Introduction to Software Architecture

Lecture-1

Software Architecture

Software architecture is a high-level abstraction that provides a logical framework, guidelines, and patterns to guide the development of a system

Software Design

Software design provides a design plan that describes the elements of a system, how they fit, and work together to fulfill the requirement of the system.

Software Architecture vs Design

• Software architecture is about the complete architecture of the overall system.	• Software design is about designing individual modules/components.
• Software architecture defines the fundamental properties.	• Software design defines the detailed properties.
• In general, it refers to the process of creating high level structure of a software system.	• In general, it refers to the process of creating a specification of software artifact which will help to developers to implement the software.
• It helps to define the high level infrastructure of the software.	• It helps to implement the software.
• Software architecture manages uncertainty.	• Software design avoids uncertainty.
• Software architecture is more about the design of entire system.	• Software design is more about on individual module/component.

Software Architecture vs Design

- It is a plan which constrains software design to avoid known mistakes and it achieves one organizations business and technology strategy.
- Some of software architecture patterns are microservice, server less and event driven.
- In one word the level of software architecture is structure.
- What we are building is software architecture.

- It is considered as one initial phase of Software Development Cycle (SSDLC) and it gives detailed idea to developers to implement consistent software..
- Some of software design patterns are creational, structural and behavioral.
- In one word the level of software design is implementation.
- How we are building is software design.

Characteristics of Software Architecture

- **High-Level Structure**: Software architecture focuses on defining the overall structure and organization of a system at a high level.
- **System Components**: Identifies major components/modules and their relationships, providing a blueprint for the system's construction.
- **Architectural Patterns**: Involves selecting and applying appropriate architectural patterns (e.g., client-server, microservices) to guide the overall system design.
- **Scalability Planning**: Considers how the system can scale to meet future demands, addressing issues of performance and resource management.
- **Interoperability**: Deals with how the software will interact with other systems, ensuring compatibility and seamless integration.
- **Strategic Decision-Making**: Involves making strategic decisions that impact the entire system, such as technology choices and major design principles.

Characteristics of Software Architecture – Cont...

- **Architectural Styles**: Defines the architectural style, which dictates how the components interact and communicate within the system.
- **Global Considerations**: Takes into account global aspects of the system, including distribution, networking, and data storage.
- **System Integrity**: Ensures the integrity of the entire system, focusing on key system-wide concerns rather than specific implementation details.
- **Long-Term Vision**: Encompasses a long-term vision for the system's evolution and adaptation to changing requirements.

Characteristics of Software Design

- **Low-Level Structure**: Software design focuses on the detailed structure of individual components and modules within the system.
- **Component Details**: Involves specifying the internal details of each module, defining how they will accomplish their specific tasks.
- **Design Patterns**: Utilizes design patterns to solve specific problems at the module or class level, promoting best practices in coding.
- **Efficiency and Optimization**: Optimizes algorithms and data structures for efficient resource utilization and improved system performance.
- **Code Readability**: Emphasizes writing clear and readable code, following established coding standards and practices.
- **Modularity**: Breaks down the system into modular components to enhance maintainability, reusability, and ease of testing.

Characteristics of Software Design – Cont...

- **User Interface Design**: Includes designing user interfaces, specifying how users interact with the system and ensuring a positive user experience.
- **Error Handling**: Incorporates strategies for handling errors and exceptions within the individual modules, promoting robustness.
- **Data Management**: Addresses how data is stored, retrieved, and manipulated within the system at the module level.
- **Task Decomposition**: Breaks down complex tasks into smaller, manageable sub-tasks within individual components.

Goals of Architecture

- Expose the structure of the system but hide its implementation details.
- Realize all the use-cases and scenarios.
- Try to address the requirements of various stakeholders.
- Handle both functional and quality requirements.
- Improve quality and functionality offered by the system.
- Improve external confidence in either the organization or system.

Goals of Architecture - Limitations

- Lack of tools and standardized ways to represent architecture.
- Lack of analysis methods to predict whether architecture will result in an implementation that meets the requirements.
- Lack of awareness of the importance of architectural design to software development.
- Lack of understanding of the role of software architect and poor communication among stakeholders.
- Lack of understanding of the design process, design experience and evaluation of design.