Proceedings of the 1989 Winter Simulation Conference
E.A. MacNair, K.J. Musselman, P. Heidelberger (eds.)

ANIMATING SIMULATIONS IN RESQME

Anil Aggarwal
Kurtiss J. Gordon
James F. Kurose
Dept. of Computer and Information Science
University of Massachusetts
Amherst, Mass. 01003

Robert F. Gordon
Edward A. MacNair

IBM Thomas J. Watson Research Center
Yorktown Heights, New York 10598

ABSTRACT

The RESearch Queueing Package Modeling Environment, RESQME, is a graphical environment for creating and modifying queueing models and for evaluating and analyzing the results. An animation facility is a natural adjunct to the graphics-oriented capabilities of this system. In such an environment, animation provides a powerful tool for effectively visualizing massive amounts of information. This makes it easier for the modeler to relate events in the model with processing in the real-world system. Animation enhances the modeler's ability to debug the model, to understand the model interactions and the impact of model changes, and to communicate the results to others. In this paper, we describe the animation facility in RESQME.

1. INTRODUCTION

The RESearch Queueing Package Modeling Environment, RESQME (Gordon et al. 1986, 1987, 1988, Kurose et al. 1986), is a graphical environment for creating and modifying queueing models and for evaluating and analyzing the results. An animation facility is a natural adjunct to the graphics-oriented capabilities of this system. In such an environment, animation of the simulation provides a powerful tool for visualizing massive amounts of data, thus reducing the modeler's need to translate from the syntax and semantics of the computer software to that of the real-world system. Models may be more easily debugged due to the modeler's enhanced ability to visually trace the movement of jobs and resource allocation in the model. By observing the evolution of the system, the modeler may also obtain a qualitative understanding of complex phenomena, which then complements the quantitative performance results provided by most software. The ability to observe the evolution of the system also offers the possibility of observing and discovering, "in the lab," phenomena which otherwise would have been "washed out" in the average, steady-state statistics. Finally, animation provides an enhanced ability to communicate the model design and the model results to others. To paraphrase: if a picture is worth a thousand words, a moving picture is most certainly worth a million words.

Animation capabilities have been present in graphics-oriented simulation tools for some time now (e.g., Melamed and Morris 1985, Conway et al. 1987, O'Keefe 1987, Standridge Pritsker and Stein 1987, Cox 1987, Miles, Sadowski and Werner 1988, Russel 1988). The particular design decisions made in developing the animation component of RESQME have been influenced by several of these earlier efforts, shaped by the semantics of the RESQ language (Sauer, MacNair and Salza 1980), and guided by RESQME's underlying design principle that a single graphical representation of the model should be the sole interface between the modeler and the modeling tool. As we will see, in this latter respect, our approach towards animation differs from that of CINEMA (Kilgore and Healy 1987, Miles Sadowsky and Werner 1988), TESS (Standridge, Pritsker and Stein 1987), and GPSS/PC (Cox 1987) in that the graphical model definition itself is used as the basis for the animation. Once a model has been constructed in RESQME, it can be immediately animated without any further effort on the part of the modeler - there is no distinction between the graphical model description and the pictorial model representation used during animation. In contrast, a mechanism such as a facility builder, position window, or static/dynamic animation layout are used in other tools to couple the graphical rendering used for animation to the different (but often graphical) model definition. The advantage of maintaining a single graphical representation of the model is the ability to move through all phases of the modeling lifecycle (from model creation, to evaluation, to output analysis, to animation) through a consistent model interface. A disadvantage, however, is that during animation the graphical model diagram is visually more primitive that the impressive 3-D renderings that can be created using the sophisticated sketching capabilities provided by some of the other animation packages.

An approach towards animation which insists upon a single, consistent graphical model representation has also been adopted in XCELL+ (Conway et al. 1987) and PAW (Melamed and Morris 1985) and our work has been influenced by these earlier efforts (particularly the latter). Our present effort differs from these efforts in several ways, but primarily in the explicit support we provide for animating large, hierarchically-structured models,

Some of the basic capabilities provided by the RESQME animation facility that will be discussed in this paper include:

- the animation of job and token movement in a hierarchical model.
- the ability to modify animation variables, such as speed, color and size of the animated objects or use of event versus linear time.
- the ability to interrupt the animation to modify the model diagram display (move, pan, zoom, layer) or to change animation options.
- selective animation of subsets of nodes, queues, or chains.
- the ability to trace an individual job as it moves through the model and its submodels.
- the ability to (re)start an animation at any point, step through the simulation based on checkpoints, and to stop an animation at breakpoints.

