

Chapter 6

Simulation Languages

- ✚ Simulation languages are versatile, the general purpose of classes of simulation software that can be used to create the magnitude of modelling operation.
- ✚ In a sense, these languages are comparable to FORTAN, C#, VB.NET or Java but also include specific features to facilitate the modeling process.
- ✚ Some examples of modern simulation languages are GPSS/H, GPSS/PC, SLX and SIMSCRIPT III.
- ✚ Other simulation languages such as SIMAN have been integrated into a broader development framework.
- ✚ Simulation language exists for discrete, continuous and agent-based modeling paradigm.

Features of Simulation Language:

Specialized features usually differentiate from a general programming language. These features are intended to free the analyst from recreating software tools and procedures used by virtually all modelling application.

Not only the development of these features be time-consuming and difficult, but without them, the consistency of a model could vary and additional debugging, validation and verification would be required.

Most simulation languages should have the following features:

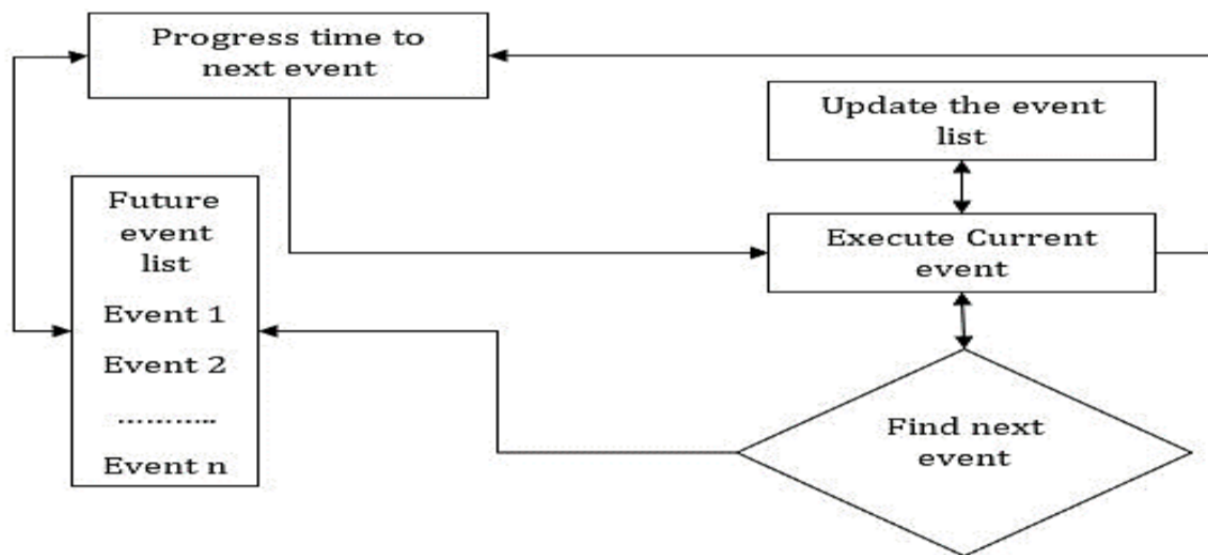
1. Simulation clock or a mechanism for advancing simulated time.
2. Methods to schedule the accuracy of the event.
3. Tools to collect and analyze statistics concerning the uses of various resources and entities.
4. Tools for reporting the result.
5. Debugging and error detection facilities.
6. Random number generator and related set of tools.
7. The General framework for model creation.

Working on Simulation Language:

Most discrete event simulation language model a system by updating the simulation clock to the time that the next event is scheduled to occur.

Events and their scheduled times of occurrence are maintained automatically on one of two order list, the current event chain or future event chain. The current event chain keeps a list of all events that will occur at the present clock time.

The future event chain is a record of all events that can occur at some point in the future. A simulation clock moves to the next event on the future events chain and changes the system state of the model based on the characteristics of that event.



Merits of Simulation Language

1. Since most the features to be programmed are in-built simulation language like comparatively less programming time and effort.
2. Since the simulation language consists blocks, specially constructed to simulate the common features, they provide a natural framework for simulation modeling.
3. The simulation models coded in simulation languages can easily be changed and modified.
4. The error detection and analysis is done automatically in simulation language.
5. The simulation models developed in simulation languages, especially the specific application packages, called simulators, are very easy to use.

