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

NETWORK II.5 TUTORIAL
NETWORK MODELING - WITHOUT PROGRAMMING

William J. Garrison
CACI
345 North York Road, Hatboro, PA 19040

ABSTRACT

This tutorial will acquaint the reader with a powerful modelling tool which can dramatically reduce the amount of time it takes to simulate a computer system. The Network II.5 world view and the class of problems it addresses will be discussed. A summary of NETWORK II.5 capabilities will be presented. To tie the tutorial concepts together, an example of using NETWORK II.5 is included.

1. BACKGROUND

Simulation languages ease the burden of programming simulation problems. However, in certain application areas higher level facilities can be designed to speed the simulation analysis. While this higher level facility would probably be written in a simulation language, it would require no programming on the part of the user. Instead, the user is interactively guided in building a description of the system to be simulated. The higher level facility would then run the model. This allows someone to dive into a simulation problem without the delay inherent in becoming fluent in a simulation language and coding up a model. NETWORK II.5 is such a higher level facility for modeling computers and communication networks.

2. NETWORK II.5 WORLD VIEW

There are two concepts at the heart of the design of NETWORK II.5. First, items to be simulated are viewed strictly based on their function. Second, timing is the most important consideration in the simulation.

There are four main functions in the computer and communications world. They are processing, transfer, storage and control. NETWORK II.5 has a separate general purpose building block to model each of these functions. Network's *processing, transfer* and *storage* building blocks are used to model hardware. Network's *module* building block models both the software and control functions that may be in either hardware, firmware or software. Every device is described in terms of the function it provides to the system to be simulated. Many real world devices require more than one NETWORK II.5 building block to be fully described because they embody more than one function. For example, a personal computer might require two processing elements (main processor and co-processor), three transfer devices (internal bus, serial port and parallel port) and two storage devices (disk and main memory).

By concentrating on providing a few powerful function oriented building blocks, the explosive progress of technology will never outpace NETWORK II.5. Whether data is being moved via a bus, satellite link, fiber optics or some advanced technology yet discovered, the function is the same. Information is transferred. In addition, the limited number of building blocks makes NETWORK II.5 a very easy tool to learn to use.

NETWORK II.5 simulates a system based on timing. Instructions that run on a processing element are not executed in the sense that at the end of an add instruction, the result is 4 and the overflow bit is reset. Instead, the effect of the instruction on system operation is modeled. If the instruction sent a message, a transfer device will be tied up for the amount of time it took to send the message. If a fetch from memory is required, at the proper time and for the proper duration, a transfer device and a storage device will be utilized. NETWORK II.5's timing orientation facilitates measurement of such system considerations as response time, conflicts, device utilization, etc.

3. THE NETWORK II.5 APPLICATION DOMAIN

NETWORK II.5 is designed to simulate systems in which devices are requesting, manipulating and distributing information and/or making decisions based on the system state. Local area networks, telephone networks, distributed database systems, automotive electronic systems, military communication systems and automated factories are just a few of the current applications of NETWORK II.5.

NETWORK II.5 is extremely flexible, allowing the portions of a computer system of special interest to be modeled at a finely detailed level while the rest of the system is modeled at a courser level. There is no arbitrary limit to the number of processing, transfer, storage and module building blocks which may be used to describe a system. In addition, any device interconnection scheme is allowed.

4. WITHOUT PROGRAMMING

You do not need to learn (or own) any computer language to run NETWORK II.5. NETWORK II.5 speaks your language! You describe the system being simulated using terms like Processing Element, Instruction and Protocol = Collision. All errors are expressed in terms easy to understand. "PE 5 does not connect to TD 2" is much easier to understand than "Subscript out of range".

NETWORK II.5 has an interactive front end called NETGIN that uses a mouse and keyboard dialog to quickly build, modify and display a system description. NETGIN will also interactively diagnose logic errors in a user's model description, allowing quick corrections. It also can produce diagrams of both the hardware and software which the user has described. When the system description is complete, NETWORK will run the simulation using an interactive dialog to set up the simulation parameters. If desired, the user can interactively follow the simulation's progress by requesting a narrative trace of the simulation. At the end of a NETWORK II.5 simulation, the user may request a post processed report called NETPLOT. By analyzing a log file produced during the simulation run, NETPLOT produces both timeline diagrams of simulation activity and utilization graphs.

