6.1 OPEN SOURCE

6.1.1 Open Source

- The term open source refers to something which can be modified and share because its design is publicly accessible.
- The term originated in the context of software development namé to a specific approach to creating computer programs. Today, however, "**open source**" means a broader set of values-what we call "**the open source way.**"
- Open source projects, products, or initiatives accepts and celebrate principles of open exchange, collaborative participation, rapid prototyping, transparency, meritocracy, and community-oriented development.

6.1.2 Open Source Operating System

GQ. 6.1.1 What is Open Source ? Explain open source Operating System.

- An operating system, for example Windows or Android or MAC OS, has many, many lines of code.
- On a closed-source OS like Windows, that code can only be modified by Microsoft, and only viewed by few selected customers like big companies. In contrast, the code the open-source operating systems is freely available for anyone to view and also to modify, use and share, under the terms of open-source licenses such as MIT, GNU Public License, and Apache 2.0.
- Allowing anyone to inspect the source code in this way has many advantages. Those with sufficient technical knowledge can customize the OS and fix problems as they arise by delving into the source code.
- It also allows the development of open-source operating systems to be community-led, or at least influenced, with technical users suggesting changes to an open-source operating system's codebase.

6.1.3 Most Popular Open Source Operating Systems

The most popular open-source operating system :

(a) **Android :** which is based on a modified version of the Linux kernel. The issue is that what many people think of as Android is not wholly open-source. At the core of the Android OS is the Android Open Source Project, which is open, as the name suggests, but built around that are closed-source Google-branded apps.

(b) **Chromium OS :** the open-source operating system designed around running web apps. While not widely distributed in on its own, Chromium OS forms the basis of the Chrome OS that runs on Chromebooks, which are particularly popular in the education market.

(c) **Linux :** based operating systems still only occupy between one to three percent share of the desktop PC market. Among that share are a mix of old favorites such as Ubuntu and Debian and newer challengers such as Linux Mint and Elementary OS.

6.1.4 Other Open Source Operating System

(a)	Cosmos	(b)	FreeDOS	(c)	Genode
(d)	Ghost OS	(e)	ITS (Incompatible time-sharing)		
(f)	OSv	(g)	Phantom OS		

6.1.5 Benefits of using Open Source Software

Open source licensing doesn't involve copyright restrictions. A relative freedom of use brings advantages to such products, which attracted many users. The main benefits of software with a publicly available source code are :

- **Flexibility :** The software can be customized to meet specific business needs. Engineers can write more code to add an extra functionality and vice versa - delete unnecessary parts.
- **Stability :** This product can be used for long-term projects with confidence because it won't disappear from the market or become outdated if its authors stop working on it. The user community will take care of an open source software.

Module 6

- **Security and reliability :** Numerous people with different skill levels may work on the same software, which may lead to code inconsistency. That's when the culture of open source is beneficial. Other developers from around the world can review, fix, and update this code. The faster code review, the more secure and reliable the software is. Authors and users improve a solution because they need it to perform well.
- **Easier evaluation :** What you (or your developers) see is what you get. Full transparency of a source code allows your team to examine and evaluate a product learning about its capabilities and flaws.
- **Better support :** As an OSS user, you have more ways of getting technical advice and support : from a vendor, a consultancy company specializing in this exact product, or from other users that are ready to share their experience and knowledge across forums or mailing lists.
- **Possible savings :** Such products may have a download price and generally require billable customer support. But let's keep in mind that the purchase cost for an open source software is usually lower than for a commercial one and free open source programs also exist.

6.1.6 Types of Open Source License

There is a range of open source licenses available to software creators. While they all follow the main principles and criteria of open source software, they differ slightly in the extent to which they allow users to modify the source code, and under which conditions. Some of the most popular licenses are :

(a) MIT license
(b) GNU General Public license (GPL) 2.0./3.0
(c) Apache license 2.0
(d) Common Development and Distribution license 1.0 (CDDL-1.0)
(e) BSD licenses

6.1.7 Open Source Companies and Programmers has Monetary Gain

GQ. 6.1.2 How Open Source Operating System companies and Programmer make monetary Gain ?

These are the few ways how Open Source Companies and Programmer make money

(a) Software as a Service (Open SaaS)
(b) Paid support
(c) Dual licensing
(d) Paid extra features or functionalities
(e) Paid certification
(f) **Earning by Customization of Code**
(g) **Earning By Providing Support**

(a) Software as a Service (OpenSaaS)

- SaaS nice way because the software is stored in the cloud; users only need a web browser to access an application.
- SaaS is a popular business model for vendors that build tools for HR, collaboration, content management, and project management.
- With an OpenSaaS model, software is purchased via subscriptions, which can offer varying levels of service.
- For example, you might offer technical support, software customization, and trainings as package options. WordPress and Sharetribe are two examples of OpenSaaS products.

(b) Paid support

- Many OSS companies provide extra services like technical support, certifications, and trainings.
- Many open source companies - including Red Hat, JBoss, and MySQl – have built their entire business by giving free solutions. Profits are made from additional services.
- MySQL, also generates revenue from selling support subscriptions for their product.

(c) Dual licensing

- Dual licensing allows companies to release commercial software (with a commercial license) that's derived from free OSS commonly distributed under the GNU General Public (GPL) license.
- Dual licensing can be implemented in a few ways.
- A company releases identical products under a commercial license and under a free license like GPL. Also company releases different versions under different licenses.
- The GPL license allows end users to run OSS, redistribute that software, and modify it. But one cannot embed OSS solutions in proprietary (commercial) software and cannot make profit under a GPL license. Hence a commercial version of an open source product is released to sell commercial software and make profit out of it.
- The most well-known example of successful dual licensing is MySQL. The company releases MySQL Classic Edition and MySQL Community Edition – under the GPL license.

(d) Paid extra features or functionalities

- Some companies distribute their software for free, but additional features, functionalities, or updates are chargeable.
- Customers feel most comfortable paying for only the services they utilize, Customers can save money on deployment and troubleshooting when these services are included in paid packages.
- For Example, GitLab distributes their developer tools in three editions. Their enterprises version includes premium support, file locking and advanced solutions for remote teams, and is billed per user.

(e) Paid certification

- Companies offer certification for popular software as an opportunity for specialists who want to validate their knowledge and skills.
- Certification is popular software has many benefits like.
  - o To differentiate yourself among other specialists with the same skills.
  - o Developers realize the importance of networking with mentors and groupmates.
  - o A developer's certification leads lends them additional professional credibility and even promotions, and can boost a company's image.
- Open source giants including Magento and Red Hat offer a variety of certification options.

(f) Earning by Customization of Code

- Companies using open-source software, can get customized version by hiring programmer as per company's needs.
- Also it is favorable for companies to hire professional for modification who have already worked on the code rather than hiring new programmer who will study and modify the code.
- This saves time though a little overhead is added by way of payment to such programmers.

(g) Earning By Providing Support

- Open-source software sometimes may not be easy to install and use. Company can train their staff and provide paid support.
- Some programmer/companies give software free and open but has many parts hidden. The need for installation and training becomes necessary which are paid.

6.1.8 Advantages and Disadvantages of Open Source Operating System

Advantages of Open Source Software

1. Cheaper than goods which are sold commercially.
2. Open source technology is increasingly helping enterprise owners conserve according to studies.

Module 6

3. Big and well-established tech firms have the financial resources to recruit the industry's finest talent to create their products.
4. There are two key explanations for making stable open source applications. First of all, they are mainly created by experienced and skilled professionals who do their best to produce high-quality programme.
5. Because you're not bound to a single system, you don't have to abide by a particular IT architecture that may allow you to constantly update the applications, and even hardware.

Disadvantages of Open Source Software

1. Many users have access to open source tech source code but not everybody has positiive intentions.
2. Although certain people are using their ability to find defects and make changes to the software, some are using this opportunity to hack the limitations of the system and build glitches that can corrupt devices, steal identity or even annoy many users.
3. However, there are many applications that are developed solely to fulfil the needs of the inventor and to bring his thoughts to life.

6.1.9 Open Source Operating System LINUX

Linux is one of popular version of UNIX operating System. It is open source as its source code is freely available. It is free to use. Linux was designed considering UNIX compatibility. Its functionality list is quite similar to that of UNIX.

(A) Components of Linux System

Linux Operating System has primarily three components :

1. **Kernel :** Kernel is the core part of Linux. It is responsible for all major activities of this operating system. It consists of various modules and it interacts directly with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.
2. **System Library :** System libraries are special functions or programs using which application programs or system utilities accesses Kernel's features. These libraries implement most of the functionalities of the operating system and do not requires kernel module's code access rights.
3. **System Utility :** System Utility programs are responsible to do specialized, individual level tasks.

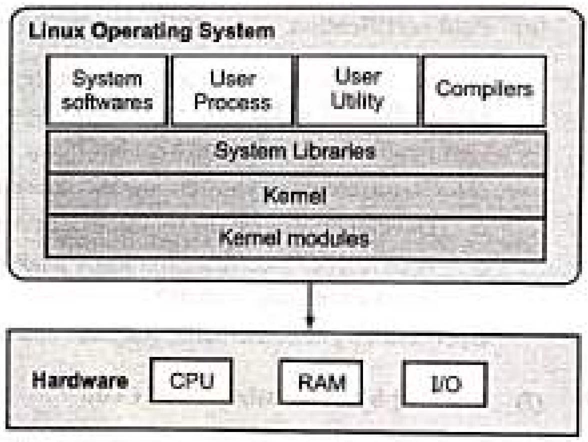

Fig. 6.1.1 : Components of Linux Operating System

(B) Kernel Mode vs User Mode

- Kernel component code executes in a special privileged mode called **kernel mode** with full access to all resources of the computer.
- This code represents a single process, executes in single address space and do not require any context switch and hence is very efficient and fast. Kernel runs each processes and provides system services to processes, provides protected access to hardware to processes.
- Support code which is not required to run in kernel mode is in System Library. User programs and other system programs works in **User Mode** which has no access to system hardware and kernel code. User programs/ utilities use System libraries to access Kernel functions to get system's low level tasks.

6.1.10 Basic Features of Linux Operating System

Following are some of the important features of Linux Operating System :

- **Portable :** Portability means software can works on different types of hardware in same way. Linux kernel and application programs supports their installation on any kind of hardware platform.

- **Open Source :** Linux source code is freely available and it is community based development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
- **Multi-User :** Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
- **Multiprogramming :** Linux is a multiprogramming system means multiple applications can run at same time.
- **Hierarchical File System :** Linux provides a standard file structure in which system files/ user files are arranged.
- **Shell :** Linux provides a special interpreter program which can be used to execute commands of the operating system. It can be used to do various types of operations, call application programs. etc.
- **Security :** Linux provides user security using authentication features like password protection/ controlled access to specific files/ encryption of data.

6.1.11 Architecture of Linux

The following illustration shows in Fig. 6.1.2 the architecture of a Linux system.

The architecture of a Linux System consists of the following layers :

- **Hardware layer :** Hardware consists of all peripheral devices (RAM/ HDD/ CPU etc).
- **Kernel :** It is the core component of Operating System, interacts directly with hardware, provides low level services to upper layer components.
- **Shell :** An interface to kernel, hiding complexity of kernel's functions from users. The shell takes commands from the user and executes kernel's functions.
- **Utilities :** Utility programs that provide the user most of the functionalities of an operating systems.