Features of Simulation Software:

1. Modeling Flexibility:

The simulation must be flexible enough to adapt to change the configuration. The domain of application should be wide and the model should be equally valid over the whole domain.

2. Ease of Modeling:

It should be easy to develop the simulation model, easy to debug the program and validate the model. The software should have an in-built interactive debugger, error detector and online health.

3. Fast Execution Speed:

The speed of execution is always a very important requirement of any simulation model. It is especially an essential feature in case of simulation of a large and complex system.

4. Compatibility to Various Computer System:

The simulation language should be compatible with various computer system like microcomputer, engineering workstation minicomputers and mainframe computers.

5. Statistical Capabilities:

A simulation software should contain multiple stream random number generators and as many standards' probability distributions as possible. It should have blocks for designing the statistics like the simulation run, warming up period and the number and the length of replication.

6. The Capability of Animation:

The animation capability of the simulation package is one of the main reasons for the popularity of simulation language. The animation is carried out in two modes i.e., concurrent mode and playback mode.

7. Report Presentation Capabilities:

The output of the simulation the model should be well documented and illustrated. The user should be able to get the information of its interest in the easily understandable form.

TYPES OF SIMULATION LANGUAGE

- ✚ Computer simulation essentially and experimentally used for studying a wide variety of system.
- ✚ The purpose of such simulation is to observe the behavior of a given system within a given environment of the system.
- ✚ The abstract model of the system being simulated takes the form of a computer program and the system behavior is given by the output as the program run.
- ✚ We can simulate system to study their behavior using general purpose language (GPL) such as FORTRAN, C. PASCAL.
- ✚ But it will become difficult to program to debug and to modify when we have to simulate even a moderately complex system.
- ✚ So, it will better to use higher lever special purpose language SPL to simulate system. The languages design specially for simulating system offer many convenient facilities.
 - ✚ They facilitate modeling
 - ✚ They reduce programming
 - ✚ Provide better controller
 - ✚ Faster generation approach of codes.
- ✚ Writing simulation in GPL might be difficult tedious and time consuming.
- ✚ Due to these reasons the SPL that is simulation languages are used to simulate any kind of system.
- ✚ There are following types of simulation languages:
 - 1)Continuous system simulation language (CSSLS)
 - 2)Discrete system simulation language (DSSLS)
 - 3)Hybrid system simulation language (HSL)

1) CONTINUOUS SYSTEM SIMULATION LANGUAGE

- ✚ Before digital computers come into common use, analog computers were being used for simulating continuous system.

- ✚ The system in and analog computers was represented by the block of electronic devices such as operational amplifier to carry out the required operation.
- ✚ As soon as the digital computer arrived some of the disadvantages of analog computer are overcome using special program packages (called CANNED programs) which are implemented in digital computer to make digital computer applies like an analog computer.
- ✚ These are so called Digital- Analog, simulator (program package).
- ✚ These are meant either to replace and analog computer or check the results the of an analog simulation using Digital computer.
- ✚ Such simulation languages are called blocked structured continuous system simulation language.
- ✚ Example: DEPI, DEPI-4, DAS, MIDAS, IBM 1130, CSMP etc.

Later more advanced expression-based simulation languages were developed, example MIMIC, S/360, CSMP DYNAMO etc

2) DISCRETE SYSTEM SIMULATION LANGUAGE

- ✚ These languages are used to simulate discrete system and provide the following facilities.
 - Automatic generation of random no.
 - Automatic data collection
 - Statistical analyses of data
 - Reporter generators, etc.
- ✚ The other classification of discrete simulation languages is based on the general world view inherent in the language. Event, activity and process form the basis of three primary conceptual frameworks (world view) within discrete event simulation.

TYPES;

- A) **Event oriented language-** Example: SIMSCRIPT, GASP
- B) **Activity oriented language** - Example: MILITRAN
- C) **Process oriented languages** - Example: ASPOL, SIMUFOR

D) Transaction flow-oriented languages:

- ✚ Sub category of process-oriented language
- ✚ System model is represented by a flow chart consisting of language.
- ✚ The program creates transaction, executes them in the blocks and makes them along the flowchart.
- ✚ These languages are flow chart oriented; the best-known example is GPSS.

