Object Oriented Analysis And Design
Aleenah Khan

Software Development Process Models

Software Development Life Cycle (SDLC)

- Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality softwares.
- The SDLC aims to produce a high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.

Software Development Life Cycle (SDLC)

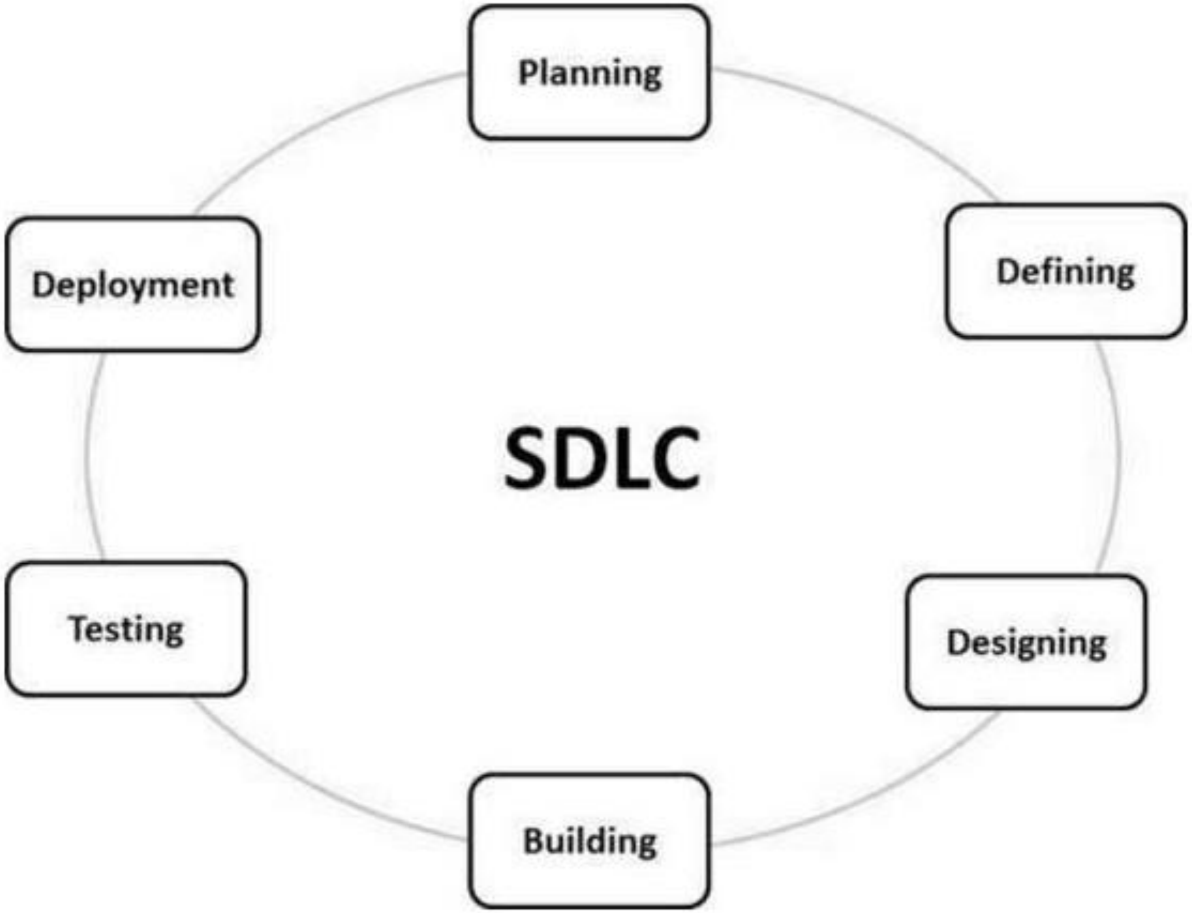
- SDLC is the acronym of Software Development Life Cycle.
- It is also called as Software Development Process.
- SDLC is a framework defining tasks performed at each step in the software development process.
- ISO/IEC 12207 is an international standard for software life-cycle processes. It aims to be the standard that defines all the tasks required for developing and maintaining software.

Software Development Life Cycle (SDLC)

- SDLC is a process followed for a software project, within a software organization.
- It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software.
- The life cycle defines a methodology for improving the quality of software and the overall development process.

Software Development Life Cycle (SDLC)

- The following figure is a graphical representation of the various stages of a typical SDLC.



Stage 1: Planning and Requirement Analysis

- Requirement analysis is the most important and fundamental stage in SDLC.
- It is performed by the senior members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry.
- This information is then used to plan the basic project approach and to conduct product feasibility study in the economical, operational and technical areas.
- Planning for the quality assurance requirements and identification of the risks associated with the project is also done in the planning stage.
- The outcome of the technical feasibility study is to define the various technical approaches that can be followed to implement the project successfully with minimum risks.

Stage 2: Defining Requirements

- Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts.
- This is done through an **SRS (Software Requirement Specification)** document which consists of all the product requirements to be designed and developed during the project life cycle.

Stage 3: Designing the Product Architecture

- SRS is the reference for product architects to come out with the best architecture for the product to be developed.
- Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a DDS - Design Document Specification.
- This DDS is reviewed by all the important stakeholders and based on various parameters as risk assessment, product robustness, design modularity, budget and time constraints, the best design approach is selected for the product.

Stage 3: Designing the Product Architecture

- A design approach clearly defines all the architectural modules of the product along with its communication and data flow representation with the external and third party modules (if any).
- The internal design of all the modules of the proposed architecture should be clearly defined with the minutest of the details in DDS.

Stage 4: Building or Developing the Product

- In this stage of SDLC the actual development starts and the product is built.
- The programming code is generated as per DDS during this stage.
- If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle.
- Developers must follow the coding guidelines defined by their organization and programming tools like compilers, interpreters, debuggers, etc. are used to generate the code.
- Different high level programming languages such as C, C++, Pascal, Java and PHP are used for coding.
- The programming language is chosen with respect to the type of software being developed.

Stage 5: Testing the Product

- This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC.
- However, this stage refers to the testing only stage of the product where product defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS.

Stage 6: Deployment in the Market and Maintenance

- Once the product is tested and ready to be deployed it is released formally in the appropriate market.
- Sometimes product deployment happens in stages as per the business strategy of that organization.
- The product may first be released in a limited segment and tested in the real business environment (UAT- User acceptance testing).
- Then based on the feedback, the product may be released as it is or with suggested enhancements in the targeting market segment.
- After the product is released in the market, its maintenance is done for the existing customer base.

SDLC MODELS

SDLC Models

- There are various software development life cycle models defined and designed which are followed during the software development process.
- These models are also referred as “**Software Development Process Models".**
- Each process model follows a Series of steps unique to its type to ensure success in the process of software development.

SDLC Models

Following are some popular SDLC models followed in the industry:

- Big-Bang model
- Code-and-fix model
- Waterfall model
- V model
- Incremental model
- RAD model
- Iterative model
- Agile model
- Prototype mode
- Spiral model

Big-Bang Model

- Big-Bang is the SDLC model in which no particular process is followed.
- Generally this model is used for small projects in which the development teams are small.
- It is specially useful in academic projects.
- This model needs a little planning and does not follow formal development.
- The development of this model begins with the required money and efforts as an input.
- The output of this model is developed software, that may or may not be according to the requirements of the customer.

