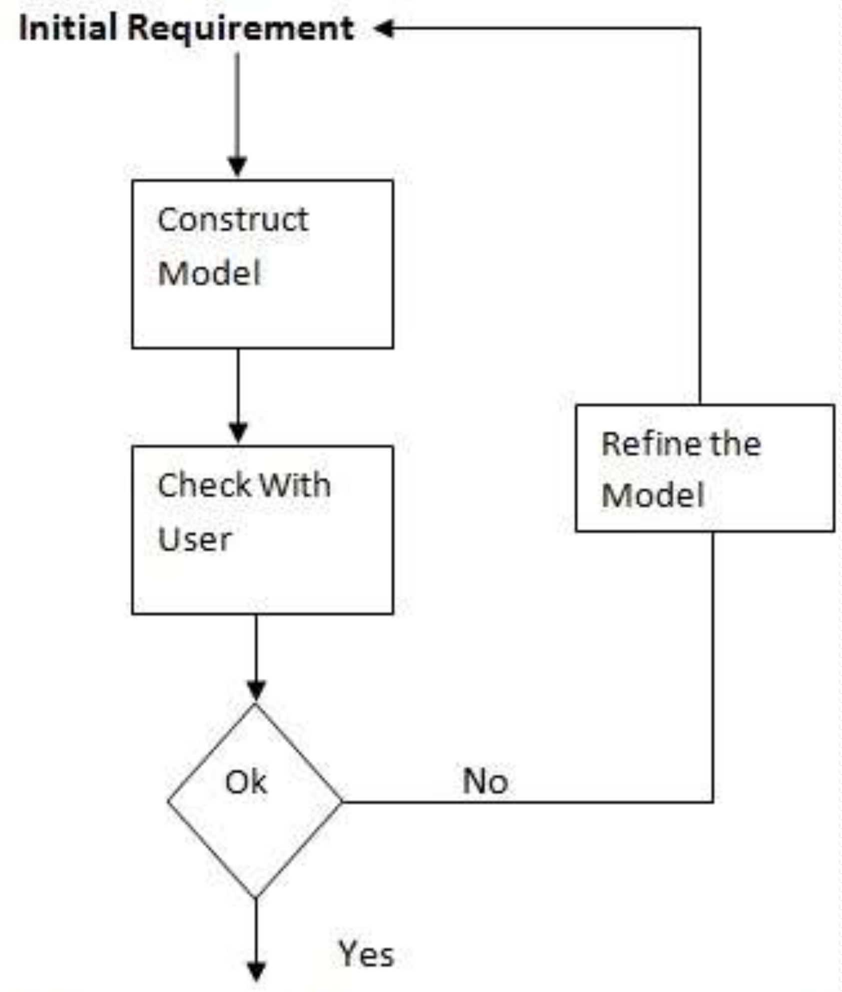
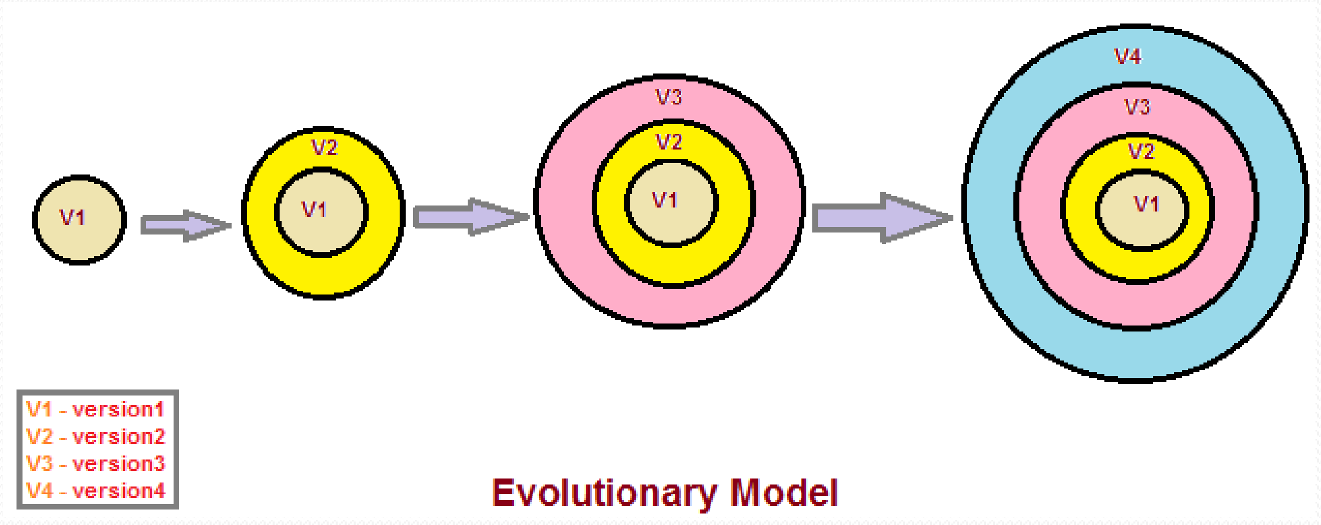
Process Models

