

INTRODUCTION TO SIMAN

C. Dennis Pegden
Associate Professor
Department of Industrial and Management Systems Engineering
207 Hammond Building
The Pennsylvania State University
University Park, PA 16802

This paper discusses the concepts and methods for simulating manufacturing systems using the SIMAN simulation language. SIMAN is a new general purpose simulation language which incorporates special purpose features for modeling manufacturing systems. These special purpose features greatly simplify and enhance the modeling of the material handling component of a manufacturing system.

1. INTRODUCTION

This paper discusses the use of the SIMAN simulation language for modeling manufacturing systems [5]. SIMAN is a new general purpose SIMulation ANalysis program for modeling combined discrete-continuous systems. The modeling framework of SIMAN allows component models based on three distinct modeling orientations to be combined in a single system model. For discrete change systems either a process or event orientation can be used to describe the model. Continuous change systems are modeled with algebraic, difference, or differential equations. A combination of these orientations can be used to model combined discrete-continuous models.

With the growing number of simulation languages available to the practitioner interested in manufacturing systems, the reader may be questioning the need to add one to their number. SIMAN differs from existing simulation languages in four important ways.

- 1) SIMAN is designed around a logical modeling framework in which the simulation program is decomposed into a model frame and an experiment frame.
- 2) SIMAN incorporates a number of unique and powerful general purpose modeling constructs which represents a natural evaluation and refinement of existing language designs.
- 3) SIMAN imbeds within this general purpose framework a set of special purpose constructs which are specifically designed to simplify and enhance the modeling of manufacturing systems. Existing general purpose languages such as GPSS [6] and SLAM [4] lack the special purpose manufacturing features provided by SIMAN. On the other hand, existing special purpose manufacturing languages such as GALS [1]

and SPEED [3] are intended for a restricted class of manufacturing systems and are not applicable to systems in general.

4) SIMAN runs on mainframe, mini, and 16 bit microcomputers. All versions, including the microcomputer versions, are completely compatible. Models can be moved between computer systems without modification.

2. THE SYSTEM MODELING FRAMEWORK

The SIMAN modeling framework is based on the system theoretic concepts developed by Zeigler [7]. Within this framework, a fundamental distinction is stressed between the system model and the experimental frame. The system model defines the static and dynamic characteristics of the system. The experimental frame defines the experimental conditions under which the model is run to generate specific output data. For a given model, there can be many experimental frames resulting in many sets of output data. By separating the model structure and the experimental frame into two distinct elements, different simulations experiments can be performed by changing only the experimental frame. The system model remains the same.

Given the system model and the experimental frames, the SIMAN simulation program generates output files which record the model state transitions as they occur in simulated time. The data in the output files can then be subjected to various data analyses such as data truncation and compression, and the formatting and display of histograms, plots, tables, etc. Within the SIMAN framework, the data analysis and display function follow the development and running of the simulation program and are completely distinct from it. One output file can be subjected to many different data treatments without re-executing the simulation program. Data treatments can also be applied to sets of output

files - this is useful when performing an analysis based on multiple runs of a model or when comparing the system response of two or more models.

The SIMAN software package divides the simulation process into three distinct activities: system model development, experimental frame development, and data analysis. As shown in Figure 1, the SIMAN software consists of five individual processors which interact through four data files. The model processor is used to construct block diagram component models. The data file defining the block diagram generated by the processor is referred to as the model file. The experiment processor is used to define the experimental frame for the system model. The data file defining the experimental frame is referred to as the experiment file. The link processor is used to combine the model file and experiment file to produce the program file. The program file is input to the run processor which executes the simulation runs and writes the results on the output files. If an event or continuous component model is included in the system model, the user-written FORTRAN subroutines are linked to the run processor before the simulation runs are executed. The output processor is used to analyze, format, and display the data contained in the output files.

The five independent processors within the SIMAN software simplify the framework by separating the distinct functional activities of simulation. This breakdown is also practical, since only one of the five processors is executing at one time, the computer memory requirements are substantially reduced. In addition, you can independently create and save the model files, experiment files, program files, and output files on disc.

3. THE MODELING ORIENTATIONS

The primary modeling orientation for discrete change systems is the process orientation, in which the model is constructed by depicting the functional operations of the systems as a block diagram. The block diagram is a linear top-down sequence of blocks which represents specific process functions such as time delays and queues.

