**UNIT-1**

**Question Bank**

**1. Which one of the following is not an Evolutionary Process Model?**

a) WINWIN Spiral Model

b) Incremental Model

c) Concurrent Development Model

d) Spiral Model

e) All are Evolutionary Software Models

**2. What is the major advantage of using Incremental Model?**

a) Customer can respond to each increment

b) Easier to test and debug

c) It is used when there is a need to get a product to the market early

d) Both b & c;

**3. The spiral model was originally proposed by**

a) IBM

b) Barry Boehm

c) Pressman

d) Royce

**4. How is Incremental Model different from Spiral Model?**

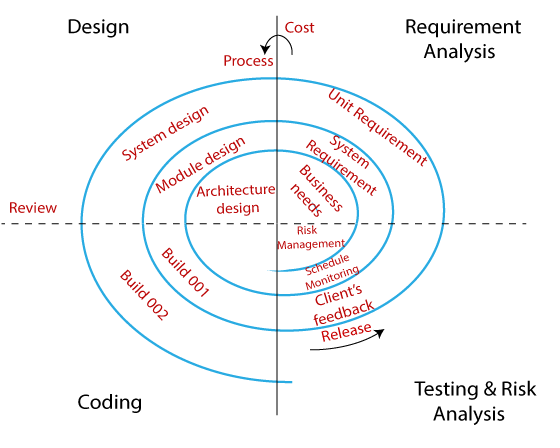
a) Progress can be measured for Incremental Model.

b) Changing requirements can be accommodated in Incremental Model.

c) Users can see the system early in Incremental Model.

**5. Give scenario base explanation of Spiral model and explain its architecture with advantages and disadvantages.**

* In this model, we create the application **module by module** and handed over to the customer so that they can start using the application at a very early stage.
* we prepare this model only when the **module is dependent** on each other.
* In this model, we develop the application in the stages because sometimes the **client gives the requirements in between the process**.
* The different **phases** of the spiral model are as follows:
* Requirement analysis
* Design
* Coding
* Testing and risk analysis
* Spiral Model



**Requirement Analysis**

* The spiral model process starts with collecting business needs.
* In this, the following spirals will include the…
* documentation of system requirements
* unit requirements
* the subsystem needs.

**Design**

The second stage of the spiral model is designed,

* logical design
* architectural design
* flow charts
* decision tree

**Coding**

After the compilation of the design stage, we will move to our next step, which is the coding stage.

* In this, we will **develop** the product based on the **client's requirement** and getting the **client's feedback** as well.
* This stage refers to the construction of the **real application in every cycle.**
* Gives excellent clarity of the requirements, and the design details of an application.
* After that, these builds are **transferred to the client** for their responses.

**Testing and Risk Analysis**

Once the development is completed successfully, we will **test the build** at the end of the first cycle.

* **Analyze the ris**k of the software on the different aspects such as **managing risks, detecting, and observing the technical feasibility**. And after that, the client will test the application and give feedback.

**Example:**

The best-suited example for the spiral model is **MS-Excel**

Because MS-Excel sheet having several cells, which are the components of an excel sheet.

Since we have to create the cells first (module P), then we can perform operation on the cells like split cells into half (module Q), merge cells into two, and then we can draw graphs on the excel-sheet (module R).

**Advantages of Spiral Model:**

* **Risk handling**
* it is best development model to follow due to the risk analysis and risk handling at every phase.
* **Flexibility in requirements**.
* We can easily change requirements at later phases and can be incorporated accurately. Also, additional Functionality can be added at a later date.
* It is good for **large and complex projects**.
* It is good for **customer satisfaction.**
* We can involve customers in the development of products at early phase of the software development. Also, software is produced early in the software life cycle.
* Strong **approval and documentation control.**
* It is suitable for **high-risk projects**, where business needs may be unstable. A highly customized product can be developed using this.