Evolutionary Process Models

- **Evolutionary model** is a combination of Iterative and Incremental model of software development life cycle.
- Incremental model first implement a few basic features and deliver to the customer. Then build the next part and deliver it again and repeat this step until the desired system is fully realized . No long term plans are made.
- Iterative model main advantage is its feedback process in every phase
- Developers build a prototype during the requirements phase
- Prototype is evaluated by end users
- Users give corrective feedback
- Developers further refine the prototype
- When the user is satisfied, the prototype code is brought up to the standards needed for a final product.
- Also known as "design a little, build a little, test a little, deploy a little model".



Complete Procedure of Evolutionary process model



- In *evolutionary development model* or evolutionary *life cycle model*, each of the version (V1, V2, V3, V4) as shown in the diagram will be released with new functionality or functionalities and added to the previous version(s).

- For example, Version V2 will have more functional capability or capabilities as compared to its previous version V1 but will exhibit less capability in terms of functionality from V3. With the incremental pattern of new version release, the software product is built over the period of time to exhibit the functionality of fully developed system.

The various phases of this model are:

1. Requirement Definition: is a step of thorough analysis used to create an initial requirements and specifications for the software.

2. Model Construction: is done having the phases design, coding and testing.

3. User Evaluation: is to check for its satisfaction and completion of work.

4. Iteration: is refining the model. If needed then any feedback can be given by the user that is implemented.

At last the end product is the working system.

Application of Evolutionary Model:

- It is used in large projects where you can easily find modules for incremental implementation.
- Evolutionary model is commonly used when the customer wants to start using the core features instead of waiting for the full software.
- Evolutionary model is also used in object oriented software development because the system can be easily portioned into units in terms of objects.

Necessary conditions for implementing this model:-

- Customer needs are clear and been explained in deep to the developer team.
- There might be small changes required in separate parts but not a major change.
- As it requires time, so there must be some time left for the market constraints.
- Risk is high and continuous targets to achieve and report to customer repeatedly.
- It is used when working on a technology is new and requires time to learn.

Advantages:

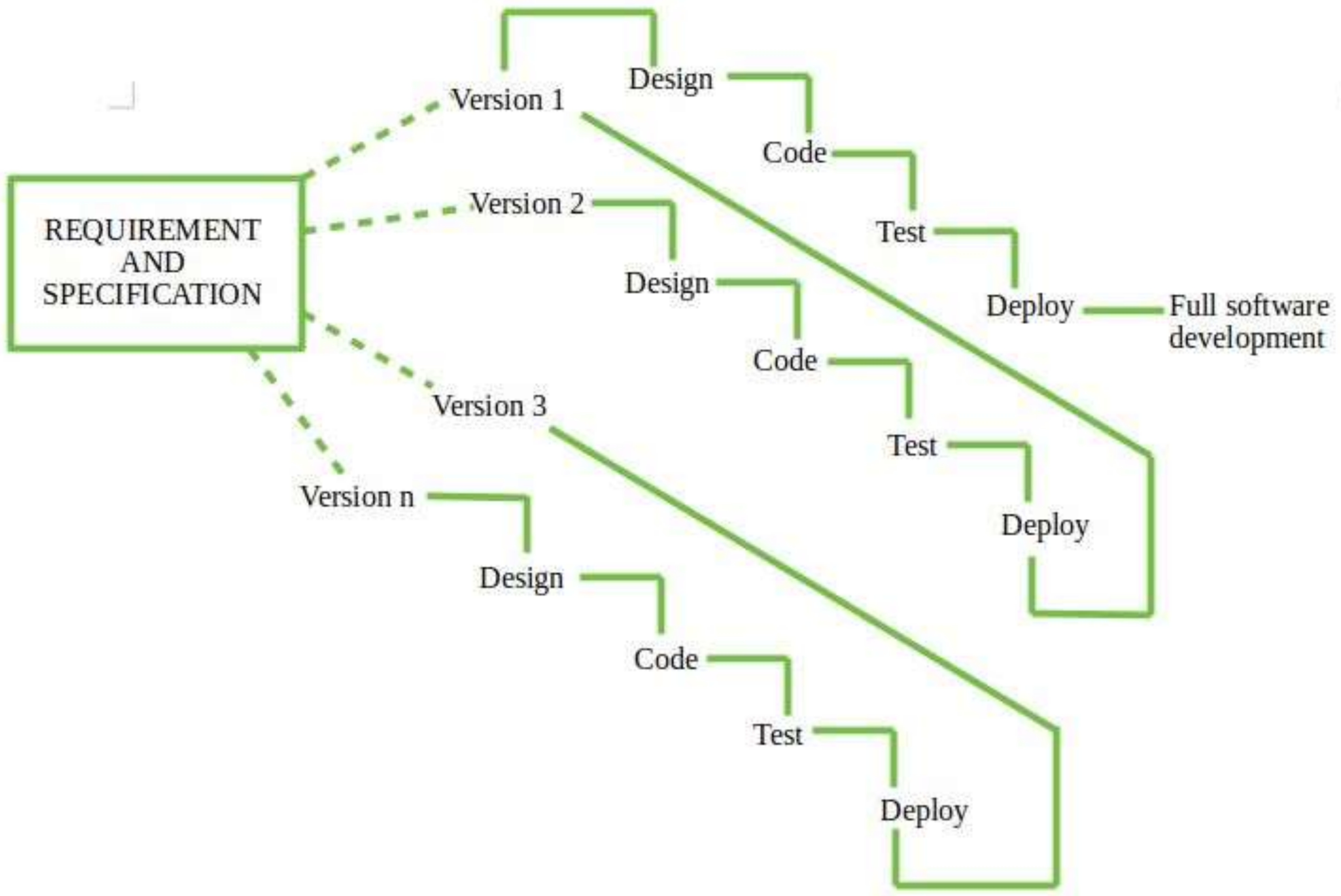
- In evolutionary model, a user gets a chance to experiment partially developed system.
- It reduces the error because the core modules get tested thoroughly.

