Proceedings of the 2000 Winter Simulation Conference
J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, eds.

AN MSE-BASED SIMULATION CAPABILITY FOR STRATEGIC AND TACTICAL LOGISTICS

Charles R. Standridge

Padnos School of Engineering
Grand Valley State University
301 West Fulton
Grand Rapids, MI 49504-6495, U.S.A.

David R. Heltne

Equilon Enterprises LLC
Westhollow Technology Center
P.O. Box 1380
Houston, TX 77251-1380, U.S.A.

ABSTRACT

We have developed and applied modeling, simulation, and analysis capabilities for addressing strategic and tactical logistics problems in the chemical industry. These problems have to do with determining capital equipment requirements and assessing alternative strategies for logistics operations. Applications require short turn around time. While similar in many respects, each application requires its own tailored solution. The modular simulation environment approach has been used to manage a set of software, including a commercial simulation environment, general purpose software, and application specific tools. This set of software tools supports quick model development and delivery of simulation results. Data organization strategies for voluminous model input data and simulation results have been defined. Through this application work, requirements for a more general MSE implementation have been established.

1 INTRODUCTION

Logistics has to do with the procurement, storage, and transportation of goods and people (Pritsker, Sigal, and Hammesfahr 1989). Strategic and tactical logistics has to do with what and how much transportation equipment, loading and unloading mechanisms, and storage is needed to meet logistics objectives. How to use the available logistics mechanisms effectively is of critical importance.

In an industrial environment, any resolution of these issues must deal with a number of interacting factors including:

1. A large number of customers whose demand for products may vary seasonally over the course of a year.
2. A large number of products.
3. Finite production capacity that may be shared between products.
4. Finite inventory storage capacity.
5. Finite loading and unloading capacity.
6. Multiple transportation modes (rail, truck, and marine).
7. Finite capacity yards for storing empty and full rail cars.

A wide variety of information must be obtained or developed and carefully studied to engineer an effective logistics system. This information includes:

1. Shipments to customers, daily and in total.
2. Inventory levels.
3. Utilization of loading and unloading equipment.
4. Time delays in converting orders to shipments.
5. Location of transportation equipment.
6. Utilization of transportation equipment.
7. Fleet size requirements given transit times.

Simulation is an ideal mechanism for constructing models of complex logistics systems that are characterized by voluminous input parameter values. Simulation experiments can generate a large number of detailed results to meet specific information requirements.

Simulation supports the evolution of models among similar projects. The input mechanisms and report generation capabilities developed for one project can be reused, modified and extended for the next. Models can be divided into modules or components (Standridge 1986). New models can be built from the components of previous models with modification and extension as necessary plus new components as required.

We describe a modular simulation environment (MSE) approach for addressing strategic and tactical logistics problems in an industrial environment with short turn around time requirements. Capabilities have evolved over a series of projects. Organization and input of a large number of input parameter values is supported. Organization and post simulation processing of a large set

of simulation results is included. An organizational structure for model components supports their evolution and reuse. The MSE architecture needed to support these applications is identified.

2 BACKGROUND

Ideally, the software tools used in performing a simulation project would be based on the specific requirements of that project (Standridge and Centeno 1994). The tool set would contain both simulation specific tools such as model builders and simulation engines as well as tools with wide applicability such as word processors, statistical analysis packages, graphical presentation packages, and spreadsheets. In addition, software developed specifically to support a particular project would be employed.

MSE concepts seek to provide the standard by which such an ad hoc collection of software tools can be used together. These concepts specify how simulation related data flows between tools in a general way so that heterogeneous software can work together. Concepts for the organization of simulation input data and experiment results are included (Standridge, et al. 1996).

Standridge (1999) proposes that an MSE implementation based on an object manager architecture. Such an architecture provides the capabilities to add and delete software tools as necessary as well as to control the flow of data between the software tools. Each software tool and each data set can be viewed as an object with certain attributes. The object manager controls the invocation of the software tools as well as meeting input data requirements and managing the results of each operation. A structure for organizing all simulation data is included.

