Course No: MCA 503 Distributed System, Parallel Computing and Simulation

(3 Lecturer/week & 1 Seminar/Tutorial Total marks: 100)

Course Content:

1. Distributed System – I
   a. Basic Concepts, Advantages and disadvantages of Distributed Systems
   b. Tightly coupled and loosely coupled Systems
   c. Hardware and software requirements. Design issues
2. Distributed Systems – II
   a. Implementation methodologies
   b. System models for organization of processors in distributed systems
   c. Processor allocation models, Distributed file system design, Synchronization aspects
3. Parallel computing
   a. Introduction to parallel computing
   b. Parallel processing terminology: decomposition, complexity, throughput, speedup, data dependence, resource dependence, level of parallelism in programs
   c. Basic architectures: Multiprocessors, vector processors, pipeline, array, systolic, cube, hypercube, CCC, pyramid, prism, butterfly and suffice exchange
   d. Elementary parallel algorithms: odd-even, Bitonic merge
4. Simulation –I
   a. Introduction, Applications, Advantages & disadvantages
   b. Examples of simulation, Continuous system simulation, examples
   c. Numerical v/s continuous system simulation
   d. Analog v/s Digital simulation
5. Simulation –II
   a. Discrete System simulation, examples
   b. Fixed time-step & event-to-event model, Simulating randomness
   c. Generation of non-uniformly distributed random numbers
   d. Monte-Carlo computation and its application
   e. System, system environment, attributes, activities, types of activities
   f. Types of model and principal used in modeling, System studies
6. Simulation – III
   a. Types of system study, Differential computational techniques used in simulation
   b. Distributed lag models, Cobweb model
   c. Process of simulating, design and evaluation of simulation experiments validation
   d. Introduction to simulation language

Main reference Books:

(A) Tanenbaum Andrew S: Distributed operating system, Addision Wesly,2001
(B) Quinn Michael J: parallel computing – theory and practice (second edition) McGraw-Hill Pub, 1984
(C) Hira D.S.: System simulation, S Chand & Co,Ltd., 2001
(D) Gorden Geoffrey : System Simulation Prentice Hall of india, New Delhi, 2001

Additional referenced Books:

(E) Deo Narsingh: System simulation with digital computer, Prentice Hall of india, New Delhi, 2001
(F) Law A M & kelton W D: Simulation modeling and analysis , Tata Mc-Graw hill,2000

Assignment –I (simulation) submission before 1st Internal test

- What is System? Give classification of system.
- What is system simulation? What are the uses of simulation? What are the Types of simulation?
- Explain steps in system simulation study? Explain Phase of system study with flowchart.
- What are the advantages and disadvantages (or Limitation) of simulation?
- What are the areas of application of simulation?
- Explain simulation of continuous system with example of a chemical reactor?
- Explain analog simulation.
- Explain disadvantages of analog simulation over digital simulation
- Explain numerical integration v/s continuous system simulation
- Explain simulation as management laboratory

Simulation –I

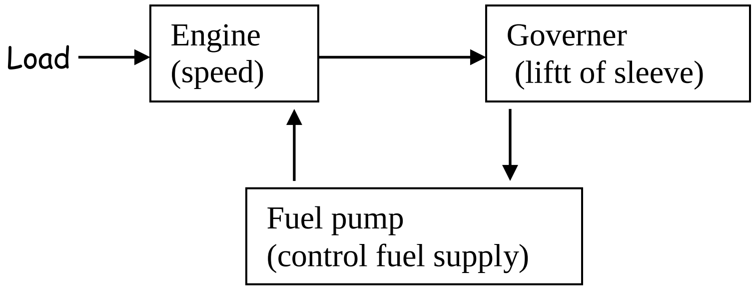
Introduction:

- Simulation is representation of real life system by another system, which describe the important characteristics of real system and allows experiments on it. In other word simulation is an imitation (follow the example of) of reality.
- Researchers, analysts, designers and other professionals in physical and non-physical experimentations and investigation have used formal use of simulation technique. In our day-to-day we use simulation, even realizing it.
- Examples of simulation used in real life.
  - Simple model of machines are used to simulate plant layouts.
  - A model aeroplane suspended in wind tunnel simulates real sized plane moving through the atmospheres and used to study the aerodynamic characteristics.
  - A planetarium represents a beautiful simulation of planetary system.
  - Environments in a geological park and in a museum of natural history are other examples of simulation.
- What is system?

  System is collections of components where in individual components are constructed by connecting inter relationships such that the systems as whole fulfill specific function in response to varying demand.

  An aggregation or assembles of objects joined in some regular interaction or interdependency.

  Physical system: example of an Engine governor



  In case of governor system the components of the system (are the engine, governor, fuel pump etc) are physical. Their interrelationships are based on well-known physical laws such system are called physical system.

  Non-physical system: The manufacturing system is often comprises by large numbers of departments with man made relationships which cannot be represented by physical objects such system are classified by non-physical systems. Managements systems, social systems, education system & political systems etc are non-physical system.

- Activities, which occurs within the system are called **endogenous** activities, while activities occur in environment are called **exogenous** activities.
- A system for which there are no exogenous activities that is the system not affected by its environments is called **closed system.** While a system with affected by the activities occurring outside its boundary is called **open system**.
- The method of tacking the system is called system approach. The term system engineering has been used for engineering employing system approach.
- Term **entity** is used to denote the component of interest, and property interest of the entity is called its **attribute**. In traffic system vehicles (bus, trucks, and cars) may be entity and speed, distance moved and numbers of accidents etc may attributes. The process, which causes change in system, is called **activity**.

Continuous and Discrete System:

- In the viewpoint of simulation system are classified in two types continuous and discrete systems. System in which the state of system changes continuously with time is called **Continuous system.** System in which state of system abruptly (suddenly) at discrete points in time are called **Discrete System.**
- Examples: In case of inventory system the demand of items as well as replenishment (fulfill again) of stock occur at discrete point in time also in discrete numbers. In case of queuing system customer arrives and leaves in system at discrete points in time. Generally the system in which relationship can be expressed by mathematical expression in engineering and physical sciences turn out be continuous system while the system encountered in operation research and management sciences are generally discrete system.

  In factory system through the start and finish of a machine are discrete points but machining process is continuous.

  Thus no specific rule can be laid for development of simulation model however effort should be made to reach judicious balance between the simplicity of model and discrete level of detail and accuracy of result.
- **System Simulation::**

  Forming of physical model & experimenting on it is simulation. Developing a mathematical model & deriving information by analytical means is simulation.

  System simulation is technique of solving problems by observation of the performance, over time, of a dynamic model of the system. Thus the dynamic model & time element are two important factor/components of system simulation.

  In many simulations the time element may not be significant parameter, but still system is analyzed by step-by-step calculations of successive stage of system.

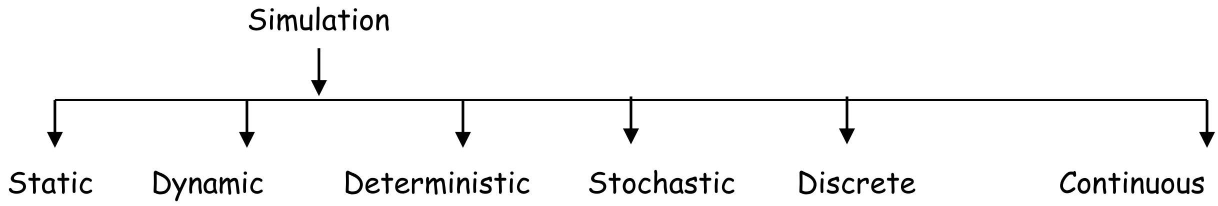
  System simulation is also some time called computer simulation, only because any simulation worth the name can only carried out within a computer.
- **Real time Simulation:**

  Systems usually involve interaction with a human being there by avoiding the need to design & Validate a model of human behavior. The aircraft cockpit simulator for training of pilots & simulated zero gravity chambers for training of astronauts are examples of real time simulators. These systems are real time because the time taken by experiment will be real that is the same as in case of a completely physical system. The real time simulation require computer that operate in real time, it means that they must be able to response immediately to signals sent from physical devices and send out signals at specific points in time.
- When to use simulation?

  Following are some purpose for which simulation may be used.