GPSS (GENERAL PURPOSE SIMULATION LANGUAGE)

- ✚ GPSS, one of the earliest DSL was developed by Geoffrey Gordon (1961-1962).
- ✚ The first release of this language was implemented on the IBM 704, 709 and 7090 computers.
- ✚ Later on improved and more powerful versions have been developed and implemented, including GPSS 2ND, GPSS 3RD (1965), GPSS 1360 (1967) and GPSS V (The latest version).
- ✚ GPSS was designed especially for those analyses who weren't necessary computer programmer because they do not write programmed in the logic as the SIMSCRIPT programmer does.
- ✚ Instead, he constructs a block diagram – an n/w of inter connected blocks, each performing a special simulation-oriented function.
- ✚ GPSS is particularly suited for traffic and queuing system.

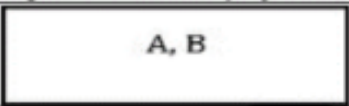
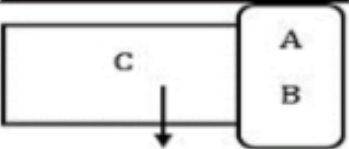

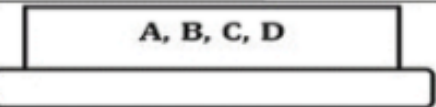
GENERAL DESCRIPTION

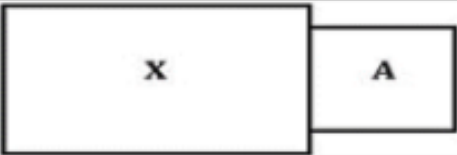
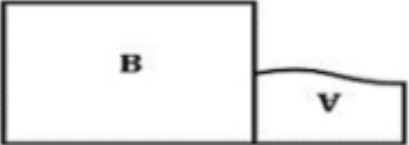


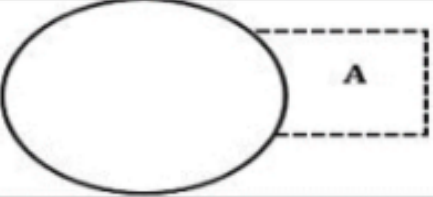

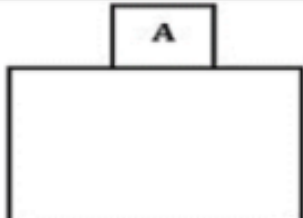
GPSS Block Diagram


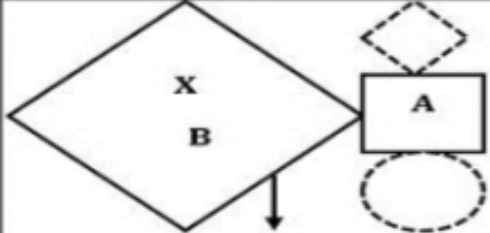
- ✚ The development of a simulation model in GPSS is a block-by-block construction.
- ✚ A set of standard blocks is arranged in the form of a block diagram that represents the flow of entities through the various paths of the system.
- ✚ Each block represents a step in the action of the system and links, joining the blocks, represent the sequence of events that can occur.
- ✚ To build a block diagram, it is essential to have a completed description of the system.


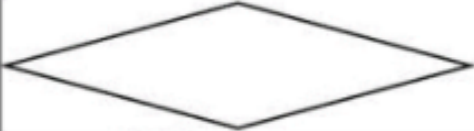


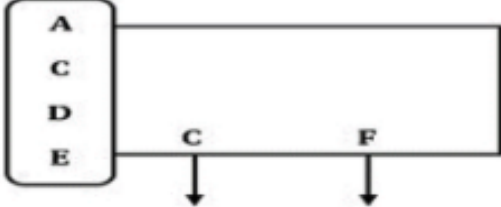
- ✚ The meanings of the blocks used in the system must be clearly defined.
- ✚ Each block must be assigned the block time, i.e., the time which the execution of the block will take.
- ✚ In total, a set of 25 specific block types have been designed, which can be used in the construction of a block diagram.
- ✚ Each block type can be used any number of times in a block diagram, but the total number of blocks should not exceed 2047.
- ✚ On the completion of the block diagram, each block is assigned a number, between 1 and 2047, called the **block number**.
- ✚ The system to be simulated in GPSS is described as a block diagram in which block represents activities and lines joining the block indicate the sequence in which the activity can be executed.
- ✚ Each block performs a simulated oriented function.
- ✚ GPSS V (latest version 5.2) provides a set of 48 different blocks, each of which can be used repeatedly.
- ✚ Each block has a name and specific task to perform.
- ✚ Each block type has a no. of data field such as A, B, C, and soon.
- ✚ Reflect the order to which they are specified.
- ✚ The entities of the system being simulated are called as **transaction**.
- ✚ Eg; costumer in a queuing system.
- ✚ **Typical blocks are;**
 - 1) **GENERATE:** create transaction
 - 2) **QUEUE:** creates a queue of transaction and maintain certain queuing statistic.
 - 3) **TABULATE:** tabulates the time it took the transaction to reach that point from the time it enters.
 - 4) **TERMINATE:** removes transaction form the system.
- ✚ There can be many transactions simultaneously block diagram.