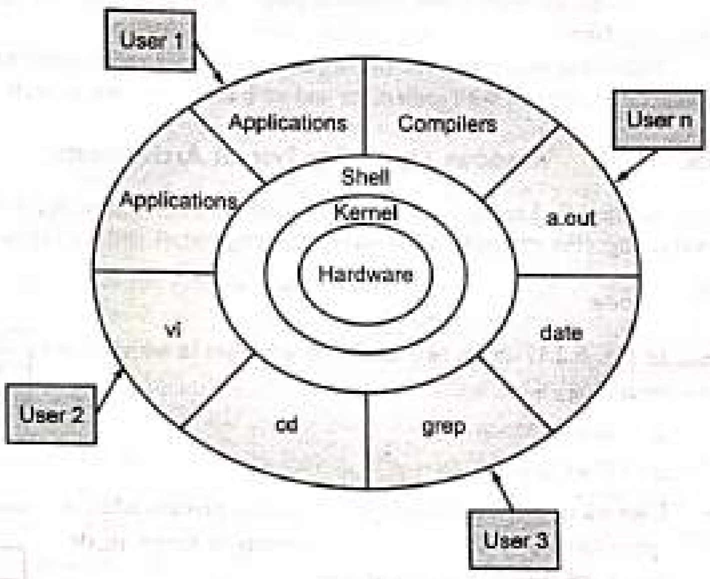

Fig. 6.1.2 : Architecture of Linux

6.2 PROPRIETARY OPERATING SYSTEM

6.2.1 Introduction

The something owned by a specific company or individual is a Proprietary; it is an adjective that describes. In the computing world, the software which are not open source and not freely distributed is called as licensed or proprietary. Examples include operating systems, software programs, and file formats.

- Proprietary software, another kind of software that you have to bough if you want to use it.
- This software belongs to someone. The code is closed, commercial and copyrighted, its use is limited at some point, especially when it is referred to distribution or modification. This software can be modified within creators limits

(A) Proprietary Operating Systems

- Proprietary operating systems cannot be modified by users or other companies. For example, Windows and OS X are both proprietary OSes.
- The Windows source code is owned by Microsoft and the OS X source code is owned by Apple. Other companies can make programs that run on these operating systems, but they cannot modify the OS itself.
- Linux and Android are not proprietary, which is why many different versions of these operating systems exist.

(B) Software Programs

- Proprietary software programs are applications in which all rights are retained by the developer or publisher. They are typically closed-source, meaning the developer does not provide the source code to anyone outside the company.

- Proprietary programs are licensed to end users under specific terms defined by the developer or publisher. These terms often restrict the usage, distribution, and modification of the software.
- Most commercial software is proprietary because it gives the developer a competitive advantage.

(C) File Formats

- Some file types are saved in a proprietary file format that can only be recognized by a specific program. These file types are typically created by proprietary software programs and are usually saved in a binary (rather than a text-based) format.
- By storing data in a proprietary format, developers can ensure that files created with their software cannot be opened with other software programs. Proprietary file types saved by software programs are also called native files.

Following topics provide the discussion about internal organization of windows, its various components and how these components interact with each other and with user programs as well.

6.2.2 Windows Operating System Architecture

A simplified version of this architecture is shown in Fig. 6.2.1. Keep in mind that this diagram is basic - it doesn't show everything. (For example, the networking components and the various types of device driver layering are not shown.)

User Mode

- In Fig. 6.2.1, shows two modes of windows in which it is operating , those are
  - Kernel Mode
  - User Mode
- User-mode threads execute in a protected process address space (although while they are executing in kernel mode, they have access to system space).
- Thus, system support processes, service processes, user applications, and environment subsystems each have their own private process address space.

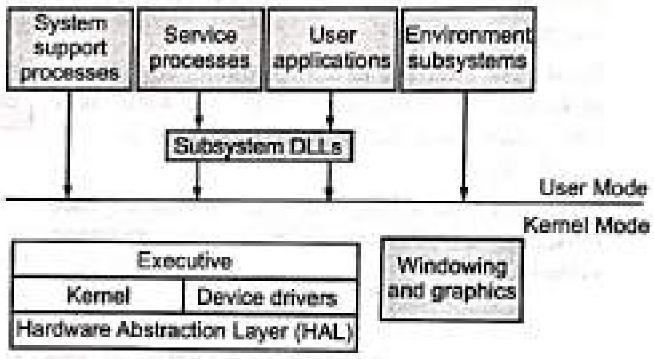

Fig. 6.2.1 : Simplified windows Architecture

(a) User Mode

The four basic types of user-mode processes are described as follows :

- Fixed (or hardwired) system support processes
- Service processes that host Windows services
- User applications,
- Environmental subsystem server processes
- **Fixed (or hardwired)** ***system support processes***, such as the logon process and the Session manager, that are not Windows services. That is, they are not started by the service control manager.
- **Service processes that host Windows services**, such as the task Scheduler and Print Spooler services. Services generally have the requirement that they run independently of user logons. Many Windows server applications, such as Microsoft SQL Server and Microsoft Exchange Server, also include components that run as services.
- **User applications**, which can be one of the following types: 32-bit or 64-bit, Windows 3.1 16-bit, MS-DOS 16-bit, or POSIX 32-bit or 64-bit. Note that 16-bit applications can be run only on 32-bit Windows.
- **Environmental subsystem server processes**, which implement part of the support for the operating system environment, or personality, presented to the user and programmer.

- Windows NT originally shipped with three environment subsystems:
  - o Windows,
  - o POSIX, and
  - o OS/2.
- However, the POSIX and OS/2 subsystems last shipped with Windows 2000. The Ultimate and Enterprise editions of Windows client as well as all server versions include support for an enhanced POSIX subsystem call Subsystem for Unix-based Applications (SUA).
- In Fig. 6.2.1, notice the "Subsystem DLLs" box below the "Service processes" and the "User applications" boxes. Under Windows, user applications don't call the native Windows operating system services directly; rather, they go though one or more *subsystem dynamic-link libraries* (DLLs).
- The role of the subsystem DLLs is to translate a documented function into the appropriate internal (and generally undocumented) native system service calls. This translation might or might not involve sending a message to the environment subsystem process that is serving the user application.

(b) Kernel Mode

Th kernel-mode components of Windows include the following :

- The Windows *executive* contains the base operating system services, such as memory management, process and thread management, security, I/O, networking, and Interprocess communication.
- The **Windows *kernel*** consists of low-level operating system functions, such as thread scheduling, interrupt and exception dispatching, and multiprocessor synchronization. It also provides a set of routines and basic objects that the rest of the executive uses to implement higher-level constructs.
- ***Device drivers*** include both hardware device drivers, which translate user I/O function calls into specific hardware device I/O requests, as well as nonhardware device drivers such as file system and network drivers.
- The ***hardware abstraction layer*** (HAL) is a layer of code that isolates the kernel, the device drivers, and the rest of the Windows executive from platform-specific hardware differences (such as differences between motherboards).
- **The *windowing and graphics system*** implements the graphical user interface (GUI) functions (better known as the Windows USER and GDI functions), such as dealing with windows, user interface controls, and drawing.

Table 6.2.2 lists the file names of the core Windows operating system components.

Table 6.2.2 : Core Windows System Files

File Name	Components
Ntoskrnl.exe (32-bit systems only)	Executive and kernel, with support for Physical Address Extension (PAE), which allows 32-bit systems to address up to 64 GB of physical memory and to mark memory as nonexecutable.
Hal.dll	Hardware abstraction layer
Win32k.sys	Kernel-mode part of the Windows subsystem
Ntdll.dll	Internal support functions and system dispatch to execute functions
Kernel32.dll,Advapi32.dll, User32.dll,Gdi32.dll	Core Windows subsystem DLLs

6.2.3 Advantages and Disadvantages of Proprietary Software's

(A) Advantages

1. Developed professionally and carefully tested
2. Support is provided to keep customers happy and keep them using the software
3. Books, magazines, articles and online tutorials give advice on how to use it
4. Updates and bug fixes meet the needs and suggestions of users

(B) Disadvantages

1. User licenses apply conditions on how the software can be used
2. The source code can't be modified by users
3. The person or organization who made it still has exclusive control
4. It is developed for many users and may not meet individual needs
5. It is not free
6. Support and updates may be expensive

6.3 FUNDAMENTALS OF DISTRIBUTED OPERATING SYSTEM

UQ. 6.3.1 Write short note on Distributed Operating System. **MU(IT) - Dec. 19, 10 Marks**

- Distributed systems make a convenient medium to share resources, speed up computation, and improve data availability and reliability. Distributed computing is provided by a distributed OS.
- The distributed system poses many challenges in terms of this constrained distributed environment. The issues related to the single-processor machine, such as mutual exclusion, deadlocks, process-scheduling, and so on, introduce numerous difficulties for implementation of distributed computation.

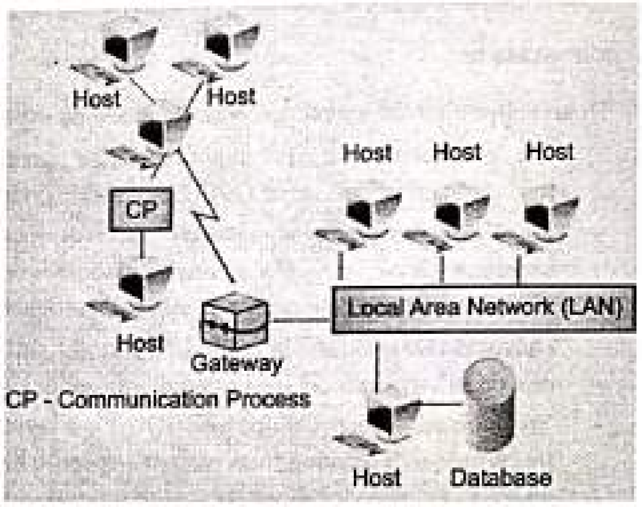

Fig. 6.3.1 : Distributed System

6.3.1 Characteristics of Distributed Operating System

Distributed systems are multi processor systems, but with the following differences :

- Each node in a distributed system is a complete computer, having a full set of peripherals, including memory, communication hardware, possibly different OS and different file system, and so on.
- The users of a distributed system have an impression that they are working on a single machine.
- The distributed system is a network of workstations, wherein the system might have a single file system that facilitates all the files accessible from all the machines in the same way.
- A distributed system, as a whole, may look and act like it is a single-processor time-sharing system, and the user working on it has the perception that the work is being done on a single machine.

6.3.2 Reasons for usage of Distributed System

(A) Economy

- Distributed systems, comprising micro-processors, have a better price performance ratio as compared to a single centralized system of the mainframe type.
- A number of cheap processors together, resulting in a distributed system, provide a highly cost-effective solution for a computation-intensive application.

(B) Resource sharing

- It is the main motive behind distributed systems. If we want to take advantage of hundreds of processors, all of them have to be on a single board, which may not be possible.
- But the multiple processors are realized as a single powerful machine in a network system, and the resulting machine is known as a distributed system.
- In this way, a number of users can share the resources of all the machines on the distributed system.

(C) Computational speed-up

- Besides resource sharing, distributed systems also provide computational speed-up, by partitioning a computation into sub-computations, which are distributed and run concurrently on various nodes on the system.
- It results in higher throughput and provides rapid response time in computing user tasks. Thus, a distributed system enhances the performance of a system.

