22-05-2023

**Introduction to Software Architectural Design**

The architecture of a system describes its major components, their relationships (structures), and how they interact with each other. Software architecture and design includes several contributory factors such as Business strategy, quality attributes, human dynamics, design, and IT environment.

We can divide Software Architecture and Design into two distinct phases:

* Software Architecture: Non-functional decisions are cast and separated by the functional requirements. It serves as a blueprint to the system. It defines a structured solution to meet all the technical and operational requirements, while optimizing the common quality attributes like performance and security.
* Software Design: Functional requirements are accomplished. Software design provides a design plan that describes the elements of a system, how they fit, and work together to fulfil the requirement of the system.

Key aspects in software architecture design:

1. **System Goals and Requirements:**

* The primary goals of the architecture are to identify requirements that affect the structure of the application.
* Some of the other goals are:
* Expose the structure of the system, but hide its implementation details.
* Realise the use-cases and scenarios.
* Try to address the requirements of various stakeholders.
* Improve quality and functionality offered by the system.

1. **Architectural Styles:**

* We need to choose the best architectural style that fits the system’s requirements and constraints properly.
* The various architectural styles include:
* Data- centred architecture
* Data-flow architecture
* Call and return architecture
* Object-oriented architecture
* Layered architecture
* Client-server architecture
* Microservices architecture
* Event-driven architecture

1. **Components and Modules:**

* Identify the major components and modules of the system. These can be logical or physical units that overlay specific functionalities or services.
* The module in software is a small part of the software that is responsible for performing any kind of functionality.
* A module can be thought as a mini each which performs any of the specific tasks of the software, and the software consists of various such mini software, which can also be called sub programs.
* Modules are added into a software so that each of the functionalities can be easily identified and be understandable for other developers.
* The components-based development of a software is a very effective way to develop software and has many benefits over developing the entire software under a single module. Code reusability of code can be achieved.

1. **Communication and Integration:**

* Define how different components will communicate and interact with each other.
* Integration in software engineering means combining software parts (so-called subsystems) into one system.
* Integrated systems demonstrate better performance comparing to the sum of all independent ones. What is more important, a holistic application delivers greater functional significance.
* It’s more convenient to distribute and use. Integrating is trending as never before these days.
* More and more mergeable systems appear on the Internet. Therefore, connecting with them is vital.
* There are 4 main methods of software integration:
* Star integration
* Horizontal integration
* Vertical integration
* Implementation of the common data format
* APIs
* Message queues

1. **Data Management:**

* This process involves the designing of data storage and management aspects of the system.
* It is the practice of collecting, keeping, and using data securely, efficiently, and cost-effectively.
* The goal of data management is to help people, organizations, and connected things optimize the use of data within the bounds of policy and regulation so that they can make decisions and take actions that maximize the benefit to the organization.
* A robust data management strategy is becoming more important than ever as organizations increasingly rely on intangible assets to create value.
* Managing digital data in an organization involves a broad range of tasks, policies, procedures, and practices.
* The work of data management has a wide scope, covering factors such as how to:
* Create, access, and update data across a diverse data tier
* Store data across multiple clouds and on premises
* Provide high availability and disaster recovery
* Use data in a growing variety of apps, analytics, and algorithms
* Ensure data privacy and security
* Archive and destroy data in accordance with retention schedules and compliance requirements.

1. **Scalability and Performance:**

* Performance is an indication of the responsiveness of a system to execute any action within a given time interval, while scalability is the ability of a system either to handle increases in load without impact on performance or for the available resources to be readily increased.
* Scalability refers to the characteristic of a system to increase performance by adding additional resources.
* Identify potential bottlenecks and apply appropriate architectural patterns like load balancing, caching, and horizontal/vertical scaling techniques.

1. **Security and reliability:**

* Incorporate security measures into the architecture, such as authentication, authorization, encryption, and secure communication protocols.
* Define fault tolerance and error handling mechanisms to ensure reliability and system availability.
* It is the concept of implementing mechanism in the construction of security to help it remain functional (or resistant) to attacks.
* The idea behind software security is building software that is secure from the get-go without having to add additional security elements to add additional layers of security.

1. **Deployment and Infrastructure:**

* Determine how the system will be deployed and hosted.
* Deployment and infrastructure are crucial considerations in software architecture design.
* They involve planning, setting up, and maintaining the hardware, software, and network resources necessary to run and support a software system.
* Proper deployment and infrastructure design ensure that the system operates efficiently, reliably, and securely.

1. **Maintainability and Extensibility:**

* Maintainability refers to the ease with which a system can be modified or updated over time.
* A system with good maintainability is flexible and easy to modify, reducing the time and effort required to make changes.
* Real-world examples of maintainable systems include the Linux operating system, which has been continuously developed and improved over the past 30 years.
* Extensibility refers to the ease with which a system can be extended to support new features or functionality.
* A system with good extensibility can adapt to changing requirements and scale to support new use cases.
* This can be achieved by following best practices such as using modular design, implementing an appropriate architecture pattern, and using appropriate design principles such as SOLID.

1. **Documentation and Communication:**

* Documentation and communication are crucial aspects of system architecture design.
* They play a vital role in ensuring the successful development, implementation, and maintenance of a system.
* Effective documentation and communication enable stakeholders to understand the system's architecture, facilitate collaboration among team members, and support future decision-making processes.
* Effective documentation and communication are ongoing processes in system design architecture.
* Continuously update and refine the documentation as the system evolves, and foster a culture of collaboration and knowledge sharing to ensure that the system architecture remains well-documented and understood throughout its lifecycle.