GRAPHICS AND SIMULATION: TOOLS AND APPLICATIONS

Robin J. Miner, Jerome D. Sabuda, David B. Wortman
Pritsker & Associates, Inc.
West Lafayette, Indiana

ABSTRACT

As the state of the art in simulation has advanced, more effective ways of building simulation models and displaying their results have been developed. Computer graphics has played a key role in these developments. Although problems still exist, the introduction of computer graphics as an aid in the simulation modeling and analysis process has greatly encouraged the effective use of simulation by engineers, analysts and managers alike, enhancing their productivity and their effectiveness.

This paper discusses the advantages of using computer graphics within the simulation modeling process and some problems which may be encountered with its use. Generally applicable graphics software tools will be presented as well as applications of the use of graphics in specific models.

INTRODUCTION

Graphics has always played a vitally important role in the preservation and communication of information. Cavemen used hieroglyphics to provide a permanent record of their ever expanding world. These same hieroglyphic representations have provided us with the means for understanding how our ancient ancestors worked, played and learned.

In more recent times, graphics has been used to assist engineers and analysts in effectively carrying out their responsibilities. Free body diagrams, computer flow charts, electrical circuit diagrams, and PERT and GERT networks are all examples of how we have used graphics to represent systems and procedures. Plots, histograms and pie charts are the means by which we can graphically portray data. Naturally, there are many other examples of graphics applications that could be listed. However, the important point is that the transfer of large amounts of complex information can be greatly facilitated by the use of graphics.

Proceedings of the 1982
Winter Simulation Conference
Highland * Chao * Madrigal, Editors

82CH1844-0/82/0000-0161 $00.75 © 1982 IEEE

Simulation has been one area that has always benefited from graphics. Many simulation languages use network diagrams to portray the elements of a system and the relationships among those elements. As such, a network model is a graphical representation of a problem situation and provides the means for communicating that problem situation to others. Such communication is a vital part of any modeling and decision-making process.

The network approach allows an analyst to decompose a complex problem situation into elements from which a model can be developed. The syntax of the network modeling language specifies the form that such a model may take. The elements of the language and the options provided for interconnecting them determine the class of problems that can be solved with a particular network language. In addition, networks provide for standardization, comprehension, and consistency in modeling activities. In all these areas, the effectiveness of network models has been clearly demonstrated.

Another advantage of networks is that they separate the model building process from the analysis process. The construction of a network model does not require knowledge of the analysis procedures to be used in computing system performance measures. This situation is analogous to establishing a set of equations to represent a system without being burdened with a requirement for a solution technique to solve that set of equations. By segregating the modeling and analysis functions within a simulation effort, network languages enhance the modeling capability of the project team and, in general, improve communication by relegating the complex analysis procedures to an independent role.

The success of simulation efforts is also enhanced by the use of graphics to portray measures of system performance derived from simulation analysis. While tabular data and statistical summaries provide a great deal of important information, plots, histograms and pie charts are often used to provide analogous pictorial representations of that same information in a more concise and readable form than possible using other methods. While it often sounds trite, the old adage that "a picture is worth a thousand words" can most definitely be applied to the portrayal and analysis of simulation outputs.

With the advent of powerful computer graphics techniques, the effective use of graphics as an aid to the simulation process can be greatly expanded. While the widespread availability of computer graphics is only a recent development, it is currently being applied by many simulation practitioners with great success.

The link between the computer and the graphical representations that are used to support simulation projects provides a number of very significant benefits. First, computer graphics increases the speed with which models can be developed, implemented and documented. Further, it allows the use of graphical representations to provide direct input into the simulation analysis technique and, in many cases, as the means for portraying "traces" of system operation during or following the execution of the simulation model. In addition it allows us to easily and automatically generate presentation quality graphical portrayals of system performance. Overall, computer graphics improves the productivity of simulation analysts and the effectiveness of simulation projects.

In building network models, the current state of computer graphics allows symbols to be defined, placed on a computer screen, moved to any desired location and interconnected with other symbols. Thus, the various symbols provided by a network simulation language can be stored on the computer and made available to the model building via menu selection, allowing him to build a network model while seated at a terminal. This also gives him the ability to use the computer to edit and revise the network, to prepare data input statements automatically, and to store the network and associated input statements in a file or database for subsequent revision or execution. In addition, hard copy outputs can be obtained automatically using a hard copy device connected to the terminal or computer system. IDEF_2, (7,16) a new simulation language designed for analyzing manufacturing systems, makes extensive use of computer graphics in building system models in network form.

Once the model is stored on the computer, execution using the underlying simulation analysis software can be requested from the terminal. Detailed information concerning the execution of the model can then be stored in a file or database for subsequent, automated analysis. The analyst can, while sitting at the terminal, request a set of data or multiple sets of data for portrayal in a graphical form on the screen. The data may represent different elements of the execution of a single scenario or the same element from a variety of system scenarios. In any event, data may be portrayed in any number of graphical forms including time series plots, histograms and pie charts upon request. These displays can then be reviewed, new displays requested, and permanent copies generated, all in a matter of minutes. In the most advanced systems, this even allows non-technical personnel to request and analyze simulated data, greatly facilitating the overall decision-making process. IDEF_2 also makes extensive use of computer graphics in this manner. In addition, SIMCHART™, a device independent interactive graphics software package, can be used with any simulation language to produce visual displays of simulation outputs.

Another important benefit that computer graphics has brought to simulation involves the viewing of "traces", or the dynamics of system operation, on either the network model or a facility diagram. Using this approach, both the analyst and the decision-maker gain confidence in the model's ability to accurately represent the system while allowing them to suggest changes that improve the accuracy of its representation. In this manner, the decision maker is brought directly into the modeling process once again, providing tremendous advantages for the model builder. Not only is his work better understood, but it also captures the decision-maker's experience, knowledge and understanding of the system. This interaction between the model builder and decision-maker is oriented towards demonstrating that the model reflects reality and, as a result, is part of the validation phase of the modeling process. Consequently, it serves to improve the decision-maker's confidence in the outputs of the model, resulting in a greater and more effective use of the model and its outputs.

A number of tools have been developed which incorporate computer graphics to assist in the simulation modeling and analysis process. In addition, applications of these tools and the computer graphics foundation on which they are built have already been successfully demonstrated. Some of these tools and applications are presented in the following section. They serve to demonstrate the power of computer graphics in a simulation environment and the tremendous benefits that can be derived from using such techniques.

GENERAL PURPOSE GRAPHICS SOFTWARE

Most simulation graphics software is based on a general purpose graphics support package. There are several such packages available today which are computer and device independent. These include the Graphics Compatibility System (GCS)(4), distributed by the National Technical Information Service (NTIS), TEMPLATE (14), distributed by Megatek Corporation, the Terminal Independent Graphics System (TIGS) (15), distributed by Control Data Corporation, and DISSPLA (1), distributed by ISSCO. Each of these packages consists of libraries of user callable programs that perform specific basic graphic functions such as drawing lines or erasing the screen. The user creates the desired displays by writing computer programs which access the basic functions in the appropriate sequence. There are several problems associated with using general purpose graphics software. First, there is currently no standardization of computer graphics hardware devices. That is, the signals which must be sent to the graphics hardware device to perform a basic graphics function will vary from one manufacturer's hardware device to another. Thus, a software function which draws a line on a Tektronix graphics hardware device will not necessarily draw a line on a Hewlett-Packard device. One would like to write a general purpose graphics software package which would run on any type of graphics hardware device, but this is not easily accomplished. There are currently hundreds of different vendors which manufacture graphics hardware devices such as terminals, plotters, and hardcopy devices. Graphics software packages which are proported to be device independent usually have a separate module to interface with each type of graphics hardware device. Thus, a user who has only one type of graphics hardware device will have to pay the overhead of storing and compiling sections of a general purpose graphics software package which will not be used.

Another problem in using a general purpose graphics software package is the cost involved. It is still fairly expensive both to store and use general purpose graphics software. There are two factors emerging which will help reduce the costs of using computer graphics in the future. First, the cost of computer hardware, including computer graphics devices, is decreasing as the associated technology improves. Second, the work being done to increase standards in the graphics industry will decrease the cost of graphics by extending the applicability of computer graphics software. As the costs of using computer graphics decrease, their use within the simulation modeling process will become commonplace.

A general purpose graphics software package provides a great deal of capability to the graphics user. There are few graphics applications which could be accomplished without the support of a general purpose graphics software package. However, if the costs associated with the use of such a package for simulation are determined to be prohibitive, a user may elect to develop or use simulation specific graphics software.

