7

MASTER

PREPRINT UCRL- 78049

CONF-760703--2

Lawrence Livermore Laboratory

MHSS--A MATERIAL HANDLING SYSTEM SIMULATOR

Charles L. Pomernacki and Roy B. Hollstien

April 7, 1976

Proceedings of the 1976 Summer
Computer Simulation Conference
Washington, D.C. July 12-14, 1976

This is a preprint of a paper intended for publication in a journal or proceedings. Since changes may be made before publication, this preprint is made available with the understanding that it will not be cited or reproduced without the permission of the author.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

NOTICE
This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Energy Research and Development Administration, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

* MHSS — A MATERIAL HANDLING SYSTEM SIMULATOR

Roy B. Wollstien* and Charles L. Pomernacki
Electronics Engineering Department
Lawrence Livermore Laboratory, Livermore, California

ABSTRACT

A Material Handling System Simulator (MHSS) program is described that provides specialized functional blocks for modeling and simulation of nuclear material handling systems. Models of nuclear fuel fabrication plants may be built using functional blocks that simulate material receiving, storage, transport, inventory, processing, and shipping operations as well as the control and reporting tasks of operators or on-line computers. Blocks are also provided that allow the user to observe and gather statistical information on the dynamic behavior of simulated plants over single or replicated runs. Although it is currently being developed for the nuclear materials handling application, MHSS can be adapted to other industries in which material accountability is important. In this paper, emphasis is on the simulation methodology of the MHSS program with application to the nuclear material safeguards problem.

INTRODUCTION

Computer simulation can play an important role in the design and evaluation of nuclear material safeguards systems (1,2) and has already been used in the simulation of REal TIme MAterial Control (RETIMAC) concepts for the U.S. Nuclear Regulatory Commission (3). There is a real need for a standardized simulation methodology in the nuclear industries to avoid the added complexity that the use of different simulation languages and different ad hoc techniques would bring to the problem of evaluating the safeguards systems of applicants seeking license to process nuclear materials.

Standard methods for material handling simulation can of course be developed using any of the established general-purpose simulation languages (GPSS, SIMSCRIPT, SIMULA, etc.), but all languages are not supported by all computer systems, so to make them generally available, it would be necessary to develop standardized methods of material handling simulation in several simulation languages. Due to basic differences in the languages themselves, their input and output formats, etc., we believe this would be impractical and that standardized nuclear material handling simulation programs should be based on procedural languages that are generally available and in widespread use.

The GASP IV Simulation Language (4), on the other hand, is a collection of FORTRAN subprograms that could be used to provide a specialized material handling system simulator, perhaps based in part on the approach used in Q-GERTS, a general-purpose queueing system simulator (5). MHSS began as an effort in this direction, but it has evolved into a conceptually different approach. We do, however, acknowledge the important influence that our first efforts to use GASP IV have had on the development of MHSS.

MHSS system models are sets of interacting, high-level blocks that represent corresponding functional elements of material handling systems. In block diagrams of these system models, the MHSS user shows only the interconnecting paths along which material is transferred from block to block, as in Fig. 1(a). Interactions among blocks are implied by their functional specifications, but the communication paths required to implement these interactions are not shown, as in Fig. 1(b), and the MHSS user does not have to be concerned with the details of the underlying simulation methodology.

MHSS is not a language in which simulation programs are written. It is a program that builds and runs model systems defined by the user in terms that parallel the functional specifications of elements of the simulated systems.

*Computer Science and Statistics Department
California Polytechnic State University, San Luis Obispo
San Luis Obispo, California

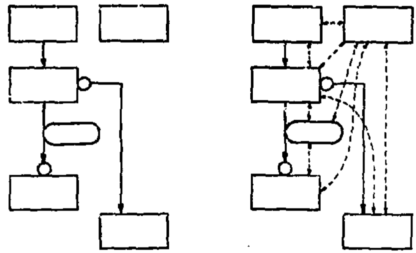

Figure 1 Material handling system block diagram notation: (a) as used in MHSS system models, and (b) showing communication paths among interacting functional blocks

THE MHSS PROGRAM

Organization

The MHSS program consists of two major subsystems: (1) an Executive Subsystem that provides dynamic storage management, event scheduling, means of communication among functional blocks, supervisory control of system models; and (2) a Modeling Subsystem that contains the functional block and utility subprograms used in material handling system simulation.

