Unit 2

Software Life cycle models

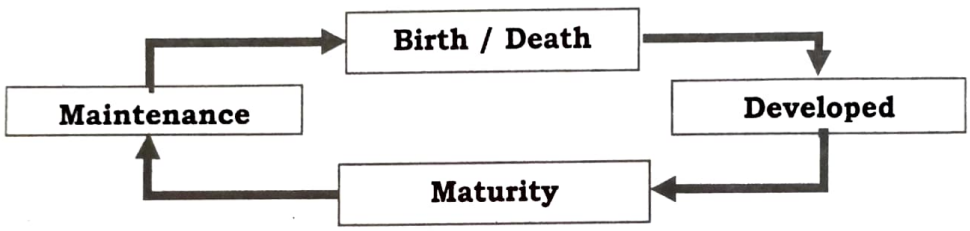
Unit Outcomes (UOs) :

2.1 Select software project model for software development

2.2 Agile development

❖ Software development models / Life cycle models

- Every system has a life cycle. It begins when a problem is recognized, after then system is developed, grows until maturity and then maintenance needed due to change in the nature of the system. So it died and new system or replacement of it taken place. (this can be shown in below figure)
- Software process models describe the sequence of activities needed to develop a software product.



System Life Cycle

- The goal of the software development process is to produce high quality software product.
- As per IEEE Standards, software life cycle is: "the period of time that starts when software product is conceived and ends when the product is no longer available for use."
- A software life cycle model is also called a Software Development Life Cycle (SDLC). More suitable definition of SDLC is given below :

Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality software, which meets exact customer's requirements and completion within time and cost estimation.

- Life cycle model comes into picture when we compare exploratory programs and modern software engineering approach.
- Software life cycle is the series of identifiable stages that a software product undergoes during its lifetime.
- Software life cycle model (process model) → is a descriptive and diagrammatic representation of the software life cycle.
- A life cycle model represents all the activities required to make a software product transit through its life cycle phases.

- General stages of software development life cycle are → feasibility study, requiremen analysis and specification, design, coding, testing and maintenance.

➥ **Need of life cycle models**

- Process models provide generic guidelines for developing a suitable process for a project
- It provides improvement and guarantee of quality product.
- Without using of a particular life cycle model the development of a software produc would not be in a systematic and disciplined manner.
- Provide monitoring the progress of the project to project managers.
- A software life cycle model defines entry and exit criteria for every phase.
- These criteria used to chart the progress of the project.
- A phase can begin only when the corresponding entry criteria are satisfied.
- If entry and exit criteria for various phases are satisfied, it's easy to monitor the progress of the project.
- The documentation of life cycle models enhances the understanding between developers and client.

➥ **Different Software Development Life Cycle (SDLC) models OR Software Development Models**

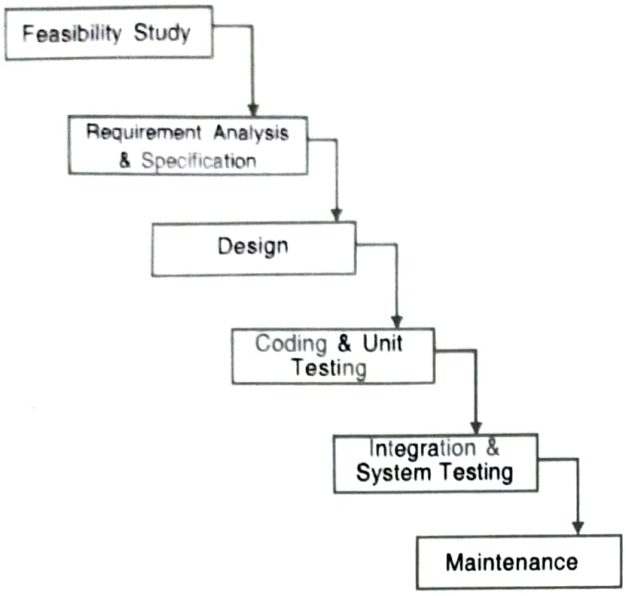
Many life cycle models have been proposed so far. Each of them has some advantages as well as some disadvantages. A few important and commonly used life cycle models are as follows:

(1) Classical waterfall model
(2) Iterative waterfall model
(3) Prototyping model
(4) Evolutionary model
(5) Spiral model
(6) RAD (Rapid Application Development) model

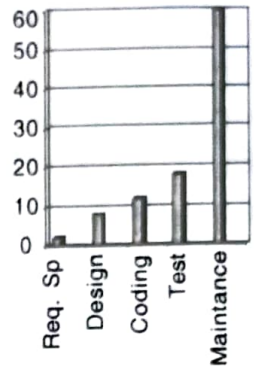
2.1.1 Waterfall model

- This model was originally proposed by Royce (1970). It is also called 'linear sequentia life cycle model'.
- It is also called 'traditional waterfall model' or 'conventional waterfall model'.
- It is the most obvious way to develop the software.
- It's just a theoretical way of developing s/w. All other life cycle models are essentially derived from it.
- This model break down the life cycle into set of phases like:
  - Feasibility study
  - Requirements analysis and specification,
  - Design
  - Coding and unit testing
  - Integration and system testing
  - Maintenance

During each phase of life cycle, a set of well defined activities are carried out. Each phase required different amount of efforts (figure 1.7).



Classical Waterfall model



Effort distribution among phases

- Phases between feasibility study and testing known as development phases.
- Among development phases, testing phase consumes the maximum effort.
- Among all phases of life cycle, maintenance phase consumes maximum effort.