Big-Bang Model

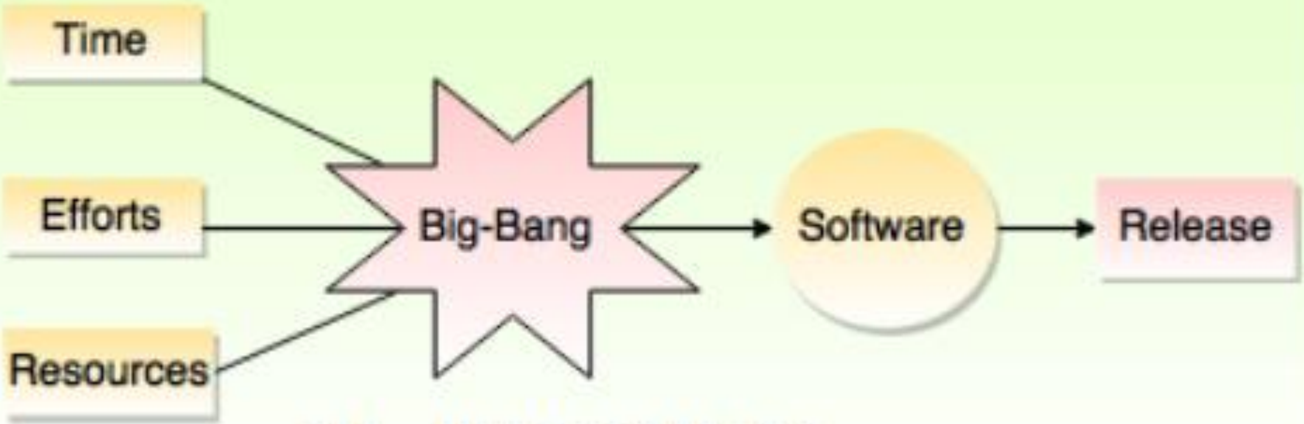


Fig. - Big-Bang Model

Big-Bang Model - Advantages

- Big-Bang model is a simple model.
- It needs little planning.
- It is simple to manage. It needs just a few resources to be developed.
- It is useful for students and new comers.

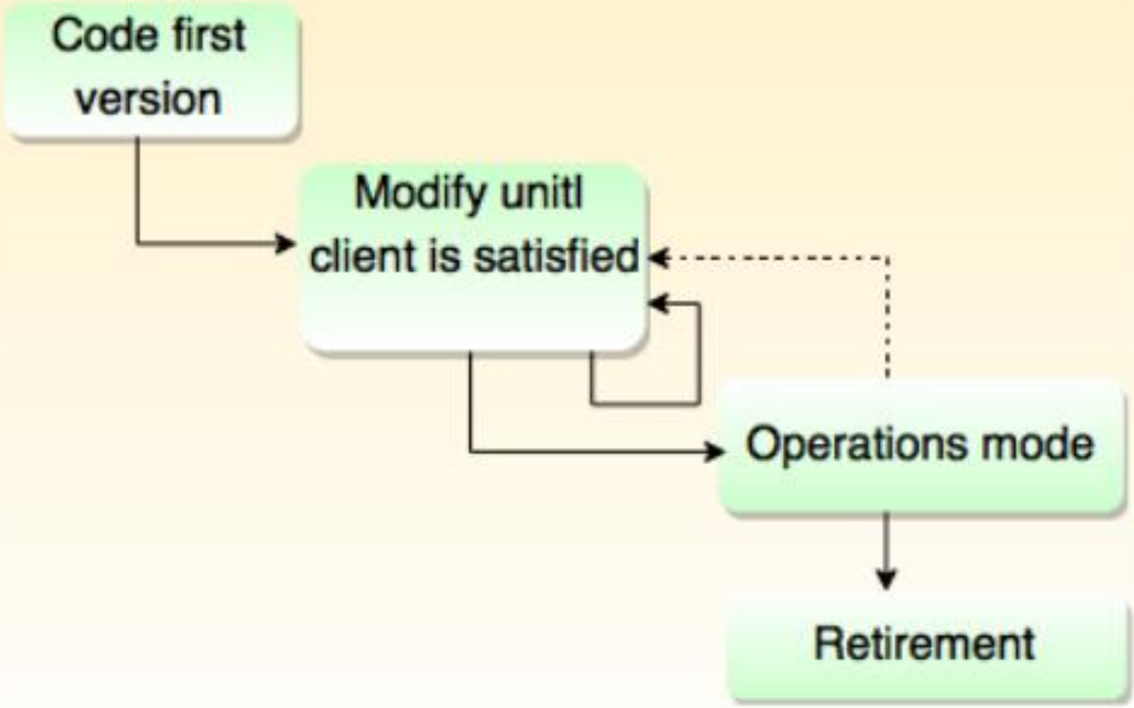
Big-Bang Model - Disadvantages

- It is a very high risk model.
- This model is not suitable for object oriented and complex projects.
- Big-Bang is poor model for lengthy and in-progress projects.

Code-and-Fix Model

- Code and fix model is one step ahead from the Big-Bang model.
- It identifies the product that must be tested before release.
- The testing team find the bugs then sends the software back for fixing.
- To deliver the fixes, developers complete some coding and send the software again for testing.
- This process is repeated till the bugs are found in it, at an acceptable level.

Code-and-Fix Model



Code first version

Modify unitl client is satisfied

Operations mode

Retirement

Fig. - Flow of Code & Fix Model

Code-and-Fix Model

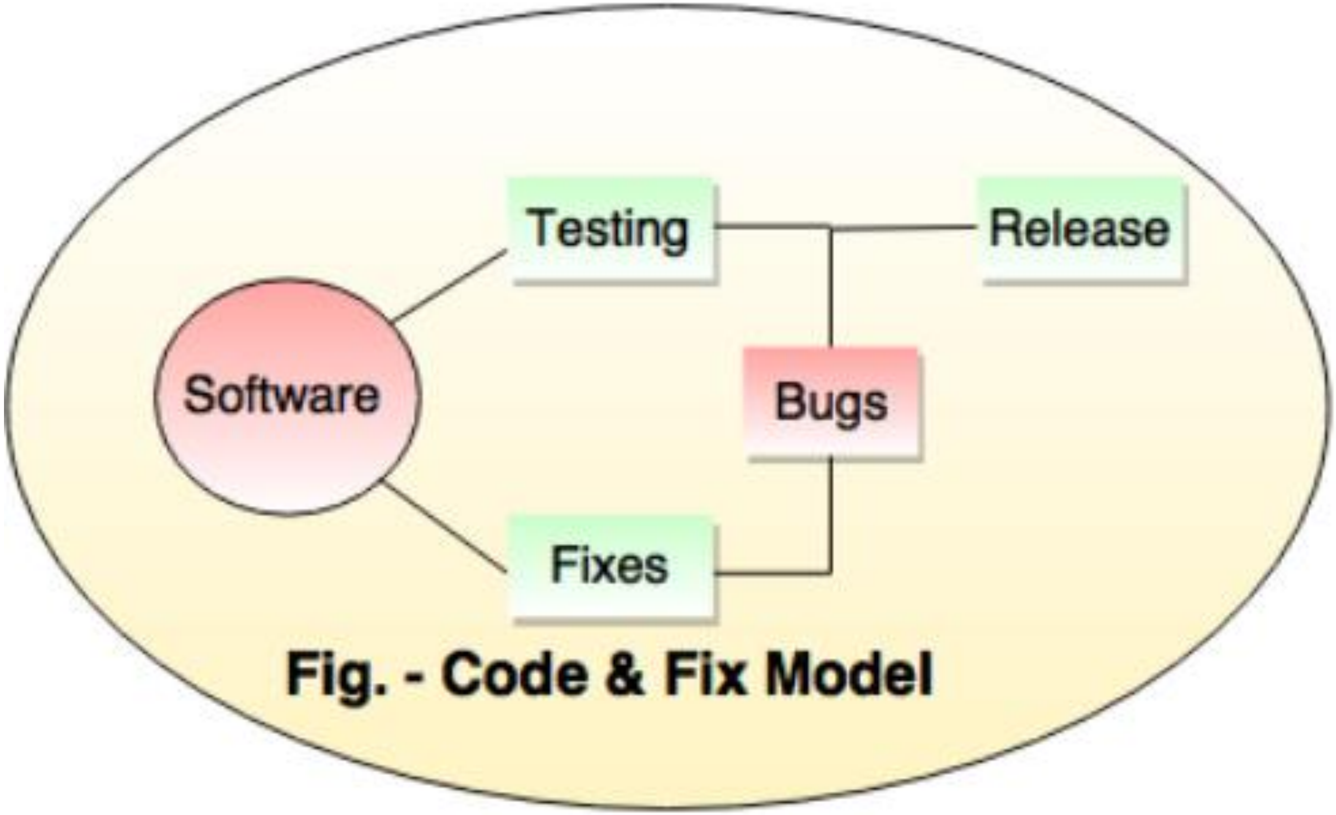


Fig. - Code & Fix Model

Code-and-Fix Model - Advantages

- This model is suitable for small projects.
- It needs less project planning.