1. Simulation is very useful for experiments with the internal interactions of complex system or of a subsystem within a complex system.
2. Simulation can be employed to experiment with new design & polices before implementing them.
3. Simulation can be used to verify the result obtained by analytical method and reinforce the analytical techniques.
4. Simulation is useful in determining the influence of changes in input variables on the output of system.
5. Simulation help in suggesting modification in the system under investigation for its optimal performances.

- **Classification of simulation/ models:**



1. A static simulation model represents system, which does not change with time or represent system at particular point in time. Static simulation some times called Monte-Carlo simulation.
2. Dynamic simulation models represent system as they change over time. Ex. Simulation of a repair shop over a month.
3. Deterministic simulation model have known set of input, which result in to unique set of outputs.
4. In stochastic model there is one or more random input variables, which lead to random output. The outputs in such a case are only estimates of true characteristics of the system. Example simulation of an automatic machine where operation times may vary from man to man & from time to time in random fashion.
5. System in which the state of system changes continuously with time is called **Continuous system simulation.** System in which state of system abruptly (suddenly) at discrete points in time are called **Discrete System simulation.**

- **Steps in simulation study:**

a. Problem formulation:

The clear description of problem, definition of the objectives of study, identification of alternatives to be considered and methodology for evaluating effectiveness of these alternatives need to be started at beginning of any study. The overall plan should include a statement of alternative system to be considered, the measures of performance to be used, the methodologies of analysis to be used and the anticipated result of study.

b. Model construction:

Model building is much of an art than science. There are no standard rules for building successful and appropriate model for all types of simulation. There are only certain guidelines, which can be followed. The art of modeling enhanced by ability to abstract the essential features of the system, to select and modify the basic assumptions & simplification that characterize system and than elaborate (work out) the model.

To start with simple model is constructed, which step-by-step modified, every time enriching and elaborating its characteristics, to achieve appropriate model, which meet desire objectives.

In some simulation, building block method is employed, where the model blocks of components of system are built and validated. These blocks are then combined to obtain model for the complete system.

c. Data collection:

The availability of input data about systems is essential for construction of its model. The kind of data to be collected depend upon the objectives of the study, the required data may be available as past history, or may be have to be collected. Data collections have constant interplay, and type & amount of data may change as model develops. Collected data not input but also validate the simulate model. Data collection takes longer time it should be started as early as possible.

d. Model programming:

Any simulation model worth the name required large amount of computations & information storage which possible only with the high-speed computers. Translation of model in to a computer recognizes format is termed as programming. Many general purpose & special purpose software simulation language is available to write simulation program. Simulation languages are usually more powerful than special purpose software used in simulation. The general-purpose languages BASIC, FORTAN, c, c++ have extensively used for writing simulation program.

e. Validation:

Validation involves both validation of the logic & accuracy of programming. This required step-by-step modification of model. It is rarely possible to develop reasonably large simulation model in its entirely in first step. This way validation in an iterative process of comparing model to actual system behavior, identify the error, applying correction & again comparing performance. This process continue still a model of desired accuracy is obtained. Data collected from actual system is of great help in validation.

f. Design of experiments:

Simulation is basically experimentation on the model of the system under investigation. In most of the situations involves stochastic variables, which result into stochastic results. The average value of result is obtained may not be of desired reliability. To make reliable & meaningful, it is essential simulation experiment be designed to such a way that result obtained are within some specified tolerance limits and at a reasonable level of confidence. Decision regarding length of simulation run, initial condition, removal of initial bias, number of replication of each run, use of variance reduction techniques etc have to be made.

g. Simulation run & analysis:

Simulation program run as per simulation design, the result are obtained and analyzed, to estimate measure of performance of the system. Based on result, a decision is made, whether or not any modification in the design of simulation experiment is needed. This step is a sort of validation of the simulation design. It may reveal that more runs or replications are required.

h. Documentation:

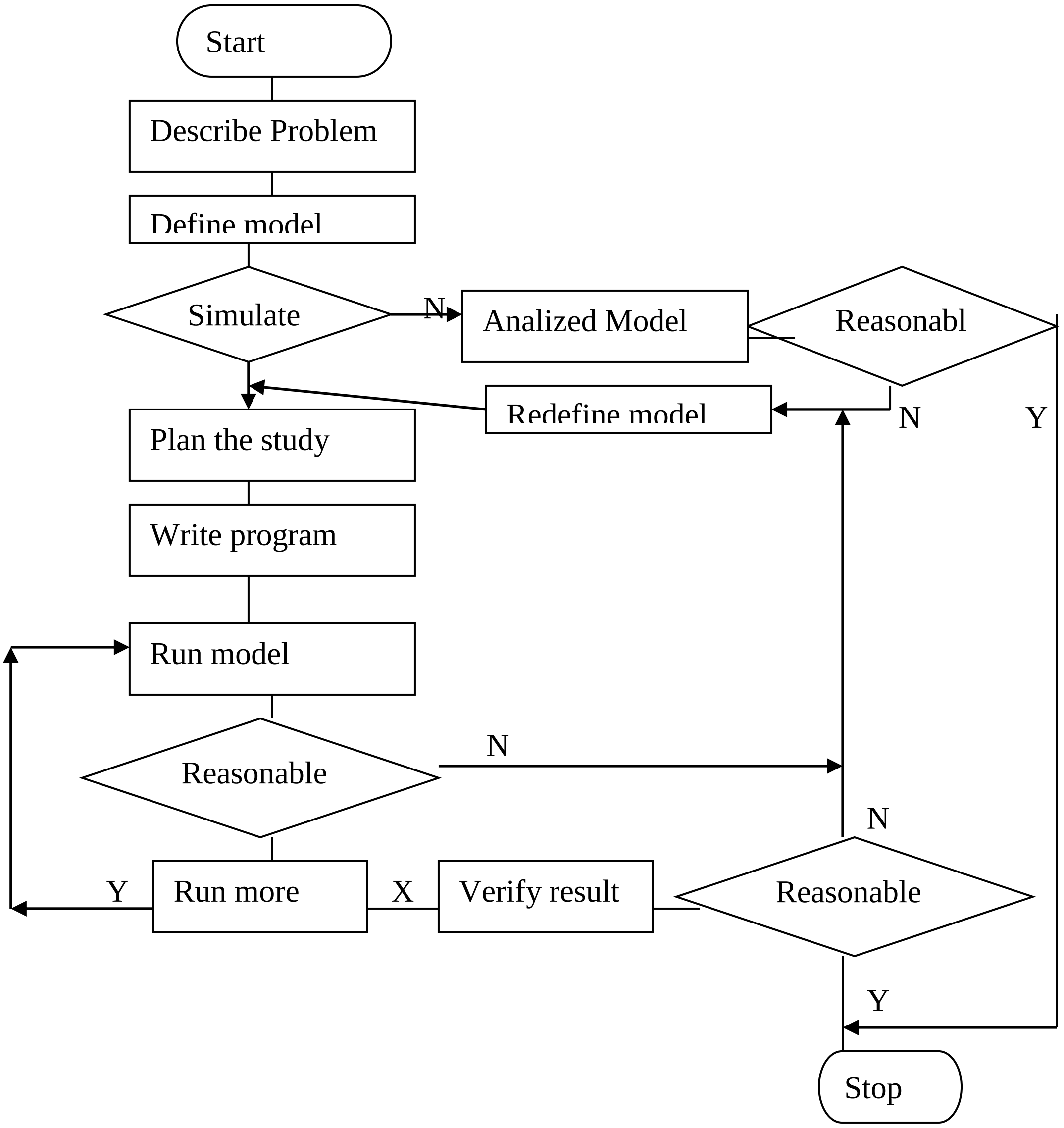
It is necessary to make documentation of simulation program can be used by the same or different analyst in feature. For feature modification program can be documented properly so that new user can easily understand it.

i. Implementation:

There will not be any problems in implementation of simulation program, if user is fully conversant with the model., and understand the nature of its inputs and outputs and underlying assumptions. Thus it is important that the model user is involved in development of simulation model from the very first step.