Animation in RESQME consists of displaying the movement of the jobs and tokens through the network. As the jobs and tokens move, the queue length and token availability at the nodes and queues are updated and displayed textually and graphically. Elements display the simulation time and event count and identify the job currently being moved.

Animation is currently accomplished in RESQME in an after-the-fact manner. This is a consequence of the fact that RESQME is run on a workstation connected to a mainframe computer. All user interaction, graphical specification and display, is done on the workstation, while the computation-intensive simulation is run on the mainframe. This cooperative processing environment is relatively transparent to the user, in that files and commands are automatically transmitted between the workstation and the mainframe.

To provide animation in this environment, we create a trace file, containing a partial event history of departure and arrival events. This file is created on the mainframe during the execution of the simulation and is then transferred to the workstation. The innermost "loop" of the animation component of RESQME simply takes an event (such as, the departure of a job from one node in the model and its subsequent arrival at another node) from the trace file and animates that event on the graphical network. As a result, a given simulation can be animated numerous times without actually re-simulating a model. An advantage of this approach is that the simulation can be replayed with the same timing and sequencing of events. Furthermore, portions of the simulation can be skipped over, while others can be examined in slow motion. A disadvantage is that the modeler cannot interact directly with an on-going simulation via the animation facility; we expect to provide this capability in the future when we can execute simulations directly on the workstation.

We will demonstrate the use of the animation facility in this paper through an example. The example is a hierarchical model of an interactive computer system. The main model consists of a fixed number of jobs (25) initially at the terminals, which go in turn to one of two host computers for processing and then return to the terminals for further input before being sent to one of the host computers again for further processing. The host computer system is modeled as a submodel consisting of page frames of memory that must be held while a job cycles a given number of times through the processor-sharing cpu and through either a hard disk or a floppy disk. The number of page frames and processing times for the cpu and the disk drives are parameters of the submodel whose values are independently provided by each of the two invocations. Figure 1 shows the main model level with the terminals and the two invocations of the submodel.

2. INITIATING ANIMATION

When setting up the model for simulation, all that needs to be added to get animation is to ask that a trace file be created. To do this, the modeler selects the trace menu item and specifies in a pop-up window the period of simulated activity for which animation is desired. This period can be specified in terms of starting and stopping simulation time or events. We specified to start the trace at event 0 and stop it after event 1000.