SIMULATION SPECIFIC GRAPHICS SOFTWARE

Simulation specific graphics software packages are those which have been specifically developed for use with simulation languages. They are capable of graphically portraying particular aspects of a simulation model of any type of system which can be represented in a general purpose simulation language. To date, little simulation specific graphics software has been developed. However, as its availability increases, its use will become more widespread.

AID™

AID (3) is a new simulation specific graphics software package which allows the user to characterize data which will be used in a simulation model or data which has resulted from a simulation analysis. AID provides an interactive procedure for fitting theoretical statistical distributions to sets of data by utilizing statistical concepts as well as graphics capabilities. This type of data characterization can often be very difficult and time consuming. AID simplifies the data characterization task by guiding the user to a suitable statistical model of a set of data by proceeding through a series of logical steps: 1) data preparation, 2) histogram plotting, 3) sample cumulative distribution plotting, 4) theoretical distribution plotting, 5) parameters adjustment and 6) goodness-of-fit testing. Each of these steps relies heavily upon the use of computer graphics.

The data preparation step involves describing the data to be analyzed, and transforming it if necessary. The second step involves displaying the data to be analyzed in histogram form. The next step may be to plot the sample cumulative distribution. This may be compared to theoretical cumulative distribution which best fits the data. Next, one may select to plot theoretical distributions on the histogram of the data. This aids in visually determining if a theoretical distribution adequately represents a data set. If desired, the parameters of a theoretical distribution may be varied to adjust the theoretical distribution to better fit the data. Figure 1 shows a sample histogram, a normal distribution fit to the data set, and a normal

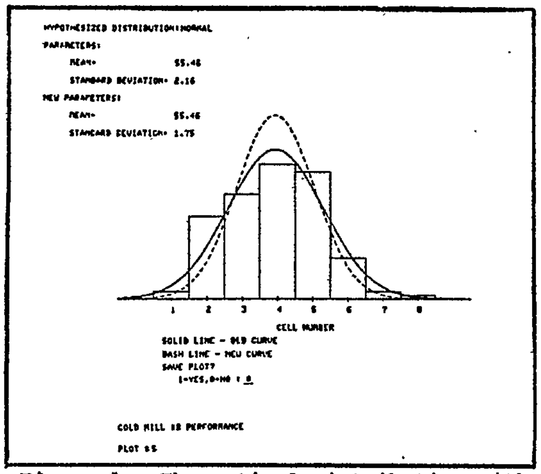

```
HYPOTHESIZED DISTRIBUTION:NORMAL
PARAMETERS:
     MEAN=              55.48
     STANDARD DEVIATION= 2.16
NEW PARAMETERS:
     MEAN=              55.48
     STANDARD DEVIATION= 1.75
SOLID LINE - OLD CURVE
DASH LINE - NEW CURVE
SAVE PLOT?
   1=YES,0=NO : 0

COLD MILL 18 PERFORMANCE

PLOT 55
```

Figure 1. Theoretical Distribution with New and Old Parameter Estimates

distribution with adjusted parameters fit to the data set. In this manner, a user may attempt to fit a distribution to a data set visually.

If a more rigorous statistical goodness-of-fit test is desired, the AID user may select either a chi-square or Kolmogorov-Smirnov (K-S) test to be performed. These tests provide a statistical basis for demonstrating that a data set fits a hypothesized distribution.

One of the unique characteristics of AID is its ability to graphically display the results of a statistical test. Figure 2 contains a graphic display of the chi-square test. The solid line on the graph bounds the acceptance region. If the test statistic falls within this area, then the hypothesis that the data fits the hypothesized statistical distribution cannot be rejected. If the test statistic falls within the critical region bounded by the dashed line, the hypothesis is rejected, and one concludes that the data does not fit the hypothesized statistical distribution.

Figure 3 contains a graphic display of the K-S test. If the sample cumulative distribution function (CDF) represented by the solid line touches or crosses the two dashed lines, which indicate the permissable deviation from the CDF, then the hypothesis that the data fits the hypothesized distribution is rejected.

The use of AID to support the simulation modeling process decreases the amount of time required to get data into a form usable by a simulation model and to summarize large quantities of output data. AID's use of computer graphics in this activity provides the modeler with unique and powerful data analysis capabilities.

SIMCHART™

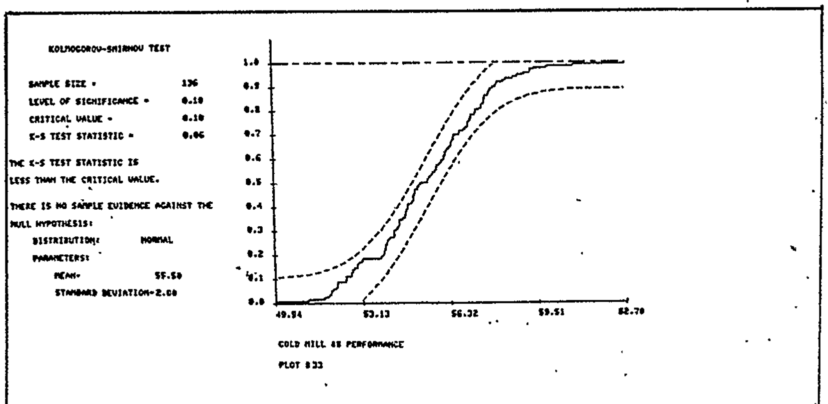
SIMCHART (2) is another graphics software package developed to interface with simulation languages. The goal of SIMCHART is to summarize large amounts of simulation data in a form familiar to the decision maker using the model. The use of SIMCHART to support the simulation modeling process has significantly reduced the time and effort required to analyze simulation results.

It enables the user to improve data analysis and presentation while reducing the labor requirements necessary to accomplish them. To use SIMCHART, the user first exercises the model to create a data file of simulation results. Then, the SIMCHART program is executed from a graphics terminal. SIMCHART operates interactively with the user to create display parameters and labels that define the presentation of data on the graphics terminal screen. As displays are requested by the user, SIMCHART accesses the simulation results database to produce the display defined by the specified parameters and labels.

SIMCHART is capable of producing four types of graphic displays: plots, histograms, pie charts, and pie graphs. Pie graphs are simply circular plots that are similar to an automobile speedometer. A SIMCHART plot of the level in a queue over time is shown in Figure 4. Figure 5 is an example of a histogram comparing two simulation runs. Figure 6 shows an example of a pie chart of the performance summary for a lathe. Figure 7 contains a pie graph of the level of crude oil in a storage tank.

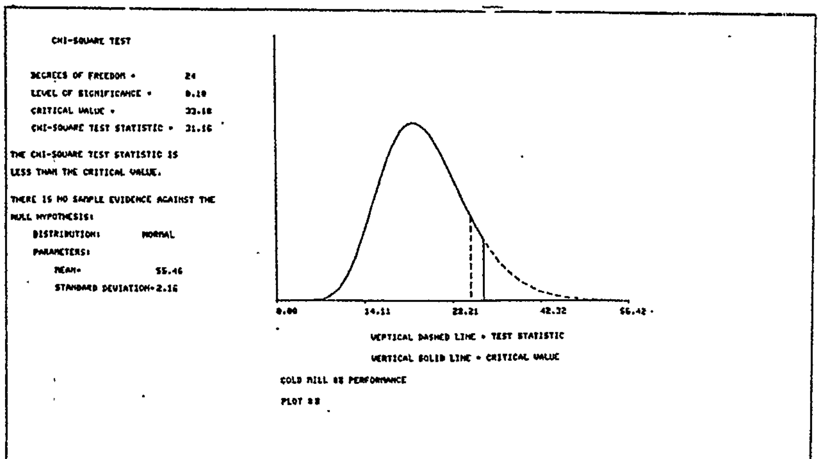
The SIMCHART user may display up to four different graphs at one time, positioning each in any one of twenty different locations on the screen. This feature facilitates the analysis of sequential operations in a system and the comparison of multiple simulation scenarios. These types of visual displays assist the analyst in grasping important cause and effect relationships quickly and efficiently.

