Proceedings of the 1990 Winter Simulation Conference
Osman Balci, Randall P. Sadowski, Richard E. Nance (eds.)

SPECIAL PURPOSE SIMULATOR DEVELOPMENT

Marvin S. Seppanen

Productive Systems
Route Five, Box 46
Winona, MN 55987

ABSTRACT

This tutorial is based on seven years of experience at developing special purpose simulators using standard simulation languages. These simulators were designed to be used by manufacturing engineers or managers with little or no simulation modeling experience. These simulators have used Lotus worksheets or model development programs to provide the system data to be simulated. The results are available by either screen displays during the simulation or via data files which can be convert to Lotus graphs or viewed using data analysis programs. This tutorial outlines the common strategies and procedures used to develop the special purpose simulators.

1. INTRODUCTION

This tutorial begins by highlighting the major features and history of two special purpose simulators developed by Productive Systems for a variety of clients. The simulation language features used to develop the special purpose simulators are outline. Input model data concerns are reviewed. Lastly, the output options for the special purpose simulators are discussed.

2. EXAMPLES

The first special purpose simulator developed by Productive Systems occurred more by accident than by any predetermined design. The client wished to convert an existing mainframe simulation model to operate on a Personal Computer. In early 1984 SIMAN was the only major simulation language available for PCs. The original model was rather easily converted to operate on a PC using SIMAN. The major difficulty arose with the need to maintain three different model/data sets to simulate three different assembly plants. Further, it was expected that the data for each assembly plant would require frequent changes to permit the simulation of alternative assembly plant configurations. This data maintenance problem was complicated because some data changes would have to be made in three separate SIMAN Experiment Frame locations. While accurately making each of those changes was not critical to the simulation results, errors could confuse the people attempting to decipher the simulation results.

Productive Systems choose to develop a primitive, special purpose simulator to help solve this data input problem. Lotus 1-2-3 was selected to maintain a pair of worksheets for each assembly plant. A Parameters worksheet contained the general simulation run information: run length, number of runs, random number generator seeds, etc. A Commodities worksheet contained the detailed information for each assembly plant. Lotus 1-2-3 permitted the development of menus and macros to help the user to select and edit the worksheet information. A Lotus provided program was used to convert the final worksheets into the DIF format. A BASIC program was written to translate the worksheet data into a SIMAN Experiment Frame and a data file to be read by the main simulation program.

The main simulation program was the standard SIMAN simulation program plus Fortran code written by Productive Systems. That code read the input data file, executed the discrete and continuous simulation logic, and generated special reports. The SIMAN based simulation also saved OUTPUT files for conversion into Lotus graphs. The entire package was linked together by a menu which generated DOS Batch files to control the various processing steps.

The initial simulator has been modified for use by several clients. Throughout these modifications the basic structure of the special purpose simulator has remained unchanged. While each simulator exercise generates a completely new SIMAN Experiment Frame, the associated SIMAN Network Model has remained constant over the six year history of the simulator.

A second special purpose simulator was developed by Productive Systems for another client shortly after completing the first simulator. This simulator was designed to simulate packaging line configurations for a single food processing plant. Again a Lotus worksheet was used to enter and edit the data for the particular configuration being simulated. Like the first example, a BASIC program reads the DIF format worksheet data and generates the SIMAN Experiment Frame and other required data files. The SIMAN Network Model was not changed as alternative configurations were simulated. This was possible because the simulator's decision logic was included in the Fortran code written by Productive Systems. SIMAN OUTPUT file and special reports were generated by the main simulation program.

Since the original development of this second special purpose simulator, two complete revisions have been made for the original client. This first revision generalized the original single plant model so that it could be used to simulate all the packaging line configurations used by the client in its worldwide operations. The same general Lotus worksheet input and output approach was retained in this revision. A second revision was done for the client after several years of successful use of the special purpose simulators. This revision eliminated the Lotus worksheet and allowed the user to interact more directly with the simulation. The Lotus worksheets and associated programs for data conversion were replaced by an interactive model development program. That program uses a series of displays through which the user constructs and edits the simulation model data. The model development program also generates the required SIMAN Experiment Frame and a single data file which contains the complete model data. That data file is used by the main simulation program and can also be read by the model development program.

