Software Engineering (ESC501)

- Prof. Poulami Dutta

Intended Learning Outcomes (ILOs)

- Explain what a prototype is.
- Explain why and when a prototype needs to be developed during software development.
- Identify the situations in which one would prefer to build a prototype.
- State the activities carried out during each phase of a spiral model.
- Identify circumstances under which spiral model should be used for software development.
- Tailor a development process to a specific project.
- Explain the advantages and disadvantages of the V-model.
- Understand the goals and objectives of the RAD model.
- Comparative study of the models.

Prototyping Model

- Before starting actual development,
  - a working prototype of the system should first be built.
- A prototype is a toy implementation of a system:
  - limited functional capabilities,
  - low reliability,
  - inefficient performance.

Reasons for developing a prototype

- Illustrate to the customer:
  - input data formats, messages, reports, or interactive dialogs.
- Examine technical issues associated with product development:
  - Often major design decisions depend on issues like:
    - response time of a hardware controller,
    - efficiency of a sorting algorithm, etc.

Prototyping Model (contd.)

- The third reason for developing a prototype is:
  - it is impossible to ``get it right'' the first time,
  - we must plan to throw away the first product
    - if we want to develop a good product.

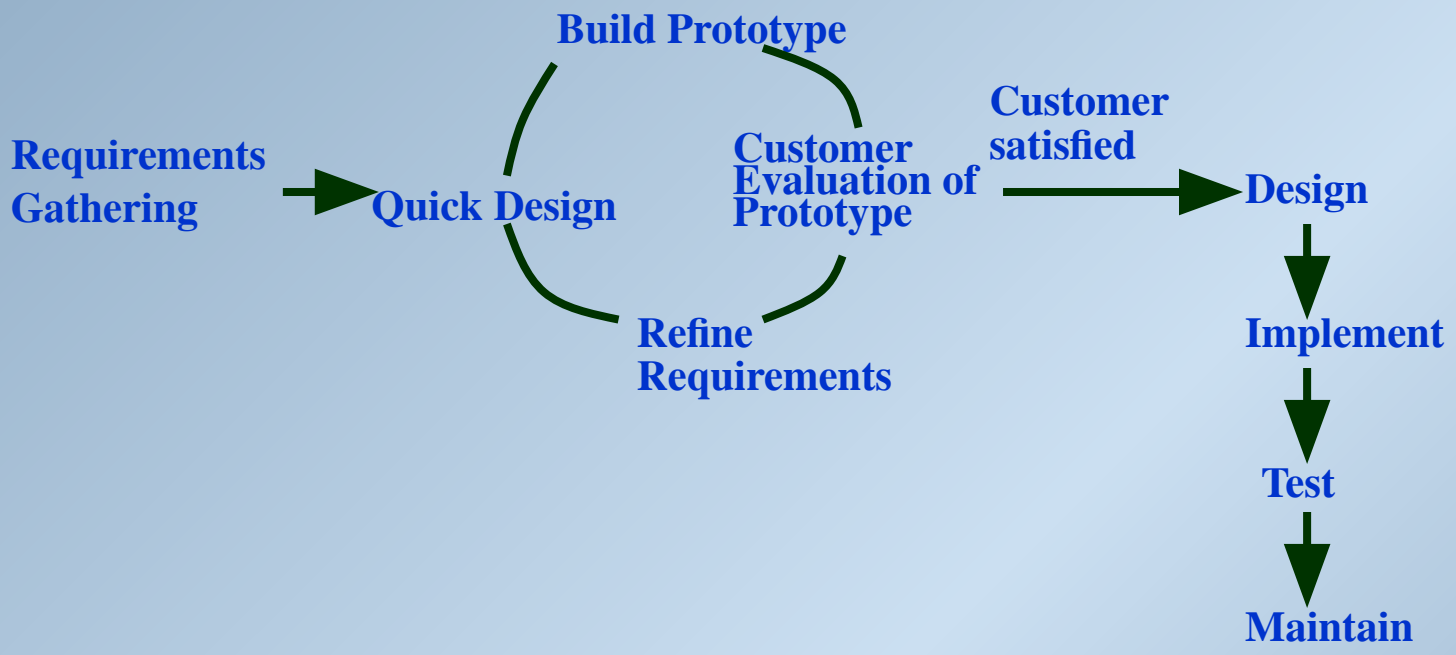
Prototyping Model (contd.)