NETANIMATION allows a user to draw the computer system simulated and watch the simulation graphically represented as the simulation progresses. A wide variety of styles and colors are available to represent objects in the simulated system. Zooming and panning allow very large systems to be displayed. Simulations may run automatically or single stepped. Descriptive trace messages may be displayed concurrent with the animation to explain the details of the simulation activity. Currently, NETANIMATION runs on PC's, SUN workstations and VAXstations. It can, however, animate NETWORK II.5 simulations executed on any other hardware by running in a post processed mode. A sample NETANIMATION display is given as Figure 2.

5. NETWORK II.5 AVAILABILITY

NETWORK II.5 is currently available on IBM (CMS and TSO), VAX/VMS, VAXstation, PC-DOS, OS/2, SUN/3, SUN/4, and SUN 386i machines. Versions for other machines will be added as required. All versions work identically, so that models can be moved from one manufacturer's machine to another without any changes.

6. NETWORK II.5 REPORTS

There are eleven basic reports provided by NETWORK II.5. The tabular reports include Module Summary, Processing Element, Transfer Device, Storage Device, Semaphore and Message Statistic reports. NETWORK II.5 also offers a Narrative Trace, a Snapshot Report, a Hardware diagram and a Software Diagram report.

The tabular reports are produced both at the end of the simulation and any other user specified times. Samples of these reports are given as figures 5 to 9. The Narrative Trace report is produced interactively upon demand and chronicles the progress of the simulation event by event as they occur. This report interacts with the user to allow the user to stop the simulation or produce additional reports if things are going wrong. The Snapshot report (figure 10) lists the current status of every hardware device, module, semaphore and message in the simulation. It is produced both as a part of the end of tabular reports and interactively during a simulation run in response to a user request. The Timeline and Utilization reports (figure 11) are post processed reports that act upon a database produced during a simulation run to show the status of every hardware device and every semaphore in the simulation. The time span plotted on these reports is user specified so that a user can go back and expand the time scale of a period of interest several times until the needed information is obtained.

7.0 THE TUTORIAL EXAMPLE

For the purpose of illustration, an extremely simplified example of using NETWORK II.5 is presented here. See the references for examples of a more realistic problem both presented and solved.

An office contains 2 computers on the same serial bus. Files are sent from one computer to another. Computer A requests a file called DATA from Computer B every 30 seconds. Computer B then responds, sending the 700 bit file. The serial bus runs at 1200 baud and adds 1 parity bit to every 7 bits of data. The simulation is to be run for 1000 (simulated) seconds.

7.1 PROBLEM FORMULATION

HARDWARE - Each Processing Element will need 1 instruction. Computer A will need an instruction which sends a message to request the file. Computer B needs an instruction that sends the requested file in the form of a message. In NETWORK II.5, all processor to processor communication is performed using messages.

SOFTWARE - Computer A will need a module which runs every 30 seconds and sends a file request message. Computer B will need a module which, when triggered by the receipt of a request message, sends the requested file.

The user prepares the input file to NETWORK using the interactive program NETGIN. NETGIN uses a mouse and keyboard dialog to build the description of the system to be simulated and provides interactive diagnosis of logic errors. A sample NETGIN screen is presented as Figure 1.

The output of NETGIN is a file which becomes the input to NETWORK. The file is easily readable and useful for documenting the simulation performed. The file produced by NETGIN to solve this example problem is given as Figure 3.

The dialog required to run this problem is included as Figure 4. If desired, a user can watch the simulation progress graphically using NETANIMATION (figure 2). The tabular reports produced by running this example problem are included as Figures 5 to 10.

REFERENCES

1. Garrison, W.J., NETWORK II.5 USER'S MANUAL, Version 5.0, CACI, Inc., September, 1989

2. Russell, E.C., BUILDING SIMULATION MODELS WITH SIMSCRIPT II.5, CACI, Inc., January, 1983.

3. CACI, INC., A QUICK LOOK AT NETWORK II.5, September, 1988.

4. Karplus and Cheung, PERFORMANCE EVALUATION TOOLS FOR SIMULATORS CONSISTING OF NETWORKS OF MICROCOMPUTERS, Computer Science Department, University of California, Los Angeles, 1984.

