COOPS—CONSTRUCTION OBJECT-ORIENTED PROCESS SIMULATION SYSTEM

Photios G. Ioannou
Associate Professor
Civil and Environmental Engineering
University of Michigan
Ann Arbor, MI 48109

Liang Y. Liu
Associate
Project Management Associates, Inc.
226 W. Liberty Street
Ann Arbor, MI 48104

ABSTRACT

This paper describes the modeling concepts and design of COOPS, a new general purpose discrete-event simulation system for construction process modeling. COOPS simulation models are represented by graphical networks that are constructed interactively on the screen using primitive objects, such as activities, queues, consolidation nodes, routers, links, flags, resources, and free text. All primitive modeling elements are implemented as integrated objects that have dual functionality for simulation and for interactive graphics. Simulation is performed directly on the graphical model constructed on the screen. In addition to standard modeling capabilities, the system provides precise simulation control, resource tracking, and resource balancing for construction process modeling, by introducing the concepts of calendars, specific resources, and link resource requirements. COOPS is a completely object oriented construction simulation system. Thus, its functionality can be readily enhanced by extending the capabilities of existing objects. Furthermore, other planning systems, such as knowledge-based expert systems, can link and communicate directly with the objects in a COOPS simulation model. COOPS runs under Microsoft Windows 3.x and is available from the authors upon request.

INTRODUCTION

Simulation of construction operations is a very effective tool for construction process design and optimization. It provides detailed process information, such as resource utilization, resource idleness, operation bottlenecks, production rates, and costs before committing real resources to construction. Even though process simulation has been applied to construction for almost 30 years, the advent of new computer science technologies allows the design and implementation of new general purpose systems, such as COOPS, that provide a far better and more powerful modeling environment than their predecessors.

COOPS is a is new general purpose discrete-event simulation system that provides an integrated environment for construction process modeling. Simulation models are represented as graphical networks of nodes and links, similar in concept to those introduced by CYCLONE[1]. The design and implementation of the two systems, however, are radically different. The simulation network can be developed interactive on screen by using graphical editing tools similar to those in vector-based drawing programs. All modeling elements, including resources, activities, queues, links, consolidations, routers, and flags, are implemented as objects that have extensive behavioral functionality. Furthermore, the concepts of specific resource objects, calendars, and link-resource-requirement units provide additional modeling flexibility and power.

The system design and implementation is based on object-oriented programming and interactive computer graphics. The statistical data during simulation are collected and kept inside the objects as the simulation proceeds. Additional functions can be added easily by enhancing the behavior of the existing objects. Other planning systems, such as expert systems, can query the data stored inside the objects of a simulation model during runtime to perform reasoning or other scenario searching and simulation. In fact, it is quite easy to extend COOPS into a concurrent simulation/expert system.

The current version of COOPS runs under Microsoft Windows 3.0 or higher and requires a 386 or 486 IBM compatible computer with at least 2MB of RAM, a VGA or better graphics adapter, and a mouse. The system and its manuals are available from the authors upon request.

COOPS MODELING CONCEPTS

COOPS (Construction Object-Oriented Process Simulation System) was originally developed by the authors to demonstrate the benefits of object-oriented simulation design and, at the same time, overcome some of the limitations of existing simulation systems in construction. COOPS adopts the scheme of a network-based simulation model that traces its roots to Q-GERT[5] and CYCLONE. COOPS utilizes a similar set of modeling elements, while providing additional capabilities for modeling construction processes.

The COOPS simulation model is a precedence network that consists of three kinds of objects: nodes, links, and attachments (Figure 1). Nodes and links are similar to those used in CPM networks. COOPS defines four types of nodes: activity, queue, consolidation, and router, each with its distinctive graphical representation and function in the network. Links are graphically represented as arrows showing the precedence relationship between two nodes and symbolizing the direction of resource flows. There are three kinds of attachment objects: specific resources, resource units, and flags. These attachments, when attached to a queue or a link, will affect the characteristics of the attached modeling element. In short, modeling in COOPS involves assembling the basic modeling elements to construct a network that represents a construction operation, and then to perform simulation.

