Day 1 - 16th May 2025

111594594

Pentakota Sai Kumar

What is SDLC?

Software development cycle (SDLC) is a step by step process the software developers team follow to design,develop, test and deploy the software application

Stages in SDLC are

1.PLANNING AND REQUIREMENT

2.DESIGN

3.DEVELOPMENT

4.TESTING

5.DEPLOY

6.MAINTENANCE

Why should we use SDLC?

Using SDLC helps you to build the right software and deliver in the right way and deliver it successfully with good quality and efficiency by providing a structured approach and also by reducing the risk by improving the customer satisfaction.

What are the stages of SDLC?

1.PLANNING AND REQUIREMENT- In this stage we have to firstly know the project objective and scope. This involves understanding the business needs and documenting all the functional and non-functional requirements for the software

2.DESIGN- create a blueprint for how the software will be build based on the requirements

3.DEVELOPMENT-To translate the design specifications into actual working code

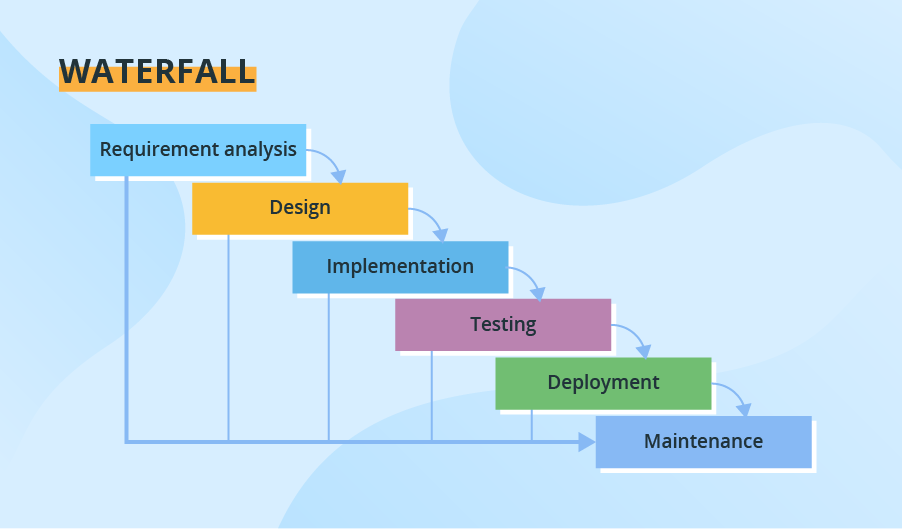
4.TESTING- To identity and fix any defects or issues in the software before it is deployed

5.DEPLOY - To make the software available for the users in the operational environment ensure the software continues to function correctly and meets the user needs.

6.MAINTENANCE- ensure the software continues to function correctly and meets the user's needs.

MODELS OF SDLC;