A second modeling orientation for discrete change systems, the event orientation, may be used to augment or replace the block diagram component. The event component consists of a set of user-written FORTRAN subroutines which contain the mathematical-logical expressions that define the instantaneous state transitions occurring at each event time. The event subroutines typically use calls to subprograms contained in the SIMAN Subprogram Library. These subprograms perform standard event modeling functions such as file manipulating, event scheduling, observation recording, etc. SIMAN controls the simulation by advancing time and making calls to the appropriate event subroutines at the correct simulated time.

In a continuous simulation model, the state of the system is represented by dependent variables which change continuously over time. Examples of continuous change variables include the temperature of an ingot in a soaking pit furnace, or the concentration of a reactant within a chemical process.

To distinguish continuous change variables from discrete change variables, the former are referred to as state variables. A continuous simulation model is constructed by defining equations for a set of state variables whose dynamic response simulates the real system.

A continuous system is modeled in SIMAN by coding the algebraic, difference, and differential equations which define the state variables in FORTRAN within subroutine STATE. Subroutine STATE is automatically called by SIMAN at small time intervals called steps to compute the response of the system over time. The value of the Ith state variable is maintained as variable S(I), and the derivative with respect to time of the Ith state variable is maintained as variable D(I). The values of these variables at the end of the last step are assigned to the variables SL(I) and DL(I), respectively. The current step size is denoted by the variable DTNOW and is automatically controlled by SIMAN based on accuracy parameters specified within the experimental frame. When differential equations are included within subroutine STATE, they are automatically integrated by SIMAN using the Runge-Kutta-Fehlberg integration algorithm to obtain the values of the state variables within an accuracy specified by the modeler.

Although SIMAN permits component models to be developed using a process, event, or continuous orientation, we will focus our attention in this paper on the process orientation in which models are constructed as block diagrams. This orientation is the one best suited for modeling most manufacturing systems.

4. BLOCK DIAGRAM MODELS

Block diagrams are the primary means by which discrete systems are modeled in SIMAN. These diagrams are linear topdown flowgraphs which depict the flow of entities through the system. The block diagram is constructed as a sequence of blocks whose shapes indicate their function. The sequencing of blocks is depicted by arrows which control the flow of entities from block to block through the entire diagram.

These entities are used to represent "things" such as workpieces, information, people, etc., which flow through the real system. Each entity may be individualized by assigning attributes to describe or characterize it. For example, an entity representing a workpiece might have attributes corresponding to due date and processing time for the workpiece. As the entities flow from block to block, they may be delayed, destroyed, combined with other entities, etc., as determined by the function of each block.

The attributes of an entity are denoted by the real array A(.) and the integer variable M. The elements of the A array are used to represent the general attributes assigned to the entity. In the example in the previous paragraph, A(1) could be used to record the due date and A(2) the job processing time. The integer attribute M is the current station number of the entity. The station number defines a physical location in the system such as the location of a workcenter or storage area. The array A and the variable M are "carried

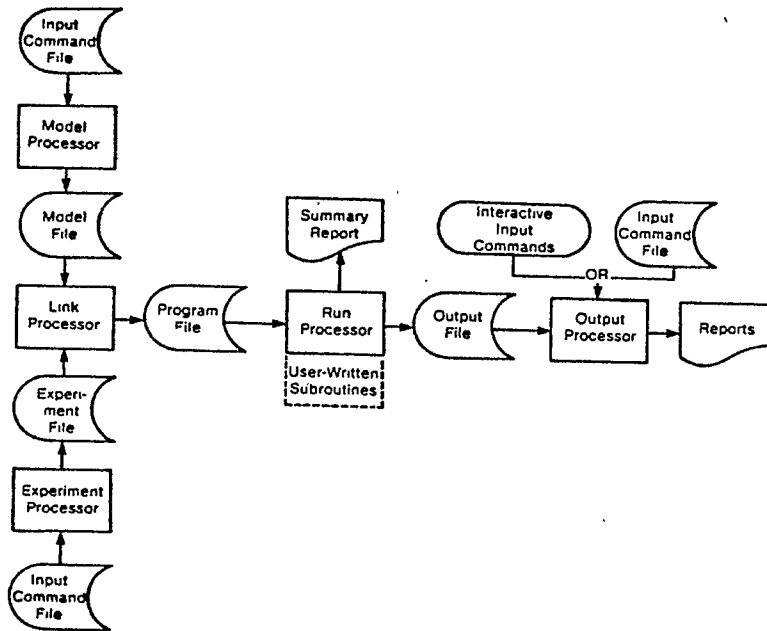


Figure 1: SIMAN Software Organization

along" with an entity as it moves from block to block within a model.