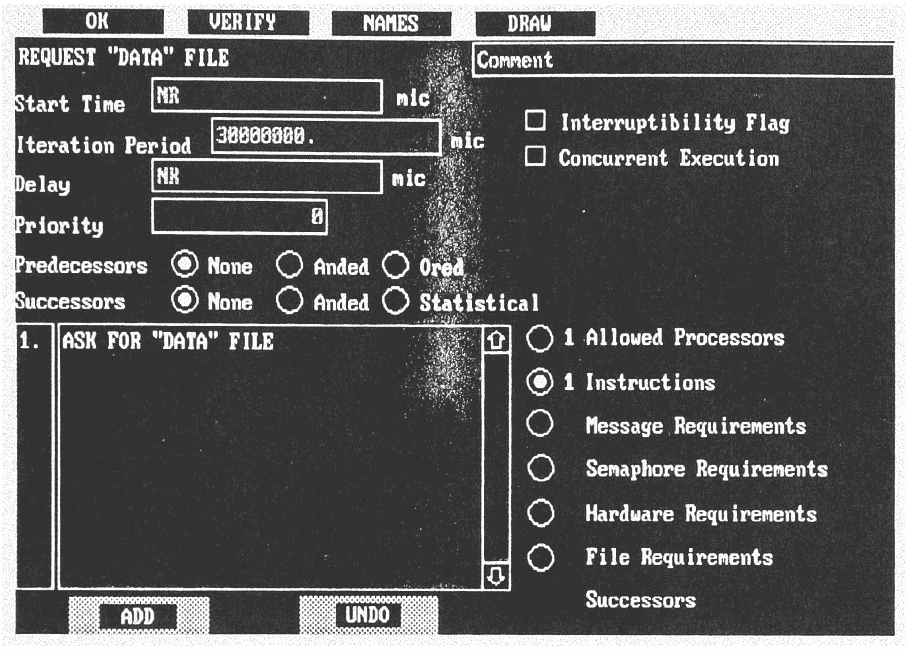

Figure 1 : Describing A Software Module with NETGIN

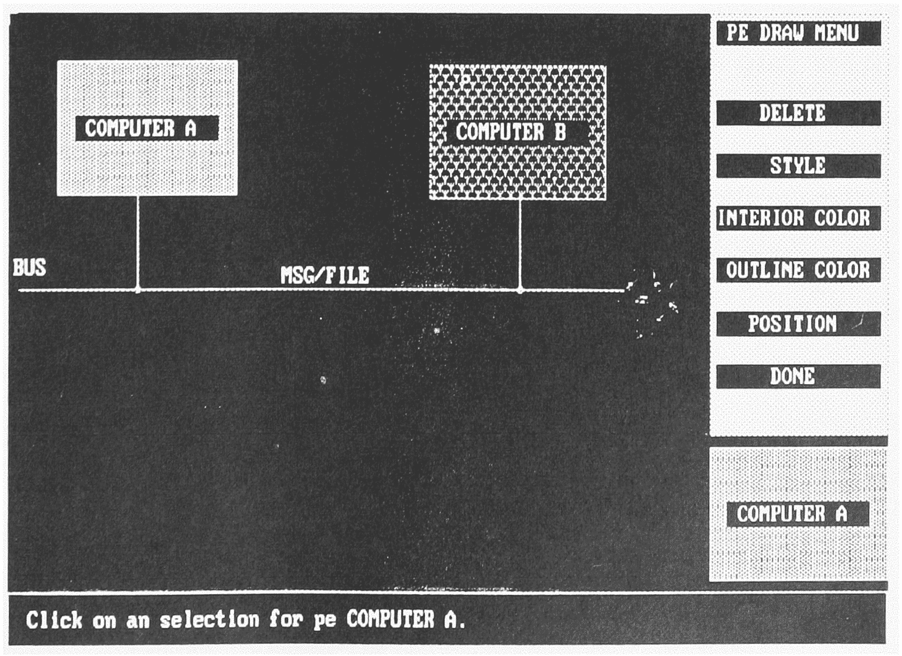

Figure 2 : NETANIMATION

```
* CASE 1A - A 2 COMPUTER, 1 BUS ARCHITECTURE
***** PROCESSING ELEMENTS - SYS.PE.SET
HARDWARE TYPE = PROCESSING
  NAME = COMPUTER A
    BASIC CYCLE TIME =      1.000000 MICROSEC
    INPUT CONTROLLER = YES
    INSTRUCTION REPERTOIRE =
      INSTRUCTION TYPE = MESSAGE
        NAME ; ASK FOR "DATA" FILE
          MESSAGE ; ASK FOR "DATA" FILE
          LENGTH ; 256 BITS
          DESTINATION PROCESSOR ; COMPUTER B
  NAME = COMPUTER B
    BASIC CYCLE TIME =      1.000000 MICROSEC
    INPUT CONTROLLER = YES
    INSTRUCTION REPERTOIRE =
      INSTRUCTION TYPE = MESSAGE
        NAME ; SEND "DATA" FILE
          MESSAGE ; SEND "DATA" FILE
          LENGTH ; 4096 BITS
          DESTINATION PROCESSOR ; COMPUTER A
***** BUSSES - SYS.BUS.SET
HARDWARE TYPE = DATA TRANSFER
  NAME = BUS
    CYCLE TIME =    833.00 MICROSEC
    BITS PER CYCLE =       1
    CYCLES PER WORD =       7
    WORDS PER BLOCK =       1
    WORD OVERHEAD TIME =    833.00 MICROSEC
    BLOCK OVERHEAD TIME =      0.   MICROSEC
    PROTOCOL = FIRST COME FIRST SERVED
    BUS CONNECTIONS =
      COMPUTER A
      COMPUTER B
***** MODULES - SYS.MODULE.SET
SOFTWARE TYPE = MODULE
  NAME = REQUEST "DATA" FILE
    PRIORITY =      0
    INTERRUPTABILITY FLAG = NO
    CONCURRENT EXECUTION = NO
    ITERATION PERIOD = 3.00E+07 MICROSEC
    ALLOWED PROCESSORS =
      COMPUTER A
    INSTRUCTION LIST =
      EXECUTE A TOTAL OF ; 1 ASK FOR "DATA" FILE
  NAME = TRANSMIT "DATA" FILE
    PRIORITY =      0
    INTERRUPTABILITY FLAG = NO
    CONCURRENT EXECUTION = NO
    ALLOWED PROCESSORS =
      COMPUTER B
    REQUIRED MESSAGES =
      ASK FOR "DATA" FILE
    INSTRUCTION LIST =