Now let's discuss the characteristics and working of every phase one by one :

1. **Feasibility Study**

- Aim of this phase is to determine whether the system would be financially and technically feasible to develop the product.
- This is an abstract definition of the system.
- It includes the analysis of the problem definition and collection of relevant information of input, processing and output data.
- Collected data are analysed for :
  - → An abstract definition
  - → Formulation of different solutions
  - → Analysis of alternative solutions
- Feasibility is evaluated from developer and customer's point of view.
- Cost/Benefit analysis is also performed at this stage.
- Three main issues are concerned with feasibility study:
- **Technical Feasibility** → contains hardware, software, network capability, reliabili and availability.
- **Economical Feasibility** → contains cost/benefit analysis.
- **Operational Feasibility** → checks the usability. Concerned with technical performanc and acceptance within the organization.

2. Requirement Analysis and Specification

- Aim of this phase is to understand the exact requirements of the customer and to document them properly.
- It also reduces communication gap between developers and customers.
- Two different activities are performed during this phase:
  1. Requirements gathering and analysis
  2. Requirements specification

➥ **Requirement gathering**: The goal of this activity is to collect all relevant information from the customer regarding the product to be developed.

- This is done to clearly understand the customer requirements so that incompleteness and inconsistencies are removed.

➥ **Requirements analysis**: Analyse all gathered requirements to determine exact needs of the customers/clients and hence improve the quality of the product.

➥ **Requirements specification**: During this activity, the user requirements are systematically organized into a Software Requirements Specification (SRS) document.

- The important components of SRS are → the functional requirements, the non-functional requirements and the constraints of the system.
- Output of this phase is → SRS document, which is also called Black box specification of the problem.
- This phase concentrates on "what" part, not "how".

Requirement gathering and requirement analysis & specification collectively called **'Requirement Engineering'**.

3. Design

- The goal of the design phase is to transform the requirements specified in the SRS document into a structure that is suitable for implementation in some programming language.
- This phase affecting the quality of the product.
- Two main approaches are concerned with this phase.
  1. Traditional design approach
  2. Object oriented design approach

➥ **Traditional design approach**

- This approach divided into two parts: structure analysis and structure design.

➢ **Structural Analysis :**

- Where the detailed structure of the problem is examined.
- Identify the processes and data flow among these processes.
- DFD is used to perform the structure analysis and to produce result.
- In structure analysis – functional requirement specified in SRS are decomposed and analysis of data flow is represented diagrammatically by DFD.

➢ **Structure Design**

- During structured design, the results of structured analysis are transformed into the software design.
- Two main activities are associated with it :

→ **Architectural design (High-level design)** – decomposing the system into modules and build relationship among them.

→ **Detailed design (Low-level design)** – identified individual modules are design with data structure and algorithms.

➥ **Object oriented design approach**

- In *object oriented approach*, objects available in the system and relationships between them are identified.
- This approach provides lower development time and effort.
- Several tools and techniques are used in designing like:

✓ Flow chart	✓ DFD	✓ Data Dictionary
✓ Decision Table	✓ Decision Tree	✓ Structured English

4. Coding and Unit testing

- It is also called the implementation phase.
- The aim of this phase is to translate the software design into source code and unit testing is done module wise.
- Each component of the design is implemented as a program module, and each unit is tested to determine the correct working of the system.
- The system is being operational (working conditions) at this phase.
- This phase affects testing and maintenance.
- Simplicity and clarity should be maintained.
- Output of this phase is → set of program modules that have been individually tested.

5. Integration and system testing

- Once all the modules are coded and tested individually, integration of different modules is undertaken.
- The modules are integrated in a planned manner.
- This phase is carried out incrementally over a number of steps and during each integration step, the partially integrated system is tested and a set of previously planned modules are added to it.
- Finally, when all the modules have been successfully integrated and tested, system testing is carried out.
- Goal of this phase is → to ensure that the developed system works well to its requirements described in the SRS document.
- Basic function of testing is to detect errors. So it is also called '*quality control measure*'.
- Testing procedure is carried out using test data: Program test and System test. Program test is done on test data and system test is done actual data.
- Proper test cases should be selected for successful testing.

- It consists of three different kinds of testing activities.
  - ✓ **α- testing :** It is the system testing performed by the development team.
  - ✓ **β-testing :** It is the system testing performed by a friendly set of customers.
  - ✓ **Acceptance testing** : It is the system testing performed by the customer himself after the product delivery to determine whether to accept or reject the delivered product.

Output of this phase is → system test plan and test report (error report).

6. Maintenance

- It requires maximum efforts among all the phases to develop software product.
- This phase is needed to keep system operational.
- Generally maintenance is needed due to change in the environment or the requirement of the system.
- Maintenance involves performing following four kinds of activities:
  - ✓ **Corrective maintenance** – Correcting errors that were not discovered during the product development phase. It is done after product development.
  - ✓ **Perfective maintenance** – Improving and enhancing the functionalities of the system according to the customer's requirements.

    It mainly deals with implementing new or changed user requirements and improving system performance.
  - ✓ **Adaptive maintenance** – Porting the software to work in a new environment and ensure working. It also consists of adaption of software in the changes of environment.
  - ✓ **Preventive maintenance** –is scheduled, routine maintenance to keep equipment running as well as prevent downtime and expensive repair cost. It reduces the likelihood of failure.

In this phase, the system is reviewing for studying the performance and knowing the full capabilities of the system.

➥ **Advantages of waterfall model**

→ It is simple and easy to understand and use.

→ Clearly defined stages. As each phase has well defined input and outputs.

→ It helps project personnel in planning.

→ Waterfall model works well for smaller projects where requirements are very well understood.

→ Well understood milestones.

→ Results are well documented.

➥ **Disadvantages of waterfall model**

→ High amounts of risk and uncertainty.