Disadvantages:

- Sometimes it is hard to divide the problem into several versions that would be acceptable to the customer which can be incrementally implemented and delivered.

Incremental Model

- Incremental Model is a process of software development where requirements divided into multiple standalone modules of the software development cycle.
- In this model, each module goes through the requirements, design, implementation and testing phases.
- Every subsequent release of the module adds function to the previous release. The process continues until the complete system achieved.


REQUIREMENT AND SPECIFICATION
Version 1
Design
Code
Test
Deploy
Full software development
Version 2
Design
Code
Test
Deploy
Version 3
Version n
Design
Code
Test
Deploy

Incremental SDLC Model

- Construct a partial implementation of a total system
- Then slowly add increased functionality
- The incremental model prioritizes requirements of the system and then implements them in groups.
- Each subsequent release of the system adds function to the previous release, until all designed functionality has been implemented.

Incremental Model Strengths

- Develop high-risk or major functions first
- Each release delivers an operational product
- Customer can respond to each build
- Uses “divide and conquer” breakdown of tasks
- Lowers initial delivery cost
- Initial product delivery is faster
- Customers get important functionality early
- Risk of changing requirements is reduced

Incremental Model Weaknesses

- Requires good planning and design
- Requires early definition of a complete and fully functional system to allow for the definition of increments
- Well-defined module interfaces are required (some will be developed long before others)
- Total cost of the complete system is not lower

When to use the Incremental Model

When to use the Incremental Model

- Risk, funding, schedule, program complexity, or need for early realization of benefits.
- Most of the requirements are known up-front but are expected to evolve over time
- A need to get basic functionality to the market early
- On projects which have lengthy development schedules
- On a project with new technology

Spiral Model

Spiral model is one of the most important Software Development Life Cycle models, which provides support for **Risk Handling**.

The *spiral model*, originally proposed by Boehm [BOE88], is an evolutionary software process model that couples the iterative nature of prototyping with the controlled and systematic aspects of the linear sequential model.

The spiral model has four phases. A software project repeatedly passes through these phases in iterations called Spirals.

- Customer communication
- Planning
- Risk analysis
- Engineering
- Construction and release
- Customer evaluation

