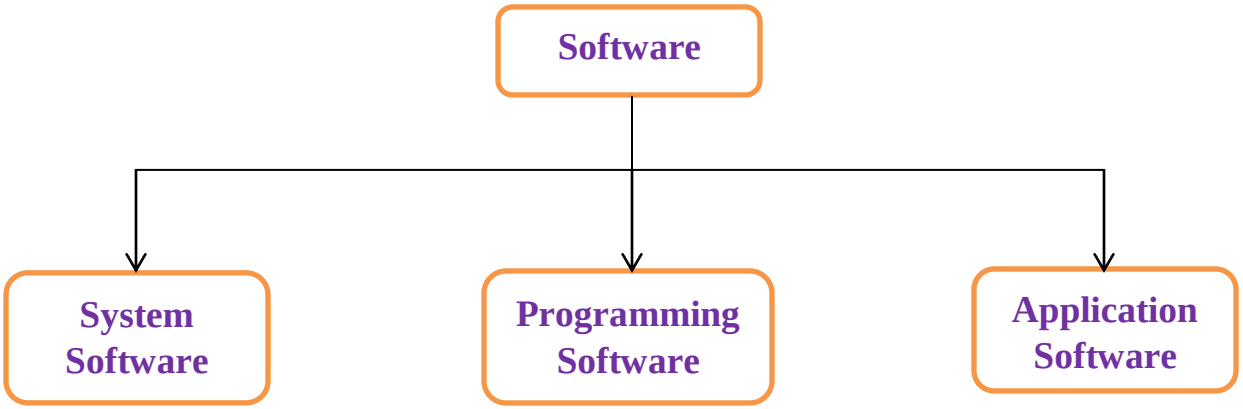
Rajmani Kumar,
Lecturer, Dept. of BCA
S.U.College, Hilsa (Nalanda)
Patliputra University, Patna

BCA-1st Year **Paper-I**

Computer Software

Software

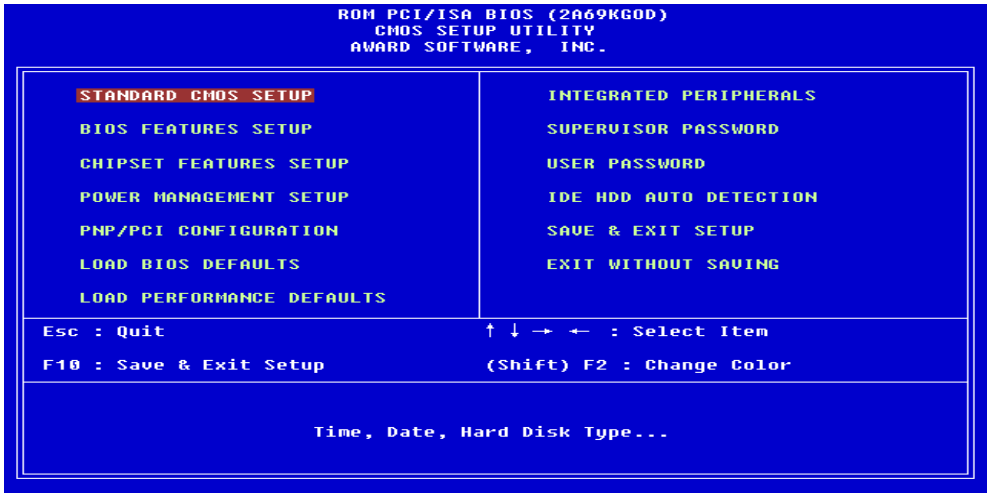
- **Computer software**, or just **software**, is a collection of computer programs and related data that provides the instructions for telling a computer what to do and how to do it.
- Any **set of instructions** that guides the hardware and tells it how to accomplish each task.



