Proceedings of the First Congress on Computing in Civil Engineering, ASCE
Washington, D.C., June 20-22, 1994
K. Khozeimeh, ed.

State And Resource Based Construction Process Simulation

Julio Martinez[1]; Photios G. Ioannou[2], A.M. ASCE; and Robert I. Carr[3], Fellow ASCE

Abstract

Stroboscope is a programming language for construction process simulation. The programming language provides access to the state of the simulation and to the properties of resources. It allows the creation of realistic models that can make utilization, consumption, and production of resources stochastic; perform dynamic resource allocations; characterize resources created at runtime by combining other resources; and make dynamic decisions regarding the sequence of operations.

Overview

Stroboscope is an acronym for STate and ResOurce Based Simulation of COnstruction ProcEsses. It is a programming language specifically designed to model construction operations. Stroboscope models are based on a network of interconnected modeling elements and on a series of programming statements that give the elements unique behavior and control the simulation.

The character of Stroboscope arises from its ability to dynamically access the state of the simulation and the properties of the resources involved in an operation. The *state* of the simulation refers to such things as the number of trucks waiting to be loaded; the current simulation time; the number of times an activity has occurred; and the last time a particular activity started. Access to the properties of resources means that operations can be sensitive to properties — such as size, weight, and cost — on an individual (the size of the specific loader used in an operation) or an aggregate basis (the sum of the weights of a set of steel shapes waiting to be erected).

Stroboscope modeling elements have attributes — defined through programming statements — that define how they behave throughout a simulation. Attributes represent such things as the duration or priority of an activity, the discipline of a queue, and the amount of resource that flows from one element to another. Most attributes can be specified with expressions and have

[1]Doctoral Candidate, Civil & Environmental Engineering Department, University of Michigan, Ann Arbor, MI 48109-2125.
[2]Associate Professor, Civil & Environmental Engineering Department, University of Michigan, Ann Arbor, MI 48109-2125.
[3]Professor, Civil & Environmental Engineering Department, University of Michigan, Ann Arbor, MI 48109-2125.

default values that provide the expected behavior. Expressions are composed of constants; system maintained variables that access the state of the simulation and the properties of resources; user-defined variables; logical, arithmetic, and conditional operators; and scientific, statistical, and mathematical functions.

The attributes of Stroboscope modeling elements allow simulation models to consider uncertainty in aspects other than time, such as in quantities of resources (e.g., the volume of rock resulting from a dynamite blast). Attributes also allow models to dynamically select the routing of resources and the sequence of operations; to allocate resources to activities based on complex selection schemes; to combine resources and dynamically assign properties to the resulting compound resource; and to activate operations subject to complex startup conditions not directly related to resource availability (e.g., do not blast rock until all crews of all trades have left the vicinity, the wiring has been inspected, and there are less than 10 minutes left in the current shift).

Example — Earth-Moving Operations

The Stroboscope simulation language is best illustrated with an example. The following earth-moving example is relatively small and makes limited use of Stroboscope's capabilities. However, it does illustrate the system's power to model complex construction processes.

An earth-moving equipment fleet is composed of pusher tractors and two types of scrapers. These load and haul earth for the construction of a 4 km road segment. Haul distance is variable because as work progresses the scrapers need to dump the hauled earth further along the constructed segment. In order to maximize production, an attempt is made to load scrapers optimally as dictated by their load-growth curves. Optimum payload and load-time for a particular scraper-pusher combination depend on haul distance, and is thus variable. Scraper speed depends on gross weight, which depends on load, and is thus also variable. In a situation such as this, simulation can be used to determine the optimal fleet composition.

Figure 1 illustrates the network used to model this operation. At an abstract level, the Combis, Normals, and Queues shown in this figure are similar in appearance and function to CYCLONE modeling elements (Halpin and Riggs 1992). The links connecting elements are named, by convention, with 2 letters that abbreviate the type of resource that flows through them followed by a number. In this model SC is the abbreviation for scrapers, PS for pushers, and ER for earth. The network shows that pushers push-load scrapers, backtrack, and then

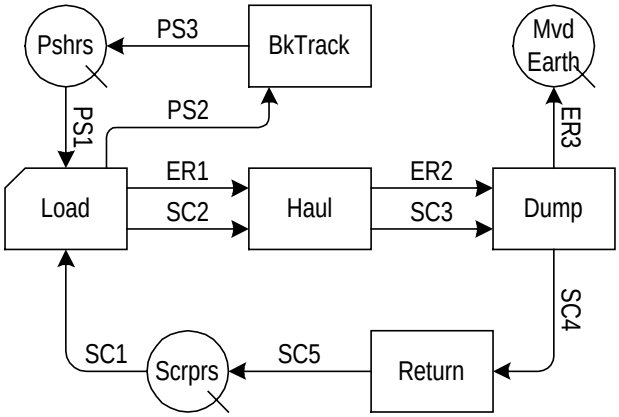

Figure 1 - Network for example

wait to push-load another scraper; scrapers are push-loaded by pushers, haul their load, dump it, return, and then wait to be push-loaded again; earth is created as result of the scraping action, it is then hauled, and finally dumped to become part of the road.

