2. Software Life Cycle Model

1. ***Software Life Cycle Model***
2. ***Types of SLCM***
3. ***Classical Waterfall Model***
4. ***Spiral Model***
5. ***Prototyping Model***
6. ***Evolutionary Model***

1. Software Life Cycle Model:

- Software life cycle consists of all the phases from its **inception to retirement**

- Descriptive and diagrammatic representation of the software life cycle

- Maintains orders among phases

- Defines **entry and exit criteria** for every phase

2. Types of SLCM:

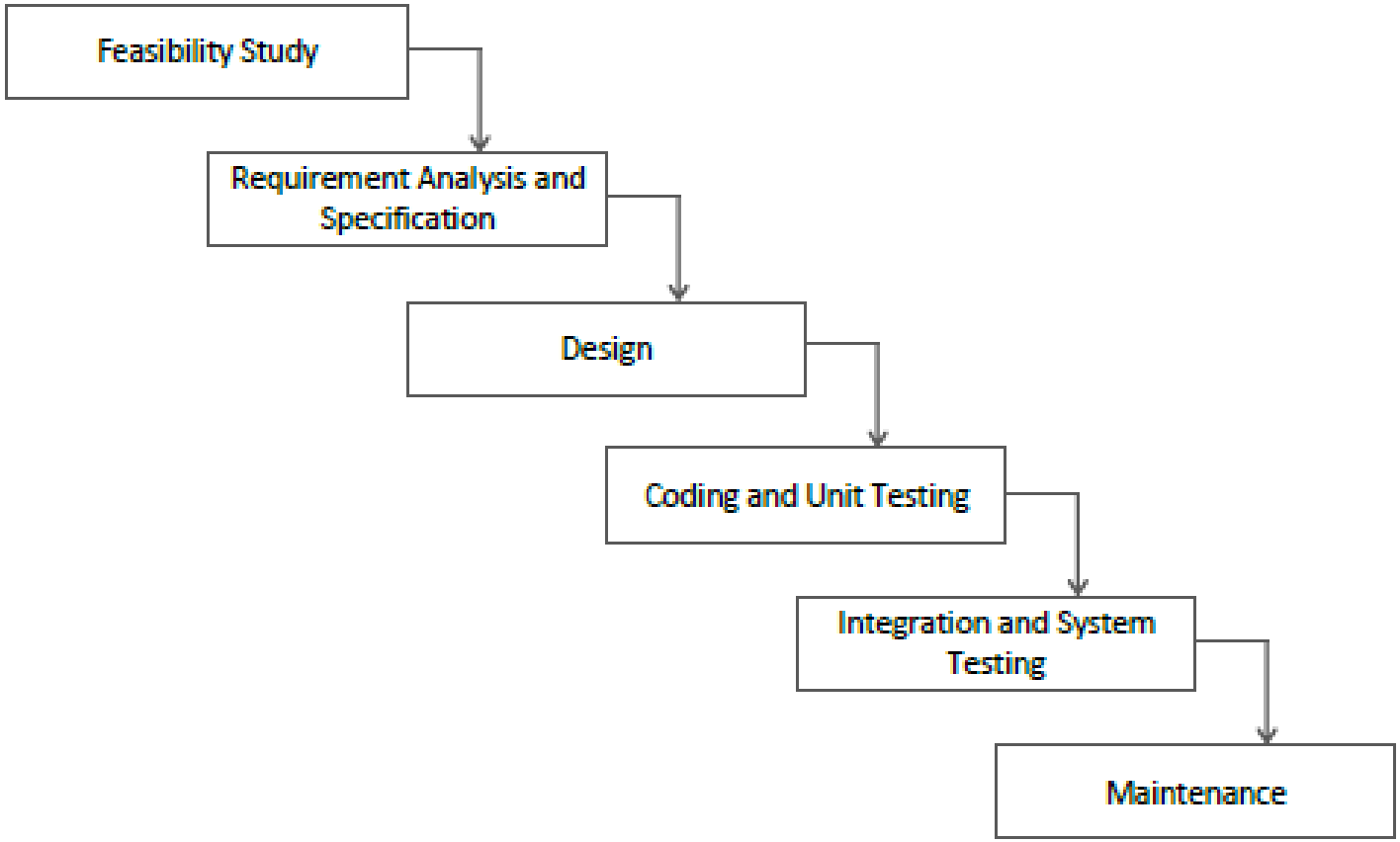
- **Classical Waterfall Model**
- **Iterative Waterfall Model**
- **Prototyping Model**
- **Evolutionary Model**
- **Spiral Model**

