriate attribute values needed to withdraw elements from the Queues, activates Normal and Combi operations according to these values, and accordingly samples durations from the probabilistic distribution of each operation duration. When a process is terminated, the status of components that appear in the OutQue is updated. The activity modeled by this process is herewith terminated and its successors will start (i.e., their queues are initialized). In this way, CIPROS steps through the entire construction plan, until no more activities can be performed.

Note that components do not flow between processes but are initialized when an activity starts. This is the essential link between the process simulation and the CIPROS knowledge bases. Resources are initialized at the earliest start of the activities whose processes compete for them. The activity products are project components that are routed to output queues and result in activity completion. The initialization of these resources and input components triggers the start of the process by making available the entities required for process operations.

The simulator tracks the state of product components, construction resources, and activity precedence throughout simulation. Process statistics are similar to those produced by Cyclone and provide information related to the process operations, tasks within processes, and queues, and thus the involved resource utilization, delays, and bottlenecks.

The statistical reports pertain to the last replication and the overall simulation. At the activity level, they report start and finish times of each activity and its process, and the activity's duration. These output reports are instrumental in measuring the quality of a construction plan and help identify areas for improvement.

CIPROS MODEL OF EXAMPLE APPLICATION

The following example shows how a user can assemble a simulation network using the CIPROS modeling components. First, steps 1 and 2 are executed. Then steps 3–5 are repeated for each activity in the plan. Activities 300 and 400 are modeled in detail to demonstrate method selection and specialization. Finally, steps 6 and 7 conclude the modeling process before simulation can take place.

Preparation Steps

Step 1: Fig. 1 shows the project design and the component attributes and values that were entered by the user.

Step 2: Fig. 2 shows the user-defined activity-level plan for the example project.

Modeling of Activity 300 Place Rebar

Step 3: The CIPROS user selected from the library of elemental methods, or custom designed a method to perform each activity, e.g., Activity 300 pertains to *Place Rebar* and the method *Place Rebar with Worker* fits this requirement (Fig. 4). By convention, all process modeling elements are numbered to start with the same digit as the activity.

Step 4: The scope of Activity 300 is (*Footing-1,8*) and (*Footing-2,4*). The Project Taxonomy components that correspond to this scope are therefore linked to the method's process chart for initialization of queues, e.g., 8 of Footing-1 and 4 of Footing-2 are initialized in CInQue 301 Design Element (Fig. 4). CIPROS also initializes the second CInQue in the process chart. Because this CInQue 320 refers to the class *rebar*, CIPROS consults the project taxonomy to find the rebar that is used in all Footing-1 and Footing-2, i.e., the rebar that is part of the components that initialized CInQue 301.

Step 5: RInQue 350 Worker Idle is initialized by the user, who chose to involve two common laborers and two ironworkers.

Explanation of Place Rebar with Worker Process

The process logic of *Place Rebar with Worker* describes that a worker will be drawn from Worker Idle (RInQue 350) to Transport Rebar (Combi 330) as soon as the Fabricated Rebar (CInQue 320) is available. This worker is either a common laborer or an ironworker, because both of these are in the *worker* class. An ironworker will be called each time Place Rebar (Combi 360) starts. This is shown by the label adjacent to each link. The ironworker assembles the Rebar Ready (IQue 340) to the Design-Element, as is shown by the Assembly Link emanating from CInQue 301.

After finishing their respective operations, the worker (common laborer or ironworker) and ironworker rejoin their RInQue, whereas the footing with rebar placed in it joins the OutQue. When the same number of footings specified in RInQue 301 are counted in OutQue 370 and all operations in this process are completed, the process itself is completed and the activity is finished. This allows the successor (here, Activity 400) to start.

