# Lecture 3 (PMP and SDLC):

## L3 - Intended Learning Objectives

1. Understand what a Process is and its relevance to Project Management.
2. Understand what a Project Management Plan (PMP) is and when it should be used.
3. Understand the components of a Project Management Plan.
4. Understand what a Software Development Lifecycle (SDLC) is and the advantages / disadvantages of various models.

**What is a Process:** A series of progressive and interdependent steps by which an end is attainted.

**What does a process have to do with PM and software engineering:**

- PM is a process as it defines a series of tasks (Planning, controlling) to deliver a specific/agreed outcome.
- System Development Lifecycle (SDLC) describes process for planning, creating, testing and deploying an info system.

**Project Management Plan:**

- Formal approved document that defines how the project is executed, monitored and controlled. May be summary or detailed.
- Owned, controlled and populated by PM.
- A good PMP provides required level of detail across key project components and is the one source of truth for all parties involved in project.

**Difference between project charter and PMP:**

- A Project charter is a summary project proposal to secure approval for the project goals and terms. (useful for business case)
- PMP is approved document showing how to achieve the approved project goals/benefits and provides details on how to execute and manage project.

**Things in PMP:**

- Executive summary
- Scope
- Milestones
- Stakeholders
- Business values
- Budget + cost estimation
- Constraints
- Roles and responsibilities
- Schedule
- Risk management
- Quality assurance

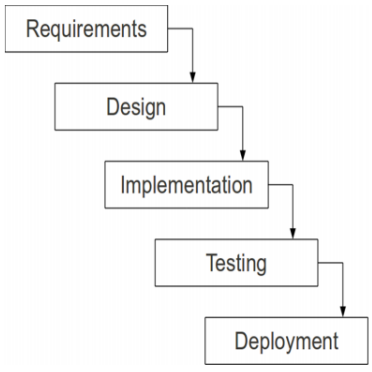
**SDLC: Systems Development Life Cycle**

- Process for planning, creating testing and deploying and info system.
- Key activities displayed in diagram.
- Formal Processes:
    - Waterfall
    - Incremental
    - V-Model
- Agile Processes:
    - Scrum
    - Kanban


**Waterfall:** Requirements first driven approach

Advantages:

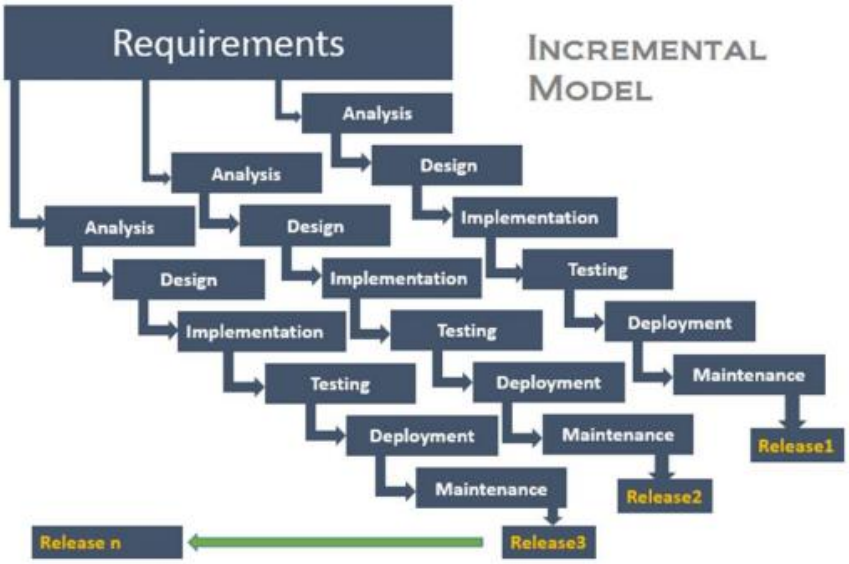
- Works well where requirements well understood and are stable
- Simple and easy to understand/use
- Easy manage due to rigidity of model
- Phases complete one at a time
- Documentation available at end of each phase



Disadvantages:

- Difficult to accommodate change after the process is underway.
- One phase must be completed before moving onto the next.
- Unclear requirements lead to confusion.
- Clients approval is in the final stage.
- Difficult to integrate risk management due to uncertainty.

**Incremental:** The whole requirements is divided into various releases. Multiple cycles take placing making the life cycle a multi-waterfall cycle. Cycles are divided into smaller more easily managed modules. Still getting requirements upfront.




Advantages:

- Each release delivers an operational product.
- Less costly to change the scope/requirements
- Customers can respond to each build
- Initial product deliver is faster
- Customers get important functionality early
- Easier to test and debug during smaller iterations

Disadvantages:

- More resources and management attention required
- Defining/partitioning increments may be difficult/unclear
- Each phase of iteration is rigid with no overlaps
- Problems may occur at the time of final integration