There are ten different basic block types in SIMAN. The symbol and functions for each of the ten block types are summarized in Table 1.

The OPERATION, HOLD and TRANSFER blocks are further subdivided into several different block functions depending upon their operation type, hold type or transfer type. These types are specified as the first operand of the block, and consist of a verb which is descriptive of the specific function which the block is to perform. For example, the operation type CREATE specifies that the block is to create entities; the operation type ASSIGN specifies that the block is to assign a value to an attribute or variable; and the operation type DELAY specifies that the block is to delay entities. A summary of the various operation types, hold types and transfer types included in SIMAN is given in Tables 2, 3 and 4.

Each block function of SIMAN is referenced by a block function name. In the case of the QUEUE, STATION, BRANCH, PICKO, OPICK, SELECT and MATCH blocks, each block type performs only one function and the block function name is the same as the basic block name. However, in the case of the OPERATION, HOLD and TRANSFER blocks, each basic block type performs several different functions. The block function name for each of these blocks is the operation type, hold type or transfer type specified as the first operand of the block. We will frequently refer to a block by its function name - for example, we will refer to the OPERATION block with the DELAY operation type as the DELAY block.



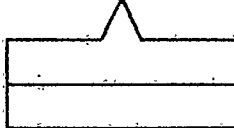
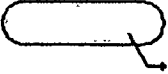

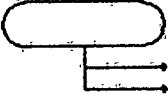
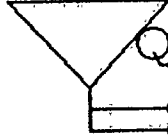
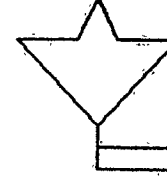
All of the basic block types, including the OPERATION, HOLD, and TRANSFER types, have operands which control the function of the block. For example, the CREATE block has operands which prescribe the time between batch arrivals, the number of entities per batch, and the maximum number of batches to create.

Blocks may optionally be assigned a block label, one or more block modifiers, and a comment line. A block label is appended to the lower left side of a block and can consist of up to eight alphanumeric characters. A block label is used for branching or referencing from other blocks. Block modifiers are special symbols appended to the right or bottom of a block and either modify or extend the standard function to be performed by the block. The comment line, if specified, is entered to the right of the block and serves to document the model.

A block diagram model can be defined in either of two equivalent forms referred to as the diagram model and the statement model. The diagram model is a graphic representation of the system using the ten basic block symbols in Table 1. The statement model is a transcription of the diagram model into statement form for input to the model processor. There is a one-to-one correspondence between blocks in the diagram model and input statements in the statement model. Once a model has been entered into the model processor and a model file has been created, it can then be displayed on a terminal in either the statement or graphic form.

The general block input statement is divided into five sections consisting of the block label (if any), the block function name, the block modifiers (if any), and the comment line (if any). The

Table 1: Basic Block Types

Name	Symbol	Function
OPERATION		The OPERATION block is used to model a wide range of processes such as time delays, attribute assignments, etc. Also see Table 1-2.
TRANSFER		The TRANSFER block is used to model transfers between stations via material handling systems. Also see Table 1-4.
HOLD		The HOLD block is used to model situations in which the movement of an entity is delayed based on system status. The HOLD block must be preceded by a queueing facility to provide a waiting space for delayed entities. Also see Table 1-3.
QUEUE		The QUEUE block provides a waiting space for entities which are delayed at following HOLD or MATCH blocks.
STATION		The STATION block defines the interface points between model segments and the material handling systems.
BRANCH		The BRANCH block models the conditional, probabilistic and deterministic branching of entities.
PICKQ		The PICKQ block is used to select from a set of following QUEUE blocks.
SELECT		The SELECT block is used to select between resources associated with a set of following OPERATION blocks.

(continued on next page)

(Continued from previous page)

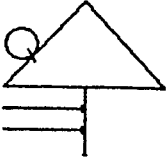
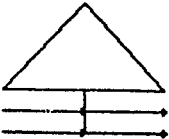
Name	Symbol	Function
QPICK		The QPICK block is used to select from a set of preceding QUEUE blocks.
MATCH		The MATCH block delays entities in a set of preceding QUEUE blocks until entities with the same value of a specified attribute resides in each QUEUE.