→ It is a theoretical model, as it is very difficult to strictly follow all the phases in all types of projects.

→ Poor model for long and on-going projects.

→ Not so good for complex and object oriented projects. Also it cannot be used for the projects where requirements are changed frequently.

→ It may happen that the error may be generated at any phase and encountered in later phase. So it's not possible to go back and solve the error in this model.

➥ **Application of waterfall model (When to use waterfall model)**

→ This model is used only when the requirements are very well known, clear and fixed.

→ When environment is stable.

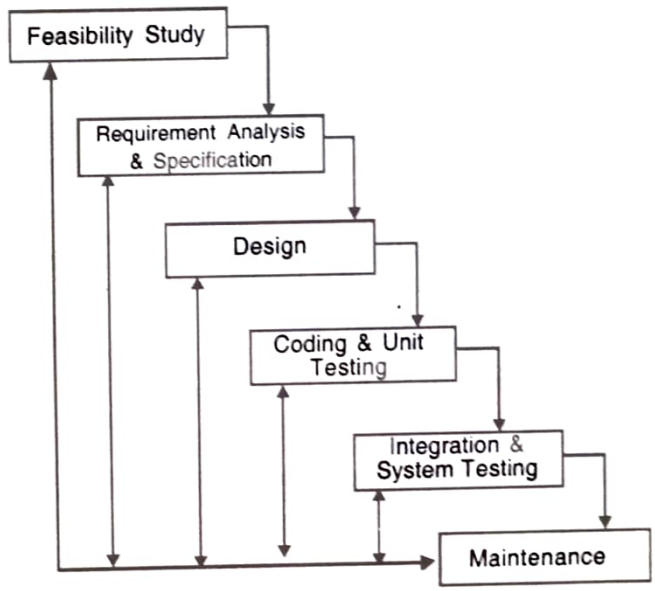
→ When product definition is stable.

→ When technology is understood.

→ When the project is small.

❖ **Iterative Waterfall model**

- Classical waterfall model is idealistic: it assumes that no defect is introduced during any development activity.
- But in practice: defects do get introduced in almost every phase of the life cycle. Even defects may get at much later stage of the life cycle.
- So, solution of this problem is → iterative waterfall model.
- Iterative waterfall model is by far the most widely used model. Almost every other model is derived from the waterfall model.
- The principle of detecting errors as close to its point of introduction as possible – is known as *phase containment of errors*.
- Phase containment of errors can be achieved by reviewing after every milestone.

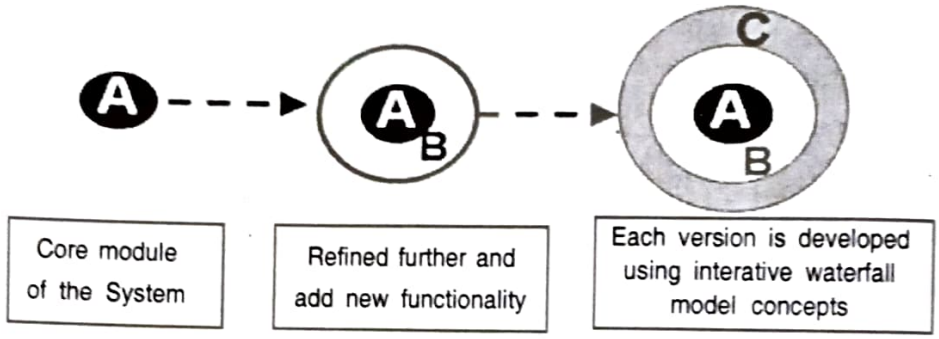


Iterative Waterfall Model

2.1.2 Incremental model

- It is also referred as the successive version of waterfall model using incremental approach and evolutionary model.
- In this model, the system is broken down into several modules which can be incrementally implemented and delivered.
- First develop the core product of the system. The core product is used by customers to evaluate the system.

- The initial product skeleton is refined into increasing levels of capability: by adding ne functionalities in successive versions.



Incremental model

- From above figure, we can analyse that at the initial stage, core module/product is developed. Then by adding customer requirements or new functionalities the incremental version is built. Each version is developed using iterative waterfall model concepts.

➥ **Advantages**

→ Each successive version performing more useful work than previous versions.
→ Initial product deliver is faster.
→ The core modules get tested thoroughly, thereby reducing chance of errors in final product.
→ The model is more flexible and less costly to change the scope and requirements.
→ User gets a chance to experiment with partially developed software.
→ This model helps finding exact user requirements
→ Feedback providing at each increment is useful for determining the better final product.

➥ **Disadvantages**

→ Sometimes it is difficult to subdivide problems into functional units.
→ Model can be used for very large problems.
→ It needs good planning and design.
→ Resulting product cost may exceed.

➥ **Applications**

→ When the problem is very large and user requirements are not well specified at initial stage.
→ When new technology is used in development.

