FACULTY OF PURE, APPLIED AND HEALTH SCIENCES.
DEPARTMENT OF MATHEMATICAL SCIENCES.
COMPUTER SCIENCE PROGRAMME

COURSE CODE	**CMP 423**
COURSE TITLE	**NET-CENTRIC COMPUTING**
COURSE UNIT	**3**
COURSE DURATION	
LECTURER-IN-CHARGE	**DR. O. TAIWO**
EMAIL ADD.	**otaiwo@aul.ed.ng**
OFFICE NO.:	**E103**

COURSE CONTENT

Net-centric computing fundamentals, mobile and wireless computing, network security, client/server computing, parallel systems, parallel programing models, build web applications, message passing programming, dependence analysis, distributed systems, systems models, distributed objects, distributed transactions, flat and nested distributed transactions, concurrency, mobile, and cloud computing, technologies for wireless communications, wireless cellular systems.

References/Further Reading

INTRODUCTION

Net-centric is a medium used to manage data, applications, and infrastructure online. Net centric computing services allows one to centralize applications with a single interface. It provides fully managed services to user's specific requirements, which are invoked in real-time as needed rather than being provided on-demand. The concept of net-centric computing enables multiple distributed clients to access a single entity's application in real-time.

Net-centric computing is often referred to as network centered computing, and it is a major area of interest to software engineers. It is an enabling technology for modern distributed computing systems and applications. Net-centric computing is a distributed environment where applications and data are downloaded from servers and exchanged with peers across a network. Net-centric computing focuses on large-scale distributed computing systems and applications that communicate through open, wide-area networks like the internet. General examples of the large-scale network centric systems are the World Wide Web and computational grids.

Net-centric computing also refers to an emerging technology architecture and an evolutionary stage of client/server computing. It is a common architecture built on open standards that provides support in different ways for different people to collaborate and to reach different information resources.

FUNDAMENTALS OF NET-CENTRIC COMPUTING

This unit of the course discusses the essentials of net-centric computing, concepts of established networks in different locations and jobs running on each simultaneously (distributed computing), mobility issues and the security issues of resources running on different autonomous systems, client/server concepts and web building.

1.0 INTRODUCTION TO DISTRIBUTED COMPUTING

A distributed system is a system whose components are located on different networked computers that communicate and coordinate their actions by passing messages to one another from any system in order to appear as a single system to the end-user. The computers that are in a distributed system can be physically together and connected by a local network, or they can be geographically distant and connected by a wide area network. A distributed system can consist of any number of possible components such as mainframes, personal computers, workstations, minicomputers, and so on. Common use cases of a distributed systems are electronic banking systems, massive multiplayer online games, and sensor networks.

The two general ways that distributed system function are:

i. Each component of the system works to achieve a common goal and the end-user views results as one combined unit.
ii. Each component has its own end-user, and the distributed system facilitates sharing resources or communication services.

Distributed systems generally consist of four distinct architectural models. They are:

i. Client-server: Clients contact the server for data, then format it and display it to the end-user.
ii. Three-tier: Information about the client is stored in a middle tier rather than on the client, to simplify application deployment.
iii. n-tier: Generally used when the server needs to forward requeststo additional enterprise services on the network.
iv. Peer-to-peer: There are no additional nodes used to provide services or manage resources. Responsibilities are uniformly distributed among components in the system, known as peers, which can serve as either client or server.

1.1 Distributed Computing

Distributed computing is computing over distributed autonomous computers that communicate only over a network. It has been around for over three decades and is a much broader technology. Distributed computing systems are different form parallel computing systems or shared memory systems where multiple computers share a common memory pool that is used for communication between the processors. Distributed memory systems use multiple computers to solve a common problem with computation distributed among the connected computers (nodes) and using message-passing to communicate between the nodes. An example of distributed computing is the grid computing where the nodes may belong to different administrative domains. Another example is the network-based storage virtualization solution that uses distributed computing between data and metadata servers.

A distributed system is also a collection of independent components located on different machines that share messages with each other in order to achieve common goals. As such, a distributed system will appear as if it is one interface or computer to the end-user. The hope of distributed system is that together, the system can maximize resources and information while preventing failures, such that if one system fails, it will not affect the availability and performance of the service. Figure 1.1 presents the diagrammatic representation of distributed computing systems.

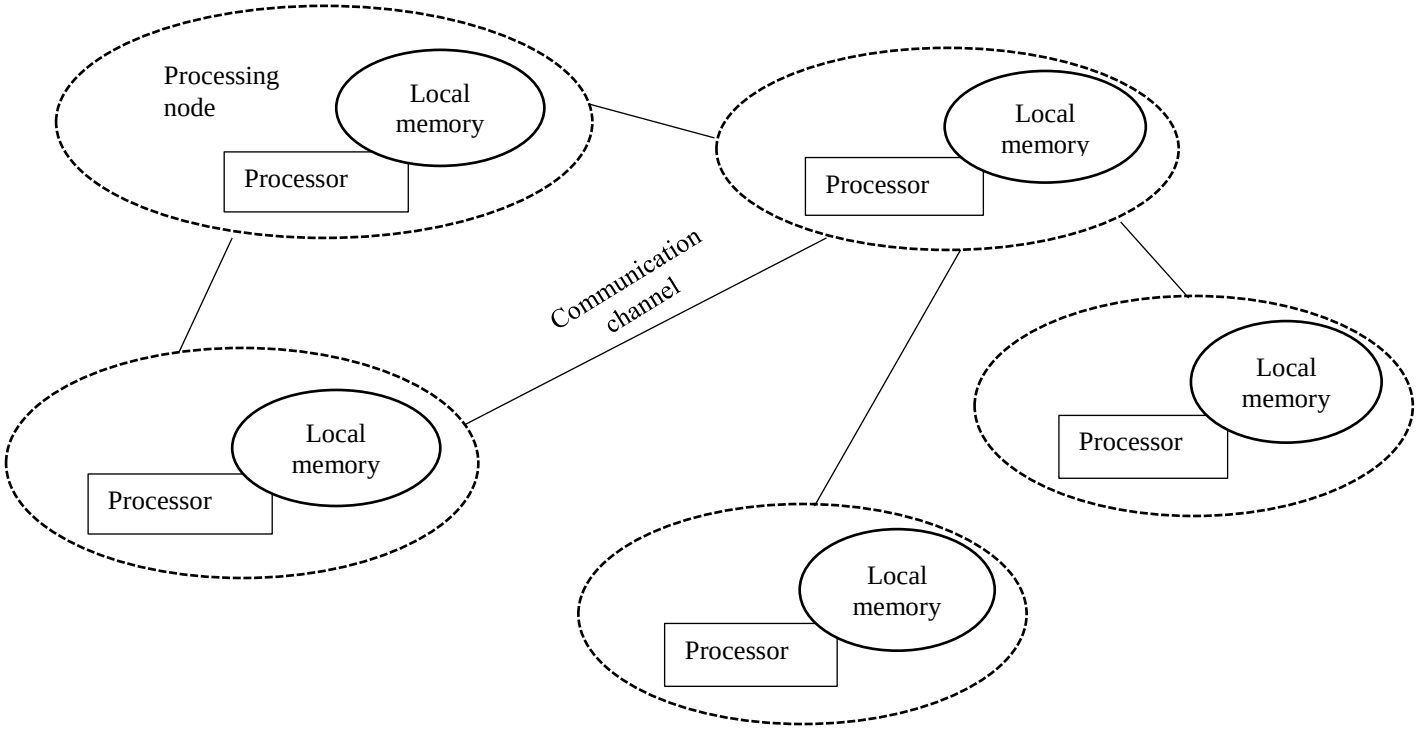

Figure 1.1: Diagrammatic Representation of a Distributed System.

Important functions of distributed computing are:

- **Resource sharing**: Hardware, software or data can be shared.
- **Openness**: Refers to how open the designed software is, to be developed and shared with each other.
- **Concurrency:** Multiple machines can process the same function at the same time.
- **Scalability:** Describes how the computing and processing capabilities multiply when extended to many machines.
- **Fault tolerance:** Ability to detect failures easily and quickly in parts of the system and recover.
- **Transparency:** How much access does one have to locate and communicate with other nodes in the system?

Components of Distributed Computing

There are several key components of a Distributed Computing System

- **Devices or Systems:** The devices or systems in a distributed system have their own processing capabilities and may also store and manage their own data.
- **Network:** The network connects the devices or systems in the distributed system, allowing them to communicate and exchange data.
- **Resource Management:** Distributed systems often have some type of resource management system in place to allocate and manage shared resources such as computing power, storage, and networking.

Characteristics of Distributed System

There are several characteristics that define a Distributed Computing System

- **Multiple Devices or Systems:** Processing and data storage is distributed across multiple devices or systems.
- **Peer-to-Peer Architecture:** Devices or systems in a distributed system can act as both clients and servers, as they can both request and provide services to other devices or systems in the network.
- **Shared Resources:** Resources such as computing power, storage, and networking are shared among the devices or systems in the network.
- **Horizontal Scaling:** Scaling a distributed computing system typically involves adding more devices or systems to the network to increase processing and storage capacity. This can be done through hardware upgrades or by adding additional devices or systems to the network..

Advantages and Disadvantages

Advantages of the Distributed Computing System are:

- **Scalability:** Distributed systems are generally more scalable than centralized systems, as they can easily add new devices or systems to the network to increase processing and storage capacity.
- **Reliability:** Distributed systems are often more reliable than centralized systems, as they can continue to operate even if one device or system fails.
- **Flexibility:** Distributed systems are generally more flexible than centralized systems, as they can be configured and reconfigured more easily to meet changing computing needs.

There are a few limitations to Distributed Computing System

- **Complexity:** Distributed systems can be more complex than centralized systems, as they involve multiple devices or systems that need to be coordinated and managed.
- **Security:** It can be more challenging to secure a distributed system, as security measures must be implemented on each device or system to ensure the security of the entire system.
- **Performance:** Distributed systems may not offer the same level of performance as centralized systems, as processing and data storage is distributed across multiple devices or systems.

Applications

Distributed Computing Systems have a number of applications, including:

- **Cloud Computing:** Cloud Computing systems are a type of distributed computing system that are used to deliver resources such as computing power, storage, and networking over the Internet.

- **Peer-to-Peer Networks:** Peer-to-Peer Networks are a type of distributed computing system that is used to share resources such as files and computing power among users.
- **Distributed Architectures:** Many modern computing systems, such as microservices architectures, use distributed architectures to distribute processing and data storage across multiple devices or systems.

1.1.1. Examples of Distributed System

Real-world examples of distributed systems are networks, telecommunication networks, distributed real-time systems, parallel processing, distributed artificial intelligence, and distributed database systems.

- **Networks:** Networks enhance computers to send messages to other systems with a local internet protocol (IP) address. The earliest example of a distributed system happened in the 1970s when ethernet was invented and LAN (local area networks) were created. Peer-to-peer networks evolved and e-mail and then the Internet as we know it continue to be the biggest, ever-growing example of distributed systems. As the internet changed from IPv4 to IPv6, distributed systems have evolved from "LAN" based to "Internet" based.
- **Telecommunication Networks:** Telephone and cellular networks are also examples of distributed networks. Telephone networks have been around for over a century, and it started as an early example of a peer-to-peer network. Cellular networks are distributed networks with base stations physically distributed in areas called cells. As telephone networks have evolved to VOIP (voice over IP), it continues to grow in complexity as a distributed network.
- **Distributed Real-time Systems:** Many industries use real-time systems that are distributed locally and globally. Airlines use flight control systems, Uber, Bolt, and In-drive use dispatch systems, manufacturing plants use automation control systems, logistics and e-commerce companies use real-time tracking systems.
- **Distributed Artificial Intelligence:** Distributed Artificial Intelligence is a way to use large scale computing power and parallel processing to learn and process very large data sets using multi-agents.
- **Parallel Processing:** There used to be a distinction between parallel computing and distributed systems. Parallel computing was focused on how to run software on multiple threads or processors that accessed the same data and memory. Distributed systems meant separate machines with their own processors and memory. With the rise of modern operating systems, processors, and cloud services these days, distributed computing also encompasses parallel processing.
- **Distributed Database Systems:** A distributed database is a database that is located over multiple servers and/or physical locations. The data can either be replicated or duplicated across systems. Most popular applications use a distributed database and need to be aware of the homogenous or heterogenous nature of the distributed database system. A homogenous distributed database means that each system has the same database management system and data model. They are easier to manage and scale performance by adding new nodes and locations. Heterogenous distributed databases allow for multiple data models, different

database management systems. Gateways are used to translate the data between nodes and usually happen as a result of merging applications and systems.