Simulation has long been used to support various types of logistics applications. A few sample applications follow.

Auterio (1974) reported the use of simulation as a means for managers to measure the productive capacity and effectiveness of the Dover Air Force Base airlift system.

Sherall et al. (1992) used a simulation model to estimate the time interval starting when an aircraft begins its landing and ending when the aircraft begins its turn onto a runway exit as well as the location on the runway where the aircraft finishes coasting after landing. This information, as well as a feasible set of runway exit locations determined separately, are input to a dynamic programming model that determines the optimal locations of runway exits.

Bruzzone and Signorile (1998) discuss a simulation model of a harbor operation that periodically invokes optimization models. Two genetic optimization models are invoke every 14 simulated days: one to prioritize the unloading and loading of ships arriving in the next 14 days and the second to specify the location of cargo to load on the ships within the harbor.

Takakuwa and Fujii (1999) discuss a modular method for modeling transshipment systems that ship multiple products to multiple customers. In route storage and further processing are allowed.

Archibald, Karabakal, and Karlsson (1999) discuss the use of simulation in comparing supply chain alternatives.

3 DATA INPUT ISSUES AND ORGANIZATION

Simulation models addressing strategic and tactical logistics issues often require voluminous data input values. For example, one particular application required about 2000 lines of input values, including embedded comments. A reliable input procedure having a short implementation time per project is required.

The input procedure must recognize a modular and hierarchical relationship among the inputs. This supports organizing inputs into multiple files.

A collection of one of each of the required input files defines a simulation scenario or case to be evaluated. Any particular file may be associated with one or more scenarios. The input mechanism must support the one-to-many relationship between input files and cases.

Within an input file, the same set of quantities may be repeated to describe multiple entities of the same type. For example, products may be described by monthly demand, transportation mode used for delivery, and a list of feasible loading spots.

Given these requirements, input data was processed as follows. Each case is given an ID consisting of two parts:

1. Project name
2. Case index

For example, the case ID for project Rail and case index A1 would be RailA1.

A file name is composed of the case ID plus an identifier of the information in the file. For example, the product definition file for project Rail and case index A1 would be named RailA1Product.

A generic parser was developed for loading input data in a specified format, including a hierarchy of input files, directly into the variables of a particular simulation language, in our case Visual SLAM (Pritsker and O'Reilly 1999). Real variables (XX), integer variables (LL), and character variables (*SZ) may be assigned values. General error checking is performed.

Consider the following example input lines for two products:

```
// Product 1
/X
/B1000
1/ID NUMBER: 1;
2/MONTHLY DEMAND: 500;
3/TRANPORTATION MODE: 1;
10..12/LOADING SPOTS: 10, 11, 12;
/S
/B100
1/PRODUCT NAME: P1;
//
// Product 2
/X
/P50
1/ID NUMBER: 2;
2/MONTHLY DEMAND: 400;
3/TRANPORTATION MODE: 2;
10..12/LOADING SPOTS: 20, 21, 0;
/S
/P1
1/PRODUCT NAME: P2;
```

The format of a data input line is:

```
Offset value/comment: input value;
```

Lines that begin with a slash (/) are parser commands. The character following the slash defines the command. Two slashes (//) beginning a line indicate a comment.

The parser supports base-offset addressing to facilitate the repeated input of the same set of values to support multiple entities of the same type. A line beginning /X tells the parser to store the following values as real values in the Visual SLAM variable XX. A line beginning /B defines the base of the address of the XX variable. The value preceding the slash on a data input line is the offset. Thus, the ID NUMBER of product 1 is stored in XX[1001].

A line beginning /S tells the parser to store the following values as characters referenced via character pointers stored in the array SZ. Thus, the pointer to the name of the first product is stored in SZ[101]

A line beginning /Pv tells the parser to add v to the base of the variable currently receiving input. Thus, the line P50 that values the line /X adds 50 to the XX variable base, giving this quantity a value of 1050. The ID NUMBER of product 2 is stored in XX[1051].