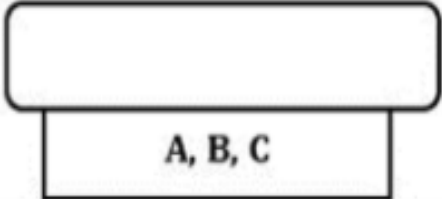
- A GPSS block diagram can consist of many blocks and given to each block an identification no. called a **location** is given to each block and movement of transaction is usually from one block to next block with next higher location.
- The block type '**ADVANCED**' is concerned with representing the expenditure of time. The program computes an interval of time called an action time for each transaction as it enters an '**ADVANCED**' block and transaction remain at this block up to that action time.
- The '**TRANSFER**' block allows some location other than the next sequential location to be selected. The choice is normally between 2 blocks referred to as next block 'A' and 'B'.
- Some symbols used for block types are shown below:

S.N.	Representation/Symbols	Meaning
1		Advance
2		Link
3		Seize
4		Assign

5		Logic
6		Tabulate
7		Depart
8		Mark
9		Terminate
10		Enter
11		Priority

12		Test
13		Gate

14		Queue
15		Transfer
16		Generate
17		Release
18		Unlink

19		Leave
20		Save Value

GPSS Program Application:

Let us consider, in a manufacturing shop, a machine tools turns out parts at one every 5 minutes then as they are finished the parts go to inspector, who takes 4 ± 3 minutes to examine each part and rejects 10% of parts. Each part will be represented on transaction and time until selected for problem will be 1 minute.

A GPSS block diagram for this system is as follows;

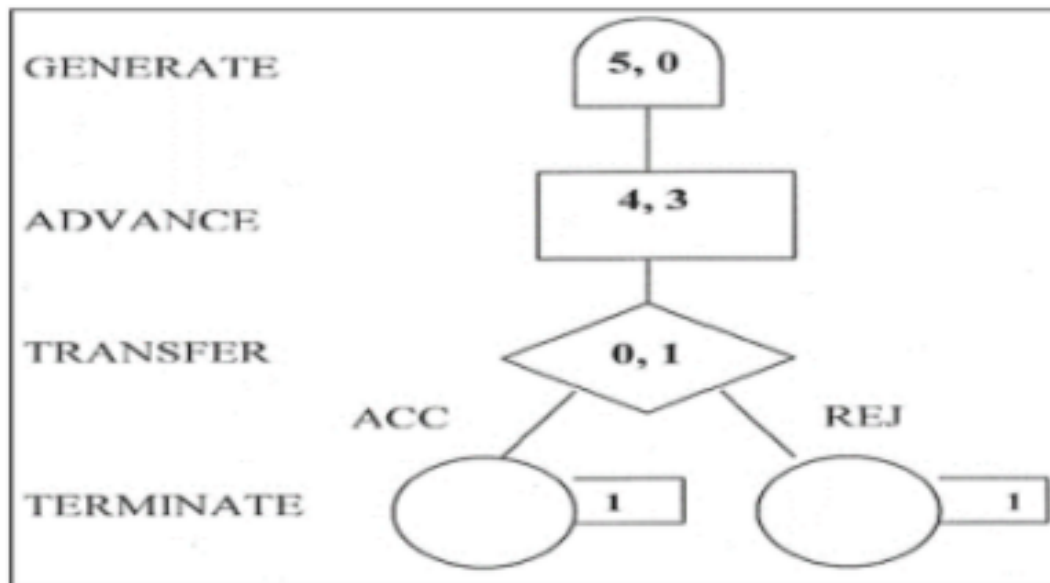


Fig: Manufacturing Shop - Model 1

Figure Description;