- Start with approximate requirements.
- Carry out a quick design.
- Prototype model is built using several short-cuts:
  - Short-cuts might involve using inefficient, inaccurate, or dummy functions.
    - A function may use a table look-up rather than performing the actual computations.

Prototyping Model (contd.)

- The developed prototype is submitted to the customer for his evaluation:
  - Based on the user feedback, requirements are refined.
  - This cycle continues until the user approves the prototype.
- The actual system is developed using the classical waterfall approach.

Prototyping Model (contd.)



Prototyping Model (contd.)

- Requirements analysis and specification phase becomes redundant:
  - final working prototype (with all user feedbacks incorporated) serves as an animated requirements specification.
- Design and code for the prototype is usually thrown away:
  - However, the experience gathered from developing the prototype helps a great deal while developing the actual product.

Prototyping Model (contd.)

- Even though construction of a working prototype model involves additional cost --- overall development cost might be lower for:
  - systems with unclear user requirements,
  - systems with unresolved technical issues.
- Many user requirements get properly defined and technical issues get resolved:
  - these would have appeared later as change requests and resulted in incurring massive redesign costs.

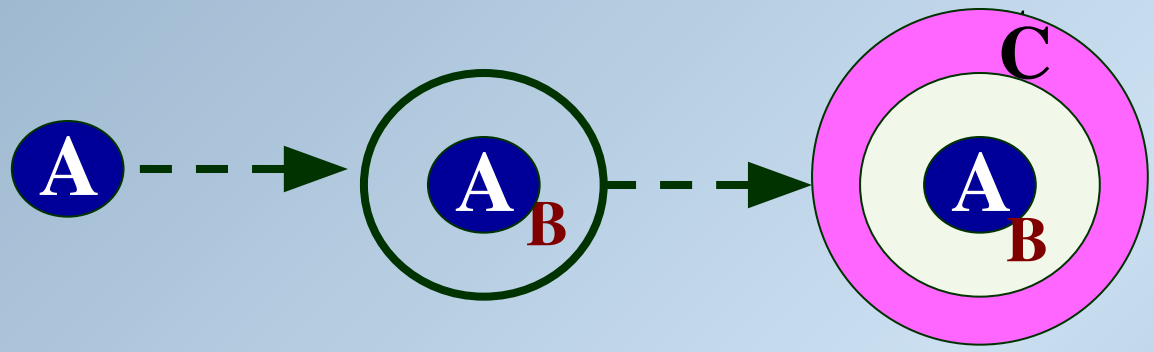
Evolutionary Model

- Evolutionary model (aka successive versions or incremental model):
  - The system is broken down into several modules which can be incrementally implemented and delivered.
- First develop the core modules of the system.
- The initial product skeleton is refined into increasing levels of capability:
  - by adding new functionalities in successive versions.

Evolutionary Model (CONT.)

- Successive version of the product:
  - functioning systems capable of performing some useful work.
  - A new release may include new functionality:
    - also existing functionality in the current release might have been enhanced.

Evolutionary Model (CONT.)



Advantages of Evolutionary Model

- Users get a chance to experiment with a partially developed system:
  - much before the full working version is released.
- Helps finding exact user requirements:
  - much before fully working system is developed.
- Core modules get tested thoroughly:
  - reduces chances of errors in final product.

Disadvantages of Evolutionary Model

- Often, difficult to subdivide problems into functional units:
  - which can be incrementally implemented and delivered.
  - evolutionary model is useful for very large problems,
    - where it is easier to find modules for incremental implementation.