**Disadvantages of Spiral Model:**

* It is not suitable for **small projects as it is expensive.**
* It is much more **complex** than other SDLC models. Process is complex.
* Too much dependable on Risk Analysis and requires **highly specific expertise.**
* Difficulty in **time management**. As the number of phases is unknown at the start of the project, so time estimation is very difficult.
* It is not suitable for **low-risk projects.**
* May be hard to define **objective, verifiable milestones**.
* Large numbers of intermediate stages require **excessive documentation.**

**6. Why software engineering is a layered Technology?**

Because while you build any software you need to **overcome the bearer between hardware to softwar**e in such way that software can work **smoothly** and **efficiently**.

Now if we think we **don't have any layered techno**logy to develop the software then this makes quite **difficult** to develop any software **within short period of time.**

Suppose, we have to work with hardware level to software level as per the software requirements. And it is difficult to work with all the seven layers of the OSI model at same time and makes it difficult to work with which protocol.

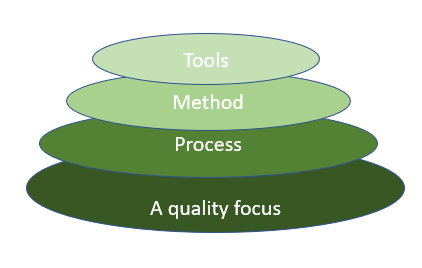
This leads to many complexities while developing any software.

**Advantages**:

* Layered (stepwise) development
* Easy to configure system
* Error detection can be easy
* Maintenance is easy

Q. **Explained Layered Technology.**

Software engineering is a fully layered technology, to develop software we need to go from **one layer to another**. All the layers are **connected** and each layer demands the fulfillment of the **previous layer.**



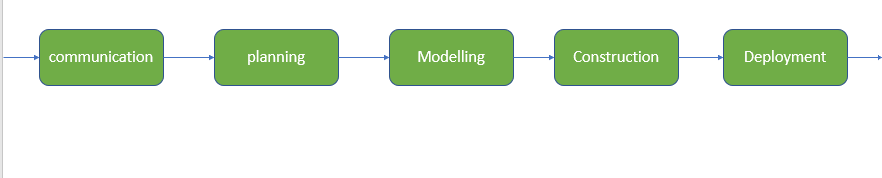
Layered technology is divided into **four parts**:

1. **A quality focus:**

* It defines the **continuous** **process improvement** principles of software.
* It provides **integrity** that means providing **security** to the software so that data can be accessed by only an **authorized person**, no outsider can access the data. It also focuses on **maintainability** and **usability**.

1. **Process:**

* It is the foundation or **base layer** of software engineering.
* It is key that **binds all the layers together** which enables the development of software **before the deadline** or on time.
* Process **defines a framework** that must be established for the effective delivery of software engineering technology.
* The software process covers all the **activities, actions**, and **tasks** required to be carried out for software development.



**Process activities** are listed below:-

**Communication:**

* It is the first and foremost thing for the development of software.
* Communication is necessary to **know the actual demand** of the client.

**Planning:**

* It basically means **drawing a map** for reduced the complication of development.

**Modeling:**

* A model is created **according to the client** for better understanding.

**Construction:**

* It includes the **coding and testing** of the problem.

**Deployment:**

* It includes the **delivery of software** to the client for evaluation and **feedback**.

1. **Method:**

* During the process of software development, the **answers to all “how-to-do”** questions are given by method.
* It has the information of all the tasks which includes **communication, requirement analysis, design modeling, program construction, testing, and support**.

1. **Tools:**

* Software engineering tools **provide a self-operating system** for processes and methods.
* Tools are **integrated** which means information created by **one tool can be used by another.**

**7. Define different types of software Myths.**

Different Types of Software Myths

* Management Myths
* Customer Myths
* Practitioner’s Myth

**Management Software Myths**

Software Myths:

Most, experienced experts have seen myths or superstitions or misleading attitudes which creates major problems for management and technical people. The types of software-related myths are listed below.

