SUBJECT CODE : CCS356

Strictly as per Revised Syllabus of

ANNA UNIVERSITY

Choice Based Credit System (CBCS)

Semester - VI (CSE / IT / CS&BS)

OBJECT ORIENTED SOFTWARE ENGINEERING

Mrs. Anuradha A. Puntambekar

M.E. (Computer)

Formerly Assistant Professor in

P.E.S. Modern College of Engineering,

Pune

TECHNICAL PUBLICATIONS

SINCE 1993 - An Up-Thrust for Knowledge

Scanned with OKEN Scanner

SYLLABUS

Object Oriented Software Engineering - [CCS356]

UNIT I SOFTWARE PROCESS AND AGILE DEVELOPMENT

Introduction to Software Engineering, Software Process, Perspective and Specialized Process Models - Introduction to Agility - Agile process-Extreme programming - XP Process - Case Study. **(Chapter - 1)**

UNIT II REQUIREMENTS ANALYSIS AND SPECIFICATION

Requirement analysis and specification - Requirements gathering and analysis - Software Requirement Specification - Formal system specification - Finite State Machines - Petrinets - Object modelling using UML - Use case Model - Class diagrams - Interaction diagrams - Activity diagrams - State chart diagrams - Functional modelling - Data Flow Diagram - CASE TOOLS. **(Chapter - 2)**

UNIT III SOFTWARE DESIGN

Software design - Design process - Design concepts - Coupling - Cohesion - Functional independence - Design patterns - Model-view-controller - Publish-subscribe - Adapter - Command - Strategy - Observer - Proxy - Facade - Architectural styles - Layered - Client Server - Tiered - Pipe and filter - User interface design - Case Study. **(Chapter - 3)**

UNIT IV SOFTWARE TESTING AND MAINTENANCE

Testing - Unit testing - Black box testing - White box testing - Integration and System testing - Regression testing - Debugging - Program analysis - Symbolic execution - Model Checking - Case Study. **(Chapter - 4)**

UNIT V PROJECT MANAGEMENT

Software Project Management - Software Configuration Management - Project Scheduling- DevOps : Motivation - Cloud as a platform - Operations - Deployment Pipeline : Overall Architecture Building and Testing - Deployment - Tools - Case Study. **(Chapter - 5)**

TABLE OF CONTENTS

UNIT I

Scanned with OKEN Scanner

UNIT II

Scanned with OKEN Scanner

UNIT I

1 Software Process and Agile Development

Syllabus

Introduction to software engineering, Software process, Perspective and specialized process models, Introduction to Agility, Agile process, Extreme programming, XP process.

Contents

Scanned with OKEN Scanner

1.1 Introduction to Software Engineering

AU : Dec.-13, 17, Marks 4

*"Software engineering is a **discipline** in which **theories**, **methods** and **tools** are applied to develop professional software **product**."*

In software engineering the **systematic** and **organized approach** is adopted. Based on the nature of the problem and development constraints various **tools and techniques** are applied in order to develop **quality software**.

The definition of software engineering is based on two terms :

- **Discipline :** For finding the solution to the problem an Engineer applies appropriate theories, methods and tools. While finding the solutions, Engineers must think of the organizational and financial constraints. Within these constraints only, he/she has to find the solution.
- **Product :** The software product gets developed after following systematic theories, methods and tools along with the appropriate management activities.

1.1.1 Defining Software

Software is nothing but a collection of computer programs and related documents that are intended to provide desired features, functionalities and better performance.

Software products may be :

1. **Generic** - That means developed to be sold to a range of different customers.
2. **Custom** - That means developed for a single customer according to their specification.

1.1.2 Software Characteristics

Software development is a logical activity and therefore it is important to understand basic characteristics of software. Some important characteristics of software are :

- **Software is engineered, not manufactured**
  - Software development and hardware development are two different activities.
  - A good design is a backbone for both the activities.
  - Quality problems that occur in hardware manufacturing phase can not be removed easily. On the other hand, during software development process such problems can be rectified.
  - In both the activities, developers are responsible for producing qualitative product.

- **Software does not ware out**
  - In early stage of hardware development process the failure rate is very high because of manufacturing defects. But after correcting such defects the failure rate gets reduced.
  - The failure rate remains constant for some period of time and again it starts increasing because of environmental maladies (extreme temperature, dusts and vibrations).
  - On the other hand software does not get affected from such environmental maladies. Hence ideally it should have an *"idealized curve"*. But due to some **undiscovered errors** the failure rate is high and drops down as soon as the errors get corrected.
  - Hence in failure rating of software the *"actual curve"* is as shown below :

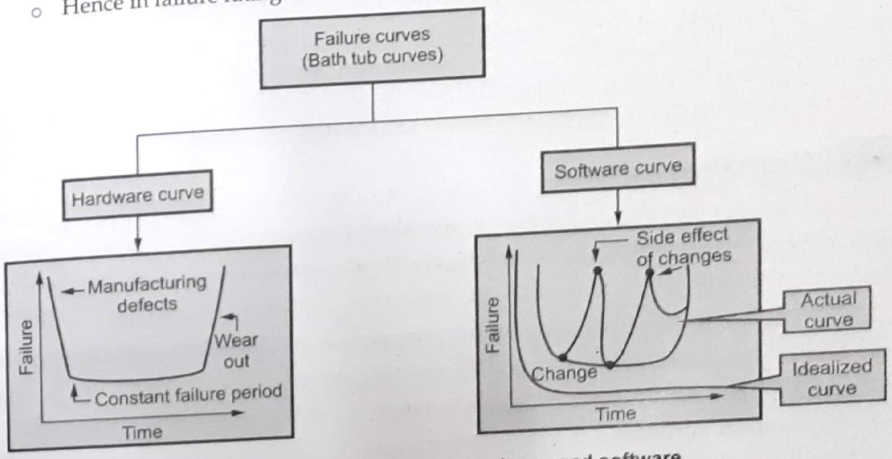

Fig. 1.1.1 Failure curves for hardware and software

  - During the life of software if any change is made, some **defects** may get **introduced**.
  - This causes failure rate to be high. Before the curve can return to original steady state another change is requested and again the failure rate becomes high.
  - Thus the failure curve looks like a spike. Thus frequent changes in software cause it to deteriorate.
  - Another issue with software is that there are ***no spare parts for software***.
  - If hardware component wears out it can be replaced by another component but it is not possible in case of software.
  - Therefore **software maintenance** is **more difficult** than the hardware maintenance.

Scanned with OKEN Scanner

- **Most software is custom built rather than being assembled from components**
  - While developing any hardware product firstly the circuit design with desired functioning properties is created.
  - Then required hardware components such as ICs, capacitors and registers are assembled according to the design, but this is not done while developing software product.
  - Most of the software is custom built.
  - However, now the software development approach is getting changed and we look for reusability of software components.
  - It is practiced to reuse algorithms and data structures.
  - Today software industry is trying to make library of reusable components.
  - **For example :** In today's software, GUI is built using the reusable components such as message windows, pull down menus and many more such components. The approach is getting developed to use in-built components in the software. This stream of software is popularly known as *component engineering*.

1.1.3 Categories of Software

Software can be applied in a situation for which a predefined set of procedural steps (algorithm) exists. Based on a complex growth of software it can be classified into following categories.

- **System software** - It is collection of programs written to service other programs. Typical programs in this category are compiler, editors and assemblers. The purpose of the system software is to establish a communication with the hardware.
- **Application software** - It consists of standalone programs that are developed for specific business need. This software may be supported by database systems.
- **Engineering/Scientific software** - This software category has a wide range of programs from astronomy to volcanology, from automative stress analysis to space shuttle orbital dynamics and from molecular biology to automated manufacturing. This software is based on complex numeric computations.
- **Embedded software** - This category consists of program that can reside within a product or system. Such software can be used to implement and control features and functions for the end-user and for the system itself.
- **Web applications** - Web application software consists of various web pages that can be retrieved by a browser. The web pages can be developed using programming languages like JAVA, PERL, CGI, HTML, DHTML.

- **Artificial Intelligence software** - This kind of software is based on knowledge based expert systems. Typically, this software is useful in robotics, expert systems, image and voice recognition, artificial neural networks, theorem proving and game playing.

Review Questions

1. *What is the impact of reusability in software development process?* **AU : Dec.-17, Marks 4**
2. *Write a note on the unique characters of a software.* **AU : Dec.-17, Marks 3**

1.2 Goals and Objectives of Software

While developing software following are common objectives.

1. **Satisfy users requirements** - Many programmers simply don't do what the end user wants because they do not understand user requirements. Hence it becomes necessary to understand the demand of end user and accordingly software should be developed.
2. **High reliability** - Mistakes or bugs in a program can be expensive in terms of human lives, money, and customer relation. For instance, Microsoft has faced many problems because earlier release of windows has many problems. Thus software should be delivered only if high reliability is achieved.
3. **Low maintenance costs** - Maintenance of software is an activity that can be done only after delivering the software to the customer. Any small change in software should not cause restructuring of whole software. This indicates that the design of software has poor quality.
4. **Delivery on time** - It is very difficult to predict the exact time on which the software can be completed. But a systematic development of software can lead to meet the given deadline.
5. **Low production costs** - The software product should be cost effective.
6. **High performance** - The high performance software are expected to achieve optimization in speed and memory usage.
7. **Ease of reuse** - Use same software in different systems and software.

Environments reduce development costs and also improve the reliability. Hence reusability of developed software is an important property.