Evolutionary Model with Iteration

- Many organizations use a combination of iterative and incremental development:
  - a new release may include new functionality.
  - existing functionality from the current release may also have been modified.

Evolutionary Model with iteration

- Several advantages:
  - Training can start on an earlier release.
    - customer feedback taken into account.
  - Markets can be created:
    - for functionality that has never been offered.
  - Frequent releases allow developers to fix unanticipated problems quickly.

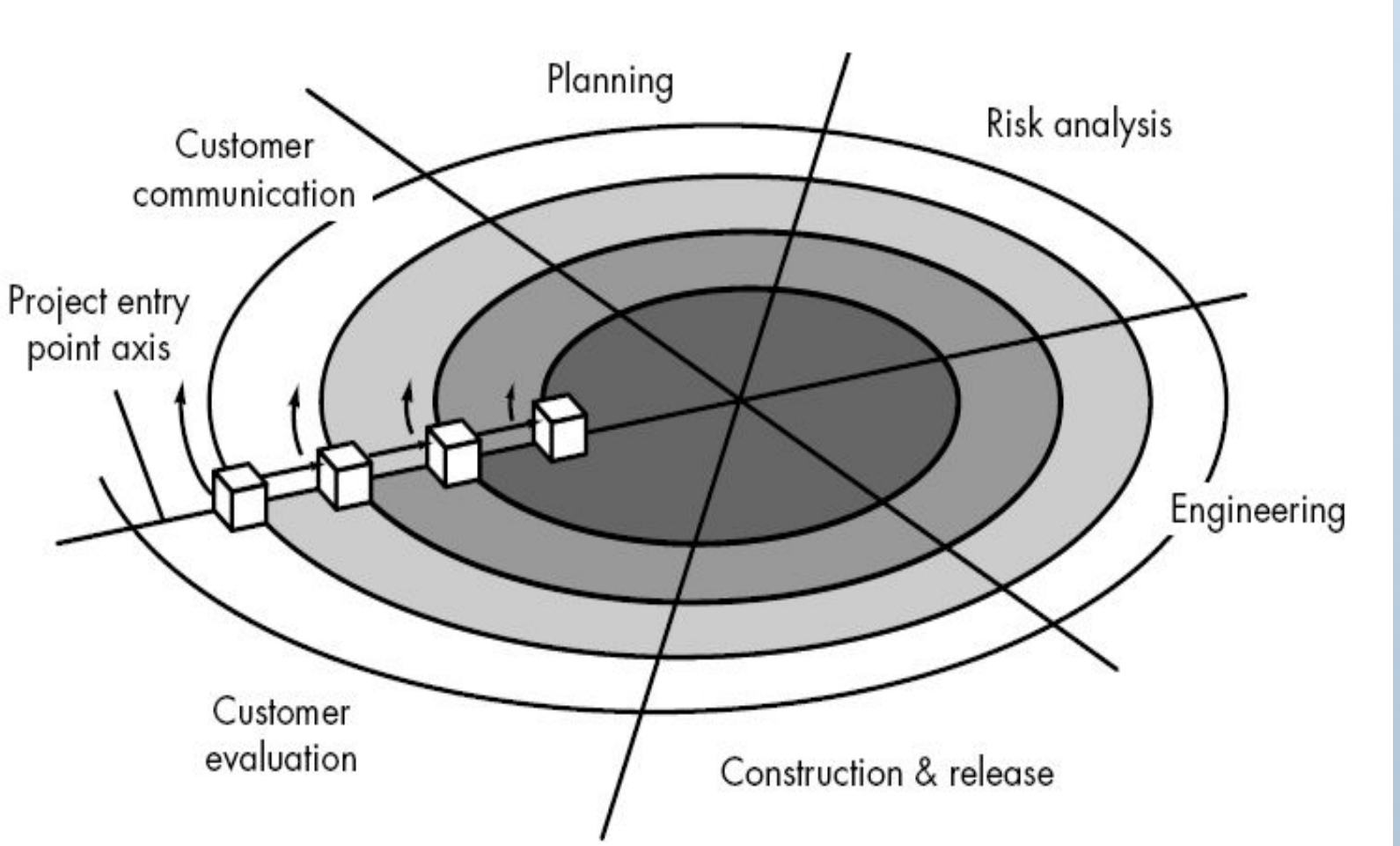
Spiral Model

- Proposed by Boehm in 1988.
- 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.
- There are no fixed phases in this model, the phases shown in the figure are just examples.
  - Each phase in this model is split into four sectors (or quadrants)