The labels on the links connecting the Stroboscope modeling elements are mandatory. They are required because Stroboscope needs to know what type of resource flows through them and because programming statements use the names to set attributes for the links.

The network provides a general picture of the earth-moving operation. Figure 2 shows more detailed data regarding the problem. Appendix II shows the Stroboscope source file for this example. The discussion that follows will reference both Figure 2 and the source listing in Appendix II. The term *row* will refer to detailed data in Figure 2. The term *line* will refer to lines in the source listing in Appendix II.

A Stroboscope source file consists of statements that end in a semicolon. Statements can span several lines and their arguments are separated by white-space. Comments are placed in source files by making the first non-white-space character after a statement a "/". The comment continues until the end of the line. The lines in Appendix II have been numbered only for the purposes of this discussion.

The scrapers that compose the fleet are of two types. Row 1 shows the characteristics of the two types of scrapers. Line 5 defines *Scraper* as a type of resource that can be characterized

Data Row	**Earth-Moving Example Data**							Source Line #
1	Scraper type	Operating Weight kN	Flywheel Power kW	Struck Capacity m^3	Maximum Speed km/hr	Hourly Cost $/hr	Trans Effic.	5 -7
	621E	299	256	10.7	51	48	80%	
	651E	583	410	24.5	55	103	83%	
2	Fleet: Three Pushers, Two 651E scrapers, Nine 621E scrapers							39-41
3	Pusher cost: $55/hr — Other costs: $200/hr							42-43
4	Earth Weight: 15.7 kN/m^3 — Shrinkage Factor: 0.95							44-45
5	Initial Haul Distance: 1000 m — Final Distance: 5000m Road Cross Section Area: 12.5 m^2 Rolling Resistance: 3% — Grade: 2%							46-50
6	Optimum Load-time (secs) = 125*(1+0.48*Ln(0.08*Distance/Power)) Optimum Payload (bcm) = Capacity*(1+Capacity/60*Ln(Distance/5000))							56-60
7	Time to spot (secs) = Beta(24 , 36, 95) Time to load (secs)= Beta(95% Opt Time, Opt Time, 110% Opt Time) Payload = Normal (Optimum Payload, 30% Optimum Payload)							61-63, 80-81
8	Boost plus transfer time (secs) = 15 Backtrack time (secs) = 40% of Optimum load-time							82
9	Actual haul time (secs) = Normal (Theoretic, 25% Theoretic)							83
10	Dump time (secs) = Beta(24, 36, 78)							84
11	Actual return time (secs) = Normal (Theoretic, 15% Theoretic)							85

Figure 2 - Detailed Data for Earth-Moving Example

by weight (Wgt), power (Pow), capacity (Cap), maximum speed (MaxV), cost (Cost), and transmission efficiency (Eff). Lines 6 and 7 define the specific values of scraper properties that *S621E* and *S651E* scrapers have.

Pushers and earth also flow through the network. Lines 11 and 12 define them as generic resources. Generic resources do not have properties attached to them and can be bulk (i.e., their amounts can have fractional values). In this example, pushers are modeled as generic because they are all identical and their properties are indirectly reflected in other data provided.

The network itself is defined in lines 17-35. Note that the second argument in a Queue definition is the type of resource stored in it. Links are defined with their predecessor node as a 2nd argument and their successor node as the 3rd argument. The 4th argument indicates the type of resource that flows through the link and is only required when the information can not be inferred from the predecessor or successor node.

Row 2 indicates the particular fleet configuration used for this run. These are actually decision variables and are defined in lines 39 - 41. Rows 2 - 5 provide additional problem parameters relating to cost, earth properties, and the constructed road. These parameters are defined in lines 42 - 50.