Scanned with OKEN Scanner

1.3 Difference between Software Product and Program

Sr. No.	Program	Software product
1.	Programs are developed by individual user and it is used for personal use.	Software product is developed by multiple users and it is used by large number of people or customers.
2.	Programs are small in size and possessing limited functionality.	Software product consists of multiple program codes, related documents such as SRS, design documents user manuals, test cases and so on.
3.	Generally only one person uses the program, hence there is a lack of user interface.	Good graphical user interface is most required by any software product.
4.	Program is generally developed by **programmer.**	Software product is developed by **software engineers** who are large in number and work in a team. Therefore systematic approach of developing software product must be applied.
5.	**For example :** Program of sorting n elements.	**For example :** A word processing software.

1.4 Layered Technology

AU : May-22, Marks 7

- Software engineering is a layered technology. Any software can be developed using these layered approaches. Various layers on which the technology is based are **quality focus layer, process layer, methods layer, tools layer.**
- A disciplined **quality management** is a backbone of software engineering technology.
- **Process layer** is a foundation of software engineering. Basically, process defines the framework for timely delivery of software.

Fig. 1.4.1 Layered technology

- In **method layer** the actual method of implementation is carried out with the help of requirement analysis, designing, coding using desired programming constructs and testing.
- Software **tools** are used to bring automation in software development process.
- Thus software engineering is a **combination** of process, methods, and tools for development of quality software.

Review Question

1. *Software engineering is a layered approach. Justify.* **AU : May-22, Marks 7**

1.5 Software Process

AU : Dec.-10,17,22, Marks 7

Software process can be defined as the structured set of activities that are required to develop the software system.

The fundamental activities are :

- Specification
- Design and implementation
- Validation
- Evolution

A software process model is an abstract representation of a process. It presents a description of a process from some particular perspective.

1.5.1 Common Process Framework

The process framework is required for representing the common process activities.

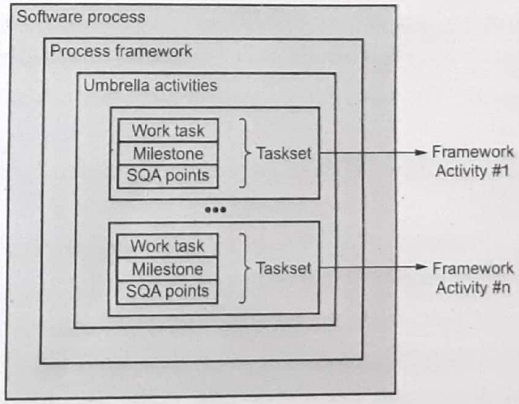

Fig. 1.5.1 Software process framework

As shown in Fig. 1.5.1, the software process is characterized by process framework activities, task sets and umbrella activities.

Process framework activities

- Communication
  - By communicating customer requirement gathering is done.
- Planning - Establishes engineering work plan, describes technical risks, lists resource requirements, work products produced and defines work schedule.
- Modeling - The software model is prepared by :
  - Analysis of requirements
  - Design

Scanned with OKEN Scanner

- Construction - The software design is mapped into a code by :
  - Code generation
  - Testing
- Deployment - The software delivered for customer evaluation and feedback is obtained.

Task sets - The task set defines the actual work done in order to achieve the software objective. The task set is used to adopt the framework activities and project team requirements using :

- Collection of software engineering work tasks
- Project milestones
- Software quality assurance points

Umbrella activities - The umbrella activities occur **throughout the process**. They focus on project management, tracking and control. The umbrella activities are

1. **Software project tracking and control** - This is an activity in which software team can **assess progress** and take corrective action to **maintain schedule**.
2. **Risk management** - The **risks** that may affect project outcomes or quality can be **analyzed**.
3. **Software quality assurance** - These are activities required to **maintain** software **quality**.
4. **Formal technical reviews** - It is required to **assess engineering work products to** uncover and **remove errors** before they propagate to next activity.
5. **Software configuration management** - Managing of configuration process when any **change** in the software occurs.
6. **Work product preparation and production** - The activities to create models, documents, logs, forms and lists are carried out.
7. **Reusability management -** It defines criteria for work product reuse.
8. **Measurement** - In this activity, the process can be defined and collected. Also **project and product measures** are used to assist the software team in delivering the required software.

1.5.2 Capability Maturity Model (CMM)

- The Software Engineering Institute (SEI) has developed a comprehensive process meta-model emphasizing **process maturity**. It is predicated on a **set of system and software capabilities** that should be present when organizations reach different levels of process capability and maturity.

- The Capability Maturity Model (CMM) is **used** in assessing how well an organization's processes allow **to complete and manage** new software projects.
- Various process maturity levels are :

Level 1 : Initial - Few processes are defined and individual efforts are taken.

Level 2 : Repeatable - To track cost schedule and functionality basic project management processes are established. Depending on earlier successes of projects with similar applications necessary process discipline can be repeated.

Level 3 : Defined - The process is standardized, documented and followed. All the projects use documented and approved version of software process which is useful in developing and supporting software.

Level 4 : Managed - Both the software process and product are quantitatively understood and controlled using detailed measures.

Level 5 : Optimizing - Establish mechanisms to plan and implement change. Innovative ideas and technologies can be tested.

Thus CMM is used for improving the software project.

Review Questions

1. *Explain the CMMI model to access the organization level.* **AU : Dec.-17, Marks 7**
2. *Discuss in brief about the typical activities involved under umbrella activities in software engineering.* **AU : Dec.-22, Marks 7**

1.6 Prescriptive Process Models

AU : May-05,06,09,15,16,17,18,22, Dec.-06,09,11,16,17,19,20,22, Marks 16

- **Definition of Process Model :** The process model can be defined as the **abstract representation of process**. The appropriate process model can be chosen based on abstract representation of process.
- The software process model is also known as **Software Development Life Cycle (SDLC) Model** or software paradigm.
- These models are called **prescriptive process models** because they are following some rules for correct usage.
- In this model various activities are carried out in some specific sequence to make the desired software product.
- Various prescriptive process models are -

Scanned with OKEN Scanner

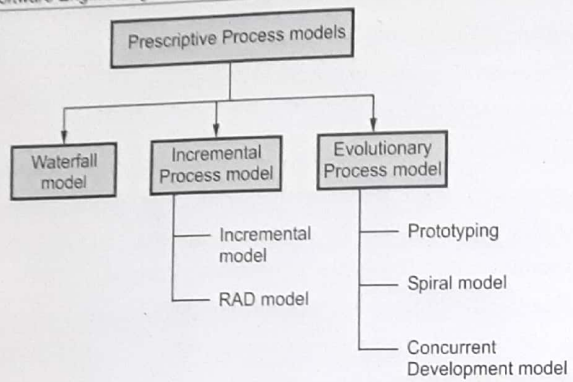

Fig. 1.6.1 Prescriptive process model

1.6.1 Need for Process Model

The software development team must decide the process model that is to be used for software product development and then the entire team must adhere to it. This is necessary because the software product development can then be done systematically. Each team member will **understand - what is the next activity** and **how** to do it. Thus **process model** will bring the **definiteness and discipline** in overall development process.

Every process model consists of **definite entry and exit criteria** for **each phase**. Hence the transition of the product through various phases is definite. If the process model is not followed for software development then any team member can perform any software development activity, this will ultimately cause a chaos and software project will definitely fail without using process model, it is difficult to monitor the progress of software product. Let us discuss various process models one by one.

1.6.2 Waterfall Model

- The waterfall model is also called as '**linear-sequential model**' or '**classic life cycle model**'. It is the oldest software paradigm. This model suggests a systematic, **sequential approach** to software development.
- The software development starts with **requirements gathering phase**. Then progresses through **analysis**, **design**, **coding**, **testing** and **maintenance**. Following figure illustrates waterfall model.
- In **requirement gathering and analysis** phase the **basic requirements** of the system must be understood by software engineer, who is also called **Analyst**. The **information domain**, **function**, **behavioural requirements** of the system are

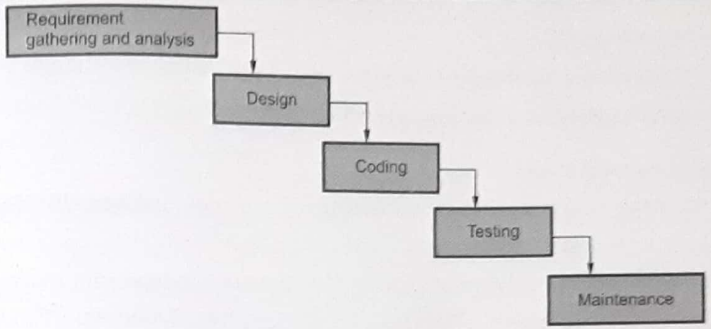

Fig. 1.6.2 Waterfall model

understood. All these requirements are then **well documented** and discussed further with the customer, for reviewing.

- The **design** is an intermediate **step between** requirements analysis and coding.

  Design focuses on program attributes such as -
  - Data structure
  - Software architecture
  - Interface representation
  - Algorithmic details.

  The requirements are translated in some easy to represent form using which coding can be done effectively and efficiently. The design needs to be documented for further use.
- **Coding** is a step in which design is translated into **machine-readable form**. If design is done in sufficient detail then coding can be done effectively. **Programs** are created in this phase.
- **Testing** begins when coding is done. While performing testing the major focus is on **logical internals of the software**. The testing ensures execution of all the paths, functional behaviours. The purpose of testing is to **uncover errors**, **fix the bugs** and **meet the customer requirements**.
- **Maintenance** is the **longest life cycle phase**. When the system is installed and put in practical use then error may get introduced, correcting such errors and **putting it in use** is the **major purpose** of maintenance activity. Similarly, **enhancing system's services** as new requirements are discovered is again maintenance of the system.

This model is widely used model, although it has many drawbacks. Let us discuss benefits and drawbacks.