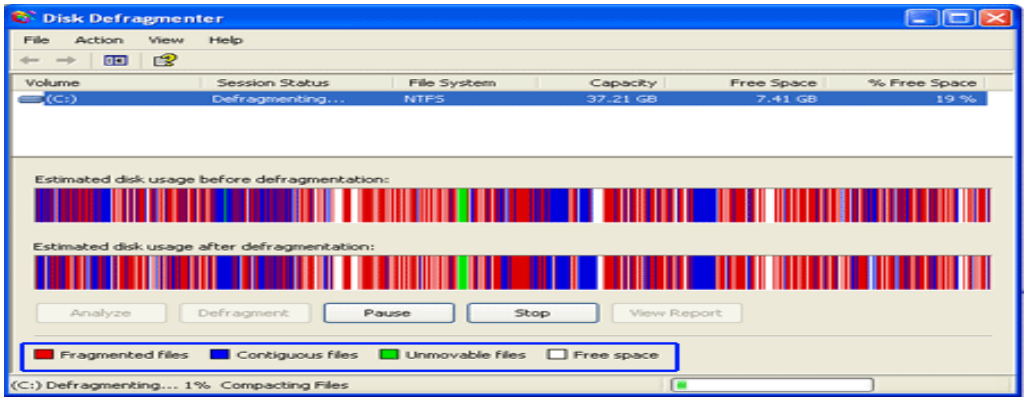
System Software

- System software is computer software designed to operate the computer hardware to provide basic functionality and to provide a platform for running application software.
- Refers to the operating system and all utility programs that manage computer resources at a low level.
- The BIOS(basicinput/outputsystem) gets the computer system started after you turn it on and manages the dataflow between the operating system and attached devices such as the harddisk, video adapter, keyboard, mouse, and printer.
- The boot program loads the operating system into the computer's main memory or random access memory(RAM).

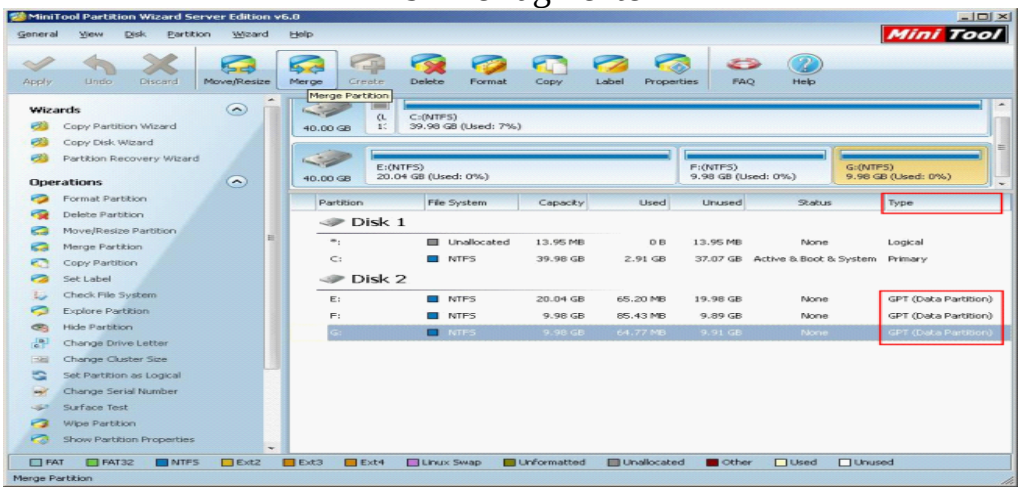
- System software also includes system utilities, such as the disk defragmenter and System Restore.



Basic Input/OutputSystem (BIOS) Program



Disk Defragmenter



Disk Partition Tool

Programming Software

- Programming software include tools in the form of programs or applications that software developers use to **create, debug, maintain**, or otherwise support other programs and applications.
- Thetermusuallyreferstorelativelysimpleprogramssuchas**compilers**,**debuggers**,**interpreters**,**linkers**,and**texteditors**,

Example of programming language:

–C

–C++

–C#

–BASIC

–JAVA

–VisualBasic

–Phyton

–HTML

–PHP

Application Software

- A program or group of programs designed for end users
- Allow send users to accomplish one or more specific (non-computer related) tasks.