Code-and-Fix Model - Disadvantages

- It is difficult to accommodate changes.
- It is not clear what will be delivered and when.
- It is difficult to assess quality.

Waterfall Model

- The waterfall model is the classic model or oldest model and is known as **mother of all the model**.
- It is widely used in government projects and many vital projects in company.
- The waterfall model is also called as **'Linear Sequential Model'** or **'Classic Life Cycle Model'.**
- In this model, each phase is executed completely before the beginning of the next phase. Hence the phases do not overlap in waterfall model.
- This model is used for small projects.
- In this model, feedback is taken after each phase to ensure that the project is on the right path.
- Testing part starts only after the development is completed.

Waterfall Model

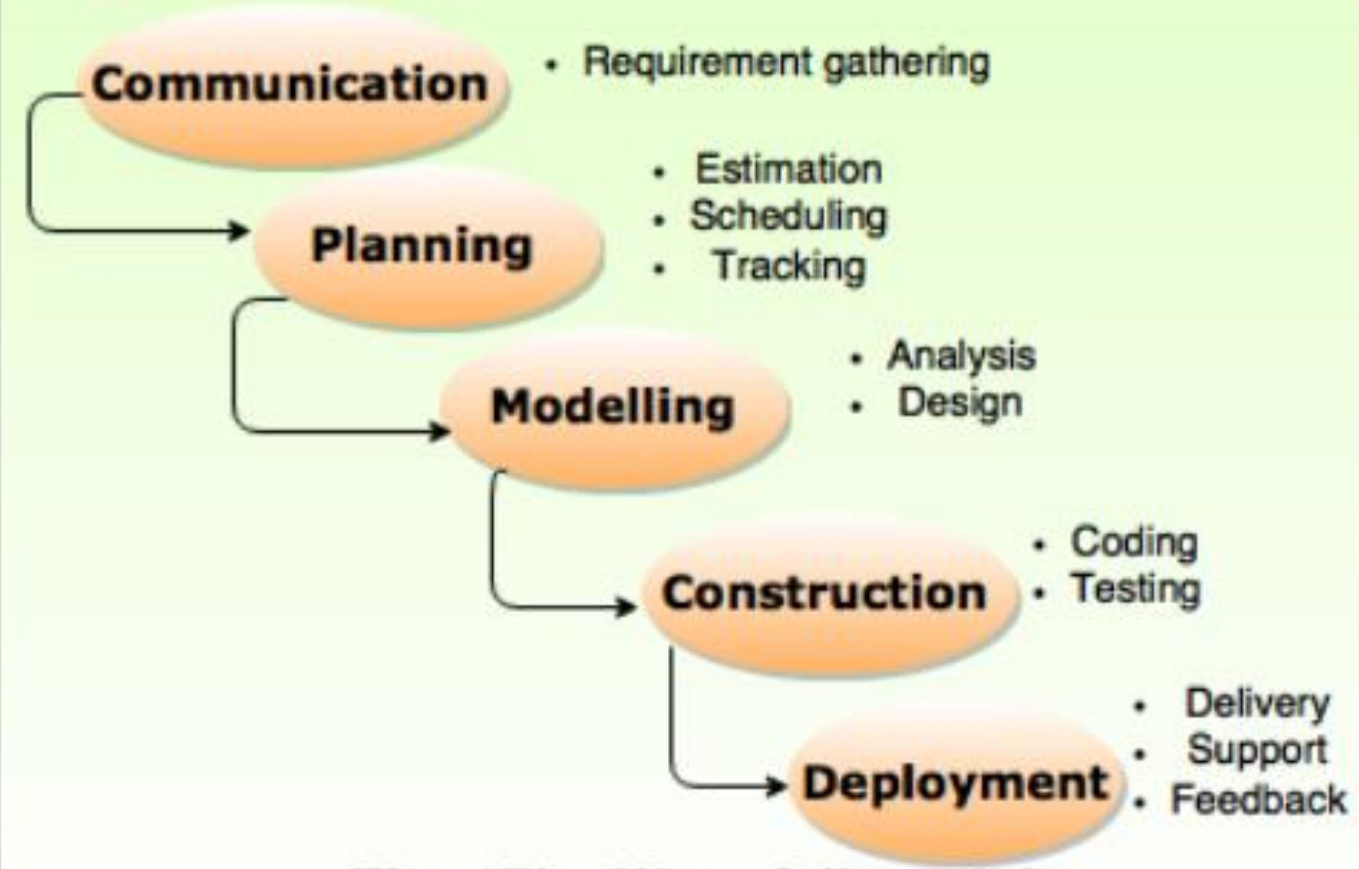


Fig. - The Waterfall model

Waterfall Model - Phases

- **Communication**

  The software development starts with the communication between customer and developer.

- **Planning**

  It consists of complete estimation, scheduling for project development.

- **Modeling**
  - Modeling consists of complete requirement analysis and the design of the project i.e algorithm, flowchart etc.
  - The algorithm is the step-by-step solution of the problem and the flow chart shows a complete flow diagram of a program.

Waterfall Model - Phases

- **Construction**
  - Construction consists of code generation and the testing part.
  - Coding part implements the design details using an appropriate programming language.
  - Testing is to check whether the flow of coding is correct or not.
  - Testing also checks that the program provides desired output.
- **Deployment**
  - Deployment step consists of delivering the product to the customer and taking feedback from them.
  - If the customer wants some corrections or demands for the additional capabilities then the change is required for improvement in the quality of the software.

Waterfall Model - Advantages

- The waterfall model is simple and easy to understand, to implement, and use.
- All the requirements are known at the beginning of the project, hence it is easy to manage.
- It avoids overlapping of phases because each phase is completed at once.
- This model works for small projects where the requirements are easily understood.
- This model is preferred for those projects where the quality is more important as compared to the cost of the project.

Waterfall Model - Disadvantages

- This model is not good for complex and object oriented projects.
- In this model, the changes are not permitted so it is not fit for moderate to high risk changes in project.
- It is a poor model for long duration projects.
- The problems with this model are uncovered, until the software testing.
- The amount of risk is high.

V Model

- V model is known as Verification and Validation model.
- This model is an extension of the waterfall model.
- In the life cycle of V-shaped model, processes are executed sequentially.
- Every phase completes its execution before the execution of next phase begins.