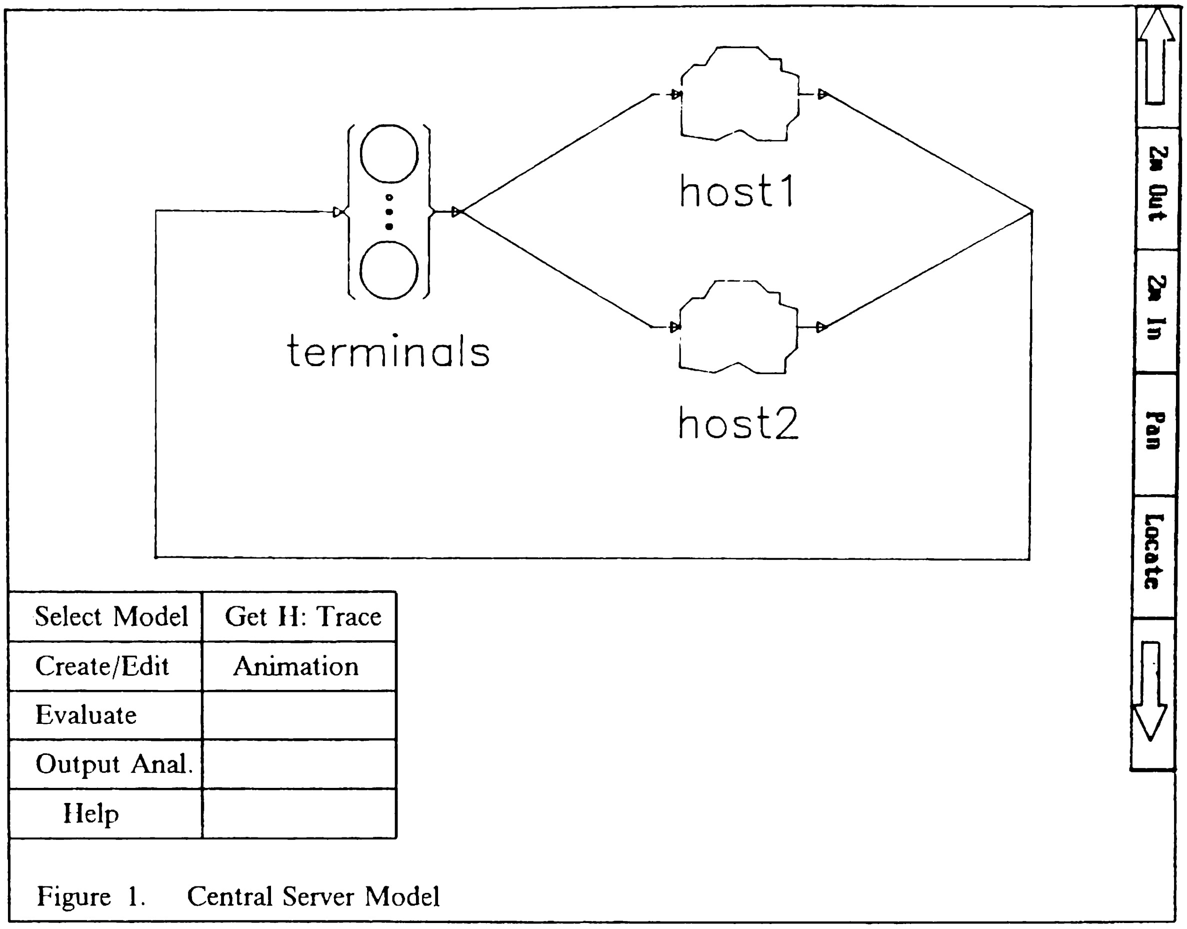


Figure 1. Central Server Model

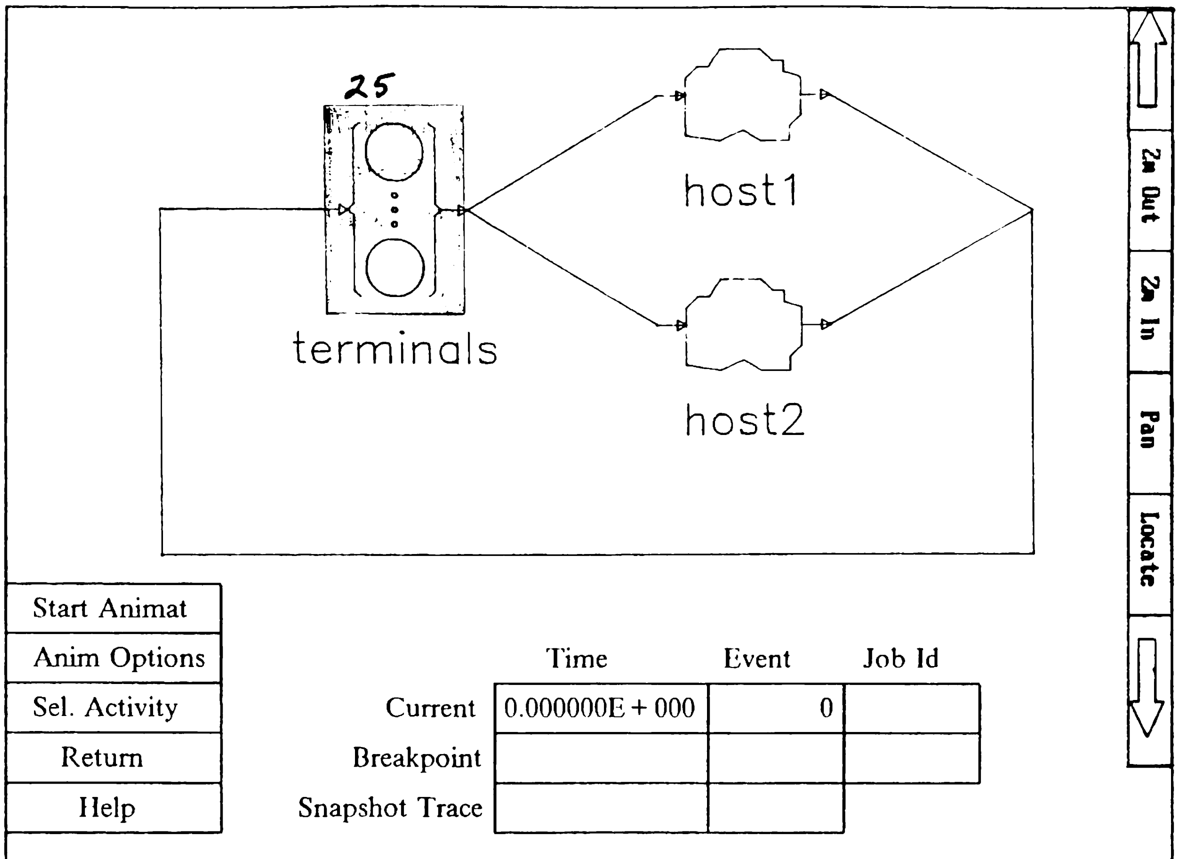
During the simulation of the model on the mainframe, the trace file is created and when the simulation run is completed, the trace file is downloaded to the workstation. At that point, the modeler can view the animation. The modeler selects the animation menu item to enter the animation facility, as shown in Figure 1. In order to be able to skip forward and backward in the animation a checkpoint file is created that contains snapshots of the system states. A pointer file is also produced that links the snapshots to the appropriate record in the trace file. This method provides an efficient and flexible way of processing what is typically a large trace file. We can then quickly restore the state using the snapshot information in the checkpoint file, and then through the use of the pointer file, we can begin animating events until the desired simulation time has been reached.