**Management Myths:**

* **Myth 1:**We have all the standards and procedures available for software development.
* **Fact:**

Software experts do not know all the requirements for the software development. And all existing processes are incomplete as new software development is based on new and different problem.

* **Myth 2:**The addition of the latest hardware programs will improve the software development.
* **Fact:**

The role of the latest hardware is not very high on standard software development; instead (CASE) Engineering tools help the computer, they are more important than hardware to produce quality and productivity. Hence, the hardware resources are misused.

* **Myth 3:**With the addition of more people and program planners to Software development can help meet project deadlines (If lagging behind).
* **Fact:**If software is late, adding more people will merely make the problem worse. This is because the people already working on the project now need to spend time educating the newcomers, and are thus taken away from their work. The newcomers are also far less productive than the existing software engineers, and so the work put into training them to work on the software does not immediately meet with an appropriate reduction in work.

**Customer Myths:**   
The customer can be the direct users of the software, the technical team, marketing / sales department, or other company. Customer has myths leading to false expectations (customer) & that’s why you create dissatisfaction with the developer.

* **Myth 1:**  
  A general statement of intent is enough to start writing plans (software development) and details of objectives can be done over time.
* **Fact:**  
  Official and detailed description of the database function, ethical performance, communication, structural issues and the verification process are important.
* **Myth 2:**  
  Software requirements continually change, but change can be easily accommodated because software is flexible
* **Fact:**It is true that software requirements change, but the impact of change varies with the time at which it is introduced. When requirements changes are requested early (before design or code has been started), the cost impact is relatively small. However, as time passes, the cost impact grows rapidly—resources have been committed, a design framework has been established, and change can cause upheaval that requires additional resources and major design modification.

**Practitioner’s Myths:**

* **Myths 1:**They believe that their work has been completed with the writing of the plan.
* **Fact:**  
  It is true that every 60-80% effort goes into the maintenance phase (as of the latter software release). Efforts are required, where the product is available first delivered to customers.
* **Myths 2:**There is no other way to achieve system quality, until it is “running”.
* **Fact:**Systematic review of project technology is the quality of effective software verification method. These updates are quality filters and more accessible than test.
* **Myth 3:**  
  An operating system is the only product that can be successfully exported project.
* **Fact:**  
  A working system is not enough, the right document brochures and booklets are also required to provide guidance & software support.
* **Myth 4:**Engineering software will enable us to build powerful and unnecessary document & always delay us.
* **Fact:**Software engineering is not about creating documents. It is about creating a quality product. Better quality leads to reduced rework. And reduced rework results in faster delivery times.

**8. In what situations will you use waterfall model? Justify your answer. Name the risk-based software development process model? What are its advantages and disadvantages?**

**Q. When to use the waterfall model?**

* Requirements are clear and fixed that may not change.
* There are no ambiguous requirements (no confusion).
* Product definition is stable.
* Technology is understood.
* The project is short and cast is low.
* Risk is zero or minimum.

**Q.** **Name the risk-based software development process model?**

* Spiral Model

**9. What do you mean by software process models? Why we need it? Distinguish between software product and software process. Name four process models that are used to develop large software systems.**

**Software Processes** is a set of activities for specifying, designing, implementing and testing software systems.

**Software process model** is an abstract representation of a process that presents a description of a process from some particular perspective.

**Specification** – defining what the system should do;

**Design and implementation**: defining the organization of the system and implementing the system;

**Validation**: checking that it does what the customer wants.

**Evolution**: changing the system in response to changing customer needs.

**Software Product**

* It is the final production phase in the project.
* It focuses on the final result.
* It follows the firm guidelines.
* It tends to be a short-term aspect.
* The goal of a process is to complete the work successfully.

**Software Process**

* It refers to a set of sequence of steps that should be followed with the goal of creating a project.
* It focuses on completing every step involved in the project that is being developed.
* It follows the guidelines consistently.
* It tends to be a long-term aspect.
* The goal of a process is to improve the quality of the project.