- **Phase of a simulation system study**:
  a. Problem formulation
  b. Model building includes model construction, data collection, programming & validation.
  c. Running model includes experiment design, simulation runs and analysis of result.
  d. Implementation includes documentation and implementations

Flowchart: Process of simulation study



- **Advantages of simulation:**
  a. It helps to learn real time system, without having the system at all. Ex. Wind tunnel testing model of an aeroplane does not requires a full sized plane.
  b. Many managerial decision making problems are too complex to be solved by mathematical programming.
  c. In many situations experiment with an actual system may not be possible at all. Ex. It is not possible to experiment, to study behavior of man on the moon surface. In other, even if experiment is possible, it may be too costly and risky.
  d. In real system, the changes we want to study make take place slowly or too fast to observe conveniently. Computer simulation can compress the performance of a system over years in to a few minute of computer run time.
  e. Simulation eliminates the need of costly trivial & error method of trying out new concepts.
  f. Simulation relatively free from mathematics can easily understand by the operating personnel & non-technical managers. This way helps in getting the proposed plans accepted & implemented.

g. Simulation model are comparatively flexible & can be modified to accommodate changing environment to real situation.
h. It is easier to use than mathematical model & can be used for wide range of situations.
i. Extensive software package available, making it very convenient to use fairly sophisticated model.
j. It very good tool of training & advantage use for managerial & operating staff in operation of complex system. Ex. Space flight simulation, airline pilots are given extensive training on flight simulation before they allowed handle real aeroplane.

- **Limitation of simulation:**
  a. Simulation does not produces optimal results when model deal with uncertainties the results of simulation are only reliable estimates subject to statistical error.
  b. Quantification of variables is another difficulty. In number of simulation, it is not possible to quantity between them make problem very unwieldy.
  c. Simulation is no mean by a cheap method of analysis. Even small simulations take considerable computer time.
  d. Stem (check or stop) from too much tendency to rely on simulation models. This results in applications of technique to some simple situations, which can be more appropriately be handled by other techniques of mathematical programming.
- **Application area / area of application of simulation:**
  a. Manufacturing:
  Design analysis & optimization of production system, material management, capacity planning, and layout planning & performance evaluation, evaluation of process quality.
  b. Business:
  Market analysis, prediction of consumer behavior & optimization of marketing strategy and logistics, comparative evaluation of marketing campaigns.
  c. Military:
  Testing alternative combat strategies, air operation, sea operation, simulated war exercises; practicing ordinance effectiveness, inventory management.
  d. Health care application:
  Planning of health services, expected patient density, facilities requirements hospital staffing, estimating the effectiveness of healthcare program.
  e. Communication application:
  Such as network design, optimization, evaluating network reliability, manpower planning, sizing of message buffers.
  f. Computer application:
  Such as designing hardware configurations & operating system protocols, sharing & networking.
  g. Economic application:
  Such as portfolio management, forecasting impact of govt. policies, international market fluctuations on economy. Budgeting & forecasting market fluctuations.
  h. Transport application:
  Design & testing alternative transportation policies, transport networks – roads, railways, and airways etc. evaluation of timetables, traffic planning.
  i. Environment policies:
  Solid waste management, performance evaluations of environments programs, evaluation pollution control system.
  j. Biological application:
  Such as population genetics & spread of epidemics (wide spread occurrence of disease on particular community at particular time).

- **Explain simulation – A management Laboratory:**
  Simulation is a very important tool of decision-making. The managerial problems are very complex to be solved by the analytical techniques. Various techniques of operation research are applicable to only specific types of situations, and required many assumptions and simplification s to be made for fitting the problem into the model. Many events occurring in real systems are random with intricate interrelationships, with their solution beyond the scope of standard probability analysis. Under the circumstances simulation is the only tool, which allows the management test the various alternative strategies. Since, simulation is a sort of experimentation, and when used for analyzing managerial problems, it rightly called the management laboratory. For training business executives, simulations called management games are used in many university and management institutes.
- **Continuous system simulation with example of chemical reactor**
  In a certain chemical reaction when two substance A and B are brought together they produce a third chemical substance C. It is known that 1 gram of A combines with 1 gram of B to produce 2 gram of C. Further more, the rate of formation of C is proportional to the produce of the amounts of A and B present.
  In addition to this forward reaction there is also a backward decomposing C back into A and B. The rate of decomposition of C is proportional to the amount of C present in the mixer. In other words, at any time t if c1, c2 and c3 are the quantities of the chemicals A and B and C present, respectively then their rates of increases are described by the following three differential equations:
  $dc1/dt=k2-k1c1c2$, $dc2/dt=k2c3-k1c1c2$, $dc3/dt=2k1c1c2-2k2c3$
  Where k1 and k2 are constant
  To keep the problem simple, it is assumed that temp, pressure and humidity are maintained constant and we have no effect on rate of formation or decomposition of chemical.
  As soon as A & B mixed in the reaction start and amount of c1, c2 & c3 in mixture go on changing as the time progress. Simulation of the reaction will determine stat of system, that is value of quantities c1, c2 & c3 at different point to time.
  Starting at zero time, a very small amount increment of time is taken in each step. If each increment c1(t) , c2(t) & c3(t)are the quantities of three chemical at time t, then at $t+\Delta t$ at quantities are $c1(t+\Delta t)=c1(t)+(dc1(t)/dt)\Delta t$
  So identical equation can be written for
  $c2(t+\Delta t)$ & $c3(t+\Delta t)$
  taking time c1(0), c2(0) & c3(0) as quantities of A,B,C at time zero that is when reaction start the state of system at Δt will be
  $c1(\Delta t)= c1(0)+[k2c3(0)-k1c1(0)c2(0)]\Delta t$
  $c2(\Delta t)=c2(0)+[k2c3(0)-k1c1(0)c2(0)]\Delta t$
  $c3(\Delta t)=c3(0)+[2k1c1(0)c2(0)-2k2c3(0)]\Delta t$
  Using the state of system at $2\Delta t$, state of system $3\Delta t$ & so on will be calculated. This will continue still specified time of reaction T is reached. At each increment of time, we can either count number of steps taken or check attained time with the prescribed simulation time.
  Following is the implementation of above chemical reactor simulation in c language.

```
//chemical reactor simulation problem
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<stdlib.h>
void main()
{int i,n,k;
```

```
float delta=.1,t=0,time=15,c1[200],c2[200],c3[200],k1=.025,k2=.01;
clrscr();
c1[0]=50,c2[0]=25,c3[0]=0;
i=0;
printf("\nTime\tC1\tC2\tC3");
while(t<=time)
{ printf("\n%5.2f\t%6.2f\t%6.2f\t%6.2f",t,c1[i],c2[i],c3[i]);
  c1[i+1]=c1[i]+(k2*c3[i]-k1*c1[i]*c2[i])*delta;
  c2[i+1]=c2[i]+(k2*c3[i]-k1*c1[i]*c2[i])*delta;
  c3[i+1]=c3[i]+2*(k1*c1[i]*c2[i]-k2*c3[i])*delta;
  i++;t+=delta;if(t>=2) delta=.2;if(t>=6) delta=.4;
  }
  printf("\nany number");scanf("%d",&n);
  }
```

- **Numerical integration v/s continuous system simulation**
  1. In simulation we always keep track of the state of system explicitly. The out come of each step of numerical calculation in simulation experiment can be interpreted directly as state of the system at some point in time. Simulation essentially consists of constructing history of a system – a succession of explicit state description at each instant as we move forward in time. In this way one to one correspondence between computers does & what take place in the system.

     In numerical solution of equation no such correspondences is preserved. Usually pure computations short cuts taken, parameter are lumped & mathematical equation maintained before computer program is developed. These destroy one-to-one correspondences between computer steps and original system from which equation is derived. Consequently the output data have to interpret in light of earlier manipulation before conclusion can draw about system.
  2. Secondly they're in difference of attitude. In case of numerical calculation we only see the given sets of differentials equation as a mathematical objects & proceed to integrate them. In simulation we look upon the equations as one of steps in the process. We know in real life system, we ware of approximations in model that being simulated.

     Finally by looking with output data, which directly represent the system. We also prepared to modify the model if necessary.
- **Analog simulation:**

The analog simulation in general, is the simulation performed by using analog model of the system. The simulation carries on by employing an analog computer is called analog system simulation. Simulation with an analog computer is based on a mathematical model than being on a physicals model.
In analog simulation various mathematical function of operation are represents by some hardware device whose behavior is equivalent to the mathematical function like addition multiplication or integration.