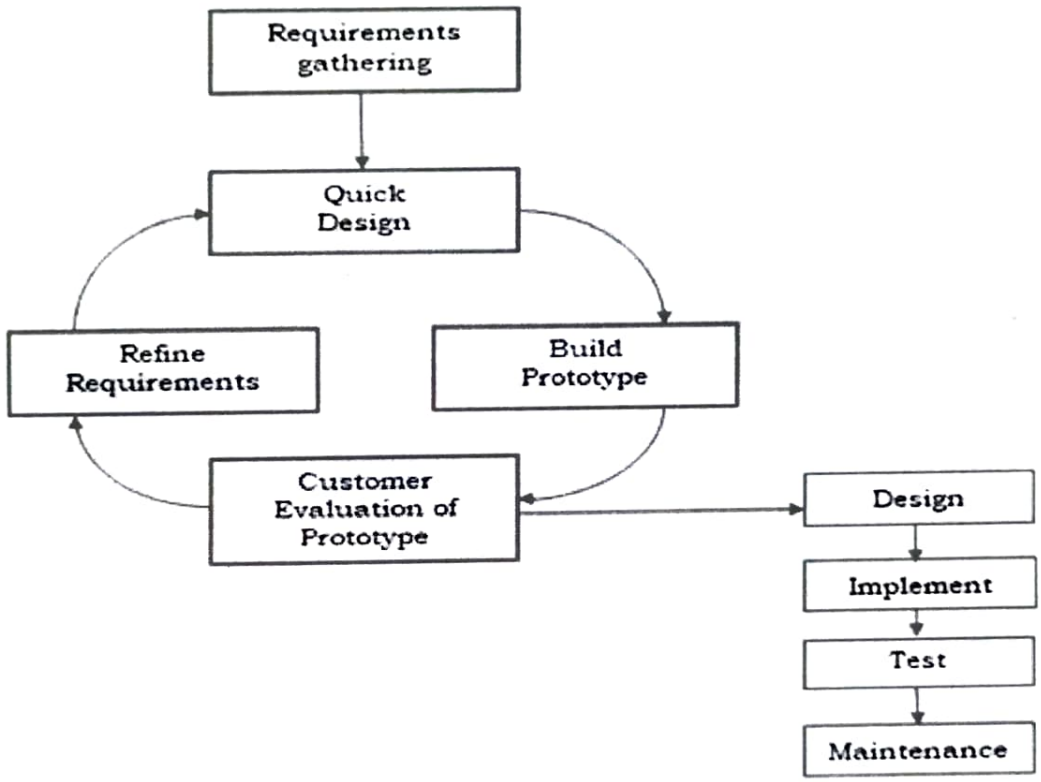
2.1.3 Prototype model

- Prototype is a working physical system or sub system. Prototype is nothing but a 'toy implementation of a system'.
- In this model, before starting actual development, a working prototype of the system should be built first.
- A prototype is actually a partial developed product.
- A prototype usually turns out to be a very crude version of the actual system.
- Compared to the actual software, a prototype usually have

- limited functional capabilities
- low reliability
- inefficient performance

- Prototype usually built using several shortcuts, and these shortcuts might be inefficient, inaccurate or dummy functions.
- Prototype model is very useful in developing GUI part of system.

The prototype model is shown in below figure.



In working of the prototype model, product development starts with initial requirements gathering phase.

- Then, quick design is carried out and prototype is built.
- The developed prototype is then submitted to the customer for his evaluation.
- Based on customer feedback, the requirements are refined and prototype is modified.
- This cycle of obtaining customer feedback and modifying the prototype continues till the customers approve the prototype.
- Then the actual system is developed using the different phases of iterative waterfall model.
- There are two types of prototype model.

I. **Exploratory development prototype model :** in which the development work is done with the users to explore their requirements and deliver a final system.

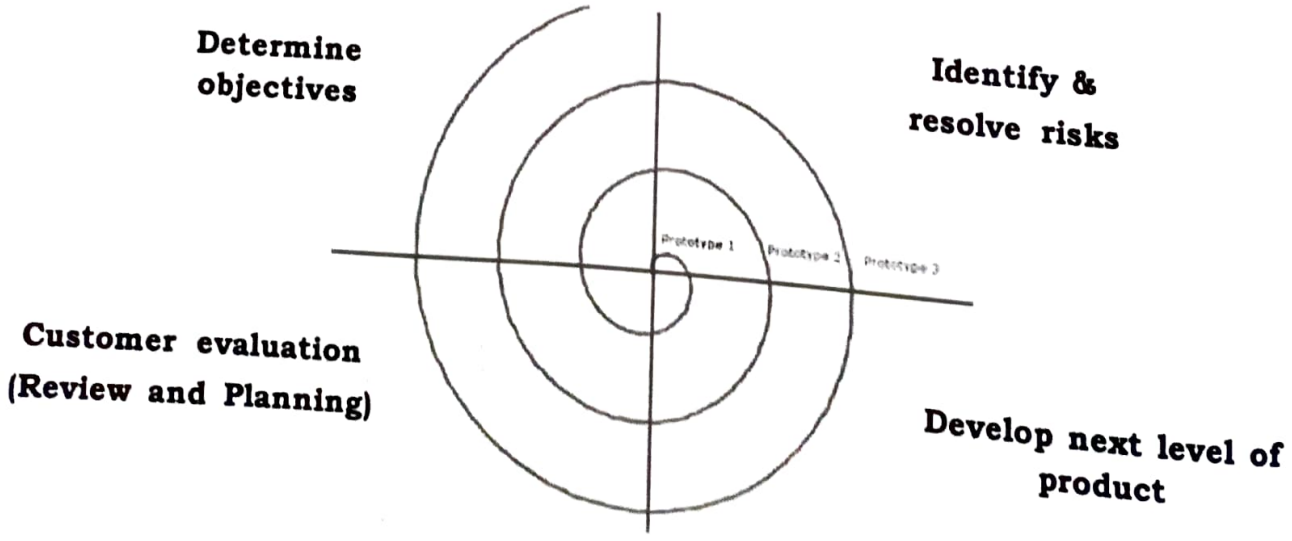
II. **Throw away prototype model :** in which a small part of the system is developed and given to the customer to try out and evaluate. Then feedback is collected from customer and accordingly main system is modified. The prototype is then discarded.