Since the release of SIMAN version 3.0, it has been possible for the simulation to be halted by pressing predefined keys on the PC. This feature was incorporated to allow the user to make certain model data changes while the simulation is being executed. In addition, the simulator periodically estimates the system's performance. If the estimated performance lies within previously defined confidence limits, the simulation is automatically terminated.

The latest simulator revision has changed its output structure. During the main simulation, on-screen performance statistics are periodically updated. After the simulation has been completed, a special purpose data analysis program is used to analyze the simulation data stored on a series of binary data files. The program displays the simulation results in both tabulator and graphical formats. In addition, the data analysis program can simultaneously display the results for up to three separate simulation runs. Statistical tests for differences are automatically performed to compare the data sets.

3. SIMULATION LANGUAGES

Several simulation language features have been exploited in order to develop the special purpose simulator described above. The primary requirement is the ability of the simulator developer to write code in Fortran or other computer language and to link that code with underlying simulation language. SIMAN and SLAM both have extensive capabilities for user generated code. GPSS/H also has similar but somewhat more limited capabilities. Because SIMAN and SLAM are very similar in this regard and because Productive Systems has the most experience with SIMAN, that language will be used throughout the remainder of this tutorial. Either Fortran or C is supported by SIMAN.

SIMAN presents the special purpose simulator developer with three opportunities for using the higher level computer language:

Beginning of Simulation Run
End of Simulation Run
During the Simulation Run

Figure 1 illustrates general options the SIMAN IV user has for incorporating Fortran or C code into the simulation language. All user routines can call SIMAN IV Support Routines. Those routines perform the standard simulation functions such as file handling and statistic collection.

3.1 Beginning of Simulation Run

SIMAN calls subroutine PRIME at the beginning of each SIMAN simulation run. Productive Systems uses this feature to read the input data file, initially fill the simulation structure with entities, and to open output data files. This initial processing can also schedule discrete events for data collection and simulation termination.

3.2 End of Simulation Run

SIMAN calls subroutine WRAPUP at the end of each SIMAN simulation run. Productive Systems uses this feature to write special reports, to determine if the simulation should be terminated, and to close the output data files.

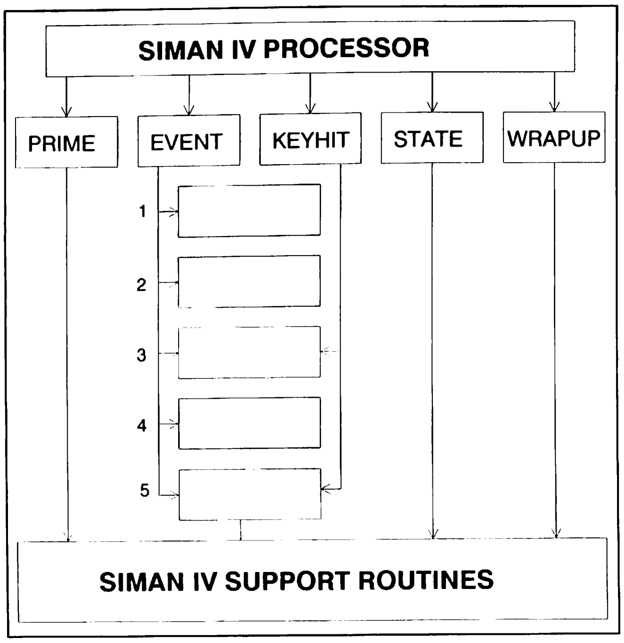


Figure 1. SIMAN IV User Interface Options

3.3 During the Simulation Run

Several SIMAN modes are available during the execution of the simulation.

All special purpose simulators written by Productive Systems have used SIMAN subroutine EVENT. Discrete events are either scheduled at the beginning of the simulation run via subroutine PRIME, rescheduled within subroutine EVENT, or started from the SIMAN Network Model via the EVENT Block. Productive Systems has found that extensive use of the EVENT subroutine tends to make the special purpose simulator more flexible. Productive Systems only uses the SIMAN Network Model when absolutely necessary, to control Resources and to drive the Cinema animation.

Several special purpose simulators written by Productive Systems have used SIMAN subroutine STATE to simulate continuous system. Subroutine STATE is automatically invoked by SIMAN. Normally subroutine STATE only calculates the revised values of the SIMAN system variables; D, S, and X. Subroutine STATE can also be used to schedule future events or to report data.