Distributed computing, however, can include heterogeneous computations where some nodes may perform a lot more computation, some perform very little computation, and a few others may perform specialized functionality (like processing visual graphics).

One of the main advantages of using distributed computing is that efficient scalable programs can be designed so that independent processes are scheduled on different nodes and they communicate only occasionally to exchange results, as opposed to working out of a shared memory with multiple simultaneous accesses to a common memory.

Cloud computing is also a specialized form of **distributed computing**, where distributed Software as a Service (SaaS) applications utilize thin clients (such as browsers) which offload computation to cloud-hosted servers (and services). Distributed computing, virtualization, service orientation, and Web 2.0 form the core technologies enabling the provisioning of cloud services from anywhere on the globe.

Web 2.0 technologies constitute the interface through which cloud computing services are delivered, managed, and provisioned. Besides the interaction with rich interfaces through the Web browser, Web services have become the primary access point to cloud computing systems from a programmatic standpoint.

Distributed computing is a foundational model for cloud computing because cloud systems are distributed systems. Besides administrative tasks mostly connected to the accessibility of resources in the cloud, the extreme dynamism of cloud systems where new nodes and services are provisioned on demand constitutes the major challenge for engineers and developers.

1.2 Service Orientation

Service orientation is the underlying paradigm that defines the architecture of a cloud computing system. ***Cloud Computing*** *is a computing platform that allows users to utilize shared pool of Cloud resources such as processors, services, storages in an on-demand mode*. The main goal of cloud computing is to make a better use of distributed resources, combine them to achieve higher throughput and be able to solve large scale computation problems. Cloud computing deals with virtualization, scalability, interoperability, quality of service and the delivery models of the cloud, namely private, public and hybrid [1]. Cloud computing provides users with resources and services using established standards and practices to allow universal access to cloud service in a standardized way [2] Services of cloud computing include computation, storage, retrieval, application, development, and web services. Investment on hardware, software, manpower and maintenance are no longer needed by users [3]. Cloud computing service is divided into three major layers. These are, Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS).

- ***Infrastructure as a Service (IaaS)*****:** IaaS is a service model in cloud computing that delivers computer infrastructure on outsourced basis to support enterprise operations. In IaaS layer, client do not need to procure the required servers, data center or network resources. Advantage of IaaS is that users pay only for the duration of time the service is being used [1]. IaaS offers

self-service such as monitoring, managing, and computing infrastructures (storage, networking etc) [4]

- ***Platform as a Service (PaaS):*** PaaS provides users with environment which has framework they can build upon to develop or customize applications. In PaaS computing platform applications typically needed by the user are already deployed on it. In essence, users do not need to through the stress of purchase and installation of hardware and software required for it. Through PaaS service, developers can have unrestricted access to all the systems and environments (such as developing, testing, deployment, and hosting of web applications) required for the life cycle of software [1].
- ***Software as a Service (SaaS):*** In Cloud computing with Software as a Service, web is used to deliver applications that are managed by a third-party vendor and whose interface is accessed on the user's end. Most SaaS applications can be run directly from a web browser without prior downloads installations, though some plug-ins might be required [5].

Characteristics of Cloud Computing

- Cloud computing allows users to have access to data, applications and other services provided by the layer with the help of a browser regardless of the device used and the user's location.
- With cloud computing, reliable services can be achieved through the use of various sites which is suitable for continuity of business and recovery in case of disaster [2].
- Efficient utilization of the infrastructure is allowed through sharing of resources and costs among a large collection of users.
- Since installation on each user's computer is not a primary requirement, maintenance is easier in cloud computing [1].
- Cloud computing is scalable as performance can be monitored.

Virtualization is another element that plays a fundamental role in cloud computing. This technology is a core feature of the infrastructure used by cloud providers. Virtualization concept is more than forty years old, but cloud computing introduces new challenges, especially in the management of virtual environments, whether they are abstractions of virtual hardware or a runtime environment.

Self-Assessment Questions.

1. Define distributed computing.
2. State functions of distributed computing.
3. Give and explain five (5) real-world examples of distributed computing.
4. Describe service orientation.
5. Describe cloud computing services.

References

[1] J. Yashpalsinh and M. Kirit, "Cloud Computing - Concepts, Architecture and Challenges," in *2012 Intenational Conference on Computing, Electronics and Electrical Technologies [ICCEET]*, 2012.

[2] Wikipedia, "Cloud Computing," [Online]. Available: https://en.wikipedia.org/wiki/Cloud_computing. [Accessed 20 May 2019].

[3] N. Zubair and A. Atif, "Clome: The Practical Implications of a Cloud-based Smart Home," 30 April 2014. [Online]. Available: https://www.researchgate.net/publication/262029806_Clome_The_Practical_Implications_of_a_Cloud-based_Smart_Home/download. [Accessed 20 May 2019].

[4] B. Gleb, "Choosing the Right Cloud Service: IaaS, PaaS, or SaaS," 8 August 2017. [Online]. Available: https://rubygarage.org/blog/iaas-vs-paas-vs-saas. [Accessed 6 June 2019].

[5] Atos, "IaaS, PaaS, SaaS(Explained and Compared)," 2019. [Online]. Available: https://apprenda.com/library/paas/iaas-paas-saas-explained-compared/. [Accessed 5 June 2019].

MOBILE AND WIRELESS COMPUTING

Introduction

Mobile computing refers to a technology that allows transmission of data, voice, and video via a computer or any other wireless enabled device. It is free from having connection with a fixed physical link. It facilitates the users to move from one physical location to another during communication. The essence of studying this provide basic and advanced concepts of mobile computing. In this unit, students will be study and understand mobile computing, mobile communication, mobile hardware and software, mobile classification, advantages and limitations of mobile computing, security issues and current trends.

2.1 Mobile and Wireless Computing

Mobile and wireless computing is a human–computer interaction concept in which a computer could be in motion during normal usage. Mobile and wireless computing involves mobile communication, mobile hardware, and mobile software, does involve the use of physical cable but devices are connected through electromagnetic waves. The farther the usage location to the network source the less the intensity of the cloud and speed of connection and vice versa.

Computing Technologies are the technologies that are used to manage, process, and communicate the data. *Wireless* simply means without any wire i.e., connecting with other devices without any physical connection. Wireless computing is transferring the data or information between computers or devices that are not physically connected to each other and having a "wireless network connection". For example, mobile devices, Wi-Fi, wireless printers, and scanners, etc. Mobile devices are not physically connected but then too we can transfer data.

2.2 Wireless/Mobile Computing Technologies.

1. Global System for Mobile Communications (GSM): GSM is a Current circuit-switched wireless data communication technology. It is established in Europe by ETSI (European Telecommunications Standards Institute) in the mid-1980s. GSM network has 4 different parks that who's functions are different: Mobile Station, BSS (Base Station Subsystem), NSS (Network Switching Subsystem), OSS (Operation and Support Subsystem).

As the name suggests, GSM is widely used for the mobile communication system. It operates in the frequency band 900-MHz, 1800-MHz, and 1900-MHz. GSM is developed using TDMA (Time Division Multiple Access) for better communication using mobile. It is the most widely used mobile communication system and is mostly required nowadays. It can achieve maximum data transmission speed or data transmission rate up to 9.6Kbps (Kilobits per second).

2. Code-Division Multiple Access (CDMA): CDMA is a type of wireless computing technology. It is developed during World War II. This technology is mostly used as it provides better network quality, more storage capacity for voice and data communications than TDMA, decreases system noise and interference using power control, provides more security by encoding the user transmission data into a unique code. CDMA does not provide any user with a specific frequency instead utilizes the entire frequency spectrum available for transmission. It operates in the frequency range of 800 MHz to 1.9 GHz. It uses Soft Handoff that reduces signal breaks.

3. Wireless in Local Loop (WLL): WLL is a widely used technology for wireless communication systems. It is also called a Fixed Wireless Loop. WLL is very easy to develop, and less time is required

to install, very cost-effective as wireless systems are less expensive because the cost of cable installation is not added.

WLL allows users to connect to the local telephone station using a wireless link and provides advanced features of customer service. It provides high-quality data transmission and a high data rate. Generally, two types of WLL techniques are available: Local Multipoint Distribution Service (LMDS) and Multichannel Multipoint Distribution Service (MMDS).

4. General Packet Radio Service (GPRS): GPRS is a type of Packet-based Wireless communication technology. It is established by ETSI (European Telecommunications Standards Institute). GPRS can achieve a data transfer rate of up to 114Kbps. It is very cost-effective, highly stable, can achieve a maximum data rate of up to 114Kbps (Kilobits per second). It supports Internet Protocol (IP), X.25 (standard protocol for packet-switched data communication), Point-to-Point protocol (PPP), and based on Gaussian minimum-shift keying (GMSK) which is a modulation technique. The Gateway GPRS Service Node (GGSN) and the Serving GPRS Service Node (SGSN) are the two core modules required to enable GPRS on GSM network or TDMA network.

5. Short Message Service (SMS): SMS is originally created for a phone/mobile that uses GSM Global System for Mobile communication). This service is used to send text messages even without the Internet connection between two or more mobile devices. This technique is very easy, user-friendly, comfortable and the most effective means of wireless communication.

In this service, less time is required for communication. It does not require any Internet connection for sending text messages. It allows the transmission of short messages i.e. up to 160 characters in length. SMS uses standardized communication protocols. SMS is received by Short Message Service Center (SMSC).

2.3 Mobile Communication

Mobile communication specifies a framework that is responsible for the working of mobile computing technology. Mobile communication refers to an infrastructure that ensures seamless and reliable communication among wireless devices. This framework ensures the consistency and reliability of communication between wireless devices. The mobile communication framework consists of communication devices such as protocols, services, bandwidth, and portals necessary to facilitate and support the stated services. These devices are responsible for delivering a smooth communication process.

2.3.1 Types of Mobile Communication

Mobile computing can be divided into four (4) major types:

1. Fixed and wired.

2. Fixed and wireless.

3. Mobile and wired

4. Mobile and wireless.

- **Fixed and Wired:** In fixed and wired configuration, the devices are fixed at a position, and they are connected through a physical link to communicate with other devices. An example is a desktop computer.

- **Fixed and Wireless:** In fixed and wireless configuration, the devices are fixed at a position, and they are connected through a wireless link to make communication with other devices. Examples are communication towers and Wi-Fi routers.
- **Mobile and Wired:** In mobile and wired configuration, some devices are wired, and some are mobile. They altogether make communication with other devices. An example is a laptop.
- **Mobile and Wireless:** In mobile and wireless configuration, the devices can communicate with each other irrespective of their position. They can also connect to any network without the use of any wired device. An example is a Wi-Fi dongle.

2.4 Mobile Hardware:

Mobile hardware consists of mobile devices or device components that can be used to receive or access the service of mobility. Examples of mobile hardware are smartphones, laptops, portable PCs, tablets, personal digital assistants. These are devices that are inbuilt with a receptor medium that can send and receive signals. These devices can send and receive signals at the same time. They do not have to wait until one device has finished communicating for the other device to initiate communications.

2.5 Mobile Software

Mobile software is a program that runs on mobile hardware. This is designed to deal capably with the characteristics and requirements of mobile applications. This is the operating system for the appliance of mobile devices. In other words, you can say it the heart of the mobile systems. This is an essential component that operates the mobile device. Software provides portability to mobile devices which ensures wireless communication. Since portability is the main factor, this type of computing ensures that users are not tied or pinned to a single physical location but are able to operate from anywhere. It incorporates all aspects of wireless communications.

2.6 Mobile Computing Classification

Mobile computing is not only limited to mobile phones, but there are various gadgets available in the market that are built on a platform to support mobile computing. Mobile computing is applied in several fields such as web or internet access, Global Position System (GPS), emergency services, entertainment services, health services and educational services. Mobile computing is classified into the following categories:

1. **Personal Digital Assistant (PDA):** The main purpose of this device is to act as an electronic organizer or day planner that is portable, easy to use and capable of sharing information with your computer systems. PDA systems are capable of sharing information with a computer system through a process or service known as synchronization. Both devices will access each other to check for changes or updates in the individual devices. With PDA devices, a user can browse the internet, listen to audio clips, watch video clips, edit and modify office documents, and many more services.
2. **Smartphones:** It combines the features of a PDA with that of a a mobile phone or camera phone. It has a superior edge over other kinds of mobile phones because smartphones have the capability to run multiple programs concurrently. These phones include high-resolution touch screens, web browsers that can access and properly display standard web pages rather than just mobile-optimized sites, high-speed data access via Wi-Fi and high-speed cellular broadband. The most common mobile Operating Systems (OS) used by modern

smartphones include Google Android, Apple's iOS, Nokia's Symbian, and RIM's Blackberry OS.

