**Lecture 3 & 4:**

**Software Development Activities**

- **Construction**: Requirements, Design, Coding, Testing
- **Management**: Planning, Quality Assurance, Configuration, Installation

---

**Ad Hoc Development**

- No structured plan or defined tasks/deliverables
- Risky: prone to delays, missed tasks, poor quality

**Case Study (SaudiTech)**: Failed due to unrealistic budget and ad hoc approach.

➞ **Solution**: Use **Software Process Models** for structured development

---

**Software Process**

Structured set of activities:

1. **Specification**
2. **Design & Implementation**
3. **Validation**
4. **Evolution**

Process models describe **what happens and in what order**.

---

**Plan-driven vs. Agile**

- **Plan-driven**: Fully planned up-front, fixed scope
- **Agile**: Flexible, adaptive, iterative planning
- Most real-world processes are a **mix** of both.

---

**Popular Software Process Models**

1. **Waterfall Model**
    - Sequential stages: Requirements → Design → Implementation → Testing → Deployment → Maintenance
    - Works best when requirements are well-understood
    - **Pros**: Easy to manage, documented phases
    - **Cons**: Inflexible, poor with changing needs

2. **V-Model (Verification & Validation)**
    - Each dev phase has a corresponding test phase
    - Emphasizes early test planning
    - **Pros**: Thorough validation, early error detection
    - **Cons**: Poor with changing requirements
3. **Evolutionary Models**
    - **Exploratory**: Start with known features, evolve with user input
    - **Prototyping**: Build a prototype to refine user requirements
    - **Pros**: Better understanding, early feedback
    - **Cons**: Scope creep, poor structure
4. **Component-Based Software Engineering (CBSE)**
    - Build systems by reusing existing components
    - **Pros**: Faster, cheaper, lower risk
    - **Cons**: Integration issues, limited flexibility
5. **Incremental Model**
    - Build software in small, usable chunks (increments)
    - Customer gets value early
    - **Pros**: Early delivery, feedback loop, reduced risk
    - **Cons**: Hard to slice system appropriately
6. **Spiral Model**
    - Risk-driven, iterative model with phases: Planning → Risk Analysis → Development → Evaluation
    - Each loop = 1 development phase
    - **Pros**: Handles risk well, suitable for large projects
    - **Cons**: Complex, expensive

---

**Conclusion**

- Ad hoc = dangerous.
- Process models help structure development.
- Choose model based on project size, risk, and clarity of requirements.