All special purpose simulators recently written by Productive Systems have used SIMAN subroutine KEYHIT. Subroutine KEYHIT allows the simulator user to halt the simulation at any time by pressing predefined keys. Productive Systems has used subroutine KEYHIT to save the current model status, clear the current model data, generate special displays, and to terminate the simulation.

A final method by which the special purpose simulator developer can access Fortran or C code is via call to function UF and UR from the SIMAN Network Model. Normally those functions are only used to calculate and return a single value to the entity in the SIMAN Network Model. However, more extensive use of these functions can make them trigger other simulator functions.

4. MODEL DATA FLOW

Figure 2 illustrates the general flow of model data in the simulators developed by Productive Systems. The letters XXXX represent the simulator name such as ALSS for the Advanced Assembly Line System Simulator, a special purpose simulator developed by Productive Systems. Table 1 lists the four user developed programs which comprise the simulator. Table 2 lists the four SIMAN IV programs which comprise the simulator.

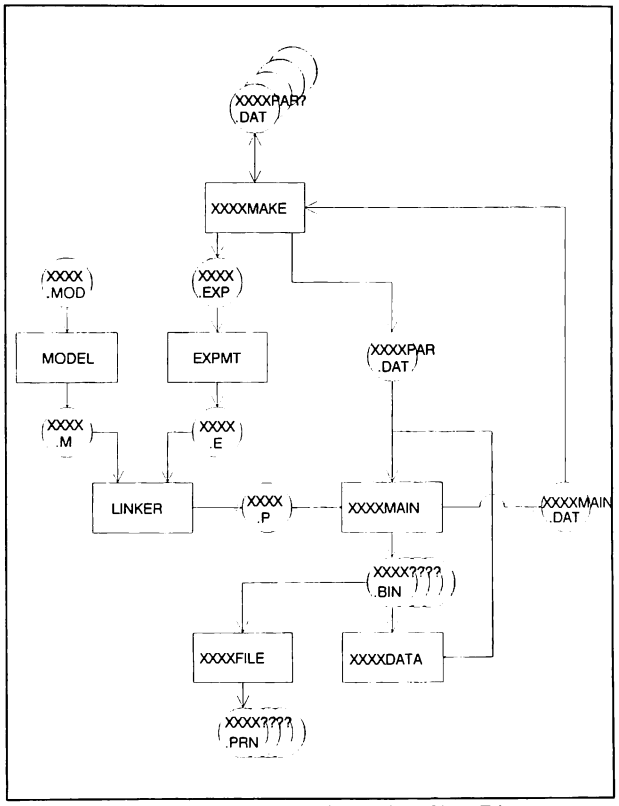


Figure 2. General Simulator Information Flow Diagram

Table 1. User Developed Simulator Programs

Program Name	Program Purpose
XXXXMAKE	Model Development Program
XXXXMAIN	Main Simulation Program
XXXXDATA	Data Analysis Program
XXXXFILE	Data File Translation Program

Table 2. SIMAN IV Simulator Programs

Program Name	Program Purpose
MODEL	Network Model Translation Program
EXPMT	Experiment Frame Translation Program
LINKER	SIMAN IV Data File Linker Program
XXXXMAIN	Main Simulation Program

The Model Development Program, XXXXMAKE, reads a model data file, XXXXPAR?.DAT, updates that model, saves the new model data as files XXXXPAR?.DAT and XXXXPAR.DAT, and constructs the corresponding SIMAN IV experiment file, XXXX.EXP. The (?) question mark is used to represent any letter or number to identify a specific simulator model. This convention allows the simulator to rapidly search to potential set of model data files. Model data file XXXXPAR.DAT is used as input for the remaining simulator programs. The experiment file XXXX.EXP is used to initiate the standard SIMAN IV progressing.

The Main Simulation Program, XXXXMAIN, incorporates the user written code into the SIMAN IV language as illustrated in Figure 1. After routine PRIME reads the model data file XXXXPAR.DAT to simulation process is begun. The main simulation program can generate a modified model data file XXXXMAIN.DAT which indicates the status the simulator at a particular time during the simulation. This model data file can be read by the model development program. The main simulation program can generate the standard SIMAN IV display output and report, standard SIMAN IV output file for processing as indicated in Figure 4, and special data files for additional simulator processing.