3. **Tablet PC and iPads:** This mobile device is larger than a mobile phone or a PDA and integrates into a touch screen and is operated using touch sensitive motions on the screen They are often controlled by a pen or by the touch of a finger. They are usually in slate form and are light in weight. Examples would include ipads, Galaxy Tabs, Blackberry Playbooks etc. They offer the same functionality as portable computers. Users can edit and modify document files, access high speed internet, stream video and audio data, receive and send e-mails, attend/give lectures, and presentations among its very many other functions. They have excellent screen resolution and clarity.
4. **Wearable computers:** Wearable computers are a type of computer that can be worn by the bearer under, with or on top of clothing. They are also known as body-borne computers or wearables, which are small electronic devices. Some examples of wearable computers are smartwatches, digital fitness bands etc.

2.7 Advantages of Mobile Computing

Advantages of mobile computing are location flexibility, time efficiency, enhanced productivity, ease of research, entertainment, and commercial and business processes.

- **Location Flexibility:** Mobile computing has enabled users to work from any location once there is an established connection. Mobility ensures that users are able to carry out numerous tasks at the same time and perform their stated jobs,
- **Time Efficiency:** Mobile computing saves time by allowing people to work at their convenient location and cut back on the time wasted on travelling from a location to another. Important documents and files can be accessed over a secure channel or portal without being confined to a specific location. Mobile computing has enhanced telecommuting in many companies and has reduced unnecessary time and expenses spent in traveling and traffic.
- **Enhanced Productivity:** With mobile computing, users can work efficiently and effectively from any location they find comfortable. This in turn enhances productivity level.
- **Ease of Research:** Research has been made easier since users with mobile computing because researchers can save data collected on field to the system and have access to data online. Also, field officers and researchers can collect and feed data from any location without having to report to physical offices before giving reports.
- **Entertainment:** Video and audio recordings can now be streamed on-the- go using mobile computing. It is easy to access a wide variety of movies, entertainment, educational, and informative material online with mobile computing. With the improvement and availability of high-speed data connections at considerable cost, users can get all the entertainment they want as they browse the internet for streamed data.
- **Streamlined Commercial and Business Process**: Business processes are now easily available through secured connections through mobile computing. Business meetings, calls, conferences, and seminars can be held remotely with the aid of mobile computing. This facilitates easier and faster decision making among stakeholders and reduce the stress of traveling for meetings. Authentication and authorization measures have also enhanced security of meetings and how users access services.

2.8 Security Issues in Mobile Computing

Mobile computing is prone to security threats as other technologies. Due to the nomadic nature of mobile computing, the usage is not easy to monitor. Also, users might have different intentions on how to utilize the privileges associated with mobile computing. Other security issues are improper and unethical practices such as hacking, industrial espionage, pirating, online fraud, and malicious destruction of services and resources credential verification, identity theft, and unauthorized access to data information by hackers.

These security issues, challenges, and threats can be minimized the following measures:

- Hiring of qualified personnel.
- Installation of security hardware and software.
- Educating users on proposer mobile computing ethics as applicable to each organization.
- Auditing and development of standard and effective policies to govern mobile computing.
- Enforcing proper access rights and permissions.

2.9 Limitations of Mobile Computing

1. **Range and Bandwidth**: Mobile Internet access is generally slower than direct cable connections, using technologies such as GPRS and EDGE, and more recently HSDPA and HSUPA 3Gand 4G networks and also upcoming 5G network. These networks are usually available within range of commercial cell phone towers. High speed network wireless LANs are inexpensive but have very limited range.

2. **Security standards**: When working mobile, one is dependent on public networks, requiring careful use of VPN. Security is a major concern while concerning the mobile computing standards on the fleet. One can easily attack the VPN through a huge number of networks interconnected through the line.

3. **Power consumption**: When a power outlet or portable generator is not available, mobile computers must rely entirely on battery power. Combined with the compact size of many mobile devices, this often means unusually expensive batteries must be used to obtain the necessary battery life.

4. **Transmission interferences**: Weather, terrain, and the range from the nearest signal point can all interfere with signal reception. Reception in tunnels, some buildings, and rural areas is often poor.

5. **Potential health hazards**: With the advent of mobile computing, most people join online meetings, seminars, discussions and so on, while attending to other things. People who use mobile devices while driving are often distracted from driving and are thus assumed more likely to be involved in traffic accidents. (While this may seem obvious, there is considerable discussion about whether banning mobile device use while driving reduces accidents or not.) Cell phones may interfere with sensitive medical devices. Questions concerning mobile phone radiation and health have been raised.

6. **Human interface with device**: Screens and keyboards tend to be small, which may make them hard to use. Alternate input methods such as speech or handwriting recognition require training.

2.10 Research Trends

This section gives a discussion on the current mobile computing technologies, devices, terms, and research areas. It's important you're aware of the new technology that's available and surrounding you.

2.10.1 Current and Future Mobile Computing Research Trends

- **Artificial intelligence (AI):** There are advancements in mobile computing AI. Common and familiar AI-driven devices and technologies in mobile computing are Alexa, Siri, Cortana, Google Assistant. All of these are examples of AI that may even be installed on your mobile devices right now. In addition to these popular forms of AI, mobile apps are now using software such as voice recognition to encourage hands-free use and ultimately optimize the customer experience. AI software is used to help developers and marketers learn more about the user. Businesses are trying to get more revenue by using this information to create relevant advertisements that target specific audiences.
- **Location-based technology:** Your smartphones and tablets are tracking your location. Mobile applications are also tracking your location, with your permission. Each time you download a new app, it requests your permission to use your location. Each time you download a new app, you'll get a notification. You may not even be able to use some apps to their full potential without giving them access to your location. For example, think about a ride-sharing app such as Uber. They need your exact location to connect you with a driver. There is an increase in apps requesting your location even if you don't think it's required to use the primary function of the app. That's because most of the apps on your smartphone share your data with third parties. They do this to enhance their marketing campaigns. If a business knows where a user is, it can send them targeted ads based on the location. An example of this is when an app uses geofencing technology. How it works - Let's say you own a restaurant and have a mobile app. If an app user walks within a few blocks of your location, they'll receive a notification about your lunch special.
- **Motion and Location Sensing:** Most mobile phones have location sensor capabilities which use multiple positioning methods to provide different granularities of location data. Knowing an individual's location to within a few meters is useful for providing highly relevant contextual information and services. Motion sensing apps are used in security, anti-theft, power-saving, and games. Location sensing is useful in Geotagging, Games, Vehicle navigation, and fitness apps. Apps exploiting precise indoor location currently use technologies such as Wi-Fi, imaging, ultrasonic beacons, and geomagnetic. In the longer run technologies such as smart lighting will also become important. Precise indoor location sensing, combined with mobile applications, will enable a new generation of extremely personalized services and information.
- **Augmented reality:** Augmented reality takes something that's real and modifies it. One of the best examples of this is the face filter options on Snapchat. Recently, Instagram implemented this feature as well. This will help businesses create brand awareness, app downloads, engagement, and revenue.
- **Syncing wearable technology with mobile devices:** Wearable technology has become increasingly popular. The smartphone will become the hub of a personal-area network consisting of wearable gadgets such as on-body healthcare sensors, smart jewellery, smart watches, display devices (like Google Glass) and a variety of sensors embedded in clothes and

shoes. These gadgets will communicate with mobile applications to deliver information in new ways. And will enable a wide range of products and services in areas such as sport, fitness, fashion, hobbies, and healthcare. Thus, wearable devices connected with smartphones will influence the next generation of mobile application development strategies. Take Fitbit as an example. All the movements of a person wearing it can be tracked through an app. Users can check their heart rates and how many miles they walked in a day, among other things. By syncing with mobile devices, these apps can be used socially as well. People can compare their progress with their friends and make it a competition. As a result, it encourages the usage of the technology and increases engagement.

- **Mobile devices syncing with homes:** Mobile apps are being developed to help improve consumers' experiences within their own homes. One can find businesses that sync your home air conditioning and heating with an app. That way, one can control temperatures whether at home or not. Instead of going to a central thermostat in the house, you can reach into your pocket and set everything on your phone. Home security has been integrated with mobile technology as well. There are apps that have a video camera synced with your doorbell so you can see who is at your front door when the bell rings. Home security cameras on the inside and outside of your home can all be controlled and monitored from mobile devices. There are even smart refrigerators that connect with mobile devices. This technology gives you the ability to see inside your refrigerator while you're at the grocery store so you can see what you need to buy.

2.10.2 Current and Future Mobile Computing Technology Trends

These are the list of the current mobile technologies starting from 5G technologies which is the hottest mobile technology available in the market.

- **5G:** In telecommunications, **5G** is the fifth-generation technology standard for broadband cellular networks. Telecoms company began to deploy it in 2019 and is the planned successor to the 4G networks providing connectivity to most current cellphones. 5G networks are predicted to have more than 1.7 billion subscribers worldwide by 2025. 5G networks cellular networks' service area is divided into small geographical areas called *cells*. All 5G wireless devices in a cell are connected to the Internet and telephone network by radio waves through a local antenna in the cell. It has an advantage of having greater bandwidth and download speeds up to 10 gigabits per second (Gbit/s). 5G is not only faster than existing networks, 5G can connect more different devices. Due to the increased bandwidth, the networks will increasingly be used as general internet service providers (ISPs) for laptops and desktop computers, competing with existing ISPs such as cable internet, and also will make possible new applications in Internet-of- Things (IoT) and machine-to-machine areas.

- **4G:** is the short name for fourth-generation wireless. It is the stage of broadband cellular network technology that precedes 5G and supersedes 5G. A 4G system must provide capabilities defined by the International Telecommunication Union (ITU) in IMT Advanced. At the most basic level, a 4G connection works via an antenna that transmits over radio frequencies, enabling mobile devices to connect to mobile networks.

  The transmission and receiving capabilities of 4G are powered by MIMO (Multiple Input Multiple Output) and Orthogonal Frequency Division Multiplexing (OFDM) technologies.

Both MIMO and OFDM enable more capacity and bandwidth in comparison to 3G. OFDM provides more speed than the primary technologies that powered 3G, which include TDMA (Time Division Multiple Access) and CDMA (Code Division Multiple Access) technology. With MIMO, 4G reduces network congestion in comparison to 3G, because more users can be supported.

4G is also an all-IP (internet protocol)-based standard for both voice and data, different from 3G, which only uses IP for data, while enabling voice with a circuit-switched network. As an all-IP network, 4G is more efficient for mobile network providers to operate and optimize than managing different network technologies for voice and data.

Recent applications include amended mobile web access, IP telephony, gaming services, high-definition mobile TV, video conferencing, and 3D television. WIMAX

- **3G or Third Generation**: 3G mobile telecommunications is a generation of standards for mobile phones and mobile telecommunication services fulfilling the International Mobile Telecommunications-2000 (IMT-2000) specifications by the International Telecommunication Union. Application services include wide-area wireless voice telephone, mobile Internet access, video calls and mobile TV, all in a mobile environment.
- **Global Positioning System (GPS):** The Global Positioning System (GPS) is a space-based satellite navigation system that provides location and time information in all weather, anywhere on or near the Earth, where there is an unobstructed line of sight to four or more GPS satellites. The GPS program empowers the military, civil and commercial users around the world, and it is the backbone for modernizing the global air traffic system, weather, and location services.
- **Long Term Evolution (LTE):** LTE is a standard for wireless communication of high-speed data for mobile phones and data terminals. It is based on the GSM/EDGE and UMTS/HSPA network technologies, increasing the capacity and speed using new modulation techniques. It is related with the implementation of fourth Generation (4G) technology.
- **WiMAX:** WiMAX (Worldwide Interoperability for Microwave Access) is a wireless communications standard designed to provide 30 to 40 megabit-per-second data rates, with the latest update providing up to 1Gbit/s for fixed stations. WiMAX is a part of the fourth generation or 4G wireless communication technology. WiMAX surpasses the 30-metre wireless range of a conventional Wi-Fi Local Area Network (LAN), offering a metropolitan area network with a signal radius of about 50 km. WiMAX offers data transfer rates that can be superior to conventional cable-modem and DSL connections, however, the bandwidth must be shared among multiple users and thus yields lower speed in practice.
- **Near Field Communication:** Near Field Communication (NFC) is a set of standards for smartphones and similar devices to establish radio communication with each other by touching them together or bringing them into close proximity, usually no more than a few centimeters Present and anticipated applications include contactless transactions, data exchange, and simplified setup of more complex communications such as Wi-Fi. Communication is also possible between an NFC device and an unpowered NFC chip, called a "tag".