Modeling of Activity 400 Place Concrete

Step 3: In a similar fashion, the user selects the appropriate construction method for Activity 400 (Fig. 5). The process chart *Place Concrete with*

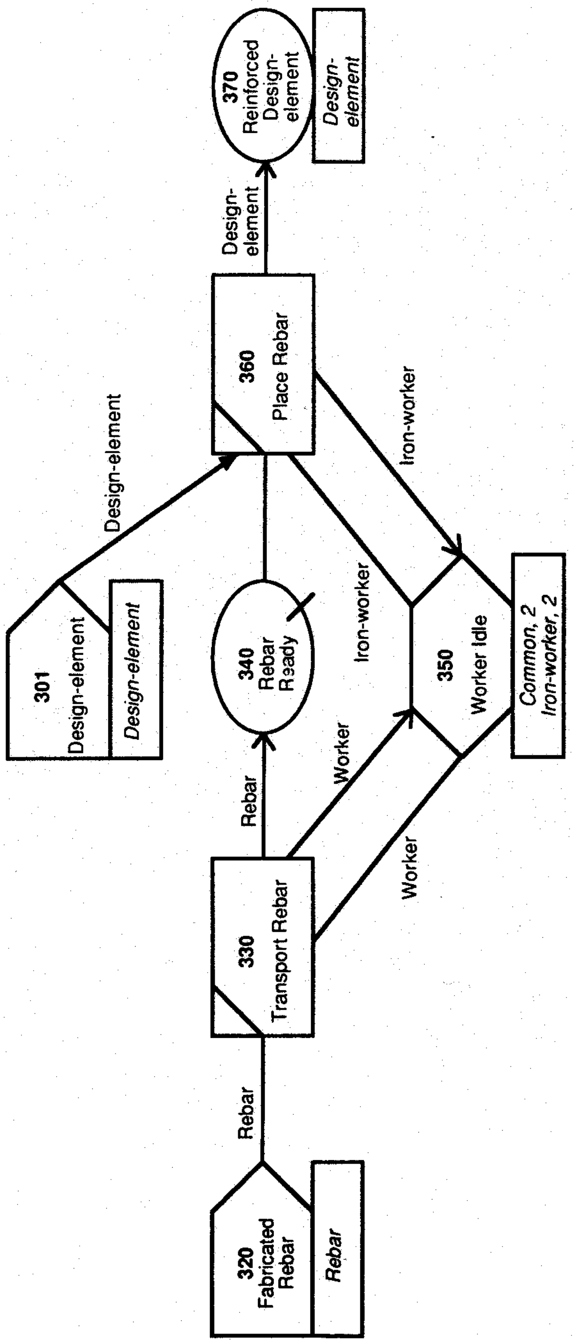


FIG. 4. Process Chart for Place Rebar with Worker Method

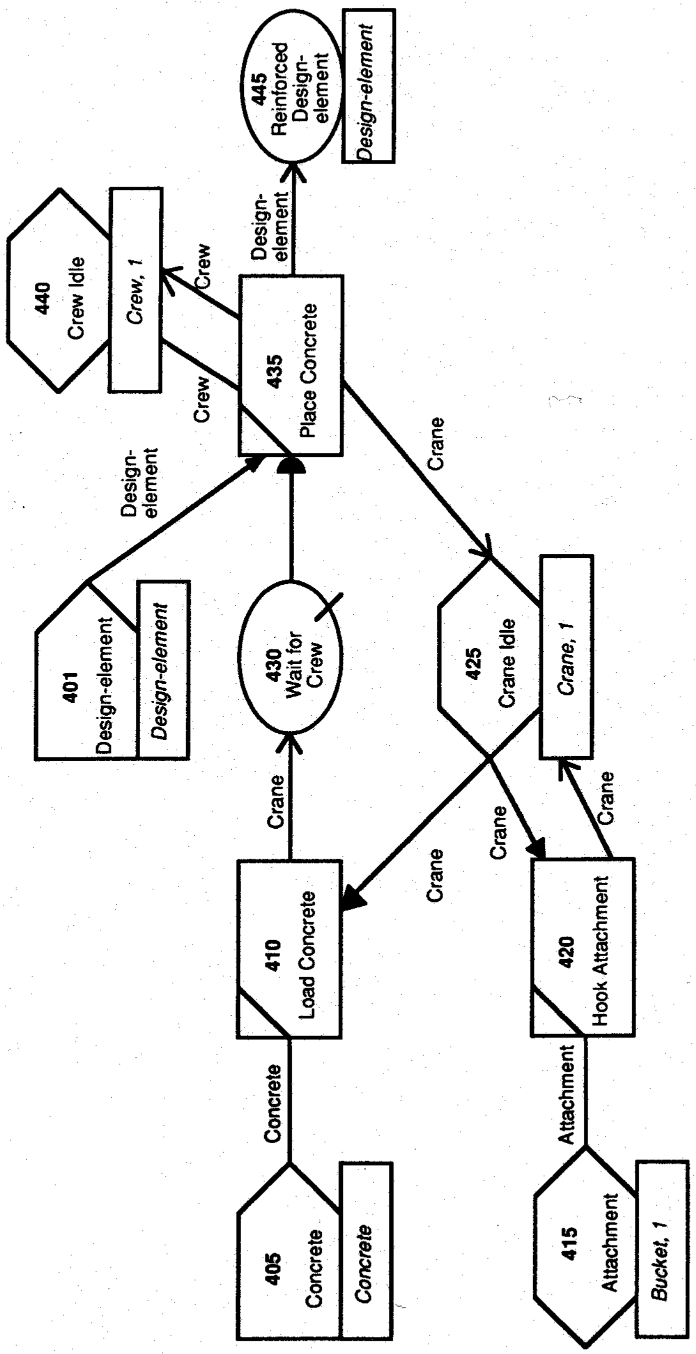


FIG. 5. Process Chart for Place Concrete with Crane-and-Bucket Method

Crane-and-Bucket is chosen and linked to the components that define the scope of Activity 400, which are again (*Footing-1,8*) and (*Footing-2,4*).

Step 4: Footing-1 and Footing-2 therefore initialize ClnQue 401. In the same way as before, the total amount of concrete needed to build all of these footings is read from the Project Taxonomy, and initializes ClnQue 405.

Step 5: The chosen process chart involves three construction resources: (1) A crane (RInQue 425); (2) its attachment (RInQue 415), and (3) a crew (RInQue 440), which the user has linked to the appropriate resources in the construction-resource hierarchy, so that the proper attribute values are taken into account in the model.

Explanation of Place Concrete with Crane-and-Bucket Process

