Software Process Modeling – IT1060

What is Software?

- Software is associated documentation and configuration files, needed to make the programs operate correctly.

Difference between Programs and Software

Program	Software
Small	Large
Single developer	Team of developers
Small in size	
Limited Functionality	
Single user	Multiple users
Simple user interface	Complex user interface
Sparse documentation	Detailed documentation
No user manual	User manual
Ad hoc development	Systematic development

Ad hoc Development

- A developer can follow his own way to develop.

Systematic Development

- A developer should follow some proper way to develop.

Software Products 2 Types

1. Generic (Open Market – Any Customer)
2. Customized (Develop for Customer Requirements)

Software Process Activities

- Software Specification
    - Documenting the Expectations on the System.
    - Written and Diagrammatic description of the services & User Requirements.
- Software Development
    - Designing and Implementing (Coding) the System according to specifications.
- Software Validation
    - Checking and verifying whether the System fulfills the requirements.
- Software Evolution
    - Maintenance – Software Upgraded with Time.
    -

Steps for Developing Software

- Feasibility Study
- Analysis
    - Requirements Gathering and Analysis
    - Requirements Specification
- Design
- Development
- Testing
- Maintenance

1. Feasibility Study

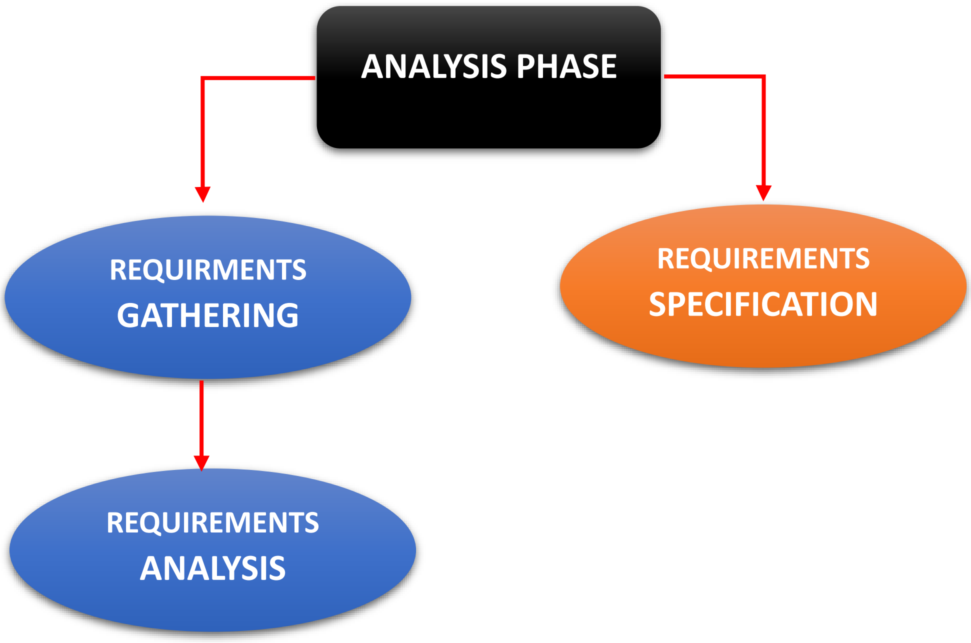
- 1^{ST} Phase of SDLC.
- Project Objective is determined in this phase.
- The client and company discuss the Pros and Cons in this phase.

Why do a feasibility study?

- To provide enough information to management.
  - Whether the Project can be done.
  - Whether the final product will benefit its user.
  - What the alternative solutions are.
  - Whether there is a preferred solution.

2. Analysis (Requirements Phase)

- Goal - Understand the Customer Requirements.



2a.1 Requirements Gathering

- **Goal – The Stakeholders to find out 'What to do'.**
- **Requirement Gathering involves collecting information through meetings, interviews, and discussions.**

2a.2 Requirements Analysis

- **Goal – Understand exactly What the Customer needs.**
  - **Data to be input to the system.**
  - **Processing to be performed on these data.**
  - **Data to be output from the system.**
  - **Characteristics of the system.**
  - **Constraints on the system/project.**

2b. Requirements Specification

- **Requirements are documented in a Software Requirements Specification (SRS) Form.**
- **SRS Form is a Legal Contract with the Customer.**
- **SE's who specialize in requirements gathering, analysis, and specifications are called (System/Business/Requirements) Analysts.**

3. Design

- After Receiving the SRS Form this Phase Begins.
- Architects and Designers craft high-level and low-level design of software.
    - Architectural Design
    - Low-Level Design
- Decisions are made about hardware, software, and system architecture.
- Design Specification Document (DSD) records this information in this Phase.