In conclusion, computing has rapidly grown from being confined to a location to access from any location. Also, with mobile computing, people can work from the comfort of any location the wish as

long as the connection and the security concerns are properly factored. In the same light, the presence of high-speed has also promoted the use of mobile computing. Being an ever growing and emerging technology, mobile computing will continue to be a core service in computing.

Self-Assessment Questions

1. Discuss extensively why the mobile software important in mobile and cloud computing.
2. Define mobile computing.
3. Discuss the current and future research and technology trends for mobile computing.

NETWORK SECURITY

3.1 Network Security

The transmission of data from one point to another on the network is of great concern and importance. Therefore, there is the need to deploy measure(s) that can secure the transmission of data from unauthorized media. This gave rise to the need for network security. This unit of the course focuses on explanation of network, discussion on network security, and identification of network security tools and techniques.

Network security is a term that describes the security tools, tactics and security policies designed to monitor, prevent and respond to unauthorized network intrusion, while also protecting digital assets, including network traffic. Network security includes hardware and software technologies (including resources such as savvy security analysts, hunters, and incident responders) and is designed to respond to the full range of potential threats targeting your network. Network security is the defense used to protect against increasing cybercrime. Focus of network security is discussed in the next subsection.

3.1.1 Focus of Network Security.

There are three key focuses that serves a foundation of any network security strategy. These are protection, detection, and response.

- ***Protection*** entails any tools or policies designed to prevent network security intrusion.
- ***Detection*** refers to the resources that allow you to analyze network traffic and quickly identify problems before they can do harm.
- **R*esponse*** is the ability to react to identified network security threats and resolve them as quickly as possible.

3.1.2 Benefits Network Security

Network security tools and devices enable organizations to protect its sensitive information, overall performance, reputation and its continuity in business. Secure and reliable networks protect not just organizational interests and operations, but also any client or customer who exchanges information with the organization, in addition to the general public. The benefits of network security are:

i. **Builds trust:** Security for large systems translates to security for everyone. Network security boosts client and consumer confidence, and it protects business from the reputational and legal fallout of a security breach.

ii. **Mitigates risk:** The right network security solution will help business owners stay compliant with business and government regulations, and it will minimize the business and financial impact of a breach if it does occur.

iii. **Protects proprietary information:** Clients and customers rely companies for security sensitive information. Business relies on that same protection, too. Network security ensures the protection of information and data shared across the network.

iv. **Enables a more modern workplace:** From allowing employees to work securely from any location using VPN to encouraging collaboration with secure network access, network security provides options to enable the future of work. Effective network security also provide many levels of security to scale with your growing business.

3.2 Network Security Tools and Techniques

Network connections encounter varying degrees of threats and therefore should be always therefore should be prepared to defend, identify and respond to a full range of attacks. However, the reality is that the biggest danger to most companies are not fly- by-night threat actors, but the attackers that are well-funded and are targeting specific organizations for specific reasons. Hence, network security strategy needs to be able to address the various methods these actors might employ. Listed below are fourteen (14) different network security tools and techniques designed to help do just that:

1. **Access control:** If threat actors or intruders cannot access a network, the amount of damage they will be able to do will be extremely limited. But in addition to preventing unauthorized access, *authorized* users can be potential threats. Access control allows network security to be increased by limiting user access and resources to only the parts of the network that directly apply to individual users' responsibilities.
2. **Anti-malware software:** Malware, in the form of viruses, trojans, worms, keyloggers, spyware, etc. are designed to spread through computer systems and infect networks. Anti-malware tools are a kind of network security software designed to identify dangerous programs and prevent them from spreading. Anti-malware and antivirus software may also be able to help resolve malware infections, minimizing the damage to the network.
3. **Anomaly detection:** It can be difficult to identify anomalies in your network without a baseline understanding of how that network should be operating. Network anomaly detection engines (ADE) allow you to analyze your network, so that when breaches occur, you will be alerted to them quickly enough to be able to respond.
4. **Application security:** For many attackers, applications are a defensive vulnerability that can be exploited. Application security helps establish security parameters for any applications that may be relevant to your network security.
5. **Data Loss Prevention (DLP):** Often, the weakest link in network security is the human element. DLP technologies and policies help protect staff and other users from misusing and possibly compromising sensitive data or allowing said data out of the network.
6. **Email security:** As with DLP, email security is focused on shoring up human-related security weaknesses. Via phishing strategies (which are often very complex and convincing), attackers persuade email recipients to share sensitive information via desktop or mobile device, or inadvertently download malware into the targeted network. Email security helps identify dangerous emails and can also be used to block attacks and prevent the sharing of vital data.
7. **Endpoint security:** The business world is becoming increasingly, "bring your own device" (BYOD), to the point where the distinction between personal and business computer devices is almost non-existent. Unfortunately, sometimes the personal devices become targets when users rely on them to access business networks. Endpoint security adds a layer of defense between remote devices and business networks.
8. **Firewalls:** Firewalls function much like gates that can be used to secure the borders between your network and the internet. Firewalls are used to manage network traffic, allowing authorized traffic through while blocking access to non-authorized traffic.
9. **Intrusion prevention systems:** Intrusion prevention systems (also called intrusion detection) constantly scan and analyze network traffic/packets, so that different types of attacks can be identified and responded to quickly. These systems often keep a database of known attack methods, so as to be able to recognize threats immediately.

10. **Network segmentation:** There are many kinds of network traffic, each associated with different security risks. Network segmentation allows you to grant the right access to the right traffic, while restricting traffic from suspicious sources.
11. **Security information and event management (SIEM):** Sometimes simply pulling together the right information from so many different tools and resources can be prohibitively difficult — particularly when time is an issue. SIEM tools and software give responders the data they need to act quickly.
12. **Virtual private network (VPN):** VPN tools are used to authenticate communication between secure networks and an endpoint device. Remote-access VPNs generally use IPsec or Secure Sockets Layer (SSL) for authentication, creating an encrypted line to block other parties from eavesdropping.
13. **Web security:** Including tools, hardware, policies and more, web security is a blanket term to describe the network security measures businesses take to ensure safe web use when connected to an internal network. This helps prevent web-based threats from using browsers as access points to get into the network.
14. **Wireless security:** Generally speaking, wireless networks are less secure than traditional networks. Thus, strict wireless security measures are necessary to ensure that threat actors are not gaining access.

In conclusion, network security tools and devices exist to help organizations protect not only its sensitive information, but also its overall performance, reputation and even its ability to stay in business.

Three key focuses that should serve as a foundation of any network security strategy are protection, detection and response Protection entails any tools or policies designed to prevent network security intrusion. *Detection* refers to the resources that allow you to analyze network traffic and quickly identify problems before they can do harm. R*esponse* is the ability to react to identified network security threats and resolve them as quickly as possible. The network security tools and techniques are access control, anti-malware, anomaly detection, application security, data loss prevention (DLP), endpoint security, firewalls and email security. Others are intrusion prevention systems, network segmentation, Security information and event management (SIEM), virtual private network (VPN), web security and wireless security.

Self-Assessment Questions

1. Discuss the tools that can be used to secure the network.
2. Explain the benefits of network security.

CLIENT-SERVER COMPUTING

Introduction

There are two configurations of networks: Client-Server and Peer-to-Peer networks. In client server, the client requests resources while the server serves same. In Peer-to-peer configuration, each node is free to communicate with others or not. The nodes under this configuration are not over-seen by any node or the other, they relate in a workgroup. This section explains the concept of client-server, description of client, and state the differences the client-server and the peer-to-peer configuration of networks.

4.1 Client-Server Computing

In client-server computing, the client requests a resource and the server provides that resource. A server may serve multiple clients at the same time while a client is in contact with only one server. Both the client and server usually communicate via a computer network, but sometimes, they may reside in the same system. A pictorial representation of client-server computing is presented in Figure 3.1.

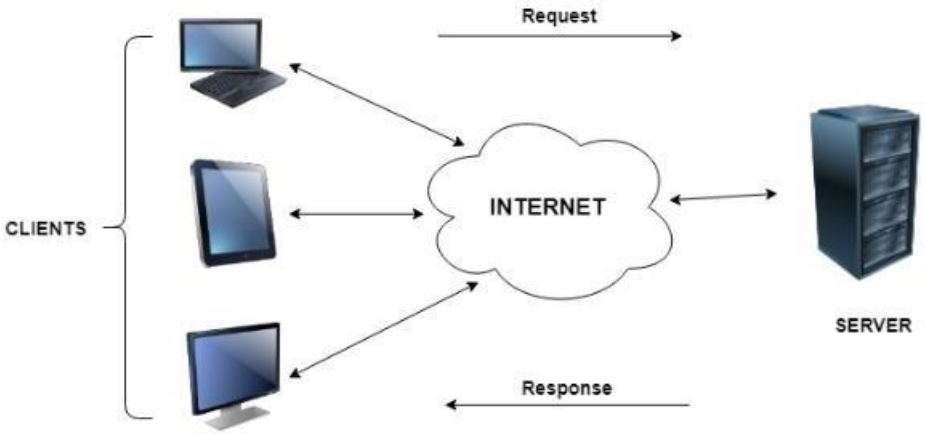

Figure 3.1: Client-Server Computing

4.1.1 Characteristics of Client-Server Computing.

The major characteristics of client-server computing are as follows:

- The client server computing works with a system of request and response. The client sends a request to the server and the server responds with the desired information.
- The client and server should follow a common communication protocol so they can easily interact with each other. All the communication protocols are available at the application layer.
- A server can only accommodate a limited number of client requests at a time. So it uses a system based to priority to respond to the requests.

- Denial of Service (DoS) attacks hinders servers' ability to respond to authentic client requests by inundating it with false requests.

- An example of a client server computing system is a web server. It returns the web pages to the clients that requested them.

4.1.2 Differences between Client-Server and Peer-to-Peer Computing

The major differences between client-server computing and peer-to-peer computing are as follows:

- In client server computing, a server is a central node that services many client nodes. On the other hand, in a peer-to-peer system, the nodes collectively use their resources and communicate with each other.

- In client server computing, the server is the one that communicates with the other nodes. In peer-to-peer computing, all the nodes are equal and share data with each other directly.

- Client-Server computing is believed to be a sub-category of the peer-to-peer computing.

4.1.3 Advantages of Client-Server Computing

- All the required data is concentrated in a single place i.e. the server. So it is easy to protect the data and provide authorization and authentication.

- The server need not be located physically close to the clients yet, the data can be accessed efficiently.

- It is easy to replace, upgrade or relocate the nodes in the client-server model because all the nodes are independent and request data only from the server.

- All the nodes i.e. clients and server may not be built on similar platforms yet, they can easily facilitate the transfer of data.

4.1.4 Disadvantages of Client-Server Computing

- If all the clients simultaneously request data from the server, it may get overloaded. This may lead to congestion in the network.

- If the server fails for any reason, then none of the requests of the clients can be fulfilled. This leads to failure of the client-server network.

- The cost of setting and maintaining a client-server model are quite high.

Summary

Client server and peer-to-peer computing are unique one from the other and so, have their merits and demerits. The choice of either is dependent on the intention of creating your network.

In client server computing the server is the one that communicates with the other nodes. In peer to peer to computing, all the nodes are equal and share data with each other directly. A server can only accommodate a limited number of client requests at a time. So it uses a system based to priority to respond to the requests. The characteristics of the client-server computing are as follows: the client server computing works with a system of request and response, client and server should follow a common communication protocol so they can easily interact with each other, a server can only

accommodate a limited number of client requests at a time and that Denial of Service (DoS) attacks hinders servers' ability to respond to authentic client requests by inundating it with false requests.