### **Waterfall model:**

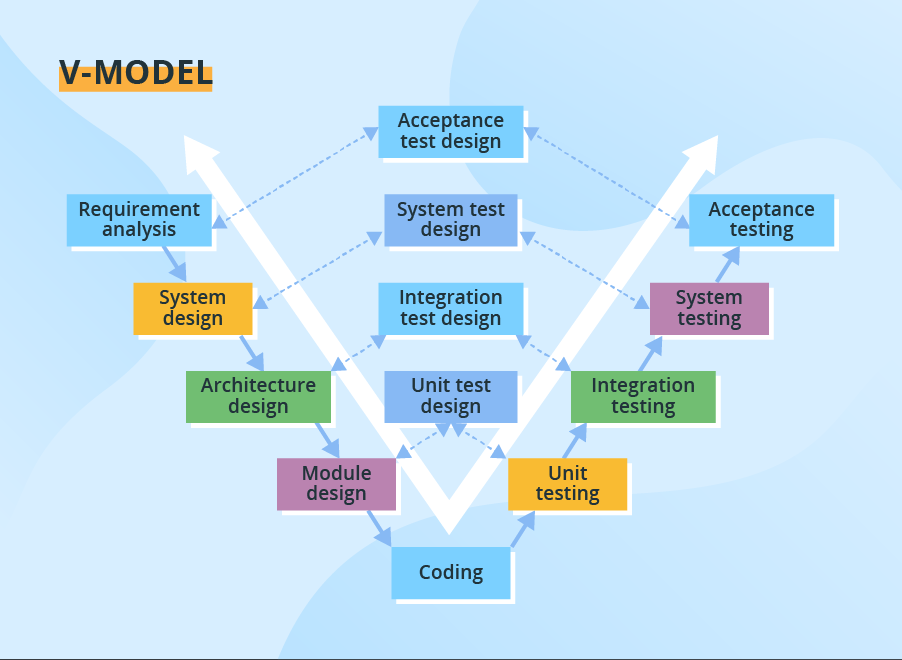


Through all development stages (analysis, design, coding, testing, deployment), the process moves in a step by step mode. Each stage has concrete deliverables and is strictly documented. The next stage cannot start before the previous one is fully completed. Thus, for example, software requirements cannot be re-evaluated further in development. There is also no ability to see and try software until the last development stage is finished, which results in high project risks and unpredictable project results. Testing is often rushed, and errors are costly to fix.

**APPLICATIONS**;

* Simple small or mid-sized projects with clearly defined and unchanging requirements (small company website development).
* Projects with the need for stricter control, predictable budget and timelines (e.g., governmental projects).
* Projects that must adhere to multiple rules and regulations (healthcare projects).
* Projects where a well-known technology stack and tools are used.
* **Advantages of Waterfall Model:**
* **Easy to Understand**
* **Individual Processing**
* **Properly Defined**
* **Clear Milestones**
* **Properly Documented**
* **Disadvantages of waterfall model**:
* Difficult to accommodate change request
* Limited flexibility
* Late defect detection

**V-model (Validation and Verification model):**



The V-model is another linear model with each stage having a corresponding testing activity. Such workflow organization implies exceptional quality control, but at the same time, it makes the V-model one of the most expensive and time-consuming models. Moreover, even though mistakes in requirements specifications, code and architecture errors can be detected early, changes during development are still expensive and difficult to implement. As in the Waterfall case, all requirements are gathered at the start and cannot be changed.

**APPLICATIONS**;

* Projects where failures and downtimes are unacceptable (e.g., medical software, aviation fleet management software).

**Advantages:**

This is a highly disciplined model and Phases are completed one at a time.

V-Model is used for small projects where project requirements are clear.

Simple and easy to understand and use.

This model focuses on verification and validation activities early in the life cycle thereby enhancing the probability of building an error-free and good quality product.

**Disadvantages:**

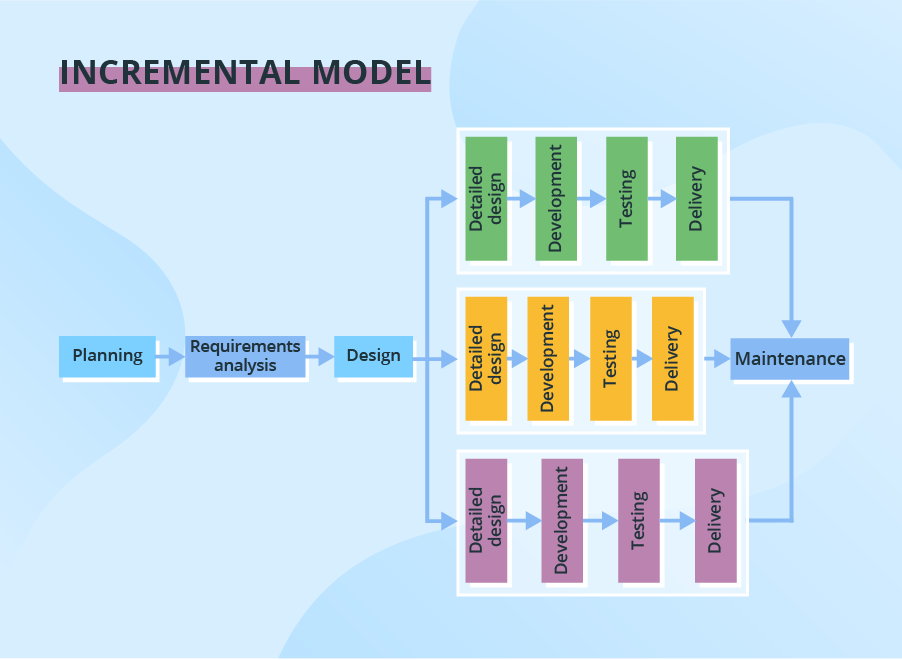
The V-Model is a linear and sequential model, which can make it difficult to adapt to changing requirements or unexpected events.