      EXECUTE A TOTAL OF ; 1 SEND "DATA" FILE
```

Figure 3 : Data File For The Tutorial Example

```
Welcome to CACI NETWORK II.5, version 5.0.

Please enter the name of the NETWORK input data file. (or ABORT)
case1
NETWORK INPUT FILE IS:  \NET\USER\CASE1.NET
NETWORK OUTPUT FILE IS:  \NET\USER\CASE1.LIS
NETWORK PLOT FILE IS:  \NET\USER\CASE1.PIN

Your input file is entitled
CASE 1A - A 2 COMPUTER, 1 BUS ARCHITECTURE

Do you want a copy of your input file in the listing file? (Y/N/ABORT)
yes
   4 Global Flags initialized.
   2 Processing Elements initialized.
   1 Transfer Devices initialized.
   2 Modules initialized.

NETWORK took  0 MINUTES  3 SECONDS to read the    74 line data file

Do you want a report in the listing file showing the input file complete with
all assumed default values?  (Y/N/ABORT)
no

NO FATAL INPUT ERRORS HAVE BEEN DETECTED.

Enter the time unit which you wish to use for input.
It must be SECONDS, MILLISECONDS, OR MICROSECONDS. (SEC/MIL/MIC)
seconds

Enter length of simulation (IN SECONDS)
1000

Do you want the statistics reset during the run? (Y/N/ABORT)
no

Do you wish to have periodic reporting?  (Y/N/ABORT)
no

Should the plot data file cover All, Part or None of the simulation? (A/P/N)
all

Do you want to specify how to handle runtime errors now? (Y/N/ABORT)
no

Do you wish to trace the event flow? (Y/N/ABORT)
no

Simulation begins.

Current Time is    23.142163 SECONDS

NETWORK took  0 MINUTES 5 SECONDS to perform the simulation

Simulation stopped at  1000.000000 SEC.
No runtime warning or error messages were issued.
```

Figure 4 : Running The Tutorial Example using NETWORK

CACI NETWORK II.5 RELEASE 5.00 06/14/1989 10:42:34 PAGE 1

CASE 1A - A 2 COMPUTER, 1 BUS ARCHITECTURE

PROCESSING ELEMENT UTILIZATION STATISTICS

FROM 0. TO 1000. SECONDS

(ALL TIMES REPORTED IN MICROSECONDS)