V Model

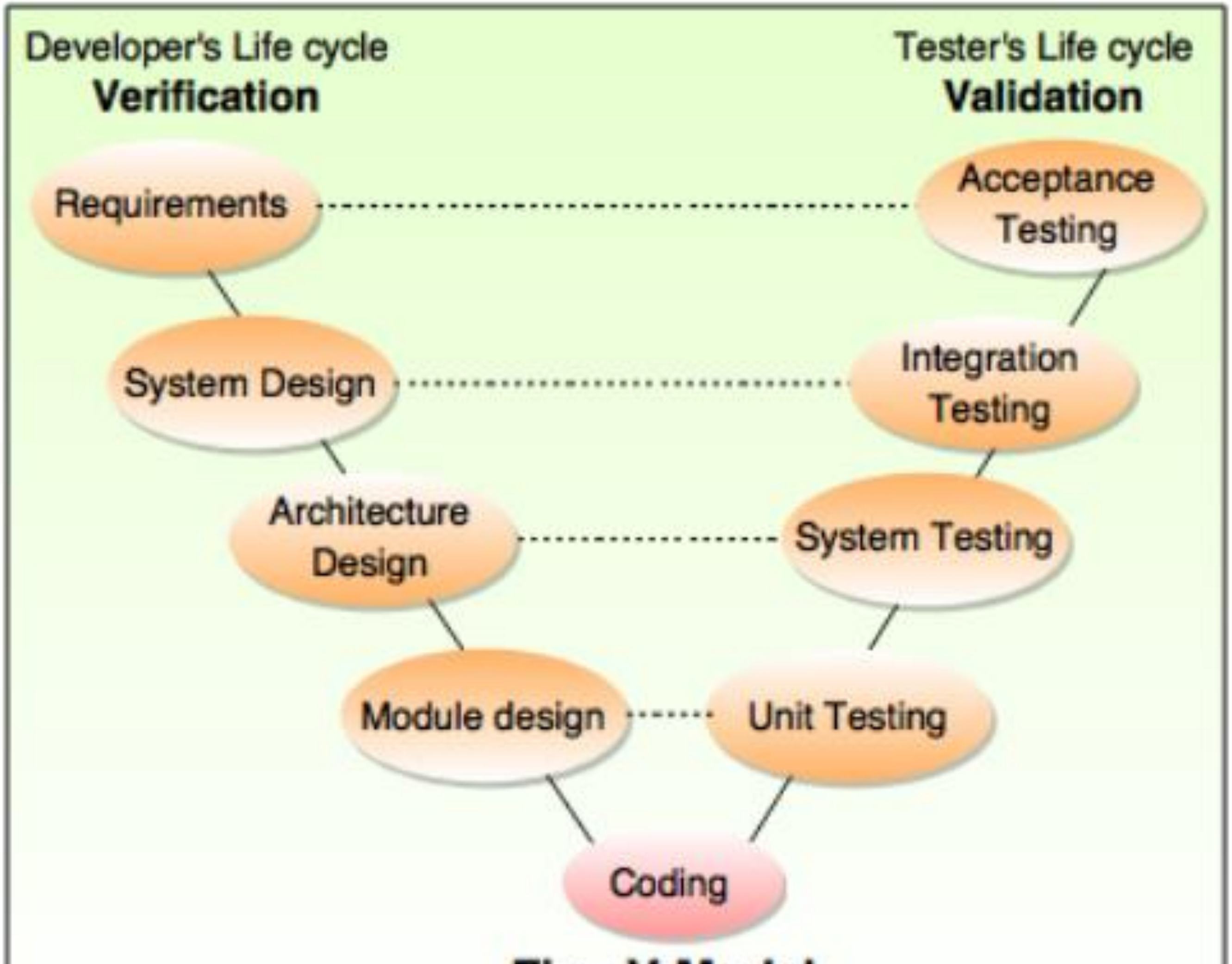
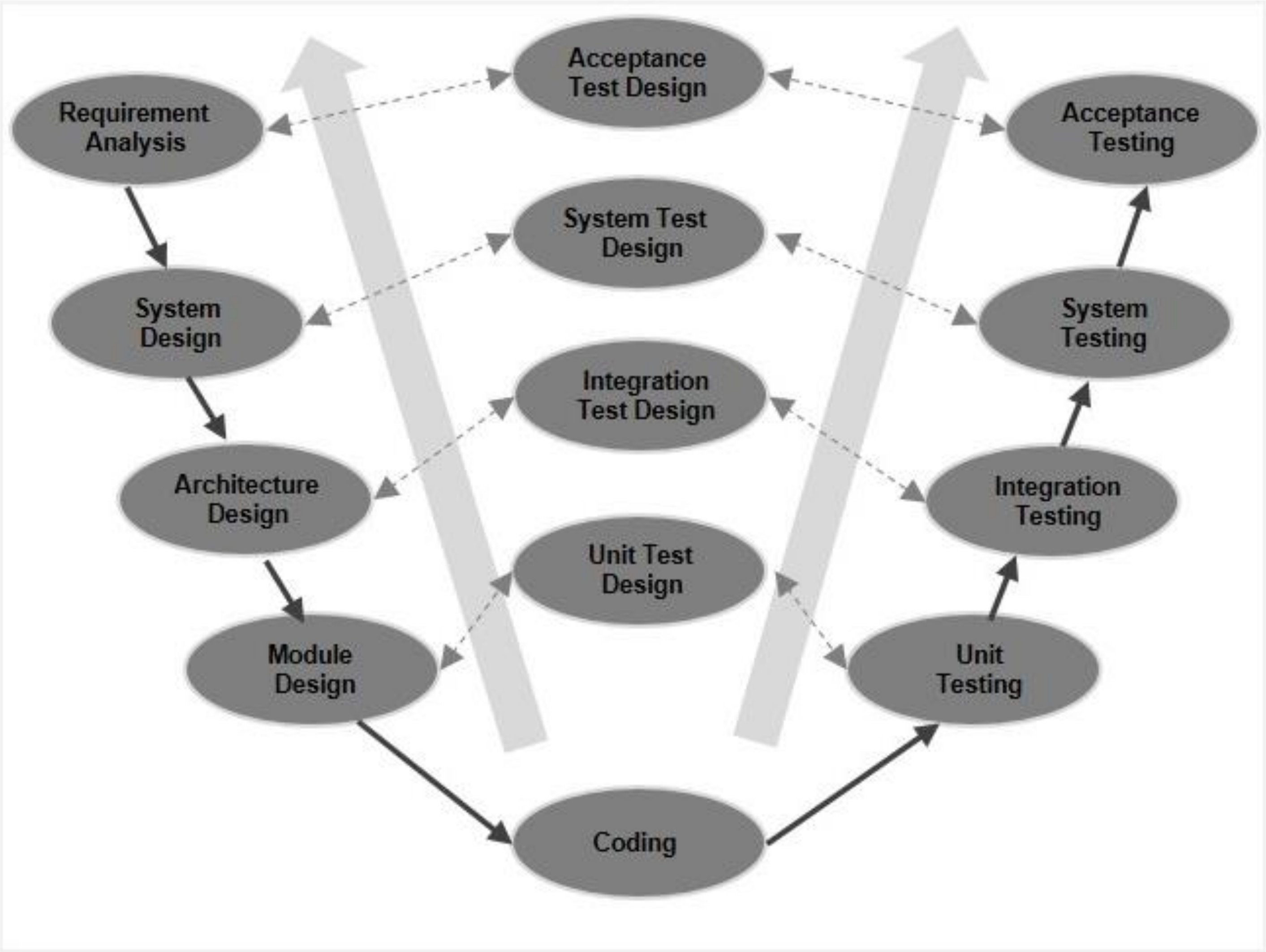
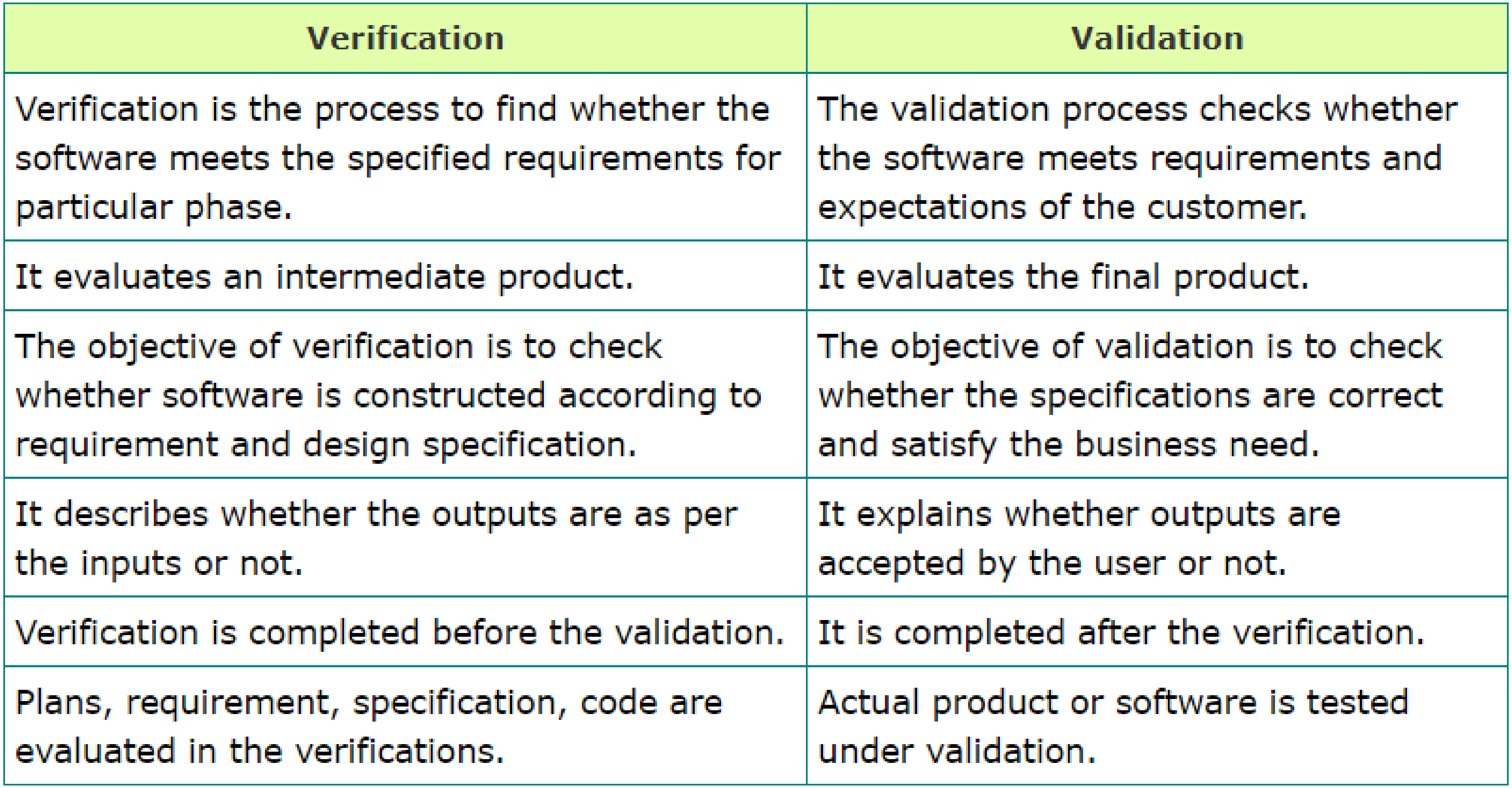