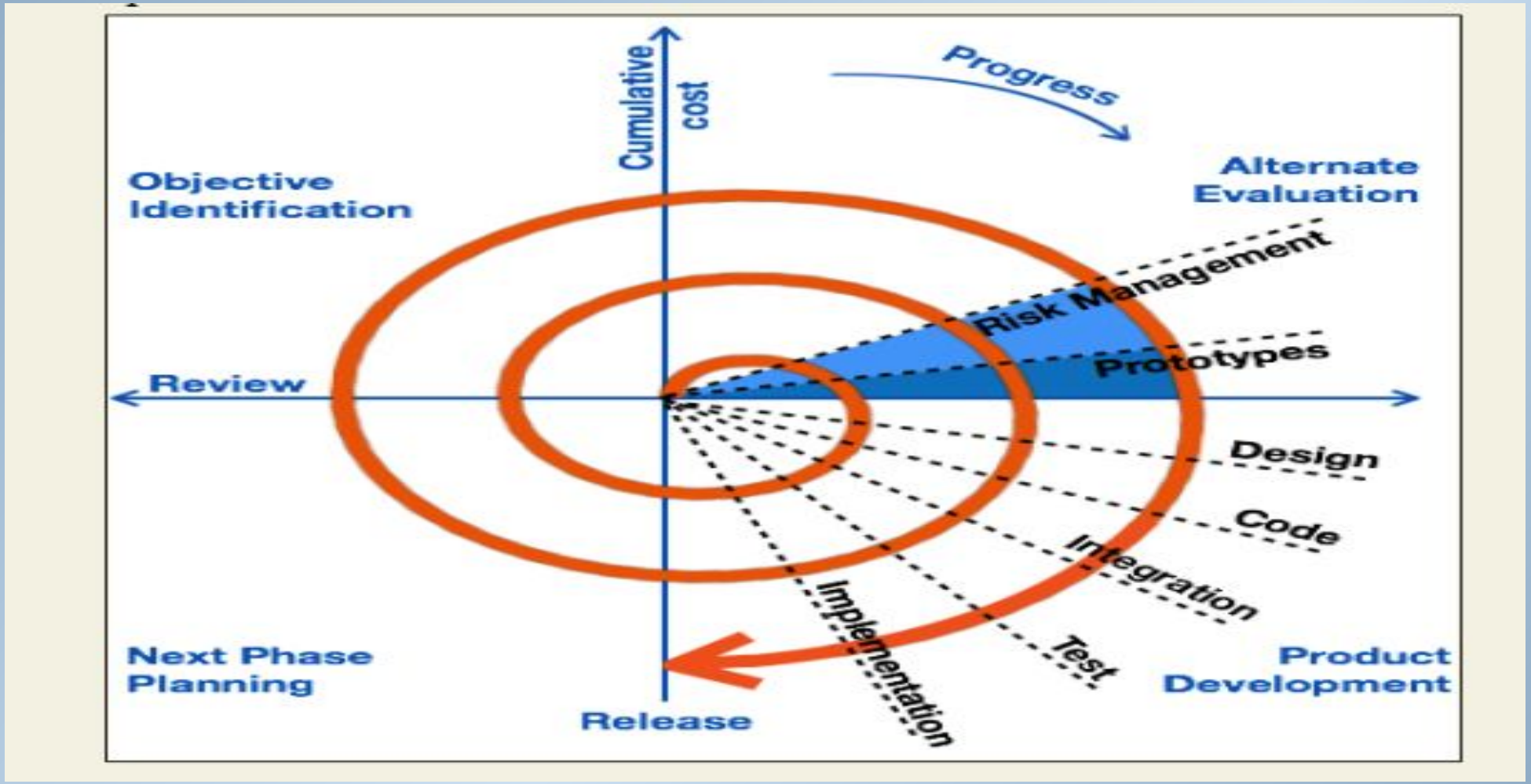
Spiral Model (CONT.)