Table 2: Operation Types

	Name	Description
<u>General Functions</u>	ASSIGN	Assign values to attributes and variables.
	CREATE	Create batch arrivals to the system.
	DELAY	Delay an entity by a specified time.
	DETECT	Detect state events associated with continuous variables.
	EVENT	Cause a specified event to occur.
	FINDJ	Find the value of the index J meeting a specified condition.
	SIGNAL	Send a signal to end the delay for an entity.
	SPLIT	Split a group into its individual members.
<u>Resource Functions</u>	ALTER	Change the capacity of a specified resource.
	RELEASE	Release units of a specified resource.
<u>Material Handling Functions</u>	ACTIVATE	Set the status of a specified transporter to active.
	EXIT	Exit a specified conveyor device.
	FREE	Exit a specified transporter device.
	HALT	Set the status of a specified transporter to inactive.
	START	Set the status of a specified conveyor to active.
	STOP	Set the status of a specified conveyor to inactive.

(continued on next page)

(continued from previous page)

<u>File Functions</u>	COPY	Copy the attributes of an entity at a specified QUEUE.
	REMOVE	Remove an entity from a specified QUEUE.
	SEARCH	Search a QUEUE for an entity meeting a specified condition.
<u>Statistics Functions</u>	COUNT	Increment a specified counter.
	TALLY	Record an observation of a specified value.

Table 3: Hold Types

	Name	Description
<u>Condition Functions</u>	SCAN	Hold the entity until a specified condition is met.
	WAIT	Hold the entity until a specified signal is received from a SIGNAL block.
<u>Resource Functions</u>	SEIZE	Hold the entity until the required number of units of a resource are idle and are allocated to the entity.
	PREEMPT	Hold the entity until one resource unit is allocated to the entity. The entity may preempt a resource currently being used.
<u>Material Handling Functions</u>	ACCESS	Hold the entity until a specified number of consecutive conveyor cells are available and allocated to the entity at the accessing station location.
	REQUEST	Hold the entity until a transporter device is allocated to the entity and arrives to the requesting station location.
<u>Set Functions</u>	COMBINE	Hold the entity until a specified number of entities reside in the preceding QUEUE block. When this occurs, the waiting entities are combined into a permanent set and a representative of the set is created. The original entities in the set are destroyed.
	GROUP	Hold the entity until a specified number of entities reside in the preceding QUEUE block. When this occurs, the entities are grouped into a temporary set and a representative of the set is created. The original entities in the set are retained and can be recovered using the SPLIT block.

Table 4: Transfer Types

	Name	Description
<u>Material Handling Functions</u>	CONVEY	Convey the entity to a specified station via a conveyor. The transmit time is determined by the distance between the station along the conveyor and the speed of the conveyor.
	ROUTE	Route the entity to a specified station. The transmit time is specified as an operand of the block.
	TRANSPORT	Transport the entity to a specified station via a transporter. The transmit time is proportional to the distance between stations.

block label is entered anywhere within the first eight columns of the input record with the remaining description starting in column ten or after. The operands are divided into line segments which are ended with a colon, where each line segment corresponds to a given line of operands within the block. An operand may be defaulted by simply omitting the entry from the line segment. The end of all operand lines and modifiers is indicated by a semicolon, which is followed by the comment line.

The general format for a block diagram model and corresponding statement model is illustrated in Figure 2 and Figure 3, respectively. This model corresponds to a TV inspection and adjustment process [6]. In this model, the vertical control setting of TV sets is tested at an inspection station. If a set is found to be functioning improperly (which occurs 15% of the time), it is sent to an adjustment station. After adjustment, it is sent back to the inspection station where the setting is again inspected. TV sets passing inspection, whether for the first time or after one or more adjustments, are sent to a packing area. The diagram model shown in Figure 2 for this example was drawn by SIMAN on a Tektronix 4025 terminal from the input statements shown in Figure 3.

A detailed description of this example will not be attempted. It is included here only to illustrate the general format conventions.

5. MODELING WORKSTATIONS

In manufacturing systems it is frequently desirable to model distinct workstations within the system. This can be done by employing the STATION block which defines the beginning of a station submodel. An entity is entered into the station submodel by sending the entity to the STATION block using a TRANSFER block. The TRANSFER block is used to represent entity movements between station submodels.

Each station submodel is referenced by a positive integer which is called the station number. This number corresponds to a physical location within the system. The station number is an operand of both the STATION block and the TRANSFER block.

When an entity enters a STATION block, the entity's station attribute, M, is set by SIMAN to

the station number of the STATION block. The entity carries this station attribute with it as it proceeds through the sequence of blocks which comprises the station submodel. The entity remains within the station submodel until it is disposed, or it is sent to a new station submodel via a TRANSFER block.