(D) Reliability

- Since a distributed system may utilize an alternative path in the network, in case of any failure, it provides reliable services and resources.
- A distributed system also provides enhanced availability of resources through redundancy of resources and communication paths, thereby increasing the reliability of the system.
- For example, a distributed file system places files on separate machines and allows many users to access the same set of files reliably, providing the view of a single file system.

(E) Communication

The users in a distributed system are able to exchange information at geographically-distant nodes, and may collaborate on a project with various communication facilities.

(F) Incremental growth

The computational power of a distributed system is dynamic. It may be increased with the introduction of any new hardware or software resources. Thus, modular expandability is possible in the distributed system.

6.3.3 Distributed Operating System

- Distributed OSs are tightly-coupled OS software on loosely-coupled hardware, that is, distributed system. These OSs, providing distributed computing facility, employ almost same communication methods and protocols, as in network OSs.
- But the communication is transparent to the users, such that they are unaware of the separate computers that are providing the service. A distributed OS has control of all the nodes in the system.
- When a user starts working on a node,
  - o The control functions like process creation,
  - o Resource allocation and so on are performed on behalf of the distributed OS.
  - o With the facility of distributed OS, a complex user computation may be divided into sub-computations.
  - o Each sub-computation may be allocated different nodes for computation.
- The user does not need to log into any other node for computation. The user will get the result on his/her own node, where the job was initiated, without knowing the details of sub-computation or their respective computation on various nodes.
- Similarly, when a user needs a resource, it can be allocated to it without the user having to know the location from where it has been provided. This is known as a single-system image or virtual uni-processor to the users.

Module 6

- Thus, the users in a distributed OS are not aware of the identities of nodes and locations of various resources in the system. It is possible only due to the unified control of the distributed OS.

6.3.4 Important tasks to be performed by a distributed OS

- Since distributed systems need to access any resource or transfer any task on any node, there are three types of migration provided by the OSs :
  (i) **Data migration :** Transferring the data from one site to another site.
  (ii) **Computation migration :** Transferring the computation on a particular node.
  (iii) **Process migration :** The process or its sub-processes may also need to be transferred to some other nodes, due to some reasons like load-balancing, computation speed, and so on.
- Distributed OS must provide the means for inter-process communication. Some methods are :
  (i) **Remote Procedure Call :** A process, on one node, may invoke a function or procedure in a process executing on another node.
  (ii) **Remote Method Invocation :** This allows a Java process to invoke a method of an object on a remote machine.
  (iii) **Common Object Request Broker Architecture (CORBA) :** It is a standardized language that supports different programming languages and different OSs for distributed communication.
  (iv) **Distributed Component Object Model (DCOM) :** This is another standard developed by Microsoft, included in the Windows OS.
- Due to multiple processes, synchronization methods should be supported by the OS.
- There may be a deadlock, when processes distributed over several nodes in a network wait for resources not released by other processes. Therefore, deadlock should also be handled by the distributed OS.

6.3.5 Issues Related of Distributed Operating System

In a distributed system, the resources and computations are distributed across various nodes in the system. If there is an imbalance of these two on the nodes, it may affect the performance of the system. Therefore, to protect the features of the distributed system, and for the convenience of the users, the following design issues must be implemented.

(a) Transparency

The most important feature of a distributed system is that it should have a single system or uniprocessor image. The implementation of this feature comes in the form of transparency. The transparency is to hide the distribution and location details from the user. The transparency can be implemented in the following forms :

(I) **Location Transparency :** The users in this type are not aware of where the hardware or software components are located. The use will know only the name of the resource and the name should not be able to reveal its location in the system.

(II) **Migration Transparency :** Along with the location transparency, the users must not be affected, if the location of resources is changed in the system. But the name of the resources must not be changed. This is known as migration transparency.

(III) Replication Transparency

- The OS may take the decision of making additional duplicate copies of the files and other resources, for the sake of fault tolerance in the system. But this replication of files and other resources is not visible to the user.

(IV) Concurrency Transparency

- Since multiple users may be allowed to work on a distributed system, it may be possible that more than one user may try to access the same resource at the same time. This is known as concurrency transparency.

(V) Parallelism Transparency

- Since the computation of a user task may be divided into multiple sub-computations, each of these sub-computations can be computed in parallel, on various nodes of the system.
- But all these details of division of the computation and their execution on various nodes are transparent to the user.

(b) Global Knowledge

- Since in a distributed system, each node is a complete computer system, there is no global memory, and even no global clock.
- Moreover, there are unpredictable message delays. Therefore, it becomes difficult to have up-to-date information about the global state of a distributed computing system.
- Therefore, in the design of a distributed OS, the design of some efficient techniques to implement system-wide control in distributed systems is required. Moreover, in the absence of a global clock, it is difficult to order all the events that occur at different times at different nodes in the system.

(c) Performance

In a distributed system, while designing, various performance parameters should be considered.

- **Communication** is typically quite slow in the distributed computing system, due to message communication among nodes. If we are distributing the computation over multiple nodes in the system, then the number of message communication also increases, thereby, causing slow communication. Therefore, the need is to minimize the number of messages. This depends on the grain size of computations.
- If there are multiple but small computations known as ***fine-grained computations***, which have been distributed over the nodes and interact highly among themselves, then the distributed system will behave poorly.
- In contrast, if multiple but large computations, known as ***coarse-grained computations***, are distributed over the nodes and their interaction rate is too low, then the distributed system will perform much better. Thus, coarse-grained based parallelism enhances the performance of a distributed computing system.
- Another performance factor may be the ***load-balancing*** in the system. Since the computations are distributed on the nodes of the system, there may be the case that some nodes are heavily loaded, while some nodes may have only few computations. This imbalance of nodes may also cause poor performance of the system. Thus, load-balancing is required, such that all the nodes in the system are equally loaded.
- To improve the performance, migration techniques must be implemented. There are three types of migration provided by distributed OSs :
  (i) **Data migration :** Transferring the data from one site to another site.
  (ii) **Computation migration :** Transferring the computation on a particular node.
  (iii) **Process migration :** The process or its sub-processes may also need to be transferred to some other nodes, due to some reasons like load-balancing, computation speed, and so on.
- Thus, the distributed OS must be able to do process/computation migration as well as data migration, for the better performance of the system.

(d) Reliability

- Since the reliability is a major goal behind designing a distributed system, as compared to a centralized system, the distributed system must take care of reliability issues also. Reliability is achieved through two properties of data:
  - Availability and
  - Reliability.
- Availability means resources must be available, despite the failure of some components in the system, thereby, increasing the reliability of the system. The availability may be achieved through fault tolerance techniques.

6.3.6 Process Synchronization

- Since there is no shared memory in the distributed systems, it is difficult to synchronize the processes executing at different nodes of the system.
- The distributed OS must be able to synchronized various processes when they try to access a shared resource concurrently, that is, mutual exclusion must be there while one process accesses the shared resource.
- Since in a distributed system, the processes may request and release resources locally or remotely, the sequence of their request and release may not be known in advance. This may lead to deadlocks in the system. The deadlocks must also be removed as soon as possible; otherwise, the system's performance may degrade.

6.4 NETWORK OPERATING SYSTEM

UQ. 6.4.1 Write Short Note on Network Operating System. **MU(IT) - Dec. 19, Q. 6(b), 10 Marks**

- The distributed system needs an OS software layer which helps in coordinating all activities to be performed on the network system.
- Thus, a network OS can be considered as a loosely-coupled OS software on a loosely-coupled hardware that allows nodes and users of a distributed system to be independent of one another, but with limited interaction.
- In a network, each node has its own local OS. A user as a node works as on the local machine with the help of local OS.
- The user working on a node is also able to perform non-local functions i.e. remote function. For example, the user may remotely log on to some other node. Also, the user may transfer the files to another node as well.
- The processes of the user instructs the network OS. If the operation to be performed on the node is local, the network OS passes the request to the local OS on the node.
- But if the operation to be performed is remote, the network OS on that node instructs the network OS on the node, which needs to be contacted for the operation.

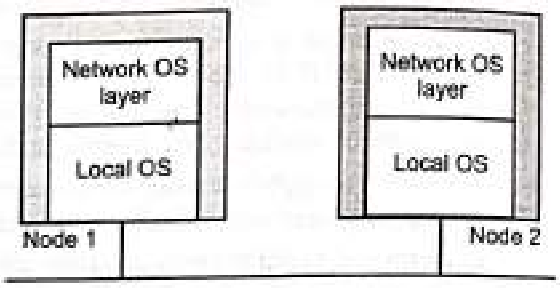

Fig. 6.4.1 : Network operating System

- A network OS also performs the resource-sharing across multiple nodes of the network, where each node has its own local OS and a layer of network OS as shown in Fig. 6.4.1.
- Thus, in a network OS-based system, a user must know where a resource is located, in order to use it, leading to poor transparency of system resources and services.

6.4.1 Characteristics of Network Operating System

- Network operating system (NOS) has ability to detect the different H/W and shared data over several machines. NOS must support several applications and third party as well.
- It contains the all possible security features of operating system. Users can manage all remote terminals.
- Network operating system can support the internet functionality and their connection management as well.
- In NOS, administrator can control the different setting such as Network security protocol, data backup for single and interconnected computer system.
- Printer and other applications can be shared.
- Users can use their backup database and web services.
- Directory and name services management. It has ability of Internetworking like as routing and WAN ports.
- It is capable of clustering.
- NOS has ability to support for all processors, application, and their hardware.
- All user s can create user account, and they can manage user logging in and logging out.

6.4.2 Types of Network Operating System

There are mainly two types of Network Operating System, they are :

(A) Peer-to-Peer (B) Client-Server

(A) Peer-to-Peer

- **Peer-to-Peer Network Operating System is an operating system in which all the nodes are functionally and operationally equal to each other.** No one is superior or inferior.
- They all are capable to perform similar kinds of tasks. All the nodes have their own local memory and resources.

- Using the Network O.S., they can connect and communicate with each other. They can also share data and resources with one another.
- One node can also communicate and share data and resources with a remote node in the network by using the authentication feature of the Network O.S.
- The nodes are directly connected with each other in the network with the help of a switch or a hub.

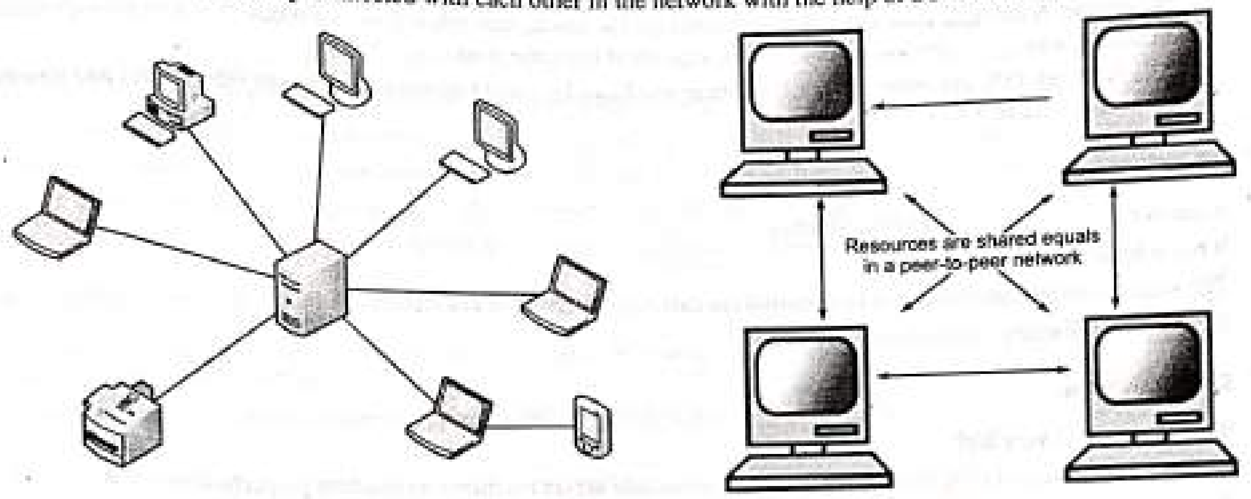

Fig. 6.4.2 : Peer-to-Peer network

Advantages

1. Easy to install and setup.
2. The setup cost is low.
3. There is no requirement for any specialized software.
4. The sharing of information and resources is fast and easy.

