Examples of Case tools are :

- Code generation CASE tools - Visual Studio .NET
- Code analysis CASE tools -Borland Audits
- CASE tools used for development of data models -UML editors
- Cleaning up code CASE tools -Refactoring tools
- Bug tracker CASE tool
- Version control CASE tool -CVS, etc.

Benefits

- Improves the productivity
- Generates small parts of code automatically
- Improves software quality
- Can be integrated with other tools say, with code editor to work with coding.

Review Questions

Q. 1 Define software and state its characteristics.

Q. 2 Differentiate between system software and application software.

Q. 3 Define Software engineering and its Objectives.

Q. 4 Explain the layered technology of Software Engineering.

Q. 5 What is the need of Software Process ?

Q. 6 Identify the following into types of software.

Operating System, Compiler, Testing Software, Sales Analysis, 2D Design Software, Statistical Analysis, Spread Sheet Application, Graphical Analysis and Design, Expert Systems, Automatic Operations Software on POW in Oven/Washing Machines/Gas Stations.

□□□

UNIT I

CHAPTER 2

Software Process

Syllabus

The Software Process, Generic Process Model, The Waterfall Model, Incremental Process Models, Evolutionary Process Models, Concurrent Models, Component-Based Development, The Unified Process Phases, Agile Development- Agility, Agile Process, Extreme Programming.

Syllabus Topic : The Software Process

2.1 Software Process

Q. Explain in brief the various types of Software Process.

A software process identifies a set of activities that are applicable to the development of any software project, regardless of their size or complexity.

A software process is a collection of work activities, actions and tasks that are to be performed when some software project is to be developed.

Software process can be categorized into :

1. Generic Process model – represents a framework activity populated by a set of software engineering activities. It includes :
   - Framework activities
   - Umbrella activities
2. Personal and Team Process models – This model helps in creating a software that best fits either the personal needs of the user or that meets the broader needs of a team. It includes :
   - Personal Software Process (PSP)
   - Team Software Process (TSP)
3. Prescriptive Process models – provides an ordered structure and an effective roadmap to software engineering work. It includes :
   - Waterfall model – Incremental model – RAD model
   - Evolutionary Process models – Prototyping – Spiral model
   - Concurrent Development model

Prescriptive models define a discrete set of activities and actions to accomplish all tasks of the software with milestones, which is used to develop the software. These Process models may not be perfect but they give very good guidance in software development process.

This model is important because,

- It is also referred as rigorous model and provides stability.
- It describes a unique set of frame work activities and organizes them into a process flow.
- These actions are used to create work product to accomplish to meet development goal.
- It finds out- the nature of project, whether is suitable for the people using it, whether it is suitable for the environment, where is it implemented.
- It prescribes a set of process elements, framework activities, software engineering actions tasks, work products quality assurance, change control mechanism of each project.

Syllabus Topic : Generic Process Model

2.2 Generic Process Model

Q. List the umbrella activities followed in generic process model.

This model defines a set of *umbrella activities* which are also a must for any software engineering process as shown in the Fig. 2.2.1.

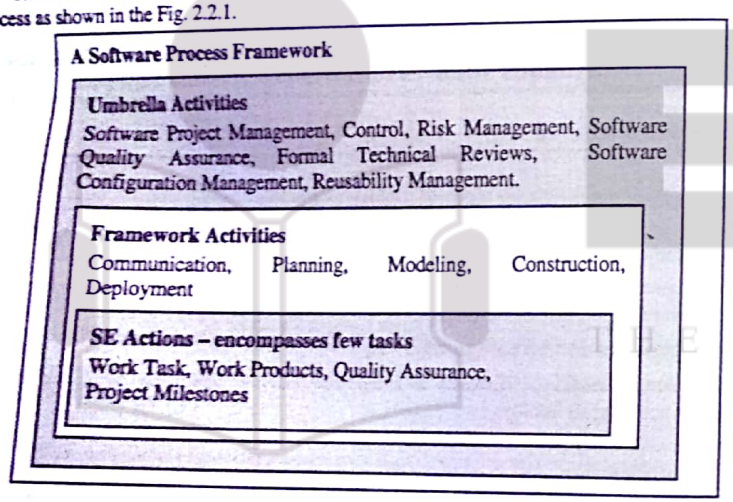

Fig. 2.2.1 : A software process framework

Again, each *framework activity* contains a set of *software engineering activities* which is a collection of tasks that develops a major software product.

2.2.1 Framework Activities

First, we take a look at the generic process framework activities that are must for the development of any software project :

☞ **Communication :**

The main cause of communication is requirement gathering. This activity establishes sufficient interaction or collaboration between the developer and the customer for gathering the requirements and knowing the expectations of the customer.

Example : We can explain the **work task** regarding the **communication activity** of a simple project as listed below :

- Making the list of the end-users, software engineers and support people for the project.
- Inviting all of them for an informal meeting.
- Ask each end-user to make a list of features and functions required.
- Discuss these requirements and prepare a final list.
- Arrange the requirements according to their priority.
- Note the areas of uncertainty.

☞ **Planning :**

This activity defines the software development process to be conducted. It describes all the needed technical tasks, possible risks, the resources that are required, the work product to be produced and the schedule to workout the whole process. This activity plans the work, identifies the resources, tasks and sets the schedule.

☞ **Modeling :**

This activity creates a model (blueprint) which clearly describes the software requirements and the design that will achieve these requirements. This is helpful to both customer and the developer respectively, to understand what he wants from the software and how he can develop it. Modeling is composed of two main activities-*analysis* (requirements gathering, elaboration, negotiation, specification and validation) and *design* (data design, interface design and each module level design).

☞ **Construction :**

This activity includes code generation either manually or using automated tools and then testing the code to correct the errors if any.

☞ **Deployment :**

The software (as a complete product or in a partial stage) is delivered to the customer who then checks the product and provides feedback on evaluation.

The framework activities are applied on every project but the degree of tasks depend on the :

- Type of the project
- Characteristics of the project
- Agreement of the project team on common views.

2.2.2 Process Iteration Activities

The above framework activities discussed in Section 2.2.1 occur in an organized pattern with respect to sequence and time. This work flow pattern of the activities is termed as ***'Process Flow'***.

1. **Linear Process Flow :** Executes the five framework activities in a sequence starting with 'communication' and ending with 'deployment'.
2. **Iterative Process Flow :** Repeats one or more of the five framework activities before proceeding to the next.
3. **Evolutionary process Flow :** Executes the five framework activities in a 'circular/cyclic' manner.
4. **Parallel Process Flow :** At a time, executes one or more activities i.e. one or more of the five framework activities are executed in parallel with the other. Say, modelling of one module is executed parallel to the construction of another module.

Fig. 2.2.2 : Linear Process Flow

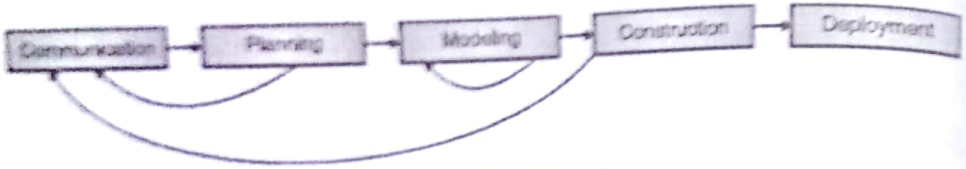

Fig. 2.2.3 : Iterative Process Flow

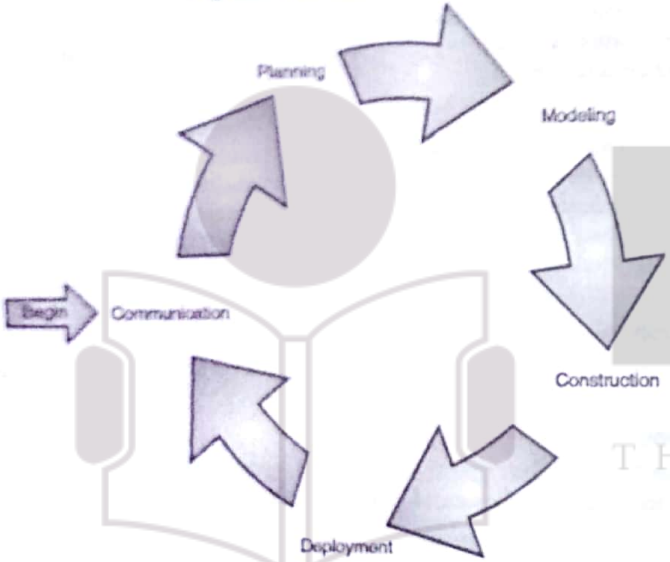

Fig. 2.2.4 : Evolutionary Process Flow

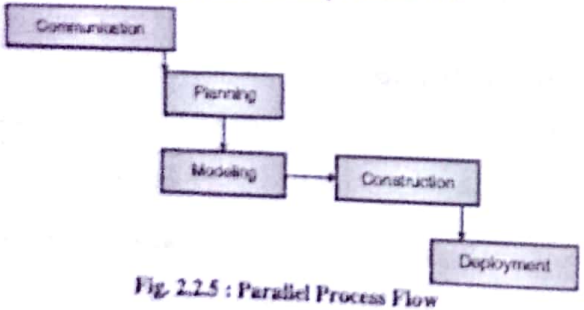

Fig. 2.2.5 : Parallel Process Flow

2.2.3 Umbrella Activities

*Now, we look at the **Umbrella activities** that are applicable throughout the software process :*

- **Software Project Tracking and Control :** Software team assesses the progress of the project plan time to time and takes necessary action to maintain the schedule. Thus, software team tracks and controls the project schedule.
- **Risk Management :** Software team assesses the risks that may affect the outcome of the project or say the quality of the product.
- **Software Quality Assurance :** Software team defines and conducts the activities needed to preserve the quality of the software product.
- **Formal Technical Reviews :** Software team assesses the technical efforts to find and remove the errors before they are forwarded to the next action.
- **Measurement :** Just the coincidence is the four P's of Software Engineering : Project (the task at hand), Process (the manner it is done), Product (the object produced) and People (by whom it is done). Software team collects all the project, process and product measures so that it can be used in combination with all other framework and umbrella activities.
- **Software Configuration Management :** It is about managing the changes and their version throughout the software process.
- **Reusability Management :** defining the criteria for work product reuse and establishes mechanisms to achieve reusable components.
- **Work Product Preparation and Production :** proper planning so as to create work products such as models, documents, logs, forms and lists.