Scanned with OKEN Scanner

Benefits of waterfall model

1. The waterfall model is **simple to implement**.
2. For implementation of **small systems** waterfall model is useful.

Drawbacks of waterfall model

There are some **problems** that are encountered if we apply the **waterfall model** and those are :

1. It is difficult to follow the sequential flow in software development process. If some changes are made at some phases then it may cause some confusion.
2. The requirement analysis is done initially and sometimes it is not possible to state all the requirements explicitly in the beginning. This causes difficulty in the project.
3. The customer can see the **working model** of the project only **at the end**. After reviewing of the working model; if the customer gets dissatisfied then it causes serious problems.
4. Linear nature of waterfall model induces **blocking states**, because certain tasks may be dependant on some previous tasks. Hence it is necessary to accomplish all the dependant tasks first. It may cause long waiting time.

Example 1.6.1 *Explain how water-fall model is applicable for the development of the following systems :*

a) University accounting system

b) Interactive system that allows railway passengers to find time and other information from the terminals installed in the station.

Solution :

a) University accounting system : If the software developers who have the experience in developing the account systems then building university account system based on existing design could be easily managed with water-fall model.

b) Interactive system that allows railway passengers to find time and other information from the terminals installed in the station.

For developing such interactive system, all the requirements must be correctly identified and analyzed before the designing of the project. The requirements of end-users must be correctly and un-ambiguously understood by the developers prior to design phase. Once the requirements are well defined then using disciplined design, coding and testing phases the required system can be built using water-fall model.

1.6.3 Incremental Process Model

In this model, the initial model with limited functionality is created for user's understanding about the software product and then this model is refined and expanded in later releases.

1.6.3.1 Incremental Model

- The incremental model has same phases that are in waterfall model. But it is iterative in nature. The incremental model has following phases.

  1. Analysis 2. Design
  3. Code 4. Test

- The incremental model delivers series of releases to the customer. These releases are called **increments**. More and more functionality is associated with each increment.

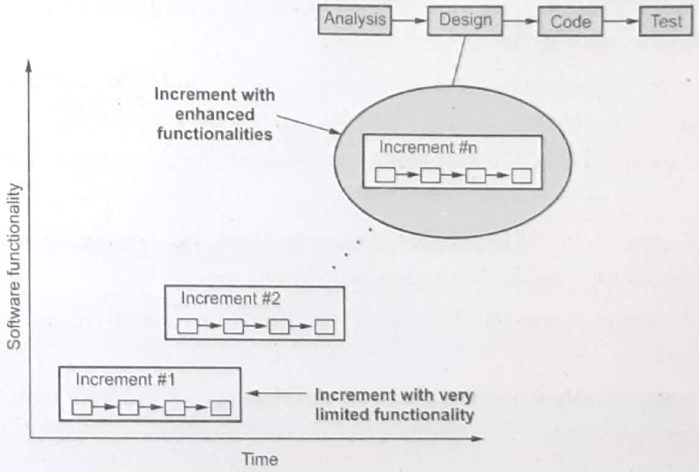

Fig. 1.6.3 The incremental model

- The first increment is called **core product**. In this release the basic requirements are implemented and then in subsequent increments new requirements are added.
- The word processing software package can be considered as an example of incremental model. In the first increment only the document processing facilities are available. In the second increment, more sophisticated document producing and processing facilities, file management functionalities are given. In the next increment spelling and grammar checking facilities can be given. Thus in incremental model progressive functionalities are obtained with each release.

Scanned with OKEN Scanner

When to choose it ?

1. When requirements are reasonably well-defined.
2. When overall scope of the development effort suggests a purely linear effort.
3. When limited set of software functionality needed quickly.

Merits of incremental model

1. The incremental model can be adopted when there are less number of people involved in the project.
2. Technical risks can be managed with each increment.
3. For a very small time span, atleast core product can be delivered to the customer.

1.6.3.2 RAD Model

- The RAD Model is a type of incremental process model in which there is **extremely short development cycle.**
- When the requirements are fully understood and the **component based construction** approach is adopted then the RAD model is used.
- Using the RAD model the fully functional system can be developed within **60 to 90 days.**
- Various phases in RAD are Requirements Gathering, Analysis and Planning, Design, Build or Construction and finally Deployment.
- Multiple teams work on developing the software system using RAD model parallely.
- In the **requirements gathering** phase the developers communicate with the users of the system and understand the business process and requirements of the software system.
- During **analysis and planning** phase, the analysis on the gathered requirements is made and a planning for various software development activities is done.
- During the **design** phase various models are created. Those models are Business model, data model and process model.
- The **build** is an activity in which using the existing software components and automatic code generation tool the implementation code is created for the software system. This code is well tested by its team. The functionalities developed by all the teams are integrated to form a whole.
- Finally the deployment of all the software components (created by various teams working on the project) is carried out.

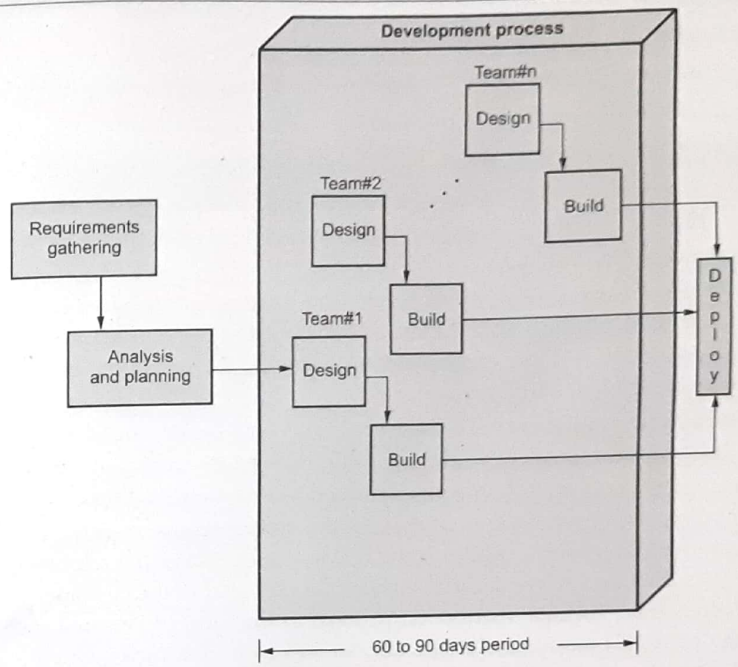

Fig. 1.6.4 Rapid application development

Drawbacks of rapid application development

1. It requires **multiple teams** or large number of people to work on the scalable projects.
2. This model requires **heavily committed developer** and customers. If commitment is lacking then RAD projects will fail.
3. The projects using RAD model requires **heavy resources**.
4. If there is no appropriate modularization then RAD projects fail. Performance can be problem to such projects.
5. The projects using RAD model find it difficult to adopt new technologies.

Example 1.6.2 *Which type of applications suit RAD model ? Justify your answer.*

Solution : The RAD model is suitable for information system applications, business applications and the for systems that can be modularized because of following reasons -

1. This model is similar to waterfall model but it uses very short development cycle.
2. It uses component-based construction and emphasises reuse and code generation.

Scanned with OKEN Scanner

3. This model uses multiple teams on scaleable projects.
4. The RAD model is suitable for the projects where technical risks are not high.
5. The RAD model requires heavy resources.

Example 1.6.3 *Provide three examples of software projects that would be amenable to incremental model. Be specific*

Solution : There can various examples of software projects that would be amenable to incremental model. For instance -

1. **Banking software service :** This service can be personal service. That means for personal banking system the incremental model can be used. In later state of increments, this system can implement insurance service, home loans and some other features of banking services.
2. **Web browser application :** The base application can be developed and distributed. This is the basic increment of the application. In the later increments the plugins can be provided to enhance the experience of web browser applications.
3. **Operating system software :** The operating system software providing the basic system handling functionalities is the first increment. After the release of the basic versions then updates or security patches are provided to the customer in the form of increments. Various distribution package in the form of versions such as basic home edition, premium, ultimate and so on can be the increments of operating system software.

1.6.4 Evolutionary Process Model

While developing the software systems, it is often needed to make modifications in earlier development phases or the tasks sets. If the development process is linear or in a straight line (from requirements gathering to deployment) then the end product will be unrealistic. In such cases, the **iterative approach** needs to be adopted. The evolutionary process model is **iterative model**.

1.6.4.1 Prototyping

- In prototyping model initially the requirement gathering is done.
- Developer and customer define overall objectives; identify areas needing more requirement gathering.
- Then a quick design is prepared. This design represents what will be visible to user in input and output format.
- From the quick design a prototype is prepared. Customer or user evaluates the prototype in order to refine the requirements. Iteratively prototype is tuned for satisfying customer requirements. Thus prototype is important to identify the software requirements.

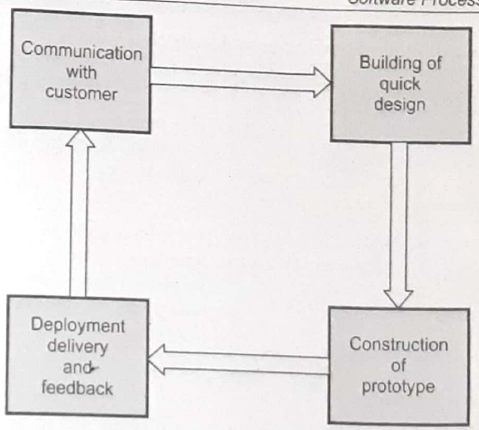

Fig. 1.6.5 Prototyping

- When working prototype is built, developer use existing program fragments or program generators to throw away the prototype and rebuild the system to high quality.
- Certain classes of mathematical algorithms, subset of command driven systems and other applications where results can be easily examined without real time interaction can be developed using prototyping paradigm.

When to choose it ?