Both AID and SIMCHART were designed to be used in conjunction with any simulation model. These graphic developments enhance the simulation modeling process by significantly reducing the time required to characterize and display simulation data. These packages have less overhead than the general purpose graphics software, but are more expensive to use than graphics tailored to specific types of simulation models.



```
     KOLMOGOROV-SMIRNOV TEST

   SAMPLE SIZE =                 136
   LEVEL OF SIGNIFICANCE =       0.10
   CRITICAL VALUE =              0.10
   K-S TEST STATISTIC =          0.06

THE K-S TEST STATISTIC IS
LESS THAN THE CRITICAL VALUE.

THERE IS NO SAMPLE EVIDENCE AGAINST THE
NULL HYPOTHESIS:
    DISTRIBUTION:        NORMAL
    PARAMETERS:
        MEAN=                55.50
        STANDARD DEVIATION=2.00
```

Figure 2. Chi-Square Test



```
     CHI-SQUARE TEST

   DEGREES OF FREEDOM =          24
   LEVEL OF SIGNIFICANCE =       0.10
   CRITICAL VALUE =              33.18
   CHI-SQUARE TEST STATISTIC =   31.16

THE CHI-SQUARE TEST STATISTIC IS
LESS THAN THE CRITICAL VALUE.

THERE IS NO SAMPLE EVIDENCE AGAINST THE
NULL HYPOTHESIS:
    DISTRIBUTION:        NORMAL
    PARAMETERS:
        MEAN=                55.46
        STANDARD DEVIATION=2.16
```

Figure 3. Kolmogorov-Smirnov Test

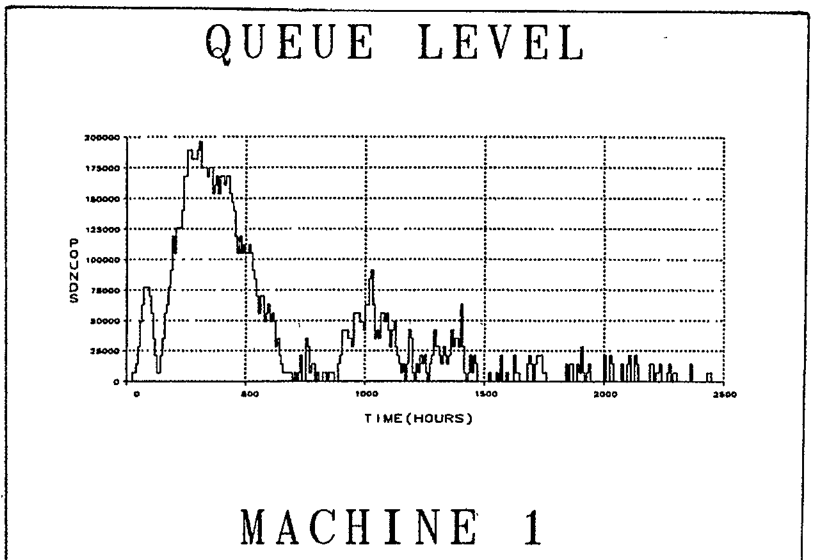


Figure 4. SIMCHART Plot of Number in Machine 1 Queue

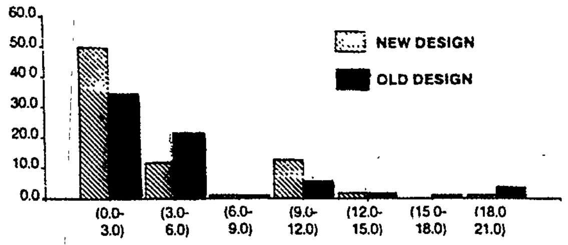
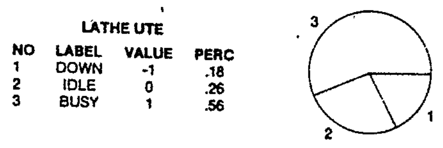
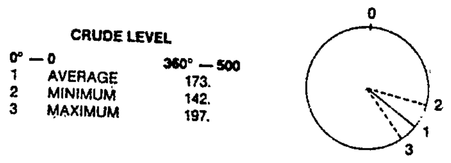


Figure 5. SIMCHART Histogram


LATHE UTE

NO	LABEL	VALUE	PERC
1	DOWN	-1	.18
2	IDLE	0	.26
3	BUSY	1	.56

Figure 6. SIMCHART Pie Chart


CRUDE LEVEL

0° — 0		360° — 500
1	AVERAGE	173.
2	MINIMUM	142.
3	MAXIMUM	197.

Figure 7. SIMCHART Pie Graph

GRAPHIC SIMULATION MODELING AND ANALYSIS SYSTEMS

In contrast to simulation specific graphics software which graphically describes data from any type of simulation model, graphic simulation modeling and analysis systems use graphics to support the simulation of a class of problems. These models typically take a graphic form which lends itself to computer graphics development and display. Two such systems are $IDEF_2$, for modeling aerospace manufacturing systems, and SOS^2, (11) for modeling nuclear safeguards systems.

$IDEF_2$

$IDEF_2$ is a new dynamics modeling language which was developed for the Air Force Integrated Computer Aided Manufacturing (ICAM) Program (7) to model the time varying behavior of aerospace manufacturing systems. The $IDEF_2$ language and software make heavy use of graphic concepts. The $IDEF_2$ language is in a network form, allowing models to be built graphically on a Tektronix 4014 graphics terminal. Once simulated, there are extensive graphic output capabilities which may be used to display simulation results.

The $IDEF_2$ language was designed with several graphic design criteria in mind. One of these was that $IDEF_2$ models have a graphic representation. Another was that the models be decomposable into small units each of which could be created and viewed individually. Thus, an $IDEF_2$ model consists of 4 submodels: the Facility Submodel, the Entity Flow Submodel, the Resource Disposition Submodel and the System Control Submodel. Each submodel has a graphic component and a non-graphic component and usually contains multiple pages.

A Facility Submodel describes the physical components of the manufacturing system being modeled. The graphic portion of the Facility Submodel is a Facility Diagram which identifies the physical components of the system and their relative locations. Figure 8 is an example of a Facility Diagram. It includes a SOURCE symbol named ARRIVE. This node indicates where materials arrive to the system to be modeled. The STORAGE symbol named STORAGE indicates where arriving materials wait to be processed on the MACHINE center named WORKBENCH by the MANPOWER type named REPAIRMAN. Materials depart the system at the location indicated by the DESTINATION node named DEPART.

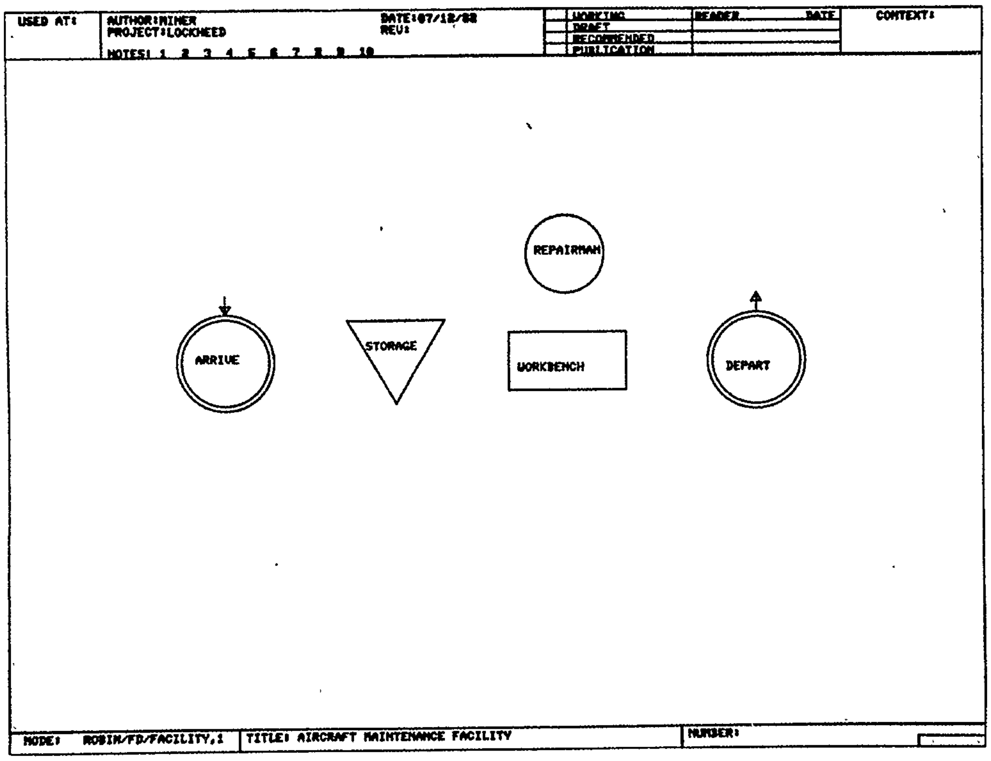


Figure 8. $IDEF_2$ Facility Diagram

An Entity Flow Submodel describes the processing of entities which flow through the manufacturing system. Entities may be such things as jobs, information, or reports. The Entity Flow Submodel contains graphic descriptions of entity processing called Entity Flow Networks as well as details concerning entity processing which are described non-graphically. An example of an Entity Flow Network is shown in Figure 9. Entities arrive to the START node named ARRIVAL and immediately pass to the QUEUE node named REPAIRQ. When one unit of resource REPAIRMAN becomes available to process an entity from REPAIRQ, the activity named REPAIR ENGINE begins. Upon completion of this activity, the entity departs this Entity Flow Network segment by proceeding to the END node named DEPART.