The V-Model can be time-consuming, as it requires a lot of documentation and testing.

High risk and uncertainty.

It is not good for complex and object-oriented projects.

**Incremental Model:**



The development process based on the **Incremental model** is split into several iterations (“Lego-style” modular software design is required!). New software modules are added in each iteration with no or little change in earlier added modules. The development process can go either sequentially or in parallel. Parallel development adds to the speed of delivery, while many repeated cycles of sequential development can make the project long and costly.

**APPLICATIONS:**

* Large, mission-critical enterprise applications that preferably consist of loosely coupled parts, such as microservices or web services.

**Advantages:**

Early Error Detection and Resolution,Flexibility and Adaptability,Faster Time to Market

**Disadvantages:**

More resources may be required.

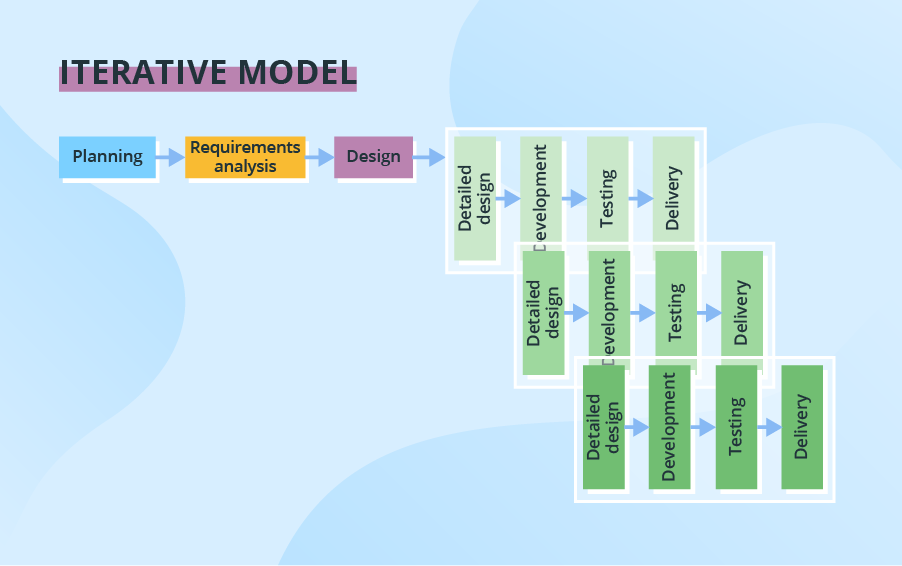
Although cost of change is lesser, but it is not very suitable for changing requirements.

More management attention is required.

It is not suitable for smaller projects.

Highly skilled resources are required for skill analysis.

**Iterative model**

****

In the Iterative model, iterative process starts with a simple implementation of a small set of the software requirements and iteratively enhances the evolving versions until the complete system is implemented and ready to be deployed.

**APPLICATIONS:**

Requirements of the complete system are clearly defined and understood

Major requirements must be defined; however, some functionalities or requested enhancements may evolve with time.

There is a time to the market constraint.

A new technology is being used and is being learnt by the development team while working on the project.