- The team must decide:
  - how to structure the project into phases.
- Start work using some generic model:
  - add extra phases,
    - for specific projects or when problems are identified during a project.

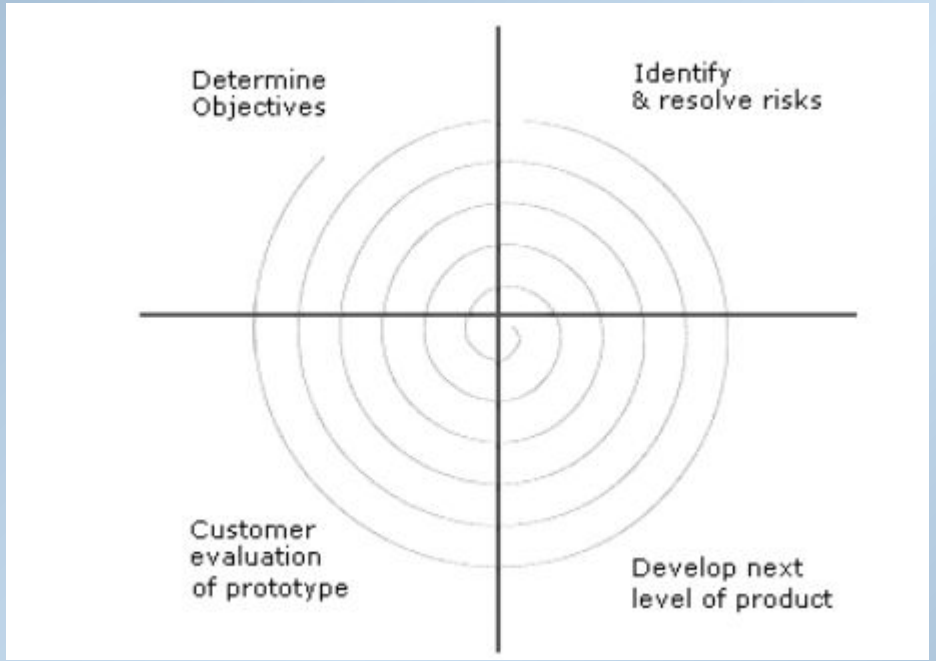
Spiral Model



Spiral Model



Spiral Model



Spiral Model (CONT.)

- **First quadrant (Objective Setting)**

  • During the first quadrant, it is needed to identify the objectives of the phase.

  • Examine the risks associated with these objectives.

- **Second Quadrant (Risk Assessment and Reduction)**

  • A detailed analysis is carried out for each identified project risk.

  • Steps are taken to reduce the risks. For example, if there is a risk that the requirements are inappropriate, a prototype system may be developed.

Spiral Model (CONT.)

- **Third Quadrant (Development and Validation)**

  • Develop and validate the next level of the product after resolving the identified risks.

- **Fourth Quadrant (Review and Planning)**

  • Review the results achieved so far with the customer and plan the next iteration around the spiral.

  • Progressively more complete version of the software gets built with each iteration around the spiral.

Spiral Model (CONT.)

- It works on:
  - Concept development projects.
  - New product development projects.
  - Product enhancement projects.
  - Product maintenance projects.

Objective Setting

- Identify objectives of the phase,
- Examine the risks associated with these objectives.
  - Risk:
    - any adverse circumstance that might hamper successful completion of a software project.
- Find alternate solutions possible.