The most widely used form of analog computer is the electronic analog computer, based on use of high gain direct current amplifiers called operational amplifiers. Voltage in computer are equaled to mathematical variables & operational amplifies can add or integrate voltages.
In such simulation depending upon system being simulated some amount of hardware devices required. Those hardware are interconnected imitate the system.

The elements performing mathematical operation like add, integration, multiplication subs traction saucer root etc. can be express in blocks, which are like subroutines.
Any continues system, can be simulation by suitably coming together a number of blocks. Different systems have different number of blocks & their interconnectivity.
Since the same blocks can be organized in different ways, it is advantages to have packages of standard subroutines to perform the operating of blocks. Hence it is more convenient to take a special purpose block oriented programming language.
A large number of such contains system simulation have been designed and implements.

- **Disadvantage of analog simulation v/s digital simulation:**
  1. **Limited Accuracy:**
     Result obtained by this has limited accuracy so more accurate results digital simulation is used. Example: where high accuracy is required as in space vehicles, guided missiles & Fusion analog simulation not used.
  2. **Magnitude scaling:**
     In analog simulation values of variables are represented in voltages, which have fixed & limited range other wise result become inaccurate. It is very difficult task, when number of variables is large & range of their variation is not known. Where digital simulation the problem of magnitude scaling does not arise, as they have very large range with floating point arithmetic, they have very large precision. Hence no magnitude scaling required in digital simulation.
  3. **Hardware set up required:**
     In analog simulation hardware elements have to combined to simulate system, they have to be tested & calibrated. Where no such setup required in digital simulation. Switch one simulation to another take time where in digital simulation no time required. A simulated program on digital computer can be easily stored for reuse.
     Finally analog simulation has two advantages over digital simulation one speed of solution & second immediate display result of simulation.

Simulation-II

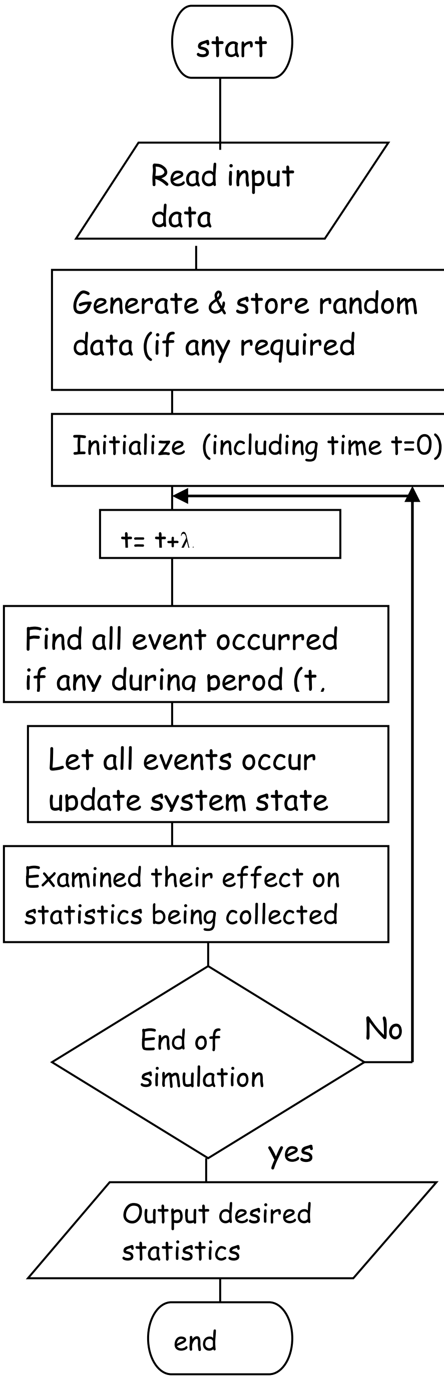
Discrete System Simulation:

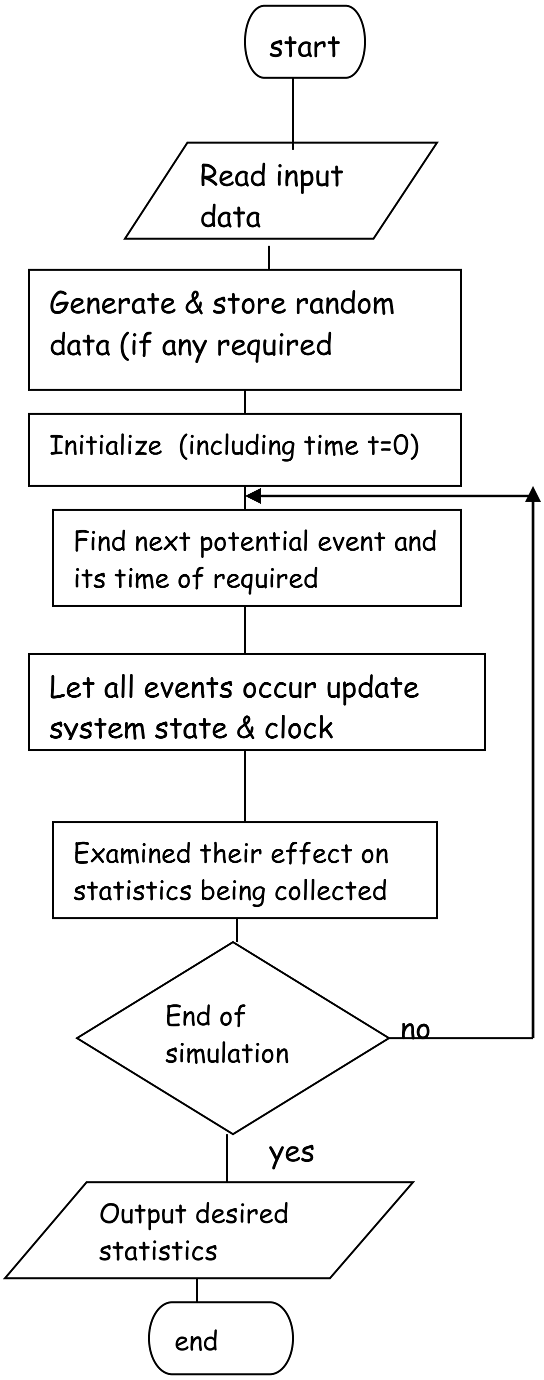
In this simulation changes are discontinuous. Each system change in system state is called **event**. For example, arrival or departure of a customer in a queue is an event. Likewise sale of an item from the stock or arrival of order to replenish the stock is an event in an inventory system. Arrival of car at an insertion is an event if we are simulating street traffic. Therefore simulation of discrete system often referred as to discrete-event-simulation.
Discrete event simulation commonly used by operation research workers to study large complex system, which does not lend themselves to a conventional approach. Example: simple inventory control, study of sea, airports, steel melting shops, telephone exchange, production line, stocks of goods, scheduling of projects etc.
Discrete system simulation is more diverse & less of a theory than continuous system simulation. There is no overall set of questions to be solving in discrete event simulation.

Fixed time step v/s event-to-event model:

- In simulation any dynamic system continuous/ discrete there must be a mechanism for the flow of time. For we must have advance time, keep track of total elapsed time, determine the state of the system at the new point in time, and terminate the simulation when the total elapsed time equals or exceeds the simulation period.
- As seen in continuous system (chemical reactor) we advanced time by in small increment of Δt as long as was needed.
- While in simulation of discrete system there are two fundamentally different models for moving a system through a time the fixed time step & event to event (or next event) model.

