7: Architectural Design

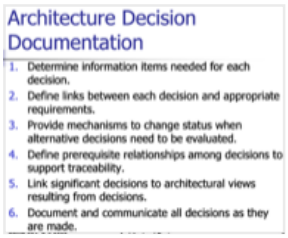
Architecture is a representation that enables a software engineer:

- To analyze effectiveness of the design
- Consider architectural alternatives when making design changes is easy
- Reduce the risks associated with the construction of the software

Why Architecture is important:

- Provides a representation that lets communicate among all stakeholders.
- Highlights early design decisions.
- Constitutes a relatively small mode of how the system is structured and how it works together.

Architecture Decision documentation:



Agility and Architecture:

- To avoid rework, user stories are used to create and evolve an architectural model before coding.
- Agile projects include delivery of architectural documentation during each sprint(work units).
- After the sprint is complete, review the working prototype for quality before the team presents it to the stakeholders in a formal sprint review.

Architectural styles:

1. Set of components: (database, computational modules) that perform a function required by a system.
2. Set of connectors: Enable communication, coordination, and cooperation among components.
3. Constraints: How components can be integrated to form the system.
4. Semantic models: Enable a designer to understand the overall properties of a system by analyzing the known properties of its constituent parts.

Types of architecture:

- Data cantered - data storage in the centre
- Data flow - one way diagram
- Call return - tree

- Object Oriented - classes
- Layered Architecture - User interface -> application -> utility -> core layer
- Model view controller

Architectural Organization and Refinement (세련된):

Control	Data
How’s control managed within the architecture.	How data communicated between components?
How control is shared among components?	What is the mode of data transfer?
How do components transfer control within the system?	Are data components passive or active?

Architectural Considerations:
- Economy: Software relies on abstraction to reduce unnecessary detail.
- Visibility: Architectural decisions and their justifications should be obvious.
- Spacing: Separation of concerns in a design without introducing hidden dependencies.
- Symmetry: Implies that a system is consistent and balanced in its attributes.
- Emergence: Self-organized behaviour is key to creating efficient software architectures.

Architectural Design:
- Software must be placed into context:
  - Design should define the external entities (other systems and devices) that the software interacts with and the nature of the interaction.
- A set of archetypes should be identified.
  - An archetype is an abstraction (similar to a class) that represents one element of system behaviour.
- Designer specifies the structure of the system by defining and refining software components that implement each archetype.

Architecture Context Diagram: Components diagram
Function Archetype: Shows element of system.
Top-Level Component Architecture: Tree design
Refined Component Architecture: Tree

Architectural Tradeoff Analysis:

1.2.: Collect scenarios, elicit requirements, constraints, environment description
3.Describe the architectural styles/patterns that have been chosen to address the scenarios and requirements using views: module, process, data flow.
4.Evaluate quality attributes by considering each one in isolation.
5.Identify the sensitivity of quality attributes to various architectural attributes for a specific arch style.
6.Critique candidate architectures developed in step3 with using sensitivity analysis in step5.

Architectural reviews:
Assess the ability of the software architecture to meet the systems quality requirements and identify potential risks.

Pattern-based architectural reviews:

1. Identify and discuss the quality attributes by walking through the use cases.
2. Discuss a diagram of system arch relation to its requirements.
3. Identify the architecture patterns used and match the system’s structure to the patterns’ structure.
4. Use existing documentation and use cases to determine each pattern’s effect on quality attributes.
5. Identify all quality issues raised by architecture patterns used in the design
6. Develop a short summary of issues uncovered during the meeting and make revisions to the walking skeleton.