**Software process model:**

* Waterfall model
* V model
* Incremental model
* RAD model
* Agile model
* Iterative model
* Spiral model
* Prototype model

**Software development methodologies**

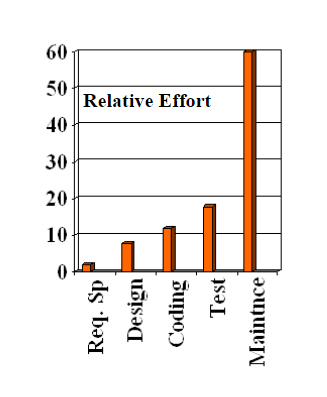
* Agile development methodology.
* DevOps deployment methodology.
* Waterfall development method.
* Rapid application development.

**10. Why does the waterfall model sometimes fail?**

* Waterfall uses the theory that what you want at the beginning is what you get at the end so there is little, if any room, for significant changes in direction.
* Being an inflexible model, which does not provide for feedback, it is difficult to highlight new requirements and thus change course

11. **What are the major phases of the entire life of the software? specify the percentage of efforts required on each phase. Which phase requires the maximum efforts? Which phase is/are more creative?**

* Requirement Gathering and Analysis.
* Design.
* Implementation or Coding.
* Testing.
* Deployment.
* Maintenance.
* **Maintenance** phase consumes maximum **effort**.
* The most **creative** and challenging phase of system life cycle is **Design**



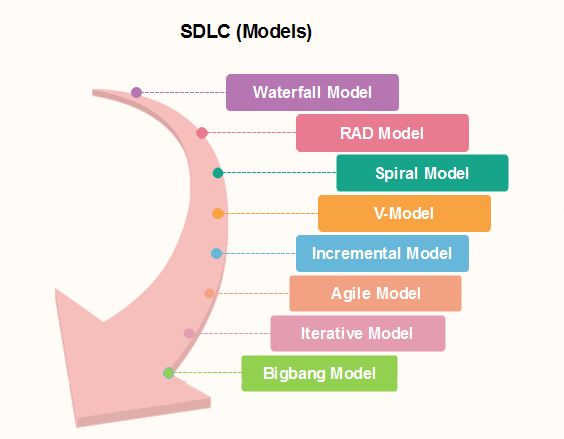
**12. If formal methods can demonstrate software correctness, why is it they are not widely used?**

Business managers have faith that formal methods can enhance the software quality, but formal methods are not widely used **because these methods are considered costly and unfeasible.**

* The most considerable doubt in formal methods, mostly from a perspective of management, is that these methods are expensive.
* because implementing successful formal methods in an organization also need to purchase the tools for supporting these methods, training of engineers and designers, and effort and time to incorporate formal methods in the existing software development process.

**13. Compare the relative advantage of using the iterative waterfall model and the spiral model of software development. Explain with the help of examples, the type of problems for which you would adopt the waterfall model of software development, and the type of problems for which you would adapt the spiral model.**

**14. What do you mean by software lifecycle models? is it compulsory to follow it while developing a project? Explain in brief.**



**15. What formal techniques are available for assessing the software process?**

**Specification** – defining what the system should do;

**Design and implementation**: defining the organization of the system and implementing the system;

**Validation**: checking that it does what the customer wants.

**Evolution**: changing the system in response to changing customer needs.

**16. Define software and software engineering. What are the characteristics of software?**

**Software** is a collection of codes, documents, and triggers that does a specific job and fills a specific requirement.

**Software Engineering** is the development of products using best practices, principles, and methods.

**Characteristics of software**

* Functionality.
* Usability (User-friendly)
* Efficiency.
* Flexibility.
* Reliability.
* Maintainability.
* Portability.
* Integrity.