Fig.- V-Model

V Model


Requirement Analysis
Acceptance Test Design
Acceptance Testing
System Design
System Test Design
System Testing
Architecture Design
Integration Test Design
Integration Testing
Module Design
Unit Test Design
Unit Testing
Coding

V Model



Verification	Validation
Verification is the process to find whether the software meets the specified requirements for particular phase.	The validation process checks whether the software meets requirements and expectations of the customer.
It evaluates an intermediate product.	It evaluates the final product.
The objective of verification is to check whether software is constructed according to requirement and design specification.	The objective of validation is to check whether the specifications are correct and satisfy the business need.
It describes whether the outputs are as per the inputs or not.	It explains whether outputs are accepted by the user or not.
Verification is completed before the validation.	It is completed after the verification.
Plans, requirement, specification, code are evaluated in the verifications.	Actual product or software is tested under validation.

V Model – Verification Phases

- **Requirements**
  - The requirements of product are understood from the customers point of view to know their exact requirement and expectation.
  - The acceptance test design planning is completed at requirement stage because, business requirements are used as an input for acceptance testing.
- **System Design**
  - In system design, high level design of the software is constructed.
  - In this phase, we study how the requirements are implemented their technical use.
- **Architecture Design**
  - In architecture design, software architecture is created on the basis of high level design.
  - The module relationship and dependencies of module, architectural diagrams, database tables, technology details are completed in this phase.

V Model – Verification Phases

- **Module design**
  - In module phase, we separately design every module or the software components.
  - Finalize all the methods, classes, interfaces, data types etc.
  - Unit tests are designed in module design phase based on the internal module designs.
  - Unit tests are the vital part of any development process. They help to remove the maximum faults and errors at an early stage.
- **Coding Phase**
  - The actual code design of module designed in the design phase is grabbed in the coding phase.
  - On the basis of system and architecture requirements, we decide the best suitable programming language.
  - The coding is executed on the basis of coding guidelines and standards.

V Model – Verification Phases

- **Unit Testing**
  - Unit tests designed in the module design phase are executed on the code during this validation phase.
  - Unit testing is the testing at code level and helps eliminate bugs at an early stage, though all defects cannot be uncovered by unit testing.
- **Integration Testing**
  - Integration testing is associated with the architectural design phase.
  - Integration tests are performed to test the coexistence and communication of the internal modules within the system.

V Model – Verification Phases

- **System Testing**
  - System testing is directly associated with the system design phase.
  - System tests check the entire system functionality and the communication of the system under development with external systems.
  - Most of the software and hardware compatibility issues can be uncovered during this system test execution.
- **Acceptance Testing**
  - Acceptance testing is associated with the business requirement analysis phase and involves testing the product in user environment.
  - Acceptance tests uncover the compatibility issues with the other systems available in the user environment.
  - It also discovers the non-functional issues such as load and performance defects in the actual user environment.

V Model – Advantages

- V-model is easy and simple to use.
- Many testing activities i.e planning, test design are executed in the starting, it saves more time.
- Calculation of errors is done at the starting of the project hence, less chances of error occur at final phase of testing.
- This model is suitable for small projects where the requirements are easily understood.

V Model – Disadvantages

- V-model is not suitable for large and composite projects.
- If the requirements are not constant then this model is not acceptable.

Incremental Model

- The incremental model combines the elements of waterfall model and they are applied in an iterative fashion.
- The first increment in this model is generally a core product.
- Each increment builds the product and submits it to the customer for suggesting any modifications.
- The next increment implements the customer's suggestions and add additional requirements in the previous increment.
- This process is repeated until the product is completed. **For example**, the word-processing software is developed using the incremental model.