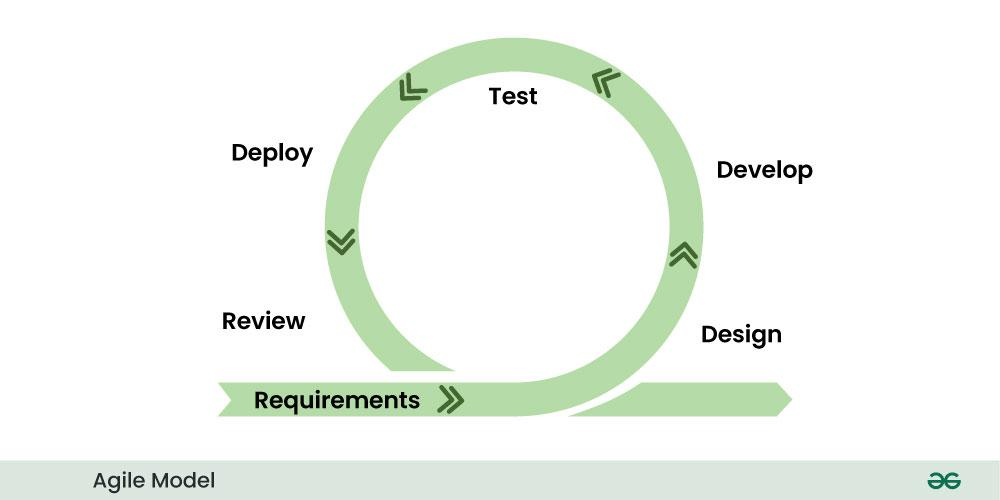
**Advantages:**

Easier to manage risk - High risk part is done first,Risk analysis is better,Better suited for large and mission-critical projects.

**Disadvantages:**

Highly skilled resources are required for risk analysis.Management complexity is more.More resources may be required.

**Agile Model**

****

Agile model is a flexible and iterative approach to software development that priortizes collaboration, adaptability and rapid delivery. It emphasizes continuous feedback and incremental improvements through out the development process.

**APPLICATIONS;**

* Practically any startup initiatives, when end users’ early feedback is required.
* Most mid-sized projects in Custom software development where business requirements cannot be confidently translated to detailed software requirements.
* Large projects that are easy to divide into small functional parts and can be developed incrementally over each iteration.