Assessment

1. Discuss the characteristics of client-server computing.
2. Discuss the advantages and disadvantages of client-server computing

TERM PAPER/PRESENATION

Title: Building a Web Application.

Design a web-based application of your choice according to your grouping.
Develop a term paper stating how your web-based application was designed and developed.

BULIDING WEB APPLICATIONS

5.0 Introduction

This section entails discussions on the concept of a web-based application, prerequisite for building a web-based application, and description of steps for building a web-based application.

A Web-based Application

An interactive computer program, built with web technologies (HTML, React, CSS, JS), which stores (Database, Files) and manipulates data (CRUD), and is used by a team or single user to perform tasks over the internet. The HTML and the CSS serves as the front-end to receive data from the user while the database, programming like Javascript and PHP serves as the back-end.

5.1 Prerequisites for Building a Web-based Application.

To make a data-centric web app from the bottom-up, it is advantageous to understand:

- Backend language (e.g. Python, Ruby) - control how your web app works
- Web front end (HTML, CSS, Javascript, React) - for the look and feel of your web app
- DevOps (Github, Jenkins) - Deploying / hosting your web app

If you do not have any experience with the points above, you need not worry. You have two options:

1. Learn the points above - there are lots of resources online to help you. Codecademy is recommended.
2. Use a web app builder like Budibase. As a builder, Budibase will remove the need to learn a backend language. On top of that, Budibase will also take care of a lot of your DevOps tasks such as hosting.

5.2 Steps in Building a Web Application.

Step 1: Source an Idea

Before making a web app, you must first understand what you intend to build and more importantly, the reason for it.

Your idea should stem from solving someone's problem. Ideally, your own problem.

It is important that developer choose an idea which interests him /her. Ask yourself:

- How much time do I have to build this app?
- What am I interested in?
- What apps do I enjoy using?
- What do I like about these apps?
- How much time/money will this app save or generate for me (as a user)?
- How much will it improve my life

Step 2: Market Research

Once you have chosen your idea(s), it is important to research the market to see:

1. If a similar product exists
2. If a market exists

The number one reason start-ups fail, is the failure to achieve product- market fit. "**Product/market fit** means being in a good market with a product that can satisfy that market." To quickly find out if a similar web app exists, use the following tools/links to search for your idea:

- Google
- Patent and trademark search
- Betalist
- Product hunt

If a similar product exists, do not worry. This can be a sign a market for your new idea exists. Your future competitors have laid the groundwork, educated the market. It is time for you to swoop in and steal the thunder. If a similar product does not exist, it is a possibility you have struck lucky. On the other hand, it is a possibility someone before has ventured down this path and hit a dead-end. Nobody wants to experience that, so it is important to dive deep into the market and source the wisdom of:

1. Your Web App's target market - Share your web app idea on forums related to your target market. If you know anyone who works within your target market, explain your idea to them. The more you talk and receive validation from your target market, the better.
2. Google Trends - A quick search of your web app idea will reveal relating trends.
3. SEO tool – MOZ/Ahrefs is recommended. Google's keyword planner will suffice. Write a list of keywords relating to your web app. If it is an 'OKR tool', use the tools to search

'OKR tool', 'OKR app', and 'objectives and key results software'. If the SEO tool indicates there are lots of people searching for your keyword terms, this is a small indicator you have a target market.

4. Social Media - Jump over to Twitter/Facebook groups and present your idea to your target market.

5. Events - If there is a local event in your area attracting people from your target market, go to it. Share your idea and record the feedback.

After completing the above steps, you should have enough information to understand if there is a market for your product. If there is a market for your product, and there is also established competition, it is important to research them.

Step 3 - Define your Web Application's Functionality

You have got your idea, you have validated the market, it is now time to list everything you want your app to do. A common mistake here is to get carried away. The more functionality you add, the longer it will take to build your web app. Quite often, the longer a web app takes to build, the more frustration you will experience.

Only define functionality which solves your target markets problems. Remember, your web app is a work in progress and the first goal is version. It will still have cool features and delight your users, but you must keep things simple. For instance, to build a customer relationship management (CRM) web-based application for an organization, the following basic functions are required.

- Users can create an account
- Users can retrieve lost passwords
- Users can change their passwords
- Users can create new contacts
- Users can upload new contacts
- Users can assign a value to contacts
- Users can write notes under contacts
- Users can label a contact as a lead, customer, or associate
- Users can filter contacts by lead, customer, or associate
- Users can view the total value of leads, customers and associates

The above list will help you define your features. Once you are done, roll up your sleeves. It is time to get creative! **Moving from the Ideation stage, to design stage.**

Step 4: Sketch your Web Application

There are multiple stages of designing a web app. The first stage is sketching using a notebook (with no lines) and pen/pencil. After steps 1, 2 and 3 discussed previously, you should have an idea of what your web app is, who your users are, and the features it will have. Sketch out the wireframe of your web apps UI (User Interface) - it does not have to be exact - this is just a sketch. When sketching, consider the following:

- Navigation
- Branding
- Forms

- Buttons
- Any other interactive elements

Sketch different versions of your web application. Consider how your web app's functionality will affect the overall design. Annotate your sketch and outline how your app should work. Taking notes will help you clarify and understand why you have designed certain elements at a later stage. Overcomplicating the design at this stage will only lead to frustration.

Step 5 – Plan your web apps workflow

It is time to put yourself in the shoes of your user. Here, we are going to plan your web apps workflow. Now is the time to go back to step 2 and look at your market research. Take your list of competitors and sign up to their free trials. Have a quick play around with their product. Write notes on what you thought was good and what you thought was bad. Pay particular attention to the workflow.

After you have finished analysing your competitor's web apps, it is time to write down different workflows for your app. Consider the following points:

- How does a user signup
- Do they receive a verification email
- How does a user log in
- How does a user change their password
- How does a user navigate through the app
- How does a user change their user settings
- How does a user pay for the app
- How does a user cancel their subscription

All of a sudden our one-page web app turns into a 10-page web app. Write a list of all the different pages your web application will have. Consider the different states of pages. For example, the homepage will have two states; logged in and logged out. Logged in users will see a different page than logged out users.

Step 6 – Wireframing / Prototyping Your Web Application

It is time to turn those sketches and that new-found understanding of your web application into a wireframe/prototype.

Wireframing is the process of designing a blueprint of your web application while Prototyping is taking wireframing a step further, adding an interactive display.

The decision to wireframe or prototype is up to you. If you have the time, I would have recommended prototyping as it will make it easier to communicate your web app when seeking validation.
You can prototype/wireframe using the following tools:

- Sketch (macOS)
- InVision Studio (macOs)
- Adobe XD (macOS, Windows)

- Figma (Web, macOS, Windows, Linux)
- Balsamiq (macOS, Windows, Web)

It is recommend you create a design system / style guide first. You can find inspiration at UXPin. Design systems improve design consistency.

Step 7: Seek early validation

By now you must have gotten a beautiful wireframe/prototype which visually describes your web app. It is time to show your beautiful wireframe to the world. At this stage we want constructive feedback.

Simply asking your friends would they use your new web app is not enough. You should start with a small number of representative users. Go to your target market's forums, watering holes, their places of work and verify the problem with them, and present your solution. Try to build a rapport with these representatives as they could become your customers. I like to use this stage to test my sales pitch - the ultimate tokens of validation are pre-launch sales. Takes notes and document all feedback. The learning from these meetings will help direct the development of your MEP (Minimal Excellent Product).

Tips before Starting the development stage.

Before you make a web app, I would like to share the following tips:

1. Attempt to get a small section of your app fully working. What we would call a "Complete Vertical".
    - Building the smallest possible section will allow you to piece all the bits together, and iron out those creases early.
    - You will get great satisfaction early by having something working - great motivation.
    - Create things that you know you will throw away later - if it gets you something working now.
2. At the start - expect things to change a lot as you learn and discover what you have not thought about.
    - Have faith that your app will stabilise.
    - Do not be afraid to make big changes.
3. Spend time learning your tools.
    - You may feel like you are wasting your time, reading, or experimenting with "hello world". Learning the correct way to do things will have a huge positive, cumulative effect on your productivity over time.
    - Where possible, "Go with the grain" of your tools. Realise that as soon as you step out of the normal flow / usage of your toolset, you are on your own and could be in a deep time sink. There are always exceptions to this of course!
4. Do not avoid issues that need to be fixed.
    - Face your issues head on - they will never go away and will only grow in stature.
    - However, If things are still likely to change - its best to spend as little time as possible on things… It's a tricky balance!

Step 8: Architect and build your database

We know roughly our web application's functionality, what it looks like, and the pages required. Now it is time to determine what information we will store in our database.

A Database

A database is simply a collection of data! Data can be stored to disk, or in memory on a server, or both. You could create a folder on your hard drive, store a few documents, and call it a database. A Database Management System (DBMS) is a system that provides you with consistent APIs to (most commonly):

- Create databases, update and delete databases
- Read and write data to databases
- Secure access to a database by providing levelled access to different areas and functions

What data you need to store and what your users need to do, will determine the type of database required to run your web app.

Database Types

There are many types of database for many different purposes. A web app will most commonly use one of the following:

SQL

You should use a SQL database if your data is very relational. Your data is relational if you have multiple, well defined record types that have relationships between them. For example, a "Customer" may have many "Invoices" stored against their record. Typically, you would create a Customer table and an Invoice table - which could be linked together by "Foreign Key" columns. E.g. `Customer.Id = Invoice.CustomerId`.

SQL databases have an extremely powerful query language that allows you to present your data in all sorts of useful ways. They have been around for decades, are very well understood, and usually a safe choice. MySQL, Postgresql, Microsoft SQLServer, MySQL workbench are some of the most common - along with many more modern offerings.

The downside of SQL databases is that you must declare all your tables and columns up front. There can be a lot of overhead to manage. If you have never used one before – you are in for a pretty steep learning curve. However, there are plenty of learning resources available, and it is always a great skill to have.

Document Database

You should use a document database if your data is not very relational. Document databases store "documents". Each record in your database is simply a big blob of structured data - often in JSON format. If you need to store relationships between your records, you will have to write code to manage this yourself. However, many other aspects of using document databases are much simpler. Your

database can be “schemaless” - meaning that you do not have to declare your records’ definitions up front. Generally speaking, the bar to entry to a document database is much lower. They also tend to be much more scalable than SQL databases. They usually offer some querying capabilities, although sometimes not as powerful as SQL. Examples of document databases are: MongoDb, CouchDb, Firebase (serverless), Dynamo Db (AWS).

Decide how to segregate your data

Each of your clients has their own, private dataset. One of the worst things that can happen to your app is for one client’s data to be seen by another client.

Even if there is only a small amount of non-sensitive leaked data, and no damage is done, an event like this will massively erode trust in the security of your app. You must architect a solid strategy for segregating your clients’ data to make sure that this never happens. Broadly speaking, you have two options - Physical Separation and Logical Separation.

Physical separation

Every one of your clients has a separate database (although could share a database server with others). This makes it much more difficult to make a mistake that leads to data leakage.

Pros:

- Most secure
- More scalable

Cons:

- Managing, maintaining and upgrading is more complex
- Query all your clients’ data together is more difficult

For example, listing all Invoices in a database will only return Invoices for one of your clients. In order to get another Client’s invoices, you need to connect to another database.

Since each of your client’s data is in its own database, you can easily spread them all across many database servers, without the need for “sharding”. Your app will be much easier to scale this way.

The code you will need to write:

- When creating a new client, you need to create a new database and populate with any seed data.
- You need to keep a record somewhere of all your clients, and how to connect to each client’s database.
- If you need to upgrade your database (e.g. add a new table), you need to code to upgrade each separately.
- If you need to query all your client’s data into one, you need to pull the data out of each and aggregate it.

Logical separation

All of your clients are stored in one giant database. Every time you need to get data for a single client, you must remember to include a filter for the client. E.g. ‘select’ from customers where `customerClientId = “1234”`

Pros:

- Easier to get started
- Easier to maintain and upgrade
- Can easily query all your clients' data with one query

Cons:

- Easy to make a mistake that will result in a data breach
- More difficult to scale

You now only have one database to manage. Setting this up and connecting to your database is easy. Your speed to market increases. When you need to upgrade your database, you can do so with a few clicks, or by typing a few commands. It is very easy to add new features. As you gain more users, your database will grow to millions of rows. Put some effort into how your database handles this extra volume and load. You will have to start tuning your queries.
When you're under pressure, it is so easy to forget to include that "`where clientId = 1234`" filter. Doing so could result in a business ending data breach.