Disadvantages

1. The performance of autonomous computers may not be so good when sharing some resources.
2. There is no centralized management.
3. It is less secure.
4. It does not have backup functionalities.
5. There is no centralized storage system.

(B) Client-Server

- The Client-Server Networking Operating System operates with a single server and multiple client computers in the network.

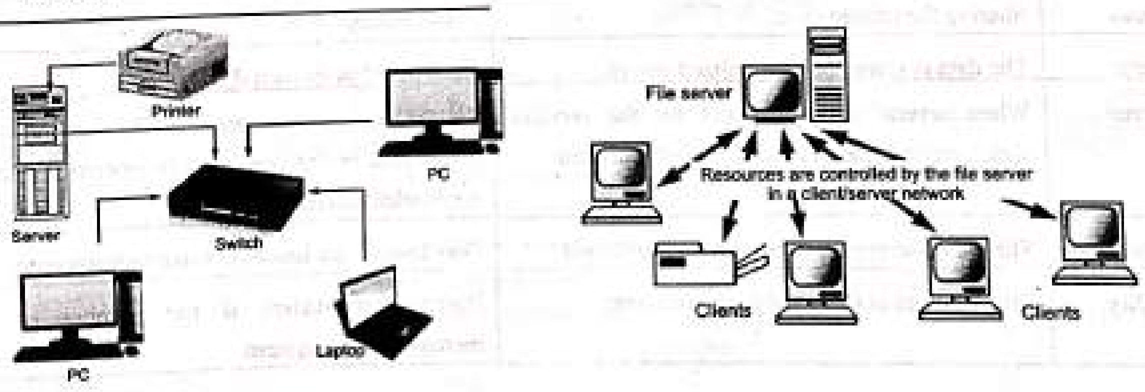
Printer
PC
Server
Switch
PC
Laptop
File server
Resources are controlled by the file server
in a client/server network
Clients
Clients

Fig. 6.4.3 : Client-Server Network

- The Client O.S. runs on the client machine, while the Network Operating System is installed on the server machine. The server machine is a centralized hub for all the client machines. The client machines generate a request for information or some resource and forward it to the server machine.
- The server machine, in turn, replies to the client machine by providing appropriate services to it in a secure manner. The server machine is a very powerful computer, that is capable of tackling large calculations and operations. It can also have the ability to administer the whole network and its resources. It can be multiprocessing in nature, which can process multiple client requests at the same time.
- The Network O.S. enhances the reach of client machines by providing remote access to other nodes and resources of the network in a secure manner.

Advantages

1. It has centralized control and administration.
2. It has a backup facility for lost data.
3. The shared data and resources can be accessed concurrently by multiple clients.
4. It has better reliability and performance.

Disadvantages

1. The setup cost is very high.
2. There is a requirement of specialized software for client and server machines to function properly.
3. There is a need for an administrator to administer the network.
4. There may be network failure, in case of central server failure.
5. A huge amount of client requests may overload the server.

6.4.3 Difference between Peer-to-Peer and Client Server Network Operating System

Table 6.4.1

Basis for comparison	Client-server	Peer-to-peer
Basic	There is a specific server and specific clients connected to the server.	Clients and server are not distinguished; each node act as client and server.
Service	The client request for service and server respond with the service.	Each node can request for services and can also provide the services.
Focus	Sharing the information.	Connectivity.
Data	The data is stored in a centralized server.	Each peer has its own data.
Server	When several clients request for the services simultaneously, a server can get bottlenecked.	As the services are provided by several servers distributed in the peer-to-peer system, a server in not bottlenecked.
Expense	The client-server are expensive to implement.	Peer-to-peer are less expensive to implement.
Stability	Client-Server is more stable and scalable.	Peer-to-Peer suffers if the number of peers increases in the system.

6.5 EMBEDDED OPERATING SYSTEMS

Q. 6.5.1 Write short note on Embedded Operating System.

Embedded systems run on the computers that control devices they that are not generally computers or computing devices and which do not accept user-installed software.

An embedded operating system is embedded and specifically configured for a certain hardware configuration.

Examples of Embedded systems are microwave ovens, TV sets, cars, DVD recorders, traditional phones, and MP3 players. An embedded system can be part of a different kind of another embedded system. Examples include computers in cars, traffic lights, digital televisions, ATMs, airplane controls, point of sale (POS) terminals, digital cameras, GPS navigation systems, elevators, digital media receivers and smart meters, among many other possibilities.

eral Definition of Embedded system

A combination of computer hardware and software, and perhaps additional mechanical or other parts, designed to perform a dedicated function.

In many cases, embedded systems are part of a larger system or product, as in the case of an antilock braking system in a car.

Embedded systems far outnumber general-purpose computer systems, encompassing a broad range of applications as shown in Table 6.5.1.

6.5.1 Requirements and Constraints of Embedded System

Small to large systems, implying very different cost constraints, thus different needs for optimization and reuse.

Relaxed to very strict requirements and combinations of different quality requirements, for example, with respect to safety, reliability, real-time, flexibility, and legislation.

Short to long life times.

Different environmental conditions in terms of, for example, radiation, vibrations, and humidity.

Different application characteristics resulting in static versus dynamic loads, slow to fast speed, compute versus interface intensive tasks, and/or combinations thereof different models of computation ranging from discrete-event systems to those involving continuous time dynamics (usually referred to as hybrid systems).

Constraints, such as required speeds of motion, required precision of measurement, and required time durations, dictate the timing of software operations. If multiple activities must be managed simultaneously, this imposes more complex real-time constraints.

Table 6.5.1 : Examples of Embedded System and Their Market

Sr. No.	Market	Embedded Device
1.	Automotive	– Ignition system – Engine control – Brake system
2.	Consumer electronics	– Digital and analog televisions – Set-top boxes (DVDs, VCRs, Cable boxes) – Personal digital assistants (PDAs) – Kitchen appliances (refrigerators, toasters, microwave ovens) – Automobiles – Toys/games – Telephones/cell phones/pagers – Cameras – Global positioning systems

Sr. No.	Market	Embedded Device
3.	Industrial control	– Robotics and controls systems for manufacturing – Sensors
4.	Medical	– Infusion pumps – Dialysis machines – Prosthetic devices – Cardiac monitors
5.	Office automation	– Fax machine – Photocopier – Printers – Monitors – Scanners

- Fig. 6.5.1 shows in general terms an embedded system organization. In addition to the processor and memory, there are a numbers of elements that differ from the typical desktop or laptop computer.
- There may be a variety of interfaces that enable the system to measure, manipulate, and otherwise interact with the external environment.
- The human interface may be as simple as a flashing light or as complicated as real-time robotic vision.
- The diagnostic port may be used for diagnosing the system that is being controlled-not just for diagnosing the computer.
- Special-purpose field programmable (FPGA), application specific (ASIC), or even nondigital hardware may be used to increase performance or safety.
- Software often has a fixed function and is specific to the application.

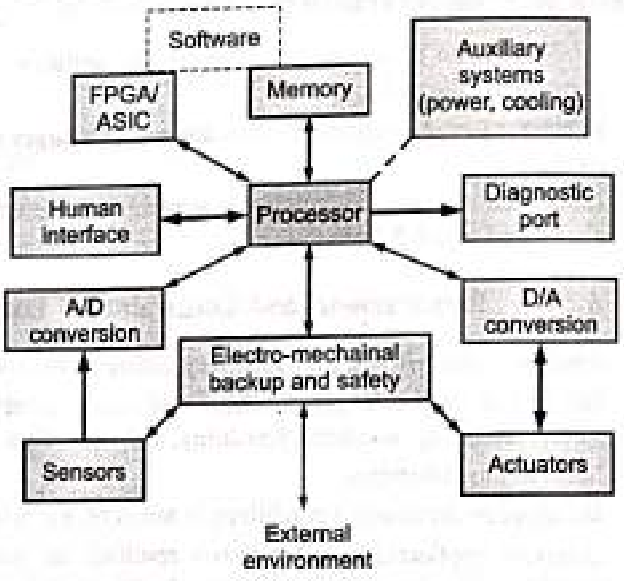

Fig. 6.5.1 : Organization of Embedded System

6.5.2 Definition of the Embedded Operating Systems

An embedded system is a device with a computer designed for a specific purpose. To achieve that, the device needs an operating system that can respond fast and is prepared to keep working in any event. That is why we cannot rely on a general-use OS, but an embedded operating system. An embedded operating system is an OS designed and optimized to :

- Improve the efficiency of managing the hardware resources
- Reduce response times specifically for the task the device was created for some examples of embedded systems are industrial robots, smart devices, IoT machines, drones, medical systems, video game consoles, and many others.

6.5.3 Operating Systems for General-Purpose vs Embedded operating Systems

GQ. 6.5.2 Compare and contrast general purpose and embedded operating System.

Table 6.5.2

Sr. No.	Features	Embedded Operating System	General Purpose Operating System
1.	Primary Goal	Run a single application	Run a Many applications
2.	App Distribution	Generally, the application and the system are distributed as a single image	The applications and the operating System are distributed separately

Sr. No.	Features	Embedded Operating System	General Purpose Operating System
3.	Architecture	Designed for a specific Purpose	Designed for general use to solve all kinds of task
4.	Processing	MCUs and CPUs	CPUs
5.	Examples	Raspbian, NOOB, windows 10-IOT core	Linux, Windows, macOS
6.	Application	ATMs, Satellite Navigation systems, Missile Guidance, Gaming Consoles	Desktop Computer

6.5.4 Characteristics of Embedded Operating Systems

- A simple embedded system is controlled by a special purpose program or set of programs with no other software. Complex embedded systems include an OS.
- Although it is possible in principle to use a general-purpose OS, such as Linux, for an embedded system, constraints of memory space, power consumption, and real-time requirements typically dictate the use of a special-purpose OS designed for the embedded system environment.