Line 54 defines a variable that indicates the rate of road advance per bank cubic meter of earth in place. This variable is defined in terms of the previously defined variables *ShrkFct* (shrinkage factor) and *RdCrsSct* (road cross-section area). Line 55 defines a variable that indicates the current haul distance. The distance is defined in terms of other variables and the system-maintained variable *MvdEarth.CurCount* which represents the amount of resource currently in Queue *MvdEarth* (in bcm of earth in place). As earth comes into the *MvdEarth* Queue, the value of *MvdEarth.CurCount* increases and the *Distance* variable is recalculated to reflect the change (these variables behave like cells in a spreadsheet). If other variables are defined in terms of *Distance*, then these are also updated. As a consequence, Stroboscope variables are always up to date.

Optimum scraper loading time and payload are a function of the particular scraper-pusher combination, the material being scraped, and the complete cycle-time net of load-time. Formulas — obtained by means not explored here — that yield optimum load-time and payload in terms of scraper capacity, scraper power, and haul distance, are shown in row 6. Corresponding variables are defined in lines 56-60. In the definition of these variables, scraper power and capacity are accessed through the system-maintained variables *Load.Scraper.Pow.SumVal* and *Load.Scraper.Cap.SumVal*; these represent the sum of the power and capacity of all the scrapers currently involved in the instance of the *Load* activity that is in context. It is necessary to use an aggregate value (e.g., the sum) because in Stroboscope several units of the same type could be involved in the same activity instance. In general, default link attributes provide only one characterized resource to an activity, but link attributes (similar to SQL queries) can override this and select as many characterized resources as needed to meet any user-specified criteria.

The *Load* Combi represents, in addition to loading time, the time required for scrapers to spot and make contact with the pushers. Row 7 shows these durations, where the Beta functions use the 5th percentile, mode, and 95th percentile as parameters. Lines 61-62 define variables that represent these times. These variables are used to set the *duration* attribute for the *Load* Combi in line 80.

Row 7 also shows the distribution of the realized payload. Line 81 sets the *release amount* attribute of link *ER1* so that the amount of earth that is released by the *Load* Combi follows this distribution. Default link attributes release an amount equal to the amount of resource (of the type that flows through the link) that came into the activity. If the *release amount* attribute had not been set, the amount of earth released would have been zero since no link brings in earth to the *Load* Combi. Links *ER2* and *ER3* transfer the amount of earth released by the *Load* activity without a need to set link attributes. Eventually, this amount of earth is added to the *MvdEarth* Queue (in bcm).

The *BkTrack* Normal activity represents the time during which pushers boost, transfer, and then backtrack to prepare for another push. Row 8 shows these times. Line 82 sets the duration of the *BkTrack* Normal to the sum of these two times.

Lines 64-70 define variables that ultimately compute the theoretical duration of a haul (variable *ThHaulTm*). Variables progress from the determination of the vehicles' gross weight (*GrossWgt*), to the force required to haul (*HaulFrc*), to the theoretical speed (*ThHaulV*), and then to the time. The *Haul.Earth.Count* system-maintained variable used to define gross weight represents the amount of earth used in the haul activity (which was received from the *Load* Combi). Actual haul time is shown in row 9. Line 83 sets the duration of the haul accordingly. Row 10 shows dump-time data, and line 84 sets the duration of the *Dump* activity.

Lines 71-76 define variables that ultimately compute the theoretical return time (*ThRetTm*). The computations are similar to those used to compute haul time, except that the vehicle is now empty, and the grade is favorable. Row 11 shows the actual return time distribution. Line 85 sets the duration of the *Return* activity accordingly.