Suppose for project rail a case is described by two input files, one for products and one for loading spots. Three input files would be defined for each case. The main input file for case a1, RailA1, would be as follows.

```
//Main input file for project Rail
(RailA1.dat)

//Product File
/IRailA1Product.dat

//Loading Spot File
/IRailA1Spot.dat
```

The main input file uses the /I parser command to input the values contained in all of the other input files. Our practices is that no input values are given in the main input file. Thus, the main input file defines a case based on the particular input files it references.

Each input file is named based on the first case in which it is used. Each input file may be included by the main input files associated with any number of cases.

Project specific input checking is coded as an AweSim user insert that is invoked at the start of each simulation run.

4 SIMULATION RESULTS ORGANIZATION AND POST-PROCESSING

Simulation results are organized into text files generated from user written code. We have found that project specific reporting is necessary. Simulation results must be organized and reported for effective use by both simulation analysts and system experts in order to gain the insights required for understanding logistic operations. Proper management of model inputs and results associated with numerous cases is required.

There are two types of files. One type contains statistical summaries and is written at the end of each replicate. Summary statistics include minimums, averages, maximums, and counts.

The other type of file periodically records variable values. Such log files are written daily, weekly, or monthly.

Files have a standard format as shown in Figure 1. Note that the name of the case and the main input file used in simulating the case are included in the results file. This documents the link between input values and results. Rows correspond to replicates for files containing statistical summaries. Result information is recorded in fixed width columns for each performance measure of interest. For log files, the first variable value is the simulation time at which the information was written.

Report Name
Report Description
Case ID
Main Input File Name

Replicate	VarName1	VarName2	…	VarNamen
1	Value1	Value2	…	Valuen
2				
.				
.				
.				
r				

Figure 1: Simulation Result File Format

The name of each file is formed from the case ID and a descriptor of the file contents. For example, the name of

the file containing product shipment summary statistics for project Rail and case index A1 could be RailA1Shipped.

Typical simulation result files for a strategic and tactical logistics project would include:

1. A log file of the volume of each product shipped each day by shipping mode.
2. A log file of the volume of each product shipped each month by shipping mode.
3. Statistical summary files for the minimum, average, and maximum volume of product shipped daily.
4. Statistical summary files for the minimum, average, and maximum time delay from receipt of a customer order to shipment.
5. Statistical summary files for the minimum, average, and maximum inventory of each product.
6. Statistical summary files for the average and maximum number of loads completed at each loading spot each day.
7. A summary file of the total volume of each product shipped from each loading spot.
8. Statistical summary files related to fleet sizing including the minimum, average, and maximum number of rail cars, trucks, and barges at each location and in transit as well as the maximum number required.

It was difficult to examine and interpret the voluminous information recorded in the simulation results files. Results in one file would need to be examined concurrently with results in other files. Not all of the results were relevant to answering questions of interest. These questions often changed over the course of the project.

Thus, a requirement arose to summarize simulation statistical results in a single file. This was accomplished in two steps.

First, a generic simulation replication analysis file was defined for use on all projects. This file has one section for each simulation statistical summary file. File header information: report name, report description, case ID, and main input file name are copied. There is one row for each variable in the simulation statistical summary file. Replication analysis is performed. Columns show the results of the replication analysis: average, standard deviation, standard error, 95% and 99% confidence intervals, coefficient of variation and number of replicates.

A standalone program was written to produce the replication analysis file. Inputs included a list of the simulation statistical results files of interest. The default name for a replication analysis file is the case ID concatenated with "Summary".

In some projects, a second set of summary files was required. These files include only the information specifically relevant to addressing particular project issues.

For example, the number of loading spots for each product may be at issue. A further complication is that each loading spot could serve more than one product. Total volume shipped and statistical summaries of daily shipping activity at each loading spot by product are relevant. The information is sorted in two ways: by loading spot and by product.

A standalone program was written to reorganize the replication analysis information as required. Inputs included the relevant replication analysis file and case ID. The program could reference the case input data as needed, primarily for labeling purposes.