PROCESSING ELEMENT NAME	COMPUTER A	COMPUTER B
STORAGE REQUESTS GRANTED	0	0
INTERRUPTED REQUESTS	0	0
AVERAGE WAIT TIME	0.	0.
MAXIMUM WAIT TIME	0.	0.
STD DEV WAIT TIME	0.	0.
GEN STORAGE REQUESTS	0	0
FILE REQUESTS GRANTED	0	0
INTERRUPTED REQUESTS	0	0
AVERAGE WAIT TIME	0.	0.
MAXIMUM WAIT TIME	0.	0.
STD DEV WAIT TIME	0.	0.
TRANSFER REQUESTS GRANTED	33	33
INTERRUPTED REQUESTS	0	0
AVERAGE WAIT TIME	0.	0.
MAXIMUM WAIT TIME	0.	0.
STD DEV WAIT TIME	0.	0.
INPUT CONTROLLER REQUEST	33	33
DEST PE REQUESTS GRANTED	0	0
AVERAGE WAIT TIME	0.	0.
MAXIMUM WAIT TIME	0.	0.
STD DEV WAIT TIME	0.	0.
RESTARTED INTERRUPTS	0	0
AVG TIME PER INTERRUPT	0.	0.
MAX TIME PER INTERRUPT	0.	0.
STD DEV INTERRUPT TIME	0.	0.
MAX INTERRUPT QUEUE SIZE	0	0
AVG INTERRUPT QUEUE SIZE	0.	0.
STD DEV INTERRUPT QUEUE	0.	0.
MAX MODULE QUEUE SIZE	1	1
AVG MODULE QUEUE SIZE	0.	0.
STD DEV MODULE QUEUE	0.	0.
PER CENT PE UTILIZATION	.805	12.870

Figure 5 : The NETWORK II.5 Processing Element Report

```
CACI NETWORK II.5  RELEASE 5.00    06/14/1989      10:42:34            PAGE   2
                   CASE 1A - A 2 COMPUTER, 1 BUS ARCHITECTURE

                          COMPLETED INSTRUCTION REPORT

                          FROM    0.  TO  1000. SECONDS

  INSTRUCTION NAME                 COUNT     INSTRUCTION NAME              COUNT

COMPUTER A
  ASK FOR "DATA" FILE                 33

COMPUTER B
  SEND "DATA" FILE                    33
```

Figure 6 : The NETWORK II.5 Instruction Execution Report

```
CACI NETWORK II.5  RELEASE 5.00    06/14/1989      10:42:34            PAGE   3
                   CASE 1A - A 2 COMPUTER, 1 BUS ARCHITECTURE

                     TRANSFER DEVICE UTILIZATION STATISTICS

                          FROM    0.  TO  1000. SECONDS

                       (ALL TIMES REPORTED IN MICROSECONDS)


TRANSFER DEVICE NAME            BUS



TRANSFER REQUESTS GRANTED        66
INTERRUPTED REQUESTS              0
AVG REQUEST DELAY                 0.
MAX REQUEST DELAY                 0.
STD DEV REQUEST DELAY             0.

INTERRUPTED TRANSFERS             0

COMPLETED TRANSFERS              66
AVG USAGE TIME              2072087.500
MAX USAGE TIME              3900106.000
STD DEV USAGE TIME          1828018.500

AVG QUEUE SIZE                    0.
MAX QUEUE SIZE                    1.000
STD DEV QUEUE SIZE                0.

PER CENT OF TIME BUSY            13.676
```

Figure 7 : The NETWORK II.5 Transfer Device Report

CACI NETWORK II.5 RELEASE 5.00 06/14/1989 10:42:34 PAGE 4

CASE 1A - A 2 COMPUTER, 1 BUS ARCHITECTURE

COMPLETED MODULE STATISTICS

FROM 0. TO 1000. SECONDS

(ALL TIMES REPORTED IN MICROSECONDS)

MODULE NAME	REQUEST "DATA" FILE	TRANSMIT "DATA" FILE
HOST PE	COMPUTER A	COMPUTER B
COMPLETED EXECUTIONS	33	33
NUM PRECONDITION TIME	33	33
AVG PRECONDITION TIME	0.	0.
MAX PRECONDITION TIME	0.	0.
MIN PRECONDITION TIME	0.	0.
STD DEV PRECOND TIME	0.	0.
AVG EXECUTION TIME	244069.000	3900106.000
MAX EXECUTION TIME	244069.000	3900106.000
MIN EXECUTION TIME	244069.000	3900106.000
STD DEV EXECUTION TIME	0.	0.