Ensure your database is secured. You should look into best practices for securing your particular database. Some databases come with a default administrator login, which people often forget to change. This could leave your data open to the world.
From the start, you should create a login with "Just Enough" access. If your app only reads and writes data, then it should authenticate to your database using a login with only data reading and writing access.

Step 9: Build the frontend

Note: In reality, you will build your backend and frontend at the same time. But for this post, we'll keep it simple.

A frontend

The Frontend is the visual element of your web application. It defines what you see and interact with. The frontend is developed with HTML, CSS, and JavaScript.

If using server pages, getting started is super easy. Your backend framework is all set up and ready to start building. This is where the huge benefit lies with server pages.
With SPA, it's a little trickier.
First, you need to set up your development environment. The components of this will be:

1. A code editor, such as VS Code, Sublime Text
2. A compilation, and packaging framework:
    - Webpack
    - Gulp
    - Grunt

This is also used for serving and "Hot Loading" your application at development time, on a nodejs web server, running on localhost.

3. A frontend framework (strictly not necessary, but highly advised unless you are an experienced frontend developer):
    - React
    - Ember

- Vue
- Svelte

The list is endless!

4. Configuring your packaging tool to talk to your backend - which is most likely running on a different port on localhost. Usually, this is done using Node's HTTP proxy. Most packaging solutions have this option built-in, or available as plugins. This point commonly gets people stuck, and may need a diagram. Remember - if you write your backend API in C Sharp (for example) then at development time, you will be running it on a local web server, through your code editor. i.e. your frontend and backend are running on 2 different web servers, in development. However, in production, your frontend should (probably) be running on the **same** web server as your backend - mainly because you want them to run under the same domain.

This means a few things:

- At dev (development) time, your frontend should make API requests to its own (Nodejs server - e.g. Webpack dev server). This Nodejs server should then proxy all "/api" request to your backend server.
- When building for production, you need to get your compiled frontend files into your backend server - so they can be served as static files. You can copy and paste the files in when you deploy, but you will want to set up some sort of script to do this.

There is always a significant time required to set up your dev (development) environment for a SPA. There are plenty of boilerplate templates out there for your frameworks of choice. However, I have never written an app that has not eventually needed some custom code on top of the boilerplate.
Still, always choose a SPA.

- The end product for a web app is a much more usable application.
- When you are up and running with your dev environment, I find SPAs much more productive to work with - which is more likely to do with the capabilities of modern javascript frameworks than anything else.
- Writing a SPA is really the only way to make a Progressive Web Application.

You should now have a better idea of how to setup your frontend and define the look and feel of your web app. In most cases, I build the frontend and backend together.

Step 10: Build your Backend

The backend is typically what manages your data. This refers to databases, servers, and everything the user cannot see within a web application. Building your backend is one of the toughest parts of web app development. If you feel overwhelmed, a tool like Budibase can take away many of the complexities - including the following tasks. If you feel confident, continue.

When building your web application, you need to choose between:

- Server Pages (Multiple Page Application) and
- Single Page Application

"But is not this the frontend?" - I hear you say. Yes! But your choice will affect how you develop your backend.

The primary jobs of the backend will be to:

- Provide HTTP endpoints for your frontend, which allow it to operate on your data. E.g. Create, Read, Update and Delete ("CRUD") records.
- Authenticate users (verify they are who they say they are: a.k.a log them in).
- Authorization. When a logged in user makes a request, the backend will determine whether they are allowed (authorized) to perform the requested action.
- Serve the frontend.

If you have chosen Server Pages, your backend will also be generating your frontend and serving it to your user.
With a single page app, the backend will simply serve your static frontend files (i.e. your "Single Page" and it is related assets).
When choosing your backend:

- Go with what is familiar.
- Try Budibase
- Server Pages / SPA should inform your decision of framework choices within your chosen language. For example, a SPA will only require an API only framework. Server pages need their own framework.
    - Django
    - Express
    - Flask

Login/User & Session Management

- How will users authenticate?
    - Username and password?
    - Open ID (i.e. sign in as Google, FB, etc)
- Be sure to read up on security best practices. I highly recommend: OWASP
    - What user levels will you create in the system?

Step 11: Host your web application

Hosting involves running your web app on a particular server. When using Budibase, this step can be automated with Budibase hosting . With Budibase, you are still required to buy a domain. If you are not using Budibase to host your web application, follow these quick steps:

1. Buy a domain - Namecheap
2. Buy/Setup an SSL certificate - Let's Encrypt
3. Choose a cloud provider:
    - Amazon
    - MS Azure
    - Google Cloud Platform
    - Lower cost: Digital Ocean / Linode - if you are happy managing your own VMs
    - Zeit Now, Heroku, Firebase are interesting alternatives that aim to be faster and easier to get things done - you should read about what they offer.

Choosing one of these hosting options will almost certainly provide you with everything you need. They have ample documentation and community supports, and are generally reliable options.

Step 12 - Deploy your web app

You have sourced your idea, validated it, designed and developed your web app, and chosen your hosting provider. You are now at the last step. Well done!

The deployment step includes is how your web application gets from your source control on your computer to your cloud hosting from step 11. How does your application get from Source Control / Your computer to your cloud hosting provider?
The following development tools provide continuous integration and will help you with deploying your web app to your cloud hosting:

- GitLab
- Bitbucket
- Jenkins

To start with, you can just deploy directly from your machine of course. You have made a web application. Well done. You should take some time to celebrate this achievement. You are the proud owner of a new web app.

Summary

Building a Web Application steps include: Source an idea, Market Research, Define your web apps functionality, Sketch your web app, Plan your web apps workflow, Wireframing / Prototyping Your Web Application, Seek early validation, Architect and build your database, Build the frontend, Build your backend, Host your web application and Deploy your web app. To make a data-centric web app from the bottom-up, it is advantageous to understand: Backend language (e.g. Python, Ruby) - control how your web app works; Web front end (HTML, CSS, Javascript) - for the look and feel of your web app; and DevOps (Github, Jenkins) - Deploying / hosting your web app. There are many types of database for many different purposes but a web app will most commonly use one of SQL and Document database. The backend is typically what manages your data. This refers to databases, servers, and everything the user cannot see within a web application.

Building your backend is one of the toughest parts of web app development. If you feel overwhelmed, a tool like Budibase can take away many of the complexities - including the follow tasks.

Self-Assessment Questions

1. Mention and explain the Database types in relation to building a web application.
2. Explain backend, backend types and what determines a choice of backend

PARALLEL SYSTEMS

Introduction

Parallel computing is a type of computation in which many calculations or processes are carried out simultaneously. Large problems can often be divided into smaller ones, which can then be solved at the same time. There are several different forms of parallel computing: bit- level, instruction-level, data, and task parallelism. Parallelism has long been employed in high-performance computing, but has gained broader interest due to the physical constraints preventing frequency scaling. As power consumption (and consequently heat generation) by computers has become a concern in recent years, parallel computing has become the dominant paradigm in computer architecture, mainly in the form of multi- core processors.

The topics of interest in this section are:

- Introduction to parallel systems.
- Parallel Programming models.
- Message passing programming
- Dependence analysis
- OpenMP programming
- Evaluation of programs.

6.1 Parallel Processing Systems

Parallel systems deal with the simultaneous use of multiple computer resources that can include a single computer with multiple processors, a number of computers connected by a network to form a parallel processing cluster or a combination of both.

Parallel Processing Systems are designed to speed up the execution of programs by dividing the program into multiple fragments and processing these fragments simultaneously. Such systems are multiprocessor systems also known as tightly coupled systems. Parallel systems deal with the simultaneous use of multiple computer_resources that can include a single computer_with multiple processors, a number of computers connected by a network to form a parallel processing cluster or a combination of both. Parallel computing is an evolution of serial computing where the jobs are broken into discrete parts that can be executed concurrently.

Each part is further broken down to a series of instructions. Instructions from each part execute simultaneously on different CPUs. Parallel systems are more difficult to program than computers with a single processor because the architecture of parallel computers varies accordingly and the processes of multiple CPUs must be coordinated and synchronized. Several models for connecting processors and memory_modules exist, and each topology requires a different programming model. The three models that are most commonly used in building parallel computers include synchronous processors each with its own memory, asynchronous processors each with its own memory and asynchronous processors with a common, shared memory.

6.2 Flynn's Classification of Parallel Systems.

Flynn has classified the computer systems based on parallelism in the instructions and in the data streams. These are:

- Single Instruction, Single Data Stream (SISD)
- Single Instruction, Multiple Data Stream (SIMD)
- Multiple Instruction, Single Data Stream (MISD)
- Multiple Instruction, Multiple Data Stream (MIMD)

6.2.1 Single Instruction, Single Data Stream (SISD)

An SISD computing system is a uniprocessor machine capable of executing a single instruction, which operates on a single data stream. In SISD, machine instructions are processed sequentially; hence computers adopting this model are popularly called *sequential computers*. Most conventional computers are built using the SISD model. All the instructions and data to be processed have to be stored in primary memory. The speed of the processing element in the SISD model is limited by the rate at which the computer can transfer information internally. Dominant representative SISD systems are IBM PC, Macintosh, and workstations. A diagrammatic representation of the SISD is presented in Figure 5.1

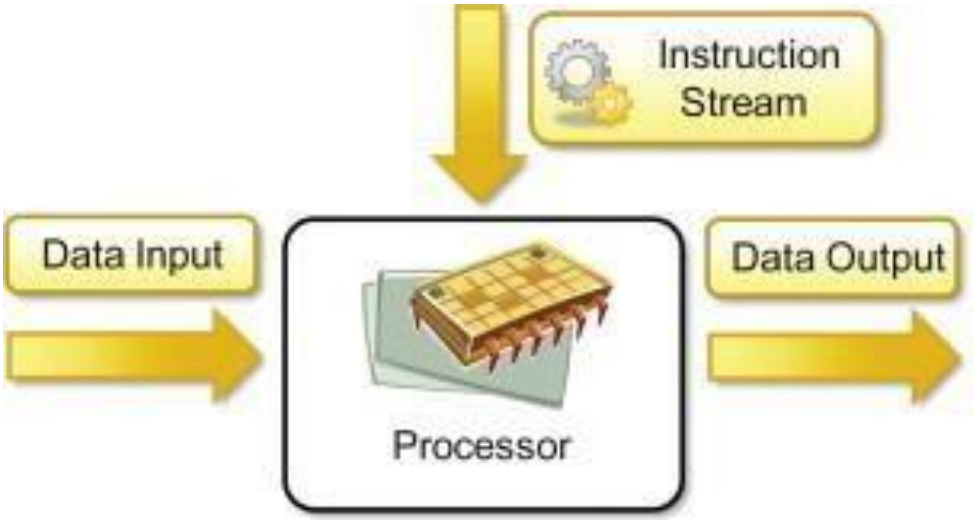

Figure 5.1: Diagrammatic Representation of a Single Instruction, Single Data Stream

6.2.2 Single Instruction, Multiple Data Stream (SIMD)

The Single Instruction, Multiple Data Stream (SIMD) model of parallel computing includes two parts such as a front-end computer of the usual von Neumann style, and a processor array. The processor array is a collection of identical synchronized processing elements adequate for simultaneously implementing the same operation on various data. Each processor in the array has a small amount of local memory where the distributed data resides while it is being processed in parallel. The processor array is linked to the memory bus of the front end so that the front end can randomly create the local processor memories as if it were another memory. Figure 5.2 gives a diagrammatic representation of the SIMD.

A program can be developed and performed on the front end using a traditional serial programming language. The application program is performed by the front end in the usual serial method, but problem command to the processor array to carry out SIMD operations in parallel. The similarity between serial and data-parallel programming is one of the valid points of data parallelism. Synchronization is created irrelevant by the lock-step synchronization of the processors. Processors either do nothing or similar operations at the same time.

In SIMD architecture, parallelism is exploited by using simultaneous operations across huge sets of data. This paradigm is most beneficial for solving issues that have several data that require to be upgraded on a wholesale basis. It is dynamically powerful in many regular scientific calculations.