Syllabus Topic : The Waterfall Model

2.3 Waterfall Model

Q. Explain waterfall model in brief.

This model was proposed by the Winston Royce. It is also called as a ***classic life cycle.*** It suggests systematic sequential approach for software development. It is oldest paradigm for software engineering.

It begins with software requirement and customer specification and progress through planning and testing.

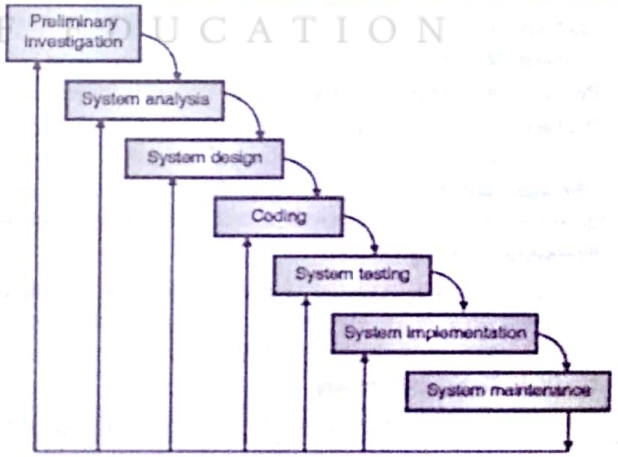

Fig. 2.3.1 : Waterfall model

Step 1 : Preliminary Investigation

Preliminary investigation means total inspection of the existing system i.e. clear understanding of the system.

Its basic task is to find out real problem of the system with causes and complexity of the problem. Its secondary but very important task is to find out all possible solutions to solve that problem and according to that which solution is feasible for the system in terms of technology, cost, operational. Its last task is to mention all benefits can be expected after problem is solved.

So this phase is divided into three main goals as follows:

1. Problem identification
2. Possible and feasible problem solution i.e. Feasibility study.
3. Expected benefits after the problems are solved.

Problem Identification

It requires to completely investigate the environment of the system. Generally it requires studying two environments - Internal environment and external environment, which are listed below :

Sr. No.	Internal Environment	External Environment
1.	Company Management	Customers
2.	Employees of all departments	Management consultant
3.	Internal auditors	External auditors
4.	Data Processing department	Government Policies
5.	Financial Reports	Competitions

There are normally seven types of problems encountered in the system :

1. **Problem of Reliability :** If system may not work properly or same procedures give different (i.e. unreliable) result.
2. **Problem of Validity :** Reports contain misleading information.
3. **Problem of Economy :** System is costly to maintain.
4. **Problem of Accuracy :** Reports have many errors.
5. **Problem of Timeliness :** Every work requires large time.
6. **Problem of Capacity :** Capacity of the system is inadequate.
7. **Problem of Throughput :** System does not produce expected results, or we can say system has more capacity but it accomplishes very less work as compared to capacity.

The main advantage of waterfall model is, it exactly pin points the problem. So it is very useful in setting all goals of the system as well as used to decide system boundaries.

Feasibility Study

Feasibility study is essential to evaluate cost and benefit of the proposed system. This is very important step because on the basis of this; system decision is taken on whether to proceed or to postpone the project or to cancel the project.

Need of Feasibility study

Q. What is the need of feasibility study? Explain its types.

1. *It determines the potential of existing system.*
2. *It finds or defines all problems of existing system.*
3. *It determines all goals of the system.*
4. *It finds all possible solutions of the problems of existing system. We can call it as proposed system.*
5. *It finds technology required to solve these problems.*

6. It determines most suitable solution
7. It determines the required hardware and software.
8. It does the cost estimation in terms of cost of hardware required, software required, designing new system, implementation and training, proposed maintenance cost.
9. It avoids costly repairs, crash implementation of new system.
10. It chooses such system which is easy for customer to understand and use so that no special training is required to train the customer. It may give some training to employees of the system.

Method-Steering committee

This committee conducts detailed study. This committee first studies existing system and identifies all problems and looks into three types of feasibility study. Those are given in Fig. C2.1:

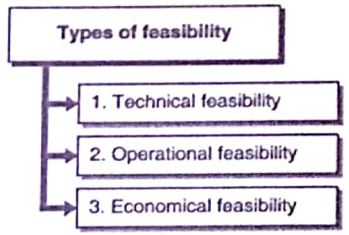

Fig. C2.1 : Types of feasibility

1. Technical feasibility

The committee first finds out *technical feasibility of the existing system.* It involves following steps :

1. It determines available hardware.
2. It determines available computer with configuration.
3. It determines available software.
4. It determines operating time of available system that is computer, hardware software.

After that it finds out *technical feasibility required for the proposed system.* It involves following steps :

1. It mentions new hardware requirements of proposed system.
2. It mentions computer with new configuration requirement of proposed system.
3. It mentions new software requirements of proposed system.
4. It mentions new operating time of available system that is computer, hardware, software.
5. It mentions old as well as new facilities which will be provided by the proposed system.
6. It mentions all benefits of the system.

2. Operational feasibility

It is also called as behavioural feasibility. It finds out whether the new technology or proposed system will be suitable using three type of aspects; that are human, organizational and political aspects. It involves following steps :

1. It finds the ease of operation of proposed system compared to existing system.
2. It finds if the users of the system require any extra training ?
3. It finds out whether the user or customer of the system requires extra training or not ?
4. If it requires any retraining then it is accepted by user as well as customer or not ?
5. It finds if any job reconstruction is required or not ?
6. It finds if this reconstruction of the job is accepted in organization ?
7. It also finds if it is acceptable then what should be the skill sets of that job.
8. Watches the feelings of the customers as well as users.
9. It should provide right and accurate information to user or customers at right place as well as at right time.

→ 3. Economical feasibility

Here, steering committee finds total cost and all benefits as well as expected savings of the proposed system.

There are two types of costs – one time cost and recurring costs.

One time cost involves :

1. Feasibility study cost.
2. Cost required to convert existing system into proposed system.
3. Cost of hardware's, OS, application software.
4. Technical experts consulting costs.
5. Cost of training.
6. Cost of Documentation

Recurring cost involves following :

1. Cost involves in purchase or rental of equipment.
2. Cost of phones and mobiles Communication equipment.
3. Cost of Personnel search, hiring, staffing.
4. Cost of Salaries of employee's.
5. Cost of supplier's.
6. Cost of maintenance of equipment.

Step 2 : System Analysis

This phase of the water fall model is nothing but complete understanding of all important facts of the business using preliminary investigation. It involves following steps :

1. It is nothing but study of all components of the system as well as inter relation between all components of the system and relation between components and environment.
2. It determines what is to be done in the organization.
3. It finds the procedures of how to do that.
4. Which is input data ?
5. What is the procedure through which inputs are to be converted into output ?
6. When transaction should occur on the data.
7. When problem arises, determine the solutions to solve it and what are the reasons of those problems.

Objectives of System Analysis: The main roles of the system analysis are :

1. Define the system.
2. Divide the system into smaller parts.
3. Finds all nature, function and inter-relationship of various parts of the systems.
4. If the system is not analyzed properly then there may be problem in the Preliminary investigation phase.

Step 3 : System Design

The main objective of this phase of waterfall model is to design proposed system using all information collected from preliminary investigation and directed by the system analyst. This is very challenging phase. It includes following steps :

1. Design of all types of inputs of proposed system.
2. Design of all types of outputs of proposed system.

3. Design of the procedures which convert input to output.
4. Design of the flow of information.
5. Design of the information which is required to store within a files and data bases Volumes.
6. Design of collection of inputs using forms (Manual forms).
7. Design in terms of program specification i.e. logical design.
8. It determines the hardware cost, hardware capability.
9. It determines the speed of software.
10. It determines error rates, and other performance characteristics are also specified.
11. It also considers the changes to be made in the organizational structure of the firm in design.
12. This phase also designs standards for testing, documentation.

Generally traditional tools are used for the designing of the procedures that are as follows :

- Flowcharts
- Algorithms
- IPO (i.e. Input Processing and output) and HIPO (Hierarchy of IPO) charts
- Decision tables
- Data Flow diagrams

1. If system design phase is facing problem during the design then first go back to the system analysis phase and redesign the system but if problem is not solved then go for preliminary investigation.
2. If System design phase produces all expected results then it goes to next phase.

Step 4 : Coding

This phase is just implementing the design in to programming language that means it actually develops the proposed system. It involves the following steps :

1. It first of all, finds out the best suitable programming language that is suitable for the design as well as also suitable in the organization.
2. It accepts design and break system modules into smaller programs.
3. It develops or writes program for each part in selected programming language.
4. Prepares documentation that means add necessary comment lines wherever necessary within a program.
5. Now it combines all small programs together and builds one big program.
6. If any problem occurs during the coding phase then waterfall model tries to solve it by repeating system design phase:
   a) If that problem does not get solved then waterfall model repeats system analysis phase and system design phase.
   b) If that problem does not get solved then waterfall model repeats from first phase preliminary phase through system analysis phase and system design phase.
7. If coding phase produces all expected result then it goes to next phase.

Step 5 : System Testing

This phase includes the testing of the code or programs developed by the coding phase. This includes following steps :

1. First of all, it finds out all possible expected results (i.e. output data) for the set of input data.
2. It also checks the validity of the input data as well as checks expected output data.
3. It finds out all wrong results and immediately tries to correct it by repeating coding phase.
4. It finds the speed of functions using special codes.
5. It determines whether each program can perform the intended tasks or not?

6. It checks result by test data.
7. It checks the logic of the individual programs.
8. It checks interfaces between various programs.
9. It checks quality of code in terms of speed and space.
10. It checks whether system has produced correct and desired results which lead to designated goals.
11. If testing does not produce expected result then waterfall model tries to solve it by repeating system design phase and coding.
    a) If that problem does not get solved then waterfall model repeats system analysis phase through system design phase and coding.
    b) If that problem does not get solved then waterfall model repeats from first phase i.e. preliminary phase through system analysis phase, system design phase and coding.
12. If testing phase produces all expected results then it goes to next phase.

Step 6 : System Implementation

System Implementation is not creative process but it is some what difficult task. This phase has two parts - *implementation* and *evaluation* of the system.

☞ Implementation

There are two ways of implementation. Those are as follows :

1. Implement proposed system with existing old system and find out performance of the both systems and slowly replace new system with older one.
2. Totally replace old system with proposed new system.

Risk factor of second type of implementation is more as compare to first one. Second step needs strict evaluation.