- In fixed time the computer simulates model a 'timer' or 'clock'. This clock is updated by a fixed time interval t and the system is examined to see if any event has taken place during this time intervals (minutes, hours, days whatever). All events that take place during this period are treated as if they occurred simultaneously at the tail end of this interval.
- In event-to-event or next model simulation the computer advance time occurrence of next event. It shifts from event-to-event the system state does not change in between. Only those points in time are kept track of when something of interest happens to system. In general next event model is preferred (except when we may forced to use fixed time) because we do not waste time in scanning those points in time nothing takes place. This waste in bound to occur if we pick up a reasonably small value for λ. On other hand, if λ is so larger that one or more events must take place during each interval then our model become unrealized & may not yield meaningful results. There for simulation of discrete system next-event model is used.
- Flow charts for fixed-time-step and event-to-event model simulation

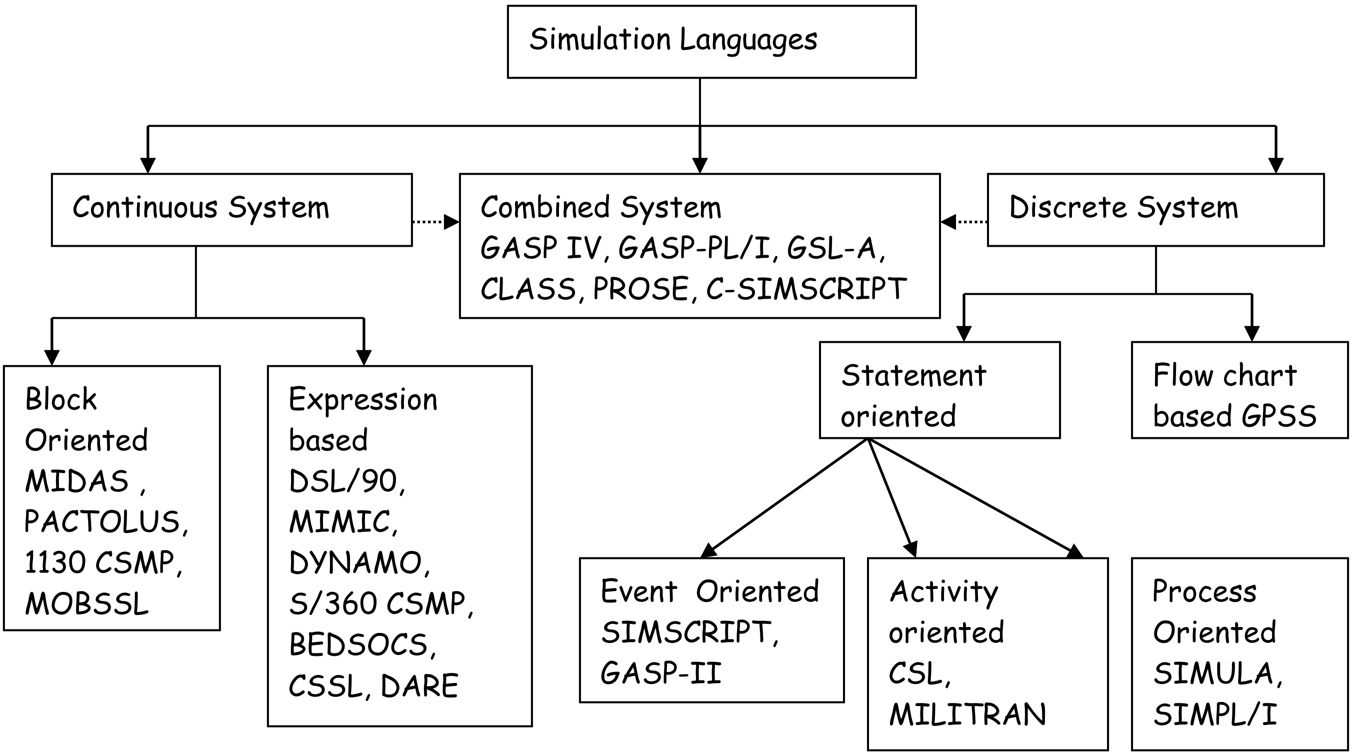




Simulation Language:

Both general-purpose computer languages like FORTRAN, C, BASIC, ALGOL and PL/I and the specially designed simulation languages (SIMSCRIPT, GASP, CSL, MILITRAN, SIMULA, GPSS) are used for simulation programs.

Classification of Simulation Languages:



Merits of Simulation Languages:

1. Since most of the features to be programmed are inbuilt, simulation languages comparatively less programming time effort.
2. Since simulation language consists of 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 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.

Merits of General purpose Languages:

1. The biggest advantage of general-purpose languages is their availability on almost all the computers. On other hand special purpose languages are not easily available to all the simulation modelers.
2. These languages are generally well known to simulation modelers, while the same is not true in case of simulation languages.
3. These languages being widely known, the simulation modeler has number of people around to discuss the programming problems, while in case of simulation languages such expertise is not easily available.
4. Program written in these languages like FORTRAN or C requires comparatively less execution time. But introduction of fast computers, this merit of general-purpose languages has lost its significance.
5. These languages may allow flexibility in programming as compared to simulation software.
6. These languages being generally available on computes are not considered to increase additional cost, while simulation software has to be specially acquired, at a high cost.

Simulation Software packages designed for following objectives:

1. To conveniently describe data elements, which commonly appear in simulation such as the generation of random deviates.
2. Flexibility of changing the design configuration of system so as to consider alternate configurations.
3. Internal timing and control mechanism, for book keeping of vital information during the simulation run.
4. To obtain conveniently, the data and statistics on behavior of the system.

Desirable Features of Simulation Software:

1. **Modeling Flexibility:** A simulation language must be flexible enough to adapt to changed configurations. The domain of application should be wide and model should be equally valid over the whole domain.
2. **Ease of modeling:** It should be easy to develop the simulation model, to debug the program and to validate the model. The software should have inbuilt interactive debugger error detector and on-line help.
3. **Fast Execution speed:** The speed of execution is always a very important requirement of any simulation model. It is specially an essential feature in case of simulation of large and complex systems.
4. **Compatibility to various computers systems:** The simulation language should be compatible to various types of computers, like the microcomputers engineer workstations, minicomputers and mainframe computers.
5. **Statistical capabilities:** A simulation software should contain multiple stream random number generators and as many standard probability distributions as possible. It should have blocks for designing the statistic like the simulation run warming up period and the number and length of replications. It should be able to carry out the statistical analysis of results, including the construction intervals for the specified measures of performance.
6. **Capability of animation:** The animation capability of the simulation packages is one of the main reasons of simulation packages is one of the main reason of the popularity of simulation languages. The animation is carried out in two modes, the "concurrent mode" and the "playback mode". In concurrent mode the animation of the system is shown concurrent with the running of the simulation, for which simulation has to be run at a slowed down speed, so as to attain visual comprehension. In playback mode the simulation is first run at its normal speed, and the data for simulation is recorded in a disk file, which is then played to see the animation of the system. The animation is though a very important feature of simulation, which helps in establishing the credibility of the mangers and other users, but it has its demerits and pitfalls too. For details the readers should refer to some text on animation.
7. **Report Presentation capabilities:** The output of the simulation model should be well documented and illustrated. The user should be able to get information of its interest in easily understandable form. It may contain summery of result as well as detailed observations. Software should be able to present the output in the from of graphical displays like bar diagram, pie charts, histograms, and curves etc.

Distributed System -I

Introduction:

From 1945 then modern computer being, until 1985 computers were large & expensive. Starting with 1980, two-advance technology began to change situation.

- 1st Development of powerful microprocessors these were 8-bits machine now 1632 & 64bib CPU become common.
- 2[nd] development was invention of high speed computer networks. LAN (local area networks) allows dozen or even hundreds of machine within a building to be connected in such a way amount of informally can be transferred between in a millisecond or so. The WAN (wide area networks) allow millions of machines all over the earth to be connected at a speed varying from 64 kbps to gigabit per second for some advanced experiment networks.

  The results of these technologies is that it is now not only feasible, but easy to put together computing systems composed of large numbers of CPUs connected by high speed network. They are usually called distributed system in correct of previous centralized system (or single processor system) consisting single CPU in memory peripherals & some terminals.

  Distributed system need radically different software than centralized system.

  A distributed system is a collection of independent computers that appear to the user of the system as a single computer.

  Definition has two aspects one deal with hardware the machines are autonomous & second one deal with software users thinks of the system as a single computers both are essential Goals to put 4fds on one pc possible but points.

Advantages of distributed system over centralized System:

Real driving force behind taken toward decentralization is economics. The most lost effective solution is frequently to harness a large numbers of a cheap cpus to gather in a system. Thus leading reason for the trend toward D.S. is that these systems potentially have a much better price / performance ratio than a single large centralities system have. In effect a D.S. gives more bang for the buck.

Normal performances at low lost or extremely high performance at greater lost D.S. have much to offer.