Figure 2 shows the resulting RESQME screen layout when the animation facility is entered.

Consistent with other tasks in RESQME, the model diagram is displayed in the main modeling area with the screen management menu on the right border. The model diagram has the initial queue lengths and token pool values displayed above the appropriate nodes, and the nodes are also shaded to reflect their initial queue lengths.

The screen management menu is used throughout all tasks in RESQME to modify the view of the model by zooming, panning, centering on a node, or layering to a submodel view. Its functions are made available within the animation facility as well, since RESQME determines the job and token movements from the endpoint node names of each traversed arc and not from its graphical position. This late binding allows the modeler to change the positions of the nodes and arcs during animation, and RESQME will animate the jobs and tokens along the revised paths.

Furthermore, if the Layer Down menu item were selected from the screen management menu, and for example, the user pointed to the host1 invocation, the animation within that invocation would be displayed in the modeling area, as shown in Figure 3.



	Time	Event	Job Id
Current	0.000000E + 000	0	
Breakpoint			
Snapshot Trace			

Figure 2. Initial Animation Display

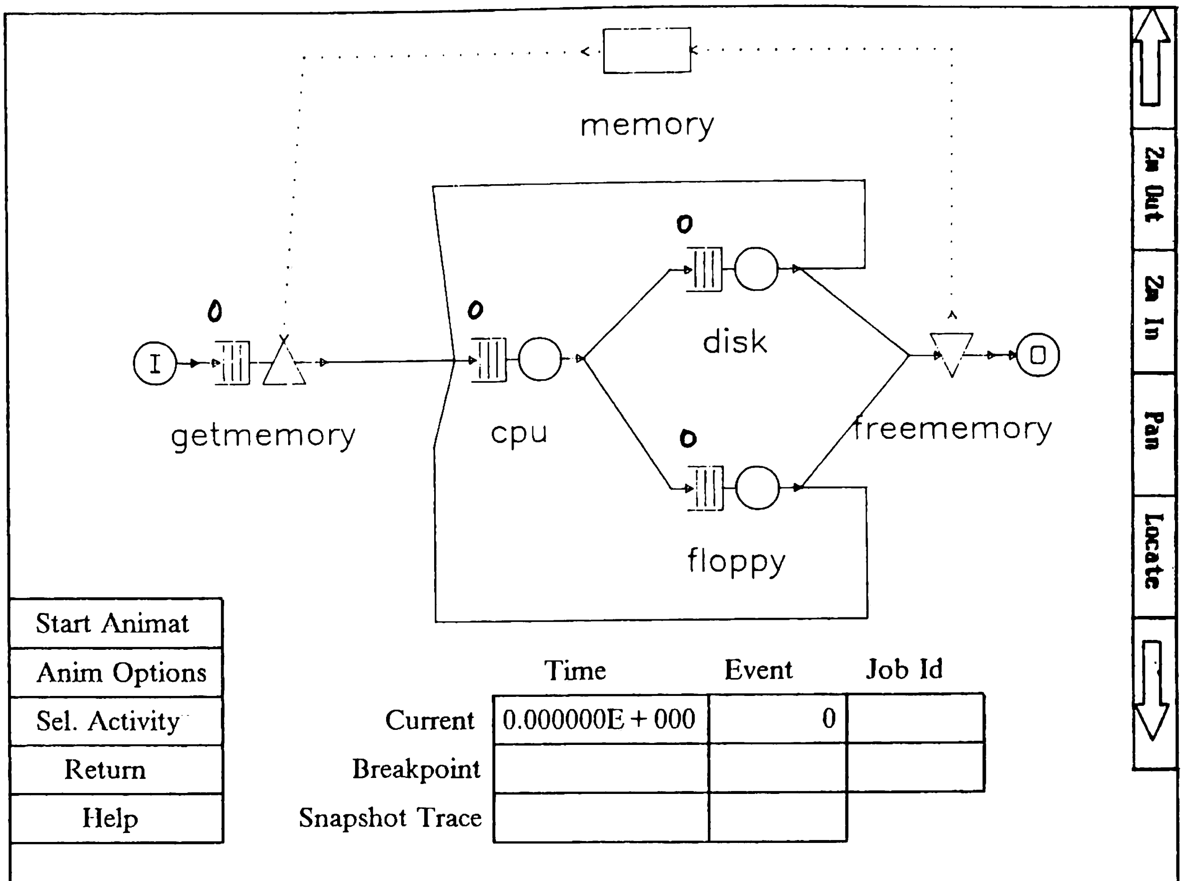
3. ANIMATION DETAILS

The main animation menu and the animation control panel are displayed in the bottom quarter of the screen. The main animation menu, located in the lower left-hand corner of the screen, contains the commands to start the animation, to change the animation options, to select specific nodes to animate, to return to the other tasks of RESQME, and to request context-sensitive help. The animation control panel is a 3-by-3 matrix of elements. The control panel is used to both display and to set time, event, and job id information for the animation.

Selecting the menu item, Start Animation, will start or restart the animation of the displayed model beginning at the current time and current event number displayed in the animation control panel. The animation will continue until either a breakpoint is reached, the end of the trace file is reached, or the modeler interrupts the animation. When the animation is stopped, the modeler can select a command from any one of the menus or change a value in one of the control panel elements. For example, the modeler can zoom in on the model by selecting the appropriate item from the screen management menu, set the time to 2 in the control panel, and then select the start animation menu item. He will see the animation of the jobs starting from time 2 on the zoom-in view of the model diagram.

A user profile contains default values for animation, such as the desired speed, size and color of the ball to represent jobs or the rectangle to represent tokens. These options can be changed between animation sequences by selecting the menu item, Animation Options. A pop-up window displays the underlying animation options and allows the user to select changes to these attributes. This follows the same interaction as in the other tasks of RESQME, such as the interaction to modify the attributes of a queue or the attributes of a chart.

In addition to being able to change the characteristics of the job and token animation, the modeler can choose timing and selective tracing properties. In particular, he can choose to display the animation in either "event time" or "linear time". If event time is selected, the events will be animated one after the other with a fixed pause time between each animated event. In this