(a) Characteristics and design requirements for embedded operating systems

- **Real-time operation :** In many embedded systems, the correctness of a computation depends, in part, on the time at which it is delivered. Often, real-time constraints are dictated by external I/O and control stability requirements.
- **Reactive operation :** Embedded software may execute in response to external events. If these events do not occur periodically or at predictable intervals, the embedded software may need to take into account worst-case conditions and set priorities for execution of routines.
- **Configurability :** Because of the large variety of embedded systems, there is a large variation in the requirements, both qualitative and quantitative, for embedded OS functionality. Thus, an embedded OS intended for use on a variety of embedded systems must lend itself to flexible configuration so that only the functionality needed for a specific application and hardware suite is provided. The linking and loading functions can be used to select only the necessary OS modules to load. Conditional compilation can be used. If an object-oriented structure is used, proper sub classes can be defined. However, verification is a potential problem for designs with a large number of derived tailored operating systems.
- **I/O device flexibility :** There is virtually no device that needs to be supported by all versions of the OS, and the range of I/O devices is large. It makes sense to handle relatively slow devices such as disks and network interfaces by using special tasks instead of integrating their drives into the OS kernel.
- **Streamlined protection mechanisms**
  - Untested programs are generally not added to the software. After the software has been configured and tested, it can be assumed to be reliable.
  - Thus, apart from security measures, embedded systems have limited protection mechanisms. For example, I/O instructions need not be privileged instructions that trap to the OS; tasks can directly perform their own I/O. Similarly, memory protection mechanisms can be minimized.
- **Direct use of interrupts :** General-purpose operating systems typically do not permit any user process to use interrupts directly. Three reasons why it is possible to let interrupts directly start or stop tasks (e.g., by storing the task's start address in the interrupt vector address table) rather than going through OS interrupt service routines:
  - Embedded systems can be considered to be thoroughly tested, with infrequent modifications to the OS or application code;
  - Protection is not necessary, as discussed in the preceding bullet item; and
  - Efficient control over a variety of devices is required.
- There are two general approaches to developing an embedded OS.
  - The first approach is to take an existing OS and adapt it for the embedded application.
  - The other approach is to design and implement an OS intended solely for embedded use.

Module 6

6.5.5 Purpose-Built Embedded Operating System

A significant number of operating systems have been designed from the ground up for embedded applications. Two prominent examples of this latter approach are eCos and TinyOS,.

Typical characteristics of a specialized embedded OS include the following :

- Has a fast and lightweight process or thread switch
- Scheduling policy is real time and dispatcher is part of scheduler instead of separate component.
- Smaller in Size
- Responds to external interrupts quickly, generally response time of less than 10 μs.
- Minimizes intervals during which interrupts are disabled.
- Provides fixed or variable sized partitions for memory management as well as the ability to lock code and data in memory.
- Provides special sequential files that can accumulate data at a fast rate.

To deal with timing constraints, the kernel

- Provides bounded execution time for most primitives
- Maintains a real-time clock
- Provides for special alarms and timeouts
- Supports real-time queuing disciplines such as earliest deadline first and primitives for jamming a message into the front of a queue.
- Provides primitives to delay processing by a fixed amount of time and to suspend/resume execution.

For complex embedded systems, the requirement may emphasize predictable operation over fast operation, necessitating different design decisions, particularly in the area of task scheduling.

6.6 CLOUD OPERATING SYSTEMS

- "The cloud" refers to servers that are accessed over the Internet, and the software and databases that run on those servers. Cloud servers are located in data centers all over the world.
- By using cloud computing, users and companies don't have to manage physical servers themselves or run software applications on their own machines.
- The cloud enables users to access the same files and applications from almost any device, because the computing and storage takes place on servers in a data center, instead of locally on the user device.
- This is why a user can log into their Instagram account on a new phone after their old phone breaks and still find their old account in place, with all their photos, videos, and conversation history.
- It works the same way with cloud email providers like Gmail or Microsoft Office 365, and with cloud storage providers like Dropbox or Google Drive.
- The cloud can also make it easier for companies to operate internationally, because employees and customers can access the same files and applications from any location.

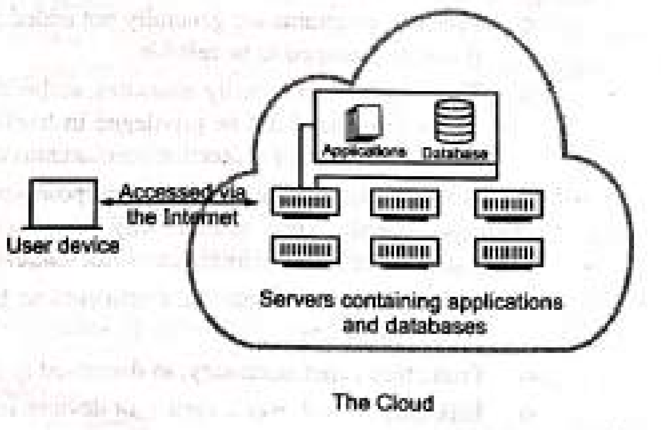
Applications
Database
Accessed via the Internet
User device
Servers containing applications and databases
The Cloud

Fig. 6.6.1 : The Cloud System

6.6.1 Cloud Computing

GQ. 6.6.1 Write short note on Cloud Operating System.

- In cloud computing, the software and the hardware required to run it are on somebody else's server. All you will be required to do is run install an application that serves as the interface to the service. The server does all the processing and saves your data, while your computer only needs enough resources to run and display the interface which could be done with just a web browser.
- As document is typed on Google Drive, a server that probably half way across the world is computing every keystroke and displaying them on my screen. My computer doesn't need to worry about running a heavy word processor and doesn't turn the heat up.
- Also owner need not have to worry about processor, software version, software Updation and all.To keep every up to date cloud vendor has work for that.
- In fact most popular e-mail services are cloud based as your mails are all stored off shore and the whole sending and receiving has been carried out by a server. The earliest reference to cloud computing was in the 50's where computers, used in academics and by corporations would be accessible via terminals which would only serve as a connection to the computer and would compute anything on their own.
- A sort of return on investment, these computers were accessed via these terminals, thus laying down the foundations for cloud based computing. With better internet speeds and stability, the idea of cloud OSes is in the process of taking off.

6.6.2 Working of Cloud Computing

- Cloud computing is possible because of a technology called virtualization. Virtualization allows for the creation of a simulated, digital-only "virtual" computer that behaves as if it were a physical computer with its own hardware. The technical term for such a computer is virtual machine.
- When properly implemented, virtual machines on the same host machine are sandboxed from one another, so they don't interact with each other at all, and the files and applications from one virtual machine aren't visible to the other virtual machines even though they're on the same physical machine.
- Virtual machines also make more efficient use of the hardware hosting them. By running many virtual machines at once, one server becomes many servers, and a data center becomes a whole host of data centers, able to serve many organizations.
- Thus, cloud providers can offer the use of their servers to far more customers at once than they would be able to otherwise; and they can do so at a low cost.
- Even if individual servers go down, cloud servers in general should be always online and always available. Cloud vendors generally back up their services on multiple machines and across multiple regions.
- Users access cloud services either through a browser or through an app, connecting to the cloud over the Internet – that is, through many interconnected networks – regardless of what device they're using.

6.6.3 Working of Operating System on Cloud

- One example is Chrome OS is that it's a cloud based operating system. While the operating systems on most computers works off the computer's hard drive, a Cloud OS or a Web OS on the other hand runs off a remote server. What actually runs on the computer is merely an interface which could even be something as basic as a web browser. All your data is saved on a remote server too.
- Computer at users end will require a smaller hard disk and less RAM and still run everything from word processors to spreadsheets, to games, to photo editors, to instant messaging and a whole lot more.
-

Fig. 6.6.2 : Plug in or else

Module 6

- Since all a cloud OS needs is just something to serve as an interface, quite a few work clean out of a browser, giving new meaning to working on the go. Just find any old computer anywhere in the world with an internet connection and log in. Everything you were working on will be right where you left it- even if you choose to access it using your smart phone.
- With cloud OSes, even old computers can become relevant again. The only catch when it comes to cloud OSes is the fact that you will constantly be connected to the internet. Which will be a problem in both the current scenario as well as the near future as with more users coming on board, the demand for faster, stable connections will rise.

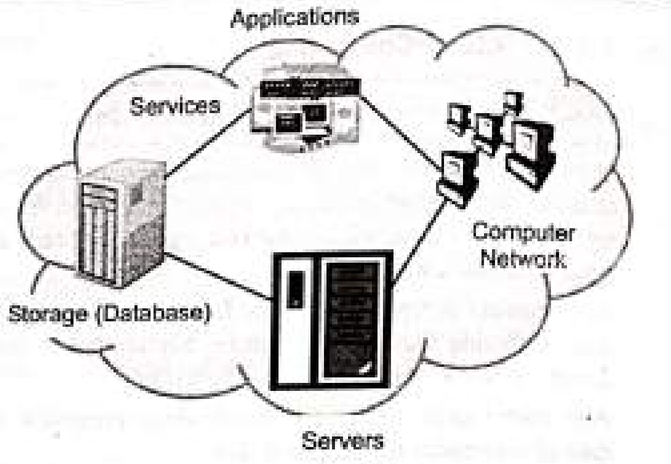

Fig. 6.6.3 : Cloud System

6.6.4 Features of Cloud Computing

GQ. 6.6.2 State features of Cloud Operating System.

Following are the Features of Cloud Computing:

1. **Resources Pooling**
   - The Cloud provider pulled the computing resources to provide services to multiple customers with the help of a multi-tenant model.
   - There are different physical and virtual resources assigned and reassigned which depends on the demand of the customer. The customer generally has no control or information over the location of the provided resources but is able to specify location at a higher level of abstraction
2. **On-Demand Self-Service**

   It is one of the important and valuable features of Cloud Computing as the user can continuously monitor the server uptime, capabilities, and allotted network storage. With this feature, the user can also monitor the computing capabilities.
3. **Easy Maintenance**
   - The servers are easily maintained and the downtime is very low and even in some cases, there is no downtime.
   - Cloud Computing comes up with an update every time by gradually making it better. The updates are more compatible with the devices and perform faster than older ones along with the bugs which are fixed.
4. **Large Network Access**

   The user can access the data of the cloud or upload the data to the cloud from anywhere just with the help of a device and an internet connection. These capabilities are available all over the network and accessed with the help of internet.
5. **Availability**

   The capabilities of the Cloud can be modified as per the use and can be extended a lot. It analyzes the storage usage and allows the user to buy extra Cloud storage if needed for a very small amount.
6. **Automatic System**

   Cloud computing automatically analyzes the data needed and supports a metering capability at some level of services. We can monitor, control, and report the usage. It will provide transparency for the host as well as the customer.

7. Economical

It is the one-time investment as the company (host) has to buy the storage and a small part of it can be provided to the many companies which save the host from monthly or yearly costs. Only the amount which is spent is on the basic maintenance and a few more expenses which are very less.

8. Security

- Cloud Security, is one of the best features of cloud computing.
- It creates a snapshot of the data stored so that the data may not get lost even if one of the servers gets damaged.
- The data is stored within the storage devices, which cannot be hacked and utilized by any other person. The storage service is quick and reliable.

9. Pay as you go

User has to pay only for the service or the space they have utilized. There is no hidden or extra charge which is to be paid. The service is economical and most of the time some space is allotted for free.

10. Measured Service

- Cloud Computing resources used to monitor and the company uses it for recording. This resource utilization is analyzed by supporting charge-per-use capabilities.
- This means that the resource usages which can be either virtual server instances that are running in the cloud are getting monitored measured and reported by the service provider. The model pay as you go is variable based on actual consumption of the manufacturing organization.

6.6.5 Advantages and Disadvantages of Cloud Computing

Advantages

Cloud computing is trending technology. Almost every company switched their services on the cloud to rise the company growth.