Incremental Model

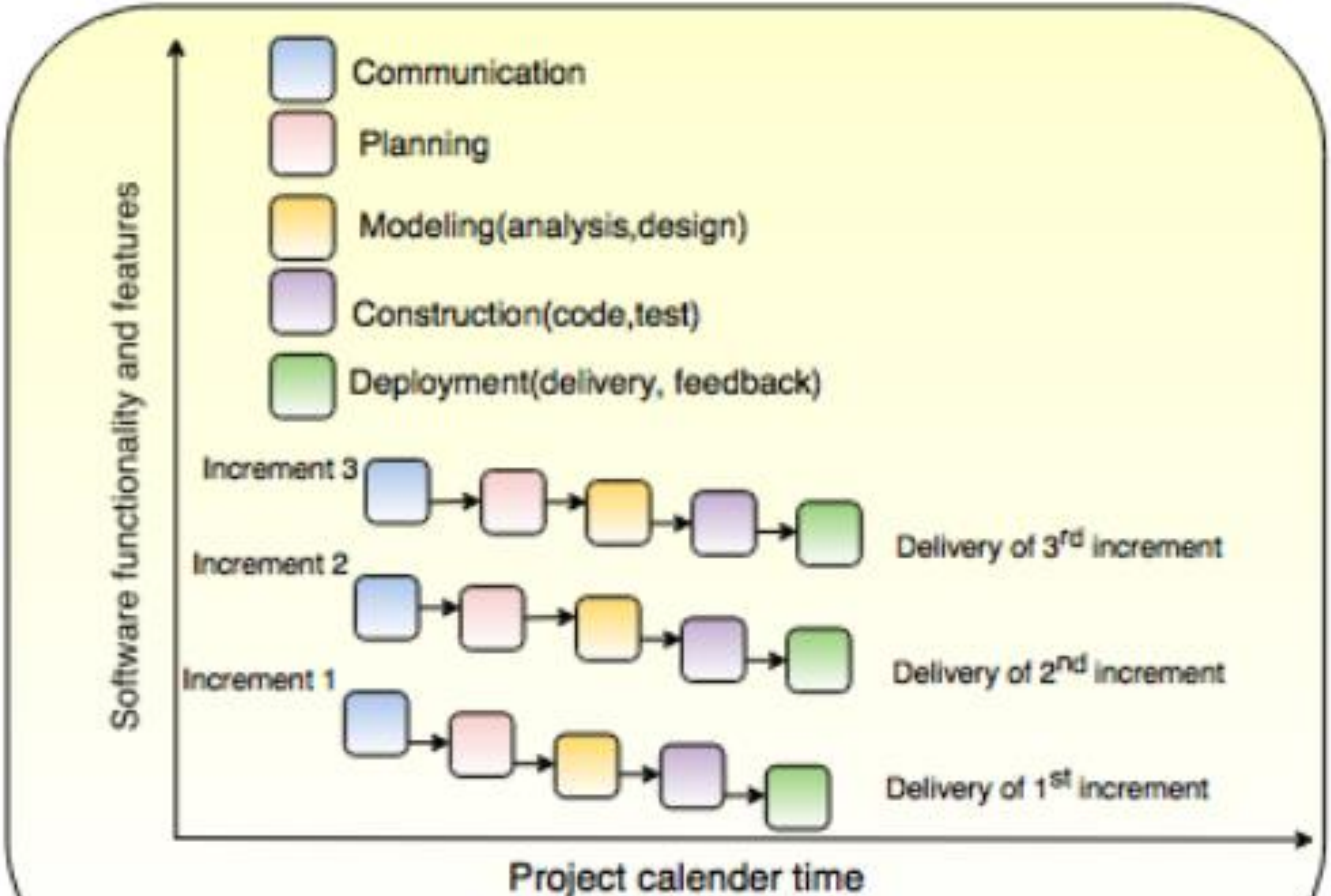

Communication
Planning
Modeling(analysis,design)
Construction(code,test)
Deployment(delivery, feedback)
Software functionality and features
Increment 3
Delivery of 3rd increment
Increment 2
Delivery of 2nd increment
Increment 1
Delivery of 1st increment
Project calender time

Fig. - Incremental Process Model

Incremental Model - Phases

- **Communication**

  The software development starts with the communication between customer and developer.

- **Planning**

  It consists of complete estimation, scheduling for project development.

- **Modeling**
  - Modeling consists of complete requirement analysis and the design of the project like algorithm, flowchart etc.
  - The algorithm is a step-by-step solution of the problem and the flow chart shows a complete flow diagram of a program.

Incremental Model - Phases

- **Construction**
  - Construction consists of code generation and the testing part.
  - Coding part implements the design details using an appropriate programming language.
  - Testing is to check whether the flow of coding is correct or not.
  - Testing also checks that the program provides desired output.
- **Deployment**
  - Deployment step consists of delivering the product to the customer and taking feedback from them.
  - If the customer wants some corrections or demands for the additional capabilities, then the change is required for improvement in the quality of the software.

Incremental Model - Advantages

- This model is flexible because the cost of development is low and initial product delivery is faster.
- It is easier to test and debug in the smaller iteration.
- The working software is generated quickly in the software life cycle.
- The customers can respond to its functionalities after every increment.

Incremental Model - Disadvantages

- The cost of the final product may cross the cost initially estimated.
- This model requires a very clear and complete planning.
- The planning of design is required before the whole system is broken into smaller increments.
- The demands of customer for the additional functionalities after every increment causes problem in the system architecture.

RAD Model

- RAD is a Rapid Application Development model.
- Using the RAD model, software product is developed in a short period of time.
- The initial activity starts with the communication between customer and developer.
- Planning depends upon the initial requirements and then the requirements are divided into groups.
- Planning is more important to work together on different modules.

RAD Model

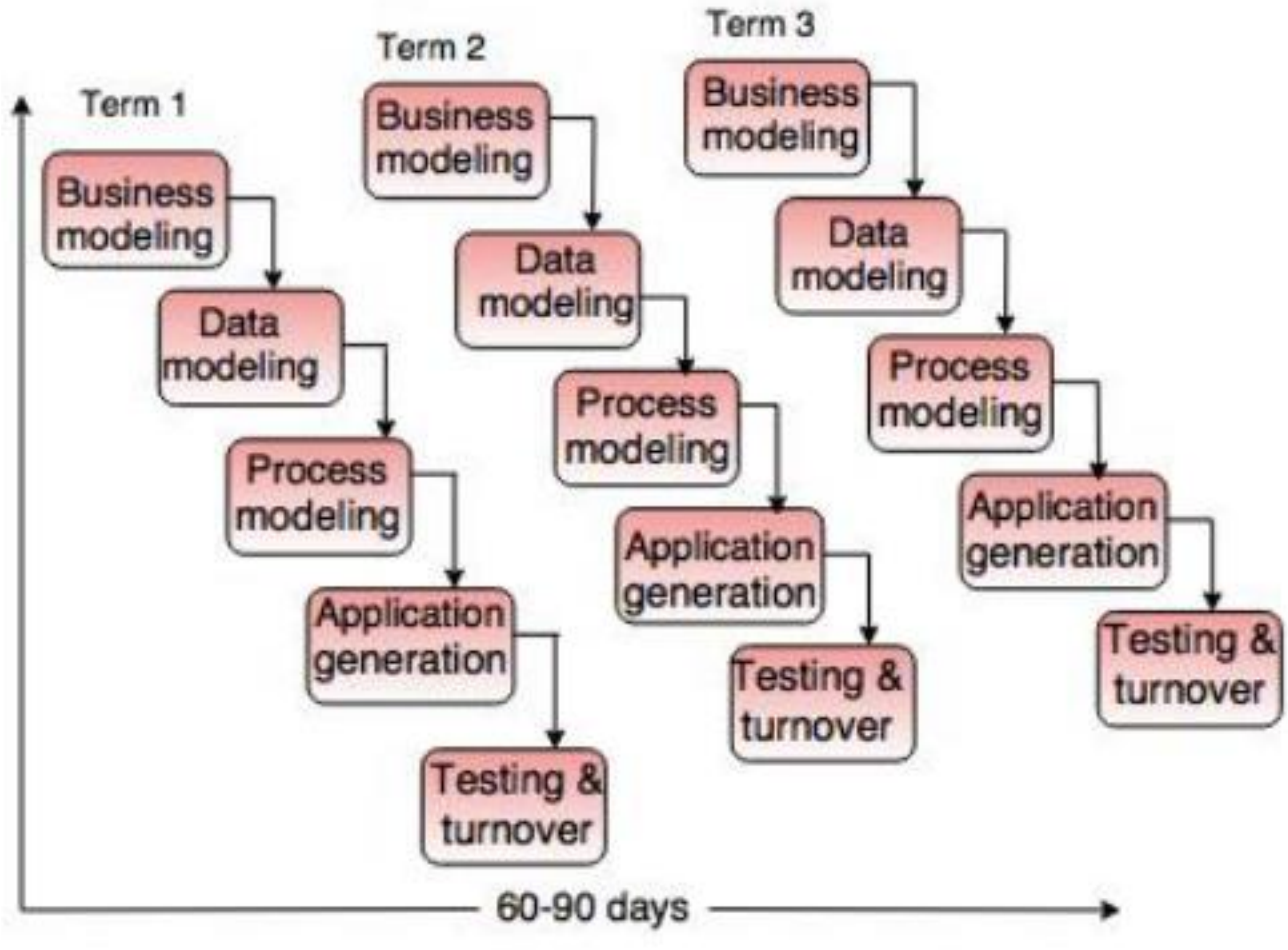


Fig. - RAD Model

RAD Model - Phases

- **Business Modeling**
  - Business modeling consists of the flow of information between various functions in the project.
  - **For example,** what type of information is produced by every function and which are the functions to handle that information.
  - It is necessary to perform complete business analysis to get the essential business information.
- **Data Modeling**
  - The information in the business modeling phase is refined into the set of objects and it is essential for the business.
  - The attributes of each object are identified and defined the relationship between objects.

RAD Model - Phases

- **Process Modeling**
  - The data objects defined in the data modeling phase are changed to fulfil the information flow to implement the business model.
  - The process description is created for adding, modifying, deleting or retrieving a data object.
- **Application Generation**
  - In the application generation phase, the actual system is built.
  - To construct the software the automated tools are used.
- **Testing and turnover**
  - The prototypes are independently tested after each iteration so that the overall testing time is reduced.
  - The data flow and the interfaces between all the components are fully tested. Hence, most of the programming components are already tested.