Both types of implementation consist of following steps :

1. It prepares site for new proposed system.
2. It installs required hardware within a system.
3. It installs required software in a system.
4. It installs a developed code i.e. programs in a system.
5. It prepares training program for user of the proposed system as well as customers of the system.
6. It prepares user manual which includes all the steps which give guidance to user.
7. It gives training to all types of user of proposed system.
8. Observe the system when users of system are using it.
9. If users are facing any problems regarding the new system, it tries to find out exact phase from where root cause of the problem starts, and accordingly starts waterfall model.

☞ *Evaluation*

Evaluation is nothing but feed back for the system. It is very essential check point of the system which is the process of verifying the capability of a system. It continuously evaluates and checks whether proposed system is meeting the objectives or not. It includes :

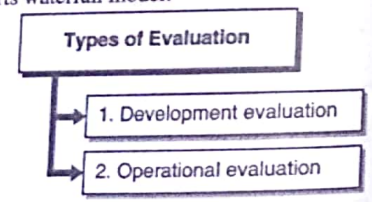

Fig. C2.2 : Types of Evaluation

➜ ***1. Development evaluation***

- *It checks whether the system is developed within time.*
- *It checks whether the system is developed within the budget.*

- System is assed by development methods and tools.

➜ **2. Operational evaluation**

- It checks response time of proposed system.
- It checks whether it is really easy to use or not?
- It checks accuracy of computations (It is seen in testing also).
- It checks storage capacity.
- It checks reliability.
- It checks functioning of the existing system.
- Collects necessary feedback from users.
- It finds all benefits of the proposed system.
- Collects information of attitude of different persons regarding proposed system.
- It evaluates cost, time and effort taken for the overall project.

Step 7 : System Maintenance

Maintenance is the process, in which it finds out essential changes (i.e. new trends) of the market or business or to correct some errors and tries to implement it in the existing system. There are usually three types of maintenance, that are :

1. **Correction :** Some times, proposed system has few types of errors; and it is the duty of software engineer to correct it as soon as it is encountered by the user. Generally there are four types of errors, that are as follows :
   - Minor changes in the processing logic.
   - Errors detected during the processing.
   - Revisions of the formats for data inputs.
   - Revisions of the formats of the reports.

   These errors can be corrected by repeating waterfall model from coding phase through testing, implementation and maintenance.
2. **Adaptation :** Some times, our proposed system is executable on Windows environment, but somebody wants to run it in LINUX environment, or some other operating system. Then we are required to design our proposed system from third phase that is from System Design phase.

   This is actually error free system. It is good so other people also want same system in
3. **Enhancement :** Because of new technology and business competition, organization needs to imply or to add new functions or additional capabilities to the proposed system.

   After some time, people think that some techniques may be used in the system so some additional features can also be added into it. Sometimes, new hardware is required to add some extra features.

   For enhancement it may repeat whole system or may repeat it from design phase or some times from coding phase.

☞ Advantages of waterfall model

1. It defines very first software development process.
2. The product of waterfall model always defines all constraints of the organization.
3. It always produces a good quality product in terms of space and time.

☞ Disadvantages of waterfall model

There are some disadvantages of the waterfall models that are as follows :

1. Real products rarely follow this sequential flow.

2. Because of iteration, changes can cause confusion as the project team proceeds.
3. It is very difficult for customer to state all the requirements in one time.
4. Many projects face this uncertainty at beginning only , so it is very difficult to design next phases.
5. Time span required for each phase could not be specified.
6. Naturally project requires more time.
7. Project becomes lengthy also.
8. Customer should have patience.

To overcome these drawbacks of waterfall model, a new model was designed known as ***'Incremental process model'***. Incremental process model has two process models 1) Incremental Model and 2) Rapid Application Development model.

Syllabus Topic : Incremental Process Models

2.4 Incremental Process Models

Method

- As soon as customer comes to software engineer and gives requisition of new project to software engineer; he starts to collect information from user to start preliminary investigation.
- He immediately increments towards second phase and third phase and starts to analyze and design the system.
- He starts to convert design into coding i.e. as time or calendar grows, some functions of software are also progressed towards completion.
- When testing of first requirement goes towards completion stage, customer comes with second requirement. Immediately he starts to collect all essential information for second increment i.e. preliminary investigation of the second requirement in parallel.
- He starts implementing first requirement i.e. module 1 as well as starts designing the second requirement i.e. module 2.
- Maintenance phase of first module is continuing then simultaneously coding of second module is starting as well as customer may ask for third requirement and he increments for the third incremental model.

This is explained in following graph ;

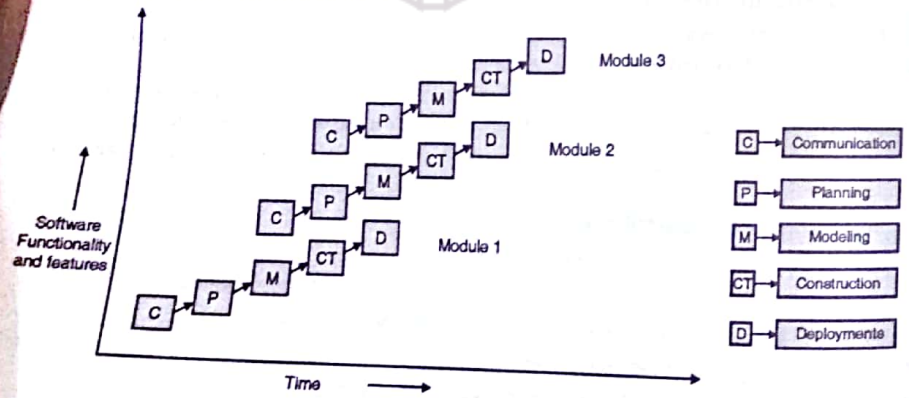

Fig. 2.4.1 : Incremental model

Example of Incremental Model

The software word processor is designed using incremental model. It is designed as follows :

1. Customer comes to software engineer and gives requisition of new project of a **word processor** to software engineer.
2. He starts to collect information from customer. Customer tells him to design file management system with all editing functions and document production functions.
3. Immediately software project team starts preliminary investigation to decide the plan.
4. Software engineer immediately increments towards second phase and third phase and starts to analyze and design the system. When third phase of first increment leads towards completion state then customer comes with second requirement i.e. about more sophisticated editing capabilities like copy, paste, and find capabilities. He starts preliminary investigation of the second increment in parallel.

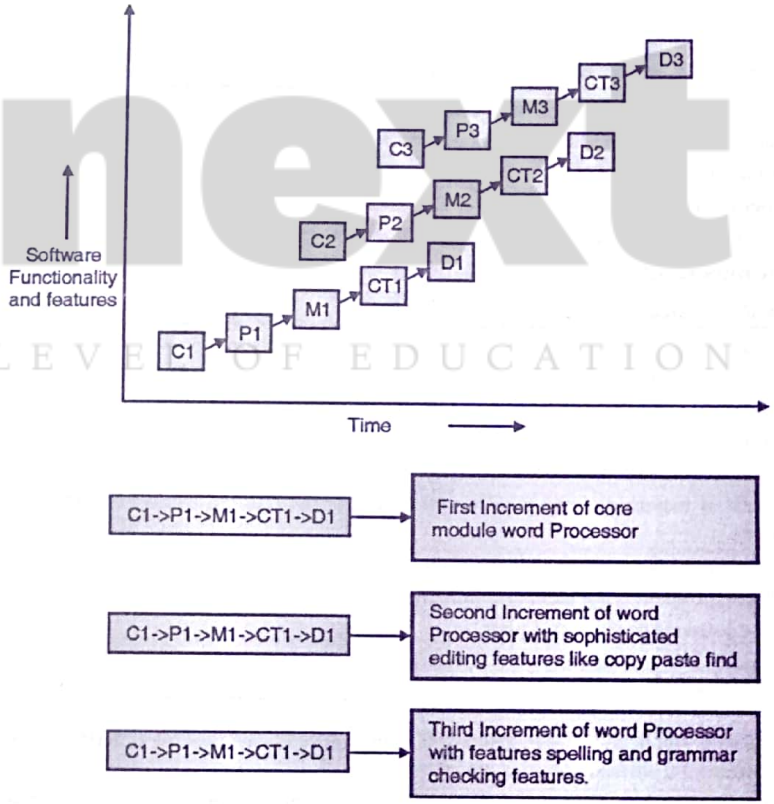

Fig. 2.4.2 : Example of incremental model

5. He starts to convert design into coding of first increments as well as he designs plan and objectives of the second increment for these sophisticated editing capabilities.
6. He starts implementing first requirement i.e. core module as well as it starts design of second requirement i.e. second module.
7. Maintenance phase of first module is continuing then simultaneously coding of second module is starting as well as customer may ask for third requirement that is spelling and grammar checking feature and it starts for the third incremental model.

This is explained in Fig. 2.4.2.

Method used in Incremental model

1. Generally in incremental model, the first increment is a core product which includes maximum basic requirements.
2. Second increments and third increments include all supplementary features (known and unknown features) which increase additional features with more functionality in a product.
3. This process is repeated till the completion of the final product.

Advantages of Incremental model

Q. List the advantages of Incremental model.

1. Preliminary investigation time is very small or reduced.
2. It requires very less time as compare to waterfall model
3. It requires less number of staff to develop the system.
4. Customer is satisfied because of quick development of the new system.
5. If system has many increments its functionality also increases; thus, the product has many features.

Disadvantages of the Incremental model

Q. List the drawbacks of Incremental model.

1. All the tasks are not decided in first phase so there may be some problem with designing phase. That means, overall design of the system is not so good.
2. It may produce software, which requires large space in memory.
3. Some times, speed of software is also slow.
4. Delivery date of the product could not be decided.
5. New levels of increments may require new hardware because old hardware is involved with old increments.
6. Some times, it produces same code in different module because of partial functionality.
7. Some times, problem is not identified properly.
8. Quality of software or product is poor.

2.5 RAD Model

RAD is Rapid Application Development. It is high speed adaptation of waterfall model. It is the example of incremental software process model.

The main disadvantage of incremental model is - the quality of product is very poor as well as it also requires more time to develop. So this model is only designed to produce a good quality product in very short duration of time.

