Proceedings of the 1986 Winter Simulation Conference
J. Wilson, J. Henriksen, S. Roberts (eds.)

PCModel AND PCModel/GAF -- SCREEN ORIENTED MODELING

David A. White
Simulation Software Systems
San Jose, CA 95121

ABSTRACT

The recent availability of relatively inexpensive personal computers has created a market for personal engineering softwre tools. One such tool designed for animating as well simulating manufacturing lines and process, called **PCModel**, is reviewed in this paper. **PCModel** takes a unique approach to manufacturing simulation by providing you with an easily understood set of instructions allowing you to animate your process on a cost effective personal computer. Through animation, you, your co-workers, and management, can quickly see that the model represents the process being studied, and immediately observe the effect of parameter changes made from the keyboard.

1. INTRODUCTION

PCModel[tm] and **PCModel/GAF**[tm] are two variants of a highly interactive and visual modeling and simulation system. The system includes a unique modeling language, both graphic and text editors, and a very user friendly runtime environment. The system is specifically designed for use on personal computers without need for custom hardware.

PCModel/GAF (Graphic Animation Facility) differs from **PCModel** in that it supports life-like high-resolution graphics in lieu of character graphic images.

2. THE OVERLAY

The most distinguishing feature of this approach to modeling is the concept of mapping the process, such as a manufacturing assembly line, into a graphic schematic representation. This mapping onto a column-row coordinate grid allows for later referencing of process features through use of row-column (or x and y) coordinates. Key coordinates of the location of entry points, work cells, queues, and exit points are symbolically referenced by assigning an eight character name to the feature. The process schematic, called the *overlay*, is easily constructed in color using the integrated graphic editor, or with popular engineering drawing software such as *CADwrite*[tm] or *AutoCAD*[tm].

3. OBJECT MOVEMENT

The overlay contains the static elements of the model. The dynamics are created by building a text file of **PCModel** language statements that direct the creation, motion, and removal of modeled objects. An *object* is represented by either an alpha-numeric character or a graphic icon. These objects are made to move about the overlay image through simple, yet powerful, instructions that indicate direction and speed of motion.

A salient feature of the modeling system is that *only one object at a time may occupy an overlay location.* Contention for shared resources, queues, or paths between work cells is automatically sensed by the underlying system. The modeler does not have to manage the details of contention!

4. INSTRUCTIONS

However, special instructions are available to test and make logical decisions based upon the presence of objects at any specified overlay location. Decisions can also be made based upon numeric values, elapsed time, or other parameters reflecting the state of the simulation.

The above instructions, involving object movement and logical decision making, are formed into a sequence of statements that describe one or more *routes* through the process. As the model runs, the objects will be seen to move about the overlay. The impression is that you are looking down from the rafters and viewing the motion of materials through your process.

5. INTERACTION

The model, however, will be simulating many times faster than the real process. You can slow it down, freeze the action, or single-step it to observe fleeting events. The overlay image can be panned, and with **PCModel/GAF** it can be zoomed to view more detail, or to see the whole process at once.

At any time during the modeling session, you can change model parameters or object attributes. Several interactive screens are available to either view or modify these values. "What-if?" analysis can be performed directly. One typical situation would be to interactively change the speed

parameter of a transporter to determine its operational limits. It takes less than 5 seconds to stop the simulation, change the parameter, and begin observing the effect of the change.

6. INPUT DATA

Simulation of many different cases, each case involving many different sets of control values, is usually required. **PCModel** minimizes this task through support of user generated files for initializing model parameters. The data can be read just prior to starting the model, or as the model runs.

7. OUTPUT DATA

Although graphically viewing the dynamics of a simulation leads to significant insight and understanding, model data for reporting or statistical analysis may be also be needed. The system allows you to write any model parameter either to the display, or to a disk file. You can include custom formatting and messages. Time-stamping and periodic writing of data is a simple process. In fact, a single key stroke will save away copies of any displayed screen of model values, or even the character-graphic screen itself. In addition, there is automatic generation of utilization data that can be later processed and graphed using Lotus 1-2-3.