**Advantages:**

Increased collaboration and communication

Flexibility and adaptability

Improved quality and reliability

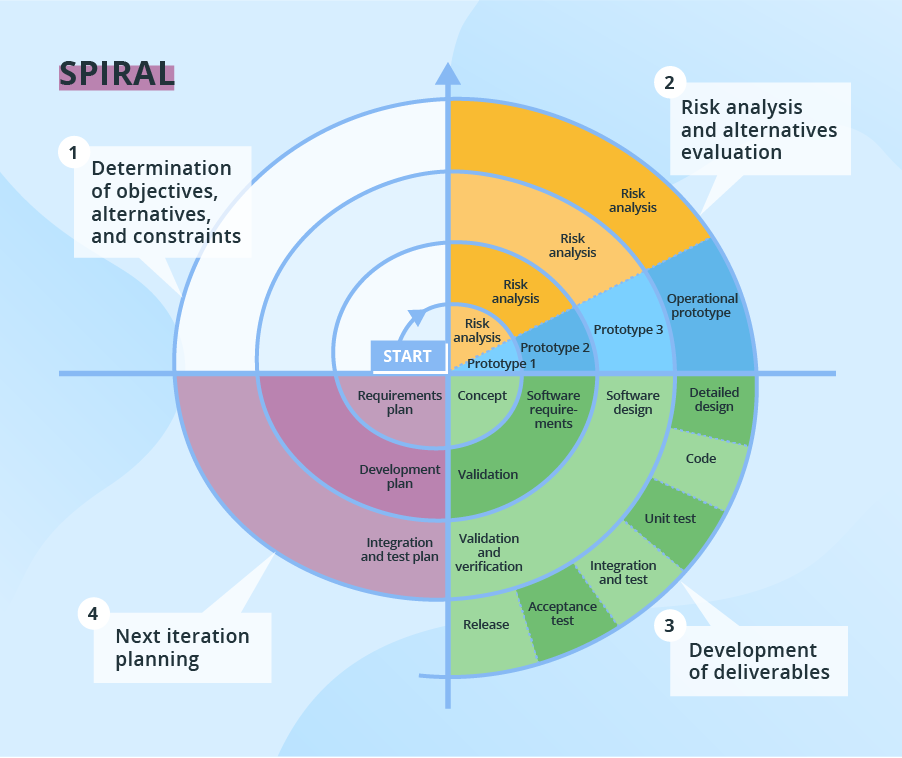
Enhanced customer satisfaction

Reduce risk

**Disadvantages:**

**Unpredictable timeline,Limited documentation**

**Spiral model**



The Spiral model puts focus on thorough risk assessment. Thus, to reap the benefits of the model to the fullest, you’ll need to engage people with a strong background in risk evaluation. A typical Spiral iteration lasts around 6 months and starts with 4 important activities - thorough planning, risk analysis, prototypes creation, and evaluation of the previously delivered part. Repeated spiral cycles seriously extend project timeframes.

**APPLICATIONS:**

* Projects with unclear business needs or too ambitious/innovative requirements.
* Projects that are large and complicated.
* Research and development (R&D) activity or the introduction of a new service or a product.

**Advantages:**

Software is produced early in the software life cycle.

It is suitable for high risk projects, where business needs may be unstable. A highly customized product can be developed using this.

**Disadvantages:**

It is not suitable for small projects as it is expensive.

Difficulty in time management. As the number of phases is unknown at the start of the project, so time estimation is very difficult.

**SCRUM:** scrum is the most popular agile model in which the iterations done in the name of sprints are usually 2 - 3 weeks long and they are preceded with thorough planning and previous sprint assessment no changes are allowed after the sprint activities have been defined.

**SPRINT:**a sprint is a fixed-length period of time, typically 1-4 weeks, during which a team works to complete a set of work and deliver a potentially releasable product increment.No changes are allowed after the sprint activities have been defined.

**DO’S**

There should be some preferred goals

Maintain communication

Continuously inspect the progress

**DONT’S**

Do not introduce new tasks in the middle of the sprint

Do not miss the daily stand up calls or daily scrum

Don't hold long problem solving discussion in the daily scrum

**User Stories:** small independent units of work that represent a user’s need or goals.

**BACKLOG** Prioritized list of all the work iteams ex features, fixes required to develop a product.

**Product Backlog:**

This is the overarching list of all potential work items, prioritized by the Product Owner, and represents the roadmap for the entire product. It's a living document that evolves as the product and customer needs change.

**Sprint Backlog:**

This is a subset of the Product Backlog, containing the specific items that the team commits to completing during a particular sprint . It's a more detailed, focused list of tasks that the team will work on during the sprint.

**Increment:** this is a working version of the product created at the end of each sprint.

**Protocol:** a protocol in a computer networking is a set of rules and procedure that how the data is transmitted and received over a network. It ensures that devices can communicate effectively with each other regardless of their hardware and software

Ex. HTTP,SMTP AND FTP

**PORT:** A Port is a virtual point where network connections start and end its a 16-bit number ranging from 0-65535. It identifies a specific process or service running on a network device

**Networking types:**

LOCAL AREA NETWORKING(LAN)

METROPOLITAN AREA NETWORK(MAN)

WIDE AREA NETWORK(WAN)

Virtual private network (vpn)

**TYPES OF SERVERS:**

**WEB SERVER:** Web servers are designed to run websites and apps through client programs (web browsers) such as Internet Explorer, Chrome, Firefox, Opera, or Safari. They are responsible for storing, processing, and delivering web content to users. They support protocols such as HTTP, FTP, and SMTP that are key to information

**MAIL SERVER:** A mail server facilitates email storage and management for clients. It uses different protocols for sending and receiving emails. For example, the Simple Mail Transfer Protocol (SMTP) is used to send an email. The server relies on the Post Office Protocol (POP3)l to store and receive an email.