RAD Model - Advantages

- The process of application development and delivery are fast.
- This model is flexible, if any changes are required.
- Reviews are taken from the clients at the starting of the development hence there are lesser chances to miss the requirements.

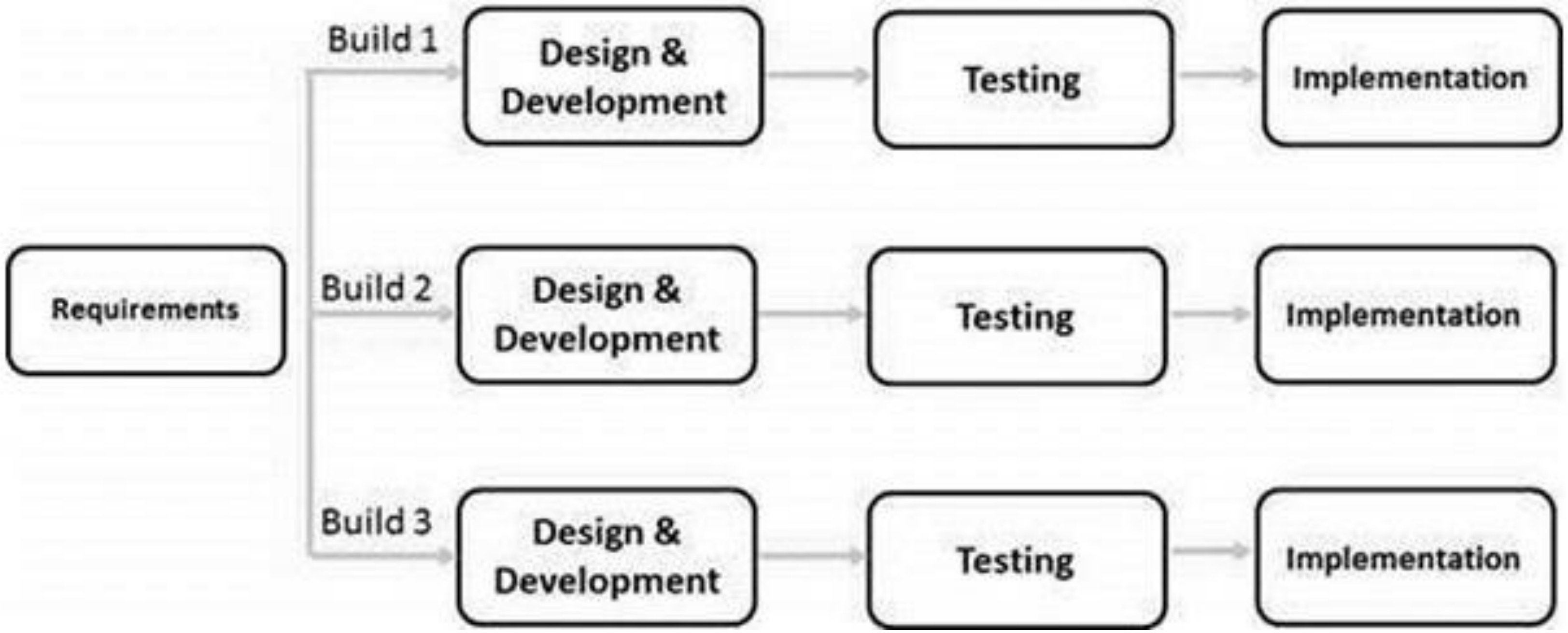
RAD Model - Disadvantages

- The feedback from the user is required at every development phase.
- This model is not a good choice for long term and large projects.

Iterative Model

- In Iterative model, the large application of software development is divided into smaller chunks and smaller parts of software which can be reviewed to recognize further requirements are implemented.
- This process is repeated to generate a new version of the software in each cycle of a model.
- With every iteration, development module goes through the phases i.e requirement, design, implementation and testing.
- These phases are repeated in iterative model in a sequence.

Iterative Model



Iterative Model - Phases

- **Requirement Phase**

  In this phase, the requirements for the software are assembled and analyzed. Generates a complete and final specification of requirements.

- **Design Phase**

  In this phase, a software solution meets the designed requirements which can be a new design or an extension of an earlier design.

- **Implementation and test phase**

  In this phase, coding for the software and test the code.

- **Evaluation**

  In this phase, software is evaluated, the current requirements are reviewed and the changes and additions in the requirements are suggested.

Iterative Model - Advantages

- Produces working software rapidly and early in the software life cycle.
- This model is easy to test and debug in a smaller iteration.
- It is less costly to change scope and requirements.

Iterative Model - Disadvantages

- The system architecture is costly.
- This model is not suitable for smaller projects.

Agile Model

- Agile model is a combination of incremental and iterative process models.
- This model focuses on the users satisfaction which can be achieved with quick delivery of the working software product.
- Agile model breaks the product into individual iterations.
- Every iteration includes cross functional teams working on different areas such as planning, requirements, analysis, design, coding, unit testing and acceptance testing.
- At the end of an iteration working product is shown to the users.

Agile Model

- With every increment, features are incremented and the final increment hold all the features needed by the customers.
- The iterations in agile process are shorter in duration which can vary from 2 weeks to 2 months.

Agile Model

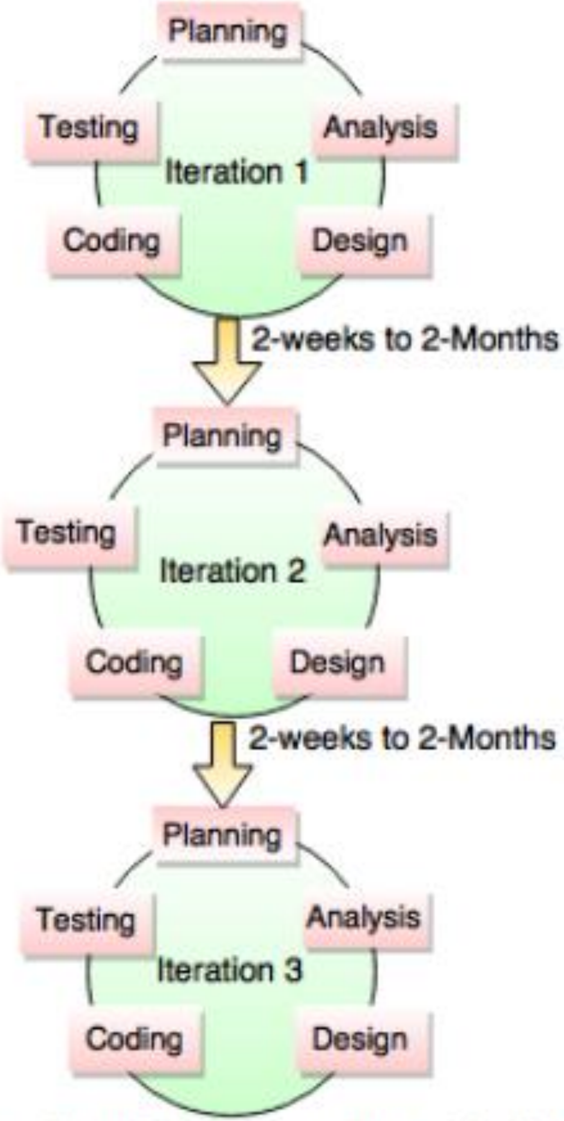


Fig. - Graphical Representation of Agile Model

Agile Model - Advantages

- Customers are satisfied because of quick and continuous delivery of useful software.
- Regular delivery of working software.
- Face to face interaction between the customers, developers and testers and it is best form of communication.
- Even the late changes in the requirement can be incorporated in the software.

Agile Model - Disadvantages

- It is totally depends on customer interaction. If the customer is not clear with their requirements, the development team can go in the wrong direction.
- Documentation is less, so the transfer of technology to the new team members is challenging.

Prototype Model

- Prototype is defined as first or preliminary form using which other forms are copied or derived.
- Prototype model is a set of general objectives for software.
- It does not identify the requirements like detailed input, output.
- It is software working model of limited functionality.
- In this model, working programs are quickly produced.