4. Development

- After Receiving the DSD / SDD (System Design Documents) this Phase Begins.
- A set of developers code the software as per the established design specifications, using a chosen programming language.
- Programming carry out some program testing to discover faults in the program and remove these faults in the debugging process.

5. Testing

- The testing phase ensures that the software requirements are in place and that the software is expected.
- When a defect is identified, testers inform the developers.
- If the defect is valid, developers resolve it and create a new version of the software which then repeats the testing phase.
- The cycle continues until all defects are mitigated and the software is ready for deployment.

6. Deployment & Maintenance

- Software is Error-free? it’s deployed into the operating environment.
- While the customers are using the software, any issues will happen, The Maintenance Team works to resolve them Immediately.

IEEE Definition for Software Engineering

- The Application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of Software.

Key Challenges in Developing Software

- Deliver quality software to the customer at the agreed time.
- The product is intangible. (Can’t touch)
- Product specific
- Keep overall costs within budget.

Software Engineering Ethics

- Accept that work involves wider responsibility than simply application of technical skills.
- Behave Ethical & Morally Responsible way.
- Shouldn’t use skills & abilities to behave in a dishonest way that will bring disrepute to the SE profession.
- Standards,
    - Confidentiality
    - Competence
    - Intellectual Property Rights
    - Computer misuse

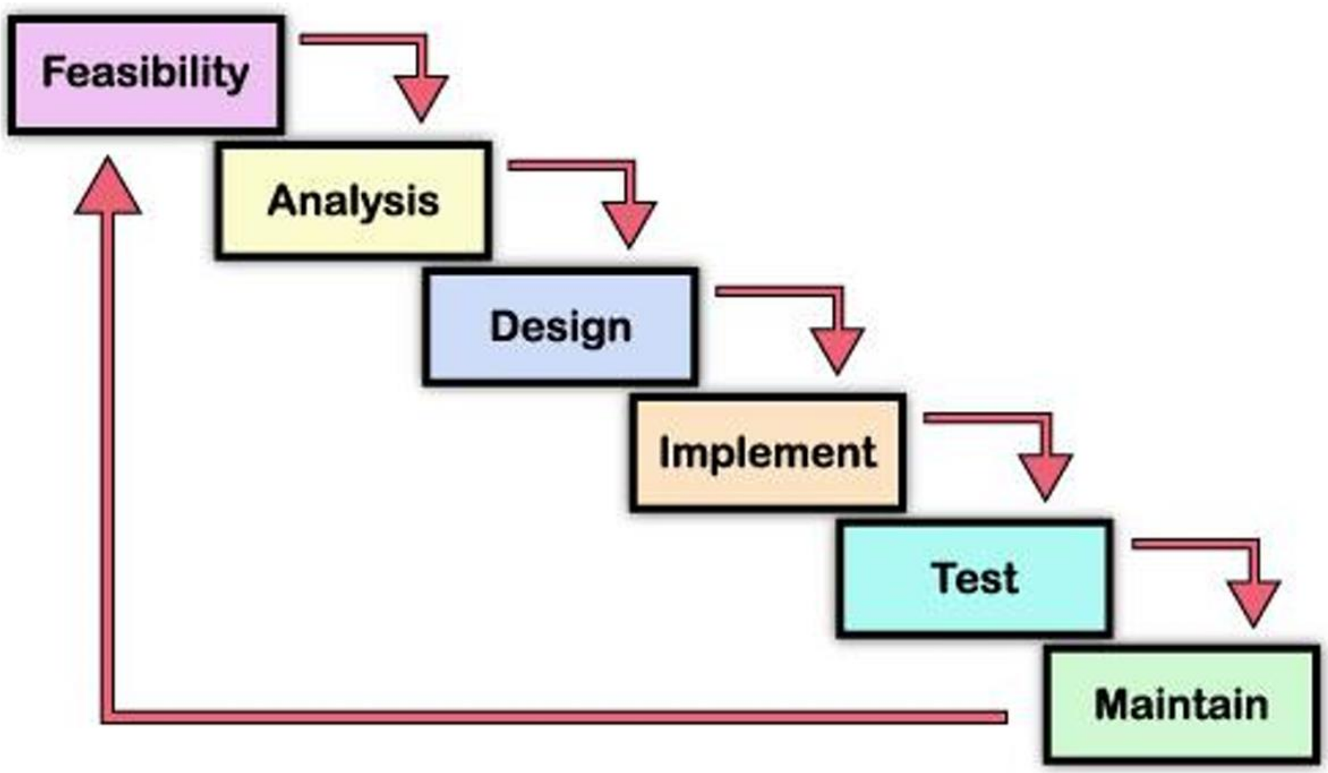
What is SDLC (Software Development Life Cycle)

- SDLC is a Framework that defines activities performed throughout the software development process.