5 MODELING AND COMPONENT MODELS

Because of short turn around requirements, it is necessary to reuse as much as possible of previously developed models on each new project. At the same time, each project must have its own tailored model.

The strategy of component models (Standridge 1986) was employed. Models were divided into modules implemented as multiple AweSim networks. Component models include:

1. Generation of demand for products.
2. Loading products and incrementing inventories.
3. Shipping products and decrementing inventories.
4. Receiving and unloading products at customer sites.
5. Managing rail yards.
6. Managing liquid inventories in tanks.
7. Managing loading spot schedules.
8. Managing production schedules and campaign wheels.

New models are constructed by combining, embellishing, and extending modules developed for previous models as well as developing new modules.

A standard structure for integrating discrete events coded in C with the AweSim networks was employed. This is shown in Figure 2.

The SLAMBASE file contains the following user written routines that Visual SLAM invokes: INTLC, EVENT, and OTPUT. INTLC invokes in turn the prompter that interactively ask the modeler for the case ID and case descriptor, the parser that loads the input data, and UserINTLC that performs all of the computations needed at the beginning of each replicate. UserINTLC invokes a user written error checking routine, UserCheck, that checks the data loaded by the parser as well as SetReport that writes the report file headers. Events needed for model execution and log file reporting are each written as C functions and invoked from EVENT. UserOTPUT writes the statistical summary report files at the end of each replicate.

The function USERF is used to compute all quantities and random samples needed in the model, time delays for example. The function ALLOC is used to determine all complex resource allocations such as which loading spot from a list to employ.

6 THE MSE-BASED ARCHITECTURE

The MSE-Based architecture for the simulation environment that supports the short turn around time development of simulation models of strategic and tactical logistics operations is shown in Figure 3.

File types within the environment include:

1. Input files, as discussed in Section 3.
2. Simulation result files, both simulation log files that periodically record variable values and statistical summary files written at the end of each simulation replicate.
3. Replication analysis files that summarize the statistical summary files, as discussed in section 4.
4. Files reorganizing the replication analysis files as discussed in section 5.