The block sequence within a station submodel defines the process which the entities at that workstation encounter. These processes normally involve the queueing of entities due to limited resources such as operators, tools, or machines. These queueing processes are modeled in SIMAN using the concept of a resource, which denotes a general class of non-positioned "units" which may be allocated to entities. The resource capacity is the number of units of the resource and is defined in the experimental frame. When a resource unit is allocated to an entity, its status is changed from idle to busy. When an entity releases a resource unit, the status of the resource is then changed from busy back to idle. A busy resource unit may not be reallocated to entities.

Resources may be allocated to entities using the HOLD block with the SEIZE hold type. The operands for the SEIZE block include the resource name, the number of units required, and the priority which is used to allocate limited resource units between competing SEIZE blocks. The SEIZE block is preceded by a QUEUE block which defines a file in which entities reside while waiting for resource units. When an entity arrives to a QUEUE-SEIZE block combination, it attempts to seize the required number of resource units. If the resource units are available, they are allocated to the entity and the entity exits the SEIZE block. However, if not all of the units are available, the entity is placed in the file of the QUEUE block where it waits to be allocated the required number of resource units.

Resource units may be released from an entity using the OPERATION block with the RELEASE operation type. The operands for the RELEASE block included the resource name and the number of resource units to be released. When an entity arrives to the RELEASE block, the status of the specified number or resource units is changed from busy to idle. The idle resource units are then available for reallocation to awaiting entities at SEIZE blocks.