Figure 8 : The NETWORK II.5 Module Report

CACI NETWORK II.5 RELEASE 5.00 06/14/1989 10:42:34 PAGE 5

CASE 1A - A 2 COMPUTER, 1 BUS ARCHITECTURE

MESSAGE STATISTICS

FROM 0. TO 1000. SECONDS

(ALL TIMES REPORTED IN MICROSECONDS)

MESSAGE NAME	ASK FOR "DATA" FILE	SEND "DATA" FILE
RECEIVING PE	COMPUTER B	COMPUTER A
NUMBER QUEUED	33	33
NUMBER USED	33	0
AVG QUEUE TIME	0.	0.
MAX QUEUE TIME	0.	0.
MIN QUEUE TIME	0.	0.
STD DEV QUEUE TIME	0.	0.

Figure 9 : The NETWORK II.5 Message Report

```
CACI NETWORK II.5  RELEASE 5.00     06/14/1989      10:42:34            PAGE    6

                     CASE 1A - A 2 COMPUTER, 1 BUS ARCHITECTURE

                           S N A P S H O T    R E P O R T

                                AT  1000. SECONDS


pe COMPUTER A is IDLE
     message SEND "DATA" FILE (  1)            4096. bits are queued
pe COMPUTER B is IDLE

td BUS is IDLE
```

Figure 10 : The NETWORK II.5 Snapshot Report

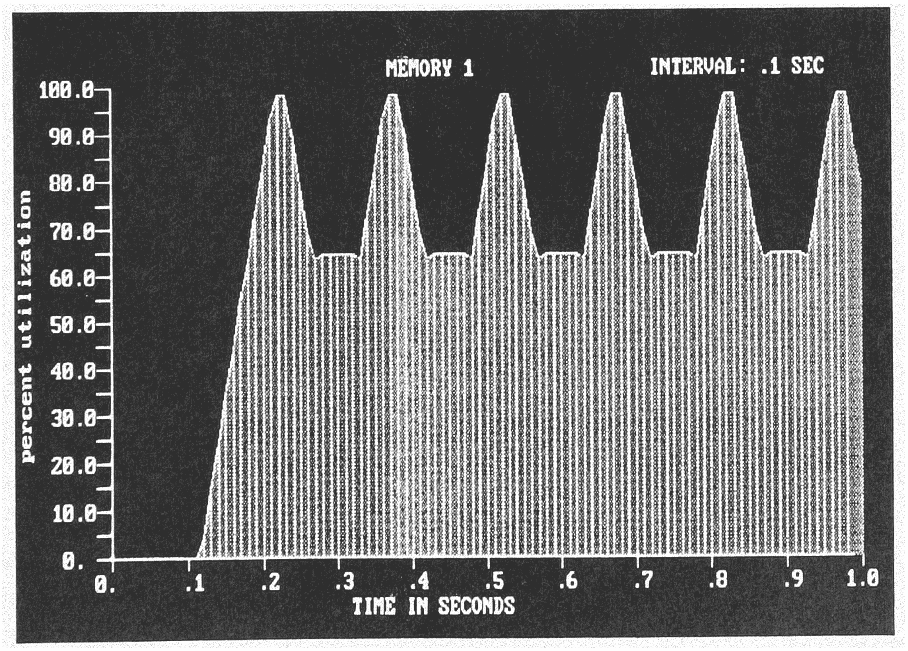

Figure 11 : NETPLOT Utilization Graph

WILLIAM J. GARRISON is an Executive Associate in the CACI Products Company. He led the development of NETWORK II.5, a Computer and Communications Network Simulator, and has been active in applying simulation to manufacturing problems. Currently, he is involved in the development of future versions of NETWORK II.5 and teaches courses and consults on the use of both NETWORK II.5 and the simulation language SIMSCRIPT II.5.

William Garrison received the B.S. degree in Electrical Engineering from the University of Pennsylvania in 1975. He received a M.S. degree in Computer Science from the University of Pennsylvania in 1980. He is a lifetime member of the engineering honor society Tau Beta Pi and a member of the IEEE Computer Society.

William J. Garrison
CACI Products Company
345 North York Road
Hatboro, PA 19040
(215) 675-0895