A Resource Disposition Submodel describes the rules for reallocating resources when they become available. The graphic portion of this submodel is a set of Resource Disposition Trees, one for each resource type in the model. Each tree describes the disposition procedures for a particular resource type. An example of a Resource Disposition Tree for the REPAIRMAN resource type is shown in Figure 10. When a REPAIRMAN becomes available the tree asks the question "ANY REQUESTS"? That is, is there an entity which is waiting for the REPAIRMAN? If the answer to this question is "YES", then one unit of the resource REPAIRMAN is allocated to process an entity waiting in the QUEUE named REPAIRQ. If there are no requests for the REPAIRMAN, then the REPAIRMAN is freed by the ACTION block labeled FREE.

The System Control Submodel describes conditions and events which affect but do not directly cause entity flow, such as breakdowns and repairs, periodic disruptions, and changes in resource levels. The graphic portion of the System Control Submodel consists of System Control Networks. An example of a System Control Network which creates the entities which will flow through the Entity Flow Network is shown in Figure 11. In this network, entities called ENGINE are created by the CREATE node called ARRIVALS. They are delayed by the activity called ENTER FACILITY and are then transferred to the Entity Flow Network to node arrival by the GOTO node.

The use of a diagram of the manufacturing system as part of a simulation model is a new idea. Most people begin the model building process by sketching the components of the system to be modeled. This sketch is frequently useful in relating to the system when building the model and also in describing or communicating the model to others. The ability to build this sketch on a graphics terminal along with the other portions of the model integrates the diagram of the system into the rest of the system representation.

The $IDEF_2$ language was designed with an emphasis on graphics support software. The $IDEF_2$ software system contains software to graphically build $IDEF_2$ models and to display outputs in graphical form. The graphic portion of each $IDEF_2$ submodel is built one page at a time, using graphics software on a Tektronix 4014 non-refresh graphics terminal. Each page of an $IDEF_2$ model is built by answering prompts in an interactive session. All model information is stored in a database which is referenced for each graphic display. Figure 12 contains an example of the Tektronix 4014 screen as an entity flow network is being created. To the left is the working area, where the user enters the commands causing the entity flow network to be created, edited or displayed. The drawing area of the screen is toward the upper right corner of the figure. This is where the Entity Flow Network is displayed during creation or editing. The lower area of the screen contains a menu of symbols and their associated parameters which are permitted to be used in the submodel being created. In the Entity Flow Submodel one may use START nodes, QUEUE nodes and ACTIVITIES. The only required display parameter for an ACTIVITY is the name of the ACTIVITY. Additional processing parameters of activity start node, activity end node and duration are necessary before an $IDEF_2$ model can be simulated.

Once an $IDEF_2$ model is created and simulated there are several means of graphically displaying simulation output with the $IDEF_2$ support software. Four types of graphical traces are available: a facility line trace, a facility animated trace, an entity line trace and an entity animated trace. The line traces track the passage of entities through Facility Diagrams or Entity Flow Networks on a Tektronix 4014 non-refresh graphics terminal. An example of a line trace on an Entity Flow Network is shown in Figure 13. Each time an entity completes the ACTIVITY named REPAIR ENGINE the hatch marks light up moving across the activity from left to right indicating the flow of an entity through the activity to the END node named DEPART. The time line across the bottom of the screen allows the user to track the occurrence of activity completions within the total simulated time. The facility animated trace and the entity animated trace are animated versions of the line traces. The animated traces are available on Tektronix 4112 refresh graphics terminals. These traces illustrate the movement over simulated time of entities through either an Entity Flow Network or a Facility Diagram. Other types of graphic displays of outputs available in the $IDEF_2$ software include histograms and plots.

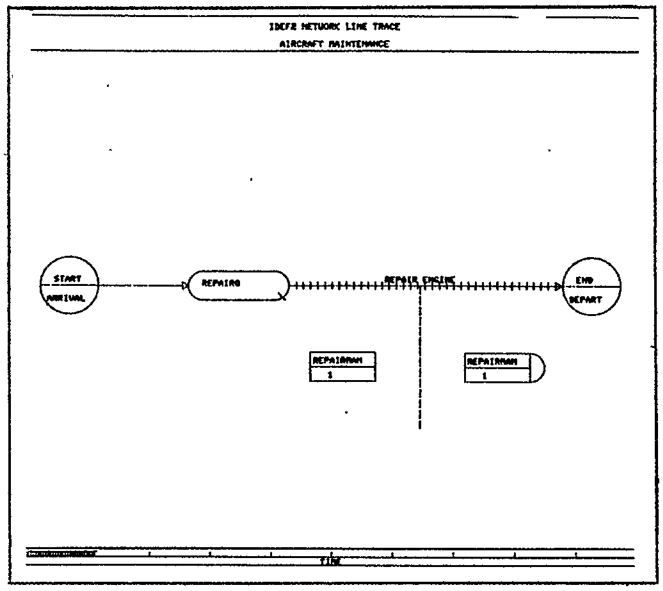


Figure 9. IDEF$_2$ Entity Flow Network

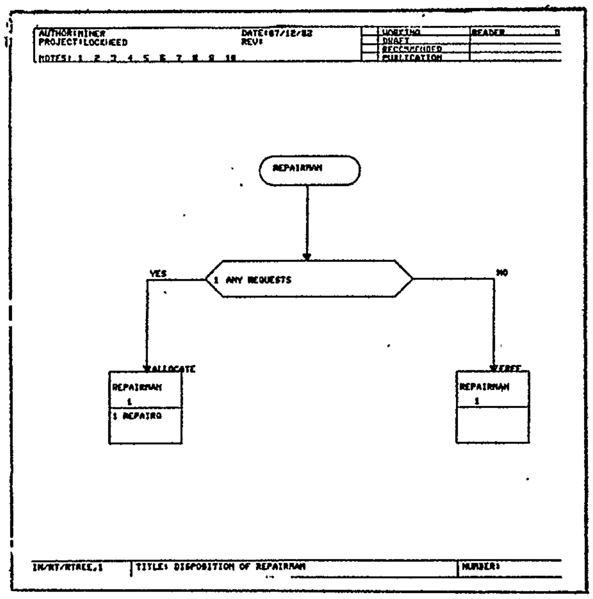


Figure 10. IDEF$_2$ Resource Disposition Tree

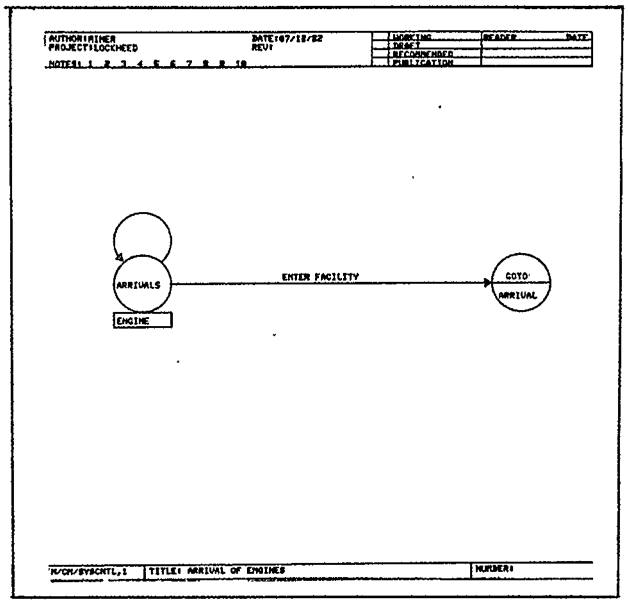
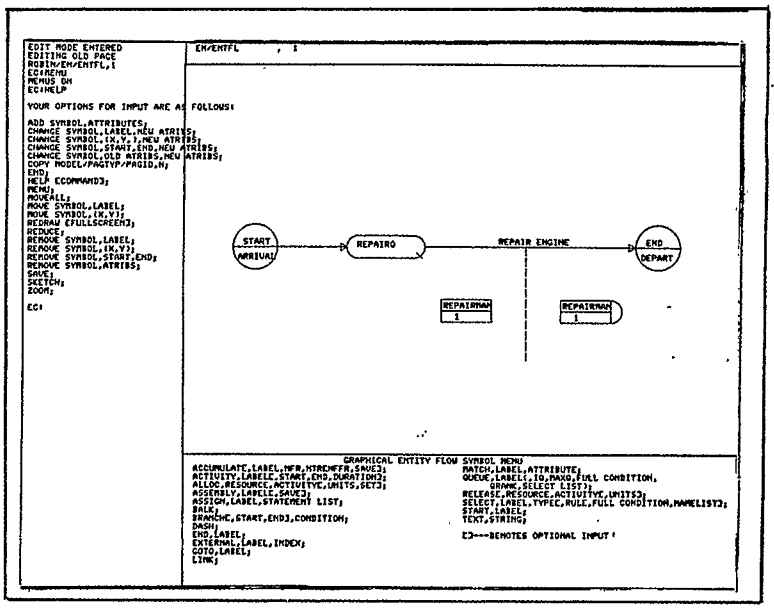