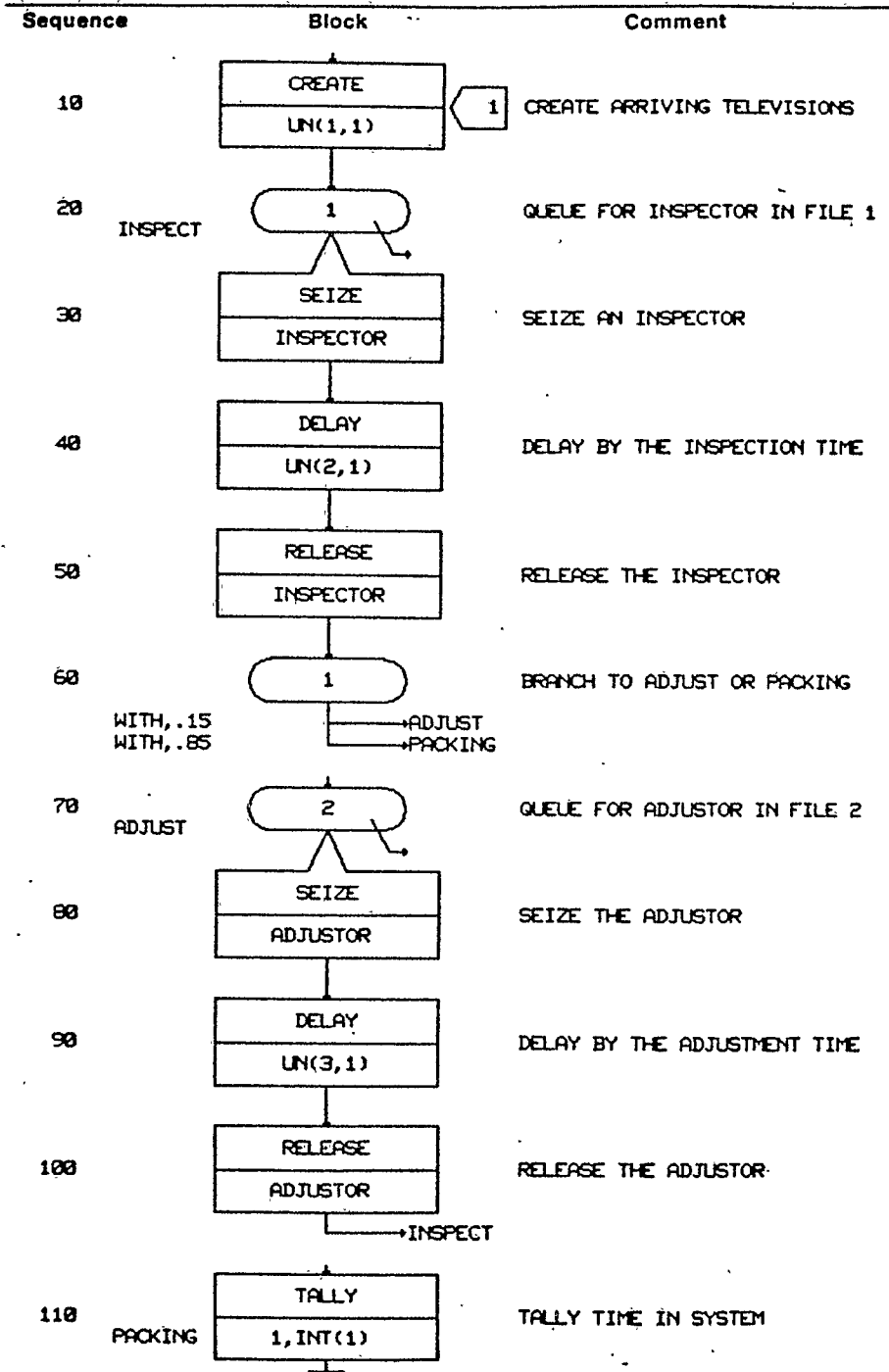


Figure 2: Block Diagram for TV Inspection and Adjustment Example


```

BEGIN;
10      CREATE:UN(1,1):MARK(1);
20 INSPECT  QUEUE,1;
30      SEIZE:INSPECTOR;
40      DELAY:UN(2,1);
50      RELEASE:INSPECTOR;
60      BRANCH,1:
        WITH,.15,ADJUST;
        WITH,.85,PACKING;
70 ADJUST  QUEUE,2;
80      SEIZE:ADJUSTOR;
90      DELAY:UN(3,1);
100     RELEASE:ADJUSTOR:NEXT(INSPECT);
110 PACKING TALLY:1,INT(1):DISPOSE;
END;
        CREATE ARRIVING TELEVISIONS
        QUEUE FOR INSPECTOR IN FILE 1
        SEIZE AN INSPECTOR
        DELAY BY THE INSPECTION TIME
        RELEASE THE INSPECTOR
        BRANCH TO ADJUST OR PACKING
        QUEUE FOR ADJUSTOR IN FILE 2
        SEIZE THE ADJUSTOR
        DELAY BY THE ADJUSTMENT TIME
        RELEASE THE ADJUSTOR
        TALLY TIME IN SYSTEM

```

Figure 3: Statement Model for TV Inspection and Adjustment Example

The duration that an entity employs a resource is typically modeled using an OPERATION block with the DELAY operation type. The DELAY block causes each arriving entity to be delayed by a duration specified as an operand of the block. This duration can be specified as a constant, random variable, system status variable, attribute, or an expression involving any of the above.

To illustrate the concept of a workstation submodel employing the concept of a resource, consider the block diagram submodel shown in Figure 4. This block diagram contains the frequently occurring sequence QUEUE-SEIZE-DELAY-RELEASE which can be used to model both a single-server and multi-server queueing system, depending on the capacity for the resource which is specified in the experimental frame.

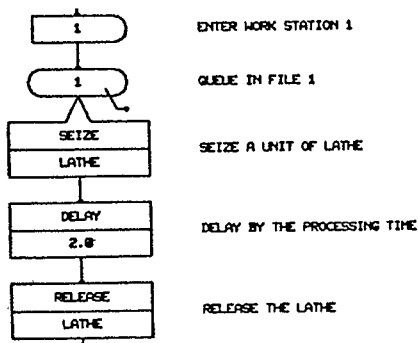


Figure 4: Station Submodel Example

In this example, each entity which enters station number 1 proceeds to the QUEUE-SEIZE block combination where it attempts to seize one unit of the resource LATHE. If there is not an idle unit of LATHE, the entity is placed in file number 1 where it waits for an idle unit. Otherwise, the entity seizes a unit of LATHE and proceeds to the DELAY block where it is delayed by 2.0 time units. This delay represents the time required for the entity to complete processing on the lathe. The entity then exits the DELAY block and enters the RELEASE block where it releases one unit of LATHE.

The symbol following the RELEASE block denotes that the entity is to be destroyed upon existing the RELEASE block.

The block sequence in this example is particularly simple and employs only a small subset of the general purpose features of SIMAN. Once the analyst becomes familiar with the wide range of block functions included in SIMAN, complex workstations can be modeled with similar ease.

6. MACRO SUBMODELS

One particular useful feature for modeling workstations in SIMAN is the macro submodel. This powerful feature permits the development of a single macro submodel to represent a set of two or more similar yet distinct workstations. For example, a typical jobshop consists of several different workstations (lathes, planners, etc.) which are functionally equivalent, and differ only in their number and type of machines, buffer sizes, etc. We can model such a jobshop by constructing a single macro submodel which represents the process encountered by a job at a general jobshop workstation. This single macro submodel can then be used to model a jobshop of arbitrary size.