This Place Concrete with Crane-and-Bucket process has as input components a set of footings and the corresponding amount of concrete. It has a crane to which a bucket will be attached for loading concrete, and a crew to help in placing concrete. The Assembly link between RlnQue 425 and Combi 420 denotes the attachment of the bucket to the crane. The Assembly link from RlnQue 425 to Combi 410 and the Disassembly link from IQue 430 to Combi 435 denote that concrete will be unloaded from the crane and bucket and placed into the footing. How much will be unloaded depends on the quantity contained in the bucket and the size of the footing. Assuming the bucket is full, the maximum available quantity is that of the bucket attribute capacity. The maximum needed quantity is that of the footing attribute *Concrete-4*. If the two do not match exactly, CIPROS may hold a footing in the Place Concrete (Combi 435) for several Combi activations, or it may use concrete in the bucket to fill several footings.

After completing their respective tasks, the crew rejoins its RInQue 440 and the crane with bucket rejoins its RInQue 425. The footing with concrete placed in it joins the OutQue 445. When the same number of Footings specified in CInQue 401 are counted in OutQue 445, the process has been completed.

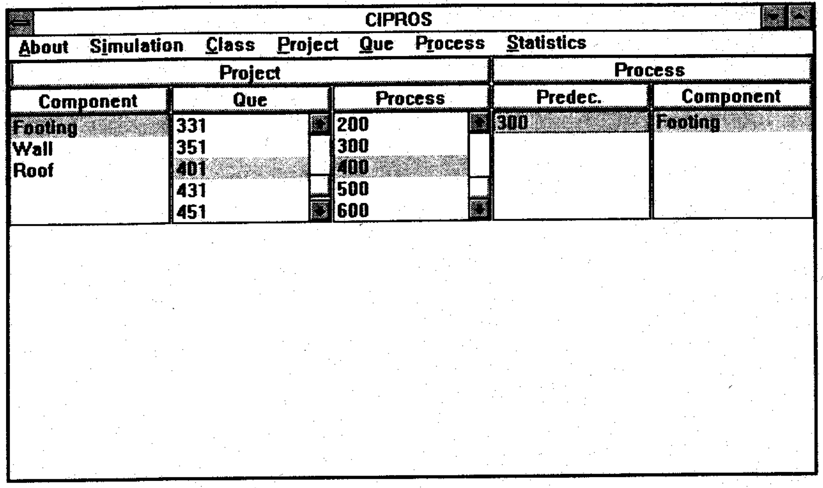
Concluding Steps

Step 6: The user must select or custom-tailor one method for each activity in the plan to produce a simulation network that fits the plan.

