Lecture: 2
Software Process & SDLC Methodologies

The phrases "Software Process", Software Life Cycle", "System Development Life Cycle", "System Life Cycle", "Development Life Cycle" are all used to describe the same concept.

A software process or software methodology is a set of related activities that leads to the production of the software. These activities may involve the development of the software from the scratch or modifying an existing system. Any software process must include the following four activities:

1. **Software specification** (or requirements engineering): Define the main functionalities of the software and the constrains around them.
2. **Software design and implementation**: The software is to be designed and programmed.
3. **Software verification and validation**: The software must conforms to it's specification and meets the customer needs.
4. **Software evolution** (software maintenance): The software is being modified to meet customer and market requirements changes.

Software Development Life Cycle:

Software development life cycle (SDLC) is a process model that **comprises six different software development phases:** (1) requirement analysis, (2) design, (3) implementation, (4) testing, (5) deployment, and (6) maintenance.

- **Requirement analysis:** Gather all the requirements about the software from stakeholders. What features should be included? How should it look like? How should it behave?
- **Design:** Choose the programming language and the database that best suits your project.
- **Implementation**: Code the software.
- **Testing:** Testers run test cases, find the bugs, and ask developers to fix them.
- **Deployment:** Once the code is approved, put it on live.

- **Maintenance:** Fix bugs and update the software to ensure it works well all the time.

Software Process Models or Software Engineering Paradigm:

To solve actual problems in an industry setting, a software engineer or a team of engineers must incorporate a development strategy that encompasses the process, methods, tools and generic phases for the development of a software. This strategy is often referred to as a process model or a software engineering paradigm.

A software process model is a simplified representation of a software process. Each model represents a process from a specific perspective.

A process model for software engineering is chosen based on the nature of the project and application, the methods and tools to be used, and the controls and deliverables that are required.

The most commonly used software development processes are:

1. **Water fall**
2. **Incremental**
3. **Spiral**
4. **Evolutionary**
5. **Prototyping**

The Waterfall Life Cycle Model:

- The waterfall model is a sequential approach, where each fundamental activity of a process represented as a separate phase, arranged in linear order.
- The phases of the waterfall model are: Requirements, Design, Implementation, Testing and Maintenance.
- In principle, the result of each phase is one or more documents that should be approved and the next phase should not be started until the previous phase has completely been finished.
- In practice, however, these phases overlap and feed information to each other
    - e.g. during design, problems with requirements can be identified and during coding, some of the design problems can be found etc.

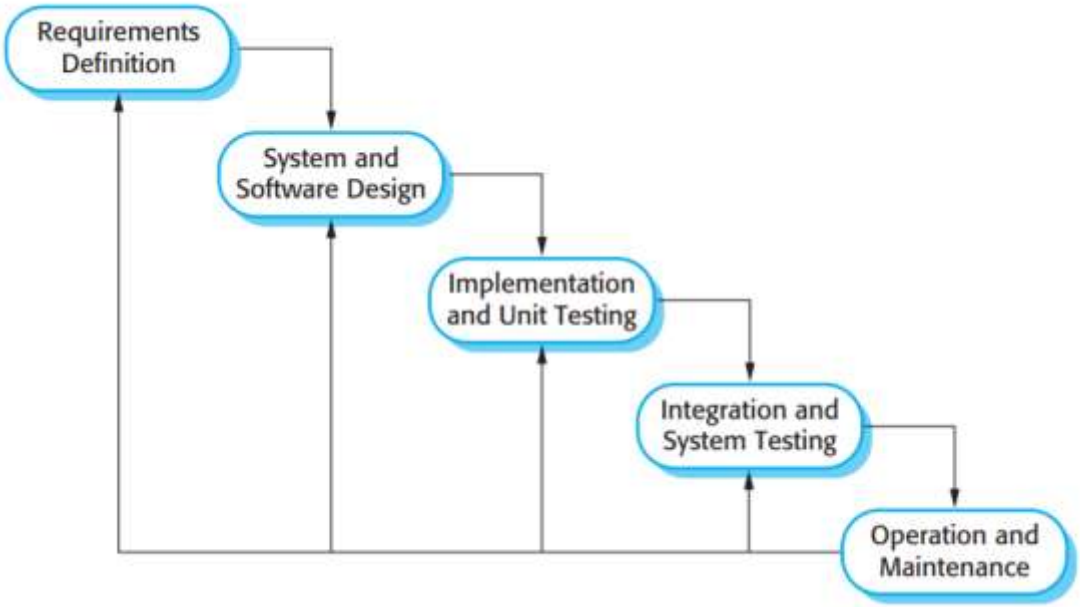

Fig: Phases of Waterfall Model

- The principal stages of the waterfall model directly reflect the fundamental development activities:
    1. **Requirements Definition:**
        - The system's services, constraints and goals are established by consultation with system users. They are then defined in detail and serve as a system specification.
    2. **System and Software Design:**
        - The system design process allocates the requirements to either hardware or software systems by establishing an overall system architecture.
        - Software design involves identifying and describing the fundamental software system abstractions and their relationships.
    3. **Implementation and Unit Testing:**
        - During this stage, the software design is realized as a set of programs or program units. Unit testing involves verifying that each unit meets its specification.
    4. **Integration and system testing:**
        - The individual program units or programs are integrated and tested as a complete system to ensure that the software requirements have been set.

- After testing, the software system is delivered to the customer.

5. **Operation and Maintenance:**
   - Normally this is the longest life cycle phase.
   - The system is installed and put into practical use.
   - Maintenance involves correcting errors which were not discovered in earlier stages of the life cycle, improving the implementation of system units and enhancing the system's services as new requirements are discovered.

Advantages of Waterfall Model:

- It can serve as a useful process model in situations where requirements are fixed and work is to proceed to complete in a linear manner.

Limitations/ Problems of Waterfall Model:

- Real projects rarely follow the sequential flow that the model proposes. Although the linear model can accommodate iteration, it does so indirectly. As a result, changes can cause confusion as the project team proceeds.
- It is often difficult for the customer to state all requirements explicitly. The waterfall model requires this and has difficulty accommodating the natural uncertainty that exist at the beginning of many projects.
- The customer must have patience. A working version of the programs will not be available until late in the project time-span. If a major blunder is undetected then it can be disastrous until the program is reviewed.