- Software applications that are relatively easy to prototype and that always involve Human-Computer Interaction (HCI), the prototyping model is suggested.
- A general objective of software is defined but not detailed input, processing or output requirements. Then in such a case prototyping model is useful.
- When the developer is unsure of the efficiency of an algorithm or the adaptability of an operating system then prototype serves as a better choice.

Drawbacks of prototyping

1. In the first version itself, customer often wants "few fixes" rather than rebuilding of the system whereas rebuilding of new system maintains high level of quality.
2. The first version may have some compromises.
3. Sometimes developer may make implementation compromises to get prototype working quickly. Later on developer may become comfortable with compromises and forget why they are inappropriate.

Scanned with OKEN Scanner

Comparison between prototyping and incremental process model

Sr. No.	Prototyping	Incremental process model
1.	Some requirements are gathered initially, but there may be change in requirements when the working prototype is shown to the customer.	The requirements are precisely defined and there is no confusion about the final product of the software.
2.	The development team has adequate domain knowledge. Similarly they can adopt the new technologies if product demands.	The development team with less domain knowledge can be accommodated due to iterative nature of this model. The change in technology in the later phase can not be tolerated.
3.	All the end-users are involved in all phases of development.	All the end-users need not be involved in all the phases of development.
4.	There can be use of some reusable software components in project development process.	There is no use of reusable components in development process.

1.6.4.2 Spiral Model

- This model possess the **iterative nature** of prototyping model and controlled and **systematic approaches** of the linear sequential model.
- This model gives efficient development of incremental versions of software. In this model, the software is developed in series of increments.
- The sprial model is divided into a number of **framework activities**. These framework activities are denoted by **task regions**.
- Usually there are **six tasks regions**. The spiral model is as shown in Fig. 1.6.6. (See Fig. 1.6.6 on next page)
- Spiral model is realistic approach to development of **large-scale systems** and software. Because customer and developer better understand the problem statement **at each evolutionary level**. Also **risks** can be identified or rectified at each such level.
- In the **initial pass**, product specification is built and in subsequent passes around the spiral the prototype gets developed and then more improved versions of software gets developed.
- During **planning phase**, the cost and schedule of software can be planned and adjusted based on feedback obtained from customer evaluation.
- In spiral model, project **entry point axis** is defined. This axis represents starting point for different types of projects.
- For instance, concept development project will start at core of spiral and will continue along the spiral path. If the concept has to be developed into actual project then at entry point 2 the product development process starts. Hence entry

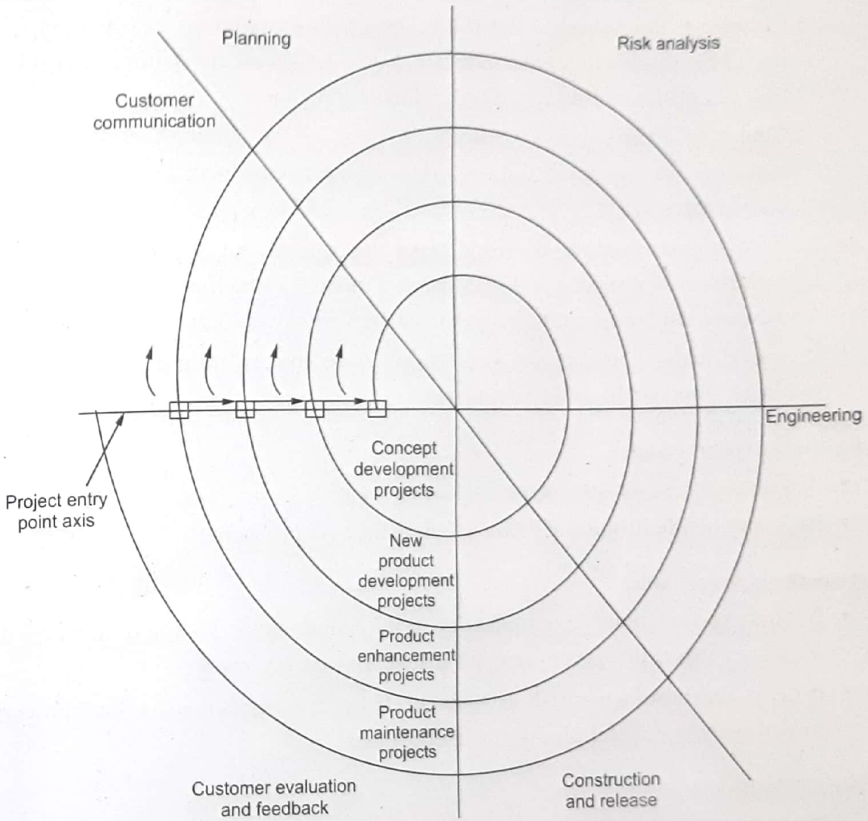

Fig. 1.6.6 Spiral model

point 2 is called product development project entry point. The development of the project can be carried out in iterations.

- The task regions can be described as :
  - **i) Customer communication** - In this region, it is suggested to establish customer communication.
  - **ii) Planning** - All planning activities are carried out in order to define resources time line and other project related activities.
  - **iii) Risk analysis** - The tasks required to calculate technical and management risks are carried out.
  - **iv) Engineering** - In this task region, tasks required to build one or more representations of applications are carried out.

Scanned with OKEN Scanner

v) **Construct and release** - All the necessary tasks required to construct, test, install the application are conducted. Some tasks that are required to provide user support are also carried out in this task region.

vi) **Customer evaluation** - Customer's feedback is obtained and based on customer evaluation required tasks are performed and implemented at installation stage.

- In each region, number of **work tasks** are carried out depending upon the characteristics of project. For a small project relatively small number of work tasks are adopted but for a complex project large number of work tasks can be carried out.
- In spiral model, the software engineering team **moves around the spiral** in a clockwise direction beginning at the core.

Advantages of spiral model

- Requirement changes can be made at every stage.
- **Risks can be identified** and rectified before they get problematic.

Drawbacks of spiral model

- It is based on **customer communication**. If the communication is not proper then the software product that gets developed will not be up to the mark.
- It demands considerable **risk assessment**. If the risk assessment is done properly then only the successful product can be obtained.

When to choose it ?

1. When the prototypes for the software functionality are needed.
2. When requirements are **not very clearly defined** or complex.
3. When the **large or high budget** projects need to be developed.
4. When the **risk assessment** is very critical and essential
5. When project is not expected within a specific limited time span.

Comparison between Spiral model and Prototyping model

Sr. No.	Spiral model	Prototyping model
1.	The development team with less domain knowledge can be accommodated due to iterative nature of this model. The change in technology in the later phase can not be tolerated.	The development team has adequate domain knowledge. Similarly they can adopt the new technologies if product demands.

2.	All the end-users need not be involved in all the phases of development.	All the end-users are involved in all phases of development.
3.	Funding are not stable for the projects that can be developed using spiral model.	Funding are stable for these type of projets.
4.	The requirements that are gathered and analyzed are high reliability requirements.	Some requirements are gathered initially, but there may be change in requirements when the working prototype is shown to the customer.

Difference between waterfall model and spiral model

Sr. No.	Waterfall model	Spiral model
1.	It requires well **understanding of requirements** and familiar technology.	It is developed in **iterations**. Hence the requirements can be identified at new iterations.
2.	**Difficult** to accommodate **changes** after the process has started.	The required changes can be made at every stage of **new version**.
3.	Can accommodate iteration but indirectly.	It is iterative model.
4.	**Risks** can be identified **at the end** which may cause failure to the product.	**Risks** can be identified and **reduced** before they get problematic.
5.	The customer can see the **working model** of the project only **at the end**. After reviewing of the working model; if the customer gets dissatisfied then it causes serious problems.	The customer can see the **working product at certain stages** of iterations.
6.	**Customers prefer** this model.	**Developers prefer** this model.
7.	This model is good for **small systems**.	This model is good for **large systems**.
8.	It has **sequential** nature.	It has **evolutionary** nature.

1.6.4.3 Concurrent Development Model

- The concurrent development model is also called as **concurrent engineering.**
- In this model, the framework activities or software development tasks are represented as **states.**
- The **modeling** or **designing** phase of software development can be in one of the states like *under development, waiting for modification, under revision* or *under review* and so on.
- Fig. 1.6.7 represents these states.

Scanned with OKEN Scanner

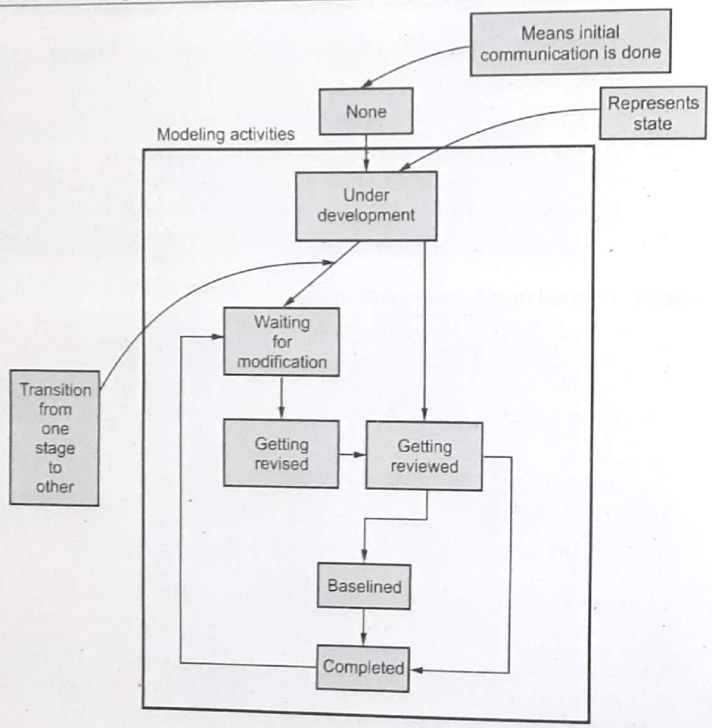

Fig. 1.6.7 Concurrent development model

- All the software development activities exist concurrently in this model but these activities can be in various **states.**
- These states make transitions. That is during **modeling,** the transition from **under development** state to **waiting for modification** state occurs.
- This model basically defines the **series of events** due to which the transition from one state to another state occurs. This is called **triggering.** These series of events occur for every software development activity, action or task.
- This model defines various activities that occur concurrently and a network of activities is defined.