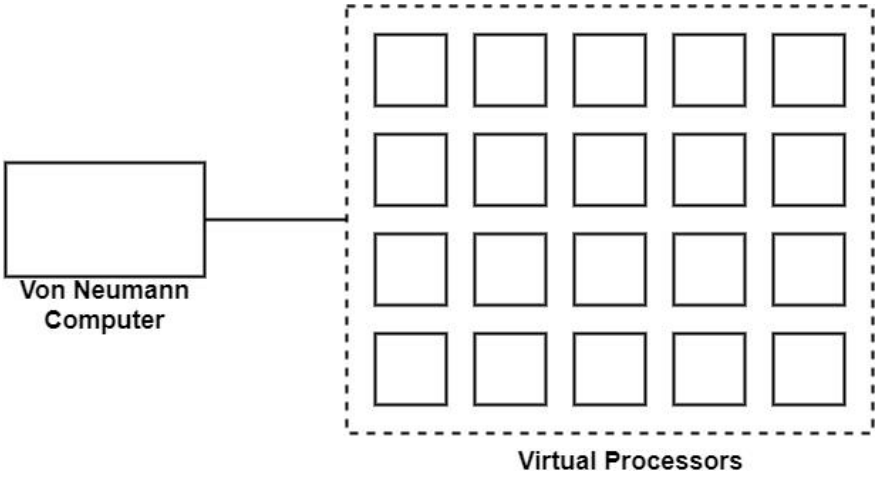

Figure 5.2: Single instruction stream, multiple data stream (SIMD)

Two main configurations have been applied in SIMD machines. In the first scheme, each processor has its local memory. Processors can interact with each other through the interconnection network. If the interconnection network does not support a direct connection between given groups of processors, then this group can exchange information via an intermediate processor. A diagrammatic representation of the first scheme is presented in Figure 2.2.1

In the second SIMD scheme, processors and memory modules communicate with each other via the interconnection network. Two processors can send information between each other via intermediate memory module(s) or possibly via intermediate processor(s). The BSP (Burroughs' Scientific Processor) used the second SIMD scheme. A diagrammatic representation of the first scheme is presented in Figure 5.2.2.

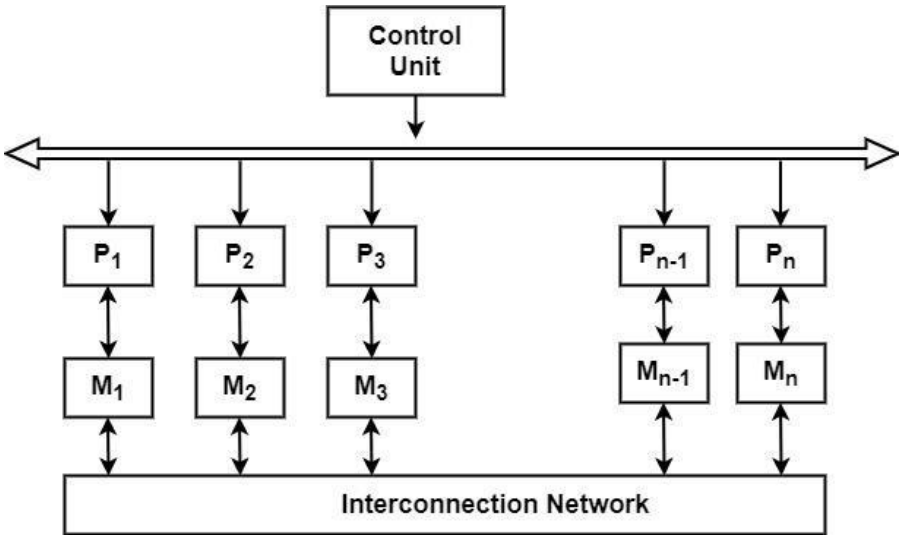

Figure 5.2.1: Single instruction stream, multiple data stream (SIMD) Scheme-1

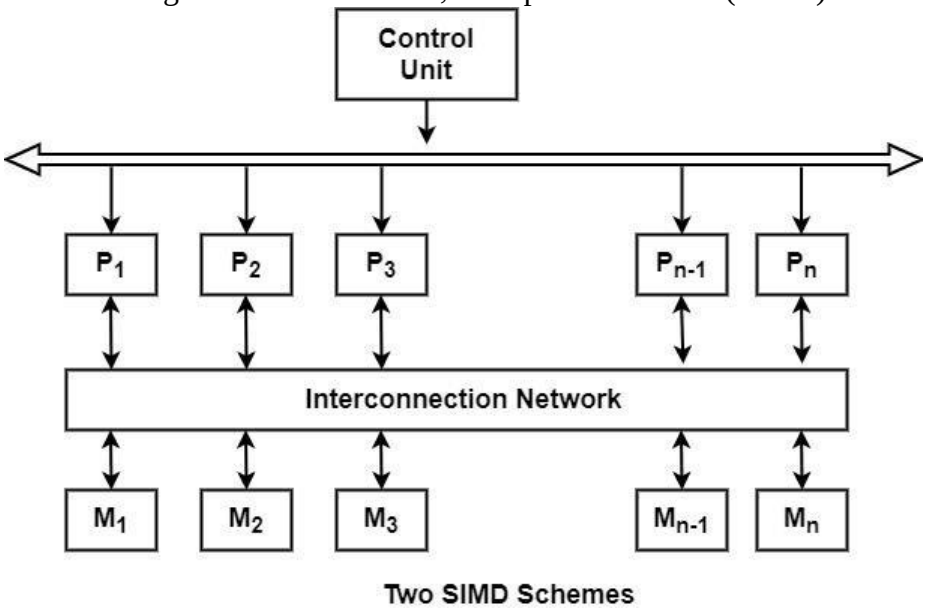

Figure 5.2.2: Single Instruction, Multiple Data stream (SIMD) Scheme-2

6.2.3 Multiple Instruction, Single Data stream (MISD).

In this association, multiple processing elements are structured under the control of multiple control units. Each control unit is handling single instruction stream and processed through its corresponding processing element. But single processing element is processing only a one data stream at a time. Hence, for handling single data stream and multiple instruction streams, multiple processing elements and multiple control units are organised in this classification. All processing elements are relate with the common shared memory for the organisation of one data stream as given in Figure 5.3. The only identified instance of a computer capable of MISD operation is the C.mmp built by Carnegie-Mellon University.

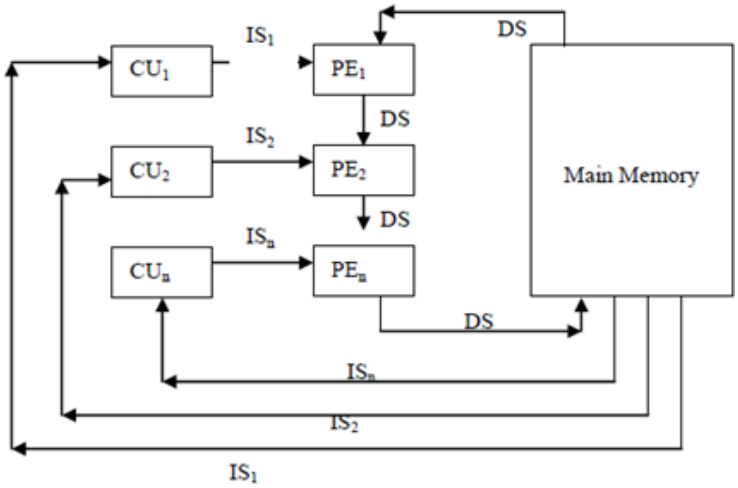

Figure 5.3: Multiple-Instruction Single-Data streams (MISD)

The MISD classification is not popular in commercial machines as the thought of single data streams implementing on multiple processors is rarely functional. But for the particular applications, MISD organisation can be very useful. For example, Real time computers need to be fault tolerant where several processors implement the same data for producing the redundant data. This is also called as N- version programming. All these redundant data are measured to as results which should be similar; otherwise faulty unit is returned. Thus MISD machines can be useful to fault tolerant real time computers.

6.2.4 Multiple Instruction, Multiple Data stream (MIMD).

MIMD stands for Multiple-instruction multiple-data streams. It includes parallel architectures are made of multiple processors and multiple memory modules linked via some interconnection network. They fall into two broad types including shared memory or message passing. A shared memory system generally accomplishes interprocessor coordination through a global memory shared by all processors. These are frequently server systems that communicate through a bus and cache memory controller.

The bus/ cache architecture alleviates the need for expensive multi-ported memories and interface circuitry as well as the need to adopt a message- passing paradigm when developing application software. Because access to shared memory is balanced, these systems are also called SMP (symmetric multiprocessor) systems. Each processor has an equal opportunity to read/write to memory, including equal access speed. A diagrammatic representation of the MIMD is presented in Figure 5.4.

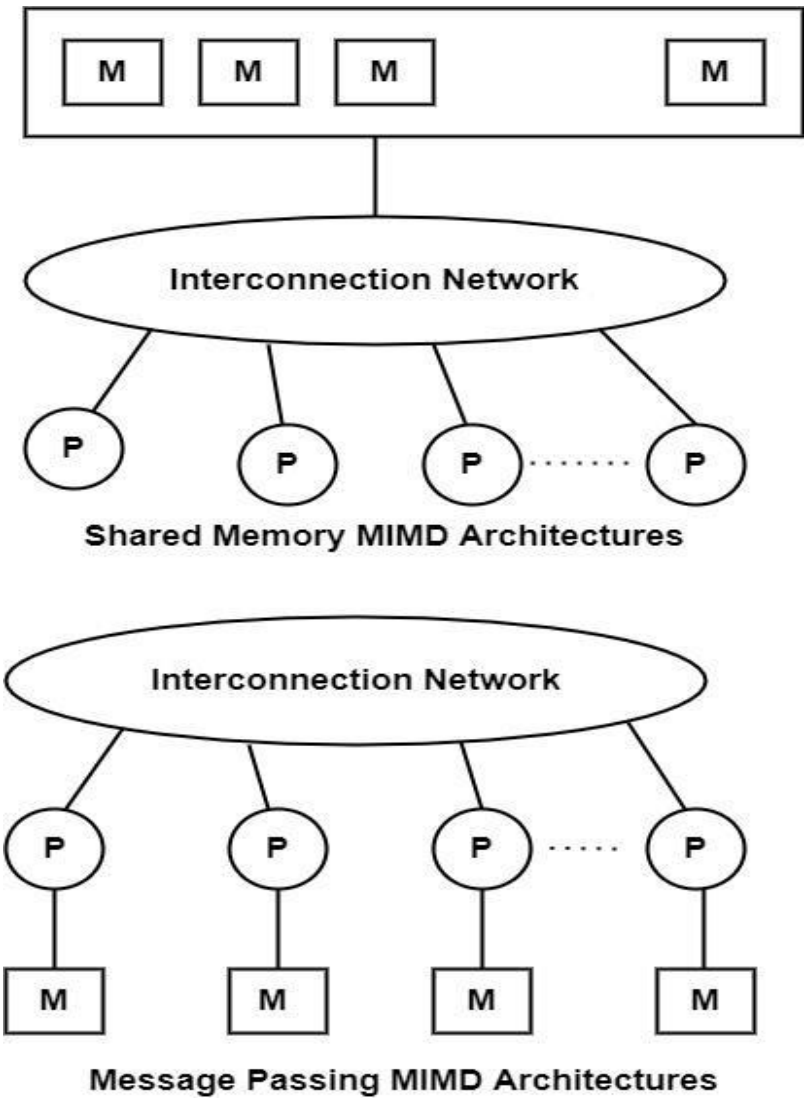

Figure 5.4: Multiple-Instruction Multiple-Data streams (MIMD)

The above (MIMD) classification of parallel computing system is focused in terms of two independent factors: the number of data streams that can be simultaneously processed, and the number of instruction streams that can be simultaneously processed. Here, by 'instruction stream' we mean an algorithm that instructs the computer what to do whereas 'data stream' (i.e. input to an algorithm) means the data that are being operated upon.

6.3 Relevance of Flynn's Classification to Parallel Systems

Even though Flynn has classified the computer 'systems into four types based on parallelism but only two of them are relevant to parallel computers. These are SIMD and MIMD computers. SIMD computers are consisting of 'n' processing units receiving a single stream of instruction from a central control unit and each processing unit operates on a different piece of data. Most SIMD computers operate synchronously using a single global clock. The block diagram of SIMD computer is shown in Figure 5.5.