General Software Process Models

- Waterfall Model (Traditional Approach)
  - Classic
  - Iterative
- Prototyping Model (Traditional Approach)
- Evolutionary Model (Traditional Approach)
  - Incremental
  - Spiral
- Agile Model (Modern Approach)

Waterfall Model (Classic)



- Each phase begins only after the previous phase is over.
- Called Linear Model.
- Document driven process.
- This model specifies what the system is supposed to do (define the requirements) before building the system. (designing)
- In this model we can't go back to the previous phase.
- If we want to go to the previous phase, we should go to the Analysis Phase again and orderly continue the following Phases.

Waterfall Modell – Strengths

- Simply & Easy to manage each phase has specific deliverables.
- Milestones are better understood.
- Sets requirements stability.
- Works well for smaller projects where requirements are very well understood.
- A schedule can be set with deadlines.

Waterfall Modell – Weaknesses

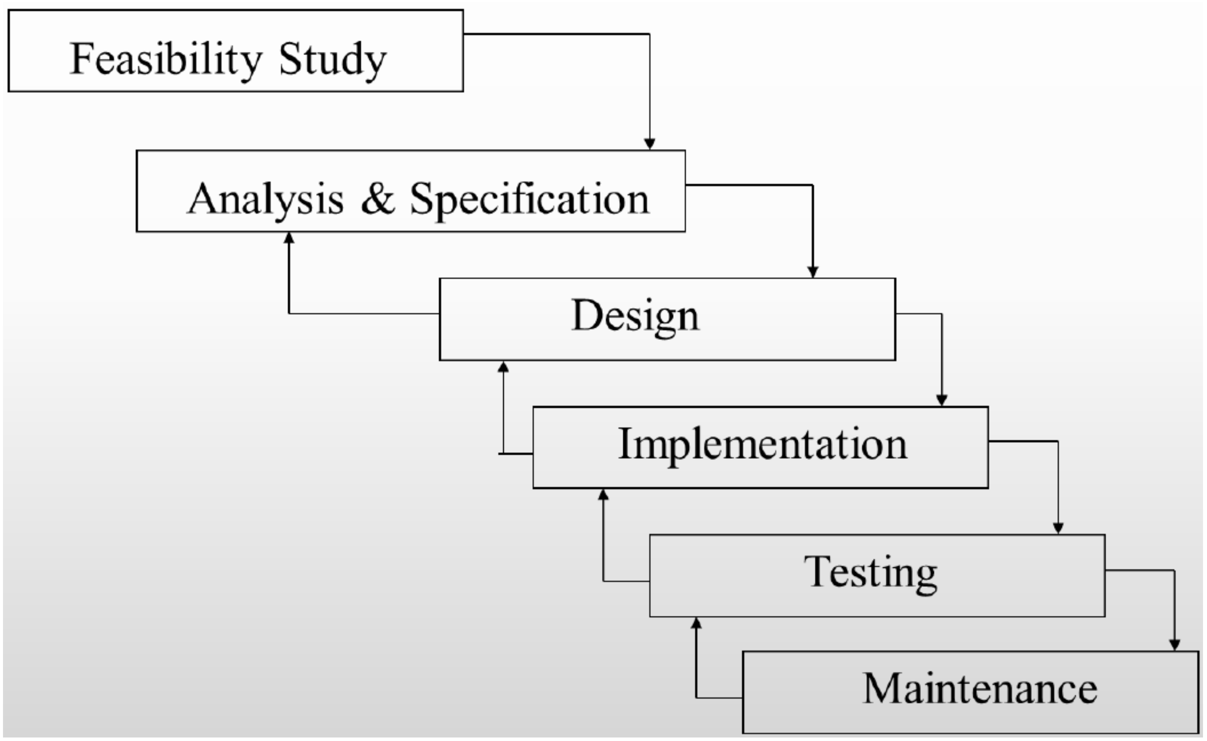
- No working software is produced until it ends.
- High Uncertainty.
- Delay discovery of serious errors.

- After the requirements phase, no formal way to make changes the requirements.
- Not a good model for,
    - Complex projects
    - High-risk requirements changing projects.

When to use Waterfall Model

- Software requirements clearly defined and known.
- Product definition is stable.
- New version of the existing software system is created.
- Software development technologies and tools are well known.
- Ample resources with required expertise are available.

Iterative Waterfall Model (Iterative)



Iterative Modell – Strengths

- **Defects are detected and fixed early through the feedback path.**

Iterative Modell – Weaknesses

- **Limited Customer Interactions.**
- **Difficult to incorporate change requests.**