**APPLICATION SERVER:**A mail server facilitates email storage and management for clients. It uses different protocols for sending and receiving emails. For example, the Simple Mail Transfer Protocol (SMTP) is used to send an email. The server relies on the Post Office Protocol (POP3)l to store and receive an email.

**DATA BASE SERVER:**Database servers offer database services to client computers. Users can access, modify, store, and retrieve data from a database by executing a query; for example, an SQL query. Database servers are responsible for handling the security and recovery of database management systems (DBMS).

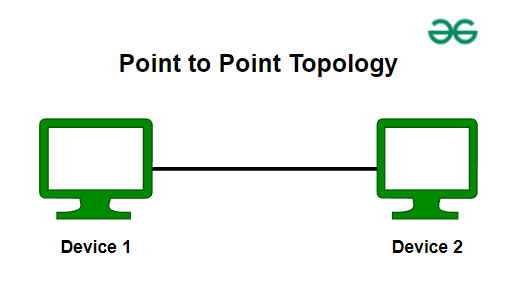
**DNS SERVER:**DNS servers are domain name servers. These computers resolve server names that reside in a network. DNS servers are an integral part of the internet as they translate user-understandable URLs

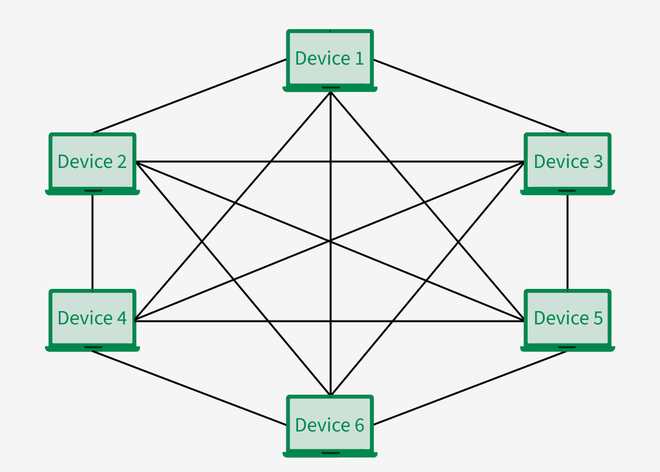
**DOMAIN NAME SYSTEM (DNS):**The Domain Name System (DNS) translates human-readable domain names (e.g., www.google.com) into machine-readable IP addresses (e.g., 142.250.190.14), enabling internet communication. It enables computers to locate and communicate with each other on the internet.

**Network Topologies:** Network topology is the way devices are connected in a network. It defines how these components are connected and how data transfer between the network. Understanding the different types of network topologies can help in choosing the right design for a specific network.

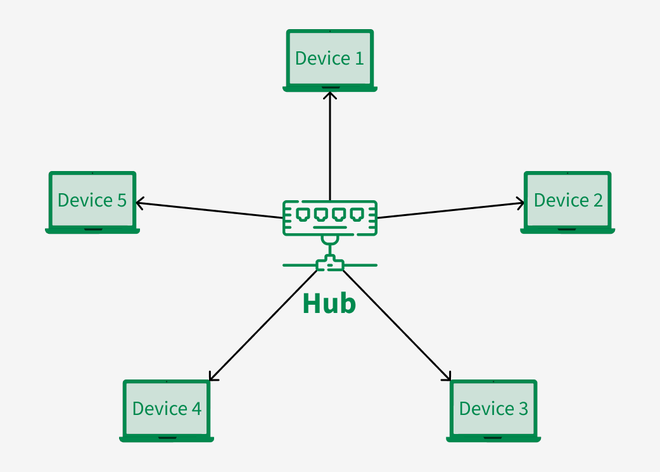
**Types of network topologies:**

**Point to Point Topology:**It is a type of topology that works on the functionality of the sender and receiver. It is the simplest communication between two nodes, in which one is the sender and the other one is the receiver. Point-to-Point provides high bandwidth.