	Time	Event	Job Id
Current	0.000000E+000	0	
Breakpoint			
Snapshot Trace			

Figure 3. Animation at the Submodel Level

mode, the animated time between events does not reflect the simulated time between those events.

If linear time is selected, an event will be animated and then the simulation clock will be incremented by a fixed amount (option set by the modeler). If no event occurred between the newly updated simulated time and the previous simulated time, the simulated time is again updated in the same manner. This continues until the simulated time reaches the point at which an event occurred, and that event is then animated. In this way, the amount of time between the display of two animated events is approximately proportional to the amount of simulated time that elapsed between those two events. There is some distortion due to the time to animate an event using a moving object, since this depends on the length of the arc traversed. For this reason, the use of blinking lines to animate flow in linear time or increasing the speed of the ball tends to produce an animation whose timing more closely corresponds to the actual time behavior of the simulation.

If the modeler chooses the Select Activity menu item, he can selectively specify (by pointing to them) the nodes, queues, and chains for which animation activity will be shown. Nodes and queues for which animation activity will be shown are displayed in green, while those where the animation will be suppressed are displayed in yellow.

The animation control panel provides the modeler with elements that display the time, event numbers and job id's during the animation and also serves as an input panel for the modeler to change these values in order to restart the animation at another point. The first row of the panel, labelled "current", displays the above-mentioned information for the job currently in motion. Either the simulated time or the event can be directly entered by the modeler by picking one of the appropriate elements and entering the desired value. This will cause the animated time and event count to be moved forward or backward to that point in the simulation. Selecting the Start Animation menu item then starts the animation at the displayed simulated time. Either an absolute or a signed value can be entered for the current time or event. In the latter case, the signed value is taken as an offset from the previously displayed absolute value and the new absolute value is computed accordingly. For example if the current simulated time were displayed as 1.377 and +.5 is entered into the current time element, the new displayed value of the current time would be the time of the first event after 1.877. On restart, that is where the view of the animation would start.

The entries in the second row in the animation control panel are used to set breakpoints. Entering an absolute time value, event count, or job Id will cause an on-going animation to stop when that breakpoint is reached. A positive signed quantity can also be entered in the time and event breakpoint windows. As above, a signed value is taken as an offset from the current time or event. When the animation proceeds, this breakpoint offset value is visually counted down until it reaches 0. Note that entering a + 1 in the event breakpoint element and continually selecting the Start Animation menu item will have the effect of single stepping the animation event by event.