**17. Show how the failure curve of software differs from that of hardware. Software doesn’t wear out but it deteriorates due to change. Justify. How do software myths affect a software project?**

**Hardware** failure is random. Hardware does have increasing failure at the last stage.

**Software** failure is systematic. Software does not have an increasing failure rate.

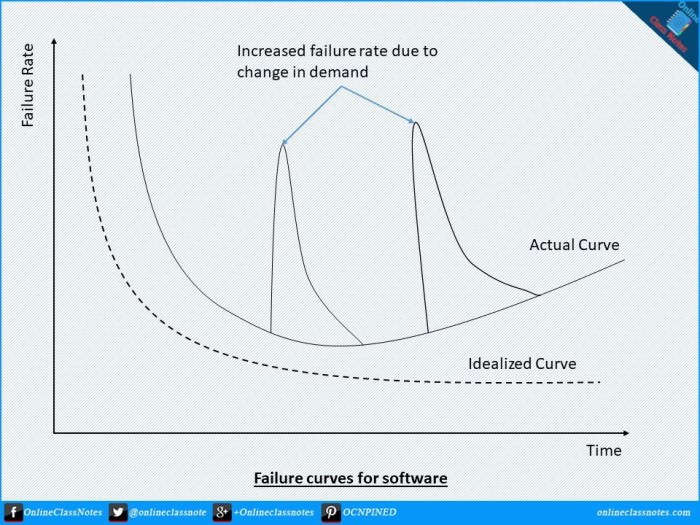
Hardware components suffer from the growing effects of many other environmental factors. Stated simply, the hardware begins to wear out.

Software is not susceptible to the environmental maladies that cause hardware to wear out.

In the following given figure, we can see how a hardware, with time, wear out. This is often called the “bathtub curve”. It indicates everything just discussed earlier. At the beginning there can be faults, which then gets corrected and reaches the steady state. And then after certain period of service, the hardware starts to show errors again and at one time it wear out.



In the following given figure, the idealized curve for software has been shown. Also, the change to failure rate due to users demand and the rise of the minimum failure rate has been shown on the actual curve.



**Q.** **How do software myths affect a software project?**

A Software process is a Coherent set of activities for designing, specifying, implementing and testing software systems.

A software myth propagates misinformation and confusion. Software myths had a number of various attributes.

For example, they appeared to be reasonable statements of fact they had an intuitive feel and they were often promulgated by experienced practitioners who know the score.

**18. If you are a lead developer of a software company and you are asked to submit a project/product within a stipulated time-frame with no cost barriers, which model will you select?**

a) Waterfall

b) Incremental

c) Spiral

d) RAD

**Explanation:**

* RAD model is inapplicable to develop cheaper products/software/projects as the cost of modeling, hiring highly skilled developers/designers and automated code generation is very high. But here the cost is not an issue, so one can select this model as it reduces development time.

**19. Explain which process model is most suitable for the following definition and justify it:**

a) A compiler for new language

b) Event management system

c) Chess

**20. Draw neat sketch of spiral model and justify why it is considered as meta model.**

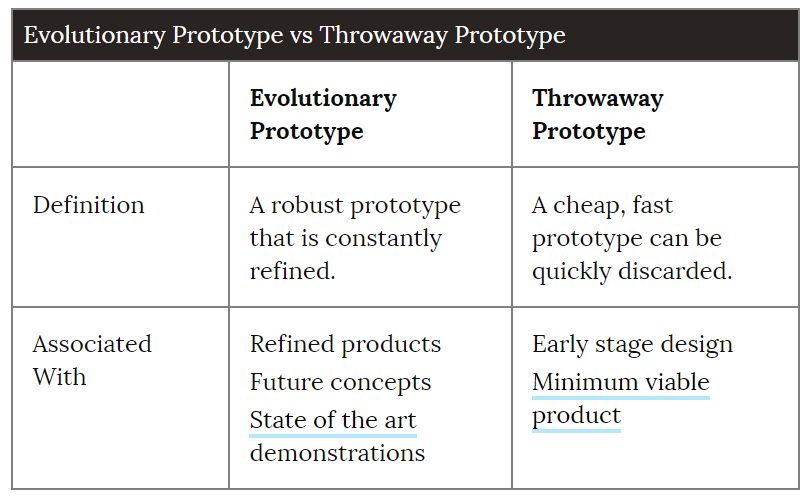
* Because it subsumes all the other SDLC models.
* For example, a single loop spiral actually represents the Iterative Waterfall Model.