Risk Assessment and Reduction

- For each identified project risk,
  - a detailed analysis is carried out.
- Steps are taken to reduce the risk.
- For example, if there is a risk that the requirements are inappropriate:
  - a prototype system may be developed.

Spiral Model (CONT.)

- Development and Validation:
  - develop and validate the next level of the product.
- Review and Planning:
  - Review the results achieved so far with the customer and plan the next iteration around the spiral.
- With each iteration around the spiral:
  - progressively more complete version of the software gets built.

Spiral Model as a meta model

- **Subsumes all discussed models:**
  - a single loop spiral represents waterfall model.
  - uses an evolutionary approach --
    - iterations through the spiral are evolutionary levels.
  - enables understanding and reacting to risks during each iteration along the spiral.
  - uses:
    - prototyping as a risk reduction mechanism,
    - retains the step-wise approach of the waterfall model.

Comparison of Different Life Cycle Models

- Iterative waterfall model
  - most widely used model.
  - But, suitable only for well-understood problems.
- Prototype model is suitable for projects not well understood:
  - user requirements.
  - technical aspects.

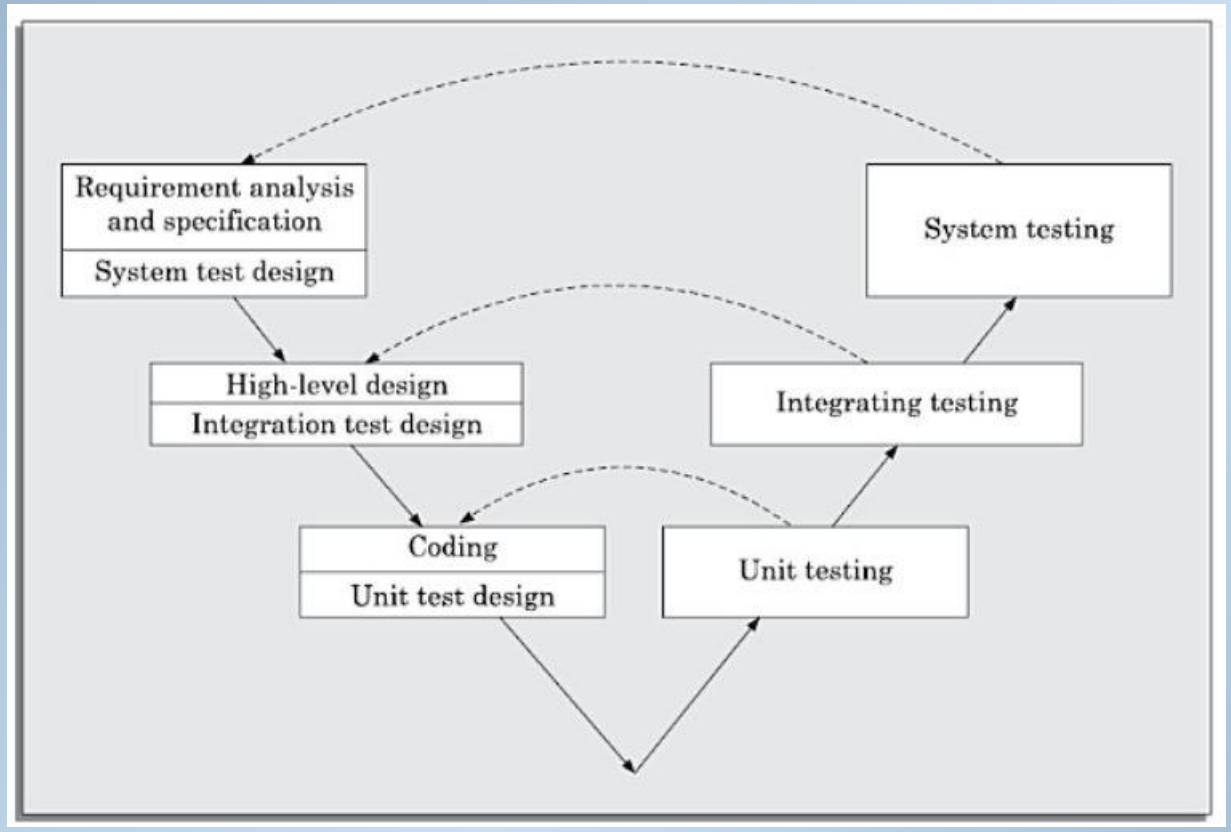
Comparison of Different Life Cycle Models (CONT.)