The third row in the animation control panel serves a dual function. If the Snapshot mode has been selected (by pointing to the word Snapshot and clicking with the mouse), elements will be shown in the first two columns. If any value is entered into either of these elements, job movement will not be animated. Instead, the state (i.e., occupancy counts) of the model will be displayed at fixed intervals of either simulated time or number of events. For example, if the current simulated time is 2.277, entering a value of 3.0 in the snapshot time element and starting the animation would cause the state of the model at simulated times 5.277, 8.277, 11.277, etc. to be displayed. In this manner, one can construct an animated "movie" of the time evolution of the occupancies of the various nodes and queues in the model. The use of snapshots as an animation control capability was introduced by Melamed and Morris 1985 in PAW.

If the Trace mode has been selected (by clicking the mouse on the word Trace), the third element (Job Id) on the bottom line will be displayed. Entering a job Id in this element will cause the animation facility to explicitly trace the activity of this single job. The animation facility will show this job in a different color, will show its occupancy at nodes with a different shading color, and will follow the job as it moves between the submodel layers of the model. Animation of the other jobs in the model can be selectively turned on or off during this tracing.

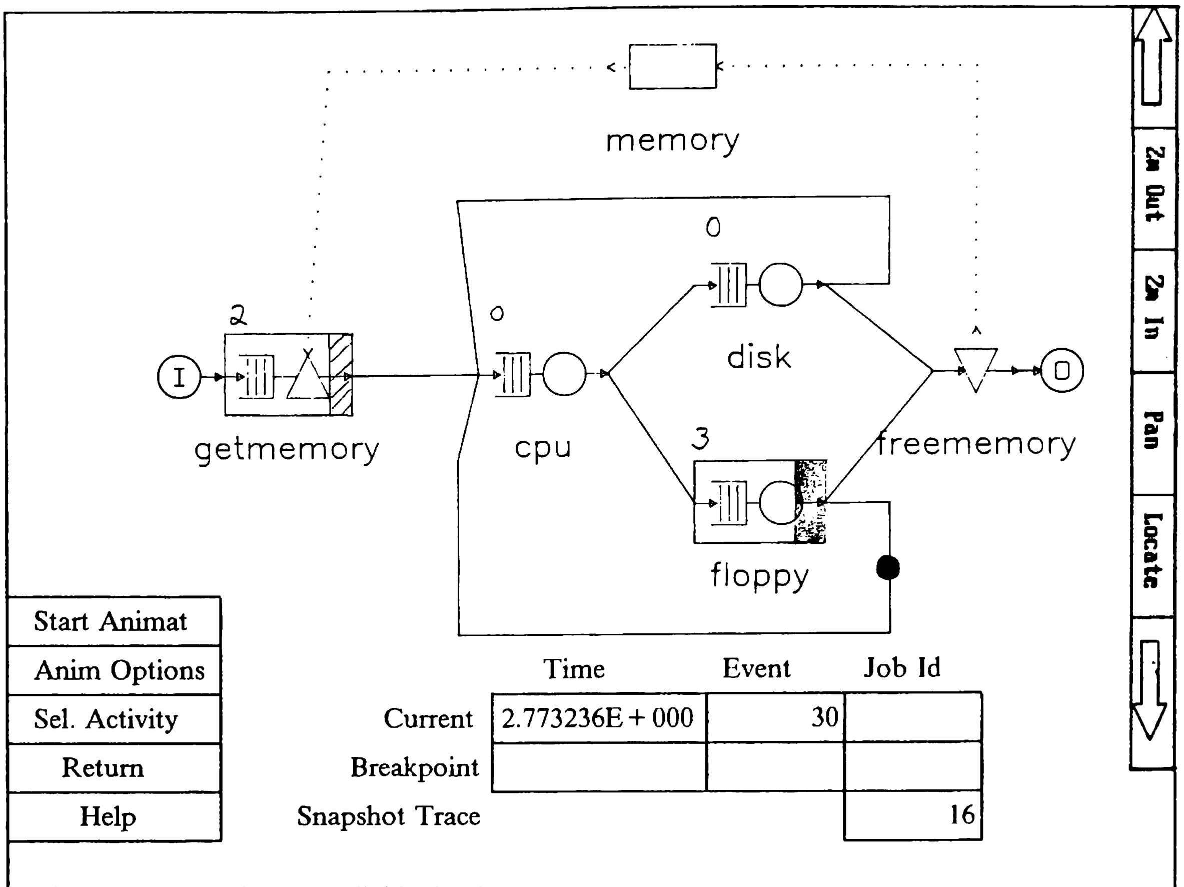
As an example, suppose when running the animation we see from the first row of elements that at time 2.77 and event 30 job Id 16 moves from the terminals node to the host2 invocation node. We could stop the animation, select trace and enter job Id 16, move the simulation time back by entering 2.0 in the time element and restart the animation. The animation automatically layers up to the main model layer as job 16 moves from the terminals to the host2 invocation node. It then layers down to the host2 invocation display to follow the movement of job 16 through this invocation of the submodel (see Figure 4). Job 16 is shown in a different color from the other jobs moving in this invocation of the submodel and when it is waiting at a node, the shading of that node is in a different color. When job 16 reaches the output node of the submodel, the main model is redisplayed to show job 16 leaving the submodel and returning to the terminals node.