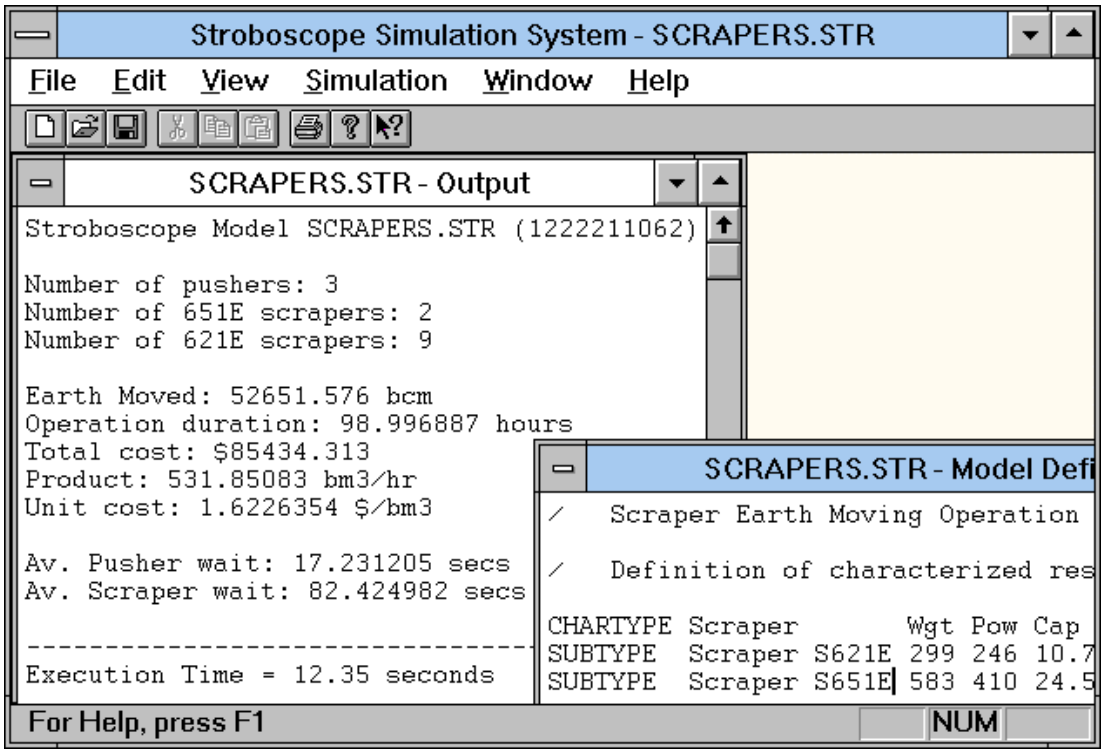

Figure 3 — Simulation Output

Resources are by default removed from Queues in the order in which they entered (FCFS). In this example the larger scrapers are given priority. Line 86 sets the *discipline* attribute of the *Scrprs* Queue so that scrapers with larger capacities are served first, regardless of how long they have been in the Queue.

Lines 90-92 set the initial content of the *Pshrs* and *Scrprs* Queues. The *Pshrs* Queue is initialized with an amount (variable *NumPshrs*). The *Scrprs* Queue is initialized twice, first with the number of 651E scrapers, and then with the number of 621E scrapers (variables *Num651E* and *Num621E* respectively).

At this point the model is completely defined and ready for the simulation to begin. The sequence of statements so far is flexible. The only requirement is that named model entities must be defined before they are used.

Line 96 instructs Stroboscope to run the simulation until the haul distance exceeds the final distance. Lines 100-104 compute the data that we want to extract from the simulation (post-processing in a spreadsheet or other application is not necessary); and Lines 108-119 instruct Stroboscope to display the information. Figure 3 shows the output from a run of this model.

Implementation

Stroboscope is implemented in two forms: a command-line compiler and an MS Windows integrated development environment (editor-debugger-compiler). Both are native MS Windows NT 32-bit programs developed in C++. The command-line compiler can run under DOS using the TNT DOS-extender. The integrated development environment can run under MS Windows 3.1, if the Win32s subsystem is installed.

Conclusion

The example presented in this paper illustrates Stroboscope's power as a general-purpose simulation system. It is easy to learn and use and encompasses the desired modeling attributes discussed earlier. Complex and realistic problems such as the example presented can be modeled, analyzed and optimized quite easily.

Appendix I - References

Halpin, Daniel W., and Riggs, Leland S. (1992). *Planning and Analysis of Construction Operations*, John Wiley & Sons., New York.