The Data Analysis Program, XXXXDATA, reads the model data file and the special data files generated by the main simulation program. The data analysis program allows the user to selectively view the simulation results in both tabular and graphical formats.

The Data File Translation Program, XXXXFILE, translates the special data files into formats suitable for input to standard programs such as, Lotus 1-2-3.

4.1 Model Input Data

Productive Systems has used both Lotus worksheets and model development programs to drive its special purpose simulators. In the past both options require computer programming. Starting with SIMAN IV, it is now possible using a READ Block in the SIMAN Network Model to directly read information from a variety of data format including Lotus worksheets. This approach may be reasonable for some limited applications, however, it cannot be used to alter the structure of the SIMAN Experiment Frame which is a characteristic of the Productive Systems' special purpose simulators.

Figure 3 illustrates the general framework used by Productive Systems to incorporate Lotus 1-2-3 as a simulator input medium. From a development time viewpoint, the Lotus worksheet option is much quicker to implement than a model development program. Most PC users are familiar with the data entering and editing features of Lotus worksheets. Lotus worksheets can be locked to prevent the accidental changing of critical data. Data editing and calculating can be built into the Lotus worksheet. During the simulator development phase, the Lotus worksheet can be easily changed or modified.

The downside of the Lotus worksheet in the need to write a special program, XXXXREAD.BAS, to read and translate the data required by the simulator. Productive Systems converts the worksheet data into the DIF format. The original versions of Lotus 1-2-3 and Symphony provided programs WKSDIF.EXE and WKRDIF.EXE to allow those conversions to be automatically done in a batch file. Unfortunately later Lotus versions omitted those programs. This forces the simulator user to follow the rather complex Lotus Translate

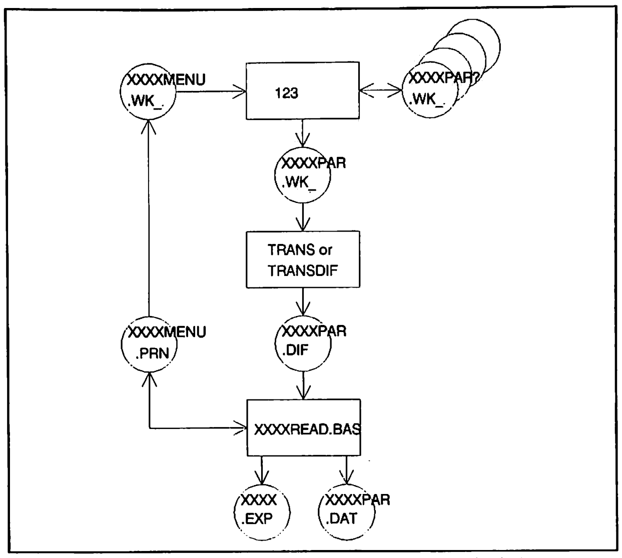


Figure 3. Simulator Input Using Lotus 1-2-3 Worksheet

menu to accomplish the required data format conversion. Fortunately, Lotus 1-2-3 version 3 again includes program TRANDIF.EXE to allow batch mode data format conversion. Productive Systems has favored the Basic language for the program to read the worksheet data and generate the SIMAN Experiment Frame. Basic programs can be quickly modified to reflect changes in the Lotus worksheet. The DIF format is difficult to read with Fortran. Some languages such as C and Prolog support the direct reading of data from the Lotus worksheet. While this method is appealing, its has not yet been used by Productive Systems.

Generating the SIMAN Experiment Frame presents some special problems. Non-alphabetic and non-numeric characters must be avoided. To be on the safe side, Productive Systems removes all special characters from the strings to be used in the SIMAN Experiment Frame. Certain numerical limits must be observed in generating the SIMAN Experiment Frame. Finally, the SIMAN Experiment Frame must be limited to 72 columns per record. This can be problem with some SIMAN Experiment Frame elements, such as, RESOURCES.