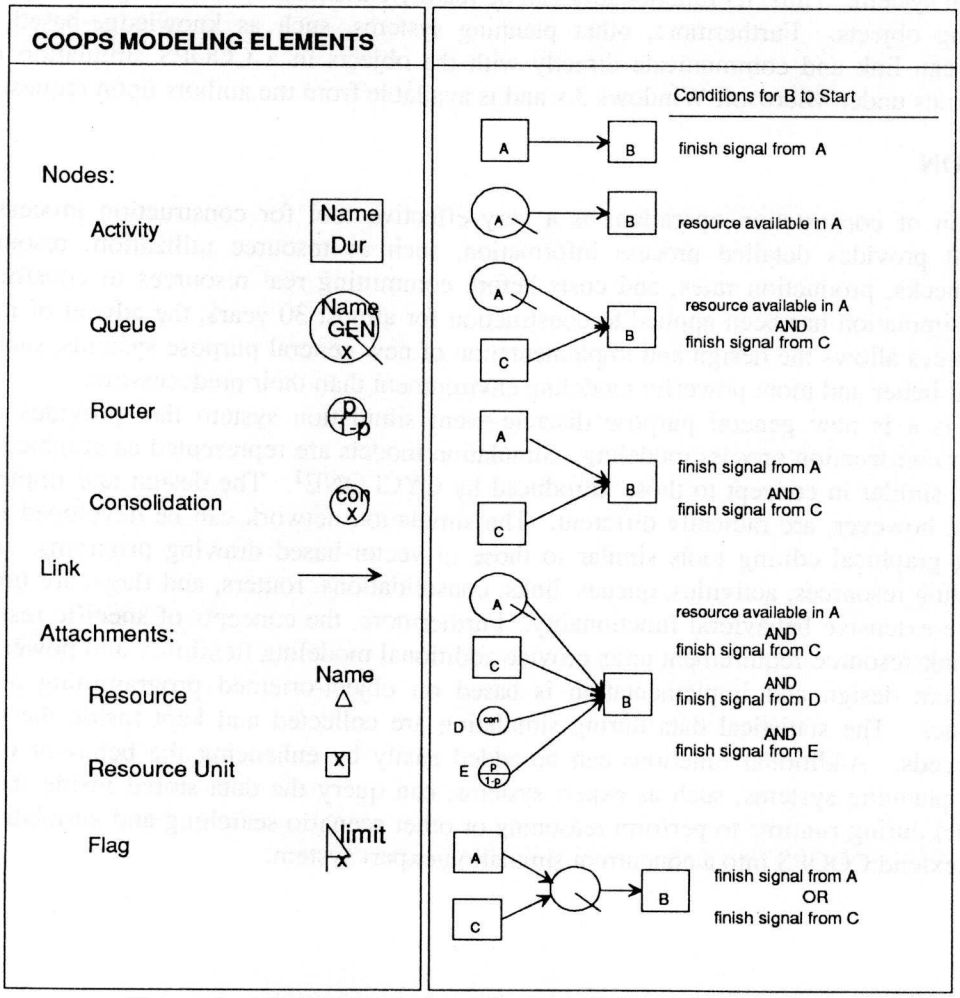

Figure 1. COOPS Modeling Elements & Precedence Rules

Resources

A resource object defines the attributes of a real world resource. In construction, typical resources are labor, equipment, materials, time, and space. COOPS defines the following resource attributes that relate directly to the production of a construction operation: units, state, flows, interactions, transformations, and queueing behavior.

COOPS can use two kinds of resources: "generic" and "specific." Generic resources are assumed to be identical and interchangeable, whereas specific resources are individually identified. For example, ten workers in a construction operation can be modeled as either a pool of ten units of generic resources or ten individual specific resources. Generic resources are modeled using the same scheme as in CYCLONE. Specific resources are graphically represented by a triangle with their names shown over the top of the triangle (Figure 1). They are used to model resources that warrant special attention because of their importance, or high costs, such as special crews or expensive machinery. The time-related statistics for these specific resources, such as the utilization rate, are collected individually. Only group-related statistics can be collected for generic resources.

Queues

A queue can be viewed as a storage area for resources. Resources in queues are in an idle state. The contents of a queue change whenever resources are captured from or released to the queue during simulation. The rules for changing the resource contents of a queue can be summarized as follows:

- When a directly-preceding node finishes, the contents of the queue are increased by the units of resource entering the queue.
- When an activity starts, the contents of each directly preceding queue are decreased by the units of resources captured by the activity.

Many modeling situations require matching of units for the various resources. For this reason, queues are associated with GEN functions that "split" each incoming resource unit into GEN number of units. This provides queues with the capability of resource *generation.* For obvious reasons, GEN functions can be used only for queues that store generic resources.