Executive Subsystem

Dynamic Storage Management MHSS does not reserve storage space for predetermined numbers of functional block types or material data types--an approach that, although straightforward, would require a large storage space, use it inefficiently, and inevitably require program modifications to tailor the reserved storage for specific cases. MHSS instead allocates storage at run time for all model dependent tasks, and when allocated space is no longer needed, returns it to a free-storage pool.

The block-chain data structure in Fig. 2 is used for dynamic storage management and throughout MHSS to utilize storage allocated in noncontiguous segments. Storage is first allocated in contiguous blocks from the beginning of the free space until the original block is exhausted and it is necessary to reallocate space linked to the end of the free store chain in the order that it becomes available.

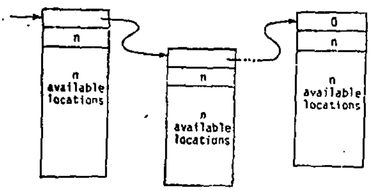

Figure 2 Block-chain data structure

* Work performed under the auspices of the U.S. Energy Research & Development Administration under contract No. W-7405-Eng-48.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

When temporary storage is needed, the free store chain is scanned and space is allocated from the first block that contains at least the required number of locations; the remainder of the block is left in the free store chain if it contains at least ten available locations--the size of the smallest storage fragment--otherwise the entire block is allocated. If no block in the free store chain is found that contains at least the required number of locations, a segment starting with the largest block found during the scan is cut from the chain and allocated; in this process, blocks are not split if fragments containing less than ten available locations would remain in the free store chain.

As small blocks accumulate toward the beginning of the free store chain, more and more time must be devoted to scanning the chain in search of large free blocks. The overhead time is reduced by aggregating free storage during the scan process. Each time it is necessary to scan the free store chain, one of the free blocks in the chain is checked for adjacency with the scanned blocks, and when two blocks are found to be adjacent, they are combined. The block checked for adjacency is incremented cyclically over the entire free store chain. Hence the time devoted to free store aggregation increases with the overhead time required to search for large free blocks.

Event Scheduling MHSS allocates a large block of contiguous locations within which a next-event chain and a free-event chain are maintained. Event notices containing
- a pointer to the next event notice,
- the time the event is to occur,
- the functional block number in which the event originates,
- an event code, and
- a pointer to additional data

are chained in increasing order of event times. Simulated time advances from event time to event time as the notices are removed from the next-event chain and executed by functional block subroutines. Event notice storage blocks are linked to the free-event chain as they are removed from the next-event chain. When an event is scheduled, the first available free-event block is used and the notice is linked into the next-event chain behind the notices with earlier event times. Events scheduled to occur at the same instant of simulated time are executed in the order scheduled. When the free-event chain is exhausted, a warning message is printed and a predetermined amount of additional space is allocated to extend the event scheduling facilities.

Communications Interaction of the executive and modeling subsystems and interactions among functional blocks of MHSS model systems are based on a simple communication data structure containing
- the address (integer block-number) of the sender,
- the address of the receiver,
- a communication type code, and
- a pointer to additional data.

The executive subsystem is assigned address zero and the MHSS user may assign block-number addresses from 1 to 100, an arbitrary maximum value determined by the space allocated for a table of block location pointers. Communication types:
- create block
- destroy block
- set parameters
- set variables
- control function
- set initial conditions
- dump block model
- event
- block interaction

are used to classify all MHSS communications and determine the highest-level structural organization of functional block subprograms.

Communications are also categorized as either *messages* or *responses*. Messages are transmitted by a sender to a receiver and must eventually be acknowledged by a response transmitted by the receiver to the sender. Several messages may be transmitted consecutively, however, before their responses are returned in reverse order. Response communications are identified by negating the communication type code of the original message.

Communications are transmitted by placing the communication data in common storage and transferring control to the recipient *block*. Since any number of instances of each functional block type may be simulated by each functional block subroutine, they must be reentrant. And since the modeling subsystem functional blocks operate, in general, at the same level of authority, the subroutines that simulate them must, in effect, be *coroutines* among which control is transferred freely during the execution of sequences of events. These features are provided in MHSS by the following scheme.