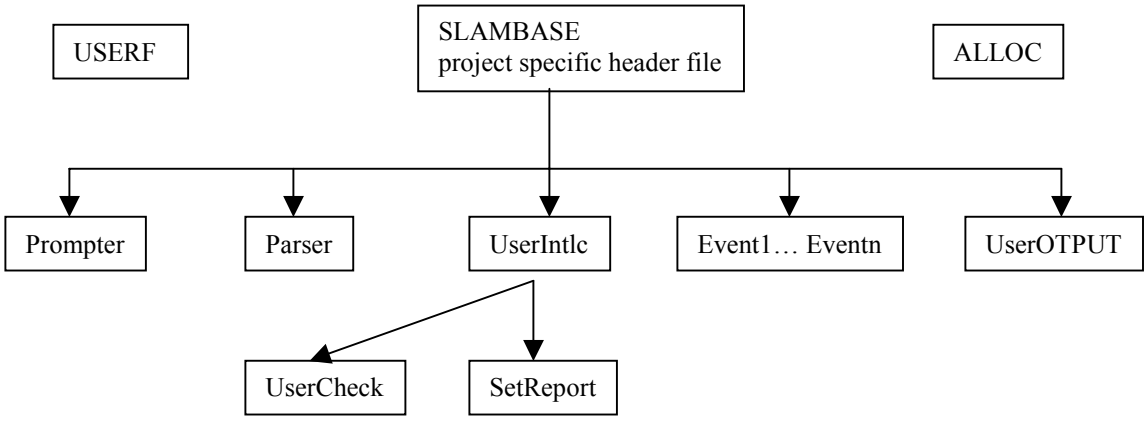

Figure 2: Standard Structure for AweSim User Inserts

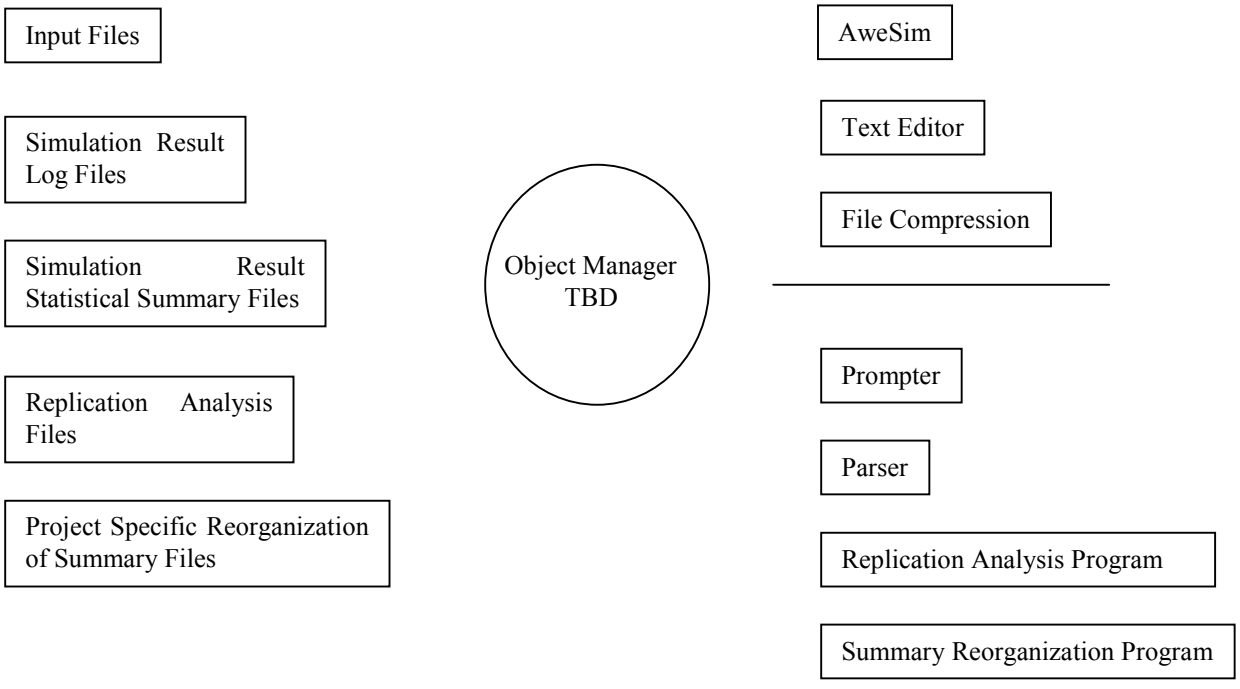

Figure 3: MSE-Based Architecture

The AweSim simulation environment is employed along with a commercial text editor and a file compression program. The text editor is used in the preparation of simulation inputs and for the examination of simulation results. The file compression program is applied to all data files associated with a particular case to facilitate archival storage and electronic transmission.

Programs for use on all of the strategic and tactical logistics projects are included in the environment. These include:

1. A prompter for the case ID and case descriptor that are include in the simulation result files.
2. A parser for model input data as was discussed in section 3.
3. A replication analysis program for statistical summary files to consolidate and to perform replication analysis on voluminous simulation results to facilitate examination and understanding.

In addition, some projects require a reorganized presentation of the replication analysis results to quickly and effectively deal with project issues.

The object manager for the MSE is under development. Requirements for data organization and management as well as program management and use have been revised and extended as a result of the strategic and tactical logistics application projects.

7 SUMMARY

We have presented an MSE-based modeling and simulation capability for industrial strategic and tactical logistics issues. The MSE includes of an organization for model inputs and simulation results. Commercially developed programs within the MSE include a simulation environment, a text editor, and a file compression program. Other programs have been written specifically to support strategic and tactical logistics applications. These include a prompter for case information, a parser for simulation inputs, and a replication analysis capability for simulation results. Additional programs are required for some applications to present simulation results in a manor that quickly addresses project issues.