  - Also called **SDLCM**
  - Each of them has some pros and cons

3. Classical Waterfall Model:

- **Theoretical way** of developing software
- Other models are essentially **derived** from it
- Divides the life cycle into **six phases**
  - **Feasibility Study**
  - **Requirements Analysis and Specification**
  - **Design**
  - **Coding and Unit Testing**
  - **Integration and System Testing**
  - **Maintenance**

Classical Waterfall Model: (Con..)



Feasibility Study:

- Understand the problem – **social feasibility**
- Know the overall requirements of the client
- Investigate different possible solutions
- Examine and pick the best solution – **technical and financial feasibility**

Requirements Analysis & Specification:

- Understand the exact requirements
- Collect all relevant data through **interviews and discussions**
- Identify all **ambiguities** and **contradictions**
- Remove all ambiguities, inconsistencies, and incompleteness

RA & Specification: (Con..)

- Prepare **SRS document**
- SRS document consists of
  - **functional requirements**
  - **nonfunctional requirements**
  - **goals of implementation**
- SRS document is signed by both the parties

Design:

- **Software architecture** is derived from the SRS document

- Two approaches are
  - **traditional design approach**: structured analysis and structured design
  - **object-oriented design approach**

- **Structure chart** is produced

Coding & Unit Testing:

- Translate software design into source code
- Each module is coded and unit tested
  - **Module**: smallest part that can't be divided further
- Unit or **module testing** ensures debugging of errors identified at this stage
- **Module documentation** is produced

Integration & System Testing:

- Modules are **integrated and tested** in a planned manner
- Carried out incrementally over a number of steps
- System testing ensures **fulfillment of the requirements** laid out in the SRS document

Integration & System Testing: (Con..)

- System testing activities are
  - **α testing**: performed by **development team**
  - **β testing**: performed by a **friendly set of customers**
  - **acceptance testing**: performed by the **customer** after the product delivery to determine whether to accept or reject the delivered product

- System testing is carried out in a planned manner according to the **test plan document**

Maintenance:

- Development to maintenance effort ratio is roughly 40:60

- Consists of three types
  - **Corrective maintenance**
  - **Perfective maintenance**
  - **Adaptive maintenance**

