SUMMARY of Chapter 6 (TASK 1)

Architectural Design is the first stage of software development process.

It helps us to recognize the components involved in a system software and the relationship to be established between them. It goes hand in hand along with requirement gathering stage in real world. Architectural model is the final outcome of the architectural design process which depicts how the overall system is organized as a set of interactive components.

Even for Agile model, it is an accepted standard to build an overall architectural model in the initial stage. Architectural design in incremental fashion is not a usual approach as It would be expensive process to maintain as it would involve refactoring of multiple components related to it.

Software architectures can be designed at different two levels of abstraction.

- Architecture can be designed in small scale. E.g., Adding of products in cart for amazon shopping website. It is a singular program subdivided into multiple sub-components.
- Architecture can be designed in large scale. E.g., The complete design of Amazon shopping website with multiple individual programs and functionalities comprising of enterprise level design.

Advantages of documenting Software Architecture

- Easier in decision making process with stakeholders and project planning
- Better analysis of System requirements
- Reusability of the architectural model in similar requirement for future use.

High Level vs Detail Software Architecture models

- High level Architecture is easier for discussions with stakeholders as it does not involve too much of detailing and intricacies of the system.

- Detail architectural models with all relationships of the components is best to broaden and modify the system in order to avoid any misunderstanding. But it might be an expensive process to maintain.

Architectural Design Decisions

Design decisions are highly dependent on functional and non-functional requirements of the system. The style of architecture is directly influenced by multiple nonfunctional requirements of the system:
Performance: If this is the key requirement for the system then architecture might be needed to be designed in a compact way for all the critical components to reduce latency in communication between related features.

Security: For higher security prone system, a layered architecture with multiple security validations and protecting the critical elements in the inner most layer is expected to mitigate security threats.

Safety: If a system is required with more safety measures to be inculcated in the architecture, in that case, we require to put all the safety related components in a tightly coupled manner in order to make it cost effective and easier for safety validations.

Availability: A system with constant availability for use if be in the requirement, then redundant systems are required for backup as a contingency plan to build a fault tolerant system.

Maintainability: If easy maintenance is the need of the system, in that case, finely grained self-contained system is required for faster changes.

Architectural Views

There are multiple perspectives in which the software architecture can be depicted as a single view would not do bring out the whole picture. **4+1 view** model of software architecture are the four fundamental architectural views. It is linked with common scenario.

The logical view: It is the object model of the design (when an object-oriented design method is used)
The process view: It captures the concurrency and synchronization aspects of the design during runtime.
The physical view: It describes the mappings of the software onto The hardware and reflects its distributed aspect.
The development view: It describes the static organization of the software in its development environment.

There is one more view called conceptual view that is used to portray the detailing of high level requirement to use it for stakeholders in the process of the decision making.

Architectural Patterns

Architectural patterns are adoptive software systems that are reused and shared in software industry.There are multiple common architectural patterns that are widely followed for different systems and environments.

Layered Architecture: It is an n – tiered architecture where each layer is organized in horizontal fashion Each layer is dependent on the layer beneath it. Modifications to any layer is localized to its layer and the adjacent layer. Layers can be split into further layers in order to divide or modify functionalities.
Example : GMAIL applications use Layered Architecture with three layers as below:

i. There is an internal layer that does all the processing.
ii. There is an external layer that communicates with the users in their language.
iii. There is also another layer that interacts with a database where user email messages are stored (millions or maybe billions).

Other possible use cases for layered architecture:

- Home banking website
- Control system for self- driving vehicles

Repository architecture :

A repository architecture consists of a central data structure (often a database) and a collection of independent components which operate on the central data structure. Examples of repository architectures include blackboard architectures, where a blackboard serves as communication centre for a collection of knowledge sources, and database systems serving several applications.

Client Server Architecture:

Client-server architecture is a computing model in which the server hosts, delivers and manages most of the resources and services to be consumed by the client. This type of architecture has one or more client computers connected to a central server over a network or internet connection. It is assumed as distributed system architectures.

Advantages:

Easier to add or upgrade the servers without affecting the networked system.

Example:

- The Automated Teller Machine (ATM) is a typical example of Client Server system.
- Cloud Computing

Application Architectures:

- Transaction processing systems: Interactive systems with asynchronous access to the database with concurrent access by multiple users to the database

  E.g. Banking system

- Language process systems

  E.g. Language Compilers