Figure 11. IDEF$_2$ System Control Network



```
EDIT MODE ENTERED
EDITING OLD PAGE
ROBIN/EN/ENTFL,1
EC:MENU
MENUS ON
EC:HELP

YOUR OPTIONS FOR INPUT ARE AS FOLLOWS:

ADD SYMBOL,ATTRIBUTES;
CHANGE SYMBOL,LABEL,NEW ATRIBS;
CHANGE SYMBOL,(X,Y,),NEW ATRIBS;
CHANGE SYMBOL,START,END,NEW ATRIBS;
CHANGE SYMBOL,OLD ATRIBS,NEW ATRIBS;
COPY MODEL/PAGTYP/PAGID,N;
END;
HELP [COMMAND];
MENU;
MOVEALL;
MOVE SYMBOL,LABEL;
MOVE SYMBOL,(X,Y);
REDRAW [FULLSCREEN];
REDUCE;
REMOVE SYMBOL,LABEL;
REMOVE SYMBOL,(X,Y);
REMOVE SYMBOL,START,END;
REMOVE SYMBOL,ATRIBS;
SAVE;
SKETCH;
ZOOM;

EC:
```

```
EN/ENTFL        , 1
```

```
                    GRAPHICAL ENTITY FLOW SYMBOL MENU
ACCUMULATE,LABEL,NFR,NTRENFFR,SAVE];     MATCH,LABEL,ATTRIBUTE;
ACTIVITY,LABEL[,START,END,DURATION];     QUEUE,LABEL[,IQ,MAXQ,FULL CONDITION,
ALLOC,RESOURCE,ACTIVITY[,UNITS,SET];          GRANK,SELECT LIST];
ASSEMBLY,LABEL[,SAVE];                   RELEASE,RESOURCE,ACTIVITY[,UNITS];
ASSIGN,LABEL,STATEMENT LIST;             SELECT,LABEL,TYPE[,RULE,FULL CONDITION,NAMELIST];
BALK;                                    START,LABEL;
BRANCH[,START,END],CONDITION;            TEXT,STRING;
DASH;
END,LABEL;                               []---DENOTES OPTIONAL INPUT'
EXTERNAL,LABEL,INDEX;
GOTO,LABEL;
LINK;
```

Figure 12. Creation of an IDEF$_2$ Entity Flow Network

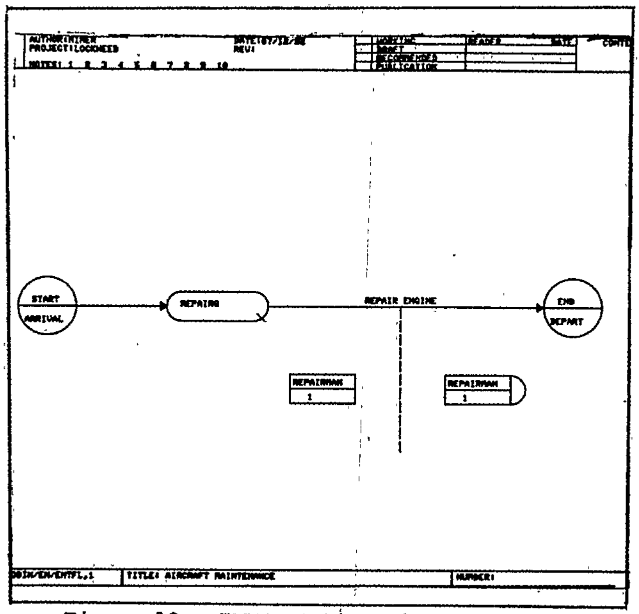


Figure 13. IDEF$_2$ Entity Line Trace

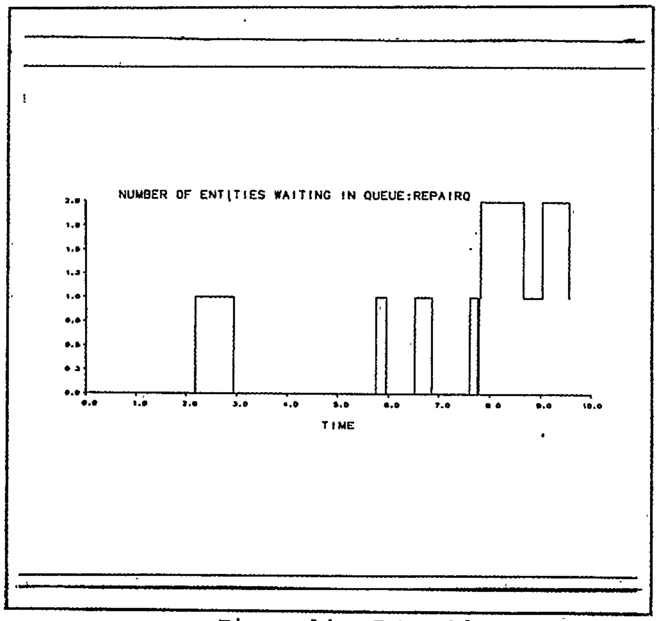


Figure 14. IDEF$_2$ Plot

Figure 14 contains an example of a plot of the number of entities in the QUEUE called REPAIRQ. The range of values displayed on the axes were defaulted for this plot, but it is possible to edit the defaulted values. For example, the Y-axis tic marks could be specified as integer values.

As has been demonstrated, the IDEF$_2$ modeling process relies heavily upon the use of graphics concepts. IDEF$_2$ models have a graphic form and are created interactively at a graphics terminal. All model information is stored in a database which facilitates the graphical editing of existing models and the graphic display of model outputs.

SOS

SNAP, (11) the Safeguards Network Analysis Procedure, was developed under contract to Sandia National Laboratories through funding provided by the U.S. Nuclear Regulatory Commission and the U.S. Navy. It was developed for the analysis of safeguards systems at nuclear facilities to provide an evaluation of their resistance to sabotage or theft of nuclear materials. Like IDEF$_2$ models, SNAP models also have a graphic form. Each SNAP model consists of three submodels: the Facility Submodel, the Guard Submodel, and the Adversary Submodel. The facility submodel defines various components of the facility and their relationships. The guard submodel describes the decision logic associated with guard movement through the facility. The adversary submodel describes the decision logic describing adversary attempts to sabotage or steal nuclear materials.

The SNAP Operating System (SOS) was developed to assist the SNAP analyst in building, maintaining, and analyzing SNAP models. The SOS includes:

* a database system in which the user may store a library of SNAP applications

* computer graphics representations of SNAP models which the user can view on the screen of a computer terminal

* a computer graphics editor to develop and modify SNAP models stored in the library

* automatic generation of data input for the SNAP computer program

* a computer graphics post-processor that displays the results of the SNAP simulation of a scenario on the screen

The SOS is an interactive command driven processor. Each major component of the SOS is an independent module that is entered and exited by invoking SOS commands. Information is shared by these components through the SOS library. Each SOS module has a set of commands associated with it to allow the user to perform the various required tasks. Overall, there are 75 user-friendly commands within the SOS. The user simply enters the desired command and is then prompted for any required responses. A HELP command is also included to provide on-line documentation of the command names and their associated syntax and purpose. A brief description of the SOS LIBRARY, EDIT, and ANALYZE modules follows.

The LIBRARY module of SOS provides a logical structure for developing and maintaining SNAP models. This function provides commands through which the user may define and link up various SNAP submodels to describe a complete SNAP model.