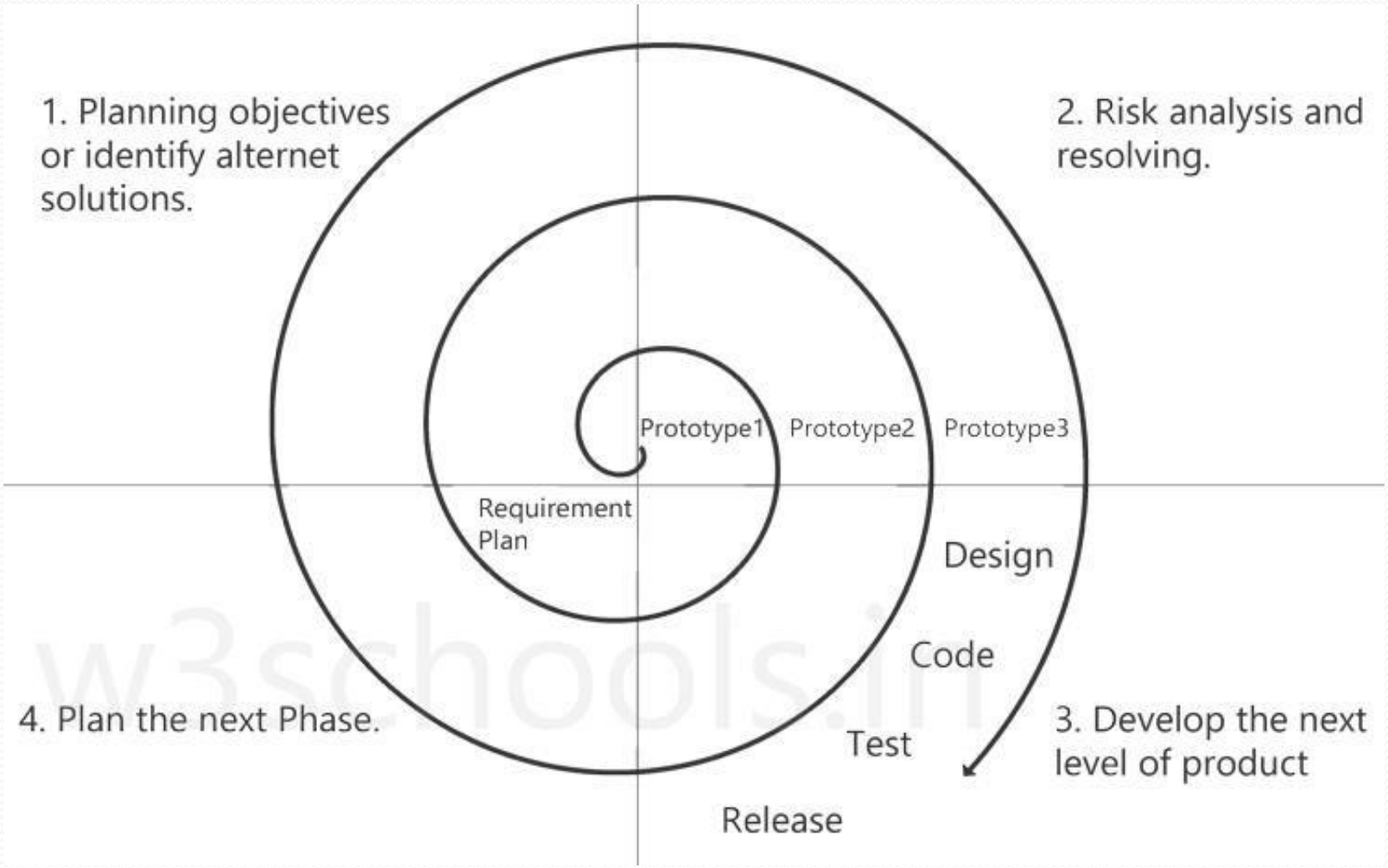
Shortcomings:

- **Idealistic**, not practical
- Defects usually get detected much later in the life cycle
- Need to go back to correct the defect and its effect on the later phases
- Need of **Iterative Waterfall Model**

4. Spiral Model:

- Looks like a **spiral** with many loops
- Each loop corresponds to a phase
- Each loop / phase is split into four quadrants
  - 1st quadrant: **objective setting**
  - 2nd quadrant: **risk assessment and reduction**
  - 3rd quadrant: **development and validation**
  - 4th quadrant: **review and planning**

Spiral Model: (Con..)



Spiral Model: (Con..)

- Number of loops varies
- Encompasses all other life cycle models
- Suitable for software products with several risks
- More complex than others

5. Prototyping Model:

- **Toy implementation** of actual system
- Limited functionalities, low reliability, and inefficient performance
- Preferred in situations such as
  - user requirements are not complete
  - technical issues are not clear

Prototyping Model: (Con..)

- Suitable when customer needs
  - how the screens might look like
  - how the user interface would behave
  - how the system would produce outputs

- Customer checks whether the prototype is acceptable or not

6. Evolutionary Model:

- **Incremental development** and delivery
- No **customer resentment**
- Gradually increasing customer confidence while using the system
- Customer orders the incremental versions as and when affordable