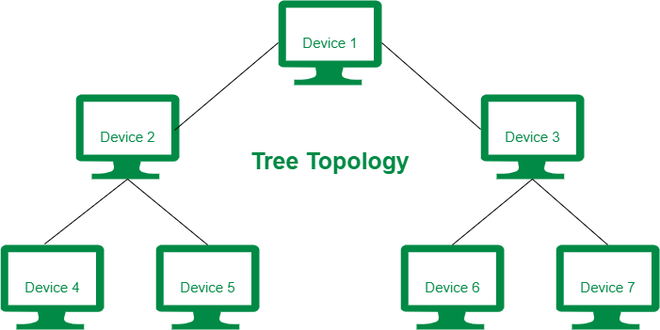
****

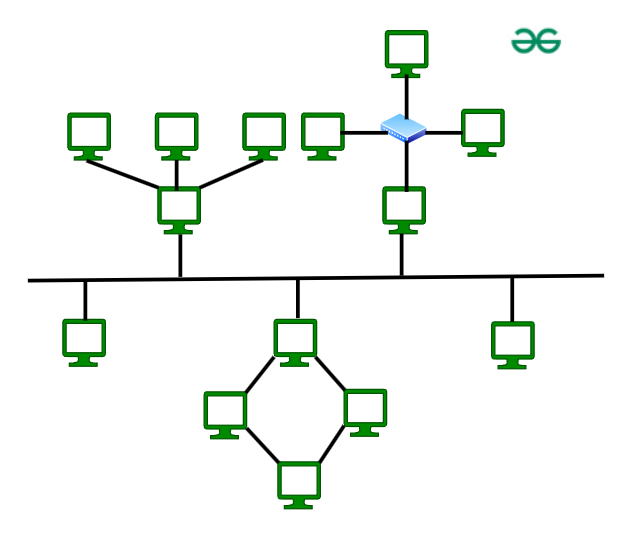
**Mesh Topology:** In a mesh topology, every device is connected to another device via a particular channel. Every device is connected to another via dedicated channels.****

**Star Topology:** In Star Topology all the devices are connected to a single hub through a cable. This hub is the central node and all other nodes are connected to the central node. The hub can be passive in nature that is not an intelligent hub such as broadcasting devices, at the same time the hub can be intelligent known as an active hub.

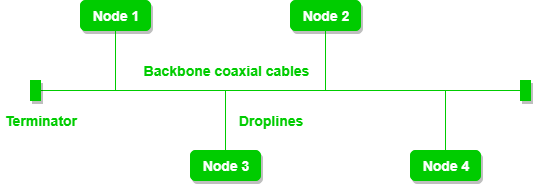
****

**Tree Topology:** Tree topology the various secondary hubs are connected to the central hub which contains the repeater. This data flows from top to bottom from the central hub to the secondary and then to the devices or from bottom to top devices to the secondary hub and then to the central hub. It is a multi-point connection and a non-robust topology because if the backbone fails the topology crashes.