The animation provides visual information about job and token flow and node and queue occupancy counts. Frequently, particularly when debugging a model, the modeler requires more detailed information about the simulation. For example, the modeler may be interested in the evaluation of the boolean expressions involved in a routing statement or the value of a random number generated at a particular time in the simulation. At any point the modeler can toggle to the mainframe to view the detailed trace file. For example, suppose the animation has been interrupted at time 8.456 (at which time, the job in motion had moved from the cpu node to the floppy disk node on the basis of a boolean value). To see the truth values for the various boolean expressions, one would toggle to the mainframe and view the trace file with the complete state information at time 8.456.

4. SUMMARY AND FUTURE WORK

We have described the animation facility in RESQME. It provides the modeler with a display of the movement of jobs through the queueing network. Individual jobs can be traced through the nodes and submodel invocations of the network hierarchy. To produce an animation, the modeler need only specify that a trace file be created during the simulation. No special symbols nor graphics layout is required. The one graphic diagram serves the dual roles of model specification and output analysis. The output analysis includes performance measure graphs and the animation discussed in this paper.

There are two main areas in which we would like to further develop the animation facility. One is to add



	Time	Event	Job Id
Current	2.773236E+000	30	
Breakpoint			
Snapshot Trace			16

Figure 4. Tracing an Individual Job

more display information. The new display information would include showing performance measures during the animation, so that the modeler can see, in moving graphs and gauges, the changes in the output statistics which now are available only at the end of the run.

The second area involves our plans to port the host simulation code to the workstation. This will give the modeler an alternative to host processing for small problems or pilot runs. With the simulation on the workstation, we can provide animation during simulation. This would provide the added value of allowing the modeler to change parameter values during the run and then be able to directly view the impact.

ACKNOWLEDGEMENTS

We would like to express our thanks to Peter Welch for his support and encouragement of this work, and to Charles Sauer for his work on RESQ and his continued interest in RESQ and RESQME. We would also like to thank Al Blum, Gary Burkland, Janet Chen, Paul Loewner and Geoff Parker for their work in improving the package. We are grateful to the many other colleagues and RESQME users who have helped improve this package.

REFERENCES

Conway, R., Maxwell, W., McClain, J., Worona, S. (1987). *User's Guide to XCELL+ Factory Modeling System*. The Scientific Press, Redwood City, CA.

Cox, S. (1987). The Interactive Graphics and Animation of GPSS/PC. *Proceedings of the 1987 Winter Simulation Symposium*, Atlanta, GA, 276-285.

Gordon, R. F., MacNair, E. A., Welch, P. D., Gordon, K. J. and Kurose, J. F. (1986). Examples of Using the RESearch Queueing Package Modeling Environment (RESQME). *Proceedings of the 1986 Winter Simulation Conference*, Washington, D.C., 494-503.

Gordon, R. F., MacNair, E. A., Gordon, K. J. and Kurose, J. F. (1987). A Visual Programming Approach to Manufacturing Modeling. *Proceedings of* *the 1987 Winter Simulation Conference*, Atlanta, GA, 465-471.

Gordon, R. F., MacNair, E. A., Gordon, K. J. and Kurose, J. F. (1988). Higher Level Modeling in RESQME. *Proceedings of the European Simulation Multiconference 1988*, Nice, France, 52-57.

Kilgore, R., Healy, K. (1987). Animation Design with Cinema. *Proceedings of the 1987 Winter Simulation Conference*, Atlanta, GA, 261-268.

Kurose, J. F., Gordon, K. J., Gordon, R. F., MacNair, E. A. and Welch, P. D. (1986). A Graphics-Oriented Modeler's Workstation Environment for the RESearch Queueing Package (RESQ). *1986 Proceedings Fall Joint Computer Conference*, Dallas, 719-728.

Melamed, B. and Morris, R. J. T. (1985). Visual Simulation: The Performance Analysis Workstation. *IEEE Computer* **18**, 87-94.

Miles, T., Sadowski, R., Werner, B. (1988). Animation with Cinema. *Proceedings of the 1988 Winter Simulation Conference*, San Diego, 180-187.