- ✚ In the block diagram, the block location is placed at the top of the block, the action time is indicated in the center in the form $T = a, b$ where a is the mean MD b is the modifier and the selection factor is placed at the bottom of the block.
- ✚ A **GENERATE** block is used to represent the output of machine by creating one transaction every five minutes of time.
- ✚ The **ADVANCE** blocks with a mean of four and modifier of 3 is used to represent inspection will therefore be anyone of values 1,2,3,4,5,6,7.

- ✚ After completion of the inspection transaction go to or a **TRANSFER** with a selection factor 'ACC' to represented accepted parts and 10% go to another location 'REJ' to represented rejected parts.
- ✚ Since there is no further interest both locations retired from the transfer blocks are terminate blocks.

GPSS coding of the above block diagram of the manufacturing shops.

MANUFACTURING SHOP

GENERATE	5
ADVANCE	4,3
TRANSFER	.1, ACC, REJ
ACC TERMINATE	1
REJ TERMINATE	1
START	1000

Fig: GPSS coding of manufacturing shop

RESOURCES IN GPSS

Facilities and storage;

- ✚ A facility is defined as an entity or resources that can be engaged by a single transaction at a time.
- ✚ Storage is defined as an entity or resource that can be occupied by much transaction at a time up to some pre-determined limit.
- ✚ Once a transaction seizes (holds) a facility any other transaction trying to seize the same facility is delayed until the first transaction releases the facility (resources). Let us consider a situation as below;

SEIZE	CPU
ADVANCE	7
RELEASE	CPU

- ✚ Here CPU is a facility and it is seemed that a transaction needs to use the resource for 7 times units.

- ✚ Any other transaction arriving at the block 'SEIZE' is refused to enter until the former transaction has entered RELEASE block.
- ✚ Resources which can be shared by several transactions are modeled using storage.
- ✚ Suppose we want to model a computer system which has 64kb of memory then we might declare 'MEMORY STORAGE 64' and then a request for 16kb memory might be represented by the sequences of blocks.

ENTER MEMORY 16

LEAVE MEMORY 16

- ✚ As the facilities a transaction arriving at ENTER block at a time when it is used by other transactions, is delayed until the previous transaction releases the necessary memory with.
- ✚ A transaction controlling and facility can be interrupted or preempted by other transactions.
- ✚ In addition, both facilities and storage can be available as occurs if the equipment they represent breaks down and can be available as occurs when, a repair has been made.
- ✚ To illustrate the use of these block types, consider again the manufacturing shop.
- ✚ Since the average inspection time is 4 minutes and the average generation rate is one every 5 minutes, there will normally be only part inspected at a time.
- ✚ Occasionally, however, a new part can arrive before the previous part has completed its inspection.
- ✚ The situation will result in more than one transaction being at the ADVANCE block at one time.
- ✚ Assuming that there is only one inspector, it is necessary to represent the inspector by a facility, to simulate the fact that only one part at a time can be inspected.

- ✚ A SEIZE block and a RELEASE block needs to be added to simulate the engaging and disengaging of the inspector.