The contents of queues for generic resources are not associated with a particular unit of measure. The system does not know how these numbers are interpreted by the user, whether for example they represent cubic yards of soil or the number of available machines. It is the user's responsibility to select the appropriate units of measure and to make sure that the contents of queues are increased or decreased by the correct amounts. Furthermore, the units of measure for one queue should be compatible with those of other queues and with those required by the succeeding activities.

Activities

Activities, similar to the activities in CPM or PERT networks, represent tasks that need to be performed in an operation. Each activity has a duration which can have one of six different types of probability distributions: deterministic, uniform, normal, beta, gamma, and triangular. Activities are also assigned priority attributes to determine which will use a resource first when two or more activities are competing for the same resource. The higher the attribute number, the higher the activity's priority.

Routers

Routers are nodes used to represent a binary random selection in a construction process. They are graphically represented as a circle with a line in the middle that separates two real numbers representing the upper and the lower link probabilities (Figure 1). Normally, a node will activate all links leading to its successor nodes. Routers are used in situations where it is desirable to activate only one of these links (and

the associated succeeding node) based on a probabilistic selection process. Although router objects provide only binary random selections, they can be combined to model multiple random selections.

Consolidations

A consolidation node represents the transformation, combination, or consumption of construction resources. It is the logical opposite of a queue with a GEN function, in that it merges a predefined number of incoming generic resources to produce one unit of a finished resource. Consolidation nodes affect only generic resources. Specific resources maintain their characteristics and simply flow through a consolidation node unchanged.

Links

Links specify the direction of resource flows and the precedence relationships between nodes. Specific resource routing is modeled by attaching specific resource objects to links. In this case, the link allows only the specific resources to flow through.

Link Resource Requirement (LRR) Units

Construction operations typically require different combinations of resources, whose units have to be balanced according to the operational logic. Link resource requirement units (LRR) are objects that are attached to a link to specify the number or units of generic resources that will be added to a succeeding queue or removed from a preceding queue. LRRs complement the use of GEN functions (in queues) and consolidation nodes when balancing resource units in a network.

Flags

Flags are special attachment objects that are attached to queues in the network to set production limits. When the contents of a queue reach the limit set by its flag, the simulation stops. More than one flag may be used in a model.

Calendars

Calendars control the availability of resources during simulation by defining the work/break schedule of each resource. For example, a calendar named "9-to-5", can have the work/break schedule: 4,1,4,15 (unit: hours). This calendar represents a typical working day of two four-hour shifts. The cycle time is the sum of all work/break times (4+1+4+15=24). By assigning the appropriate combination of work/break time pairs any calendar can be defined.

For efficiency, COOPS uses a system calendar to define the default working schedule. The system calendar defines the working schedule for all resources in a model except for those that have their own calendar, which, when defined, will override the system calendar. For example, some resources may work overtime while the system (default) calendar works 9 to 5.

Obviously, since actual calendars control the availability of resources, they also affect the calculation of the finish time of an activity. When an activity starts, it captures the required resources from its preceding nodes. During the duration of an activity, if any of the captured resources would need to take a break based on their calendar, the duration of the activity must be increased and its finish time adjusted.

PRECEDENCE RULE AND NETWORK LOGIC

The logic of a network dictates the processing of a simulation. An activity cannot start until all the required resources in its preceding nodes are available. When an activity precedes another activity, the latter can start only after a signal is issued by the former. The conditions under which an activity can start are summarized in Figure 1.

EXAMPLE MODEL 1—LOADING OPERATION

COOPS uses interactive graphics and windows for model building. The basic modeling elements—nodes (activities, queues, consolidation, routers), links, and attachments (specific resources, flags, and resource requirement units)—can be selected from the toolbox in the upper left corner of the screen (Figure 2) and placed in the model area to form a simulation network. The network in Figure 2 shows a model of an earthmoving operation with two types of trucks (Truck1: 20 CY, Truck2: 30 CY) and a 5-CY loader. In this example, trucks are modeled as generic resources and are initially assigned to the queues "Truck1" and "Truck2"; the loader is modeled as a specific resource to collect detailed statistics. The specific resource symbol for the loader is placed in its starting queue. This represents the initial condition of the model. The loader resource symbol is also attached to the two returning links to specify its direction of flow when it is released by the loading activities. Because the capacities of the trucks are different, different units of soil will be removed from the queue "Soil" or placed in the queue "Quantity." This is modeled by attaching four LRRs to links. Priority values are assigned to both loading activities ("Load T1": 5, "Load T2": 10). The priority values decide which truck type has priority for loading with soil, when both truck types are available. The names and corresponding distributions of activity durations are displayed in the activity symbols. Calendars can be assigned to the queues "Truck1" and "Truck2" to define the working schedules of trucks. The loader can carry its own calendar which can be different from the trucks. A production limit of 500 is set on the queue "Quantity," so that when the contents of the queue reaches 500, the simulation stops.