- **Evolutionary model is suitable for large problems:**
  - can be decomposed into a set of modules that can be incrementally implemented,
  - incremental delivery of the system is acceptable to the customer.
- **The spiral model:**
  - suitable for development of technically challenging software products that are subject to several kinds of risks.

V-Model

- A variant of the waterfall model.
- It gets its name from the visual appearance.
- Here, verification and validation activities are carried out throughout the development life cycle thereby reducing the chances of bugs in the product.
- Suitable for use in projects concerned with development of safety-critical software.

V-Model (contd….)



V-Model (contd….)

- **2** main phases:
  - **Development** – left half of the model.
  - **Validation** – right half of the model.
- In the development phase, along with the development of a work product, test case design and plan for testing of the work product is also carried out.
- Actual testing is done in the validation phase.
- In the validation phase, testing is carried out in three steps. The purpose is:
  - To detect defects in the corresponding phases of software development,
  - Requirements Analysis and Specification,
  - Design and coding.

Advantages and Disadvantages of V-Model

- Much of the testing activities (test case design, test planning etc.) are carried out in parallel with development activities.
- Thus, significant part of the testing activities is already complete before testing phase starts.
- This model leads to a shorter testing phase and an overall faster product development.
- Test cases are designed when schedule pressure has not yet built up, thereby ensuring better quality.
- Testing team is reasonably occupied throughout the development cycle leading to efficient manpower utilization.
- Testing team is associated with the project since its inception thereby having a sound knowledge and understanding of the development artifacts.

- Being a derivative of the classical waterfall model, it inherits most of the weaknesses of the waterfall model.

RAD (Rapid Application Development)-Model

- It was proposed to overcome the rigidities of the waterfall model and its derivatives that make it difficult to accommodate change requests from the customer.
- It has features of both the prototyping and evolutionary models.
- It deploys the evolutionary model to obtain and incorporate customer feedbacks.
- Here, prototypes are constructed and delivered to the customer.
- Unlike prototyping model, prototypes are not thrown away but are enhanced and used in S/W construction.

Goals of RAD Model

- To decrease the time taken and cost incurred to develop S/W systems.
  - Minimal use of planning,
  - Heavy reuse of any existing code through rapid prototyping.
- To limit costs of accommodating change requests.
  - Customers give change requests pertaining to an already developed feature.
  - Incorporation of such change requests right after the delivery of an incremental feature saves cost since it is carried out prior to large investments made in development and testing.
- To reduce communication gaps between the customer and the developer.
- The lack of long-term and detailed planning gives the flexibility to accommodate requirement changes that may be required later.
- The RAD model tries to overcome the problem inherent in iterative waterfall model by inviting and incorporating customer feedbacks on successively developed and refined prototypes.

Working of RAD Model

- Development takes place in a series of short cycles or iterations.
- Development team focuses on the present iteration and plans are made for one increment at a time.
- The time planned for each iteration is called a time-box.
- Each iteration enhances the functionality of the application by only a small amount.
- During each iteration a quick-and-dirty prototype style software is developed.
- The customer evaluates the prototype and provides feedback on specific improvements and based on this the prototype is refined.
- Prototype is not released to the customer.

Applications of RAD Model

- Customized Software.
- Non-critical Software.
- Highly constrained project schedules.
- Large Software.

Exceptions

- Generic Products.
- Requirement of optimal performance and reliability.
- Lack of similar products.
- Monolithic Entity.


THANK YOU