- **Advantages**
  - → A partial product is built at initial stage, so customers can get a chance to have a look of the product.
  - → New requirements can be accommodate easily.
  - → Missing functionalities identified quickly.
  - → It provides better flexibility in design and development.
  - → As the partial product is evaluated by the end users, more chance of user satisfaction.
  - → Quick user feedback is available for better solution.
- **Disadvantages**
  - → The code for prototype model is usually thrown away. So wasting of time is there.
  - → The construction cost of developing the prototype is very high.
  - → If end user is not satisfied with the initial prototype, he may lose interest in the final product.
  - → This model requires extensive participation and involvement of the customers that is not possible every time.
- **Application**
  - → This model used when desired system needs to have a lot of interactions with end users.
  - → This type of model generally used in GUI (Graphical User Interface) type of development.

2.1.4 Spiral Model

- Spiral model is proposed by Boehm in 1986.
- In application development, spiral model uses fourth generation languages (4GL) and developments tools.
- The diagrammatic representation of this model appears like a spiral with many loops.

Figure of spiral model shown below.



Spiral Model

Each loop of the spiral represents a phase of the software process:

- the innermost loop might be concerned with system feasibility,
- the next loop with system requirements definition,
- the next one with system design, and so on
- Number of phases is not fixed in this model.
- This model is more flexible compared with other models.
- Each loop in the spiral is split into four sectors (quadrants)

❖ **1st Quadrant: Determine objectives**

- Identifying the objectives, identifying their relationships and find the possible alternative solutions. Objectives like: performance, functionality, hardware software interface etc.
- Also examine the risks associated with these objectives.

❖ **2nd Quadrant: Identify and resolve risks**

- In this phase, detailed analysis is carried out of each identified risk
- Alternate solutions are evaluated and risks are reduces at this quadrant.

❖ **3rd Quadrant: Develop next level product**

- Develop and validate the next level of the product after resolving the identified risks.
- Activities like design, code development, code inspection, testing and packaging are performed.
- Resolution of critical operations and technical issues of next level product are performed.

❖ **4th Quadrant: Review and Planning (Customer evaluation)**

- In this part, review the results achieved so far.
- Plan the next iteration around the spiral. Different plans like development plan, test plan, installation plan are performed.
- With each iteration around the spiral: progressively more complete version of the software gets built.
- In spiral model → at any point, Radius represents: cost and Angular dimension represents: progress of the current phase.
- At the end, all risks are resolved and s/w is ready to use.

➥ **Advantages**

- It is more flexible, as we can easily deal with changes.
- Due to user involvement, user satisfaction is improved.
- It provides cohesion between different stages.
- Risks are analysed and resolved so final product will be more reliable.
- New idea and additional functionalities can be easily added at later stage.

➥ **Disadvantages**

- It is applicable for large problem only.
- It can be more costly to use.

- It is more complex to understand.
- More number of documents are needed as more number of spirals.
- Risk analysis phase requires much higher expertise.

➥ **Application (when to use spiral model)**

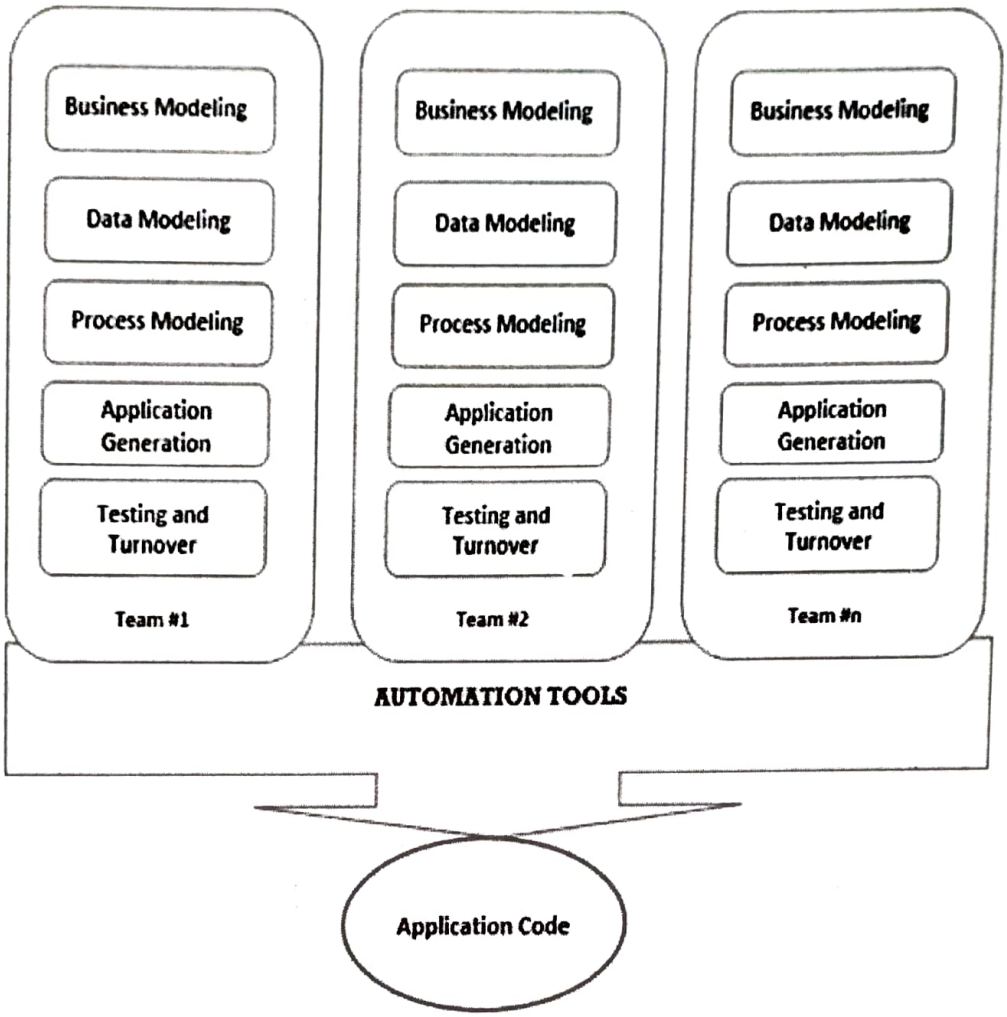
- Used when medium to high risk projects.
- When users are unsure for their needs.
- When requirements are complex.