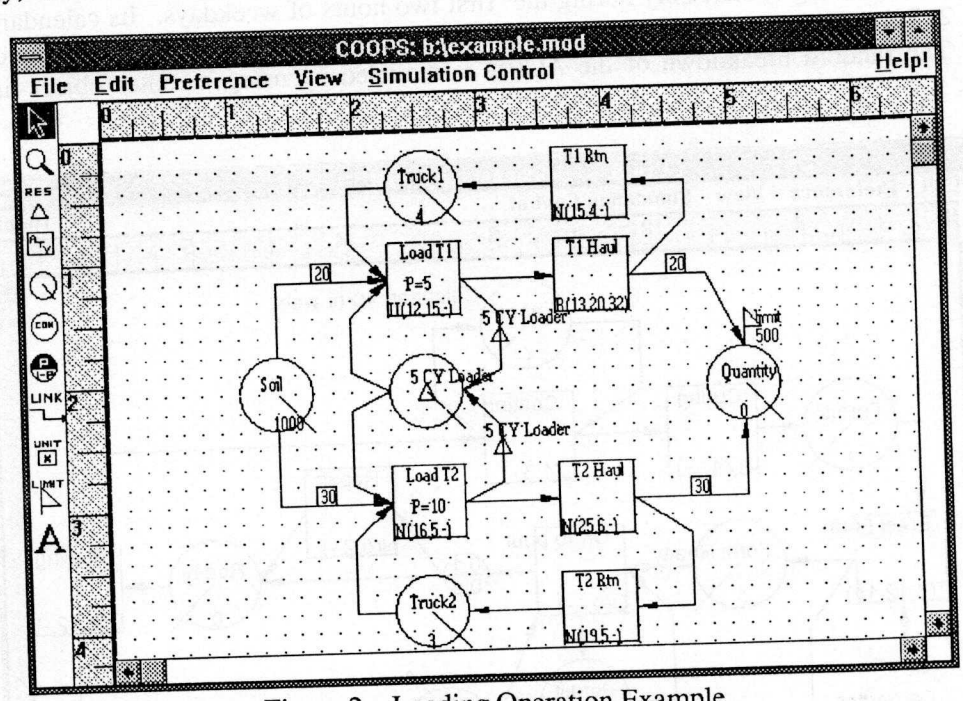

Figure 2—Loading Operation Example

Once the data for the elements are defined, a user can set the simulation time limit and random number seed to run the simulation. The result of a simulation consists of four parts: the network model, event list, simulation trace, and the time-related statistics for specific resources, queues, and activities, similar to that of UM-CYCLONE^2 standard output.

EXAMPLE MODEL 2—CONCRETE PLACEMENT OPERATION

Figure 3 shows a COOPS model of a simple continuous concrete placement operation using an ACPM (automatic concrete placing machine). It illustrates different working schedules, random machine breakdowns, and the use of decision rules. Concrete is placed into two sets of reusable forms stripped and erected by carpenters. Placement is only allowed during the first two working hours of weekdays. Once placement starts, the ACPM and its crew will continue working until finished. A curing compound is then applied by laborers. Carpenters and laborers follow a regular 40-hr/week schedule. There is a 10% chance that the ACPM will break down and require on-site repair. Activity durations are shown on the network by a letter that stands for the distribution type, followed by its parameters. All times are given in hours. For example:

D(24):	Deterministic (24)
T(8,12,15):	Triangular (lower bound=8, mode=12, upper bound=15)
N(5,2):	Normal (mean=5, standard deviation=2
U(8,12):	Uniform (lower bound=8, upper bound=12)

Queues are assigned three types of calendars representing different working schedules. "Carpenter" and "Labor" queues are assigned the calendar "40-hr-week," (i.e., 8, 16, 8, 16, 8, 16, 8, 16, 8, 64). The "Permit" queue allows pouring to start only during the first two hours of weekdays. Its calendar is (2, 22, 2, 22, 2, 22, 2, 70). All other resources are assigned the "System" calendar, which works continuously without breaks. The random breakdown of the ACPM is modeled by using a router object linked with a "Repair" activity.