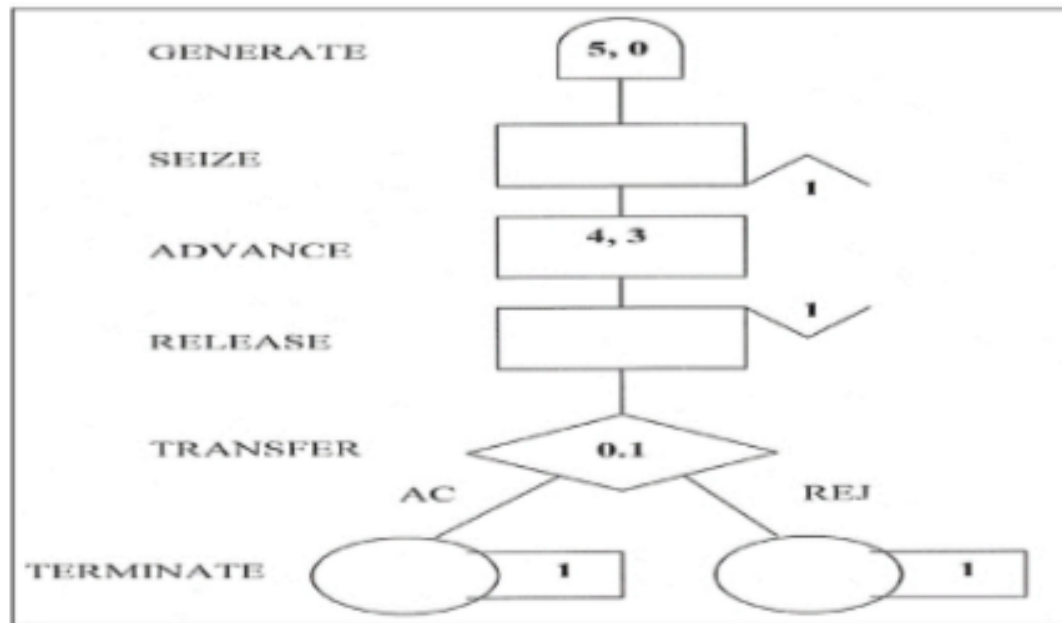


Fig: Manufacturing Shop – Model 2

- ✚ If more than one inspector is available, they can be represented as a group by storage with a capacity equal to the number of inspectors.
- ✚ The SEIZE and RELEASE blocks of the previous model are then replaced by an ENTER and a LEAVE block, respectively.
- ✚ Suppose, for example, the inspection time were three times as long as before; three inspectors would be justified.
- ✚ The case of a single inspector can be modeled by using storage with capacity 1 instead of the facility. An important logical difference between the two possible representations is that, for a facility, only the transaction that seized the facility can release it, whereas, for storage, entering and leaving can be separate actions carried out independently by different transactions.

SIMSCRIPT PROGRAM:

- ✚ SIMSCRIPT is a very widely used language for simulating discrete system.
- ✚ The language can be considered as more than just a simulation language since it can be applied to general programming problems.
- ✚ The description of the language given is organized in five levels.
 1. Beginning with simple teaching level to introduce the concept of programming.
 2. The description add level corresponding to a specific programming language.
 3. A general-purpose language.
 4. A list processing language, needed for creating the data structure of a simulation model.
 5. The simulation-oriented function needed to control the simulation and gathering statistics.

SIMSCRIPT SYSTEM CONCEPTS

- ✚ The system to be simulated is considered to consist of entities having attributes that interact with activities.
- ✚ The interaction causes event that change the state of the system.
- ✚ In describing the system, SIMSCRIPT uses the terms, entities attributes, sets, event routine.
- ✚ An entity is a program element similar to a subscripted variable.
- ✚ When an entity is created it can be interpreted as a genetic definition for a class of entities.
- ✚ Individual entities have valued all attributes which define particular state of the entity attributes are named not a number. For eg we may define employee to be an entity and AGE, SALARY are attributes.

Entities can be 2 types

1) Permanent

2) Temporary

Permanent (specify for all time the proceeds)

Temporary (specify dynamically all the program proceeds)

Temporary entities are physically created and destroyed their special statements.

Permanent entities have their attributes stored as array

- ✚ Both temporary and permanent entities can belong to sets.
- ✚ And own sets. SIMSCRIPT uses pointer to chain together entities that are members of sets.
- ✚ Commands are available to manipulate these sets.
- ✚ The user can define sets and facilities are provided for entering and removing entities into and prompt sets.
- ✚ Activities are considered as extending over time with their beginning and their end being mark as events occurring instantaneously.
- ✚ Each type of event is described by a event routine, each of which is given a name and programmed as a separate closed sub routine. Activities may be endogenous and exogenous.

ORGANISATION OF A SIMSCRIPT PROGRAM:

- ✚ Since event routines are closed routines, some means must be provided for transferring control between them.
- ✚ It is done by the use of event notice.
- ✚ These event notices are created when it is determined that event is scheduled.
- ✚ The event notice is filled in chronological order.
- ✚ When all events that can be executed at a particular time have been processed, the clock is updated to the time of the next event notice and control is passed to the event routine identified by the notice.
- ✚ Their actions are automatic and do not need to be programmed and event notice do not usually go to more than one activity.

