Software Architecture - How to Approach it in 2021

History & Evolution

⇒ Software architecture's structure has been influenced by the new trends, impacting architectural style.

⇒ The biggest drawback of physical servers was scalability

⇒ The resources were very limited, it became important to utilize these resources efficiently, hence mono-lithic architecture was used as all the resources were on a single server.

⇒ With the introduction of Docker it was possible to split the applications into many pieces and scan each of them separately.

⇒ Later, cloud was introduced which changed the approach to-wards the software and system design.

⇒ AWS (Amazon Web Services) only had EC2 initially but they introduced new services like ECS or EKS, it became possible to build all kinds of non-monolithic distributed sowtware arch. & scale components independently.

Monolithic Software Architecture:-

⇒ It was the first and the most common high level architecture of a system, where the main idea was to make a master single application which does everything.

⇒ It was very easy to break and hence to over come a new architecture called Modular monolithic arch. was introduced, as it was able to hande and was best to build such systems, it was called the holy grail of a monolith.

Distributed Software Architetture

⇒ In this, is was possible to split the application into small pieces, which could be then scaled independently for both the application as well as software eng. team.

⇒ Service oriented approach - microservices, nanoservices, & event driven arch. etc, these are usually case-specific. Some services are - CQRS (Command Query Responsibility Segregation) Hexagonal arch. & Event Sourcing.

Best Software Development practices for Soft. Arch.:

→ Understand current status of the business.

→ have clear vision of the goal to be achieved & know what the key drives for the change to be happed are.

→ gather all constraints to be followed.

→ prepare what will be the key metrics,

→ transfer architecture vision into diagrams and let the developers follow it & apply them in the development cycle.

Future of Software Architecture

⇒ Approaches & techniques that were used decades ago are still being used, nobody can accurately predict the future of software architecture, as it remains constant.

⇒ There just might be change in the technologies used to implement the architecture, hence as trends suggest, companies will migrate to serverless architecture.

⇒ There will be more demand and importance of data as the ~~data~~ lakes will become the core of businesses, this inturn will impact in the decision making and data will rule in every field of IT.

TRENDS & NEW DIRECTIONS IN SOFTWARE ARCHITECTURE

⇒ The definition of software architecture is the set of structures that are needed to reason about system, which comprise software elements, this relationship, and properties of both, allows engineering trade offs.

⇒ Over the years, there has been lot of changes in the field of software architecture:

→ Architectural patterns & styles that allows a vocabulary for design and analysis.

→ component based approaches that take a container strategy with interfaces that make assumptions about quality attributes.

→ company specific production lines with architecture, that allows to manage the variation and at the same time capitalize on commonality and model based approches where architectural models are used to generate code.

→ Most recently frameworks and pleatforms that form the basis of ecosystems where the comm-unication protocol are of paramount importance.

⇒ Technical debt is a design or construction approach that's expedent in the short term, but that creates a technigal content that increases complexity & cost in the long-term.

⇒ The state of practice in DevOps is focusing largely on the culture and training. There are a lot of processes used to monitor, status check, tooling to understand the runtime performance & operational performance of the system. Some DevOps tips include:

→ don't let designing for deployability be an after-thought.

→ use measureable deployability quality attributes.

→ consider architectural tactics that promote modifiability, testability and operational resilience.

→ use architectural abstractions to reason & about deployability implications of design options & trade' off.

→ establish monitoring mechanisms.

⇒ Perspectives of software architecture in cloud computing:

→ Cloud Computing & architecting.

→ SLA's cannot prevent failures.

→ In cloud environments.

→ cloud consumers have to design & architect system