4.2 Model Output Data

The standard summary report provided by SIMAN at the end of each simulation run is the simplest and least meaningful simulation output format. Before SIMAN IV, using the standard summary report meant either printing the report as the simulation is executed or capturing the screen display on a data file. This later method has been extensively used by Productive Systems. It has two distinct drawbacks. First, capturing the screen display for the entire simulation duration make it impossible to use the SIMAN Debug Option or KEYHIT to check the status of the simulation during its execution. This problem has been addressed by SIMAN IV, which automatically saves the screen display on file with an extension of OUT. Second, the standard SIMAN summary report is all but useless for anyone not intimately familiar the underlying SIMAN model. Therefore, Productive Systems no longer makes direct use of the standard SIMAN summary report, rather, special reports or data files are saved as the simulation progresses. Before SIMAN IV, no option existed for generating special reports or data files without writing C or Fortran code. The SIMAN IV WRITE Block allows the saving of data directly from the SIMAN Network Model.

A second output data option available from SIMAN is the OUTPUT file. OUTPUT files can be saved for most statistics defined in the SIMAN Experiment Frame. The OUTPUT files can then be viewed or translated using the SIMAN OUTPT processor. Productive Systems has used the SIMAN OUTPT processor feature to create DIF file which can be translated into Lotus worksheets. Those worksheets can then be used to generate Lotus graphs. Productive Systems has been able to total automate this process. However, this method is slow and limited to the number of OUTPUT files which can be saved by SIMAN. Figure 4 illustrates the general flow of OUTPUT data. The SIMAN OUTPT program DIFFILE command converts the SIMAN OUTPUT files into the DIF format. Productive Systems has written program XXXXOUTP to combine a number of DIF format files into a single file. Lotus 1-2-3 version 1A program DIFWKS converts the combined file into a single worksheet. The translation program from later versions of Lotus 1-2-3 can also be used but may not support batch processing. Worksheet XXXXGRPH.WK_ supplies the graph labels and formats.

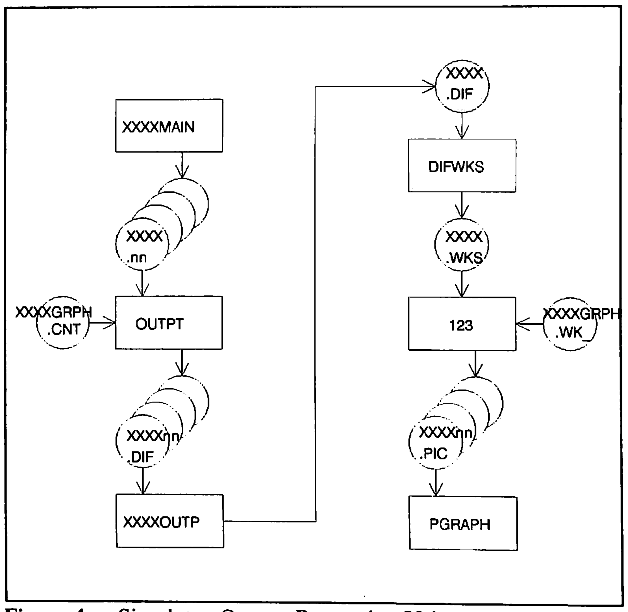
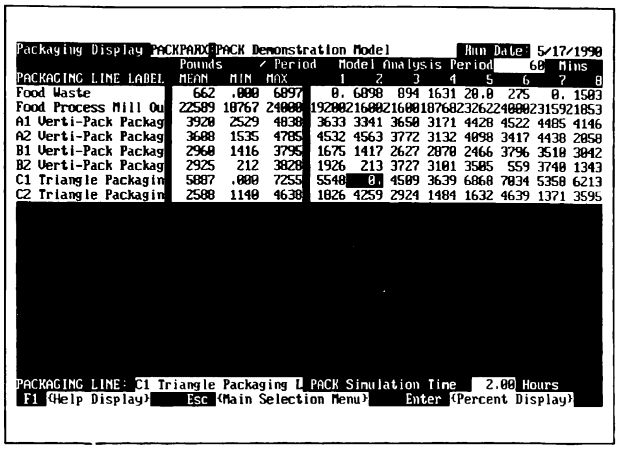