**Which model is called as meta process model?**

A process meta-model is at the meta-type level with respect to a process." There exist standards for several domains: Software engineering.

Software Process Engineering Metamodel (SPEM) which is defined as a profile (UML) by the Object Management Group.

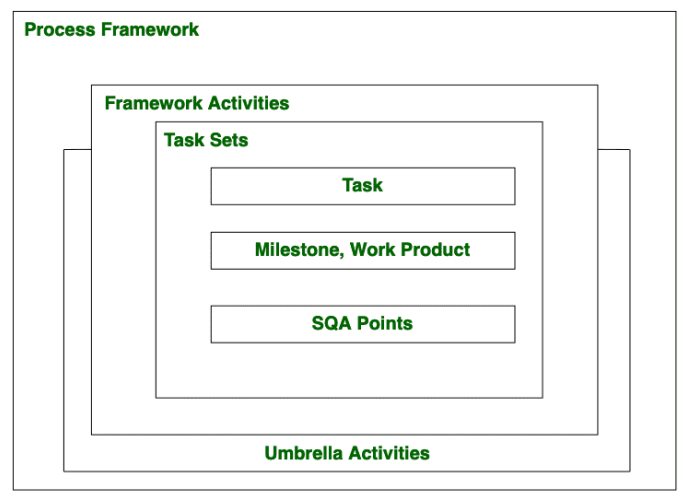
**21. Differentiate throw-away v/s evolutionary model.**

****

**22. Explain with a neat sketch “A software process framework”.**

**Software Process Framework**

* Framework is a Standard way to build and deploy applications.
* Software Process Framework is a foundation of complete software engineering process.
* Software process framework includes all set of umbrella activities.
* It also includes number of framework activities that are applicable to all software
* projects.



A generic process framework encompasses five activities which are given below one by one:

**Communication:**

* In this activity, heavy communication with customers and other stakeholders, requirement gathering is done.

**Planning:**

* In this activity, we discuss the technical related tasks, work schedule, risks,

required resources etc.

**Modeling:**

* Modelling is about building representations of things in the ‘real world’.
* In modelling activity, a product’s model is created in order to better

understanding and requirements.

**Construction:**

* In software engineering, construction is the application of set of procedures

that are needed to assemble the product.

* In this activity, we generate the code and test the product in order to make better product.

**Deployment:**

* In this activity, a complete or non-complete products or software are

represented to the customers to evaluate and give feedback.

* On the basis of their feedback we modify the products for supply better product.

**Umbrella activities include:**

* + Risk management
  + Software quality assurance (SQA)
  + Software configuration management (SCM)
  + Measurement
  + Formal technical reviews (FTR)

**23. How does a framework activity change as the nature of the project changes?**

**24. What is a process pattern?**

**Uses of the process pattern:**

* At any level of abstraction, patterns can be defined.
* They can be used to describe a problem and solution associated with framework activity in some situations.
* While in other situations patterns can be used to describe a problem and solution associated with a complete process model.

**It is of three types:**

**Stage pattern**:

* Problems associated with a framework activity for process are described by stage pattern.
* Establishing Communication might be an example of a staged pattern.
* This pattern would incorporate task pattern Requirements Gathering and others.

**Task-pattern:**

* Problems associated with a software engineering action or work task and relevant to successful software engineering practice
* (e.g., Requirements Gathering is a task pattern) are defined by task-pattern.

**Phase pattern**:

* Even when the overall flow of activities is iterative in nature, it defines sequence of framework activities that occurs within process.
* Spiral Model or Prototyping might be an example of a phase pattern.

**Examples:**

* Customer communication (a process activity).
* Analysis (an action).
* Requirements gathering (a process task).