Step 7: Finally, CIPROS simulates project execution in its entirety by simulating the plan's activities in the sequence of and according to the priorities established in the preceding steps. Statistics provide output at the process and at the project planning level. This output includes data collected during simulation and derivatives thereof. Examples of process data are the start and finish times of Normal or Combi activations, the resources used in each activation, and resource idle time throughout the process's activation. Examples of derived data are "activity start" and "activity finish" at the planning level.

IMPLEMENTATION SOFTWARE AND HARDWARE

CIPROS is implemented in a PC Windows environment using the ACTOR (Franz 1989; Whitewater Group 1990) object-oriented programming language. The input environment is user-friendly with menu-driven windows, input dialogs, and scanning facilities that allow viewing and updating input parameters with ease. Fig. 6 shows the menu where the user has



	Project		Process	
Component	Que	Process	Predec.	Component
Footing	331	200	300	Footing
Wall	351	300		
Roof	401	400		
	431	500		
	451	600		

FIG. 6. Example CIPROS User Interface

selected "Process 400," Place Concrete with Crane-and-Bucket, for the Place Concrete activity. For Process 400, the *Component Footing* is to be initialized in *Queue*(ClnQue) *401*. The menu also shows that process 400 has process *300* as its *predecessor* and that *Component Footing* must complete process 300 for process 400 to begin.

The CIPROS architecture is generic; its modeling concepts apply to a variety of construction facilities, techniques, and methods. In its current implementation, however, the scope of CIPROS's knowledge bases was limited to construction of structural systems for high-rise buildings.

CIPROS CAPABILITIES AND LIMITATIONS

CIPROS takes advantage of the modularity, expressiveness, and expandability of its object-oriented implementation environment. Its capabilities in terms of resource characterization, resource tracking, and control allow for realistic modeling of complex systems. CIPROS's strength lies in its ability to consider specific information about a project, enabling its users to map such information to construction processes with common productive resources, and to simulate these processes in accordance with the user-defined project plan.

The CIPROS modeling methodology has been tested on more substantial examples than the one detailed here. A 22-activity network for the construction of a two-story steel-frame building is fully elaborated in Odeh (1992). While the presented methodology has been applied to construction management, it would equally apply to other domains in which methods play a central role.

The CIPROS object hierarchy and knowledge base are expandable. Depending on the requirements of the project at hand, the user may use the model's existing classes or create new specialized classes. A newly defined class can be a subclass of an existing class, and can inherit variables and methods from it. Thus, users need not develop new classes from scratch. Furthermore, they may define and add new processes representing construction techniques and methods to the knowledge base. Once a new class or process is defined, it becomes part of the model, and it can be accessed and used repeatedly.

CIPROS's processes are modular. Except for Input Queues of common productive resources, each process has its own network. Consequently, users can model different entity interactions and resource utilization strategies by altering input parameters without affecting the process chart. Users can also model different scope and precedence strategies of activities without affecting the corresponding process definitions. Process modularity coupled with the knowledge base of construction techniques and methods enables the user to model alternative construction plans and experiment with competing construction methods with relative ease.

CIPROS users must specify the appropriate rules-of-thumb to determine quantities of materials and allowable waste. They must identify individual components, select applicable methods to perform activities in the plan, and choose probability distributions characterizing operation durations. These are by no means trivial tasks, and little support is provided by CIPROS in this regard.

CIPROS's meticulous initialization and tracking of resource and design element data comes at a computational cost. Many operation functions can be evaluated only at run-time, and most inherited object-attribute values are not being cached in the current implementation of CIPROS, which slows the simulation down.