As an a side some authors make distinction between D.S. which are designed to allow many users to work together and parallel systems whose only goals is to achieve maximum speed up on a single problem.

We prefer to use the term 'D.S.' in the broadest sense to dente any system in which multiple interconnected CPU work together.

Inherently distribution works e. g. super market chain have may stores, each of which gets good delivers locally make local sales and make local decision about which vegetables are so old or rotten that they must be thrown out. It therefore makes sense to keep track of inventory at each store on a local computer rather than centrally at corporate headquarters. After all, most queries and updated will be done locally.

D.S. also called computer supported cooperative work, in which groups of people, located far from each other, are together, for example, to produce a joint report.

Another potential advantage of a distributed system over a centralized system is higher reliability. By distribution the workload over many machines, a single chip failure will bring down at most one machine, leaving the rest intact.

Incremental growth is also potentially a big plus.

Summary of advantages:

Item	Description
Economics	Microprocessors offer a better price /performance than mainframes.
Speed	A distributed system may have more total computing power than a mainframe
Inherent distribution	Some applications involve spatially separated machines.
Reliability	If one machine crashes, the system as a whole can.
Incremental growth	Computing power can be added in small increments.

In the long term, the main driving force will be the existence of large numbers of personal computers and need for people to work together and share information in convenient way without being bothered by geography or the physical distribution of people, data, and machines.

Advantages of D.S. over Independent PCs:

Many users need to share data. For example, airline reservation clerks need access to the master database of flights and existing reservation. Giving each clerk his own private copy of the entire database would not work; since nobody would know which seats the other clerks had already sold. Shared data are absolutely essential to this and many other applications, so the machines must be interconnected. Interconnecting the machines leads to a distributed system.

Expensive peripherals such, as color laser printers, Phototypesetters, and massive archival storage devices are also candidates.

A third reason to connect a grope of isolated computers, into a distributed system is to achieve enhanced person-to-person communication. For many people, electronic mail has numerous attractions over paper mail, telephone, and FAX. It is much faster than paper mail, does not require both parties to be available at the same time as does the telephone, and unlike FAX, produces documents that can be edited, rearranged, stored in the computer, and manipulated with text processing programs.

SUMMERY:

A distributed system is potentially more flexible than giving each user an isolated personal computer. The workload can be spread over the computers more effectively and the loss of a few machines may be compensated for by letting people run their jobs elsewhere.

Item	Description
Data sharing	Allow many user access to a common data base
Device sharing	Allow many users to share expensive peripherals like color printers
Communication	Make human communication easier, for example, by electronic mail
Flexibility	Spread the workload over the available machines in the most cost effective way.

DISADVANTAGES:

- **Software:**

With the current state-of-the-art, we do not have much experience in designing Implementing, and using distributed software. What kind of O.S. programming languages, and applications are appropriate for these systems? How much system do and users do?

- **Communication network:**

  It can lose messages, which requires specials software to be able to recover, and it can become overloaded. When network saturates, it must either be replaced or a second one must be added.

- **Security:**

  Easy access also applies to secrete data.

Feature of Distributed Database System:

1. Collection of data distributed over computers of network.
2. Autonomous process capability of site of each network.
3. Each site could perform local application.
4. Existence of Global application, which requires accessing data at several sites using communication subsystem.
5. Participation of each site in the execution off at least one global application.

   In summarize say a distributed system is result of co-operation between autonomous sites.

Why we need Distributed System?

1. The major reason for going distributed system is its organizational & economics motivation in decentralized organization distributed system is opted.
2. In case of organization where several existing database are to be structured for global application distributed database are more suitable & advantageous over creating new centralized database.
3. Considering fact of feature expansion e.g. opening new branch etc change in database system is essential in compare to centralized v/s distributed system degree of such impact is minimum.
4. Handling local application in case centralized system involves use of communication network thus use increasing overhead cost but in distributed system comprises autonomous system which handle such application on its own which clearly reduces overhead on communication.
5. Multiprocessor systems used in distributed system too enjoy the advantages of parallelism.
6. Through numbers of components used in distributed system may large thus lead to failure increases the effect of such failure do not effect system as whole. Hence complete system crash is very much avoided this is known as graceful degradation property.
7. Further redundant data used here to obtain higher reliability and availability.

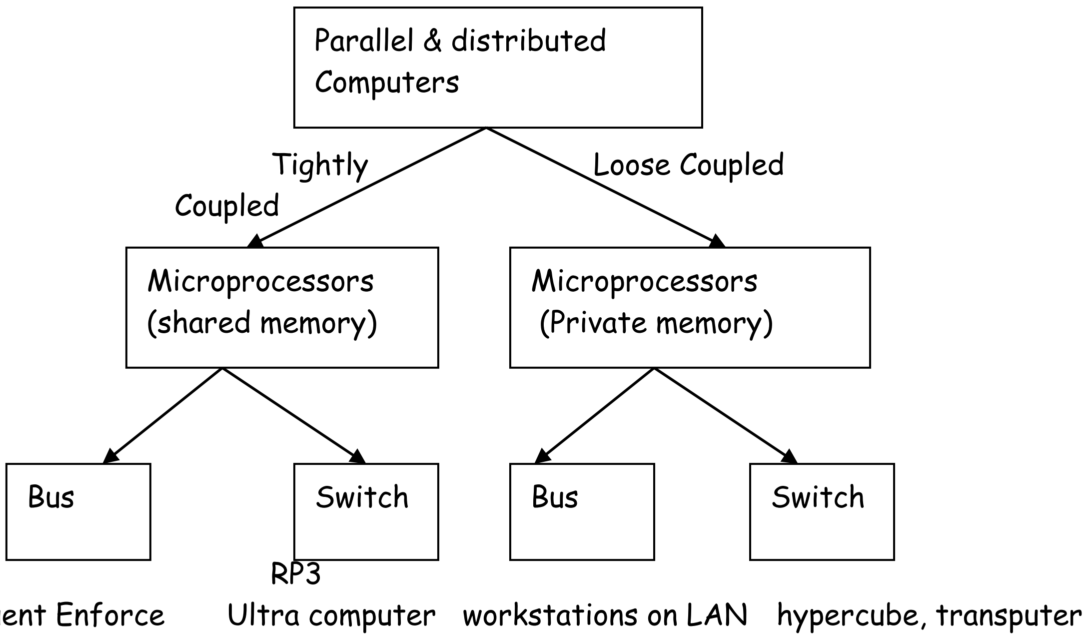
Hardware concepts:

SISD: A computer with a single instruction stream and a single data stream is called SISD. All traditional uniprocessor computers (i.e. those having single CPU) fall in this category, from personnel computer to large mainframes.

SIMD: Single Instruction stream, multiple data stream. This type refers to array processors with one instruction unit that fetches an instruction, then command many data units to carry out in parallel, each of with its own data. These machines are useful for computation that repeats the same calculation on many sets of data, for example adding up all the elements of 64 independent vectors. Some super computers are SIMD.

MISD: Multiple instruction streams, single data stream. No known computers fit this model.

MIMD: Which essentially means of group of independent computers each with its own program counter, program & data. All distributed systems are MIMD, so this classification system is not tremendously use for distributed systems.



MIMD Two category:

- Those that have shared memory called multiprocessors.

  In multiprocessor, there is a single virtual address space is shared by all CPUs. If any Cpu writes e.g. the value 44 to address 1000, any other CPU subsequently reading from its address 1000 will be 44, all share same memory.
- Those have not shared memory called multi computers. : In Multicomputer, every machine has its own memory. If one CPU write value 44 to address 1000, when another CPU read address 1000 it will get whatever value was there before. The write off 44 does not affect its memory at all.

→ Each of these categories can be further divided base don architecture of network Bus and switched.

→By bus we mean that there is single network back plane, bus, cable, or other medium that connects all the machines. Cable television uses a scheme like this; the cable company runs a wire down the street, and all the subscribers have taps running to it from their television sets.