The EDIT module of SOS allows the user to interactively build a SNAP model using a graphics terminal. The user creates the model by commanding SOS to draw and link SNAP symbols on the screen. There are two levels of commands provided by the SOS editor. The first level allows the user to manipulate SNAP symbols one at a time. The second level has modular commands that allow the user to easily model recurring functions such as guard patrols and adversary attacks.

Figure 15 illustrates the use of the modular PATROL command to easily create a complete SNAP network representation of guards patroling a facility.

The ANALYZE module of SOS allows the user to interactively display the simulation of a SNAP model on a schematic representation of the facility. This aids not only in the presentation and analysis of SNAP results but also in the development of SNAP models by providing a convenient debugging method which expedites the review of the SNAP trace. Figure 16 illustrates the type of display produced by the ANALYZE processor.

The SNAP Operating System provides a much needed modeling and analysis capability for safeguards systems analyses. Its extensive use of graphics concepts enhances the modeling process by increasing the accuracy of the model and decreasing debugging time.

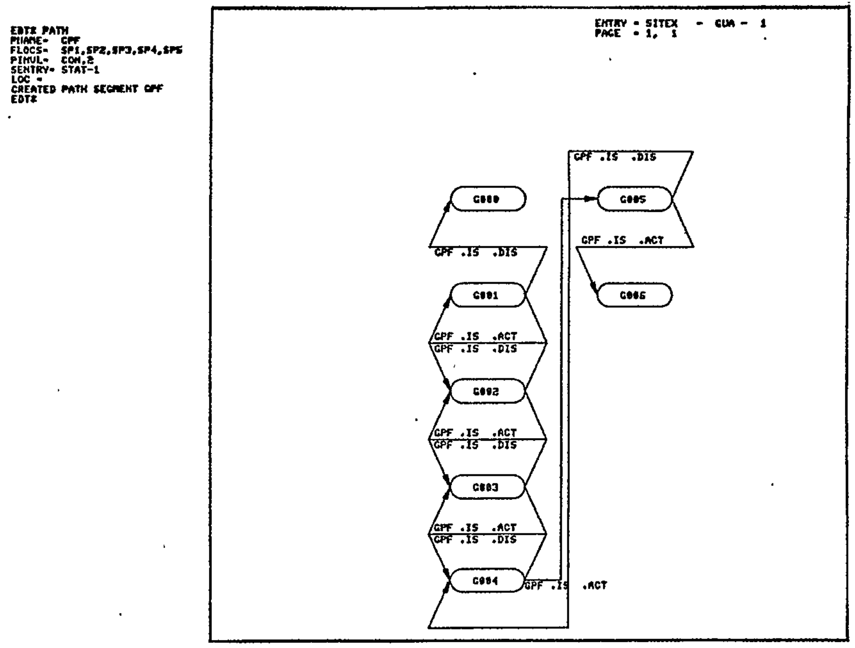
Both $IDEF_2$ and SOS use graphics extensively to support the building and analysis of simulation models of particular classes of problems. Although their graphics capabilities are not as generally applicable as general purpose graphics software or simulation specific graphics software, the $IDEF_2$ and SOS graphics capabilities greatly increase the simulation modeler's understanding of aerospace manufacturing and nuclear safeguards systems.

GRAPHIC APPLICATIONS IN PROBLEM SPECIFIC SIMULATIONS

The development of graphics to support individual simulation models can enhance the understanding of particular situations. Two recent models which relied heavily upon the use of graphics to display model output are the Aircraft Maintenance Model (6) which was developed for the Sacramento Air Logistics Center at McClellan Air Force Base, Sacramento California, and the Rose Bowl Staffing Model (13) developed for the Los Angeles County Sheriff's Department.

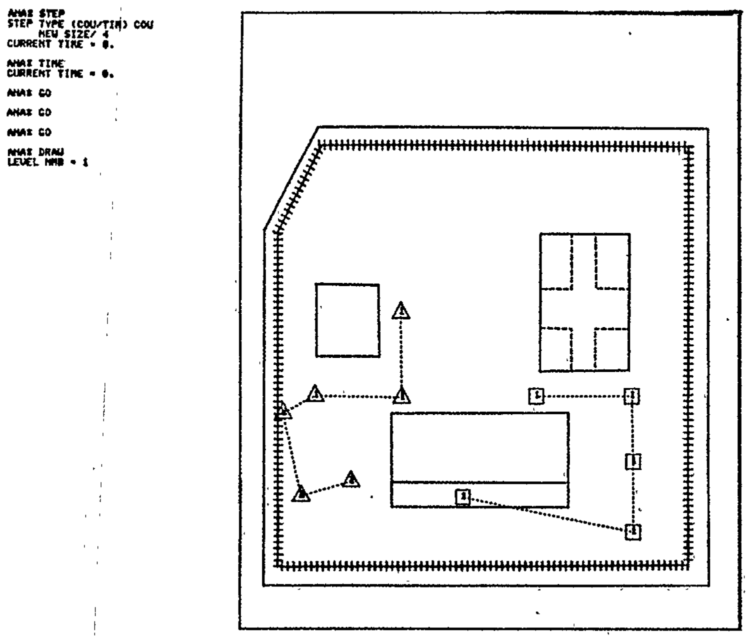
AIRCRAFT MAINTENANCE MODEL

The Aircraft Maintenance Model is a Q-GERT™(10) network simulation model which describes the flow of aircraft through a maintenance facility. The goal



```
EDT* PATH
PNAME-  GPF
FLOCS-  SP1,SP2,SP3,SP4,SP5
PIMUL-  CON,2
SENTRY- STAT-1
LOC -
CREATED PATH SEGMENT GPF
EDT*
```

Figure 15. SOS Modular Command Example



```
ANAS STEP
STEP TYPE (COU/TIM) COU
     NEW SIZE/ 4
CURRENT TIME = 0.

ANAS TIME
CURRENT TIME = 0.

ANAS GO

ANAS GO

ANAS GO

ANAS DRAW
LEVEL NMB = 1
```

Figure 16. SOS Facility Trace

of the model was to help make timely and informed decisions regarding facility operations.

The facility essentially operates like a generalized flow-shop model. To effect proper maintenance, an aircraft is sent through a series of work centers. These work centers involve flight preparation, defueling, mod disassembly, washing, mod assembly, cold testing, final selling, x-raying, painting, fueling, and flight preparation.

The model is composed of three major components. The first component is the network model, which describes all the possible combinations of paths an aircraft can take through the ALC maintenance process. The second component is the SDL™ (12) data base, which manages the data needed to run the model as well as to describe its performance. The third component of the model is a graphics capability designed to help the analyst interpret the results by making it possible for him to recognize and compare key characteristics in data patterns. Two general computer graphics capabilities are included in the Aircraft Maintenance Model. First is a capability that allows him to plot the following information stored in the SDL data base:

* Number in Facility Queue

* Facility Utilization

* Time-In-System by Aircraft type

* Ability to meet deadlines

Figure 17 contains an example of a plot of the number in the facility queue for three different facility operations.

The second computer graphics capability provided allows the user to view an individual aircraft trace through the entire facility. Figure 18 is an example of a display of this type, which shows how aircraft number 10177 is processed through the facility. The bar clock on the bottom of the display shows the simulation time that this aircraft went through each of the required operations.

ROSE BOWL STAFFING MODEL

The Rose Bowl Staffing Model was developed in 1981 for the Los Angeles County Sheriff's Department. This model supports staffing decisions for office paramedics, ambulances, and fire units for the 1982 Rose Bowl football game. Incidents such as civil disturbances, medical emergencies, fires, and bomb threats which can occur within the perimeter fence of the Rose Bowl facility were included in the model.

The Rose Bowl model was developed using the SLAM simulation language. It uses computer graphics capabilities extensively to display the results of the simulation as it progresses. Computer graphics was used to display the location of the various events and the corresponding movement of officers, paramedics, ambulances, and fire units on a two dimensional schematic of the Rose Bowl facility.