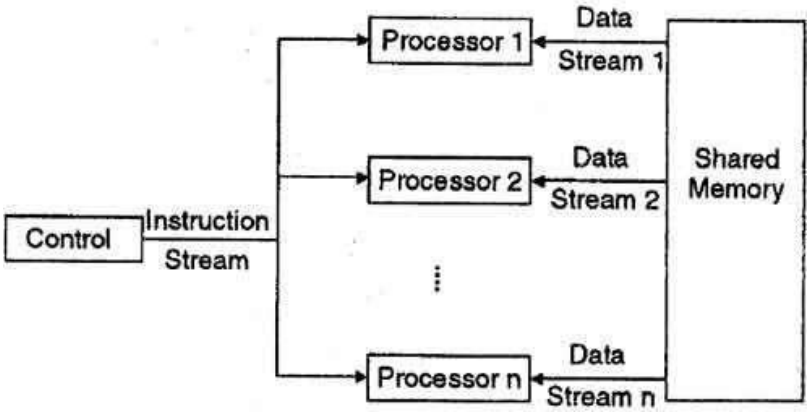

Figure 5.5: Single Instruction, Multiple Data stream (SIMD) Block Diagram

MIMD computers are consisting of 'n' processing units; each with its own stream of instruction and each processing unit operate on unit operates on a different piece of data. MIMD is the most powerful computer system that covers the range of multiprocessor systems. The block diagram of MIMD computer is shown in Figure 5.6.

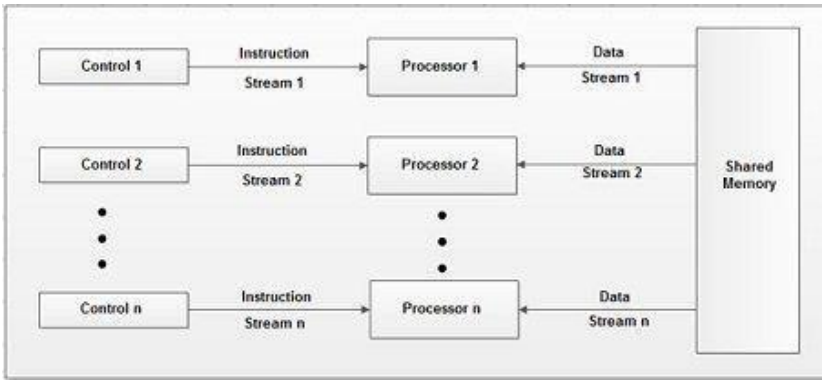

Figure 5.6: Multiple Instruction, Multiple Data stream (MIMD) Block Diagram

The SIMD systems are easier to program because it deals with single thread of execution. On the hand, the MIMD machines are more efficient because you can utilize the full machine power.

5.4 Parallel Computers and Applications.

Parallel operating systems are primarily concerned with managing the resources of parallel machines. A parallel computer is a set of processors that are able to work cooperatively to solve a computational problem. So, a parallel computer may be a supercomputer with hundreds or thousands of processors or may be a network of workstations.
A few years ago, parallel computers could be found only in research laboratories and they were used mainly for computation intensive applications like numerical simulations of complex systems. Today, there are a lot of parallel computers available in the market; used to execute both data intensive applications in commerce and computation intensive applications in science and engineering.

Today, new applications arise and demand faster computers. Commercial applications are the most used on parallel computers. A computer that runs such an application should be able to process large amount of data in sophisticated ways. These applications include graphics, virtual

reality, and decision support, parallel databases, medicine diagnosis and so on. We can say with no doubt that commercial applications will define future parallel computers architecture but scientific applications will remain important users of parallel computing technology.

Concurrency becomes a fundamental requirement for algorithms and programs. A program has to be able to use a variable number of processors and also has to be able to run on multiple processors computer architecture. According to Tanenbaum, a distributed system is a set of independent computers that appear to the user like a single one. So, the computers have to be independent and the software has to hide individual computers to the users. MIMD computers and workstations connected through LAN and WAN are examples of distributed systems. The main difference between parallel systems and distributed systems is the way in which these systems are used. A parallel system uses a set of processing units to solve a single problem. A distributed system is used by many users together.

Self-Assessment Questions.

1. What is parallel systems?
2. List and explain the Flynn's classification of parallel systems.
3. Explain the relevance of Flynn's classification of parallel systems.

PARALLEL PROGRAMMING MODELS

7.1 Introduction to Parallel Programming Model

A **parallel programming model** is an abstraction of parallel computer architecture, with which it is convenient to express algorithms and their composition in programs. The value of a programming model can be judged on its generality: how well a range of different problems can be expressed for a variety of different architectures, and its performance: how efficiently the compiled programs can execute. The implementation of a parallel programming model can take the form of a library invoked from a sequential language, as an extension to an existing language, or as an entirely new language. Consensus around a particular programming model is important because it leads to different parallel computers being built with support for the model, therebyfacilitating portability of software. In this sense, programming models are referred to as bridging between hardware and software.

A parallel programming model is a set of program abstractions for fitting parallel activities from the application to the underlying parallel hardware. It spans over different layers: applications, programming languages, compilers, libraries, network communication, and I/O systems. Two widely known parallel programming models are:

1. Shared memory and
2. Message passing.

In some cases, there are also different combination of both.

7.1.1 Shared Memory Programming Model

In the shared-memory programming model, tasks share a common address space, which they read and write in an asynchronous occurring at different times (i.e. having many actions occurring at a time, in any order, without waiting for each other) manner. The communication between tasks is implicit. If more than one task accesses the same variable, the semaphores or locks can be used for synchronization. By keeping data local to the processor and making private copies, expensive memory accesses are avoided, but some mechanism of coherence maintenance is needed when multiple processors share the same data with the possibility of writing.

Shared memory based parallel programming models communicate by sharing the data objects in the global address space. Shared memory models assume that all parallel activities can access all of memory. Consistency in the data need to be achieved when different processors communicate and share the same data item, this is done using the cache coherence protocols used by the parallel computer. All the operations on the data items are implicit to the programmer. For shared memory based parallel programming models communication between parallel activities is through shared mutable state that must be carefully managed to ensure correctness. Various synchronization primitives such as locks or transactional memory are used to enforce this management.
The advantages with using shared memory based parallel programming models are presented below:

- Shared memory based parallel programming models facilitate easy development of the application than distributed memory based multiprocessors.

- Shared memory based parallel programming models avoids the multiplicity of the data items and the programmer doesn‘t need to be concerned about that, which is the responsibility of the programming model.
- Shared memory based programming models offer better performance than the distributed memory based parallel programming models.

The disadvantages with using the shared memory based parallel programming models are described below:

- The hardware requirements for the shared memory based parallel programming models are very high, complex and cost oriented.
- Shared memory parallel programming models often encounter data races and deadlocks during the development of the applications.

7.1.2 Message-Passing Programming Model

In the message-passing programming model, tasks have private memories, and they communicate explicitly via message exchange. To exchange a message, each sends operation needs to have a corresponding receive operation. Tasks are not constrained to exist on the same physical machine. Message passing models are also known as parallel programming models. These are the kind of models that allows communication between processors by using the send/receive communication routines. Message passing models avoids communications between processors based on shared/global data. Typically used to program clusters, where each processor in the architecture gets its own instance of data and instructions.
The advantages of distributed memory based programming models are:

- The hardware requirement for the message passing models is low, less complex and comes at very low cost.
- The message passing models avoids the data races and as a consequence the programmer is freed from using the locks.

The disadvantages with distributed memory based parallel programming model.

- Message passing models on contrast encounters deadlocks during the process of communications.
- Development of applications on message passing models is hard and takes more time.
- The developer is responsible for establishing communication between processors.
- Message passing models are less performance oriented and incur high communication overheads.

7.2 The Programming Models

This section discusses some of the other existing programming models aside the two(2) major ones (Shared Memory and Message passing programming models). The models described in this section are a subset of the two models that were discussed in the previous section.

7.2.1 Message Passing Interface (MPI)

The MPI is a library of routines with the bindings in Fortran, C, and C++ and it is an example of an explicitly parallel API that implements the message-passing model via library function calls. The set of processes with separate address spaces coordinate the computation by

explicitly sending and receiving messages. Each process has a separate address space, its own program counter, and its own call stack. However, high- level constructs such as synchronization, communication, and mapping data to processes are left to a programmer to implement. MPI supports point-to-point communication between any two processes. It also enables the collective communication operations where a group of processes perform global/collective operations, such as gather, scatter, reduce, and scan. In a heterogeneous environment, in order to optimize the performance, an MPI implementation may map processes to processors in a particular way. Similarly, an MPI implementation may optimize the way processes communicate during a global operation.

7.2.2 Open Multi-Processing (OpenMP)

On the other side, OpenMP is an example of mainly implicit parallel API intended for shared-memory multiprocessors. It exploits parallelism through compiler directives and the library function calls. Unlike MPI, where all threads are spawned at the beginning of the execution and are active until the program terminates, in OpenMP, a single master thread starts execution, and additional threads are active only during the execution of a parallel region. To reduce the overheads, these threads are spawned when the program enters a parallel region for the first time, and they are blocked while the program is executing a nonparallel region.

Sections work-sharing construct breaks work into multiple distinct sections, such that each section is entirely executed by a single thread. It is an example of task parallelism paradigm.

7.2.3 MapReduce

One of the most widely used parallel programming models today is MapReduce. MapReduce is easy both to learn and use, and is especially useful in analyzing large datasets. While it is not suitable for several classes of scientific computing operations that are better served by message-passing interface or OpenMP, such as numerical linear algebra or finite element and finite difference computations, MapReduce's utility in workflows frequently called “big data” has made it a mainstay in high performance computing. MapReduce programming model and the Hadoop open-source framework supports it.

7.2.4 Open Computing Language (OpenCL)

OpenCL has some advantages over other parallel programming models. First of all, it is the only one of the “open” standards for which there, actually, are implementations by all major vendors—unlike for OpenMP or OpenACC. The level of vendor support, however, is a different story. OpenCL is a library that can be used with any C/C++ compiler, which makes it independent of additional tools. The kernels are written separately in a C-like language and compiled at runtime for the present hardware. The kernel compiler comes with the OpenCL implementation provided by the hardware vendor. A kernel written in OpenCL will run everywhere, including conventional CPUs, Intel Xeon Phi coprocessors, GPGPUs, some FPGAs, and even mobile devices.

OpenCL programs are divided into host and kernel code. Only the latter is executed on the compute device. In the host program, kernels and memory movements are queued into command queues associated with a device. The kernel language provides features like vector types and additional memory qualifiers. A computation must be mapped to work- groups of

work-items that can be executed in parallel on the compute units (CUs) and processing elements (PEs) of a compute device. A work-item is a single instance of a kernel function. For each kernel-call, an NDRange (*n*-dimensional range) specifies the dimension, number, and shape of the work-groups. Global synchronization during the execution of a kernel is unavailable. Work-items inside a work-group can be synchronized. OpenCL provides a complex memory model with a relaxed consistency.

7.2.5 The CUDA (Compute Unified Device Architecture) programming model

The CUDA programming model is a parallel programming model that provides an abstract view of how processes can be run on underlying GPU architectures. The evolution of GPU architecture and the CUDA programming language have been quite parallel and interdependent. Although the CUDA programming model has stabilized over time, the architecture is still evolving in its capabilities and functionality. GPU architecture has also grown in terms of the number of transistors and number of computing units over years, while still supporting the CUDA programming model.

Until 2000 GPU architectures supported fixed pipeline functionality tightly coupled with graphics pipeline. Separate silicon real estate was dedicated to each state of the pipeline. Around 2001 programmability for 2D operations (pixel shaders) and 3D operations (vertex shaders) were introduced. Then from approximately 2006 through 2008 all these operations were combined to be executed by a shared and common computational unit using a much higher-level programmable feature. This programmability was introduced as the CUDA programming model. Since then the CUDA programming model has been used to implement many algorithms and applications other than graphics, and this explosion of use and permeability of CUDA with hitherto unknown applications has catapulted the GPU's near ubiquitous use in many domains of science and technology. Since then all the GPUs designed are CUDA-capable. It should be noted that before CUDA was released, there were attempts to create high-level languages and template libraries. But such efforts tapered down with the introduction of CUDA, and more effort was spent on refining CUDA and building libraries using its constructs.

Self-Assessment Questions

1. Define the term parallel programming.
2. State and discuss extensively, the two widely known parallel programming models.
3. Define the following
    a. Message Passing Interface (MPI)
    b. Open Multi-Processing (OpenMP)
    c. MapReduce
    d. Compute Unified Device Architecture (CUDA) programming model