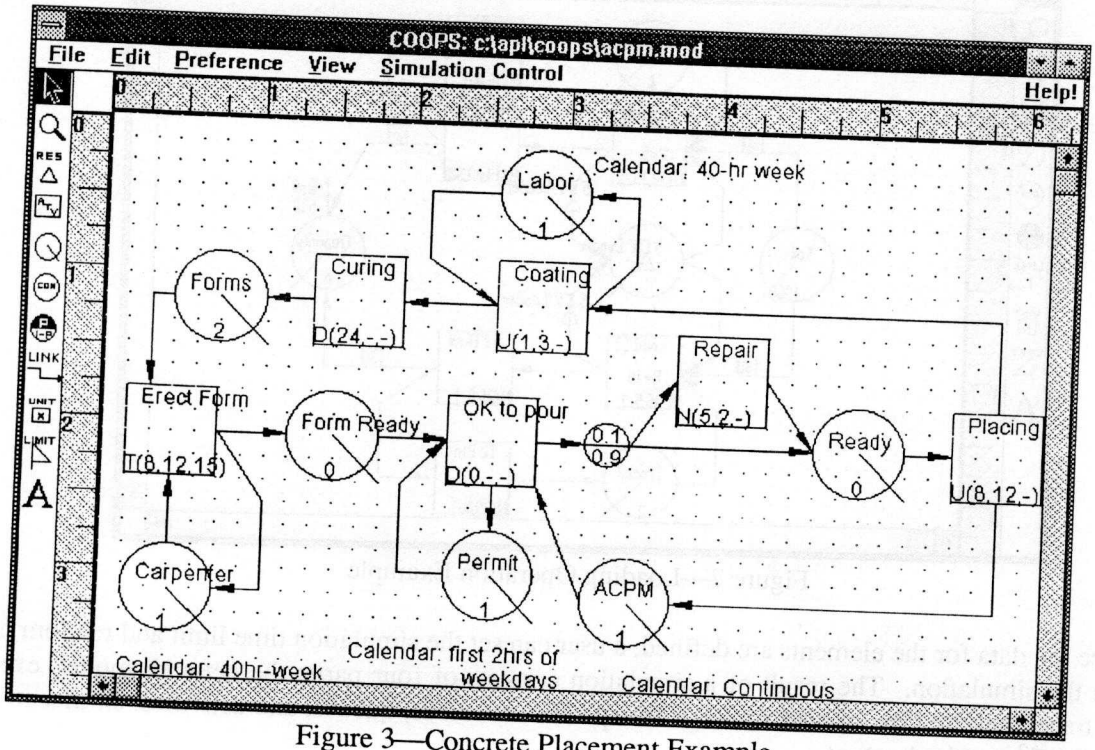

Figure 3—Concrete Placement Example

A COOPS simulation produces four kinds of reports: (1) Network model, (2) Statistical output for each modeling element, (3) Simulation event list, and (4) Simulation trace record (this is available when the

simulation is in "Trace Mode" and shows each step of execution). Tables 1,2 and 3, show parts of these reports.

Table 1—Statistical Report

```
**** COOPS SIMULATION SYSTEM ***   MODEL:   C:\APL\COOPS\ACPM.MOD
SIMULATION TIME LIMIT:  216.00
RANDOM NUMBER SEED:     266301881
TOTAL SIMULATED TIME =  216.00

                GENERIC RESOURCE UNITS
QUEUE           @START     @END    OUTPUT   AVERAGE     STD_DEV
=============================================================
ACPM              1         1        3       0.842       0.365
Carpenter         1         1        4       0.274       0.446
Form Ready        0         1        3       0.355       0.478
Forms             2         0        4       0.467       0.499
Labor             1         1        3       0.975       0.157
Permit            1         1        3       1.000       0.000
Ready             0         0        3       0.000       0.000

ACTIVITY    OUTPUT GROSS_RATE NET_RATE   AVG_DUR     STD_DUR
=============================================================
Coating        3      0.014     0.020      1.813       0.675
Curing         2      0.009     0.043     24.000       0.000
Erect Form     4      0.019     0.020     39.231      24.661
OK to pour     3      0.014     0.021      0.000       0.000
Placing        3      0.014     0.019     11.393       0.230
Repair         0      0.000     0.000      0.000       0.000
```