Method

1. It also adapts generic framework activities like waterfall model that means five phases.
2. This allots sufficient time for study of the existing system.
3. It accepts sufficient number of staff for development of the system.
4. Staff is distributed into various teams (i.e. senior programmer junior programmer ,team leader, project leader etc).
5. Communication phase is used to understand business problem as well as it tries to identify all type of problems and gather all type of information.
6. Planning is essential because it divides work into manageable different parts and distributed among the various teams.
7. Modeling will be accomplished by different teams simultaneously. It includes three different phases. Business modeling, Data modeling and process modeling.
   - Business modeling collects all essential information about the product from market view or from the business point of view.
   - Data modeling shows the flow of data using some techniques like DFD.
   - Process modeling performs each process which handles a flow of data. It decides activities performed in each process.

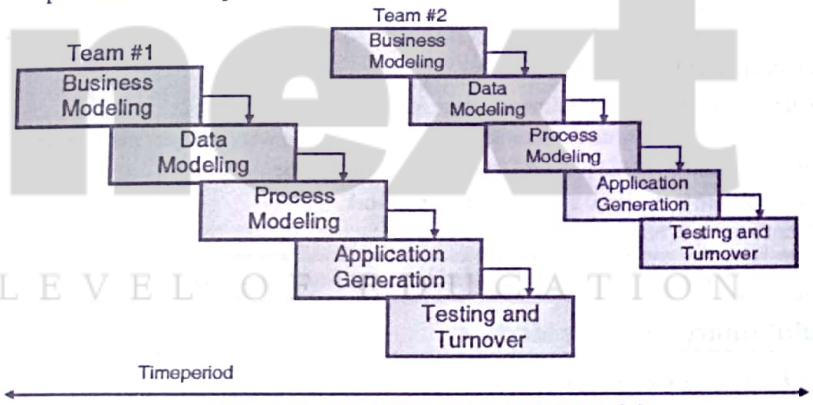

Fig. 2.5.1 : Modeling phase of RAD Model

   - Application Generation is about using a set of automated tools to facilitate the construction of the software say for example, the 4th GL techniques.
   - Testing and Turn over : Many of the components that will be used in the development of the proposed system might have already been tested since RAD focuses on reuse which reduces overall testing time. It is needed to concentrate on testing the new components.

Once modeling activity of each team is over, it immediately starts construction phase.

8. In construction phase, each team develops the code of our product as well as performs testing. It reuses old software components and develops new functions which are called as automatic code generation.
9. Deployment collects each code from different team and clubs them into single project implement it and if required then perform iteration among the previous phases.

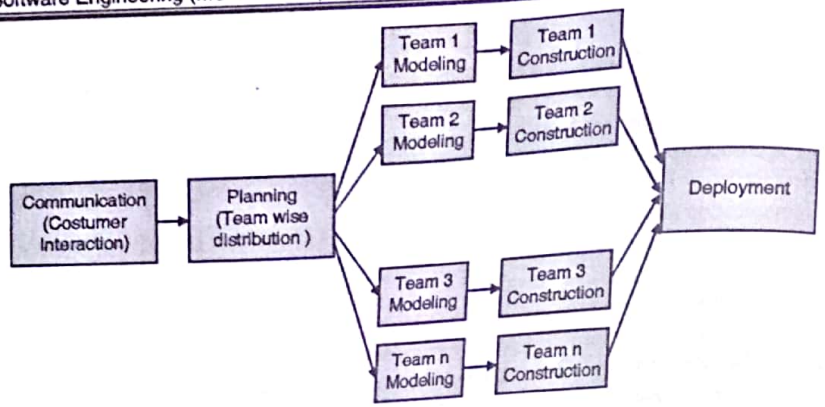

Fig. 2.5.2 : RAD model

Advantages of RAD model

1. It requires very less time to develop the product.
2. Planning and Design is performed before construction so it is not lengthy.
3. Product may be produced before its delivery date.

Disadvantages of RAD model

1. Large projects require sufficient or more human resources.
2. If developers and customers do not interact with each other then whole project may elapse.
3. If system is to be modularized properly then RAD is problematic.
4. If high performance is the Issue then RAD does not work.
5. It also doesn't work in high technical risk.

Syllabus Topic : Evolutionary Process Models

2.6 Evolutionary Process Models

The main drawbacks of Incremental process model are :

1. Overall design of the system is not so good.
2. It occupies large space in memory.
3. Some time speed of software is also slow.
4. Delivery date of the product is not decide.
5. Some times, problem is not identified properly.
6. Quality of software or product is poor.
7. Lack of interaction between developers and customers.
8. Poor modularization.
9. Doesn't produce high performance.
10. It is also not work in high technical risk.
11. Not suitable for complex product

These drawbacks are overcome in Evolutionary process model.

Main objectives of Evolutionary Process model are :

1. It should be suitable for complex system.
2. As time grows business also changes and product requirement may change during its development or construction phase.
3. It makes straight line path between customer requirement, business requirement and unrealistic product.

4. Evolutionary products should be iterative.

Though Evolutionary process model provides these advantages, it also has some issues.

Problems faced while using Evolutionary process models

1. Business competition makes product unrealistic
2. Tight market deadlines.
3. During development, product extension may be possible.
4. Software engineers must be flexible to requirements of customer and business changes.

2.6.1 Prototyping

This is the method or model of evolutionary process model. When customer tells his requirements, he is not aware about the development process, detailed input processing and also developer may be unsure of efficiency of the algorithm and adaptability of operating system, then prototype is the first step which is a best approach. This is explained in Fig. 2.6.1.

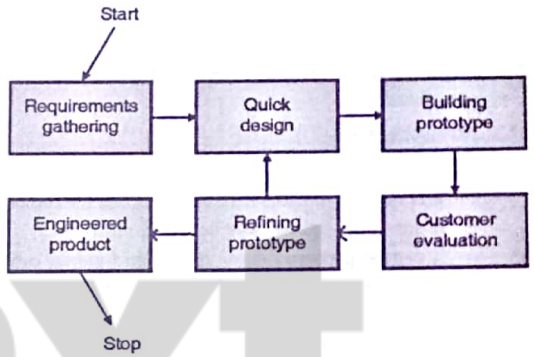

Fig. 2.6.1 : Prototyping model

Method

1. It begins with ***requirement gathering*** phase. Here customer and software engineer meet and define the overall objective of the product, It includes
   - Problem identification
   - Requirement analysis
   - Outline areas where further definition is mandatory

   According to overall objective of the product, software engineer makes planning of the design quickly.
2. ***Quick design*** focuses on the representation of human interface i.e. those aspects that are visible to the end user and customer. It uses different tools for design. Then, the analysts estimate a prototyping cost and inform about it to the management.
3. ***Building Prototype*** : Depending upon the quick design, construction of prototype is done that includes the software programs having the following capabilities :
   - accept input,
   - validate data,
   - perform calculations,
   - interact with files
   - produce outputs
   - Application generators

   Such a prototype can be built using several tools such as :

Screen generators :

- Input data screens with data validation are prepared.
- Meaningful prompt messages with each data entry
- Output screens are prepared.

- Table should have column headings and row heading.
- Labels, Messages, Colours, Fonts etc are decided.

☞ **Report generators :**

It shows output according to users requirement where reports are made from records extracted from data base.

4. ***Customer Evaluation*** **:** The built prototype is then evaluated by the customers or end-users to find if any changes are required. If so, then the requirements of the proposed software are refined.
5. ***Refining Prototype*** : The prototype is again refined based upon the feedback of the end-users about what they need and what they don't expect. This process is repeated till both the end users and the developers feel that all the requirements are fulfilled in the software and thus there is no need of refinement.
6. ***Engineered Prototype*** **:** This completed product is then given to the customer as per the specification.

☞ **Advantages**

- It is simple and an iterative process.
- It is revised to satisfy the needs of the customer.
- It does not require more cost to build.
- It can be prepared using pencil and paper, or computer software like screen generators, report generators and application generators.
- It saves time of development.
- It develops the product through ongoing communication with the user.
- So easily finds user friendly services and non user friendly services.
- Readymade tools are used, to development of code.
- Product is to be delivered within proper time.
- It provides training to user.
- Prototypes are also used for testing.

☞ **Disadvantages**

- It does not contain all features or perform all the necessary functions of the final system.
- Customer/user evaluates the module/prototype and they suggest addition and modifications so it may become lengthy.
- So may produce poor level quality product.
- Developer may compromise with operating system.
- It is not linear.
- It is not finding the risk of the project.

Q. State the difference between :Classic life cycle model and Prototyping model.

Table 2.6.1 : Comparison between Classic life cycle and Prototyping model

Sr. No.	Classic Life Cycle Model	Prototyping Model
1.	It is not an iterative process.	It is an evolutionary i.e. iterative process model
2.	During development, mostly product extension or addition of new requirements is not encouraged	During development, product extension may be possible.

Sr. No.	Classic Life Cycle Model	Prototyping Model
3.	Testing is not conducted from the initial phases of the SDLC – therefore when errors are detected after the coding phase it costs a lot for refining the product.	Building and refining of product is involved before the actual engineering process – therefore, much cost is not involved
4.	Included Manual coding	Uses readymade tools such as CASE tools for coding
5.	Communication with the users is done at the start i.e. while requirement gathering and then while testing phase. No ongoing communication with the users is encouraged	develops the product through ongoing communication with the user

2.6.2 The Spiral Model

Q. What does the radius imply in Spiral model ?

The Spiral model is also an evolutionary software process model that couples the iterative nature of prototyping. It is proposed by Boehm. The main objectives of spiral model are ;

- It provides controlled and systematic aspects of the linear sequential model.
- It uses potential of rapid development of incremental versions of the software.
- It finds all risks in the project.
- It finds future risks also.
- It is explained in following diagram.

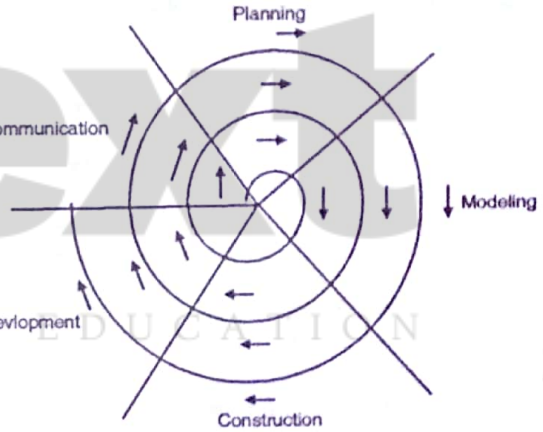

Fig. 2.6.2 : Basic spiral model

☞ **Method**

Requirement Gathering includes;

- Interaction with customer and user.
- Problem identification regarding existing system.
- Deciding product specification.
- Deciding objectives of the system.