The sender and receiver addresses, communication type, and assigned statement label for resumption of subroutine execution upon receipt of the corresponding response communication are pushed onto a stack each time a communication is transmitted and popped off the stack each time a response is transmitted. The sender and receiver addresses and type of the current communication are checked for compliance with the aforementioned message/response rule and the assigned statement label is used to transfer control to the appropriate point in the subroutine.

The control structure of the MHSS program is illustrated in Fig. 3. The main program communicates with functional blocks A, B, and C before and between simulation runs in order to, for example, create or destroy blocks, modify values of block parameters or variable initial conditions, or to request reports of data collected during previous runs. To perform a simulation run, MAIN calls CLOCK, the MHSS time advance mechanism, which removes event notices from the next-event chain in temporal sequence and transmits event messages to the appropriate functional blocks. Many message/response communications may be transmitted among functional blocks affected by an event that originates in one particular block before an event response communication is returned to the executive subsystem block-number zero indicating that the event has been completed and that simulated time may be advanced to the time of the next event. All communications are transmitted by transfer of control through an executive subroutine SEND.

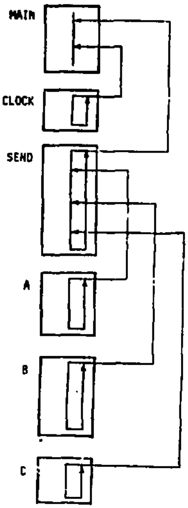

Figure 3 MHSS program control structure

Control of System Models The MISS user builds material handling system models and conducts simulation experiments using five types of commands summarized below.

CREATE *block-number* *block-type*
comment
parameter values
variable initial condition values

DESTROY *block-number* *block-type*
comment

PARAMETERS *block-number* *block-type*
comment
identifier *list-index* *type-code* *integer* *real*

VARIABLES *block-number* *block-type*
comment
identifier *list-index* *type-code* *integer* *real*

CONTROL *block-number* *function-code*
comment
data

RUN
comment
NREP *TSTOP* *IDUMP* *ITRACE*

END
comment

System models are built by CREATing each functional block, and the model is RUN for a time interval *TSTOP* over a set of *NREP* replications with a system model dump prior to the first run if *IDUMP* is 1 and with all communications printed if *ITRACE* is 1.

Simulation jobs may be stacked by using a logical string of commands. For example, a sequence of CREATE commands that build a model system and a RUN command that conducts a particular experiment may be followed by groups of commands that modify the structure and/or parameters of the model system and RUN the system again under the modified conditions. Blocks may be added to the previously defined system using CREATE commands or they may be removed by DESTROY commands. Parameter and initial condition values are modified by PARAMETER and VARIABLE commands respectively. CONTROL commands are used, for example, to print reports of data gathered during previous runs. Sets of commands that define sequences of MISS simulation jobs are terminated by an END command. Comments may be supplied with each command to annotate the results of simulation experiments.

Modeling Subsystems

MISS functional blocks are divided into eight categories identified by the first digit of the block-type number:

1xx	MONITOR	blocks that collect and report system performance data
2xx	CONTROL	blocks that simulate management and control functions of operators or automatic equipment
3xx	RECEIVING	blocks that accept material from outside the simulated system
4xx	SHIPPING	blocks that send material outside the simulated system
5xx	INVENTORY	blocks that determine quantities of materials present within the simulated system
6xx	TRANSPORT	blocks that simulate the conveyance of materials from one location to another
7xx	STORAGE	blocks that hold materials for extended periods of time
8xx	PROCESSING	blocks that modify the form or containment of materials

We next describe some blocks in detail to illustrate the functional specifications used in MISS modeling of material handling systems. In the initial development of the program, emphasis has been on blocks that simulate basic elements of nuclear material processing and safeguards systems, and the blocks described here—although they may be used in other applications—are not intended to provide a complete general-purpose repertoire. We plan to add more general-purpose blocks in the future, but feel the principal advantage of MISS is in the specialized functional correspondence of high-hierarchical-level elements of real and model systems. Block diagram symbols for the functional blocks described here are shown in Fig. 4.