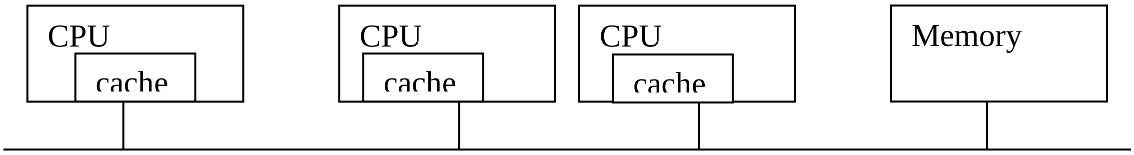
→In switched systems do not have a single backbone like cable television. Instead, there are individual wires from machine to machine, with an explicit switching decision made at each step to route the message along one of the outgoing wires. The worldwide public telephone system is organized in this way.

- Another dimension to our taxonomy is that in some systems the machines are tightly coupled. And in others they are loosely coupled.

Tightly coupled	Loosely coupled
In tightly coupled system, the delay experienced when a message is sent from one computer to another is short, and the data rate is high; that is the number of bits per second that can be transferred is large.	In opposite is true, the inter machine message delay is large and data rate is low.
e.g. two cpu chips on the same printed circuit board and connected by wires etched onto the board are likely to be tightly coupled.	e.g. two computers connected by a 2400 bit/sec modem over the telephone system are certain to be loosely coupled.
Tightly coupled systems tend to be used more as parallel systems (working on a single problem).	Its ones tend to be used as distributed systems (working on many unrelated

	problems)
Multiprocessors tend to be more tightly coupled than Multicomputer, because they can exchange data at memory speeds.	But some fiber optic based Multicomputer can also work at memory speed.

Bus-based multiprocessors:



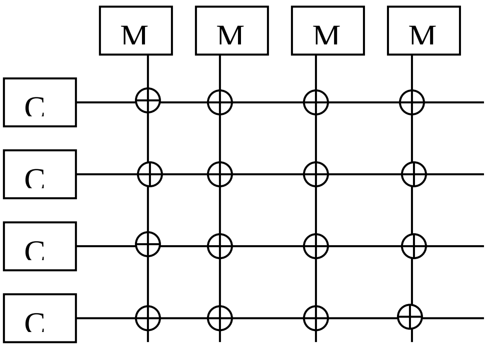
Bus-based multiprocessors consist of some number of CPU's all connected to a common bus, along with a memory module. A simple configuration is to have a high-speed back plane or motherboard into which CPU and memory cards can be inserted.

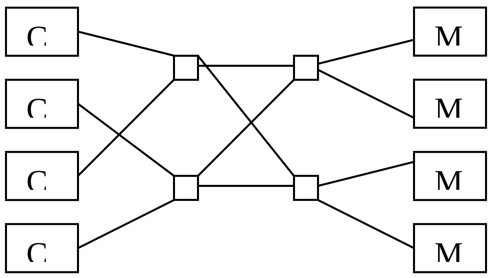
→A typical bus has 32 or 64 address lines, 32 or 64 data lines and perhaps 32 or more control lines, all of which operate in parallel.

Switched multiprocessors:

To build a multiprocessor with more than 64 processors, a different method is needed to connect CPU with the memory.

One possibility is to divide the memory up into modules and connect them to CPU with a crossbar switch. An every intersection is a tiny electronic cross point switch that can be opened & closed with hardware. When a CPU wants to access a particular memory, the cross point switch connecting them is closed momentarily, to allow the access to take place.





Cross point switch n CPU & n memory
Need n^2 cross point switch
Crossbar switch

2*2 switch
Omega switching network

Bus based Multicomputer:

Multicomputer collection of workstation on a LAN. It looks topologically similar to the bus based multiprocessor, but since there will be much less traffic over it. It needs not to be a high-speed back plane bus. In fact, it can be a much lower speed LAN (typically, 10-100 Mbps, compared to 300 Mbps and for a back plane bus)

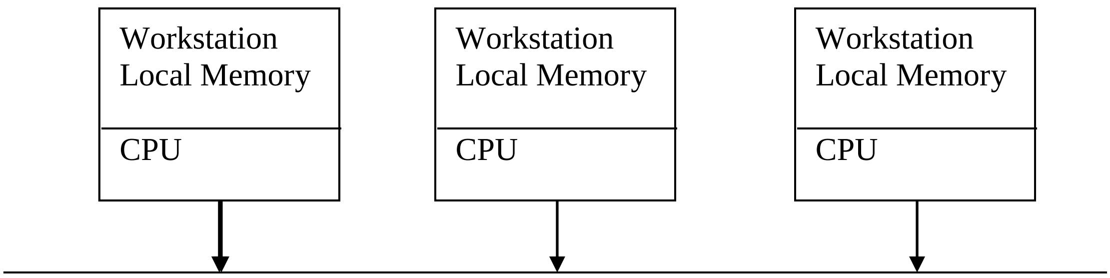
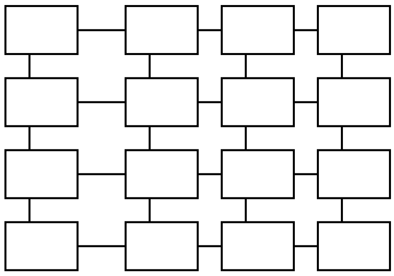
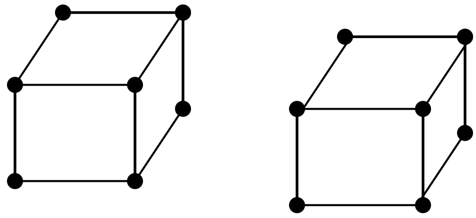


Figure: A Multicomputer consisting of workstations on a LAN.

Switched Multicomputer:





(a) Grid (b) Hypercube

Above fig a and b shows two popular topologies a grid & hypercube.

Grid is easy to understand and lay out on printed circuit boards. They are best suited to problems that have an inherent two-dimensional nature, such as graph theory or vision (e.g., robot eyes or analyzing photographs).

A hypercube is an n-dimensional cube. The hypercube of fig. B is four-dimensional. It can be thought of as two ordinary cubes, each with 8 vertices and 12 edges. Each vertex is a CPU. Each edge is a connected between two CPUs. The corresponding vertices in each of the two cubes are connected.

Hypercube with 1024 CPUs have been commercially available for several years, and hypercube with as many as 16,384 CPUs are starting to become available.

Software concepts:

Although the hardware is important, the software is even more important. The operating system software, not the hardware, largely determines the image that a system presents to its users, and how they think about the system. In this section we will introduce the various types of operating systems for the multiprocessor we have just studied, and discuss which kind of software goes with which kind of hardware.

Loosely coupled software allows machines and users of a distributed system to be fundamentally independent of one another, but still to interact to a limited degree where that is necessary.

Example: A grope of personal computers, each of which has its own CPU, its own memory, its own hard disk and its own operating system, but which share some resources, such as laser printers and data base over a LAN. This system is loosely couple since the individual machines are clearly distinguishable, each with its own job to do. If the network should go down for some reason the individual machines can still continue to run to a considerable degree, although some functionality may be lost (e.g. the ability to print files)

Tightly coupled software:

Example; A grope of personal computers, each of which has its own Cpu, its has own memory, its own hard disk, and its own operating system, but which share some resources, such as laser printers and data bases over a LAN. This system is loosely coupled, since the individual machines are clearly distinguishable, each with its own job to do. If the network should go down for reason, the individual machine can still continue to run to a considerable degree, although some functionality may be lost (e.g., the ability to print files).

Tightly coupled software:

Example: A multiprocessor dedicated to running a single chess program in parallel. Each CPU is assigned a board to evaluate, and it spends its time examining that board and all the boards that can be generated from it. When the evaluation is finished, the CPU reports back the results and are given a new board to work on. The software for this system required to support it, is clearly much more tightly coupled than in our previous example.

Now we have four kind of distributed hardware (Bus-base multiprocessors and switch multiprocessor kind of distributed software (loosely and tightly).