Examples of Computer Application Software

•Word processor

•Spreed sheet

•Presentation Software

•Database Management System

•Desktop Publisher

•Graphic Editor

•Web Browser

Machine language

Machine Language is the language written as strings of binary 1`s and 0`s. It is the only language which a computer understands without using a translation program.

A machine language instruction has two parts. The first part is the operation code which tells the computer what function to perform and the second part is the operand which tells the computer where to find or store the data which is to be manipulated. A programmer needs to write numeric codes for the instruction and storage location of data.

Disadvantages –

- It is machine dependant i.e. it differs from computer to computer.
- It is difficult to program and write
- It is prone to errors
- It is difficult to modify

Assembly Language

It is a low level programming language that allows a user to write a program using alphanumeric mnemonic codes, instead of numeric codes for a set of instructions.

It requires a translator known as assembler to convert assembly language into machine language so that it can be understood by the computer. It is easier to remember and write than machine language.

Assembler – It is a computer program which converts or translates assembly language into machine language. It assembles the machine language program in the main memory of the computer and makes it ready for execution.

Advantages –

It is easy to understand and use
It is easy to locate and correct errors
It is easier to modify

Disadvantages –

It is machine dependant

High level Language

It is a machine independent language. It enables a user to write programs in a language which resembles English words and familiar mathematical symbols. COBOL was the first high level language developed for business.

Each statement in a high level language is a micro instruction which is translated into several machine language instructions.

Compiler

A **compiler** is a translator program which translates a high level programming language into equivalent machine language programs. It compiles a set of machine language instructions for every high level language program.

Source code: It is the input or the programming instructor of a procedural language.

The compiler translates the source code into machine level language which is known as object code. Object code can be saved and executed as and when desired by the user.

Source Code → Language Translator Program → Object code

High level language → Machine level language

Linker:

A program used with a compiler to provide links to the libraries needed for an executable program. It takes one or more object code generated by a compiler and combines them into a single executable program.

Interpreter:

It is a translator used for translating high level language into the desired output. It takes one statement, translates it into machine language instructions and then immediately executes the result. Its output is the result of program execution.

Advantages of High level Language –

- It is machine independent
- It is easier to learn and use
- It is easier to maintain and gives few errors

Disadvantages –

- It lowers efficiency
- It is less flexible

Centralized System

We start with centralized systems because they are the most intuitive and easy to understand and define.

Centralized systems are systems that use client/server architecture where one or more client nodes are directly connected to a central server. This is the most commonly used type of system in many organizations where client sends a request to a company server and receives the response.

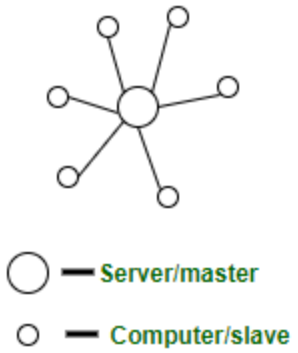


Figure – Centralised system visualisation

Characteristics of Centralized System –

- **Presence of a global clock:** As the entire system consists of a central node(a server/ a master) and many client nodes(a computer/ a slave), all client nodes sync up with the global clock(the clock of the central node).
- **One single central unit:** One single central unit which serves/coordinates all the other nodes in the system.
- **Dependent failure of components:** Central node failure causes entire system to fail. This makes sense because when the server is down, no other entity is there to send/receive response/requests.

Scaling –

Only vertical scaling on central server is possible. Horizontal scaling will contradict the single central unit characteristic of this system of a single central entity.

Components of Centralized System –

Components of Centralized System are,

- Node (Computer, Mobile, etc.).
- Server.
- Communication link (Cables, Wi-Fi, etc.).

Architecture of Centralized System –

Client-Server architecture. The central node that serves the other nodes in the system is the server node and all the other nodes are the client nodes.

Limitations of Centralized System –

- Can't scale up vertically after a certain limit – After a limit, even if you increase the hardware and software capabilities of the server node, the performance will not increase appreciably leading to a cost/benefit ratio < 1.
- Bottlenecks can appear when the traffic spikes – as the server can only have a finite number of open ports to which can listen to connections from client nodes. So, when high traffic occurs like a shopping sale, the server can

essentially suffer a Denial-of-Service attack or Distributed Denial-of-Service attack.