The beginning of a macro submodel is defined by a STATION block. The range of station numbers represented by the macro submodel is specified as the operand of the block. An entity is entered into the macro submodel by sending it to the STATION block using a TRANSFER block. All entities sent to a station number in the specified range of the STATION block are processed as arrivals to the block. Upon entering the STATION block the macro station attribute M of the entity is set by SIMAN to the station number to which the entity was sent.

When a macro submodel is employed, the station attribute is typically incorporated in the operands of one or more of the blocks which follow the STATION block. In this way the operation of the macro submodel can depend upon the station number of the entity. For example, the file number at a QUEUE block can be specified as a function of M.

The station attribute can also be used to specify a resource to be seized or released through the use of indexed resources. An indexed resource allows a single name to be assigned to a set of two or more different resource types, with each resource in the

set having its own capacity. The resource types within the set are distinguished by an index appended to the resource name. For example MACHINE (1) and MACHINE (2) represents two distinct resources with different capacities. These resources are completely independent other than sharing the common name MACHINE.

To illustrate the use of indexed resources within a macro submodel, consider the block diagram shown in Figure 5. When an entity arrives to the STATION block, the station attribute M of the entity is set to its current station number. Since this macro models stations 1 through 10, M in this case will be between 1 and 10. The entities proceeds to the QUEUE-SEIZE combination where they attempt to seize a unit of MACHINE (M). If all units of the resource are busy, the entity is placed into file number M where it waits for an idle unit of MACHINE (M). Once a unit is seized, the entity continues to the DELAY block where it is delayed by a uniformly distributed random time. The entity then arrives to the RELEASE block where one unit of MACHINE (M) is released. The remaining blocks of the model are not shown.

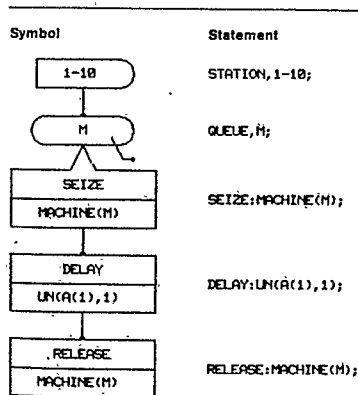


Figure 5: Macro Submodel Example

MODELING MATERIAL HANDLING SYSTEMS

Within a manufacturing system, the movement of entities between workstations is accomplished by the material handling system. This is an extremely critical function in most manufacturing systems. Apple [2] notes that the material handling function can easily account for 50 to 70% of the production activity.

In simple terms, the function of material handling is the movement of material from one point to another. There is a large variety of material handling devices which have been developed to support this function. Within the material handling literature, these devices are frequently categorized into the following equipment classes.

1. Industrial Trucks - hand or powered vehicles used for intermittently moving

items by maneuvering across a fixed surface. Common examples are forklift trucks, hand carts, and platform trucks.

2. Cranes, Hoist, and Manipulators - mechanical devices used for intermittently moving items through space. Common examples are overhead cranes, jib cranes, industrial robots, hoists, and monorail devices.
3. Conveyors - gravity or powered devices used in moving items continuously and simultaneously along a fixed path. Common examples are belt conveyors, bucket conveyors, hook conveyors, and trolley conveyors.

From a systems modeling perspective, the devices in the above three equipment categories all perform one of two basic movement functions. The first function we will call the transport function and corresponds to the intermittent movement of items, one load at a time, along a fixed or varied path. The term load unit as applied here could denote a box, a roll of material, or a pallet containing a number of items grouped together. The transport function is performed by devices in the first two equipment categories. Devices in the first equipment category perform the transport function along the ground, whereas the devices in the second equipment category perform the transport function above the ground. We will refer to the devices in either of these two equipment categories as transporters.

The second basic movement function we will call the convey function and corresponds to the continuous and simultaneous movement of items along a fixed path. The convey function is performed by the devices listed in the third equipment category. We will refer to devices in this category as conveyors.

The categorization of material handling equipment into the two basic movement functions of transport and convey provides the basis for modeling these devices in SIMAN. Special blocks and experimental elements are included in SIMAN which allow both of these basic movement functions to be modeled in a straight-forward manner.