Some conceptual limitations of CIPROS are those inherent to simulation models, stemming from uncertain resource needs and uncertain or uncontrollable future events scheduling. To initialize the quantity of needed materials based on the design specifications, rules-of-thumb must be applied (e.g., to estimate how much concrete must be ordered to cast a beam of known dimensions). Also, the present CIPROS model cannot make decisions based on events external to the operation. An entity with an incomplete task, along with its captured resources, could be held idle in an operation for a long period of time due to improper resource strategies of other operations in the process. In such a case, the operation has neither knowledge nor control over when the entities needed for task completion will be available. This is in contrast with real-life systems, in which managers can coordinate and control entity interactions and flows of different processes.

A resource utilization strategy that defines completion of an assembly task, especially those involving nonbulk entities, might lead to unanticipated and undesired situations. The general issues are: (1) What must be done to complete an almost-complete task when no more resources are available? (2) Should a task be terminated when its available indivisible, nonbulk entities are less in quantity than what is required for task completion? (3) For how long should an idle entity with an incomplete assembly task wait before the task should be terminated? and (4) When can resources captured in an incomplete task be freed? The solutions to such questions are based on knowledge regarding the future flow of materials to the operation, and such knowledge is generally not available in the simulation model. One way of dealing with such questions is to allow freeing of captured resources or terminating assembly tasks if a user-defined time has elapsed since the task entity became idle. Alternatively, the assembly task may be terminated if a user-defined percentage of task completion has been achieved at the end of operation activation.

CONCLUSIONS

CIPROS can realistically model construction processes by integrating data from design drawings and specifications, and construction plans, in combination with a library of construction methods. The availability of these knowledge bases provides leverage to the CIPROS simulation engine that can exploit diversity in resources. The model also alleviates knowledge-intensive simulation-network construction and initialization tasks. This results in a model that extends beyond current planning and simulation models in its ability to integrate activity- with process-level planning. This new methodology provides a successful means for integrating designer, planner, and contractor knowledge into a unified system. It demonstrates how data generated by each participant in the facility engineering process contributes to the overall system definition.

The CIPROS object-oriented, interactive, knowledge-based system implements the presented methodology and delivers proof of concept of its usefulness and practicality. Its sophisticated discrete-event simulation engine takes advantages of data stored in the various knowledge bases. The tight integration of design and specification data, project planning, and process simulation achieved in CIPROS, illustrates that discrete-event simulation can—and should—be an integral part of construction management tools.

ACKNOWLEDGMENTS

The CIPROS system was designed and implemented by A. M. Odeh, as part of his doctoral research conducted under the supervision of Professors Carr and Tommelein. Dr. Odeh's (1992) dissertation provides more detail on this system.

This work was funded in part by a scholarship awarded to Dr. Odeh by Yarmouk University and in part with funds from the Bottum Endowment of the Center for Construction Engineering and Management at The University of Michigan in Ann Arbor. This support has been greatly appreciated.

APPENDIX. REFERENCES

AbouRizk, S. M., and Halpin, D. W. (1992). "Modeling input data for construction simulation." *Proc., 8th Conf. Computing in Civ. Engrg. and Symp. on Geographical Information Systems,* ASCE, New York, N.Y., 1147–1154.

Bernold, L. E. (1989). "Simulation of nonsteady construction processes." *J. Constr. Engrg. and Mgmt.*, ASCE, 115(2), 163–178.

Carr, R. I. (1979). "Simulation of construction project duration." *J. Constr. Div.*, ASCE, 105(2), 117–128.

Chang, D. Y. (1986). "RESQUE: A resource based simulation for construction process planning," PhD dissertation, Dept. of Civ. Engrg., Univ. of Michigan, Ann Arbor, Mich.

Cherneff, J., Logcher, R., and Sriram, D. (1991). "Integrating CAD with construction-schedule generation." *J. Computing in Civ. Engrg.*, ASCE 5(1), 64–84.

Dabbas, M. A. A., and Halpin, D. W. (1982). "Integrated project and process management." *J Constr. Div.*, ASCE, 108, 361–374.

Franz, M. (1989). *Object-oriented programming featuring actor.* Scott Foresman and Co., Glenview, Ill.

Halpin, D. W. (1989). *MicroCYCLONE user's manual.* Div. of Constr. Engrg. and Mgmt., Purdue Univ., West Lafayette, Ind.