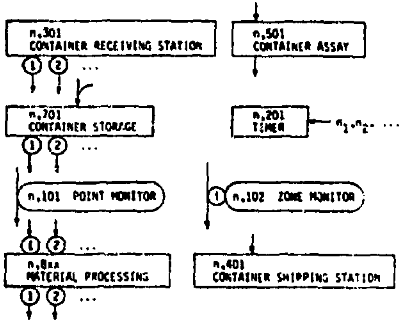

Figure 4 Some MISS block diagram symbols

Container Receiving Station 301 This functional block type simulates the arrival of batches of materials in discrete containers. The number of containers received is uniformly distributed with a specified mean and spread, and there may be any number of different types of material in the containers; the mass of each type is normally distributed with a specified mean, standard deviation, maximum, and minimum value. The materials are homogeneously mixed, and the containers are not tagged.

Batches of containers are scheduled to arrive at periodic delivery times, but they actually arrive at random times with uniformly distributed deviations from the scheduled arrival times. The receiving stations may be operated continually, i.e., accept deliveries whenever they arrive, or their hours of operation may be controlled by specified timer blocks. When operating under the control of a timer block, deliveries that arrive while a receiving station is closed are accepted at the beginning of the next shift or interval.

The receiving station blocks have no storage capacity so they must be connected to one or more blocks that will remove all of the containers at the time of arrival. If blocks are connected to more than one receiving station output terminal, they are notified of the material available for transfer in the numerical order of the output terminal numbers. If all of the containers are not removed by the block connected to the first output terminal, the block connected to the second output terminal is notified, and so on, until all of the containers have been removed or all of the output blocks have been notified that material is available. Priority of output blocks to material available at the receiving station is thus determined by the order of output terminal connections.

To CREATE Container Receiver Station 301 blocks the user supplies the following data:
- number of deliveries per period
- mean number of containers in batch deliveries
- spread of the number of containers in batch deliveries
- number of material types in the received containers
- number of output terminals
- control option: continual, during timer shifts or during timer intervals
- timer block-number
- period of the delivery schedule

spread of delivery time deviations from the scheduled times
list of output connections
list of scheduled delivery times relative to the beginning of each period

and for each material in the received containers, the
material type
mean mass
standard deviation of mass
maximum mass
minimum mass and
density.

These data are printed in input format so the user may check for modeling errors and verify that the data has been read correctly.

Each functional block in the model system is an instance of its functional block type which has a specific data structure. The general block data structure contains a block header at location LB, a parameter list at location LP, a variable list at location LV, and an initial condition list at location LI. For example, the Container Receiving Station 301 block data structure is shown below.

```
LB+ 0   block type = 301
  + 1   block size = 19
  + 2   block number
  + 3   number of parameters = 11
  + 4   number of variables = 2
LP+ 0   number of deliveries per period
  + 1   mean number of containers per batch
  + 2   spread of the number of containers per batch
  + 3   number of material types received
  + 4   number of output terminals
  + 5   control option code
  + 6   timer block-number
  + 7   period of the delivery schedule
  + 8   spread of delivery time deviations
  + 9   pointer to list of output terminal connections
  +10   pointer to list of scheduled delivery times
  +11   pointer to list of material specifications
LV+ 0   miscellaneous
  + 1   pointer to material data
LI+ 0   0
  + 1   0
```

Normally, the MSAS user will not be concerned with functional block data structure; but to aid in debugging new functional block subroutines and provide a complete model verification capability, the complete system model can be dumped in block data structure format just before simulation runs are started by setting IDUMP to 1 in RUN commands.

Error codes are printed by the executive subsystem and by functional block subroutines when illegal conditions are encountered. The Container Receiving Station 301 blocks, for example, in addition to routine error codes common to all functional blocks, print an error code and stop the program if the user specifies an illegal control option or if the output block(s) fail to remove all of the containers received.

<u>Container Storage 701</u> This functional block type simulates the storage of transportable material containers. When an operational storage block receives notice that containers are available, they are transferred into the block and placed on a container chain in the order received. Any number of containers may be stored and may be removed from storage in any order by blocks connected to the output terminals; these blocks are notified whenever the status of material available in storage is changed. When operating under the control of a timer block, material is transferred into storage only during the specified shifts or intervals.

An option may be invoked to determine the maximum number of containers stored, the first time the maximum is reached, the mean, and standard deviation of the number of containers stored. These data are reported upon CONTROL commands in the user input data or under the direction of report generator blocks.

To CREATE Container Storage 701 blocks, the user supplies the following data:
number of output terminals
control option: continual, during timer shifts or during timer intervals
timer block-number
control option: suppress data collection or collect performance data list of output connections.