Table 2—Simulation Event List

```
...
*********************************************************
* Activity Name:          Erect Form
* Description:            Carpenters erect forms
* Activity Duration:      Triangular(8,12,15)
*********************************************************
     Start_Time              End_Time
    ================================
          0.00                26.23
         26.23                53.59
         96.00               172.22
        172.22               199.33

....
*********************************************************
* Activity Name:          Placing
* Description:            -
* Activity Duration:      Uniform(8,12,-)
*********************************************************
      Start_Time              End_Time
     ================================
         48.00                59.41
         72.00                83.16
        192.00               203.62
*********************************************************
*    COOPS Event List (SimulationControl)
*********************************************************
       ACTIVITY       START_TIME       END_TIME
      ========================================
    Erect Form            0.00           26.23
    OK to pour           48.00           48.00
    Erect Form           26.23           53.59
       Placing           48.00           59.41
```

Table 3—Trace Mode Record

```
**** COOPS SIMULATION TRACE ****   MODEL:   C:\APL\COOPS\ACPM.MOD

Activity starts: ..... Erect Form
TNOW: ..... 0.
Sampled duration: ..... 10.22529665
Scheduled end time: ..... 26.22529665
<<
Advance the simulation clock
TNOW: ..... 26.22529665
<<
Activity starts: ..... Erect Form
TNOW: ..... 26.22529665
Sampled duration: ..... 11.36792676
Scheduled end time: ..... 53.59322342
<<
Queue: Permit
is not available @ ..... 26.22529665
Scheduled available time: ..... 48.
<<
Advance the simulation clock
TNOW: ..... 48.
<<
Activity starts: ..... OK to pour
TNOW: ..... 48.
Sampled duration: ..... 0.
Scheduled end time: ..... 48.
...
```

COOPS DESIGN

COOPS is a completely object-oriented system. All program code has been implemented by defining classes and adding methods and instance variables according to object behavior analysis. After the classes were defined, instances of these classes (i.e., objects) were used to create COOPS. The simulation functions are achieved by objects sending messages to each other and receiving messages from the user. The objects that coexist in COOPS to perform simulation and some typical messages are shown in Figure 4.

The system's user-interface is a combination of three window objects: SimuWind, SimPalette, and SimDraw. They are meshed together as if they were a single window. SimuWind is the main window which provides menu selections such as loading model files, editing modeling elements, and setting simulation control parameters. SimPalette is the toolbox at the upper left corner of the COOPS modeling environment. SimDraw provides the drawing area for creating and editing COOPS network models. The modeling elements created by the user, such as activities, queues, resources, consolidation, routers, and links, are stored in the Network object, one of SimDraw's instance variables.

When a user sends a startSimulation message, the SimuWind object receives and relays the message to the SimulationControl object which acts as simulation control center. The SimulationControl object's instance variables store vital simulation information, such as TNOW, FEL, and ActvScanList. TNOW is the simulation clock time, FEL (Future Event List) is a list of future events in the simulation, and ActvScanList is a list of all activity objects in a COOPS model. During simulation, the SimulationControl object sends messages to the Network object to retrieve the data stored in the modeling elements. If the activity duration is a random variable, messages are sent to the Distribution object to sample the duration from a particular distribution, such as uniform, normal, triangular, etc.