8. RANDOM NUMBERS

More advanced modelers will appreciate the built-in random number generators providing uniform, normal, and exponential functions. You can also define your own custom discrete and continuous distributions.

9. CREDIBILITY

The visual basis of the system lends tremendous credibility to your modeling effort. Programming errors and bugs are very apparent simply by viewing the simulation. The interactive nature of the system further lends itself to verifying the model under extremes of conditions. A unique trace feature enables you to follow, line-by-line, the statement executions associated with one or more modeled objects. Thus you can always associate the animated action of an object with the specific file statement directing the action.

10. APPLICATIONS

This screen-oriented approach to modeling has successfully simulated not only manufacturing assembly processes, but communication networks, stacker-crane retrieval systems, Automated Guided Vehicle traffic, and even ocean-going cargo container operations. Some applications have involved over 100 work stations and several hundred active objects.

11. USER IMPRESSIONS

Building models with this system can be a very enjoyable and rewarding activity. Users of the system report that the high degree of interactivity and quick turn-around time for model changes keeps them enthused and motivated. They maintain a sense of creativity. The ability to share their findings, visually, with associates and management, has made the simulation task more rewarding. Validation of the model by others, and communicating results to management is found to be much easier through direct observation of the running simulation.

12. SUMMARY

The character-graphic version of **PCModel** has been commercially available for about two year. There are currently in excess of 400 locations using the system. **PCModel/GAF**, the engineering quality high-resolution graphics version, has only recently become available. Its additional realism, including the ability to move objects in three dimensions, and along graphically defined *paths*, makes it a preferred modeling and simulation facility.

FIGURES

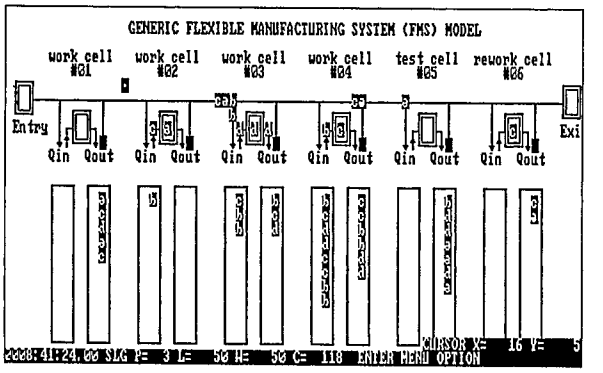

Figure 1: Overlay Image Screen

This is a typical application's overlay image. Although only 80 columns by 24 rows of data are displayed, you can work with a *logical* screen of over 32,000 column-row character cells.

If you are using a color monitor, you can use the built-in color attribute Overlay Editor to assign color to selected areas of the image.

The line and box graphics are created using the *box-drawing* special set of characters. Schematic representations of equipment and work areas are easily constructed with the aid of the Overlay Editor.