❖ **Spiral model can be viewed as a Meta model**

- Spiral model subsumes almost all the life cycle models.
- Single loop of spiral represents Waterfall Model.
- Spiral model uses a prototyping approach by first building the prototype before developing actual product.
- The iterations along the spiral model can be considered as supporting the Evolutionary Model.
- The spiral model retains the step-wise approach of the waterfall model.

2.1.5 RAD (Rapid Application Development) model

- Full form of RAD is - Rapid Application Development. This model was proposed by *IBM* in 1980.
- Rapid application development (RAD) is an incremental software development process model that emphasizes on extremely short development cycle.
- It emphasize on reuse.
- The RAD model is a "high-speed" adaptation of the linear sequential model in which rapid development is achieved by using component-based construction.
- If requirements are well understood and project scope is limited, the RAD process enables a development team to create a "fully functional system" within very short time periods (e.g., 60 to 90 days).
- In this model, user involvement is essential from requirement analysis to delivery of product (project).
- For this model, requirements must be cleared and well understood initially.
- Figure of this model is shown below. Many development teams are working parallel to complete the task.



RAD Model

➥ **Phases of RAD model are**

1. **Business modeling**

- The information flow among business functions is carried out in this phase. Business functions are modelled.

2. **Data modeling**

- The characteristics of objects (attributes) and their relationships are defined.

3. **Process modeling**

- The data objects defined in data modeling are transformed to implement business functions.

4. **Application generation**

- Automated tools are used to convert process models into code and the actual system. Tools like VB, VC++and JAVA etc are used.
- RAD works to reuse existing components or create new ones.

5. **Testing and turn over**

- As RAD uses the components that already been tested so the time for developing and testing is reduced.

➥ **Advantages**

→ Application can be developed in a quick time.

→ This model highly makes use of reusable components.

→ Reduce time for developing and testing.

→ Due to full customer involvement, customer satisfaction is improved.

➥ **Disadvantages**

→ Requirements must be cleared and well understood for this model.

→ It is not well suited where technical risk is high.

→ In it, highly skilled and expert developers are needed.

→ User involvement is necessary.

➥ **Applications**

→ The RAD model is mostly used when the system is modularized and all the requirements are well defined.

→ When a system needs to be produced in a short span of time (2-3 months).

→ RAD SDLC model should be chosen only if resources with high business knowledge are available.

→ When the technical risk is less.

❖ **Comparison of different Life cycle models**

- **The Classical waterfall model**

→ Can be considered as a basic model as all other models are derived from this model.

→ But, it is not used in practical development as no mechanism to handle errors committed during the phase.

- **The Iterative waterfall model**

→ Most widely used model.

→ But, suitable only for well-understood problems.

- **The Prototype model**

→ Suitable for projects in which user requirements and technical aspects are not well understood.

→ Especially popular for user interface part of the project (GUI based projects).

- **The Evolutionary model**

→ It is suitable for large problems: as it can be decomposed into a set of modules that can be incrementally implemented.

→ Also used for object oriented development projects.

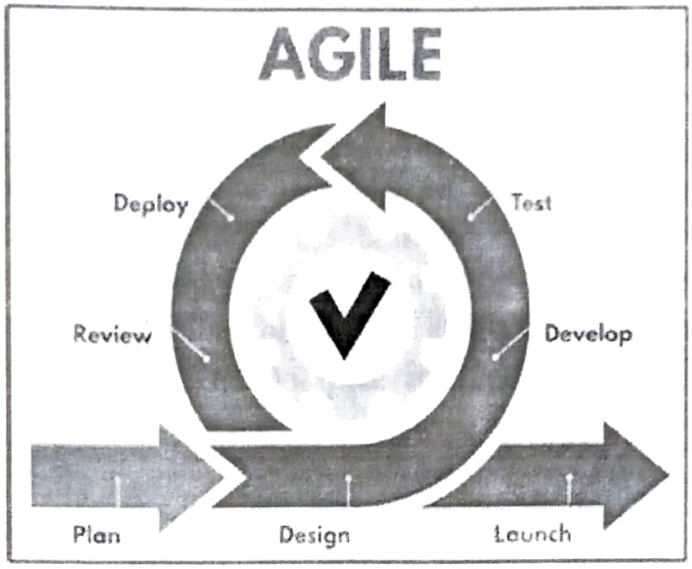
→ But can be used only if the incremental delivery of the system is acceptable by the customer.

- **The Spiral model**

→ Used to develop the projects those have several kinds of risks

→ But, it is much complex than other models.