Planning decides the project plan. It includes

- Cost measurement of the system.
- Deciding schedule of the system.
- Collecting feedback from the user.
- Adjusted and planned number of iterations required to complete software.

Risk Analysis task is involved in each of the iterations of the project. It includes;

- Identification of risk areas
- Identification of technical as well as managerial risks
- Measuring cost related risk
- Efforts are taken to resolve the risk

Engineering phase immediately designs the prototype of the proposed system depending upon the objectives decided in the planning phase. It also converts design into coding and does the needful testing. It then shifts to *deployment phase* which includes :

- Implementing the developed system
- Do the Verification
- Gives training to users
- Collect feedback

Customer Evaluation : The product is then evaluated by the customers or end-users to find if any changes are required. If so, then the above phases are repeated again in loops.

Q. State the difference between : Waterfall Model and Spiral Model.

Table 2.6.2 : Comparison between Waterfall and Spiral Model

Sr. No.	Waterfall Model	Spiral Model
1.	Process flows from top to bottom like a flow of water from a hill to ground. Flow of water can't be reversed - similarly any new changes cannot be incorporated in the middle of the project development.	Best suitable for projects associated with risks.
2.	Process goes to the next phase only after the completion of the previous phase. Here, end user feedback is not taken into consideration for any change in SRS. This will result in restarting the work from beginning.	Each and every step goes through testing which makes it easy to recover any error and fix it then and there itself. In this model we don't have to start work from beginning.
3.	Purely a pre planned /strategic in nature.	Used to build a product which doesn't have adequate requirement gathering.
4.	Stresses more on requirement gathering.	focuses more on risk analysis which is not much considered in waterfall model.
5.	Customer feedback is not considered at every step of project development.	Customer feedback is considered at every step of project development.

☞ ***Advantages***

- *It develops large scale systems and software.*
- *Better understanding of developers and customers.*
- *It performs risk analysis from the point of view of technical aspects as well as managerial aspects.*
- *It follows systematic and stepwise approach of SDLC.*
- *Wherever needed, it uses prototyping approach during development.*

☞ ***Disadvantages***

- *It is not suitable for fixed budget development.*
- *It is not panacea.*

- It demands for expertise for risk assessment.
- If major risk is not covered then problem is never solved.

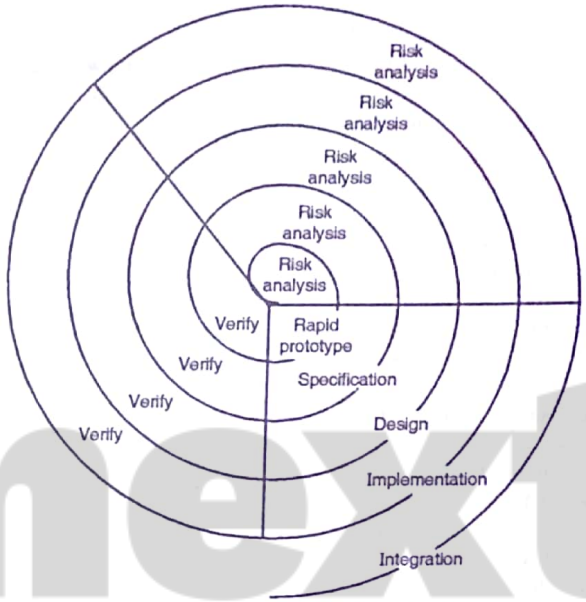

Fig. 2.6.3 : Spiral model used in industries

Now days, spiral view is to do risk analysis and prototyping that means to follow traditional framework then go for verification. If necessary goes for second iteration. This is well explained in Fig. 2.6.3.

Syllabus Topic : Concurrent Models

2.7 Concurrent Model

It is also called as a concurrent engineering. It also represents systematic framework activities with associated states. It is explained as follows;

- Every phase is associated with states.
- Every state is having some meaning full names.
- It is applicable for all types of the software development.
- It shows us exact state of the project.
- It supports parallel development of the project.
- It defines series of event transitions from state to state for each development activity.
- It defines network of activities, actions and tasks.
- It finds which activities or actions or tasks can be performed simultaneously.

☞ **Example of Modeling phase in Concurrent process model**

- It consists of 6 states that are ;

1. **Under development state :** When communication is over then it starts modeling and its state is Under development state.
2. **Awaiting changes :** It always collects requirements from the user. As soon as user enters requirements then the current state of the project is Awaiting changes state.
3. **Under revision :** It accepts changes from customer and starts working on it by revising the system. At this time, state of the system will be under revision state.
4. **Under review :** It collects necessary information from communication phase and Awaiting changes phase and study them.
5. **Base lined :** It design the system
6. **Done :** Once design is completed and there is no new requirement from user then the state of modeling is done that means, we can start construction phase.

- When communication is over, it starts modeling phase.
- It collects information of user from awaiting changes state.
- Then it revises the system and goes for under review state.
- Now prepare base line of design.
- Once design is prepared and there are no changes from customer then current state of modeling phase is done that means we can continue for new state that is construction state.

This is explained in Fig. 2.7.1.

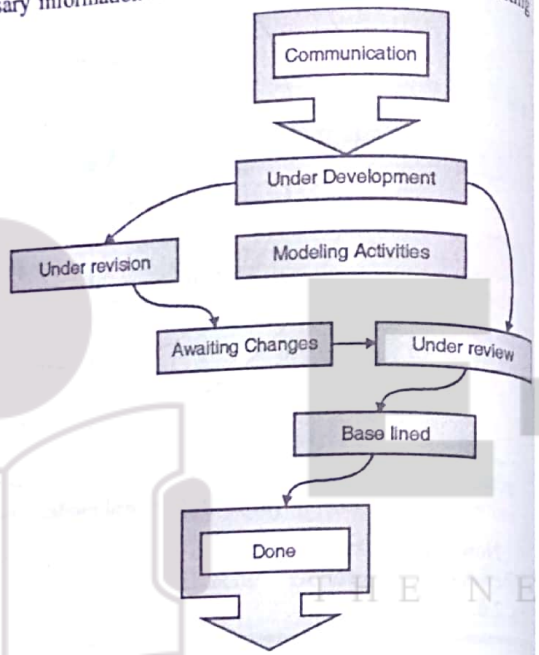

Fig. 2.7.1 : Concurrent development model

Syllabus Topic : Component-Based Development

2.8 Component-Based Development

- *Component-based development is about the **assembly of components**. It enables **Reuse**.*
- *Component-based development is a set of pre-build, standardized software components that are made available to fit in a specific architectural style for some or the other application domain.*
- *The application is then formed by assembling these available components, instead of assembling "discrete parts" of a conventional programming language.*

***Example :** when we purchase a "stereo system", we see that each component has been designed to fit in a specific architectural style - connections are standardized, communication protocol has been pre-established. Also, assembling of the components in this stereo system is easy because you don't have to build the system from scratch; instead you have all the components ready and you just have to connect them properly to each other. This is what 'CBSE' is all about.*

Design Principles

- Component-based development is *rapid assembly* and maintenance of component-based systems; where components have well-defined properties.
- A software component is a logically cohesive, loosely coupled module that denotes a single abstraction and can be ***reused*** in the system development.
- These components are ***independent*** of each other i.e. they do not interfere with each other because;
  - Every Component implementation is hidden from the other.
  - Communication between components is through well-defined interfaces.
- A software component is a service provider : The services offered by the component are made available through an interface and all component interactions take place through that interface.
- A software component is deployable only if it is self-contained and operates as a stand-alone entity on some component platform that implements the component model.

Advantages

- Increases competitiveness by;
  - Reducing the cost of software development.
  - Increasing software productivity.
- Limited human talent by

  Increasing software / person - It is possible by reuse of existing solutions, rather than invent them.
- The reuse of a component requires less time than the development of a new component. Therefore, systems can be built faster using CBSE process.

Component-based development Process

- Outlining the system requirements.
- Searching for components.
- Modifying the requirements according to available functionality in the components.
- Searching once again to find if there are better components that may meet the revised requirements.

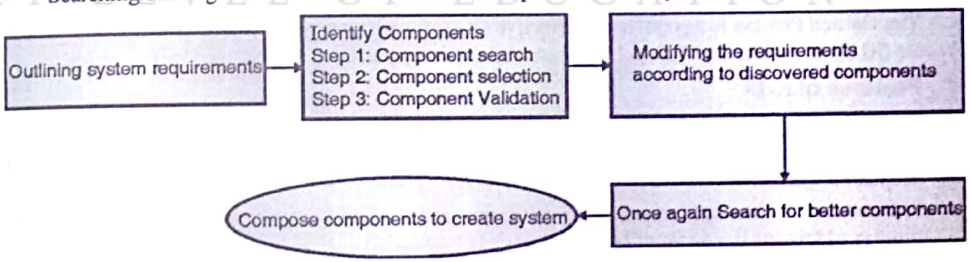

Fig. 2.8.1 : Component-based development process

Component Composition

- Composition is the process of assembling components to create a system.
- It involves integrating the components with each other and with the component infrastructure.
- Usually, 'glue code' is written to integrate the components.

Composition types

1. **Sequential composition :** The composed components are executed in a sequence.
2. **Hierarchical composition :** One component calls on the services of another.

3. **Additive composition :** The interfaces of two components are put together to create a new component.

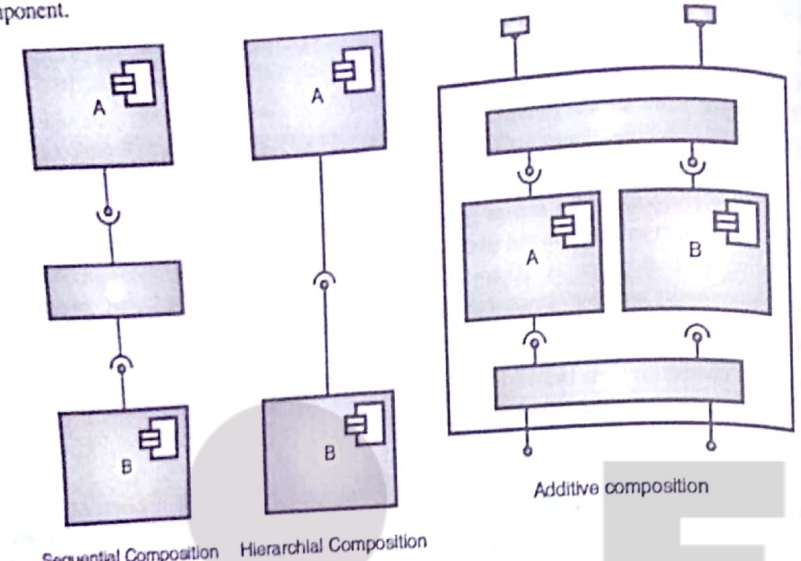

Fig. 2.8.2 : Composition types

Syllabus Topic : The Unified Process Phases

2.9 The Unified Process Phases

Q. State the purpose, advantages and drawbacks of RUP?

The Unified Process is a popular and effective OO software development process. Rational Unified Process (RUP) is modified at Rational Software and is widely practiced and adopted by industries.