Advantages

1. All types of software development can be done using concurrent development model.
2. This model provides accurate picture of current state of project.
3. Each activity or task can be carried out concurrently. Hence this model is an efficient process model.

Example 1.6.4 *Compare and contrast the different life cycle models.* **AU : Dec.-11, Marks 6**

Solution :

Waterfall model	Spiral model	Prototyping model	Incremental model
Requirements must be clearly understood and defined at the beginning only.	The **requirements** analysis and gathering can be done in iterations because requirements get changed quite often.	**Requirements analysis** can be made in the later stages of the development cycle, because requirements get changed quite often.	The **requirements analysis** can be made in the later stages of the development cycle.
The **development team** having the adequate experience of working on the similar project is chosen to work on this type of process model	The **development team** having less experience of working on the similar projects is allowed in this process model	The **development team** having less experience of working on the similar projects is allowed in this process model.	The **development team** having the adequate experience of working on the similar project is chosen to work on this type of process model.
There is **no user involvement** in all the phases of development process.	There is **no user involvement** in all the phases of development process.	There is **user involvement** in all the phases of development process.	There is **user involvement** in all the phases of development process.
When the requirements are reasonably well defined and the development effort suggests a purely linear effort then the **waterfall model is chosen.**	Due to iterative nature of this model, the risk identification and rectification is done before they get problematic. Hence for handling real time problems the **spiral model is chosen.**	When developer is unsure about the efficiency of an algorithm or the adaptability of an operating system then the **prototyping model is chosen.**	When the requirements are reasonably well defined, the development effort suggests a purely linear effort and when limited set of software functionality is needed quickly then the **incremental model is chosen.**

Example 1.6.5 *For the scenario described below, which life cycle model would you choose? Give the reason why you would choose this model.*
You are interacting with the MIS department of a very large oil company with multiple departments. They have a complex legacy system. Migrating the data from this legacy system is not an easy task and would take a considerable time. The oil company is very particular about processes, acceptance criteria and legal contracts. **AU : Dec.-11, Marks 5**

Solution : Legacy applications are database management systems. The software companies normally migrate these applications to some relational databases. For data migrating projects a cohesive method that enables the developer to cycle or spiral through

Scanned with OKEN Scanner

the migration process is required. In this migrating process following steps need to be followed -

1. Analyze the data.
2. Extract the data which is to be transformed.
3. Transform the extracted data.
4. Validate the transformed data.
5. Load the validated data to the target system.

These above mentioned steps are repeated until the migration is successfully completed. The risk management must be done properly. All these factors suggest that the spiral model is suitable for the given project.

Example 1.6.6 *Describe at least one scenario where*
1. *RAD model would be applicable and not the waterfall model.*
2. *Waterfall model is preferable to all other models.* **AU : Dec.-11, Marks 10**

Solution : 1. RAD model would be applicable and not the waterfall model :

Following are some scenario where RAD model would be applicable and not the waterfall model.

A) The projects in which users are involved in all phases.

B) The projects in which users are experts of problem domain.

C) The project is enhancement of existing system.

2. Waterfall model is preferable to all other models :

In the following scenario waterfall model is preferable to all other models.

The project for which requirements are easily understandable and defined.

Example 1.6.7 *How does a spiral model represent a process suitable to represent a real time problem ?* **AU : CSE, May-05, Marks 8**

Solution : Spiral model represents a process suitable to represent a real time problem because of following reasons -

1. Software evolves as the project progresses. And at every evolutionary level the risks are identified and managed and risks are reduced at every stage.
2. It enables the developer to apply the prototype approach at any stage in the evolution of the product. It helps in adopting the approach systematic stepwise development of the product.
3. The iterative frameworks help in analyzing the product at every evolutionary stage.

4. The spiral model demands a direct consideration of technical risks at all stages of project. The risks are reduced before they get problematic.

Example 1.6.8 *Which type of applications suit RAD model ? Justify your answer.* **AU : May-06, Marks 10**

Solution : The RAD model is suitable for information system applications, business applications and the for systems that can be modularized because of following reasons -

1. This model is similar to waterfall model but it uses very short development cycle.
2. It uses component-based construction and emphasises reuse and code generation.
3. This model uses multiple teams on scaleable projects.
4. The RAD model is suitable for the projects where technical risks are not high.
5. The RAD model requires heavy resources.

Example 1.6.9 *Discuss the major differences between software life cycle model and a process model.* **AU : CSE, Dec.-06, Marks 4**

Solution :

Software life cycle model	Process model
This model is based on common four activities : analysis, design, code and testing.	This model is based on problem definition, technical development, software integration and existing status.
The software development process can be clearly and systematically defined in phases.	The software development process cannot be clearly defined in phases.
Customer interaction is possible in every stage of software development process in this model.	Customer interaction is only at initial stage of software development.
Large scale projects can be handled using this approach.	Large scale projects may be caught in chaotic situation using this approach.

Example 1.6.10 *As you move outward along with process flow path of the spiral model, what can we say about the software that is being developed or maintained ?* **AU : CSE, May.-09, Marks 4**

Solution : When software engineering team moves around the spiral, the first circuit around the spiral results in development of product specification. The subsequent passes around the spiral might be used to develop prototype in more subsequent manner. In each pass, through planning region, some adjustments to project plan are made. Cost and schedule adjustments can also be made according to customer feedback.

Scanned with OKEN Scanner

Example 1.6.11 *A software project which is considered to be very simple and the customer is in position of giving all the requirements at the initial stage, which process model would you prefer for developing the project ?* **AU : IT, Dec.-09, Marks 16**

Solution : The linear sequential model or a waterfall model is appropriate when a simple project has to be developed with already known requirements.

Waterfall model - Refer section 1.6.2

Example 1.6.12 *Assume that you are the technical manager of a software development organization. A client approached you for a software solution. The problems stated by the client have uncertainties which lead to loss if it not planned and solved. Which software development model you will suggest for this project – justify. Explain that model with its pros and cons and neat sketch.* **AU : Dec.-16, Marks 16**

Solution : The uncertainties in problem statement leads to risks in the project. Hence the spiral model can be used for development of such project.

Spiral model - Refer section 1.6.4.2.

Example 1.6.13 *What is the role of user participation in the selection of a life cycle model?* **AU : May-16, Marks 8**

Solution : Due to users' participation , their understanding about the project or system increases. Following are the issues that depict the role of user participation in selection of life cycle model.

1. Prototype model and RAD model are the life cycle model in which **user involvement** is expected in **all the phases.**
2. There is no user involvement expected in all the phases of waterfall, spiral model and evolutionary development model.
3. The waterfall model, spiral model, iterative enhancement model or evolutionary development model **do not require limited user participation.**
4. RAD model and Prototype model does not allow limited user participation.
5. Prototype model, RAD Model, Evolutionary development model require that the **users must be experts of problem domain.** They should understand the system properly well in advance.
6. For the selection of waterfall model, it is **not** necessary that the **user has expertise of problem** domain.
7. The prototype, iterative enhancement model, spiral model and Evolutionary model can be selected by the users **having no previous experience of participation in similar projects.**
8. For selection of waterfall model and prototype model, the users must have **participated earlier in similar type of projects.**

Example 1.6.14 *Describe the type of situations where iterative enhancement model might lead to difficulties.* **AU : May-16, Marks 8**

Solution :

1. In the prototyping model, during the first prototype itself, the customer often wants **few fixes** rather than rebuilding of new system.
2. Sometimes developer may make implementation compromises to get prototype working quickly. But later on the developer may become **comfortable with compromises** and forget why they were inappropriate.
3. The spiral model is mainly based on customer communication. If the **communication** is **not proper** then the software product that gets developed will not be up to the mark.
4. During iterative enhancement cycle of project development, if the **risk assessment** is **not done** properly then the product will become total failure.

Example 1.6.15 *What is process model ? Describe the process model that you would choose to manufacture a car. Explain by giving suitable reasons.* **AU : May-17, Marks 13**

Solution : Process model : Refer section 1.6.

The process model that can be used for car manufacture is **spiral model.**

Spiral model : Refer section 1.6.4.2.

Reasons :

1) This is a model in which **risks** can be **identified** and rectified before they get problematic.
2) The **iterative framework** of this model helps in analyzing the problem at every evolutionary stage.
3) The required **changes can be made at every stage** of new version.
4) This model is **good for large systems.**

Review Questions

1. *Neatly explain the following process models and write their advantages and disadvantages. i) Spiral model, ii) Rapid application development model.* **AU : May-15, Marks 8**
2. *Discuss the prototyping model. What is the effect of designing a prototype on the overall cost of the software project ?* **AU : May-16, Marks 8**
3. *Which process model is best suited for risk management ? Discuss in detail with an example. Give the advantages and disadvantages of the model.* **AU : Dec.-16, Marks 16**
4. *What is the significance of the spiral model when compared with other models.* **AU : Dec-17, Marks 3**
5. *Which software process model is good for risk management ? Explain the model. Describe how the model is used to layout the objectives, risks and plans for quality improvement.* **AU : May-18, Marks 16**

Scanned with OKEN Scanner

6. *Outline the spiral life cycle model with a diagram.* **AU : Dec.-19, Marks 13, May-22, Marks 7**
7. *Compare and contrast waterfall model, spiral model and iterative model.* **AU : Dec.-20, Marks 13**
8. *Discuss about the waterfall model and spiral model with advantages and disadvantages* **AU : Dec.-22, Marks 7**

1.7 Specialized Model

AU : Dec.-13,14,15,17, May-16,19, Marks 16

The specialized models are used when only collection of specialized technique or methods are expected for developing the specific software.