1. **Back-up and restore data :** Once the data is stored in the cloud, it is easier to get back-up and restore that data using the cloud.
2. **Improved collaboration :** Cloud applications improve collaboration by allowing groups of people to quickly and easily share information in the cloud via shared storage.
3. **Excellent accessibility :** Cloud allows us to quickly and easily access store information anywhere, anytime in the whole world, using an internet connection. An internet cloud infrastructure increases organization productivity and efficiency by ensuring that our data is always accessible.
4. **Low maintenance cost :** Cloud computing reduces both hardware and software maintenance costs for organizations.
5. **Mobility :** Cloud computing allows us to easily access all cloud data via mobile.
6. **Services in the pay-per-use model :** Cloud computing offers Application Programming Interfaces (APIs) to the users for access services on the cloud and pays the charges as per the usage of service.
7. **Unlimited storage capacity :** Cloud offers us a huge amount of storing capacity for storing our important data such as documents, images, audio, video, etc. in one place.
8. **Data security :** Data security is one of the biggest advantages of cloud computing. Cloud offers many advanced features related to security and ensures that data is securely stored and handled.

Disadvantages

1. **Internet Connectivity :** In cloud computing, every data (image, audio, video, etc.) is stored on the cloud, and we access these data through the cloud by using the internet connection. If internet connectivity is not good, then accessing the data is difficult. Also there is no any other way to access data from the cloud.

Module 6

2. **Vendor lock-in :** Vendor lock-in is the biggest disadvantage of cloud computing. Organizations may face problems when transferring their services from one vendor to another. As different vendors provide different platforms, that can cause difficulty moving from one cloud to another.
3. **Limited Control :** Cloud infrastructure is completely owned, managed, and monitored by the service provider, so the cloud users have less control over the function and execution of services within a cloud infrastructure.
4. **Security :** Although cloud service providers implement the best security standards to store important information. But, before adopting cloud technology, you should be aware that you will be sending all your organization's sensitive information to a third party, i.e., a cloud computing service provider. While sending the data on the cloud, there may be a chance that your organization's information is hacked by Hackers.

6.7 IOT OPERATING SYSTEM

GQ. 6.7.1 What is IoT? Explain IoT Operating System.

- The Internet of Things (IoT) describes the network of physical objects-"things"-that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the internet. These devices range from ordinary household objects to sophisticated industrial tools. With more than 7 billion connected IoT devices today, experts are expecting this number to grow to 25 billion by 2020 and 50 billion by 2025.
- Over the past few years, IoT has become one of the most important technologies of the 21st century. Everyday objects like-kitchen appliances, cars, thermostats, baby monitors-to the internet via embedded devices, seamless communication is possible between people, processes, and things.

Fig. 6.7.1 : Internet of Things(IoT)

- By means of low-cost computing, the cloud, big data, analytics, and mobile technologies, physical things can share and collect data with minimal human intervention. In this hyper connected world, digital systems can record, monitor, and adjust each interaction between connected things. The physical world meets the digital world—and they cooperate

6.7.1 Operating Systems for the IoT

- Several development platforms have been developed in a short span of time and are now competing with other developers and manufacturers.
- Also, the trends of miniaturization in devices, ubiquitous internet connectivity has become a priority, and the marketplace is witnessing a development in hardware-platform options targeted at IoT implementations.

- Large scale deployments, however, require rapid development tools, standardization, easy maintenance, and porting of applications across a wide variety of hardware platforms.
- Robust operating systems that are hardware diagnostic can guarantee these capabilities. Windows, Apple iOS, and Android stand as prominent examples of the dominance and impact an operating system can have in shaping not just the evolution of new technology paradigms but also in controlling the availability of applications to serve stakeholder needs. Many vendors are now vying for similar dominance.
- The key requirements that characterize operating systems for IoT and distinguish between systems for devices/end nodes and gateways.

6.7.2 Considerations for Operating Systems for IoT

Unlike personal computers and mobile devices, the architecture of IoT systems involves a large number of end-nodes (ex. sensors) connected to aggregating devices (gateways), which in turn are connected to remote cloud platforms. Fig. 6.7.2 represents a simplified view of an IoT system.

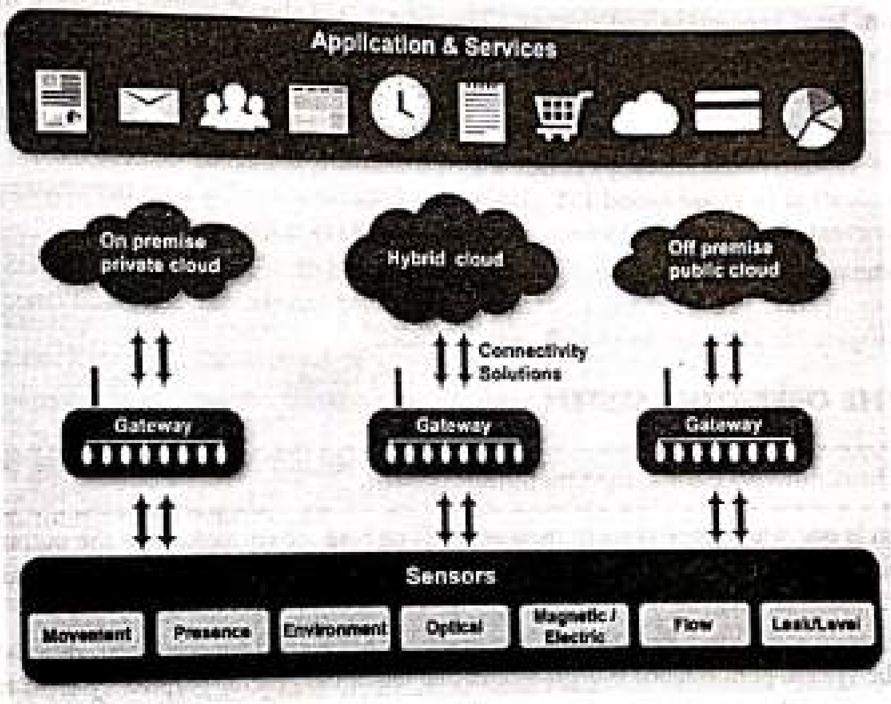

Fig. 6.7.2 : An IoT System – From Sensors to Applications

The functions of sensor nodes and their form factors and target applications vary considerably and create context for the suitability of any specific operating system for IoT implementations. The key characteristics and requirements that differentiate OS for IoT are below :

1. **Small memory footprint :** Sensor nodes are typically small and have limited memory available. This restricts the memory footprint of the OS Real-time capabilities - IoT applications like medical devices, automotive controls, and security systems are critical and require precision in timing.
2. **Energy efficient operation :** Sensors nodes are characterized by low power draws and are often battery powered. Furthermore, it is commonly prohibitively expensive to replace batteries etc. It is thus crucial that the OS be highly energy efficient.
3. **Hardware agnostic operation :** Owing to the diversity of hardware platforms available for various IoT applications, it is important that the OS support a variety of platforms to simplify interconnectivity and drive standardization and to lower costs of ownership.
4. **Network Connectivity and Protocol Support :** Crucial to IoT device operation is continuous connectivity to the network and to devices in immediate proximity. This requirement is achievable by providing support for a variety of connectivity protocols like Wi-Fi, Cellular, Bluetooth, etc. Operating system should simplify the connectivity process.

5. **Security :** It is imperative that the OS for IoT adhere to strict security expectations and meet stringent requirements imposed by deployments in sensitive and critical settings.
6. **Eco-system and Application Development :** The degree to which the OS can act as an eco-system for application development can make a big impact on the speed of development and time-to-market. OS that come with toolkits to ease the development and deployment of IoT applications are clearly beneficial.

Sensor end-nodes function as **sensors** or actuators, and at a basic level sense, they transmit and receive data. Gateways, on the other hand, are responsible for providing data services: routing, data shaping, and decision making. In addition, they act as firewalls to protect downstream devices (sensors, actuators, and other **embedded systems**).

Gateways form bridge between the real-world devices and virtual IT environments. Similarly, gateways can perform their own analytics and operations on real-time data and take independent action with minimal human intervention. To perform these important functions, additional requirements need to be met by the OS at the gateway level:

- **Protocol Support and data bridge :** In addition to supporting a variety of wired (USB, Serial, Ethernet, etc.) and wireless (Wi-Fi, BT, ZigBee, LoRA, etc.) protocols, the gateway OS should also support web protocols that typically have low transmission overhead like CoAP, MQTT, or UDP in addition to HTTP.
- **Data aggregation, local processing and storage :** The gateway OS should support decision making processes in real time and minimize the amount of data going to the cloud (ex: building management or video surveillance systems).
- **Security :** Gateway OS Security should afford protections at both the hardware and the network levels. Support for encryption, SSL/TLS certification management, user authentication, VPN connectivity, firewall, and application whitelisting is essential.
- **Reliable communication to cloud-based IoT platforms :** Capabilities to back up transmissions and to manage data and devices in the event of network disruption is a key attribute of OS at the gateway.
- **End device management :** Since a key function of gateways is end-device management, the OS should support remote upgrades, provisioning, and two-way communication to the cloud and the devices. Simultaneously, support for web-based tools to configure the gateways and end-devices is needed.

6.8 REAL-TIME OPERATING SYSTEM

GQ. 6.8.1 Write short note on Real Time Operating System.

- A real-time system is one whose logical correctness depends on both the correctness of the outputs and their timeliness. The timeliness factor makes these systems different from other systems. Real-time systems are the systems in which data need to be processed at a regular and timely rate.
- The tasks in the system must meet their defined deadlines for a specified time interval, otherwise the purpose of the system is lost or the system performance is degraded. In one sense, the general-purpose systems may also be considered as real-time.
- Today real-time systems are meant not only for applications such as nuclear reactors or aircrafts but also are playing a huge role in airline reservation systems, consumer electronics (music systems, air-conditioners, microwaves, etc.), card verifiers in security systems, control systems, industry automation, and so on.

6.8.1 Structure of a Real-time System

- A real-time system consists of two parts :
  - o controlling system (computer) and
  - o a controlled system (environment).
- The controlling system interacts with its environment, based on information available about the environment.
- The real-time computer controls a device or process through sensors that provide inputs at periodic intervals, and the computer must respond by sending signals to the actuators.

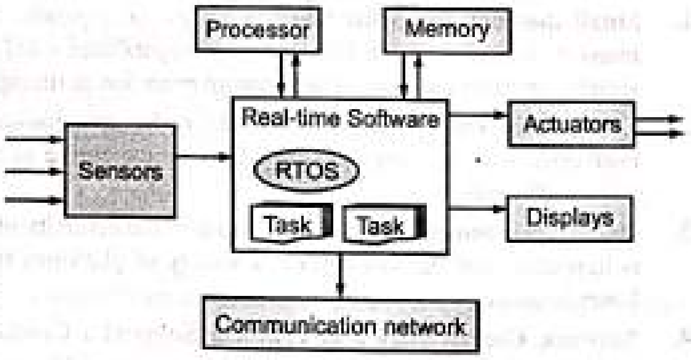

Fig. 6.8.1 : Structure of a real-time system

There may also be irregular events, which must receive a response as well. But in any case, periodic or aperiodic, there will be a time limit within which the response should be delivered. Thus, a real-time system consists of the following components as shown in Fig. 6.8.1.

) **Computing Hardware :** It includes the processor, memory, and communication networks.

) **Sensors and Actuators**

- The sensors are used to determine the state of the physical world relevant to the software. They sense the status of the system.
- Actuators are used to actuate the system according to the current status of the system. The output of the whole real-time system is fed to the actuators and displays.

6.8.2 Real-time Software

The software is the embedded part of the real-time system that interacts with sensors and actuators. It also interacts directly with the computing hardware. It consists of the tasks and real-time OS.