Figure 4. Simulator Output Processing Using Lotus 1-2-3

Data analysis programs are the favored approach taken by Productive Systems. While the development time for such programs is extensive, the result is an interactive program which allows the simulator user to explore a vast set of simulation results. A combination of tabular and graphical displays have proved helpful. Figure 5 illustrates a tabular display generated by the data analysis program for a special purpose simulator developed by Productive Systems. Figure 6 illustates a graphical display. The ability to compare the results from two or more simulation runs has proved to be very beneficial.



```
Packaging Display PACKPARX PACK Demonstration Model             Run Date: 5/17/1990
                  Pounds       / Period    Model Analysis Period        60 Mins
PACKAGING LINE LABEL  MEAN   MIN   MAX      1    2    3    4    5    6    7    8
Food Waste             662  .000  6897     0. 6898  894 1631 20.0  275   0. 1503
Food Process Mill Ou 22589 10767 24000 1920021600216001876823262240002315921853
A1 Verti-Pack Packag  3920  2529  4838   3633 3341 3650 3171 4428 4522 4485 4146
A2 Verti-Pack Packag  3688  1535  4785   4532 4563 3772 3132 4098 3417 4438 2058
B1 Verti-Pack Packag  2960  1416  3795   1675 1417 2627 2870 2466 3796 3510 3042
B2 Verti-Pack Packag  2925   212  3828   1926  213 3727 3101 3505  559 3740 1343
C1 Triangle Packagin  5887  .000  7255   5548   0. 4509 3639 6868 7034 5358 6213
C2 Triangle Packagin  2588  1140  4638   1026 4259 2924 1484 1632 4639 1371 3595



PACKAGING LINE: C1 Triangle Packaging L PACK Simulation Time    2.00 Hours
 F1 {Help Display}      Esc {Main Selection Menu}      Enter {Percent Display}
```

Figure 5. Special Purpose Simulator Data Analysis Program Tabular Display

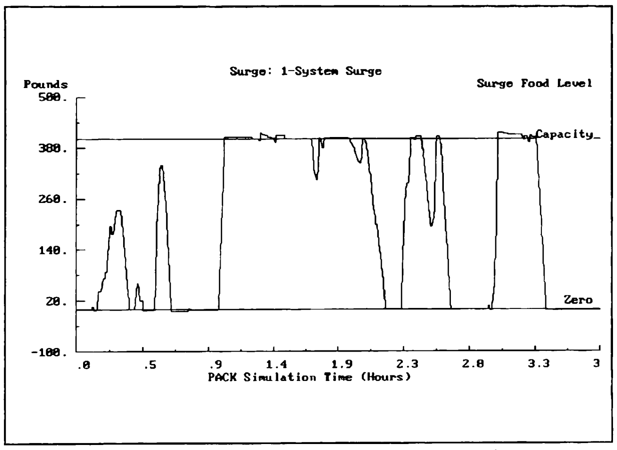


Figure 6. Special Purpose Simulator Data Analysis Program Graphical Display

5. CONCLUSIONS

The development of special purpose simulators using standard simulation languages have proven to be an effective tool in certain applications. First, the intended users should be engineers or managers with little or no simulation modeling experience. Second, the special purpose simulator must have a long expected life. The extended development time prohibits it's use for single use simulation models, except where those models involve numerous replications with different data sets.

The key to developing a good special purpose simulator is to write a small, fixed Network Model and to automatically generate the Experiment Frame each time the simulator is used. The first iteration of special purpose simulator development should use Lotus worksheet inputs and standard simulation outputs. Initially only a limited set of model options should be attempted. After the simulator has proven successful, it can be generalized and fitted with special purpose input and output programs. Productive Systems has yet to find a fast mechanism for the design and development of those programs.

Productive Systems has chosen to base its special purpose simulators on a major simulation language for several reasons. First, the simulation language supports the detailed simulation work; random variate generation, event scheduling, and statistic collection. Second, the simulation language provides an animation option, Cinema for SIMAN. Third, the simulation language provides an existing user base which adds to the simulator's creditability. Finally, the underlying major simulation language allows the experienced simulator user to use the simulator language to model systems which cannot be handled by the special purpose simulator.

REFERENCES

Pegden, C.D. (1982), *Introduction to SIMAN*, Systems Modeling Corporation, State College, PA.

Pritsker, A.A.B. (1984), *Introduction to Simulation and SLAM II*, Second Edition, Systems Publishing Corporation, West Lafayette, IN.

Systems Modeling Corporation (1989), *SIMAN IV Reference Guide*, Systems Modeling Corporation, Sewickley, PA.