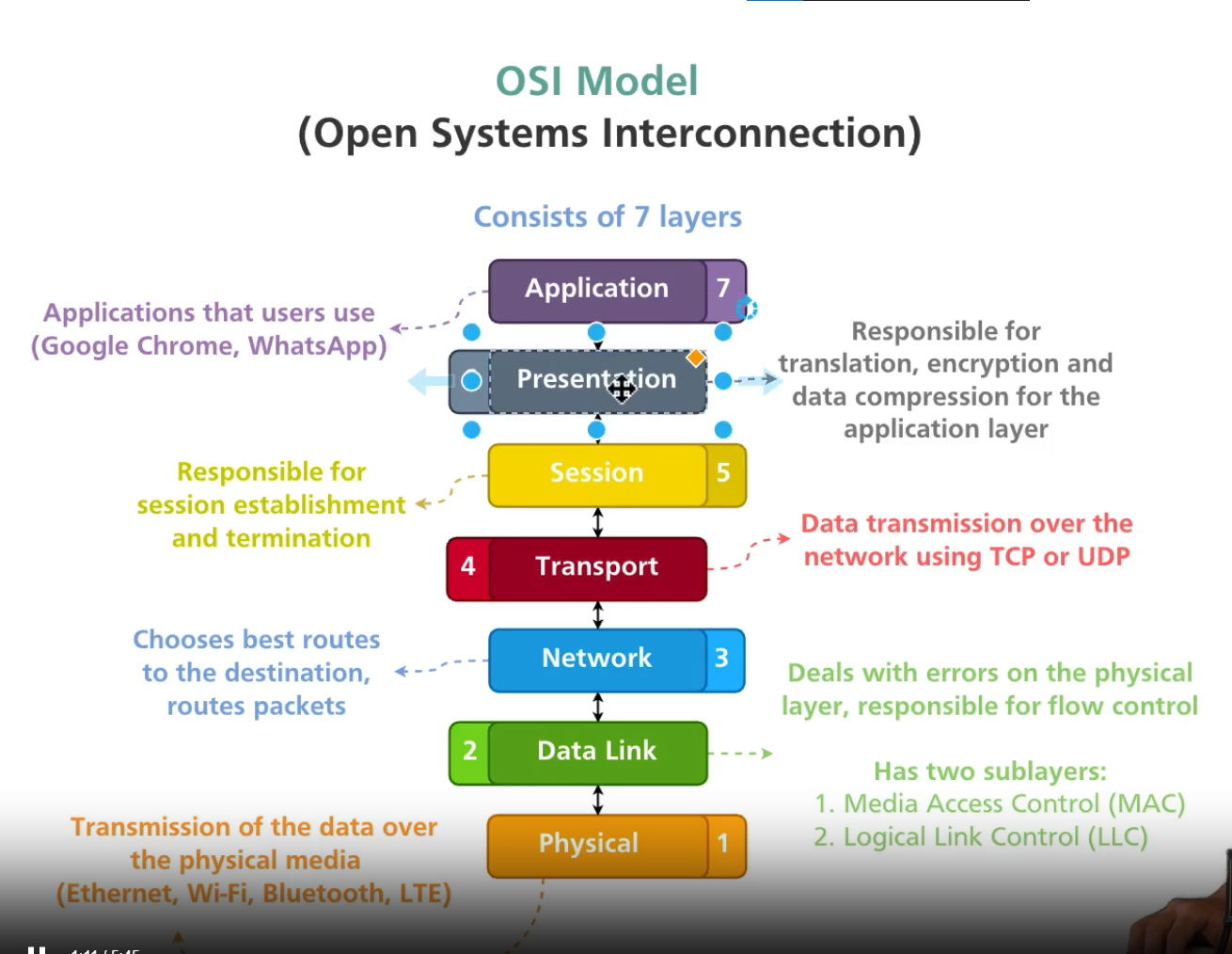


**Hybrid Topology:** Hybrid Topology is the combination of all the various types of topologies we have studied above. Hybrid Topology is used when the nodes are free to take any form. It means these can be individuals such as Ring or Star topology or can be a combination of various types of topologies seen above.

**Bus Topology:**Bus Topology is a network type in which every computer and network device is connected to a single cable. It is bi-directional. It is a multi-point connection and a non-robust topology because if the backbone fails the topology crashes.



**OSI MODEL:** OPEN SYSTEM INTERCONNECTION is a framework that describes how data communication occurs through out the network

****

**Physical Layer:** The physical layer is responsible for the transmission and reception of unstructured raw data between a device

**Data Link Layer:** The data link layer provides node-to-node data transfer—a link between two directly connected nodes. It detects and possibly corrects errors that may occur in the physical layer

**Network Layer:** The network layer provides the functional and procedural means of transferring packets from one node to another connected in different networks

**Transport Layer:**  This layer provides reliable or unreliable end-to-end delivery of data between applications. It handles segmentation of data, flow control, and error control using TCP and UDP

**Session Layer:** This layer manages and controls the connections (sessions) between applications

**Presentation Layer:** This layer is reasonable for data formatting encryption and comparison .

**Application Layer:** This is the layer closest to the end-user. It provides the interface between network applications and the underlying network services