2.1.1 Agile process and principles



- A simple meaning of agile is → ready to move in quick and easy way.
- Agile is being very popular now a days in software development area.
- Agile methodology is a "step by step" dynamic focused on short-term visibility but never losing the long-term product goal.
- The agile methodology is the way or manner to divide the project into small phases and make it easily manageable. It includes constant stakeholder collaboration and continuous improvement at each phase.
- This methodology adopt constant changes via repetitive or iterative approach to software design and development.
- Each iteration is considered as a short time "frame" in the agile process model, which typically lasts from one to four weeks. The division of the entire project into smaller parts helps to minimize the project risk and to reduce the overall project delivery time requirements.
- Each iteration involves a team working through a full software development life cycle including planning, requirements analysis, design, coding, and testing before a working product is demonstrated to the client.
- At the end of each iteration, a working product with a new feature has to be delivered.
- Most popular and common agile methodology examples (or frameworks) are:
  - ✓ Scrum
  - ✓ Kanban
  - ✓ XP (eXtreme Programming)
  - ✓ Dynamic System Development Method (DSDM)
  - ✓ Feature Driven Development (FDD)
  - ✓ Adaptive Software Development (ASD)
  - ✓ Adaptive Project Framework (APF)
  - ✓ Lean Software Development (LSD)

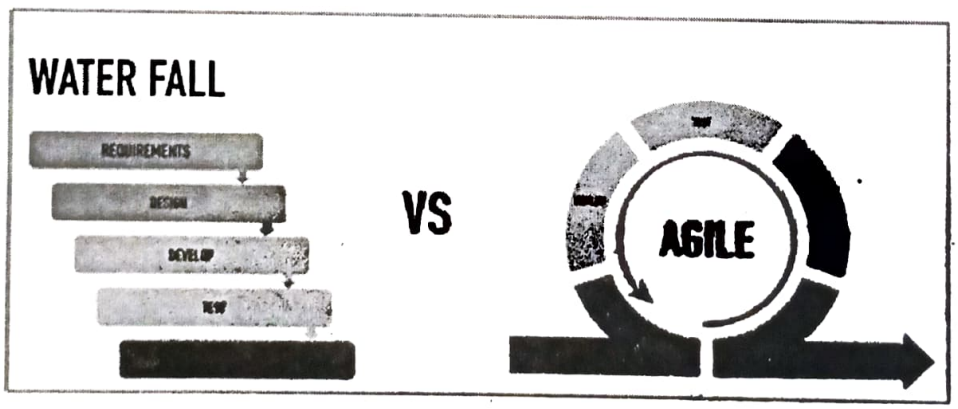
- Agile methods can help teams to manage work more efficiently and do the work more effectively while delivering the highest quality product within the constraints of the budget.

❖ **Agile principles**

Some important principles of agile are listed below:

- Highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcomes changing requirements, even late in development.
- The whole-team approach avoids delays and wait times.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shortest timescale.
- Build projects around motivated individuals. Give them the environment and the support they need, and trust them to get the job done.
- Working software is the primary measure of progress.
- Simplicity the art of maximizing the amount of work not done is essential.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Agile processes promote sustainable development due to their constant rhythm and technical excellence, which in turn improves productivity.

2.2.2 Comparison of agile development with traditional models

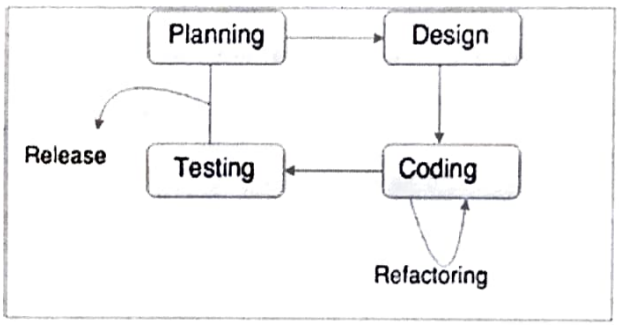


Traditional model approach	Agile approach
→ It is process oriented.	→ It is people oriented.
→ It is based on command and control project management.	→ It is based on leadership and collaboration project management.
→ In general – waterfall, spiral or prototype approach.	→ In it – evolutionary approach.
→ Does not involve customers continuously.	→ Involvement of customers is continuous.
→ Clear initial requirements.	→ Creative, innovative and unclear requirements.
→ This approach is better for bigger projects.	→ This is better for smaller projects.

2.2.3 eXtreme programming (XP) model

- This is a typical agile development framework, developed by Kent Beck.
- This type of methodology is used when customers are constantly changing demands or requirements, or when they are not sure about the system's performance.
- XP is a lightweight, efficient, low-risk, flexible, predictable, scientific, and fun way to develop a software.
- It advocates "frequent releases" of the product in short development cycles, which inherently improves the productivity of the system and also introduces a checkpoint where any customer requirements can be easily implemented.
- The XP develops software keeping customer in the target. And keeping customer satisfaction with quality product in mind.
- Teamwork is extremely important in XP, since, when there is a problem, it is solved by the whole team of managers, developers or customers, bringing them together.
- The sprint is released in very shorter life cycle (almost in 15-20 days).
- XP is achieved with :
  - ✓ Emphasis on continuous feedback from the customer
  - ✓ Short iterations
  - ✓ Design and redesign
  - ✓ Coding and testing frequently
  - ✓ Eliminating defects early, thus reducing costs
  - ✓ Keeping the customer involved throughout the development
  - ✓ Delivering working product to the customer

The process of XP model shown in below figure



- **Planning**

→ The first stage, customer meets to the developer team and give his requirements and desired result. Then the team broken down the project into smaller iterative tasks and creates a release plan.

- **Designing**

→ Then after deigning is conducted to bring logic and structure to the system and allows to avoid unnecessary complexities and redundancies.

- **Coding**

→ It is the phase during which the actual code is created by implementing specific XP practices such as coding standards, pair programming, continuous integration, an collective code ownership.

→ Code factoring technique is used to continuously improve the code.

→ Refactoring is about removing redundancy, eliminating unnecessary functions, increasing code coherency.

- **Testing**

→ It is the core of extreme programming. It is the regular activity that involves both unit tests and acceptance tests.

Everyone in a team works jointly at every stage of the project.