A real-time operating system (RTOS) is an operating system that works in real time, with deterministic constraints that require efficient time usage and power to process incoming data and relay the expected results without any unknown or unexpected delays. RTOS software is time dependent, meaning that it should process input and offer output within a short predetermined deterministic period.

However the key to an RTOS, and the most important demand of RTOS software is that a request and response for data is guaranteed to occur.

If a Windows OS has request and response calls that are fast 90% of the time, yet the remaining 10% of the time an input/output request takes too long, then the real-time application is not performing correctly.

Thus an RTOS is not meant to be only fast, it is more importantly meant to be dependable.

6.8.3 Why use an RTOS?

Following are the important reasons for using RTOS :

- It offers priority-based scheduling, which allows you to separate analytical processing from non-critical processing.
- The Real time OS provides API functions that allow cleaner and smaller application code.
- Abstracting timing dependencies and the task-based design results in fewer interdependencies between modules.
- RTOS offers modular task-based development, which allows modular task-based testing.
- The task-based API encourages modular development as a task, will typically have a clearly defined role. It allows designers/teams to work independently on their parts of the project.
- An RTOS is event-driven with no time wastage on processing time for the event which is not occur.

6.8.4 Applications of Real-Time Operating Systems

An RTOS can be flexible but is usually designed for set purposes. Most RTOS subsystems are assigned certain tasks and leave anything and everything else not designated to it for the Windows OS itself to handle. An RTOS offers mostly operational solutions, including applications such as :

1. **Control systems :** The RTOS is used to monitor and execute control system commands. Real-time systems are used to control actuators and sensors for functions like digital controllers. Controlled systems include aircraft, brakes, and engines. Controlled systems are monitored with the help of sensors and altered by actuators. The RTOS reads the data from sensors and then performs calculations and moves the actuators so that movement in a flight can be simulated.
2. **Image processing :** Computers, mobile gadgets, and cameras must achieve their intended duty in realtime, which means that visual input is needed in real-time with the utmost precision so that industrial automation, for instance can control what is happening on conveyors or an assembly line when an item is moving down its path and there is a defect, or the item has moved its location. Real-time image processing is essential for making real-time adjustments for moving

Module 6

objects.

3. **Voice over IP (VoIP) :** VoIP relies on Internet protocols to transmit voices in real time. As such, VoIP can be implemented on any IP network like intranets, local area networks, and the Internet. The voice is digitalized, compressed and converted to IP packets in real time before being transmitted over an IP network.

6.8.5 Considerations for Choosing an RTOS

Consider the following factors. :

(a) Performance	(b) Unique features
(c) Your IT team	(d) Middleware

(a) Performance

- Performance is a core factor that must be considered when choosing an RTOS. Real-time operating systems are different and perform differently.
- Key aspect for an RTOS is that its determinism guarantees that request and responses of data happen within a set period of time no matter what else is happening in the PC system.
- When determining the best RTOS, ask questions such as whether the system is showing any jitter within your tolerance range and thereby providing the determinism that you need.
- RTOS performance should be determined by a system's dependable in executing calls within a specified period, regardless of anything else happening on the system.

(b) Unique features

- Every real-time operating system has unique features that determine how it operates to execute commands.
- As such, you must evaluate the features required to run your system effectively and choose an RTOS with the relevant features. A good RTOS should be scalable and feature efficient memory protection systems.

(c) Your IT team

- Most people overlook their IT team when selecting the ideal RTOS.
- A good RTOS should favor your IT team by reducing their labor intensity so they have more time to focus on product differentiators and learn how to setup and integrate a real-time operating system. So, decide on an RTOS that your IT team is familiar with and can work with.

(d) Middleware

- Almost all real-time operating systems feature middleware components or third-party components that are integrated with the RTOS. You should evaluate the middleware to ensure that it has a seamless integration method. If your RTOS lacks middleware support, you may have to deal with time-consuming integration processes. Ensure that your middleware features components like TCP/IP and file systems.
- Good examples of real-time operating systems are the RTX (32-bit) and RTX64 (64-bit) solutions that allow you and your team to focus on adding value to your applications. This software is designed to serve as a hard real-time system that delivers output within a specified time frame to improve embedded systems' quality.

6.9 MOBILE OPERATING SYSTEM

GQ. 6.9.1 Explain Design issues of Mobile Operating System.

- Mobile devices are multifunctional devices serving both consumer and business use. In consumer electronics market, these devices have a wide range of products today.
- Furthermore, these devices have emerged as an extension to the conventional computer as most of the activities we do on a computer are possible now on these mobile devices.

- While we work conveniently on these mobile devices irrespective of place or time, the same can be synchronized with our work on the PC itself.
- For example, we use smartphones and tablets for e-mail, instant messaging, web browsing, chatting, working on official documents, and lots more. In general, a mobile device consists of the following hardware :
  - o Microprocessor
  - o Memory (volatile and non-volatile)
  - o Radiofrequency communication capability
  - o Input-output units such as keypad, LCD screen
  - o Power source, i.e., battery

6.9.1 Types of Mobile Devices

a. Personal Digital Assistant	b. Smartphones	c. Tablet PC

a. Personal Digital Assistant

- Personal digital assistants (PDAs) are electronic organizers that are also able to share information with our computer.
- A typical PDA manages well our daily-life activities such as Contact list and task planner. These are also known as hand-held or palm-top devices.
- Today, PDAs with the advancement of technology are equipped with more advanced features. For example, they are able to connect to the Internet, cellular phone, web browser, personal information organizer, and so on.

b. Smartphones

- The conventional cellular phone was not equipped with all modern features and it was in use as a phone only. With the advancement of technology, the concept of smartphones emerged with integration of the PDA features on the mobile phone.
- Later on, new features kept on adding in smartphones. Some features are ability to access digital media, digital cameras, pocket video games, GPS navigation, high resolution touch screens, Internet access, and so on.

c. Tablet PC

- Tablet PC is a mobile computer with all its components placed in a single unit. Tablets do not use the conventional input device such as mouse or keyboard.
- Primarily, these devices are operated with touch screens and are equipped with camera, microphone, buttons to control basic features like volume, ports to communicate, and so on. The size of tablets is in general larger than a PDA or smartphone. For a longer battery life, they use ARM processor.

6.9.2 Characteristics of Mobile Devices

Mobile devices are different as compared to other computing devices. These devices require very limited hardware configuration as these are portable. Following are characteristics :

Limited Processing Power

- The processing speed is in accordance with the power of battery. The processors used are of low processing power normally in the range of 50 MHz to 100 MHz.
- If high-power processors are used, then they need high-power batteries and high-power batteries cannot be adopted in mobile devices due to limitation of space.

Small Screen Size

The mobile devices have a very limited screen area due to limited space available on the device.

Limited Keypad

The input keys on the keypad available are also designed to utilize the space available. All the keys present on normal keyboard are not available on the mobile devices.

Limited Memory

- There are two types of memories in mobile devices as well: read only memory (ROM) and random access memory (RAM). ROM is for the OS and pre-installed programs and RAM is for user data.
- Due to processing power and space limitations, RAM is of low capacity. Mobile devices use flash memory as non-volatile memory instead of hard disk.

Limited Battery Capacity

- The mobile devices are based on secondary batteries connected to them. These batteries are of low power as they cannot be of high power due to space limitations.
- The battery power cannot be increased beyond a limit as it demands more space in the mobile device. Consequently, the processing speed is limited due to limited power of battery. Therefore, power management for the mobile devices is a big challenge. In case of any power loss, the data safety must also be provided.

6.9.3 MOBILE OS

- A mobile operating system is the foundation that allows you to run your favorite apps and programs on your mobile device. Like we said before, people aren't usually shopping based on the specific hardware.
- But, the operating system is that hardware that allows users to play their chosen games, access an array of settings, and utilize the array of services their mobile device offers. It's essentially the backbone for all of the capabilities of a mobile device.
- The mobile devices are different as compared to conventional desktop systems, they operate on operating system called as *mobile OSs*. Different Mobiles devices have different OS depends on the capabilities of the mobile devices and what they support. For example, the mobile OS for a PDA is different from that of a smartphone.
- Since in mobile devices some tasks are real-time driven, that is, must be handled within the deadline, a mobile OS utilizes
  - o a real-time operating for embedded systems including drivers for peripheral devices,
  - o communication software, and
  - o libraries shared between applications.
- The application software is placed on top of these mobile OSs. The mobile OS is designed keeping in view the physical and other limitations of the mobile devices.
- Moreover, the features of mobile devices and their types keep on changing. Therefore, an efficient OS is needed to support all these existing and upcoming new features.

6.9.4 Design Issues of Mobile Operating System

GQ. 6.9.2 Write short note on Mobile Operating System.

a. Power Management

- Since a mobile device gets operating power through a battery only, it is important that limited battery power be utilized efficiently. It is helpful in lower heat dissipation also, which consequently reduces the effect on environment.
- For example, smartphones are being used for phone calls, accessing Internet, making video recording, writing e-mails, and much more of what the user does on the desktop. Therefore, power management has been an important issue to give users more battery life.

System Power Management

- The OS must collect the information from devices so that the device that is not being used can be turned off.
- The OS may collect the information from application and user settings so that the system as a whole may be put into a low-power state. For system-power management, the following power states may be implemented:

Working

In this state, the device is in running state. If devices and processor are not in use, they are put into low-power state. However, these devices can be turned on depending on the type and its use of the device.

Sleeping

When the device is not in use or the user has pressed power button, the OS must put the device in the sleeping state. The nature of a sleeping state may differ on how an event can change the state to working and how much time it takes.

Device Power Management

To manage the power of all the devices, the OS may use standard methods specified in ACPI to send commands to the devices. The standard methods specify the following :

- The operations that can be used to manage power of devices
- The power states the devices may put into

The OS tracks the information about the state of all the devices on the bus. After having the current device requirements on that bus, it puts the bus in a particular power state accordingly. If all the devices do not require power presently, the OS sends a command to bus control chip to remove power from the bus.

Processor Power Management

The power can also be saved if the processor power is managed. When the OS enters in its idle state, it first determines how much time will be spent in idle loop with the help of ACPI power management timer. After determining the idle time, the OS puts the processor in low-power state, thereby saving the power.

b. Battery Management

- The OS collects the information about the state of the batteries in the mobile device and, through the user interface, notifies/warns the user about the low-power state of the device.
- In case of battery discharge, the OS must perform an emergency shutdown. However, in this shutdown, the damage to the mobile device integrity must be minimized.

c. Thermal Management

As the mobile devices operate with batteries, heat dissipation is also an issue. Therefore, thermal management is also mentioned in the ACPI where OS may reduce the power consumption of devices at the cost of performance to reduce the temperature of the mobile device.

d. Memory Management

- The mobile OS should consume very less space in the memory, that is, it should have a small footprint of about 100KB along with plug-in modules if any. However, to avoid RAM space, the kernel code, application, and libraries should be directly executed in ROM area instead of copying into the RAM area.
- This is known as *execute-in-place mechanism*. However, this increases the ROM space and the access in ROM is slower as compared to RAM. Thus, to reduce the memory space, the kernel and libraries should be customized to utilize the memory space. The microkernel structure of mobile OS may also help in reducing the memory.
- The OS should be able to manage memory automatically. For example, when memory is low at one instant of time, it must make space for the application or process by killing a process that is inactive for a long time.

e. Shortening Boot-up Time

The boot-up time in mobile devices must be managed to be short such that initializing time for the devices is short. Some of the following solutions will help in reducing the boot-up time :

- Small footprint as discussed in memory management helps in shortening the boot-up time.
- If initialization of device drivers can be deferred for some time, the boot-up time is reduced.
- The structure of OS should be microkernel.