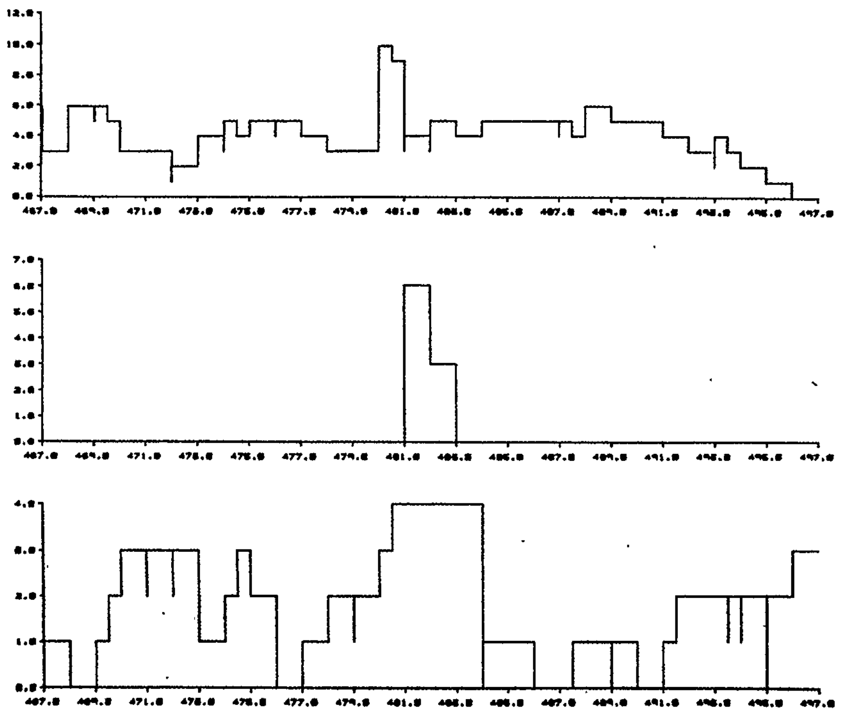


Figure 17. Aircraft Maintenance Model Plots

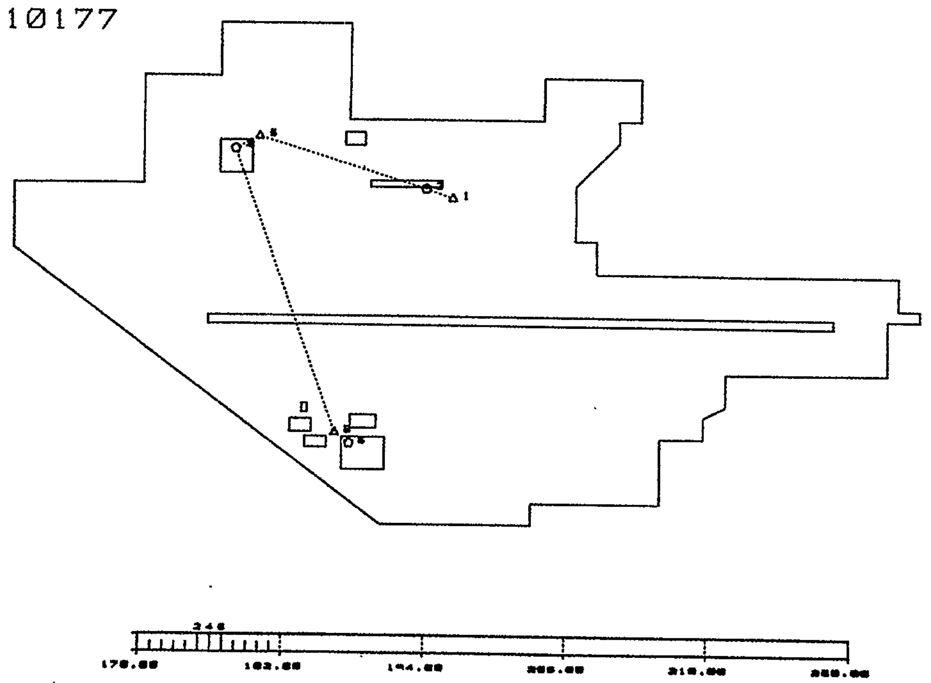


Figure 18. Aircraft Maintenance Model Trace

Shown in Figure 19 is the facility schematic displayed on the screen when the simulation model execution begins. The upper right hand portion of the screen is used to display the Rose Bowl. The left side of the screen displays the important detailed information about the on-going events. The bottom of the screen displays a menu of the options that the user may invoke during the simulation. Additionally, a bar clock that displays the current simulation time is displayed at the top of the screen.

As the simulation progresses, the various types of events occur and are displayed in the appropriate location on the Rose Bowl schematic using different symbols for the three general classes of events: civil, medical, and fire.

The movement of the responding units is illustrated on this schematic using open circles to denote the current location of units, filled-in circles to denote previous locations of units, and dashed lines to denote the units movement path. Figure 20 illustrates the resulting graphic display after the first three events have occurred.

A unique feature of this model is that the user may halt execution at any point desired by simply depressing the space bar. After simulation interruption, the user has several options:

* Resume the simulation
* Erase and redraw the display
* Invoke a stepwise mode so that the simulation interrupts automatically after each step
* Interactively add an event of any type to occur at a specific location with a specified number of responding units

Using this unique interactive simulation and graphics technique the Los Angeles County Sheriff's Office was able to effectively evaluate the Rose Bowl staffing policies prior to the game.

The graphic capabilities of both the Aircraft Maintenance Model and the Rose Bowl Staffing Model were key in helping non-technical personnel to understand and use the simulation models. These non-technical personnel were able to view the graphic outputs produced by the models and see that the simulation model performed like the real world systems. They were thus able to aid in model verification and validation.

CONCLUSION

As this paper has demonstrated, the use of computer graphics can greatly enhance the simulation modeling process. It can be used to help build accurate models quickly, to represent data in the form required by the model, to verify that the model is performing as intended, to validate with decision makers that the model represents reality and to analyze situations more quickly. These uses of graphics provide significant productivity improvements for those engineers, analysts and managers who build and use simulation models.

REFERENCES

1. Disspla Reference Manual, Integrated Software Systems Corporation (ISSCO), 4186 Sorrento Valley Blvd., San Diego, California, 92921.

2. Duket, Steven D., O'Reilly, Jean J., Hannan, Robert J., SLAM II™: Enhanced Simulation Capabilities, Pritsker & Associates, Inc., P. O. Box 2413, West Lafayette, Indiana 47906. June 1981.

3. Duket, Steven D., Hixson, Alonzo F., Rolston, Laurie , The SIMCHART™ User's Manual, Pritsker & Associates, Inc., P. O. Box 2413, West Lafayette, Indiana, 47906. June 1981.

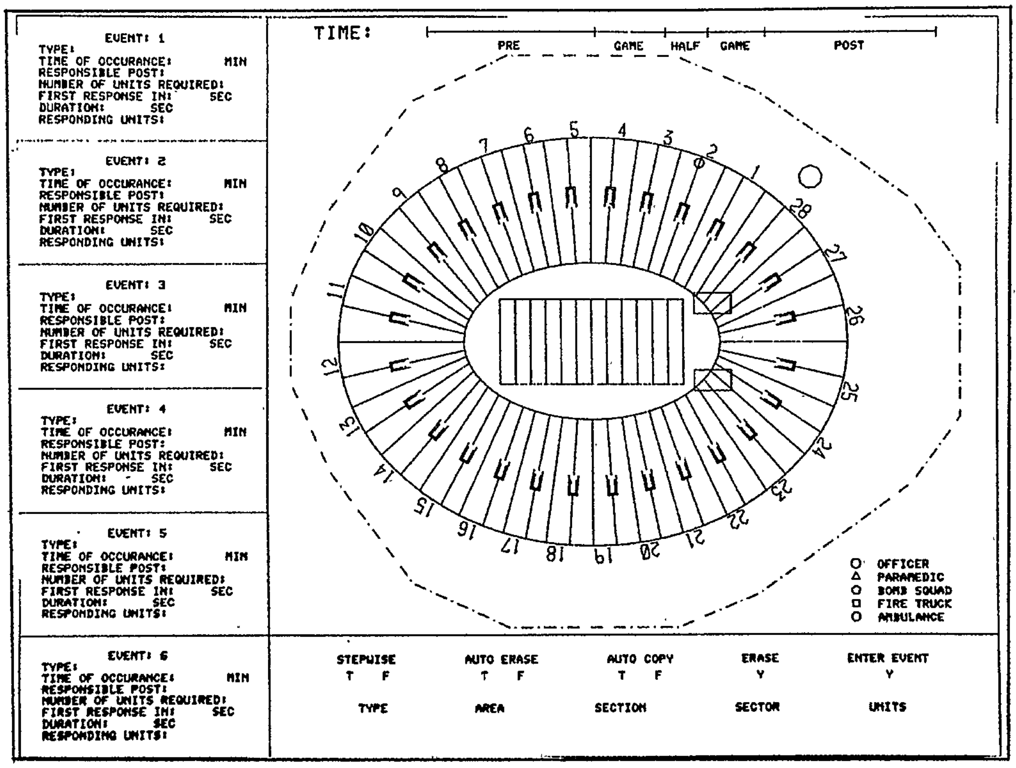
4. Graphics Compatibility System (GCS), U. S. Army Engineer Waterways Experiment Station, Automatic Data Processing Center, P. O. Box 631, Vicksburg, Mississippi. 1975 (Revised 1979).