The block functions which are used to model material handling systems employ the concept of a station submodel as discussed earlier. All movement functions are made relative to station numbers assigned to each station submodel. A material handling system is represented in SIMAN as a component of the system model which models the utilization of material handling devices to move from one station submodel to another. The travel time for entities between stations is based on the speed of the material handling device and the spatial relationship of the origin and destination stations relative to the device. Both of these are specified by the modeler in the experimental frame.

The generic term transporter is used in SIMAN to denote a general class of movable devices which may be allocated to entities. Each transporter device has a specific station location in the system and requires time to travel from one station to another.

Examples of devices which might be modeled as transporters are carts, cranes, and mechanical manipulators.

The characteristics of each transporter type are specified in the experimental frame and include the transporter name, capacity, distance set number, and the initial station position and operational status of each of the transporter units for that type. The transporter name is an arbitrary alphanumeric name assigned by the modeler to each transporter type. The transporter capacity is the number of independent movable units of that transporter type. The distance set number is a cross-reference to a matrix containing the travel distances between all pairs of stations which each transporter unit of that type may visit. This matrix is specified in the experimental frame.

Transporter units are allocated to entities at HOLD blocks using the REQUEST hold type. The operands of the REQUEST block are the request priority and the name and index of the transporter unit requested. The request priority is an integer priority number which is used to allocate an idle transporter unit when multiple requests compete for the same transporter unit.

Once an entity has been allocated a transporter unit at a REQUEST block, the entity can be transported from one station to the next using the TRANSFER block with the TRANSPORT hold type. The operands of the TRANSFER block are the transporter unit and the station number of the destination station. The duration of the transport is automatically computed by SIMAN based on the distance to the station and the speed of the transporter unit. At the end of the transport duration, the entity enters the STATION block of the destination station submodel.

A transporter unit may be released using the OPERATION block with the FREE operation type. The FREE block changes the status of the specified transporter from busy to idle.

The generic term conveyor is used in SIMAN to denote a general class of devices which consist of positioned cells which are linked together and move in unison. Each cell represents a location on the device and can be either empty or occupied. Entities which access the conveyor must wait at the entering station until the specified number of consecutive empty cells are located at the station. The entity then enters the conveyor and the status of the cells are changed from empty to occupied. The entity remains in the cells until the conveyor is exited at the entities' destination station.

The representation of conveyor devices as linked cells which are either empty or occupied imposes the restriction that items can only enter the device at discrete points along the conveyor. This is representative of discrete spaced conveyor devices such as bucket, cable, and magnetic conveyors for which items can enter the device only at fixed positions along the conveyor. However, other devices, such as belts, permit continuous spacing of items along the device. These devices can be reasonably approximated by defining a small spacing between each cell. The number of

cells accessed would be specified as the length of the item in cell widths.

Each conveyor device moves along a fixed path defined by one or more segments. A segment is a section of a conveyor path which connects two station submodels. Segments can be connected to form either open or closed loop conveyor paths. A closed loop path is one in which an item on the conveyor can return to a station by continuing on the device. An open loop path is one that is not closed. The segments defining a conveyor path are specified in the experimental form.

Conveyor cells are allocated to entities at HOLD blocks with the ACCESS hold type. Once a conveyor has been accessed, the entity can be conveyed to its destination workstation using the TRANSFER block with the CONVEY transfer type. A conveyor is exited at its destination station using the OPERATION block with the EXIT operation type. A conveyor may be stopped and started using the OPERATION block with the STOP and START operation types, respectively.

7. SUMMARY

In this paper we have given only a brief overview of the modeling features of SIMAN with an attempt to highlight those features which are particularly relevant to manufacturing systems. Only a small subset of the block functions were discussed, and no attempt was made to describe the enhanced general purpose features included in the language. In addition, a detailed discussion of the event and continuous modeling orientations included in SIMAN was omitted from the paper.

REFERENCES

- Advanced Manufacturing Methods Group, "Introductory User's Manual for GALS," Illinois Institute of Technology Research Institute, Chicago, 1978.
- Apple, J., Plant Layout and Material Handling, John Wiley, 1977.
- Gross, J. and J. Ippolito, Simulation with Speed, ORSA/TIMS National Conference, San Diego, CA 1982.
- Pegden, C. D. and A. A. B. Pritsker, Simulation Language for Alternatives Modeling, Simulation, Vol. 33, No. 5, Nov., 1979.
- Pegden, C. D. The SIMAN User's Manual, 1982.
- Schriber, T. Simulation Using GPSS, John Wiley, New York, 1974.
- Zeigler, B. P., Theory of Modeling and Simulation, John Wiley, 1976.