Three operating system:

1 Network operating system

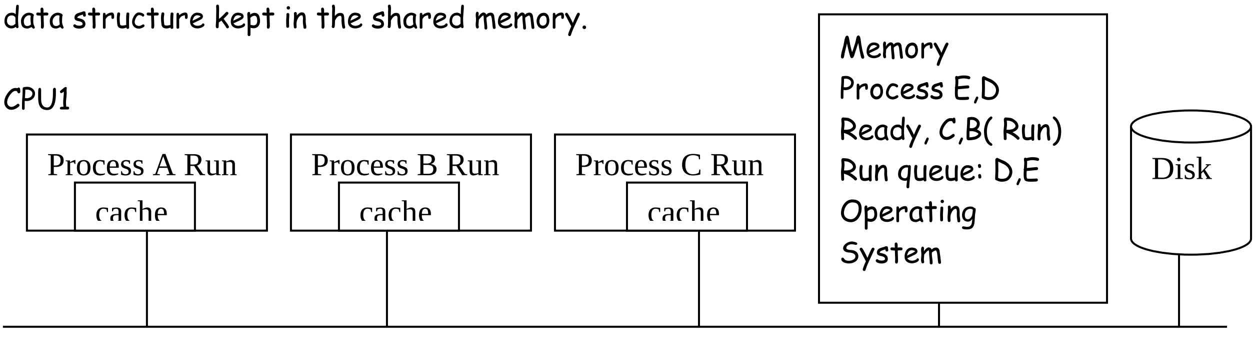
2 True distributed system

3 Multiprocessor (time sharing system) O.S.

Item	Network operating system	Distributed operating system	Multiprocessor operating system
Does it look a virtual uniprocessor?	No	Yes	Yes
Do all have to run the same operating system?	No	Yes	Yes
How many copies of the operating system are there?	N	N	1
How is communication achieved?	Shared files	Messages	Shared memory
Are agreed upon network protocols required?	Yes	Yes	No
Is there a single run queue?	No	No	Yes
Does file sharing have well defined semantic?	Usually no	Yes	Yes

Multiprocessor timesharing systems:

The key characteristic of this class of system is the existence of a single run queue: a list of all the processes in the system that are logically unblocked and ready to run. The run queue is a data structure kept in the shared memory.



A shown above fig. It has three CPUs and five processes that are ready to run. All five processes are located in the shared memory, and three are currently executing: process A on CPU 1, process B on CPU 2, and process C on CPU 3. The other two processes D and E, are also in memory, waiting their turn.

Design issue:

A. Transparency:

How do the system designers fool everyone into thinking that the collection of machines is simply an old-fashioned timesharing system? A system that realizes this goal is often said to be transparent.

Transparency can be achieved at two different levels. Easiest to do is to hide the distribution from the users. For example, when a Unix user types make to recompile a large number of files in a directory, he need not be told that all the compilations are proceeding in parallel on deferent machines and are using a variety of file servers to do it. To him only thing that is unusual is that the performance of the system is halfway decent for a change. In terms of commands issued from the terminal and result displayed on the terminal, the distributed system can be made to look just like a single-processor system.

Kind	Meaning
Location transparency	The users cannot tell where resources are located.
Migration transparency	Resources can move at will without changing their names
Replication transparency	The users cannot tell how many copies exist.
Concurrency transparency	Multiple users can share resources automatically
Parallelism transparency	Activities can happen in parallel without users knowing.

B. Flexibility:

The second key design issue is flexibility. It is important that the system be flexible.

The monolithic kernel:

The monolithic kernel is basically today's centralized operating system augmented with networking facilities and the integration of remote services. Trapping to the kernel, having the work performed there, and having the kernel return the desired result to the user process makes most system calls. With this approach, most machines have disks and manage their own local file systems.

It basically provides just four minimal services:

1. An inter process communication mechanism.
2. Some memory management.
3. A small amount of process management and scheduling.
4. Low-level input/output.

It does not provide the file system, directory system, full process management, or much system call handling. The services that the micro kernel does provide are included because they are difficult or expensive to provide anywhere else. The goal is to keep it small.

All the other operating system services are generally implemented as user level servers. To look up a name, read a file, or obtain some other service, the user sends a message to the appropriate server, which then does the work and returns the result.

The advantage of above method is that it is highly modular: there is a well-defined interface to each service and every service is equally accessible to every client, independent of location. In additional it is easy to implement, install, and debug new service since adding or changing a service does not require stopping the system and booting a new kernel, as is the case with a monolithic kernel. It is precisely this ability to add, delete, and modify services that gives the micro kernel its flexibility. Furthermore, users who are not satisfied with any of the official services are free to write their own.

The advantage of monolithic kernel is performance.

C. Reliability

One of the original goals of building distributed systems was to make them more reliable than single-processor system. The idea is that if a machine goes down, some other machine takes over the job. In other words, theoretically the overall system reliability could be the Boolean or of the component reliabilities.

Example:

With four file servers, each with a 0.95 change of being up at any instant, the probability of all for being down simultaneously is 0.05^4=0.000006, so the probability of at least one being available is 0.999994, far better than of any individual server.

A highly reliable system must be highly available, but that is not enough. Data entrusted to the system not be lost or garbled in any way, and if files are stored redundantly on multiple servers, all the copies must be kept consistent.

In general the more copies that are kept, the better the availability, but the grater the chance that they will be inconsistent especially if updates are frequent. The designers of all distributed systems must keep this dilemma in mind all the time.

Another aspect of overall reliability is security. Files and other resources must be protected from unauthorized usage. Another issue relating to reliability is fault tolerance.

In general, distributed systems can be designed to mask failure that is to hide them from the users. If a file service or other service is actually constructed from a group of closely cooperating servers, it should be possible to construct it in such a way that users do not notice the lost of one trick is to arrange this cooperation so that it does not add substantial overhead to the system in the normal case, when everything is functioning correctly.

D. Performance:

Building a transparent, flexible, reliable distributed system will not win you any prizes if it is as shown as molasses. In particular, when running a particular application on a distributed system, it should not be appreciably worse than running the same application on a single processor. Unfortunately, achieving this is easier said than done.

Various performance metrics can be used. Response time is one but so are throughput (number of jobs per hour), system utilization and amount of network capacity consumed. To obtain optimize performance; one often has to minimize the number of massages.

The difficulty with this strategy is that the best way to gain performance is to have many activities running in parallel on different processors, but doing so requires sending many messages.

One possible way out is to pay considerable attention to the gain size of all computation. Starting up a small computation remotely, such as adding two integers, is rarely worth it, because the communication overhead dwarfs the extra CPU cycles gained.

On other hand, starting up a long compute-bound job remotely may be worth the trouble. In general, jobs that involve a large number of small computations may cause to exhibit fine gained parallelism. On the other hand, jobs that involve large computations low interaction rates, and little data that is coarse-grained parallelism, may be better fit.

Good reliability is often best often best achieved by having several sever closely cooperating on a single request.

E. Scalability:

For a distributed systems are guiding principle is clear: avoid centralized components, tables and algorithms.

Concept	Example
Centralized components	A single mail for all users
Centralized tables	A single on-line telephone book
Centralized algorithms	Doing routing based on complete information

Potential bottlenecks that designers should try to avoid in very large

Only decentralized algorithms should be used. These algorithms generally have the following characteristics, which distinguish them from centralize algorithms:

1 No machine has complete information about the system state.
2 Machines make decisions based only on local information.
3 Failure of one machine does not ruin the algorithm.
4 There is no implicit assumption that a global clock exists.

Summary

Distributed systems are consisting of autonomous CPUs that work together to make the complete system look like a single computer.

They have number of potential selling points, including good proce/performance ratios the ability to match distributed applications well potentially high reliability, and incremental growth as the workload grows.

They have some disadvantages such as more complex software, potential communication bottlenecks, and weak security. Nevertheless, there is considerable interest worldwide in building and installing them.

Modern computer systems often have multiple CPUs. These can be organized as multiprocessors (with shared memory) or as Multicomputer (without shared memory). Both type can be bus-based or switched. The former tend to be tightly coupled, while the latter tend to be loosely coupled.

The software for multiple CPU systems can be divided into three rough classes. Network operating systems allow users at independent workstations to communicate via a shared file systems but other wise leave each user as the master of his own workstation.

Distributed operating systems turn the entire collection of hardware and software into a single integrated system, much like a tradition timesharing system.

Share memory multiprocessors also offer a single system image, but do so by centralizing everything, so there really is only a single system.

Shared memory multiprocessors are not distributed systems.

Distributed systems have to be designed carefully, since there are many pitfalls for the unwary. A key issue is transparency-hiding all the distribution form the users and even from the application programs. Another issue is flexibility. Since the field is only now in its infancy, the design should be made with the idea of making future changes easy. In this respect, micro kernels are superior to monolithic kernels. Other important issues are reliability, performance, and scalability.