5. Musselman, Kenneth J., Penick, William R., Grant, Mary E., AID™ Fitting Distributions to Observations: A Graphical Approach, Pritsker & Associates, Inc., P. O. Box 2413, West Lafayette, Indiana, 47906. June 1981.

6. Musselman, Kenneth J., and Hannan, Robert J., "Description and Documentation Associated with the Sacramento ALC Aircraft Maintenance Model and Supporting Program", Pritsker & Associates, Inc., P. O. Box 2413, West Lafayette, Indiana 47906. January 1981.

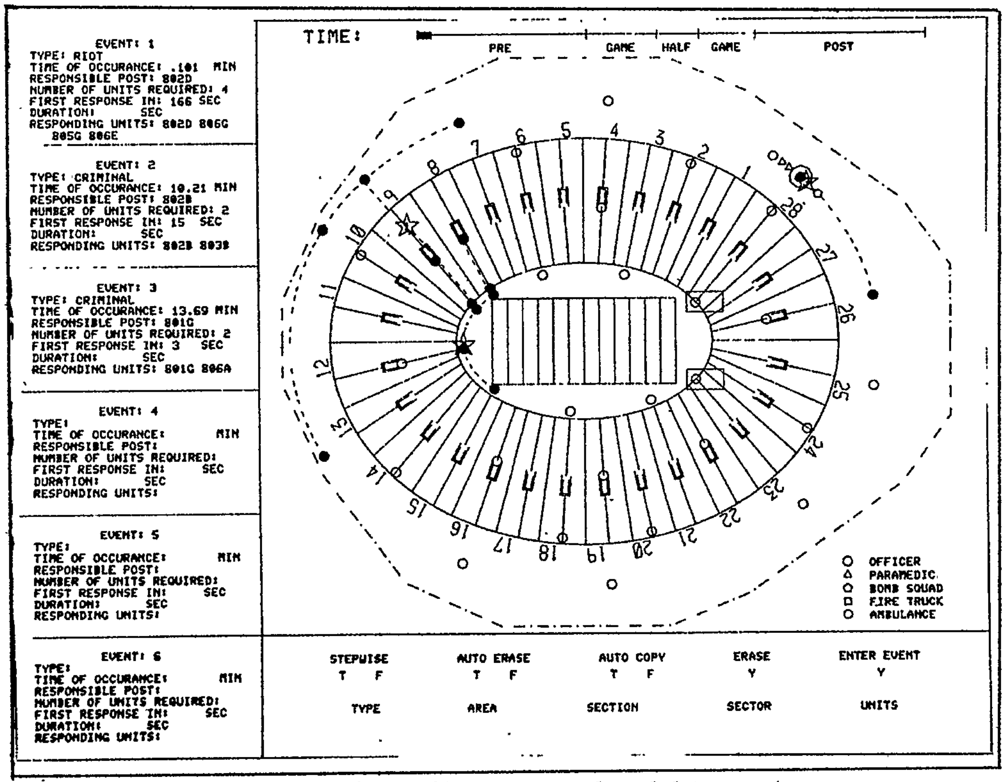
7. Pritsker A. Alan B., and Miner, Robin J., "Dynamics Modeling: The ICAM Definition Language ($IDEF_2$) Open Approach", Proceedings of the 1980 Fall Industrial Engineering Conference, Pages 159-167.

8. Pritsker, A. Alan B., Simulation and Graphics: Distinguished Engineer Lecture Series on the Practice of Engineering. April 1981.



```
EVENT: 1
TYPE:
TIME OF OCCURANCE:        MIN
RESPONSIBLE POST:
NUMBER OF UNITS REQUIRED:
FIRST RESPONSE IN:      SEC
DURATION:       SEC
RESPONDING UNITS:

EVENT: 2
TYPE:
TIME OF OCCURANCE:        MIN
RESPONSIBLE POST:
NUMBER OF UNITS REQUIRED:
FIRST RESPONSE IN:      SEC
DURATION:       SEC
RESPONDING UNITS:

EVENT: 3
TYPE:
TIME OF OCCURANCE:        MIN
RESPONSIBLE POST:
NUMBER OF UNITS REQUIRED:
FIRST RESPONSE IN:      SEC
DURATION:       SEC
RESPONDING UNITS:

EVENT: 4
TYPE:
TIME OF OCCURANCE:        MIN
RESPONSIBLE POST:
NUMBER OF UNITS REQUIRED:
FIRST RESPONSE IN:      SEC
DURATION:   -   SEC
RESPONDING UNITS:

EVENT: 5
TYPE:
TIME OF OCCURANCE:        MIN
RESPONSIBLE POST:
NUMBER OF UNITS REQUIRED:
FIRST RESPONSE IN:      SEC
DURATION:       SEC
RESPONDING UNITS:

EVENT: 6
TYPE:
TIME OF OCCURANCE:        MIN
RESPONSIBLE POST:
NUMBER OF UNITS REQUIRED:
FIRST RESPONSE IN:      SEC
DURATION:       SEC
RESPONDING UNITS:
```

Figure 19. Initial Rose Bowl Facility Schematic



```
EVENT: 1
TYPE: RIOT
TIME OF OCCURANCE: .101  MIN
RESPONSIBLE POST: 802D
NUMBER OF UNITS REQUIRED: 4
FIRST RESPONSE IN: 166 SEC
DURATION:       SEC
RESPONDING UNITS: 802D 806G
   805G 806E

EVENT: 2
TYPE: CRIMINAL
TIME OF OCCURANCE: 10.21 MIN
RESPONSIBLE POST: 802B
NUMBER OF UNITS REQUIRED: 2
FIRST RESPONSE IN: 15  SEC
DURATION:       SEC
RESPONDING UNITS: 802B 803B

EVENT: 3
TYPE: CRIMINAL
TIME OF OCCURANCE: 13.69 MIN
RESPONSIBLE POST: 801G
NUMBER OF UNITS REQUIRED: 2
FIRST RESPONSE IN: 3   SEC
DURATION:       SEC
RESPONDING UNITS: 801G 806A

EVENT: 4
TYPE:
TIME OF OCCURANCE:        MIN
RESPONSIBLE POST:
NUMBER OF UNITS REQUIRED:
FIRST RESPONSE IN:      SEC
DURATION:       SEC
RESPONDING UNITS:

EVENT: 5
TYPE:
TIME OF OCCURANCE:        MIN
RESPONSIBLE POST:
NUMBER OF UNITS REQUIRED:
FIRST RESPONSE IN:      SEC
DURATION:       SEC
RESPONDING UNITS:

EVENT: 6
TYPE:
TIME OF OCCURANCE:        MIN
RESPONSIBLE POST:
NUMBER OF UNITS REQUIRED:
FIRST RESPONSE IN:      SEC
DURATION:       SEC
RESPONDING UNITS:
```

Figure 20. Rose Bowl Model Execution

9. Pritsker, A. Alan B., and Pegden, Claude Dennis, Introduction to Simulation and Slam , Systems Publishing Corporation 1979.

10. Pritsker, A. Alan B., Modeling and Analysis Using Q-GERT Networks, Halsted Press, A Division of John Wiley & Sons, Inc., New York, 1977.

11. Sabuda, J., Polito J., Walker, J., and Grant III, F.H.,The SNAP Operating System Reference Manual, Pritsker & Associates, Inc., P. O. Box 2413, West Lafayette, Indiana, 47906. October 1981.

12. Standridge, Charles R., SDL: Simulation Data Language Reference Manual, Pritsker & Associates, Inc., P. O. Box 2413, West Lafayette, Indiana, 47906. June 1981.

13. Starks, D., Parmelee W., and Erdbruegger, D. "SLAM II Model of the Rose Bowl Staffing Plan", Pritsker & Associates, Inc., P. O. Box 2413, West Lafayette, Indiana, 47906. December 1981.

14. Template Reference Manual, Megatek Corporation, 3921 Sorrento Valley Blvd., San Diego, California, 92121. 1981.

15. Terminal Independent Graphics System (TIGS) Version 1.0 Reference Manual, St. Paul, Minnesota. June 1978.

16. Yancey, David P., and Miner, Robin J.,"The Use of $IDEF_2$ for Analyzing Material Handling Problems", Proceedings of the 1981 Fall Industrial Engineering Conference, Pages 396-403.