Prototype Model

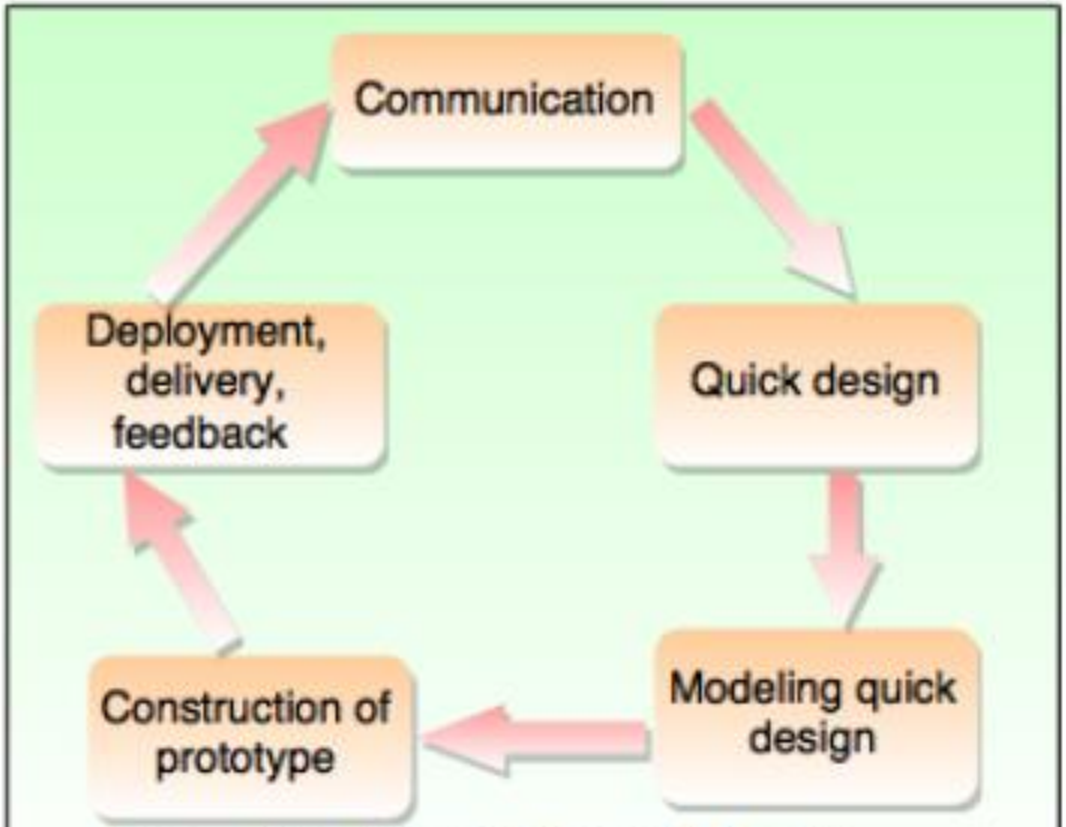


Fig. - The Prototyping Model

Prototype Model - Phases

- **Communication**

In this phase, developer and customer meet and discuss the overall objectives of the software.

- **Quick Design**
  - Quick design is implemented when requirements are known.
  - It includes only the important aspects i.e input and output format of the software.
  - It focuses on those aspects which are visible to the user rather than the detailed plan.
  - It helps to construct a prototype.
- **Modeling Quick Design**
  - This phase gives the clear idea about the development of software as the software is now constructed.
  - It allows the developer to better understand the exact requirements.

Prototype Model - Phases

- **Construction of Prototype**
  The prototype is evaluated by the customer itself.
- **Deployment, Delivery, Feedback**
  - If the user is not satisfied with current prototype then it is refined according to the requirements of the user.
  - The process of refining the prototype is repeated till all the requirements of users are met.
  - When the users are satisfied with the developed prototype then the system is developed on the basis of final prototype.

Prototype Model - Advantages

- In the development process of this model users are actively involved.
- The development process is the best platform to understand the system by the user.
- Earlier error detection takes place in this model.
- It gives quick user feedback for better solutions.
- It identifies the missing functionality easily. It also identifies the confusing or difficult functions.

Prototype Model - Disadvantages

- The client involvement is more and it is not always considered by the developer.
- It is a slow process because it takes more time for development.
- Many changes can disturb the rhythm of the development team.
- It is a throw away prototype when the users are confused with it.

Spiral Model

- It is a combination of prototype and sequential or waterfall model.
- This model is a risk driven process model.
- Every phase in the Spiral model starts with a design goal and ends with the client review.
- The development team in this model begins with a small set of requirements and for the set of requirements team goes through each development phase.
- The development team adds the functionality in every spiral till the application is ready.

Spiral Model

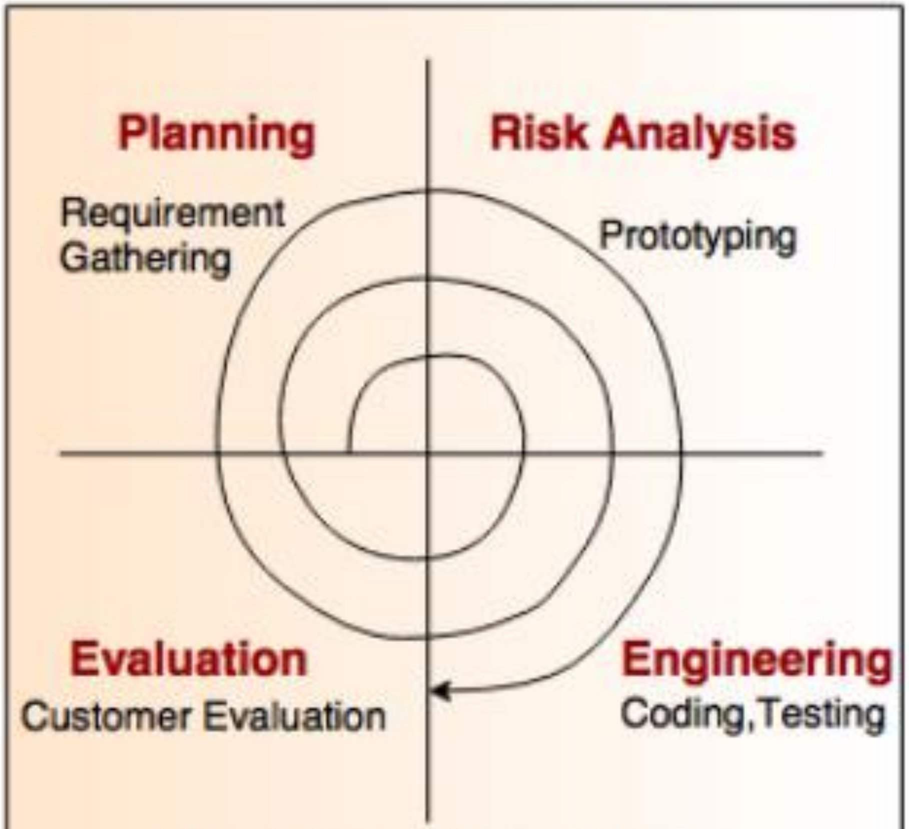


Fig. - The Spiral Model

Spiral Model - Phases

- **Planning**
  - This phase, studies and collects the requirements for continuous communication between the customer and system analyst.
  - It involves estimating the cost and resources for the iteration.
- **Risk Analysis**

  This phase, identifies the risk and provides the alternate solutions if the risk is found.
- **Engineering**

  In this phase, actual development i.e coding of the software is completed. Testing is completed at the end of the phase.
- **Evaluation**

  Get the software evaluated by the customers. They provide the feedback before the project continues to the next spiral.

Spiral Model - Advantages

- It reduces high amount of risk.
- It is good for large and critical projects.
- It gives strong approval and documentation control.
- In spiral model, the software is produced early in the life cycle process.

Spiral Model - Disadvantages

- It can be costly to develop a software model.
- It is not used for small projects.