☞ Features of RUP

- The most important feature/idea in RUP is *Iterative Development*. Iterative Development is sequentially expanding and refining a system through multiple iterations, using feedback and adaptation.
- RUP can be a lightweight process addressing the needs of small projects - to more comprehensive process addressing the needs of large projects.
- RUP does early and continuous documentation of the most urgent and the most probable risks by proper planning and keeping follow up. This helps in mitigating the risks at early phases of software development.
- RUP uses visualization methods like UML to build models to understand the complexity of the system.
- RUP uses *use-cases* as test cases which allows end-user documentation and helps in designing.

☞ RUP Phases

- ***RUP is divided into 4 phases of iterative development :***
- Each phase has iterations, each having the purpose of producing an executable piece of software. The duration of iteration may vary from phase to phase.

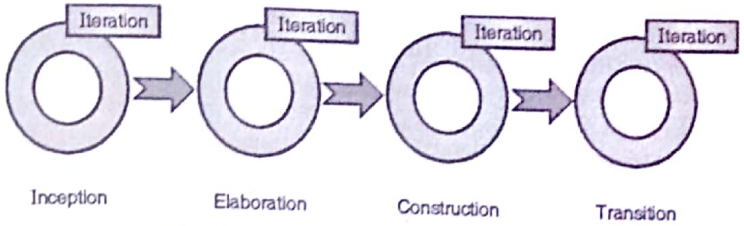

Fig. 2.9.1 : Four phases of Iterations in RUP

(I) Inception

Inception means start i.e. this is the point where the project is proposed.

Aim of Inception Phase : It constitutes of Business modeling i.e. Define the Problem, Scope of the System, and Initiate the Project.

☞ Purpose

- Define the problem : The objectives of the project are stated, so that the needs (requirements) of every stakeholder are considered.
- Define the scope of the system : The Scope and boundary conditions, acceptance criteria and some requirements are established. External entities (actors) with which the system will interact are identified and the nature of the interaction is defined on a high-level by identifying all use cases. It also includes identifying the business case i.e. identifying the success criteria, risk assessments and estimation of the resources needed and a phase plan showing dates of major milestones.
- Initiate the ***project*** : And then, you can start working on the project.

☞ Inception Outcome

- Vision document
- Initial use-case study (10%-20% complete)
- Initial business case
- Initial risk assessment
- Project plan
- Stakeholders decide whether to commence a full scale project or not.

(II) Elaboration

Elaborate means refinement (careful development).

Aim of Elaboration Phase : Detailed analysis of the problem resulting in the definition of an architectural foundation for the project. It constitutes of ***requirements, analysis*** and ***design*** phases.

☞ Purpose

- Analyze the problem
- Develop the project plan
- Remove the highest risk elements of the project.
- Gives you a *mile wide and inch deep* view i.e. little bit deeper view of the system.

Elaboration Outcome

- Use-case model is 80% complete.
- Additional requirements capturing the non functional requirements and requirements that are not associated with a specific use-case are identified.
- Description of the Software Architecture.
- An executable architectural prototype is developed.
- A revised risk list and revised business case is developed.
- A development plan for the whole project, including iterations and evaluation criteria for each iteration also specifying the process i.e. to be used.

(III) Construction

Construction means to build i.e. it is a manufacturing process.

Aim of Construction Phase : It constitutes of implementation i.e. detailed design and construction of source code.

Purpose

- Emphasize on managing resources and controlling operations to optimize costs, schedules and quality. This phase is broken into several iterations.
- All components and application features are developed and integrated into the product and tested.

Construction Outcome

- An executable product that is ready to put in the hands of the end users.
- The developed and ready to use software product
- A user manual
- Description of the current release.

(IV) Transition

Transition means delivery. The transition phase is the phase where the product is put in the hands of its end users.

Aim of Transition Phase : It constitutes of deployment i.e. delivery of the system to the user community. It involves issues of marketing, packaging, installing, configuring, supporting the user-community, making corrections, etc.

Purpose

- Deliver the software product to the user community.
- Issue new releases
- Perform beta-testing to validate new system against user expectations
- The system might run in parallel with a legacy system
- Roll-out the product to marketing, distribution, and sales team

Transition Outcome

- Achieving user self -supportability
- Achieving stakeholders' agreement that deployment is complete and consistent.

Six Best Practices of RUP

1. Develop iteratively

- Software must be developed in small increments and short iterations.
- An iterative process breaks a development cycle into a *sequence of 4 phases* each of which includes a *series of iterations.* (You can see this in Fig. 2.9.1.

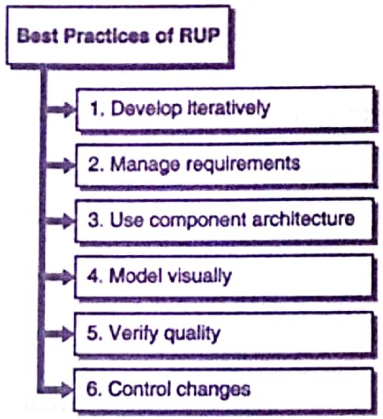

Fig. C2.3 : Best Practices of RUP

2. Manage requirements

- RUP allows accommodating requirement changes in system development strategy.
- Those requirements that change over time and those requirements that have greater impact on project goals are identified.
- It is a continuous process to identify requirements.
- Managing requirements include :
  - Elicit, organize (according to the priority), and document the required functionalities and constraints,
  - Evaluate the impact of changes and
  - Track and document the decisions.

3. Use component architecture

- The process focuses on early development and design of independent executable modules, prior to committing for full-scale development.
- Components that are most likely to change and components that can be re-used are identified and built.

4. Model visually

- Models must be built using visualization methods like UML, to understand the complexity of the system.
- This helps you to understand the different aspects of your software and see how the different elements of the system communicate with each other.
- Maintains uniformity between design and its implementation.
- Promotes unambiguous communication between developer and end user.

5. Verify quality

- Quality of the software is maintained by its frequent testing.
- Testing is done to remove defects at early stages, thus reducing the cost at later stages. In particular, high risk areas are tested more thoroughly.
- The software released at the end of every iteration, is tested and verified.
- Test cases are created based on use cases (and its scenario).
- Decisions are made on real test results.

6. Control changes

- Any changes to requirements must be managed and their effect on software should be tracked.
- All change control goes through the convener of the CCB (Change Control Board).

- Members of CCB can be representatives from different areas, say: test designer, project manager, system analyst or stakeholders.

Advantages of RUP

- RUP helps in addressing very early high risks areas.
- It allows change in the requirements as the project evolves.
- The main focus is quality of the software product.

Drawbacks of RUP

- It fails to provide any clear implementation guidelines.
- RUP leaves the tailoring entirely to the user.

Nine Workflows

There are *6 core* and *3 supporting* process workflows involved in the 4 phases of RUP, which represent a partitioning of all workers and activities into logical groups. You can see these total 9 workflows. We will study about it in this section.

1. **Business Modelling**
   - It is about modelling the ***business context*** and the ***scope*** of your system using use cases. This workflow is part of ***Inception*** and ***Elaboration*** phase.
   - **Active workers during this process :** Business process analyst, Business Designer and Business Model Reviewer.
2. **Understanding Requirements**
   - The purpose of this workflow is to elicit the requirements for the project, including their identification, modeling, and documentation.
   - The goal of this process is the Software Requirements Specification (SRS) which encompasses the captured requirements. This describes what the system should do and allows the developers and the customer to agree on that description.
   - Active **workers during this process :** System analyst and software architect.

   Works of system Analyst :
   - Elicit stakeholders requests
   - Develops requirements management plan
   - Finds actors and use-cases

   Works of software Architect :
   - Prioritize use-cases
   - Review requirements
3. **Analysis and Design**
   - The purpose of this workflow is to ***develop a robust architecture*** for the system based on the requirements, to transform the requirements into a design, and to highlight the major risks (if any).
   - The goal of Analysis is to determine the risks, the stability of the product and expenses of resources.
   - The goal of Design is to show *how* the system will be understood in the implementation phase.
   - Active workers during this process : Architect and Designer

   Works of Architect : Architectural analysis

   Works of Designer :

   - Use-case analysis
   - Use-case design
   - Class design
4. **Implementation**
   - The purpose of implementation is to ***code and test*** the system.
   - The goal of implementation is to develop ready to execute module independent of other modules.
   - **Active workers during this process :** Implementer, integrator and code reviewer.

   Works of Implementer :
   - Implement a module
   - Fix a defect
   - Perform unit testing

   Works of Integrator :
   - Plan system integration
   - Plan subsystem integration
   - Integrate subsystem
   - Integrate system

   Works of Code Reviewer :
   - Review code
5. **Testing**
   - The purpose of the test workflow is to design test case procedures and other verification methods.
   - The goal of test process is to design and execute test cases for the system inorder to eliminate defects.
   - Active workers during this process : Test designer, tester

   Works of Test Designer :
   - Plan test
   - Design test
   - Implement test
   - Evaluate test

   Works of Tester :
   - Execute test
6. **Deployment**
   - The purpose of deploy is to install the software at the end-user.
   - The goal of deployment is to deliver the system to the user community. It involves issues of marketing, packaging, installing, configuring, supporting the user-community, making corrections, etc.
   - **Active workers during this process:** Deployment manager and technical writer.

   Works of Deployment Manager :
   - Develop deployment plan
   - Manage acceptance test
   - Define bill of materials

Works of Technical Writer :

- Write release notes
- Develop support materials

7. **Configuration and Change Management**
   - The purpose of Configuration and Change Management (CM) is to monitor and administrate changes in the project work so that they are consistent with the requirement.
   - The goal of CM is to release and control the version (changes made) of the system.
   - **Active workers during this process:** Configuration Manager and Change Manager.