The XP model suggests quick release of a small chunk and further developing the product by making small and incremental updates.

Small releases allow developers to frequently receive feedback, detect bugs early, and monitor how the product works in production.

➥ **Advantages of XP model**

- Continuous testing and refactoring practices help create stable well-performing systems with minimal debugging.
- Less number of documents needed.
- Results can be derived soon.

➥ **Disadvantages of XP model**

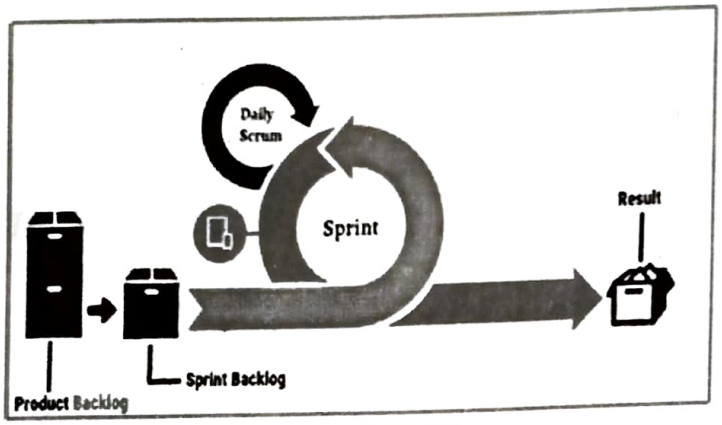
- Pair programming in this model takes more time.
- Customer involvement is necessary all the time. In some situation customer may not be free to give feedback.
- It requires more self

➥ **Application of XP model**

- Mainly used for smaller projects.
- Used to build projects involving new technology or research projects.
- High cost to build projects using this method.
- XP is not the best option if programmers are separated geographically.

2.2.4 Scrum model

- Scrum is the most used one of the agile methodoogies.
- It is a lightweight agile process framework used primarily for managing software development.
- Scrum is an agile development process focused primarily on ways to manage tasks in team-based development conditions.
- This model based on an iterative and incremental processes.
- Scrum is adaptable, fast, flexible and effective agile framework that is designed to deliver value to the customer throughout the development of the project.
- Scrum is characterised by cycles or stages of development, known as **'sprints'**.
- Every day starts with a small 15-minute meeting, called 'daily scrum', which takes the role of synchronising activities and finding the best way to plan out the working day.



In the process of scrum

- A Product Owner orders the work for a complex problem into a **Product Backlog**.
- The Scrum Team turns a selection of the work into an Increment of value during a **Sprint**.
- The Scrum Team and its stakeholders inspect the results and adjust for the next Sprint.
- Repeat the process until the team get desired sprint output.

➥ **There are three roles in this methodology, and their responsibilities are**

i. **Scrum master :** The scrum can set up the master team, arrange the meeting and remove obstacles for the process.

ii. **Product owner :** The product owner makes the product backlog, prioritizes the delay and is responsible for the distribution of functionality on each repetition.

iii. **Scrum Team :** The team manages its work and organizes the work to complete the sprint or cycle.

➥ **Advantages of scrum model**

- It is adaptable and flexible.
- Focus on quality is a constant with the scrum method.
- Team motivation is good because programmers want to meet the deadline for every sprint.
- Good sprint planning is prioritised, so that the whole scrum team understands the "why, what and how" of allocated tasks.
- Improved motivation in core team.
- More chance of customer satisfaction.

➥ **Disadvantages of scrum model**

- This methodology must require a team environment. Though this method doesn't work well with more number of team members.
- Development team should be well experience.
- Daily meetings sometimes frustrate team members.
- If any team member leaves in the middle of a project, it can have a huge negative impact on the project.
- This model is not appropriate for large and complex projects.

ASSIGNMENT

MCQs

1. Which phase of waterfall model consumes maximum efforts ?
   (a) testing (b) coding **(c) maintenance** (d) analysis
2. Which of the following model works better in handling risks ?
   (a) waterfall (b) prototype (c) incremental **(d) spiral**
3. ..... model also called meta model.
   (a) spiral (b) prototype (c) incremental (d) waterfall
4. What does the RAD software process stand for ?
   (a) Rapid Application Development (b) Recent Application Development
   (c) Relative Application Development (d) Rapid Application Design
5. What is the first step in the software development lifecycle ?
   (a) system design (b) coding
   (c) testing **(d) preliminary investigation**
6. The agile software development model is built based on ..... .
   (a) linear development
   (b) incremental development
   (c) iterative development
   (d) both incremental and iterative development

True / False

1. Success model is one of the life cycle models. **Answer : False**
2. Among development phase, maintenance phase of waterfall model consumes maximum efforts. **Answer : False**
3. Waterfall model is meta model. **Answer : False**
4. Users can get the chance to use partially developed product in prototype model. **Answer : True**
5. Iterative waterfall model is based on agile principles. **Answer : False**
6. Software maintenance costs are expensive in contrast to software development. **Answer : True**

Short Questions

1. Give the full form of :→ RAD, XP model, SDLC
2. List the phases which are coming in 'development phase' of waterfall model.
3. Define feasibility study of software ?
4. In which situation prototype model is used ?
5. List the phases of RAD model.
6. State situation where RAD model is used.
7. Justify: spiral model can be viewed as a meta model.
8. State the importance of agile development.

Descriptive Questions

1. Explain waterfall model with neat figure. **OR** particular two phases can be asked.)
2. Explain iterative waterfall model.
3. Explain spiral model.
4. Explain prototype model.
5. Explain umbrella activities with diagram.
6. Explain RAD model.
7. Explain incremental model.
8. Write a short note on scrum.
9. Explain XP methodology.

* * *