Various types of specialized models are -

1. Component based development
2. Formal methods model
3. Aspect oriented software development

Let us discuss them in detail.

1.7.1 Component based Development

- The commercial off-the-shelves components that are developed by the vendors are used during the software built.
- These components have specialized targeted functionalities and well defined interfaces. Hence it is easy to integrate these components into the existing software.
- The component based development model makes use of various characteristics of **spiral model**. This model is evolutionary in nature. That means the necessary changes can be made in the software during the each iteration of software development cycle.
- Before beginning the modelling and construction activity of software development the **candidate component** must be searched and analyzed. The components can be simple functions or can be object oriented classes or methods.
- Following steps are applied for component based development -
  - Identify the component based products and analyze them for fitting in the existing application domain.
  - Analyze the component integration issues.
  - Design the software architecture to accommodate the components
  - Integrate the components into the software architecture.
  - Conduct comprehensive testing for the developed software.
- **Software reusability** is the major advantage of component based development.
- The **reusability** reduces the development cycle time and overall cost.

1.7.2 Formal Methods Model

- This model consists of the set of activities in which the formal mathematical specification is used.
- The software engineers specify, develop and test the computer based systems using the mathematical notations. The notations are specified within the formal methods.
- **Cleanroom software engineering** makes use of the formal method approach.
- The **advantage** of using formal methods model is that it overcomes many problems that we encounter in traditional software process models. **Ambiguity, incompleteness and inconsistency** are those problems that can be overcome if we use formal methods model.

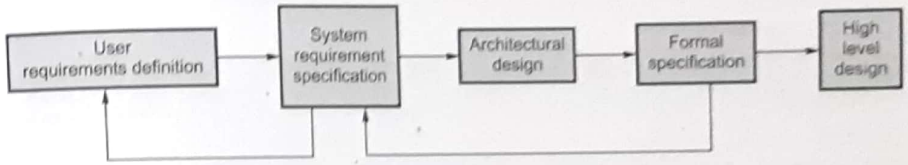

Fig. 1.7.1 Activities for formal method model

- The formal methods model offers **defect-free** software. However there are some **drawbacks** of this model which resists it from getting used widely. These drawbacks are
  - The formal methods model is **time consuming** and **expensive**.
  - For using this model, the developers need the strong **mathematical background** or some extensive training.
  - If this model is chosen for development then the **communication** with customer becomes very **difficult**.

1.7.3 Aspect Oriented Software Development

- In traditional software development process, the system is decomposed into multiple units of **primary functionality**. But there are other issues of **concern** that do not fit into these primary functionalities. Later on this becomes programmers' duty to code modules corresponding to the primary functionality and to incorporate all other concerned issues wherever appropriate.
- Programmers need to keep in mind all the things that need to be done, how to deal with each issue, the problems associated with them and the correct execution. Due to these concerns there are chances of appearing serious problems during

Scanned with OKEN Scanner

application development and maintenance. The coding process for **realizing a concern** becomes very critical.

- **Aspect-Oriented Software Development** focuses on the identification, specification and representation of **cross-cutting concerns and their modularization** into separate functional units as well as their automated composition into a working system.
- **Aspectual requirements** define these cross-cutting concerns that have impact on the software architecture.
- **Aspect Oriented Software Development (AOSD)** is often referred as **Aspect oriented programming.**
- It is a relatively new software engineering paradigm and is not matured enough. But is likely that it will adopt the characteristics of both the spiral and concurrent process models.

Example 1.7.1 *What are pros and cons of using mathematical approach for software development?* **AU : Dec.-15, Marks 6**

Solution : Pros : 1) The mathematical approach is useful in building scientific model, critical systems or simulation of real time systems.

2) The ambiguity or incosistency problems can be eliminated if mathematical approach is adopted.

3) Developers are exceptionally committed to the project.

4) The controlled and optimized system can be developed using mathematical approach.

Cons :

1) The developers need the strong mathematical background or some extensive training.

2) The methods of software development are time consuming and expensive.

3) Many times-if mathematical approach is adopted for software development, then communication with customer becomes very difficult.

Example 1.7.2 *Compare the following life cycle models based on their distinguishing factors, strengths and weaknesses, - Waterfall Model, RAD Model, Spiral Model and Formal Methods Model. (Present in the form of table only - use diagrams wherever necessary)*

AU : Dec.-13,14, Marks 16; May-19, Marks 13

OR

Elucidate the key features of the software process models with suitable examples.

AU : May-16, Marks 8

Solution :

Waterfall model	RAD model	Spiral model	Formal methods model
Requirements must be **clearly understood** and defined at the beginning only.	**Requirements** must be **clearly understood** and defined at the beginning only.	The requirements analysis and gathering can be done in iterations because **requirements get changed quite often.**	**Requirements** must be **clearly understood** and defined at the beginning only.
The **development team** having the adequate experience of working on the similar project is chosen to work on this type of process model.	The **development team** having the adequate experience of working on the similar project is chosen to work on this type of process model.	The **development team** having less experience of working on the similar projects is allowed in this process model.	The **development team** having the adequate experience of working on the similar project is chosen to work on this type of process model.
If the development team has **less domain knowledge** or if it new to the technology then such a team is allowed for this kind of process model.	If the development team has **less domain knowledge** or if it new to the technology then such a team is **not allowed** for this kind of process model.	If the development team has **less domain knowledge** or if it new to the technology then such a team is allowed for this kind of process model.	If the development team has **less domain knowledge** or if it new to the technology then such a team is **not allowed** for this kind of process model.
There is **no user involvement** in all the phases of development process.	There is **user involvement** in all the phases of development process.	There is **no user involvement** in all the phases of development process.	**Limited community** makes **use** of formal methods model for their projects because this methodology is based on mathematical theorems, formal methods and automata theory.
Types of Projects : When the **requirements** are reasonably **well defined** and the development effort suggests a purely linear effort then the waterfall model is chosen	When there is a **tight project schedule** or project is **enhancement** of the **existing projects** or if there is use of **reusable components** in the project then for developing such projects this process model is suitable.	Due to iterative nature of this model, the risk identification and rectification is done before they get problematic. Hence for handling real time problems the **spiral model is chosen.**	Whenever there is a need to build the **scientific models** based on mathematical techniques, or **simulation** of some **real time systems** or when there is a need to build the systems that can contribute to the reliability and robustness (normally **critical systems**) then the formal methods models are used.
Diagram : Refer Fig. 1.6.2.	**Diagram :** Refer Fig. 1.6.4.	**Diagram :** Refer Fig. 1.6.6.	**Diagram :** Refer Fig. 1.7.1.

Scanned with OKEN Scanner

Review Question

1. *Explain the component based software development model with a neat sketch.* **AU : Dec.-17, Marks 9**

1.8 Introduction to Agility

- The Agile Manifesto, also called the **Manifesto for Agile Software Development,** is a formal declaration of **four key values** and **12 principles** to guide an iterative and people-centric approach to software development.
- The agile methods were developed to overcome the weakness of conventional software engineering.
- The agile manifesto is represented by following Fig. 1.8.1.

Individuals and interactions	Over	Process and tools
Working software	Over	Comprehensive documentation
Customer collaboration	Over	Contact negotiation
Responding to change	Over	Following a plan

Fig. 1.8.1 Agile manifesto

1.9 Agile Process

AU : Dec.-16, 19, May-19, Marks 4

- In 1980's the **heavy weight, plan based software** development approach was used to develop any software product.
- In this approach too many things are done which were not directly related to software product being produced.
- This approach was rigid. That means if requirements get changed, then rework was essential. Hence new methods were proposed in 1990's which are known as agile processes.
- The agile processes are the light-weight methods are **people-based** rather than plan-based methods.
- The agile process forces the development team to **focus** on **software** itself rather than design and documentation.
- The agile process believes in **iterative method.**

- The aim of agile process is to **deliver** the working model of software **quickly** to the customer.
- **For example :** Extreme programming is the best known of agile process.

Conventional Software Development Methodology

- As the software project makes the progress, the cost of the changes increases non linearly.
- It is easy to accommodate changes during the requirement gathering stage. At this stage to accommodate the changes - usage scenarios are modified, list of functions can be extended, or written specification be edited.
- As the progresses and if the customer suggest the changes during the testing phase of the software development life cycle then to accommodate these changes the architectural design needs to be modified and ultimately these changes will affect other phases of software development cycle. These changes are actually costly to execute.

Agile Methodology

- The agile method proponents claim that if the software development is carried out using the agile approach then it will allow the software team to accommodate changes late in a software project without dramatic cost and time impact.
- In other words, if the incremental delivery is combined with agile practices such as continuous unit testing and pair programming then the cost of changes can be controlled.
- The following graph represents the how the software development approach has a strong influence on the development cost due to changes suggested.