A warning message is printed if a storage block receives a block interaction communication indicating that material is available but there are no transportable containers at the indicated material location.

<u>Container Assay 501</u> This functional block type simulates the nondestructive assay of specified material types in transportable containers. Using the actual mass of assayed materials in the containers, corresponding tag mass values are determined with simulated uncertainty characterized by statistics of absolute and relative components of measurement error. The error components are normally distributed, but the total error is truncated so as to not exceed a specified maximum value or produce a negative tag amount of material in any container.

Containers are assayed one-at-a-time and are transferred into the assay station from the beginning of the available container chain of the block connected to the input terminal. Container assay times are uniformly distributed with specified mean and spread. When operating under the control of a timer block, containers are transferred into the block for assay only if the time remaining in the current shift or interval exceeds the mean measurement time. If the assay station is operated continually, available containers are transferred into the block at any time the previously assayed container is removed. Measurements are halted if the block connected to the output terminal does not remove an assayed container. These blocks normally receive containers from and deliver containers to storage blocks.

To CREATE Container Assay 501 blocks, the user supplies the following data:
control option: continual, during timer shifts or during timer intervals
timer block-number
output connection
number of material types assayed
mean measurement time
spread of measurement time

and for each material type assayed, the
material type
mean absolute measurement error
standard deviation of absolute measurement error
mean relative measurement error
standard deviation of relative measurement error and
maximum total measurement error.

A warning message is printed if a container is in the block at the time a start-measurement event was scheduled to occur; and errors are printed if the user specifies an illegal control option, a finish-measurement event notice is received with no container in the block, or the material type to be assayed is not defined in a container data structure.

<u>Timer 201</u> This functional block type schedules periodic shifts and intervals of operation for any number of other blocks that refer to the timer blocks to obtain the
time the current period will end
current shift
time the current shift will end
time the next shift will begin
current interval
time the current interval will end and the
time the next interval will begin.

Any number of timer blocks may be used to establish different operating schedules for subsets of system functional blocks. The n_1, n_2, ... in the timer block diagram symbol indicate the blocks that refer to a particular timer block.

To CREATE Timer 201 blocks, the user supplies the following data:
timer period
number of shifts per period
number of intervals per period
option: suppress messages; print messages at schedule times
list of shift starting times
list of shift ending times
list of interval starting times
list of interval ending times

<u>Point Monitor 101</u> This functional block type is used to observe material flow at specific points in material handling systems. Any number of material types may be monitored either continually or under the control of timer blocks.

Tag, actual, or error (tag - actual) values of material count, mass, or volume may be tabulated each time a discrete amount of material is transferred through the monitor point, or the cumulative values of these quantities may be tabulated at a specified time in replicated simulation runs. Point Monitor 101 blocks are not used to monitor continuous material flow.

The mean and standard deviation and histogram of tabulated values are printed in point monitor report format upon CONTROL command or under the control of report generator blocks.

Point monitor blocks have one input and one output terminal. Any number of blocks may be connected to the input if they are all compatible with the block type connected to the output.

To define Point Monitor 101 blocks, the user supplies the following data.

- output connection
- number of material monitoring specifications

and for each monitored material, the

- material type
- option: tag, actual, error
- option: count, mass, volume
- option: continual, during timer shifts, during timer intervals or at replicated run times
- timer block-number
- number of histogram cells
- width of histogram cells
- lower limit of first histogram cell
- sampling time in replicate runs

<u>Zone Monitor 102</u> This functional block type is used to observe the quantity of specified materials within bounded regions (zones) of material handling systems. A zone may represent a connected region such as room or entire building, or it may represent a disjoint region such as a set of storage areas that are distributed throughout a material handling system.

Zone monitor specifications are similar to those of point monitor blocks, however, a zone monitor must observe all material entering or leaving the zone and must therefore have, in effect, point monitors at every place materials are transferred across the zone boundary. Terminal numbers, 1, are used in the zone monitor block diagram symbol to distinguish points of material entry and exit, and several symbols are generally required to represent a single zone monitor functional block.

<u>Report Generator 103</u> This functional block type reads annotations and abbreviated report request data from the user input stream and issues the appropriate CONTROL commands for printing of functional block reports.

<u>Container Shipping Station 401</u> This functional block type simulates the shipment of material in discrete containers from a storage block connected to its input terminal. A schedule of shipping times for prescribed numbers of containers (subject to random variations) is specified in a manner similar to that for Container Receiving Station 301 blocks described previously.