Advantages of Centralized System –

- Easy to physically secure. It is easy to secure and service the server and client nodes by virtue of their location
- Smooth and elegant personal experience – A client has a dedicated system which he uses(for example, a personal computer) and the company has a similar system which can be modified to suit custom needs
- Dedicated resources (memory, CPU cores, etc)
- More cost efficient for small systems upto a certain limit – As the central systems take less funds to set up, they have an edge when small systems have to be built
- Quick updates are possible – Only one machine to update.
- Easy detachment of a node from the system. Just remove the connection of the client node from the server and voila! Node detached.

Disadvantages of Centralized System –

- Highly dependent on the network connectivity – System can fail if the nodes lose connectivity as there is only one central node.
- No graceful degradation of system – abrupt failure of the entire system
- Less possibility of data backup. If the server node fails and there is no backup, you lose the data straight away
- Difficult server maintenance – There is only one server node and due to availability reasons, it is inefficient and unprofessional to take the server down for maintenance. So, updates have to be done on-the-fly(hot updates) which is difficult and the system could break.

Applications of Centralized System –

- Application development – Very easy to setup a central server and send client requests. Modern technology these days do come with default test servers which can be launched with a couple commands. For example, express server, django server.
- Data analysis – Easy to do data analysis when all the data is in one place and available for analysis
- Personal computing

Use Cases –

- Centralized databases – all the data in one server for use.
- Single player games like Need For Speed, GTA Vice City – entire game in one system(commonly, a Personal Computer)
- Application development by deploying test servers leading to easy debugging, easy deployment, easy simulation
- Personal Computers

Organisations Using –
National Informatics Center (India), IBM

DECENTRALIZED SYSTEMS:

These are another type of systems which have been gaining a lot of popularity, primarily because of the massive hype of Bitcoin. Now many organisations are trying to find the application of such systems.

In decentralized systems, every node makes its own decision. The final behavior of the system is the aggregate of the decisions of the individual nodes. Note that there is no single entity that receives and responds to the request.

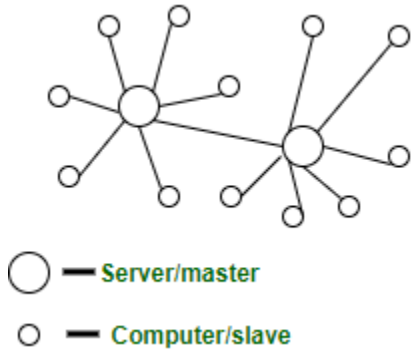


Figure – Decentralised system visualisation

Characteristics of Decentralized System –

- **Lack of a global clock:** Every node is independent of each other and hence, have different clocks that they run and follow.
- **Multiple central units (Computers/Nodes/Servers):** More than one central unit which can listen for connections from other nodes
- **Dependent failure of components:** one central node failure causes a part of system to fail; not the whole system

Scaling –

Vertical scaling is possible. Each node can add resources(hardware, software) to itself to increase the performance leading to increase in performance of the entire system.

Components –

Components of Decentralized System are,

- Node (Computer, Mobile, etc.)
- Communication link (Cables, Wi-Fi, etc.)

Architecture of Decentralized System –

- peer-to-peer architecture – all nodes are peers of each other. No one node has supremacy over other nodes
- master-slave architecture – One node can become a master by voting and help in coordinating of a part of the system but this does not mean the node has supremacy over the other node which it is coordinating

Limitations of Decentralized System –

- May lead to problem of coordination at the enterprise level – When every node is owner of its own behavior, its difficult to achieve collective tasks
- Not suitable for small systems – Not beneficial to build and operate small decentralized systems because of low cost/benefit ratio
- No way to regulate a node on the system – no superior node overseeing the behavior of subordinate nodes

Advantages of Decentralized System –

- Minimal problem of performance bottlenecks occurring – The entire load gets balanced on all the nodes; leading to minimal to no bottleneck situations
- High availability – Some nodes(computers, mobiles, servers) are always available/online for work, leading to high availability
- More autonomy and control over resources – As each node controls its own behavior, it has better autonomy leading to more control over resources

Disadvantages of Decentralized System –

- Difficult to achieve global big tasks – No chain of command to command others to perform certain tasks
- No regulatory oversight
- Difficult to know which node failed – Each node must be pinged for availability checking and partitioning of work has to be done to actually find out which node failed by checking the expected output with what the node generated
- Difficult to know which node responded – When a request is served by a decentralised system, the request is actually served by one of the nodes in the system but it is actually difficult to find out which node indeed served the request.