O'Keefe, R. (1987). What is Visual Interactive Simulation. *Proceedings of the 1987 Winter Simulation Symposium*, Atlanta, GA, 462-464.

Russel, E. (1988). SIMSCRIPT II.5 ¯ and SIMGRAPHICS: A Tutorial. *Proceedings of the 1988 Winter Simulation Conference*, San Diego, 115-124.

Sauer C., MacNair, E., Salza, S. (1980). A Language for Extended Queueing Network Models. *IBM Journal of Research and Development* **24**, 747-755.

Standridge, C., Pritsker, A., Stein, C. (1987). A Tutorial on Tess, The Extended Simulation Support System. *Proceedings of the 1987 Winter Simulation Conference*, Atlanta, GA, 238-246.

AUTHORS' BIOGRAPHIES

ANIL AGGARWAL received his B.E. Honors degree from Birla Institute of Technology and Science, Pilani, Rajasthan, India in 1985, and his M.S. in Computer Science from the University of Massachusetts in 1988. He is currently working for Intel in Oregon as a Software Engineer specializing in the Unix Operating System.

Anil Aggarwal
Intel Corp.
5200 N.E. Elam Young Parkway
Hillsburo, OR 97124, U.S.A.
(503) 696-2253

KURTISS J. GORDON received his B.S. in Physics from Antioch College in 1964, his M.A. and Ph.D. in Astronomy from the University of Michigan in 1966 and 1969, and his M.S.E.C.E. in Computer Systems from the University of Massachusetts in 1985. Until 1984, he taught in the Five-College Astronomy Department, then was a Senior Postdoctoral Research Associate in the Department of Computer and Information Science at the University of Massachusetts in Amherst. Since 1987, he has been Coordinator of Scientific Imaging Applications in the University Computing Center. Dr. Gordon's interests include the display and interpretation of large bodies of data, modeling and performance evaluation, and graphical user interfaces. He is a member of the American Astronomical Society, Sigma Xi, ACM, and IEEE.

Kurtiss J. Gordon
University Computing Center
University of Massachusetts
Amherst, Mass. 01003, U.S.A.
(413) 545-2690

JAMES F. KUROSE received a BA degree in Physics from Wesleyan University in Middletown, Conn. in 1978 and an MS and PhD degree in Computer Science from Columbia University in 1980 and 1984, respectively. Since 1984, he has been an Assistant Professor in the Department of Computer and Information Science at the University of Massachusetts, Amherst, MA., where he currently leads several research efforts in the areas of computer communication networks, distributed systems, and modeling and performance evaluation. He has also been associated with the performance modeling methodology group at the IBM T.J. Watson Research Center as a consultant since 1980 and has served as a consultant for various other companies as well. Professor Kurose is a member of Phi Beta Kappa, Sigma Xi, IEEE, and ACM. He is an Associate Editor for IEEE Transactions on Communications and a Guest Editor for the IEEE Journal on Selected Areas in Communications.

James F. Kurose
Department of Computer and Information Science
University of Massachusetts
Amherst, Mass. 01003, U.S.A.
(413) 545-1585

ROBERT F. GORDON is a research staff member in the modeling and analysis software systems group at the IBM Thomas J. Watson Research Center. He received a B.S. in mathematics and physics from the City College of New York in 1964, an M.S. in mathematics from Carnegie Institute of Technology in 1965 and Ph.D. in mathematics from Carnegie-Mellon University in 1969. From 1968 to 1974, he was Manager of Mathematics and Programming for Hoffmann-La Roche, Inc., where he developed mathematical models for marketing, production planning and distribution. From 1974 to 1983, Dr. Gordon was Director of Information Management Services at Avis, where he headed the operations research, timesharing systems, and systems and programming groups. Dr. Gordon is an adjunct professor at Hofstra University. He is a member of Phi Beta Kappa, Sigma Xi and ORSA.

Robert F. Gordon
IBM Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598, U.S.A
(914) 789-7170

EDWARD A. MACNAIR joined IBM in 1965. He is a Research Staff Member in the Computer Science Department at the IBM Thomas J. Watson Research Center. He is currently in the Modeling and Analysis Software Systems project developing modeling programs to solve extended queueing networks. He is one of the developers of the Research Queueing Package (RESQ), a tool for the solution of generalized queueing networks. He is a coauthor with Charles H. Sauer of *Simulation of Computer Communication Systems*, Prentice-Hall, 1983 and *Elements of Practical Performance Modeling*, Prentice-Hall, 1985. He received a B.A. in mathematics from Hofstra University in 1965 and an M.S. in Operations Research from New York University in 1972. He is a member of the ACM, ORSA and TIMS.

Edward A. MacNair
IBM Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598, U.S.A
(914) 789-7561