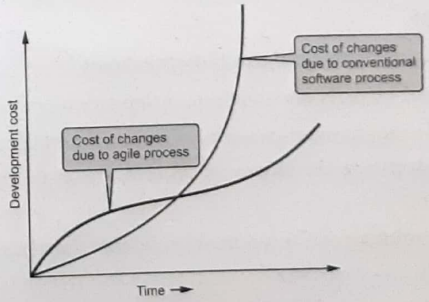

Fig. 1.9.1 Influence of software development approach on agile process

Scanned with OKEN Scanner

Pros

1. **Efficient Delivery :** Agile software development approach emphasizes on delivering the small increments of working software quickly.
2. **Flexibility :** The Agile software development methodology allows for changes and adaptions to be made throughout the development process.
3. **Continuous Improvement :** Agility encourages continuous reflection and improvement in the software.
4. **Transparency :** During agile software development, daily stand-up meetings makes all the team members and stakeholders aware of the project's progress, challenges and decisions.
5. **Reduced Risk :** Transparency helps to identify and mitigate risks early on. When everyone is aware of the project's challenges, they can work together to develop solutions.

Cons

1. **Need for Disciplined Team :** Agile team requires more communication and collaboration for successful software development. This is difficult for teams that are new to Agile.
2. **More Collaboration and Communication :** For Agile Software Development, there needs more collaboration and communication which is challenging for large and distributed projects.
3. **Lack of Documentation :** Agile prioritizes working software over comprehensive documentation, which can be a disadvantage in situations where documentation is essential, such as in highly regulated industries.

1.9.1 Agile Principles

There are famous 12 principles used as agility principles -

1. Satisfy the customer by early and continuous delivery of valuable software.
2. The changes in the requirements must be accommodated. Even though the changes occur late in the software development process, the agile process should help to accommodate them.
3. Deliver working software quite often. Within the shorter time span deliver the working unit.
4. Business people and developers must work together throughput the project.

5. Motivate the people who are building the projects. Provide the environment and support to the development team and trust them for the job to be done.
6. The working software is the primary measure of the progress of the software development.
7. The agile software development approach promote the constant project development. The constant speed for the development of the product must be maintained.
8. To enhance the agility there should be continuous technical excellence
9. The proper attention to be given to technical excellence and good design.
10. The simplicity must be maintained while developing the project using this approach.
11. The teams must be the self-organizing team for getting best architecture, requirements and design.
12. At regular intervals the team thinks over the issue of becoming effective. After the careful review the team members adjusts their behavior accordingly.

Review Questions

1.	*List the principles of agile software development.*	**AU : Dec-16, Marks 4**
2.	*Define Agility. List any five principles of agility.*	**AU : May-19, Marks 5**
3.	*What is agility ? Elaborate the agile principles.*	**AU : Dec.-19, Marks 13**

1.10 Extreme Programming

AU : May-19,22, Dec.-20,22, Marks 13

Extreme Programming (XP) is one of the best known agile process. It is suggested by Kent Beck in 2000.

1.10.1 XP Values

Beck defined the set of five values that serve as a basis for the work performed in XP. These values are -

1. **Communication :** The effective communication must be established between software developers and stakeholders in order to convey the important concepts and to get the important feedback.
2. **Simplicity :** XP focuses on the current needs instead of futures needs to incorporate in the design. Hence the XP believes that the Software design should be simple.
3. **Feedback :** The feedback for the software product can be obtained from the developers of the software, customers, and other software team members.

Scanned with OKEN Scanner

4. **Courage :** The strict adherence to certain XP practices require courage. The agile XP team must be disciplined to design the system today, recognize the future requirements and make the changes dramatically as per the demand.
5. **Respect :** By following the above states XP values the agile team can win the respect of stakeholders.

1.10.2 Process

- The extreme programming process is explained as follows -
- Customer specifies and priorities the system requirements. Customer becomes one of the important members of development team. The developer and customer together prepare a **story-card** in which customer needs are mentioned.
- The developer team then aims to implement the scenarios in the story-card.
- After developing the story-card the development team breaks down the total work in **small tasks**. The efforts and the estimated resources required for these tasks are estimated.
- The customer priorities the stories for implementation. If the requirement changes then sometimes unimplemented stories have to be discarded. Then release the complete software in **small** and frequent **releases**.
- For accommodating new changes, **new story-card** must be developed.
- Evaluate, the system along with the customer.
- This process is demonstrated by the following Fig. 1.10.1.

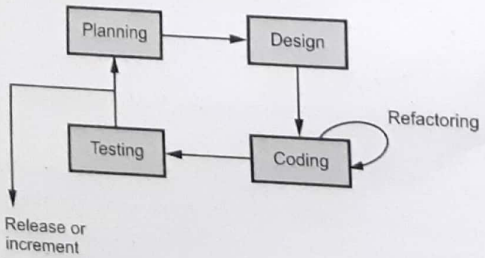

Fig. 1.10.1 XP process

- Various rules and practices used in extreme programming are as given below –

	XP Principle	Description
Planning	User story-cards	Instead of creating a large requirement document user stories are written by the customer in which what they need is mentioned.
	Release planning	A release plan for overall project is prepared from which the iteration plan can be prepared for individual iteration.
	Small releases	The developer breaks down the user stories into small releases and a plan for releasing the small functionalities is prepared.
	Iterative process	Divide the development work into small iterations. Keep the iteration of nearly constant length. Iterative development helps in quick or agile development.
	Stand up meetings	The stand up meetings must be conducted for the current outcomes of the project.
Designing	Simple design	Simple design always takes less time than the complex design. It is always good to keep the things simple to meet the current requirements.
	Spike solution	For answering the tough technical problems create the spike solutions. The goal of these solutions should be to reduce the technical risks.
	Refactoring	Refactoring means reductions in the redundancy, elimination of unused functionalities, redesign the obsolete designs. This will improve the quality of the project.

Scanned with OKEN Scanner

Coding	Customer availability	The most essential requirement of the XP is availability of the customer. In Extreme programming the customer not only helps the developer team but it should be the part of the project.
	Paired programming	All the code to be included in the project must be coded by groups of two people working at the same computer. This will increase the quality of coding.
	Collective code ownership	By having collective code ownership approach the everyone contributes new ideas and not any single person becomes the bottleneck of the project. Anyone can change any line of code to fix a bug or to refactor.
Testing	Unit testing	The test-first development is done in XP. The test framework that contains the automated test case suite is used to test the code. All the code must be tested using unit testing before its release.
	Continuous integration	As soon as one task is finished integrate it into the whole system. Again after such integration unit testing must be conducted.
	No overtime	Working overtime lose the spirit and motivation of the team. Conduct the release planning meeting to change the project scope or to reschedule the project.

As mentioned earlier in Extreme Programming instead of creating large requirement documents the user story-card is prepared. For example : If a project is for creating a banking software then the sample story card can be prepared as follows -

1. Customer makes the balance enquiry.
2. Customer withdraws some amount.
3. Customer deposits some amount.

Banking software checking balance, depositing and withdrawing
When the customer wants to **check the balance** then he just enters his account number and the total amount present in his account is displayed to him.
Then customer can **withdraw** some amount from his account. The limit of amount to be withdrawn must be specified. If the customer crosses that limit then he must be warned. The amount withdrawn must be deducted from the available balance.
The customer can **deposit** the desired amount in his account. The deposited amount must be added to the available balance.

Note that, the user story is broken into various tasks and simple classes must be created for these tasks. From above example it is clear that the useful class in the project could be *'account'* . Various data attributes can be *account_no, customer name, address, balance* and the useful functions can be *deposit(), withdraw(), check_balance()*.

Each task can be separately tested using the automated test cases.

Example 1.10.1 *What is spike solution in XP ?*

Solution : The Extreme Programming (XP) is one of the known agile process. In this process, to answer complex and difficult technical or design problems spike solution is created.

The spike solution is very simple program to explore potential solution. The spike is bulid to only address the problem under examination and all other things can be overlooked.

The spike solution reduces the risk of technical problem and increases the reliability of user story's estimate.

1.10.3 Industrial XP

- The Industrial XP (IXP) can be defined as the organic evolution of XP. It is customer centric. It has expanded role for customers and advanced technical practices.
- Various new practices that are appended to XP to create IXP are as follows -

1. **Readiness Assessment :** In this assessment, following issues are assessed -
   i. Proper environment must be available to support XP.
   ii. Team should contain appropriate and skilled stakeholder
   iii. The organization should support quality programs and continuous improvement.
   iv. The organizational culture should support new values of agile team.
2. **Project Community :** Skilled and efficient people must be chosen as the agile team members for the success of the project. The team is referred as the **community** when extreme programming approach is considered. The project **community** consists of **technologies**, **customers** and **other stakeholders** who play the vital role for the success of the project. The **role** of the community members must be **explicitly defined**.
3. **Project Chartering :** Project chartering means **assessing** the justification for the project as a business application. That means, the IXP team assess whether the project satisfies the goals and objectives of the organization.

Scanned with OKEN Scanner

4. **Test Driven Management :** For assessing the state of the project and its progress th industrial XP needs some **measurable criteria**. In test driven management th project is tested with the help of these measurable criteria.
5. **Retrospectives :** After delivering the software increment, the **specialized review** i conducted which is called as **retrospective**. The intention of retrospectives is t improve the industrial XP process.
6. **Continuous Learning :** The team members are inspired and encouraged to lear new methods and techniques that can **improve the quality** of the product.

Review Questions

1. *Explain the phases in extreme programming process.* **AU : May-19, Marks 8**
2. *Define agile programming. Explain the 12 practices of extreme programming.* **AU : Dec.-20, Marks 13**
3. *Explain Pros and Cons of agile software development.* **AU : May 22, Marks 6**
4. *Discuss about the various practices of extreme programming.* **AU : May-22, Marks 6**
5. *Briefly explain the 12 principles of extreme programming.* **AU : Dec.-22, Marks 6**
6. *State the principles to be followed to achieve agility in software engineering and mention the type of programming that are widely used for agile software development.* **AU : Dec.-22, Marks 6**

1.11 Two Marks Questions with Answers

Q.1 Write the IEEE definition of software engineering. **AU : Dec.-17, May-19**

Ans. : Software engineering is defined as the application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software.