REFERENCES

Archibald, G., N. Karabakal, and P. Karlsson. 1999. Supply chain vs. supply chain: using simulation to compete beyond the four walls. In *Proceedings of the 1999 Winter Simulation Conference*, ed., P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, 1207-1214. Institute of Electrical and Electronics Engineers, Piscataway, NJ.

Auterio, V. J. 1974. A Q-GERT simulation model of air terminal cargo facilities. In *Proceedings, Pittsburgh Modeling and Simulation Conference*, 5, 1181-1186.

Bruzzone, A. and R. Signorile. 1998. Simulation and genetic algorithms for ship planning and shipyard layout. *Simulation*, 71(2), 74-83.

Pritsker, A. A. B., C. E. Sigal, and R. D. J. Hammesfahr. 1989. *SLAM II network models for decision support.* Englewood Cliffs, NJ: Prentice-Hall.

Pritsker, A. A. B., and J. J. O'Reilly. 1999. *Simulation with Visual SLAM and AweSim, 2nd edition***.** New York: Halsted Press.

Sherali, H. D., A. G. Hobeika, A. A. Trani, and B. J. Kim. 1992. An integrated simulation and dynamic programming approach for determining optimal runway exit locations. *Management Science*, 38(7): 1049-1062.

Standridge, C. R. 1999. Modular simulation environments: an object manager based architecture**.** In *Proceedings of the 1999 Winter Simulation Conference*, ed., P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, 598-602. Institute of Electrical and Electronics Engineers, Piscataway, NJ.

Standridge, C. R., 1996, Progress in modular simulation environments. In *Proceedings of the 1996 Winter Simulation Conference*, ed., J. Charnes, D. Morrice, D. Brunner, and J. Swain, 714-720. Institute of Electrical and Electronics Engineers, Piscataway, NJ.

Standridge, C. R. and M. A. Centeno. 1994. Concepts for modular simulation environments. In *Proceedings of the 1994 Winter Simulation Conference*, ed., J. D. Tew, S. Manivannan, D. A. Sadowski, and A. F. Seila, 657-663. Institute of Electrical and Electronics Engineers, Piscataway, NJ.

Standridge, C. R. 1986. An approach to model composition from existing modules. In *Modeling and Simulation in the Artificial Intelligence Era***,** ed., M. S. Elzas, T. I. Oren and B. P. Zeigler. North-Holland.

Takakuwa, S. and T Fujii. 1999. A practical module-based simulation model for transshipment-inventory systems. In *Proceedings of the 1999 Winter Simulation Conference*, ed., P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, 1324-1332. Institute of Electrical and Electronics Engineers, Piscataway, NJ.

AUTHOR BIOGRAPHIES

CHARLES R. STANDRIDGE is an associate professor in the Padnos School of Engineering at Grand Valley State University. He has over 25 years of simulation experience in academia and industry. He has performed many simulation applications, developed commercial simulation software, and taught simulation at three universities. His current research interests are in the development of modular simulation environments (MSE). He is working

with industry on the application of MSE to strategic and tactical logistics problems. His simulation teaching interests are in the use of computer aided teaching studios for instruction of introductory undergraduate and graduate courses using a case-based approach. He has a Ph.D. in Industrial Engineering from Purdue University. His email address is <standric@gvsu.edu>.

DAVID R. HELTNE is a member of the technical staff in the Statistics Department, Engineering RD&T Directorate, Westhollow Technology Center, Equilon Technology. He works as an internal OR consultant to all the Shell companies. Prior assignments have emphasized the optimization and simulation modeling for decision support and resource allocation. Projects have ranged from rail systems design to batch plant testing to nonlinear refinery planning. He joined the Shell companies in 1980. Prior to this time, he taught in the areas of design optimization, simulation modeling and operations research in the Industrial Engineering Program at the University of Iowa. Dave has a Ph.D. in Chemical Engineering from the University of Iowa. His email address is <drheltne@equilon.com>.