Works of Configuration Manager :

- Set up CM environment
- Establish CM policies
- Write CM plan

Works of Change Manager :

- Establish change control process
- Review change request

8. **Project management**
   - The purpose of software project management is balancing the competing objectives, managing risk, and overcoming constraints so as to deliver a successful product which meets the needs of both customers and users.
   - **Active workers during this process:** Project manager and Project Reviewer.

Works of Project Manager

- Initiate project
- Develop iteration plan
- Develop quality assurance plan
- Monitor project status
- Schedule and assign work
- Report status
- Handle exceptions and problems

Works of Project Reviewer :

- Project approval review
- Project planning review
- Iteration plan review
- Iteration evaluation criteria review

9. **Environment**
   - This provides the software development environment –both processes and tools - that are needed to support the development team.
   - **Active workers during this process:** Process Engineer, Software architect and Tool Specialist.

Works of Process Engineer :

- Development case
- Project specific templates

Works of Software Architect :

- Design guidelines
- Programming guidelines

Works of Tool Specialist :

- Tool guidelines
- Tools

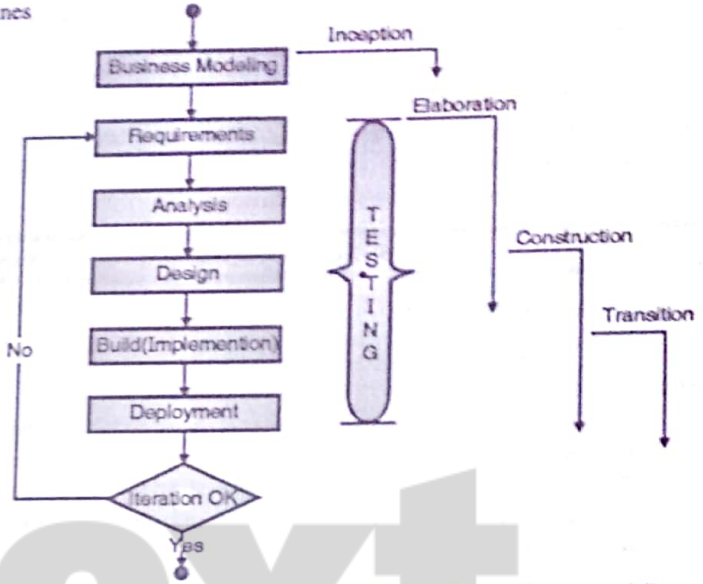

Fig. 2.9.2 : The 4 phases of RUP and their 6 core workflows

Syllabus Topic : Agile Development - Agility

2.10 Agile Development

2.10.1 Agility

Q. What is agility ?

Agility is the unending process, which always accepts specific requirement of the product from the customers or end user then it checks whether the requirements are growth oriented or not ? If it is growth oriented then it aggressively changes the existing system.

The Agile alliance [AG103] defines 12 principles to achieve agility

1. Customer satisfaction by continuous delivery of the software.
2. Always accept changes in requirement though it is late in development.
3. Deliver working software frequently in shorter time span.
4. Business people and developers must work together during complete development of the project.
5. Develop the project from the software project team. It includes;
   - Motivate each individual of the project.
   - Provide then necessary resources and environment to build the project.
   - Trust them to get the job done.
6. Do the face to face conversation, if essential or urgent.
7. Primary measure of the progress is the working software.

- Incremental development is supported through small frequent releases of the system. Where requirements of these increments depend on customer stories and process planning.
- New build of the software is accepted only if all tests execute successfully.
- Change in a system is supported through development, through test first development, as well as through continuous integration.
- XP follows basic framework activities which consist of planning, designing, coding and testing.
- Each activity consists of set of rules and regulation.
- XP process is illustrated in Fig. 2.10.1.

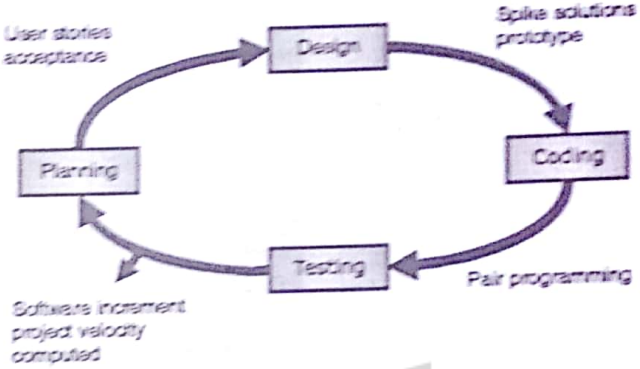

Fig. 2.10.1 : Basic XP process model

- Customer or representative of customer (i.e. may be end user) is involved in through out development process.
  - Customer or representative of customer and end users are the part of agile team so they should be active during the development process.
  - They should be active in defining all requirements of the project.
  - They should be active during the planning also.
  - They may passive during designing and coding.
  - But they are again active during testing if they are accepting the test results then project is to be implemented in actual office site for implementation and maintenance purpose.
- Coding supports pair programming which includes;
  - Collective ownership of the programs
  - Sustainable development of the program
  - It does not involve excessively long working hours.
- Maintaining Simplicity through
  - Simple designing that do not anticipate future enhancement.
  - Constant refactoring to improved quality of code.

Table 2.10.1 : Experts view regarding extreme programming

Experts View
"Extreme Programming is a discipline of software development based on values of simplicity, communication, feedback and courage." **Ron Jeffries**

- The detailed phase of XP process model is

1. Agile process promotes sustainable development. (That means somebody from agile team doing a good task is helpful for development should get promotions).
2. Keeps Continues track to technical excellence and best design increases agility.
3. It collects information about amount of work not done and tries to find out reasons of that.
4. Agile team collects requirement, and fulfil them using best architectures.
5. Continuous thinking about how the performance of team can be increased.

Agility can be applied to any software process.

Syllabus Topic : Agile Process

2.10.2 Agile Process

An Agile software process is characterized in a manner that addresses 3 key assumptions of software project.

1. It is difficult to predict which requirement is changing and which is not changing?
2. Some type of software, design and construction phase are related with each other, so again it is difficult to predict exactly how much time required to design process.
3. Time required for analysis, design and construction are not predictable.

This unpredictability can be managed by;

- Agile software development adapts incrementally.
- It accepts feedback from the customer.
- If requires then use prototype approach.

Agile Process models

- The main objective of agile process model it to provide best quality software in allotted time span.
- It follows conventional approach.
- It follows philosophy and guidelines suggested in manifesto for agile software development.
- There are many Agile Process models that are :
  - XP
  - ASD
  - DSDM
  - Scrum
  - Crystal
  - FDD
  - AM.

Syllabus Topic : Extreme Programming

2.10.3 Extreme Programming

Q. Explain XP in detail.

Extreme programming is most widely used Agile development model; it is also called as XP. It was popular in late 1980's. Idea of this extended programming is developed by the Kent Beck, Jeffries, Beck and Fowler. Extended programming is associated with ideas and methods.

- Extended programming (XP) uses all techniques of object oriented paradigm during the development process.
- XP follows an extreme approach to iterative development.
- New versions of software may be built several times per day and increments are delivered to customers roughly every two weeks.

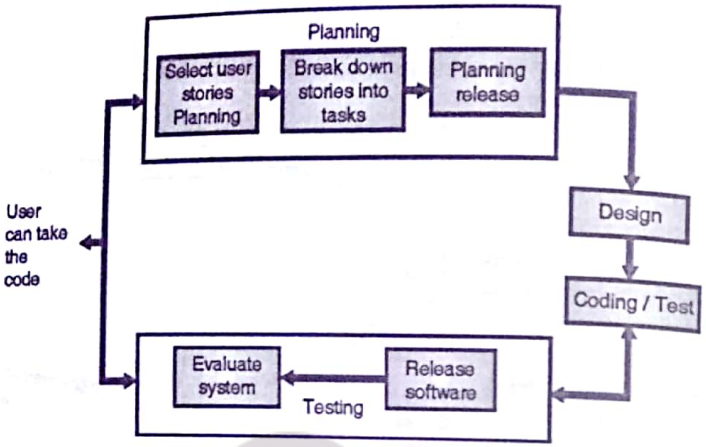

Fig. 2.10.2 : Detailed view of XP process model

I. Planning

Planning activity means gathering the information from customer and plans the project. It consists of :

- Planning activity is begins with creation of set of stories, which consist of features and functionality of the software to be built.
- Each story is written by the customer, which is placed on an index card or story card. Where each index card having;
  - A value assigned by customer which is nothing but priority.
  - Stories (i.e. features and functions) based on business.
  - The value of story may also depend on presence of another story.
  - Example of story card is given below :

Table 2.10.2 : Story card or index card

Downloading and printing an article
First you select article that you want from displayed list. You then have to tell the system how you will pay for it-this can be either from subscription, through company account or by credit card. After this, you get copy write from the system to fill in. When you have submitted this, the article you want is downloaded on to your computer. You choose the printer and a copy of article is printed. You tell the system printing has been successful. If the article is print only article, you cant keep PDF version, so it is automatically deleted from your computer.
Cost of Story : 1 Week

- Other members of XP team assess the each story and assigns cost to each story card. Where,
  - Cost of each story card is nothing but total time span required to complete development of all tasks (i.e. functions and features) mentioned in the card.

  - This time span is measured in hours, days, or weeks.
- If story requires more than three development weeks then customer is asked to split story in to smaller stories and again priorities are assigned and costs are evaluated.
- New stories can be written at any time.
- Customers and XP team work together and to group stories to next increments of the software developed by the XP team.
- Once basic commitment or agreement (It includes delivery date with another project matter) is made up for a release, the XP team orders the stories that will be developed using three ways;
  1. All stories will be developed and implemented quickly using first in first out method.
  2. The story with highest value will be moved up in the schedule and implemented first.
  3. The riskiest stories will be moved up in the schedule and implemented first.
- First increment is called as a first project implementation. Similarly second iteration development is called as a second increment or second project implementation.
- After the delivery of first increment of the project XP team computes project velocity. Where project velocity can be find out as number of stories implemented during the first or one increment in some time span i.e. some few weeks.
- Project velocity is used for
  - Computing the delivery dates.
  - It finds whether over commitment is made or not? And if over commitment is occurs then immediately it again compute the exact delivery dates.
  - As development work proceeds, customer can add new stories.
  - Change the priorities of existing stories.
  - Split stories.
  - Delete stories.
- XP teams decides schedule of development according to project velocity and project priorities.

II. Design

XP design rigorously follows the KIS principles.