Applications of Decentralized System –

- Private networks – peer nodes joined with each other to make a private network.
- Cryptocurrency – Nodes joined to become a part of a system in which digital currency is exchanged without any trace and location of who sent what to whom. However, in bitcoin we can see the public address and amount of bitcoin transferred, but those public addresses are mutable and hence difficult to trace.

Use Cases –

- Blockchain
- Decentralized databases – Entire database split in parts and distributed to different nodes for storage and use. For example, records with names starting from ‘A’ to ‘K’ in one node, ‘L’ to ‘N’ in second node and ‘O’ to ‘Z’ in third node
- Cryptocurrency

Organisations Using –

Bitcoin, Tor network

DISTRIBUTED SYSTEMS:

This is the last type of system that we are going to discuss. Lets head right into it In decentralized systems, every node makes its own decision. The final behaviour of the system is the aggregate of the decisions of the individual nodes. Note that there is no single entity that receives and responds to the request.

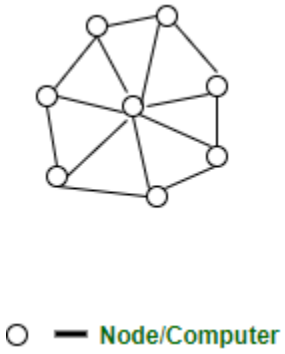


Figure – Distributed system visualisation

Example –

Google search system. Each request is worked upon by hundreds of computers which crawl the web and return the relevant results. To the user, the Google appears to be one system, but it actually is multiple computers working together to accomplish one single task (return the results to the search query).

Characteristics of Distributed System – :

- **Concurrency of components:** Nodes apply consensus protocols to agree on same values/transactions/commands/logs.
- **Lack of a global clock:** All nodes maintain their own clock.
- **Independent failure of components:** In a distributed system, nodes fail independently without having a significant effect on the entire system. If one node fails, the entire system sans the failed node continue to work.

Scaling –

Horizontal and vertical scaling is possible.

Components of Distributed System –

Components of Distributed System are,

- Node (Computer, Mobile, etc.)
- Communication link (Cables, Wi-Fi, etc.)

Architecture of Distributed System –

- peer-to-peer – all nodes are peer of each other and work towards a common goal
- client-server – some nodes are become server nodes for the role of coordinator, arbiter, etc.
- n-tier architecture – different parts of an application are distributed in different nodes of the systems and these nodes work together to function as an application for the user/client

Limitations of Distributed System –

- Difficult to design and debug algorithms for the system. These algorithms are difficult because of the absence of a common clock; so no temporal ordering of commands/logs can take place. Nodes can have different latencies which have to be kept in mind while designing such algorithms. The complexity increases with increase in number of nodes. Visit this link for more information
- No common clock causes difficulty in the temporal ordering of events/transactions
- Difficult for a node to get the global view of the system and hence take informed decisions based on the state of other nodes in the system

Advantages of Distributed System –

- Low latency than centralized system – Distributed systems have low latency because of high geographical spread, hence leading to less time to get a response

Disadvantages of Distributed System –

- Difficult to achieve consensus
- Conventional way of logging events by absolute time they occur is not possible here

Applications of Distributed System –

- Cluster computing – a technique in which many computers are coupled together to work so that they achieve global goals. The computer cluster acts as if they were a single computer
- Grid computing – All the resources are pooled together for sharing in this kind of computing turning the systems into a powerful supercomputer; essentially.

Use Cases –

- SOA-based systems
- Multiplayer online games