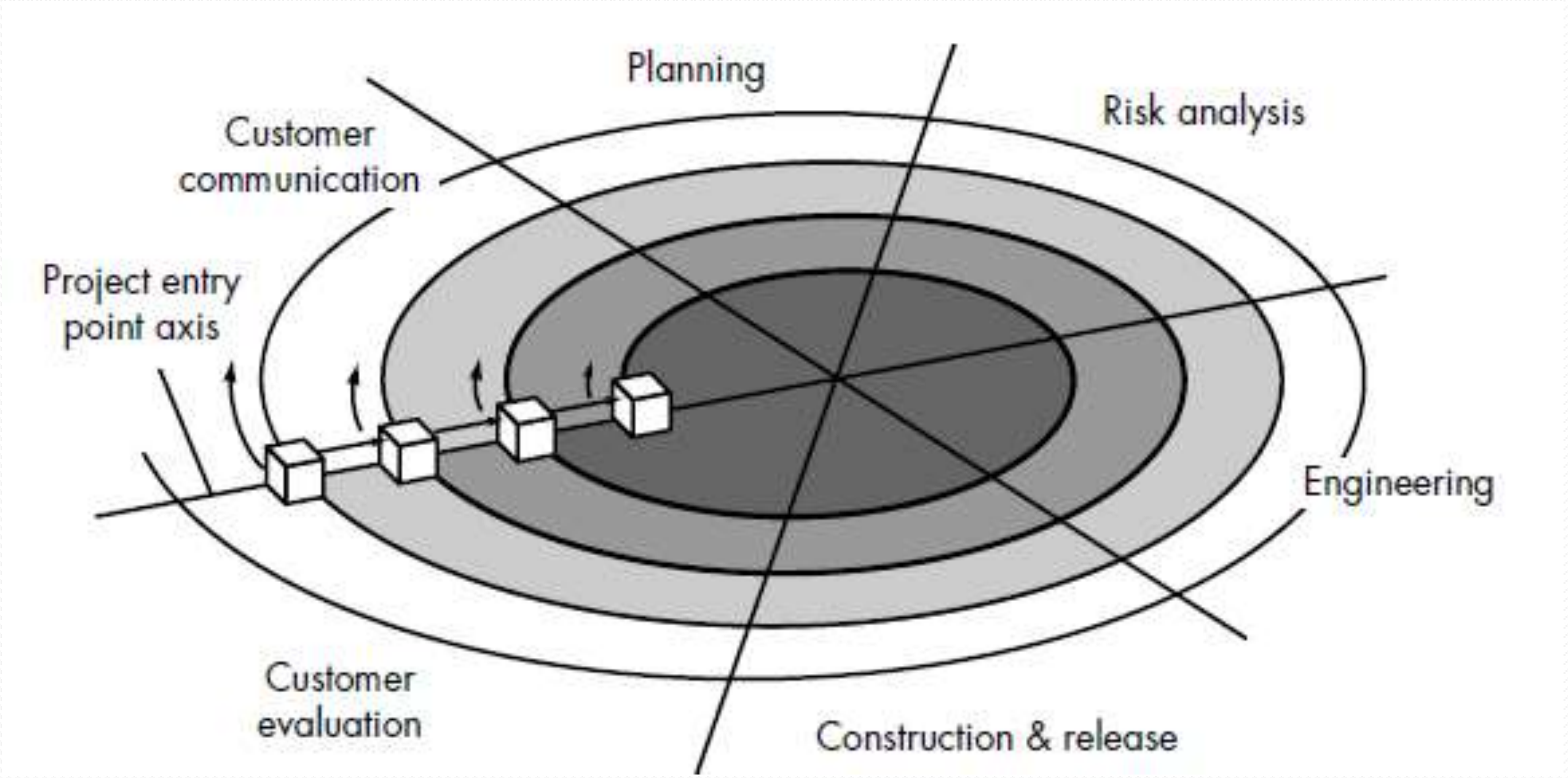


Figure : Spiral Model

- **Customer communication**—tasks required to establish effective communication between developer and customer.
- **Planning**—tasks required to define resources, timelines, and other project related information.
- **Risk analysis**—tasks required to assess both technical and management risks.
- **Engineering**—tasks required to build one or more representations (Prototype) of the application.
- **Construction and release**—tasks required to construct, test, install, and provide user support (e.g., documentation and training).
- **Customer evaluation**—tasks required to obtain customer feedback based on evaluation of the software representations created during the engineering stage and implemented during the installation stage.

Advantages of Spiral Model:

Below are some advantages of the Spiral Model.

- **Risk Handling:** The projects with many unknown risks that occur as the development proceeds, in that case, Spiral Model is the best development model to follow due to the risk analysis and risk handling at every phase.
- **Good for large projects:** It is recommended to use the Spiral Model in large and complex projects.
- **Flexibility in Requirements:** Change requests in the Requirements at later phase can be incorporated accurately by using this model.
- **Customer Satisfaction:** Customer can see the development of the product at the early phase of the software development and thus, they habituated with the system by using it before completion of the total product.

Disadvantages of Spiral Model:

Below are some main disadvantages of the spiral model.

- **Complex:** The Spiral Model is much more complex than other SDLC models.
- **Expensive:** Spiral Model is not suitable for small projects as it is expensive.
- **Too much dependability on Risk Analysis:** The successful completion of the project is very much dependent on Risk Analysis. Without very highly experienced experts, it is going to be a failure to develop a project using this model.
- **Difficulty in time management:** As the number of phases is unknown at the start of the project, so time estimation is very difficult.