- KIS stands for Keep It Simple.
- Design gives guidelines for story writing i.e. nothing less, nothing more..
- The design of extra functioning is discouraged.
- These design guidelines should be followed in every software engineering method although there are times when sophisticated design notation and terminology may get in the way of simplicity.
- XP encourages the CRC (Class Representative Collaboration) cards.
- CRC identifies and organizes the object oriented classes that are relevant to the current software increment. That means it uses inheritance property of object oriented programming language.
- CRC card is nothing but to design work product produced.
- XP design phase converts each story card into CRC card.
- During the design of some story card it founds some difficulties. It follows operational prototyping approach to design complex part of that story card. This is called as a **spike solution.**
  - Spike solution is used to design complex part of the story card.
  - It uses operational prototype to design.
  - It is immediately implemented and evaluated from the end user.
  - So it reduces risk when true implementation is started.
  - It easily validates the original estimate.

- It encourages for refactoring.
  - Refactoring means constructing such a technique which is also used to design techniques.
  - It improves internal structure of the code.
  - It's disciplined way to clean up code.
  - This keeps code simple and maintainable.
  - Easy way to develop new code.
  - It minimizes bugs of the design as well as chance of introducing new bugs in the design.
  - Time required to create new design is very less.
  - Refactoring can be presented before, as well as after coding, that means continuous programming.

Table 2.10.3

Experts View

"Refactoring is the process of changing a software system in such a way that it does not alter the external behavior of the code yet improves the internal structure. It is disciplined way to clean up code [and modify/simplicity the internal design] that minimizes the chances of introducing bugs. In essence, when you refactor you are improving the design of the code after it has written."

Fowler

- Some times, work product is other than CRC card or spike solution that means they can be developed using procedure oriented programming language.
- A central notion of XP is that - design occurs both before and after coding.
- Construction activity provides XP teams guidance on how to improve the design.

III. Coding

Once stories are converted into CRC cards, then XP team moves towards code development. It develops the code using following steps.

- The team exercises on each story and creates a series of unit test.
  - This approach is similar to knowing the previous exam questions before we begin the study.
  - It makes studying easier by focusing attention only on asked questions. Developer is better able to focus on what must be implemented to pass the unit test.
  - In this way, developer can construct a code, which is ready for unit test.
- Once code is complete, it can be unit tested immediately;
  - Test is taken by developer itself;
    1. It checks validity of the all input data.
    2. Whether code is producing expected result or not?
    3. Calls customer or representative of customer or end user to test the output of constructed code.
  - Feedback is collected.
- The key concept of coding activity is pair programming.

Pair programming

XP recommends that two people work together at one computer workstation to create code for a story. It is very innovative practice to distribute story card among the pairs of programmer. They

actually sit together at same workstation to develop the software. Development does not always involve the same pair of people working together. This supports:

- Pairs are created in sucha away that every team member gets a chance to work with every other team member i.e. all team members may work with each other during the development process.
- Mechanism for real time problem solving.
- Mechanism for real time quality assurance.
- Keeps developers focused on the problem at hand.
- Each person of the team plays different role in development.
- One person may think about coding details of particular portion of the code.

  Other person ensures that coding standards are being followed and code that is generated will "fit" into the border design for the story.
- As pair programmers complete there work, the code they developed is integrated with the work of others.
- Sometimes, this is performed on daily basis by the integration team.
- Daily integration gives following advantages:
  - Avoids compatibility.
  - Avoids interfacing problems.
  - Provides smoke testing environment.
  - It early uncovers the errors.

Advantages of pair programming are :

It supports idea of common owner ship and responsibility for the system. It encourages;

- Egoless programming (proposed by Weinberg).
- Software is owned by the team.
- Individual is not held responsible for the problem.
- So individuals do not face that much stress.
- Collectively team has responsible to resolve these problems.

It acts as a informal review process because each line of the code is looked at by at least two people.

- Code inspections and review are very successful.
- Discovering high percentage of software errors.
- It is much cheaper inspection process than formal program inspection.
- Formal program inspections are time consuming but it is not much time consuming.

It helps supports refactoring, which is process of software improvements.

Drawbacks of pair programming

- Each story needs two programmers instead of one.
- If they are not comfortable with each other because of some human factors they will not be develop good software.

IV. Testing

Till now we know that there are two development strategy, Plan based development and Iterative development. Each development uses different testing techniques. Our Agile view process supports iterative development. XP spares more emphasis on than other Agile methods on the testing process.

- Testing procedure is central to XP where approach has been developed that reduces the same functions development in to the existing software.
- The XP Team assesses each story and breaks it into many tasks.

- Each task represents discrete features of the system.
- Unit test for each task are designed.
- Each task can generate one or more unit tests.
- Each unit test describes the test cases.
- The role of customer in the testing process is to help in developing acceptance tests for the st that are to be implemented in the next release of the system.
- Acceptance testing is the process where the system is tested using customer data to check th meets the customer's real needs.
- XP supports incremental testing.

Acceptance test of each story involves

- Making several document selections.
- Paying for them in different ways.
- Printing them on different printers.
- Data validity test.
- Expect output tests.
- Series of different tests rather than single tests.
- Writing code for testing.

The Key advantages of testing in XP are:

1. Test Driven Development (TDD)
2. Incremental test development from scenarios.
3. User involvement in test development and validation.
4. The use of automated tests.

Q. What is Test First Development ?

Test Driven Development (TDD also known as Test First Development) : It is the mo important innovations in XP. Writing tests first implicitly defines both interfaces and a specification b behavior for the functionality being developed. It consists of :

- Test first development is the technique, where test code is written first, and then code is generate for that test.
- This test code includes :
  - Once software is developed, then immediately test can be executed.
  - Testing components should be stand alone.
  - It should simulate submission of the input to be tested.
  - It should check that the result meets the output specification.
- It consists of quick and easy executed tests. This helps to;
  - Checks all functionality of newly added codes.
  - Finds almost all problems of newly added codes.
- Because of test, first development task developers have thoroughly understand the specification of story of customers.
  - Code does not have any ambiguity.
  - Code does not omit any important specification.
- It avoids **'test-lag'** where, because the developer of the system works at faster pace than the tester. That means developer only gives his or her attention to develop and implement the code rather than testing which helps to extreme programming.

Disadvantages of the Test First Code are :

1. Programmers always prefer to write a code for tasks not for test.
2. Some time test codes are incomplete.
3. Some test codes are not checks any exceptional cases.
4. Some tests are very difficult to write because of complexity of the code.
5. Difficult to judge completeness of the test code.
6. Crucial part of the system may not be executed so it remains untested.
7. Customer does not give sufficient time for development.

Incremental test development from scenario :

This is common approach, which is adapted in any development. When coding is over, then code is given to another team for the testing purpose. It consists of following tests.

- Checking each individual developed component of the software.
- Validity of all input data.
- Expected results with respect to the inputs.
- Checking logic of each module.
- Checking all expected features and functionality of the software are accomplished by the code or not ?

User involvement in test development and validation :

This type of test is totally based on end user of the system. It is virtually impossible for a software developer to check whether the software is really use full to end user and how it is useful ? The main objective of the software development is the customer satisfaction. So it includes :

- End user checks all features and functionality of the software.
- All requirements are accomplished or not ?
- It conducts Alpha test and Beta test.
- The Alpha test is conducted at developers site by end user.
- Beta test is conducted at end user's site.

The use of automated test harnesses :

Here automatic test conducting software is built by the developer. This test software is developed after the code development is over.

- Customer and developer forms pair.
- Tests are prepared from pair.
- Test codes are developed by developer.
- Testing is done by user and developer.

Comparison of XP tests :

Comparison of all is given below :

Sr. No.	TDD	Incremental test development	User involvement in test development and validation	Automated test harness
1.	Tests are designed even before the coding begins.	It uses conventional white box testing	It uses conventional black box testing and acceptance testing.	Tests are designed after coding

Sr. No.	TDD	Incremental test development	User involvement in test development and validation	Automated test harness
2.	It ensures all requirements are fulfilled	Not all requirements are fulfilled.	It ensures all requirements are fulfilled	It ensures all requirements are fulfilled
3.	quick and fast	Lengthy	Lengthy	time taking to develop tests but then it is fast.
4.	Less time spent on testing.	More time spent on testing.	More time spent on testing.	More time spent on testing.

Terminologies used in XP :

Terminologies	Description or definition
Incremental planning	Requirements are recorded on story cards and the stories to be included in a release are determined by the time available and their relative priority. Developers break these stories into development tasks and make the schedule of development according to priorities of stories.
Small releases	– The minimal useful set of a functionality that provides business value is developed first is called as a small release. – Through new increments it adds some extra features or functionality is also called as a small release.
Simple design	Small design is carried out to meet the current requirements is called as a simple design.
Test first development	It is an automated unit test frame work, which is developed before actual task development.
Refactoring	Refactoring is nothing but improving the quality of code continuously such that it can be used in future development.
Pair programming	Each group is developed code one story. Where each group consists of two developers and they are checking each others work.
Collective ownership	Different pair of developers are work on distinct areas of one project. So development ownership of the project is not goes to any single developer. It goes to all developer working on this project so it is called as collective ownership.
Continuous integration	As soon as a pair of developer completes their task with proper testing, immediately that code is integrated into the whole system. After integration, system should pass through the unit test.
Sustainable pace	Large amount of overtime is not acceptable because it reduces quality of the code and medium term productivity.
On-site customer	When customer or representative of the customer or end user is working with XP team for the development as team member of Agile team then this customer is called as on-site customer.

Application of XP

- Planning game
- Small releases
- Customer acceptance tests
- Simple design
- Pair-driving programming
- Test driving development
- Refactoring the program / code
- Continuous progressive integration
- Collective code ownership
- Use of coding standards
- Metaphor for modeling
- Maintaining sustainable pace of development.

Review Questions

Q. 1 Explain in brief the various types of Software Process.
Q. 2 List the umbrella activities followed in generic process model.
Q. 3 Explain waterfall model in brief.
Q. 4 What is the need of feasibility study? Explain its types.
Q. 5 List the advantages and drawbacks of Incremental model
Q. 6 State the difference between :
(a) Classic life cycle model and Prototyping model
(b) Waterfall model and Spiral model
Q. 7 What does the radius imply in Spiral model ?
Q. 8 State the purpose, advantages and drawbacks of RUP?
Q. 9 What is agility ? Explain XP in detail.
Q. 10 What is Test First Development ?

❑❑❑