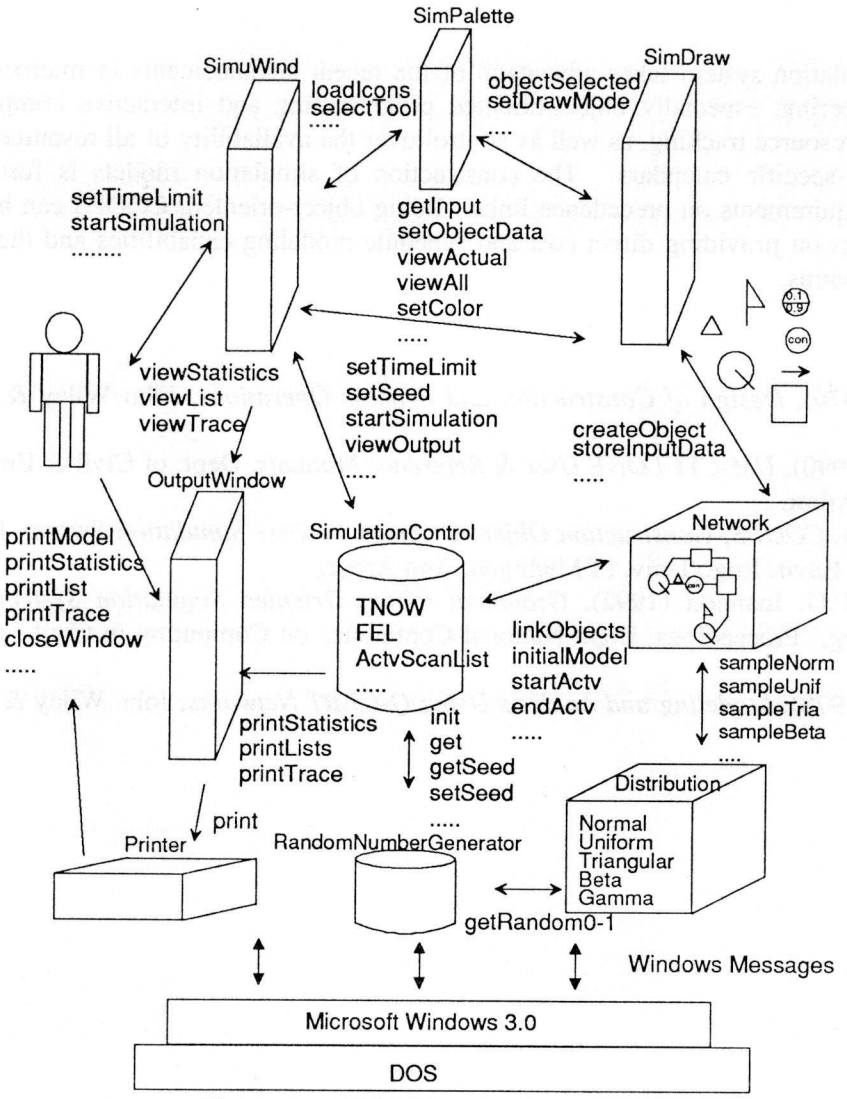

Figure 4—COOPS Conceptual Design

The SimulationControl object performs two tasks alternatively: (1) scanning and scheduling activity end-event times and (2) advancing the simulation clock time to the earliest future event. Example messages sent by the SimulationControl object include startSimulation, getNextEvent, scheduleEvent, stopActivity, startSuccessors, sampleDuration, etc. The scanning/scheduling and advancing the clock tasks proceed alternately until the simulation time limit or the production limit has been met. During simulation, time-related information is posted to the objects' timeStatistics instance variable, which can later respond to a printReport message to print out the statistic of a queue, a resource, or an activity.

LINKAGE WITH EXPERT SYSTEMS

Because of its object-oriented simulation system design, the knowledge representation in COOPS matches closely that of frame-based expert systems. All components of COOPS are implemented as objects. As a result, these objects can be directly queried or controlled by other planning systems through message passing. This linkage can be achieved at the system level during runtime instead of the input/output level.

CONCLUSIONS

COOPS simulation system takes advantage of the recent advancements in microcomputer hardware and software engineering, especially object-oriented programming and interactive computer graphics. It provides individual resource tracking, as well as control over the availability of all resources via an unlimited number of resource-specific calendars. The construction of simulation models is further simplified by defining resource requirements on precedence links. Being object-oriented, COOPS can be easily extended. Current work focuses on providing direct cost and schedule modeling capabilities and the extension to link with other expert systems.

REFERENCES

1. Halpin, D.W.(1976), *Design of Construction and Process Operations*, John Wiley & Sons, New York, N.Y.
2. Ioannou, P.G. (1990), *UM-CYCLONE User & Reference Manuals*, Dept. of Civil & Envir. Eng., Univ. of Michigan, Ann Arbor.
3. Liu, L.Y. (1991), *COOPS, Construction Object-Oriented Process Simulation System*, Ph.D. Dissertation, Dept. of Civil & Envir. Eng., Univ. of Michigan, Ann Arbor.
4. Liu, L.Y. and P.G. Ioannou (1992), *Graphical Object-Oriented Simulation System for Construction Process Modeling*, Proceedings, Eight National Conference on Computing in Civil Engineering, Dallas, TX, June 7-9.
5. Pritsker, A.A. (1978), *Modeling and Analysis Using Q-GERT Networks*, John Wiley & Sons, New Yoirk, N.Y.