Halpin, D. W., and Riggs, L. S. (1992). *Planning and analysis of construction operations.* Wiley-Interscience, New York, N.Y.

Huang. R.-Y., and Halpin, D. W. (1993). "Dynamic interface simulation of construction operations." *Proc., 10th ISARC, Automation and Robotics in Construction*, Watson, G. H., Tucker, R. L., Walters, J. K., eds., Elsevier Science Publishers, 503–510.

Ioannou, P. G. (1988a). "UM-Cyclone Reference Manual." *Tech. Rep. UMCE-89–11*, Civ. Engrg. Dept., Univ. of Michigan, Ann Arbor, Mich.

Ioannou, P. G. (1988b). "UM-Cyclone User's Guide." *Tech. Rep. UMCE-89–12*, Civ. Engrg. Dept., Univ. of Michigan, Ann Arbor, Mich.

Kalk, A. (1980). "INSIGHT: Interactive simulation of construction operations using graphical techniques." *Tech. Rep. 238*, Civ. Engrg. Dept., Stanford Univ., Calif.

Law, A. M., and Kelton, W. D. (1991). *Simulation modeling and analysis*, 2nd Ed., McGraw-Hill Book Co., Inc., New York, N.Y.

Lehmann, A. (1988). "Taxonomy and application of expert systems in simulation." *Artificial intelligence, expert systems and languages in modeling and simulation.* Elsevier Science Publishers B.V. IMACS, The Netherlands.

Liu, L.-Y. (1991). "COOPS: construction object-oriented process simulation system," PhD dissertation, Civ. and Envir. Engrg. Dept., Univ. of Michigan, Ann Arbor, Mich.

Lluch, J., and Halpin, D. W. (1982). "Construction operations and microcomputers." *J. Constr. Engrg. and Mgmt.*, ASCE, 108(1), 129–145.

Morua Padilla, E., and Carr, R. I. (1991). "Resource strategies for dynamic project management." *J. Constr. Div.*, ASCE, 117(2), 279–293.

Odeh, A. M. (1992). "CIPROS: Knowledge-based construction integrated project and process planning simulation system," PhD dissertation, Civ. and Envir. Engrg. Dept., Univ. of Michigan, Ann Arbor, Mich.

Odeh, A. M., Tommelein, I. D., and Carr, R. I. (1991). "Using design drawings and project plans to construct discrete-event simulation networks." *Proc., ISARC 91*, Springer-Verlag, Stuttgart, Germany, 419–428.

Odeh, A. M., Tommelein, I. D., and Carr, R. I. (1992). "Knowledge-based simulation of construction plans." *Proc., 8th Conf. Computing in Civ. Engrg.*, ASCE, New York, N.Y., 1042–1049.

O'Keefe, R. (1986). "Simulation and expert systems—a taxonomy and some examples." *Simulation*, 46(1), 10–16.

Paulson, B. C. Jr. (1978). "Interactive graphics for simulation construction operations." *J. Constr. Div.*, ASCE, 104(3), 69–76.

Paulson, B. C. Jr., Chan, W.-T., Koo, C. C. (1987). "Construction operations simulation by microcomputer." *J. Constr. Engrg. and Mgmt.*, ASCE, 113(2), 302–314.

Reddy, R. (1987). "Epistemology of knowledge based simulation." *Simulation*, 49(4), 162–166.

Riggs, L. S. (1987). "Graphic input to Cyclone." *J. Computing in Civ. Engrg.*, ASCE, 1(3), 175–182.

Shannon, R. E., Mayer, R., and Adelsberger, H. H. (1985). "Expert systems and simulation." *Simulation*, 44(6), 275–284.

Touran, A. (1990). "Integration of simulation with expert systems." *J. Constr. Engrg. and Mgmt.*, ASCE, 116(3), 480–493.

Whitewater Group. (1990). *ACTOR user's manual Vols. 1 and 2.* Evanston, Ill.

Zozaya-Gorostiza, C., Hendrickson, C., and Rehak, D. R. (1989). *Knowledge-based planning for construction and manufacturing.* Academic Press, Inc., London.