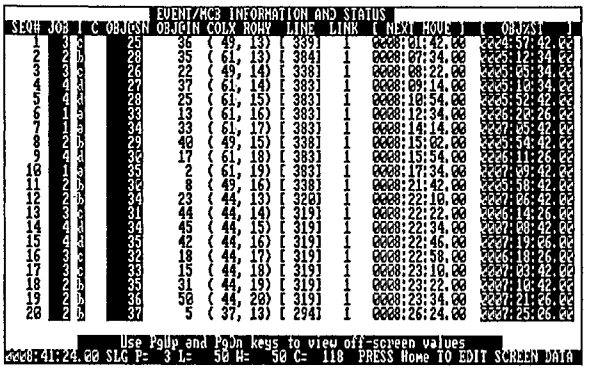

Figure 2: Event List Screen

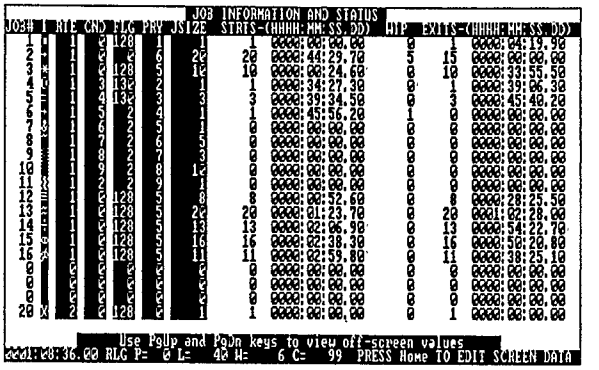

Figure 3: Jobs Screen

This screen provides information about the status of each individual object in the model. It is a display of most of the information **PCModel** uses to track and move each object.

You can use the information to analyze the flow of individual objects and as an aid in debugging complex models.

Also, all data in reverse image fields can be changed from the keyboard. Thus you can single out an object by changing its ID to a unique character. It then becomes quite easy to monitor its flow through the model.

The paging keys can be used to display entries not currently on the screen.

The entries are ordered by time of next scheduled move (event). If an entry's next move time is less than the current clock, it means that the object is blocked. The reversed character field at the far left of the screen (not visible in the figure above) is used to indicate the split between entries that are blocked and entries waiting for their time delays to elapse.

The *FIRST-MOVE* field holds the clock time of when the object started into the simulation. This is a user changeable parameter with the system name **OBJ%ST.**

The *ROUTE* column indicates the relative offset into the internal routing code for the current instruction controlling the movement of the object. It can be of help in debugging you application. The *LINK* column shows the current nesting level of any calls to Links.

This screen is used to display Job parameters as defined in your application routing file. If more than 20 jobs are defined, you can use the **PgDn** and **PgUp** keys to view them. All data shown in reverse image fields can be changed from the keyboard.

The screen also displays for each job the number of objects released into the active model, the count of objects still in the active model (**WIP**), and the count of objects completed. It also displays the simulated time of the release of the first objects from each job, as well as the time the last object was completed.

1	set Monochrome display mode
2	set Color display mode
3	set Black & White display mode
A	display/edit Arrays
D	display Description screen
E	display/edit Event list data screen
F	save current screen image to disk File
G	Go execute model
H	display Help menu
I	Initialize model for rerun
J	display/edit Job release and status screen
L	Load application file of routings
N	Name and Open a file to save utilization statistics
O	use the Overlay Editor
P	display/edit object Parameters screen
Q	Quit the modeling session
R	Restore saved simulation session from disk file
S	Save current simulation session to disk file
T	use the Text Editor
*	set trace mode (toggle)
U	display Utilization statistics data screen
V	display/edit variable symbol Values

Space-Bar show run-time screen
Ctrl-Break interrupt execution

Figure 4: PCModel Help Menu Commands

F1	Help Menu
F2	Look-Ahead/Increment (toggle)
F3	Increase Pace Delay (slow down the model)
F4	Decrease Pace Delay (speed up the model)
F5	Increase Work-In-Process limit
F6	Decrease Work-In-Process limit
F7	Deposit/Pick-up Blocking character
F8	Ten-hour Halt/Go (toggle)
F9	Single Step/Run (toggle)
F10	Quit (Prompt to return to DOS)

Figure 5: PCModel Help Menu Functions

C=(precision)	Clock precision
D= ...25 lines of text..$	Description
I=(=|*|filespec)	Include MIF file
J=(jb,id,rte,cond,flg,pr,sz)	Jobs
L=(ii)	wipLimit (Initial WIP)
N=(lines)	Number of lines
O=(=|*|filespec)	Overlay
R=(xx,yy)	Relative reference
S=(ii)	Symbols (allocate space)
U=(id,c..c,*)	Utilization Statistics
V=(xx,yy)	Viewing Window loc'n