Q.2 What is Software ? List the characteristics. **AU : May-18**

Ans. : Software : Software is a collection of computer programs and related documents that are intended to provide desired features, functionalities and better performance.

Characteristics :

1) Software is engineered not manufactured.

2) Software does not ware out

3) Most software is custom built rather than being assembled from components.

Q.3 What are two type of software products ? **AU : May-12**

Ans. : There are two types of software products -

1. **Generic -** These products are developed and to be sold to the range of different customers.

2. **Custom -** These type of products are developed and sold to the specific group of customers and developed as per their requirements.

Q.4 What is software engineering ? **AU : Dec.-13**

OR

Define software engineering. **AU : Dec.14,22, May-22**

Ans. : Software engineering is a discipline in which theories, methods and tools are applied to develop professional software product.

Q.5 Software doesn't wear out. Justify. (Refer section 1.1.2) **AU : Dec.-13**

Q.6 Distinguish between process and methods. **AU : IT, May-04, 22**

Ans. : Software process can be defined as the structured set of activities that are required to develop the software system. Various activities under software process are-

- Specification
- Design and implementation
- Validation
- Evolution

Method is used mainly in object-oriented programming, the term method refers to a piece of code that is exclusively associated either with a class (called class methods) or with an object (called instance methods).

Q.7 Why software architecture is important in software process ? **AU : IT, May-05**

Ans. : The system architecture defines the role of hardware, software, people, database procedures and other system elements. The architectural design of the system help the developer to understand the system as a whole. Hence system architecture must be built before specifications are completed. Thus architectural design is important in software engineering.

Q.8 What are the drawbacks of rapid application development life cycle model ?

(Refer section 1.6.3.2) **AU : CSE, May-05**

Q.9 List out the problems encountered in linear sequential model.

(Refer section 1.6.2) **AU : May-06**

Q.10 What is meant by 'blocking states' in linear sequential model ? **AU : IT, Dec.-06**

Ans. : The linear nature of linear sequential model brings a situation in the project that some project team members have to wait for other members of the team to complete the dependent tasks. This situation is called "blocking state" in linear sequential model. For example, after performing the requirement gathering and analysis step the design process can be started. Hence the team working on design stage has to wait for gathering of all the necessary requirements. Similarly the programmers can not start coding step unless and until the design of the project is completed.

Scanned with OKEN Scanner

Q.11 What are the advantages of prototyping model ? AU : CSE, Dec.-06

Ans. : Advantages of prototyping model -

1. The working model of the system can be quickly designed by construction of prototype. This gives the idea of final system to the user.
2. The prototype is evaluated by the user and the requirements can be refined during the process of software development.
3. This method encourages active participation of developer and user.
4. This type of model is cost effective.
5. This model helps to refine the potential risks associated with the delivery of the final system.
6. The system development speed can be increased with this approach.

Q.12 Write any two software engineering challenges. AU : IT, May-07, Dec.-22

Ans. : The key challenges faced by software engineering are -

1. Delivery times challenge

There is increasing pressure for **faster delivery** of software. When the complexity of the systems that we develop increases, this challenge becomes harder.

2. Heterogeneity challenge

Sometimes systems are distributed and include **a mix of hardware and software**. This implies that software systems must **cleanly integrate** with other different software systems, built by different organizations and team which may be using different hardware and software platform.

Q.13 Identify in which phase of the software life cycle the following documents are delivered.

a) Architectural design b) Test plan

c) Cost estimate d) Source code document. AU : IT, May-07

Ans. :

	Document	Phase
(a)	Architectural design	Design
(b)	Test plan	Testing
(c)	Cost estimate	Project management and planning
(d)	Source code document	Coding

Q.14 Define the terms product and process in software engineering. AU : IT, May-07

Ans. : The product in software engineering is a standalone entity that can be produced by development organization and sold on the open market to any customer who is able to buy them. The software product consists of computer programs, procedures, and associated

documentation (documentation can be in hard copy form or it may be in visual form). Some of the examples of software product are databases, word processors, drawing tools.

The process in software engineering can be defined as the structured set of activities that are required to develop software system. Various activities under software process are -

- Specification
- Design and implementation
- Validation
- Evolution

Q.15 What are the phases encompassed in the RAD model ? AU : CSE, Dec.-07

Ans. : Various phases in the RAD model are

1. Business modelling
2. Data modelling
3. Process modelling
4. Application generation
5. Testing and turnover

Q.16 Define a system and computer based system. AU : CSE, Dec.-07

Ans. : System : A system can be defined as a purposeful collection of inter-related components working together to achieve some common objectives.

Computer based system : The computer based system can be defined as a set or an arrangement of elements that are organized to accomplish some predefined goal by processing information.

Q.17 Which process model leads to software reuse ? Why ? AU : IT, Dec.-07

Ans. : The object oriented model is used for the software reuse because - this model is based on the incremental development of the software product. This can be done in one or more iterations.

Q.18 State the benefits of waterfall life cycle model for software development. AU : IT, May-08

Ans. : 1. The waterfall model is simple to implement.

2. For implementation of small systems the waterfall model is used.

Q.19 How does "Project Risk" factor affect the spiral model of software development ? AU : CSE, May-09

Ans. : The spiral model demands considerable risk assessment because if a major risk is not uncovered and managed, problems will occur in the project and then it will not be acceptable by end user.

Q.20 Define software. AU : IT, Dec.-09

Ans. : Software is a collection of computer programs and related documents that are intended to provide desired features, functionalities and performance. A software can be of two types - 1. Generic software and 2. Custom software.

Scanned with OKEN Scanner

Q.21 Write down the generic process framework activities that is applicable to any software project. (Refer section 1.5.1) **AU : Dec.-10**

Q.22 What is software process model ? On what basis it is chosen ?

Ans. : The software process model can be defined as abstract representation of process. It is based on nature of software project.

Q.23 What is software process ? **AU : May-13, Dec.-19**

Ans. : Software process can be defined as the structured set of activities that are required to develop the software system. The fundamental activities are -

1. Specification 2. Design and Implementation 3. Validation 4. Evolution

Q.24 Write the process framework and Umbrella activities **AU : May-15**

Ans. : **Process Framework :** Process framework is required for representing the common process activities. The process framework activities are -

1. Communication 2. Planning 3. Modeling
4. Construction 5. Deployment

Umbrella Activities : The umbrella activities focus on project management, tracking and control. These activities are -

1. Software Project tracking and Control
2. Risk Management
3. Software Quality Assurance
4. Formal Technical Review
5. Software Configuration Management
6. Work Production Preparation and production
7. Reusability Management
8. Measurement

Q.25 What are the pros and cons of Iterative software development models ? **AU : Dec.-06, 15**

Ans. : **Pros :** 1) The changes in requirements or additions of functionality is possible.

or

2) Risks can be identified and rectified before they get problematic.

Cons : 1) This model is typically based on customer communication. If the communication is not proper the software product that gets developed will not be exactly as per the requirements.

2) The development process may get continued and never finish.

Q.26 What led to the transition from product oriented development to process oriented development to process oriented development ? **AU : May -16**

Ans. : The software process model led to the transition from product oriented development to process oriented development.

Q.27 Mention the characteristics of software contrasting it with characteristics of hardware. **AU : May -16**

Ans. : 1) Software is engineered and not manufactured.

2) Software does not ware out. 3) The software is custom built rather than being assembled from components.

Q.28 If you have to develop a word processing software product, what process model will you choose ? Justify your answer. **AU : Dec.-16, May-17**

Ans. : The incremental process model will be used to develop word processing software product.

Justification : 1) The working software can be generated quickly and early during the software life cycle.

2) The customers can respond to its functionalities after every increment.

Q.29 Depict the relationship between work product, task, activity and system. **AU : Dec.-16**

Ans. :

- Each framework activity under umbrella activities of software process framework consists of various task sets.
- Each task set consists of work tasks, work products, quality assurance points and project milestones. The task sets accomodate the needs of the system getting developed.

Q.30 List two deficiencies in waterfall model. Which process model do you suggest to overcome each efficiency ? **AU : May-17**

Ans. : i) It is difficult to define all the requirement at the beginning of project, this model is not suitable for accomodating any change.

To overcome this efficiency : Prototyping model.

ii) It does not scale up to large projects.

To overcome this efficiency : Spiral model

Q.31 If you have to develop a word processing software product, what process model will you choose ? Justify your answer. **AU : May-18**

Ans. : The incremental model can be used for developing a word processing software product. This is because the basic functionality of word editing tasks can be developed and verified from the customer in the initial increments. Then in subsequent increments the advanced editing features can be added.

Scanned with OKEN Scanner

Q.32 Compare prototyping approaches in a software process. AU : May-18

Ans. : There are two prototyping approaches in a software process -

1) **Evolutionary Prototyping** - In this approach of system development, the initial prototype is prepared and it is then refined through number of stages to final stage.

2) **Throw-away Prototyping** - Using this approach a rough practical implementation of the system is produced. The requirement problems can be identified from this implementation. It is then discarded. System is then developed using some different engineering paradigm.

Q.33 List any two agile process models. AU : May-19

Ans. : 1) Rational Unified Process Model(RUP) 2) SCRUM.

Q.34 Define evolutionary prototype. AU : Dec.-19

Ans. : Evolutionary prototyping is a software development method where the developer or development team first constructs a prototype. After receiving initial feedback from the customer, subsequent prototypes are produced, each with additional functionality or improvements, until the final product is built.

Q.35 List out the goals of software engineering. *(Refer section 1.2)* AU : Dec.-20

Q.36 What are the various categories of software ? *(Refer section 1.1.3)* AU : Dec.-20

❑❑❑

Scanned with OKEN Scanner