- ✚ If the event executed by a routine result in another event, either at the current clock time or in future, the routine must create a new event notice and file it with other notice.
- ✚ In the case of the exogeneous event, a series of exogeneous event statement is created one for each event.
- ✚ All exogeneous event statements are sorted to chronological order and they are read by the programs when the time for the event is due to occur.
- ✚ The general manner in which simulation proceed is illustrated in below fig.

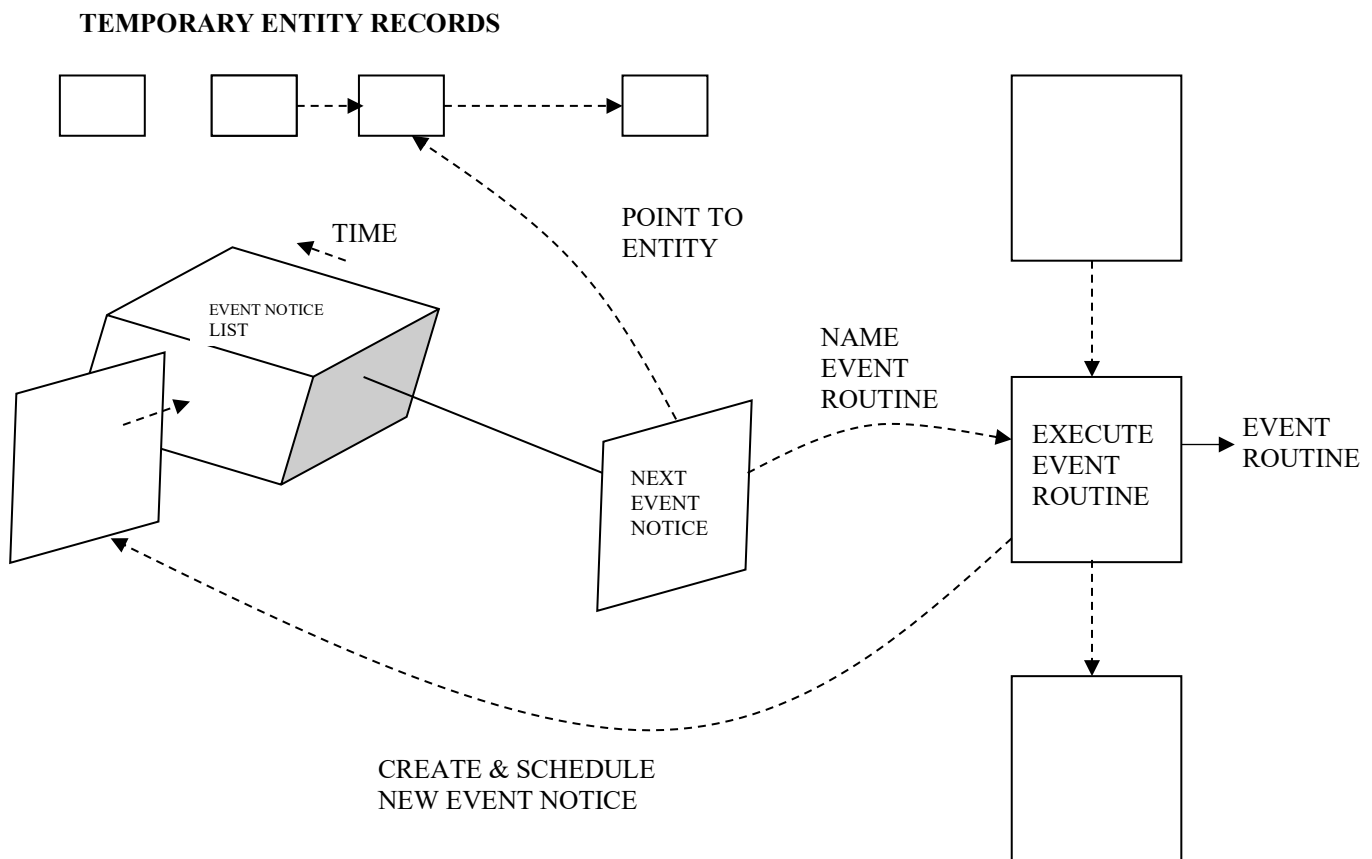


Fig. SIMSCRIPT EXECUTION CYCLE