Figure 6: PCModel Load-Time Directives

AO(vv,+|-|*|/,nn)	Arithmetic Operation
BL(!c..c[,xy,tt])	Begin Link
BR(route,xy,tt)	Begin Route
CF(DAT|RPT)	Close File
CL(xy)	CLear route location
DV(vv)	Decrement Value
EL	End Link
ER	End Routing
GD(value[,...,value])	Get Data
GV(value)	Get Value
IF(nn,cond,nn,then,else)	IF (true jmp1 else jmp2)
IV(vv)	Increment Value
JB([A,]xy[,..,xy],:c..c)	Jump Blocked (Abs)
JB(R,(+|-xx,+|-yy)[,..,(+|-xx,+|-yy)])	Jump Blocked (Relative)
JC([A,]xy[,..,xy],:c..c)	Jump Clear (Abs)
JC(R,(+|-xx,+|-yy)[,..,(+|-xx,+|-yy)])	Jump Clear (Relative)
JP(:c..c)	JumP to routing label
LK(!c..c)	LinK to link label
MA(xy,tt)	Move Absolute
MD(vv,tt)	Move Down
ML(vv,tt)	Move Left
MR(vv,tt)	Move Right
MU(vv,tt)	Move Up
OF(filespec.DAT|filespec.RPT)	Open File
PM(F|xy,msg[",])	Print Message
PO(xy)	POst location
PV(F|xy,vv[,])	Print Value
QX	Quit eXecuting
RM([vv,]+xx|-xx,+yy|-yy,tt)	Relative Move
RS(nn)	Random Seed
RV(@rand, lolimit, hilimit)	Random Value (uniform)
RV(U,rand,mean, variance)	Random Value (Unif)
RV(N,rand,mean, deviation)	Random Value (Normal)
RV(E,rand,mean)	Random Value (Expon)
RV(C,rand,table)	Random Value (Cont)
RV(D,rand,table)	Random Value (Discrete)
SA(bg,fg)	Set color Attribute
SE	Simulate Event
SV(vv,nn)	Set Value of symbol
TP([A,]xy[,...,xy])	Test Position (Absolute)
TP(R,(+|-xx,+|-yy)[,..,(+|-xx,+|-yy)])	Test Position (Relative)
VW(xy)	Viewing Window control
WA	Wait Advance
WC(%)	Wait Clock
WE	Wait Event
WK	Wait for Keyboard entry
WT(tt)	Wait Time

Figure 7: PCModel Run-Time Instructions

REFERENCES

Chung, Y. G. (1986). *An Animated Simulation Model for a Transtainer-Based Container Handling Facility.* Masters Thesis, Oregon State University, Corvallis, Oregon.

Grant, J. W. and Weiner, S. A. (August, 1986). Factors To Consider In Choosing A Graphically Animated Simulation System. *Industrial Engineering*, Vol 18-8, 36-40.

Wang, S. T. (1985). Animated Graphic Simulation of an Automatic Guided Vehicle System (AGVS). In: *Proceedings of the 1985 Winter Simulatin Conference* (D. T. Gantz, G. C. Blais, and S. L. Soloman, eds.). Institute of Electrical and Electronics Engineers, San Francisco, California.

White, D. A. (1986). *PCModel, Personal Computer Character Graphic Modeling System.* Simulation Software Systems, San Jose, California.

White, D. A. (September, 1983). Single-Unit Processing is Solved by Flexible Manufacturing. *Electronic Packaging & Production*, Vol 23-9,102-106.

AUTHOR'S BIOGRAPHY

DAVID A. WHITE is president and founder of Simulation Software Systems. He received his B.S. in Electrical Engineering from San Jose State University in 1974. Prior to founding his company, he worked as an Advisory Manufacturing Engineer for IBM's General Products Division. In that capacity he had technical responsibility for establishing manufacturing and test processes for that company's strategic products. His current activities include enhancement of personal computer based modeling and simulation software and techniques. He is a member of SCS and IIE.

David A. White
Simulation Software Systems
2470 Lone Oak Drive
San Jose, CA 95121, U.S.A
(408)270-2300