Appendix II - Stroboscope Source Listing for Earth-Moving Example

```
 1  /   Scraper Earth Moving Operation
 2
 3  /   Definition of characterized resources.
 4
 5  CHARTYPE Scraper       Wgt Pow  Cap MaxV Cost  Eff;
 6  SUBTYPE  Scraper S621E 299 246 10.7   51   32 0.83;
 7  SUBTYPE  Scraper S651E 583 410 24.5   55  105 0.81;
 8
 9  /   Definition of generic resources.
10
11  GENTYPE Pusher;
12  GENTYPE Earth;
13
14  /   Statements that define the network. These can be
15  /   inferred directly from the network drawing.
```

```
16
17  QUEUE   Pshrs       Pusher;
18  QUEUE   Scrprs      Scraper;
19  QUEUE   MvdEarth    Earth;
20  COMBI   Load;
21  NORMAL  BkTrack;
22  NORMAL  Haul;
23  NORMAL  Dump;
24  NORMAL  Return;
25  LINK    ER1 Load    Haul        Earth;
26  LINK    ER2 Haul    Dump        Earth;
27  LINK    ER3 Dump    MvdEarth;
28  LINK    PS1 Pshrs   Load;
29  LINK    PS2 Load    BkTrack     Pusher;
30  LINK    PS3 BkTrack Pshrs;
31  LINK    SC1 Scrprs  Load;
32  LINK    SC2 Load    Haul        Scraper;
33  LINK    SC3 Haul    Dump        Scraper;
34  LINK    SC4 Dump    Return      Scraper;
35  LINK    SC5 Return  Scrprs;
36
37  /   Model parameters and decision variables
38
39  VARIABLE NumPshrs 3;
40  VARIABLE Num651E  2;
41  VARIABLE Num621E  9;
42  VARIABLE PshrCost 55;
43  VARIABLE OthrCost 200;
44  VARIABLE EarthWgt 15.7;
45  VARIABLE ShrkFct  0.95;
46  VARIABLE InitDst  1000;
47  VARIABLE FinDist  5000;
48  VARIABLE RdCrsSct 12.5;
49  VARIABLE RollRst  0.03;
50  VARIABLE Grade    0.02;
51
52  /   Definition of auxiliary variables
53
54  VARIABLE DstPerBV 'ShrkFct/RdCrsSct';
55  VARIABLE Distance 'InitDst+DstPerBV*MvdEarth.CurCount';
56  VARIABLE OptLdTm  '125*(1+0.48*Ln[0.08*Distance/
57                        Load.Scraper.Pow.SumVal])';
58  VARIABLE OptPayLd 'Load.Scraper.Cap.SumVal*
59                        (1+Load.Scraper.Cap.SumVal/60*
60                        Ln[Distance/5000])';
61  VARIABLE SpotTime 'Pertpg[24,36,95]';
62  VARIABLE ActLdTm  'Pertpg[0.95*OptLdTm,
63                        OptLdTm,1.1*OptLdTm]';
64  VARIABLE GrossWgt 'Haul.Scraper.Wgt.SumVal+
65                        Haul.Earth.Count*EarthWgt';
66  VARIABLE HaulFrc  'GrossWgt*(Grade+RollRst)';
67  VARIABLE ThHaulV  'Min[Haul.Scraper.Pow.SumVal*
68                        Haul.Scraper.Eff.SumVal/HaulFrc,
```

```
                        Haul.Scraper.MaxV.SumVal/3.6]';
VARIABLE ThHaulTm 'Distance/ThHaulV';
VARIABLE RetFrc   'Return.Scraper.Wgt.SumVal*
                        (RollRst-Grade)';
VARIABLE ThRetV   'Min[Return.Scraper.Pow.SumVal*
                        Return.Scraper.Eff.SumVal/RetFrc,
                        Return.Scraper.MaxV.SumVal/3.6]';
VARIABLE ThRetTm  'Distance/ThRetV';

/   Set non-default element attributes

DURATION   Load    'SpotTime+ActLdTm';
RELEASEAMT ER1     'Normal[OptPayLd,0.3*OptPayLd]';
DURATION   BkTrack '0.15+0.40*OptLdTm';
DURATION   Haul    'Normal[ThHaulTm,0.25*ThHaulTm]';
DURATION   Dump    'Pertpg[24,36,78]';
DURATION   Return  'Normal[ThRetTm,0.15*ThRetTm]';
DISCIPLINE Scrprs  '-Cap'; / Comment out line for FIFO

/   Initialize the queues

INIT    Pshrs   NumPshrs;
INIT    Scrprs  Num651E   S651E;
INIT    Scrprs  Num621E   S621E;

/   Simulate until the 4 km segment is complete

SIMULATEUNTIL 'Distance>=FinDist';

/   Define variables to extract interesting results

VARIABLE TotHrCst 'NumPshrs*PshrCost+Num621E*S621E.Cost+
                        Num651E*S651E.Cost+OthrCost';
VARIABLE TotHrs   'SimTime/3600';
VARIABLE Product  'MvdEarth.CurCount/TotHrs';
VARIABLE UnitCost 'TotHrCst/Product';

/   Present results

DISPLAY "Number of pushers: " 'NumPshrs';
DISPLAY "Number of 651E scrapers: "    'Num651E';
DISPLAY "Number of 621E scrapers: " 'Num621E';
DISPLAY;
DISPLAY "Earth Moved: "'MvdEarth.CurCount' " bcm";
DISPLAY "Operation duration: " 'TotHrs' " hours";
DISPLAY "Total cost: $" 'TotHrCst * TotHrs';
DISPLAY "Product: " 'Product' " bm3/hr";
DISPLAY "Unit cost: " 'UnitCost' " $/bm3";
DISPLAY;
DISPLAY "Av. Pusher wait: " 'Pshrs.AveDur' " secs";
DISPLAY "Av. Scraper wait: " 'Scrprs.AveDur' " secs";
```