<u>Transfer Control 601</u> This functional block type controls the transfer of material from one block output terminal to two or more input terminals. The block operates in either of two modes: In priority mode, blocks are notified of material available in the order of their connection to the block output terminals. In random mode, the blocks connected to the output terminals are notified with a specified discrete probability distribution.

<u>Bulk Storage 202</u> This functional block type simulates one or more stationary storage vessels. Available material is accepted when any of the simulated vessels is below a specified capacity. Materials are homogeneously mixed within each vessel, and are available only in the current mixture ratio. Incoming material is placed in one vessel and outgoing material is removed from the same vessel until it is either filled or emptied.

<u>Material Processing Bxx</u> Several functional block types planned for simulation of nuclear fuel manufacturing processes are represented by the general material processing block diagram symbol in Fig. 4. These processes include

- unloading of containers into bulk storage vessels
- bulk storage, analysis, and feed hopper loading
- batch blending of mixed-oxide powders
- powder compaction
- pellet sintering
- pellet grinding
- pellet inspection
- fuel rod loading
- fuel rod inspection
- fuel rod dismantling
- pellet crushing
- clean scrap recovery and
- waste treatment.

All of these processes can be characterized by a few functional block types with provisions for simulation of

- selective material transfer from input terminals
- generation of scrap
- generation of waste
- accumulation (holdup) of material within the process
- sampling of material for analytical services
- delay and transformation of product material form or containment and
- selective material transfer to output terminals.

<u>Utility Subroutines</u>

MHSS functional block subroutines are simplified by subprograms that perform the following operations:

- create functional block instance
- dump functional block model
- print error or warning message
- locate in block-chain data structure
- determine the amount of a material type available at a material data location
- prepare notice of material available
- generate random variable with specified distribution
- schedule event
- set parameter value
- set variable value and
- transfer all or a specified number of containers.

<u>Structured FORTRAN Programming Conventions</u>

The MHSS program is written in FORTRAN using simple translations of the structured programming constructs

- <u>if</u> <u>then</u>
- <u>if</u> <u>then</u> <u>else</u>
- <u>case</u>
- <u>while (until)</u> <u>do</u>
- <u>do</u> <u>while (until)</u> and
- <u>for</u> <u>do</u>

<u>POTENTIAL APPLICATIONS TO NUCLEAR MATERIAL SAFEGUARDS</u>

An expanding nuclear industry, with its associated increased quantities of nuclear fuel, has generated concern for the safeguards status of Special Nuclear Material (SNM) within a processing facility. Various safeguards measures now under consideration, especially inventory control, can benefit from the use of a MHSS program to assess the effectiveness of a proposed safeguards system. Additionally, this program can be used to perform sensitivity studies for diverse instrument deployment strategies, thereby, arriving at a cost-effective configuration.

Aside from its utility in safeguards analysis, a MHSS program can operate as a means of evaluating the functional requirements of the processing elements themselves. For example, it could be used in scheduling material flow and in pinpointing bottlenecks in the process, thus indicating the need for additional parallel processing units. Alternately, it could also locate inefficiently used equipment.

In summary, when confronted with a complex interrelated problem, simulation can be an effective technique for performance appraisal of nuclear material processing facilities.

REFERENCES

1. Kenneth E. Sauders, "Simulation of Nuclear Fuel Manufacture as a Safeguards Tool", Proceedings of the 1975 Summer Computer Simulation Conference, pp. 1398-1405, July 21-23, 1975.

2. Charles C. Pomernacki, et.al., "Modeling, Simulation, and Estimation for Real-Time Materials Control", INMM 15th Annual Meeting, June 19-21, 1974.

3. Russel H. Sanborn, "MO200: A Model for Evaluating Safeguarding Through Material Accountability in a 200 Tonne Per Year Mixed-Oxide Fuel Rod Fabrication Plant", Proceedings of the 1976 Summer Computer Simulation Conference, July 12-14, 1976.

4. A. A. B. Pritsker, The GASP IV Simulation Language, John Wiley & Sons, New York, 1974.

5. A. A. B. Pritsker, The Q-GERIS User's Manual, Pritsker and Associates, Inc., 1201 Wiley Drive, West Lafayette, Indiana, 197[illegible].