6.9.5 Open Source Operating System - Android

A. What is Android?

- Android is an open source and Linux-based **Operating System** for mobile devices such as smartphones and tablet computers. Android was developed by the *Open Handset Alliance*, led by Google, and other companies.
- Android offers a unified approach to application development for mobile devices which means developers need only develop for Android, and their applications should be able to run on different devices powered by Android.
- The first beta version of the Android Software Development Kit (SDK) was released by Google in 2007 where as the first commercial version, Android 1.0, was released in September 2008. Android 11 is the eleventh major release and 18th version of Android, in September 2020.
- The source code for Android is available under free and open source software licenses. Google publishes most of the code under the Apache License version 2.0 and the rest, Linux kernel changes, under the GNU General Public License version 2.

C. Features of Android

Android is a powerful operating system competing with Apple 4GS and supports great features. Few of them are listed below :

Sr. No.	Feature and Description
1.	**Beautiful UI :** Android OS basic screen provides a beautiful and intuitive user interface.
2	**Connectivity :** GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE, NFC and WiMAX.
3	**Storage :** SQLite, a lightweight relational database, is used for data storage purposes.
4.	**Media support :** H.263, H.264, MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, AAC 5.1, MP3, MIDI, Ogg Vorbis, WAV, JPEG, PNG, GIF, and BMP.
5	**Messaging :** SMS and MMS
6.	**Web browser :** Based on the open-source WebKit layout engine, coupled with Chrome's V8 JavaScript engine supporting HTML5 and CSS3.
7.	**Multi-touch :** Android has native support for multi-touch which was initially made available in handsets such as the HTC Hero.
8.	**Multi-tasking :** User can jump from one task to another and same time various application can run simultaneously.
9.	**Resizable widgets :** Widgets are resizable, so users can expand them to show more content or shrink them to save space.
10.	**Multi-Language :** Supports single direction and bi-directional text.
11.	**GCM :** Google Cloud Messaging (GCM) is a service that lets developers send short message data to their users on Android devices, without needing a proprietary sync solution.
12.	**Wi-Fi Direct :** A technology that lets apps discover and pair directly, over a high-bandwidth peer-to-peer connection.
13.	**Android Beam :** A popular NFC-based technology that lets users instantly share, just by touching two NFC-enabled phones together.

D. Android Applications

- Android applications are usually developed in the Java language using the Android Software Development Kit.
- Once developed, Android applications can be packaged easily and sold out either through a store such as **Google Play, SlideME, Opera Mobile Store, Mobango, F-droid** and the **Amazon Appstore.**
- Android powers hundreds of millions of mobile devices in more than 190 countries around the world. It's the largest installed base of any mobile platform and growing fast. Every day more than 1 million new Android devices are activated worldwide.

6.9.6 Types of Mobile Operating Systems

When you purchase a mobile device the manufacturer will have chosen the operating system for that specific device. Often, you will want to learn about the mobile operating system before you purchase a device to ensure compatibility and support for the mobile applications you want to use. Some Popular Mobile OS are :

1. Android OS (Google Inc.)
2. Bada (Samsung Electronics)
3. BlackBerry OS (Research In Motion)
4. iPhone OS / iOS (Apple)
5. MeeGo OS (Nokia and Intel)
6. Palm OS (Garnet OS)
7. Symbian OS (Nokia)
8. webOS (Palm/HP)
9. Windows Mobile (Windows Phone)

6.10 MULTIMEDIA OPERATING SYSTEM

- Multimedia is the field that integrates text, graphics, images, videos, animation, audio, and any other media and represents, stores, transmits, and processes every type of information digitally.
- Multimedia applications largely consist of audio, video, and animation, in addition to traditional media. Movies, video clips, songs, webcasts of events, and so on, are all examples of multimedia applications. Thus, multimedia data are stored in the file system like any other data but captured by distinct devices.
- For example, digital video can be created using cameras or recorded from broadcasts. Compared to regular files, multimedia files have two unique characteristics.
  - o First, multimedia files are accessed at a specific rate. For example, to view a video, a series of images are displayed in rapid succession. In general, a video is displayed at 25 or 30 frames per second. The higher the rate at which these are displayed, the finer the video appears to the user. The same is the case with audio files. These are also known as *continuous media data.*
  - o The second unique feature of multimedia files is the need of real-time delivery when delivered on the network. In the sense of real time delivery, multimedia files are of two types:
    - o live streaming and
    - o on-demand streaming.
  - o In live streaming, multimedia contents are delivered as the event occurs, for example, the lecture delivery of a person through video-conferencing.
  - o On the other hand, in on-demand streaming, the delivery of contents is from stored systems, for example, on-demand movies from the cable television operator. In on-demand streaming, the event is not actually taking place when it is being delivered.

6.10.1 Multimedia

- The word multimedia depicts a large range of applications which are in popular use now-a-days. It includes audio and video files like MP3 audio files, movies, short video clips and flash / animation files of movie previews or news stories that are being downloaded via the Internet.
- Multimedia also includes live webcasting (i.e. broadcasting over the World Wide Web) of various events or sports. Multimedia applications are mixture of both.

Module 6

- A multimedia system should have the following desirable features :
  - It should run on a system having very high processing power.
  - The file system should support multimedia files.
  - The system should support efficient and high I/O.
  - The system should support high capacity storage.
  - The system should have network support.

Stream delivery

- The multimedia file is have to be accessed at an exact rate as it was produced and stored but accessing a regular file requires no particular timing. The video is usually represented by a sequence of images which is known as frames which are used to display in rapid succession. The faster the frames are displayed to users the smoother the video looks. When any data gets delivered from the local file system, you can refer to that delivery as 'local playback'. Multimedia files can also be stored on a remote server and delivered to a client across a network with a technique called streaming.
- There are two types of streaming techniques :
  - progressive download and
  - real-time streaming
- With a progressive download, any media file having both audio and video can be downloaded and stored on the users' local file system.
- Real-time streaming varies from progressive download wherein the media file gets streamed to the client but can be only played-and not stored/downloaded - by the client.

6.10.2 MULTIMEDIA OSs

- Since multimedia applications require soft-real-time response and consist of continuous media data, a general OS cannot fulfil specific rate and timing requirements.
- Therefore, multimedia OSs are designed to support multimedia applications. The requirements of specific data rate and timing requirements of continuous media in multimedia OSs are known as quality of service(QoS) guarantees. The following are some features that must be supported by these specialized Oss :
  - There may be multiple scheduling requirements. Besides the traditional scheduling, there may be requirement of both hard-real-time and soft-real-time scheduling. Thus, the scheduler must guarantee hard and soft deadlines through tasks with certain priorities. Each real time task has a timing constraint and resource requirements.
  - In multimedia applications, frames have a deadline and the system cannot afford to drop any frame or have any delay.
  - In multimedia applications, audio and video files that are different as compared to general text files having an internal structure. Moreover, these files are captured as well as played back by different devices.
  - A file in a multimedia application may consist of sub-files, for example, a digital movie file may consist of video file, audio files, text files, and so on. All the files should be properly synchronized so that the play back is proper.
  - There should be low latency and high responsiveness. There must be some acceptable ranges within which the delay or other parameter is confined. For example, end-to-end delay for audio streams should be below 150 ms for most multimedia applications.
  - Since multimedia streams tend to be periodic and consistent, schedulability consideration is much easier.To cope with the high bandwidth requirement, compression and lower resolution may be used.

6.10.3 Process Scheduling

- The multimedia scheduling algorithm must allocate resources used by different tasks so that all deadlines are met. For example, while running a movie, all frames must be displayed at the right times.
- Since multimedia systems have the requirement of traditional scheduling and soft as well as hard-real-time scheduling, they adopt multiple scheduling algorithms. One scheduling algorithm adopted in multimedia systems is the *proportional share scheduling*.

- In this scheduling, resource requirements are specified by the relative share of the resource. The resource is allocated on the basis of share of the competing requests. Another criterion is to have a combination of more than one scheduling algorithms that support any type of scheduling (real-time and non- real-time) requirement in a hierarchy.

6.10.4 File System

The access speed of storage devices has not been much improved as compared to the exponentially increased performance of processors and networks. Since multimedia files are too large in size, it is difficult for general file systems to handle multimedia data.

Moreover, the real-time playback and retrieval of multimedia data require intense bandwidth on the storage device. Thus, general file systems are not designed to meet the real-time performance requirement of the audio and video data retrieval. To cope with this issue, the following are required :

- The physical bandwidth of the disk must be exploited effectively.
- The disk fragmentation must be avoided by tailoring the file system layout, meta data structure, file organization, file allocation, and so on.
- The time to locate the data blocks on disk must be minimized.

Different file systems for multimedia systems

a. **Partitioned File Systems**
   - This file system consists of multiple sub-file systems.
   - Each sub-file system targets handling of a specific data type.
   - It also has one integration layer that provides transparent access to the data handled by different sub-file systems.

b. **Integrated File Systems**

   These systems multiplies all available resources in the system among all multimedia data.

6.10.5 File Allocation

Since multimedia files demand very high storage space, general file allocation may not be suitable. The following methods are being used for allocating files in multimedia Oss :

a. **Scattered Allocation :** In this method, blocks are allocated at arbitrary locations on the disk. The random placement is beneficial for mixed-media workloads in multimedia systems.

b. **Contiguous Allocation :** The data blocks in this method are allocated successively on the disk. It is better when compared to scattered allocation, but this causes external fragmentation. This allocation works well in case a video server is preloaded with movies that will not change afterwards.

c. **Locally Contiguous Allocation :** In this method, a file is divided into fragments. All blocks of a fragment are then stored contiguously. However, fragments can be scattered. This method causes less external fragmentation as compared to contiguous allocation.

d. **Constrained Allocation :** This method puts the constraint of distance measured in tracks for a finite sequence of block allocation, that is, the allocation is for finite blocks measured in finite number of tracks.

e. **Distributed Allocation**
   - This is applicable to systems where multiple storage devices exist. These multiple storage devices, say disks, can be used to store multimedia data. One simple method is to use one full file on only a single disk. For instance, one full movie is stored on a single disk *A*. Another movie is stored on disk *B*. One disk may also contain more than one full movie. But the drawback of this method is that if one disk fails, one or more full movies are lost. Moreover, the load on the disks is not distributed.
   - Data striping may be implemented to have a large logical sector by accessing many physical sectors from multiple disks. One method is to stripe each movie on multiple disks.

Module 6

- Here the assumption is that all frames are of same size, so that a fixed number of bytes from a movie are written to disk *A* and the same number of bytes are written to disk *B*, and so on.
- One drawback in this method is that all movies start on the first disk only, thereby increasing the load on this disk. To balance the load among disks, *staggered striping* may be used, wherein the starting disk is staggered every time.
- Another way to balance the load is to use *random striping*, wherein each file may be randomly allocated. The assumption of all frames with same size cannot be implemented every time.
- For example, in MPEG-2 movies, different frames are of varying sizes. So the idea is to stripe by frame. In *frame string*, the first frame of a movie is stored on disk *A*.
- The second frame goes to disk *B* and so on. This striping will not help in increasing the accessing speed but will help in spreading the load among disks in a better way. Another method may be *block striping*. In this method, fixed-size units are written on the disks as blocks. Each block may contain one or more frames.