Name: IHSAN ULLAH

ID#: 6948

Course: Software Design and Architecture

Submitted to: Aasma Khan

Module: 6th Semester

Date: 23rd June 2020

- **Attempt all questions.**
- **Marks will be given as per the DEPTH of the answer, not LENGTH.**

Question No: 01 (a) What is Software Architecture? Why is software architecture design so important?

Answer:

Software Architecture serves as a blueprint for a system. It provides an abstraction to manage the system complexity and establish a communication and coordination mechanism among components. It defines a structured solution to meet all the technical and operational requirements, while optimizing the common quality attributes like performance and security. Further, it involves a set of significant decisions about the organization related to software development and each of these decisions can have a considerable impact on quality, maintainability, performance, and the overall success of the final product. These decisions comprise of −

a. Selection of structural elements and their interfaces by which the system is composed.
b. Behavior as specified in collaborations among those elements.
c. Composition of these structural and behavioral elements into large subsystem.
d. Architectural decisions align with business objectives.
e. Architectural styles guide the organization.

Importance of software architecture design.

Software architecture design is the foundation of a software system. Like other types of engineering, the foundation has a profound effect on the quality of what is built on top of it. As such, it holds a great deal of importance in terms of the successful development, and eventual maintenance, of the system.

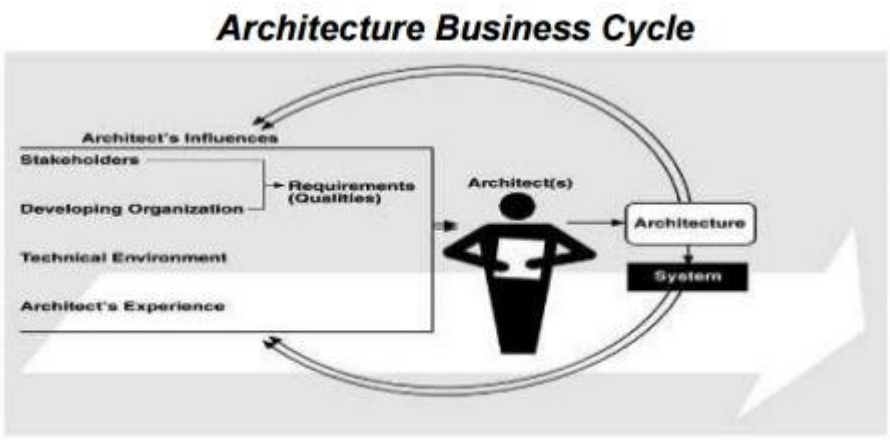
b) Explain any four tasks of architect.

1. Identifying business requirements and requirements of the stakeholders on the project
2. Designing the entire system based on the received requirements
3. Choosing the system architecture and each individual component of this system at a high level
4. Choosing the technologies for the implementation of each component and connections between the components

(Question No: 02) Explain Architecture Business Cycle (ABC) in detail with figure.

Answer

Software architecture is a result of technical, business, and social influences. Its existence in turn affects the technical, business, and social environments that subsequently influence future architectures. We call this cycle of influences, from the environment to the architecture and back to the environment, the Architecture Business Cycle (ABC).



1.The organization goals of Architecture Business Cycle are effected requirements, which create an architecture, which creates a system. The architecture flows from the architect's experience and the technical environment of the day.

2.Three things required for ABC are as follows:

- **Case studies** of successful architectures crafted to satisfy demanding requirements, so as to help set the technical playing field of the day.
- **Methods** to assess an architecture before any system is built from it, so as to mitigate the risks associated with launching unprecedented designs.
- **Techniques** for incremental architecture-based development, so as to uncover design flaws before it is too late to correct them.

The architecture affects the –

- Structure of the developing organization.
- Goals of the developing of the organization.
- Customer requirements with reusability.
- The process of the system building will affect the architect’s experience with subsequent systems.

Architecture business cycle changes- ◦

- Org. goals to req.
- Req. to arch.
- Arch. to systems.
- Systems to org

Influences

- Technical, business, social.
- Stakeholders, other source.

ABC activities include

- Create the business case.
- Understand the requirement.
- Create the architecture.
- Document & communicate the architecture.
- Analyse the architecture.
- Implement the system based on architecture
- Confirms the implementation.

(Question No: 03) Explain ABC Activities?

ABC includes the following activities

a. Create the business case.
b. Understand the requirement.
c. Create the architecture.
d. Document & communicate the architecture.

e. Analyse the architecture.
f. Implement the system based on architecture
g. Confirms the implementation.

Creating the business case for the system

It is simple to create a business case than understanding the needs of market How much should be the product cost? What is the Targeted market? What is the targeted time to market? Will it need to interface other system? Are there system limitations

Understanding the requirements

There are variety of techniques to understand requirements from stakeholders. Object oriented analysis: use cases & scenarios Safety Critical Systems: Finite state machine models Formal specification languages Quality attributes Prototypes Regardless of technique used, -- the desired qualities of the system to be constructed determine the shape of architecture. | Website for Students

Creating the architecture

Conceptual integrity A small no. of minds coming together to design the system's architecture.

Communicating the architecture

For effective architecture It must be communicated clearly and unambiguously to all stakeholders. Developers must understand work assignments. Testers must understand the task structures Management must understand the scheduling implications

Analyzing the architecture

Out of multiple designs, after analyzing, some design will be accepted or some are rejected. Evaluating an architecture for the qualities it supports is essential to ensure the stakeholders satisfaction (needs). Scenario- based techniques are for evaluation of architecture. | Website for Students

Implementing based on the architecture

Concerned with keeping the developers faithful to the structures. Should have an environment that assists developers in creating the architecture. Ensuring conformance to an architecture Finally, when an architecture is created and used, it goes into maintenance phase. Constant vigilance is required to ensure that actual architecture and its implementations remain faithful to each other.

Confirming the implementations

The final step in the cycle is to confirm the implementations and reviewed by a single architect or small group of architects. gather both the functional requirements and a well specified, prioritized list of quality attributes. be well documented, with at least one static view and one dynamic view. be reviewed by the system's stakeholders. be analyzed for applicable quantitative measures and formally evaluated for quality measures.

Question No 04:

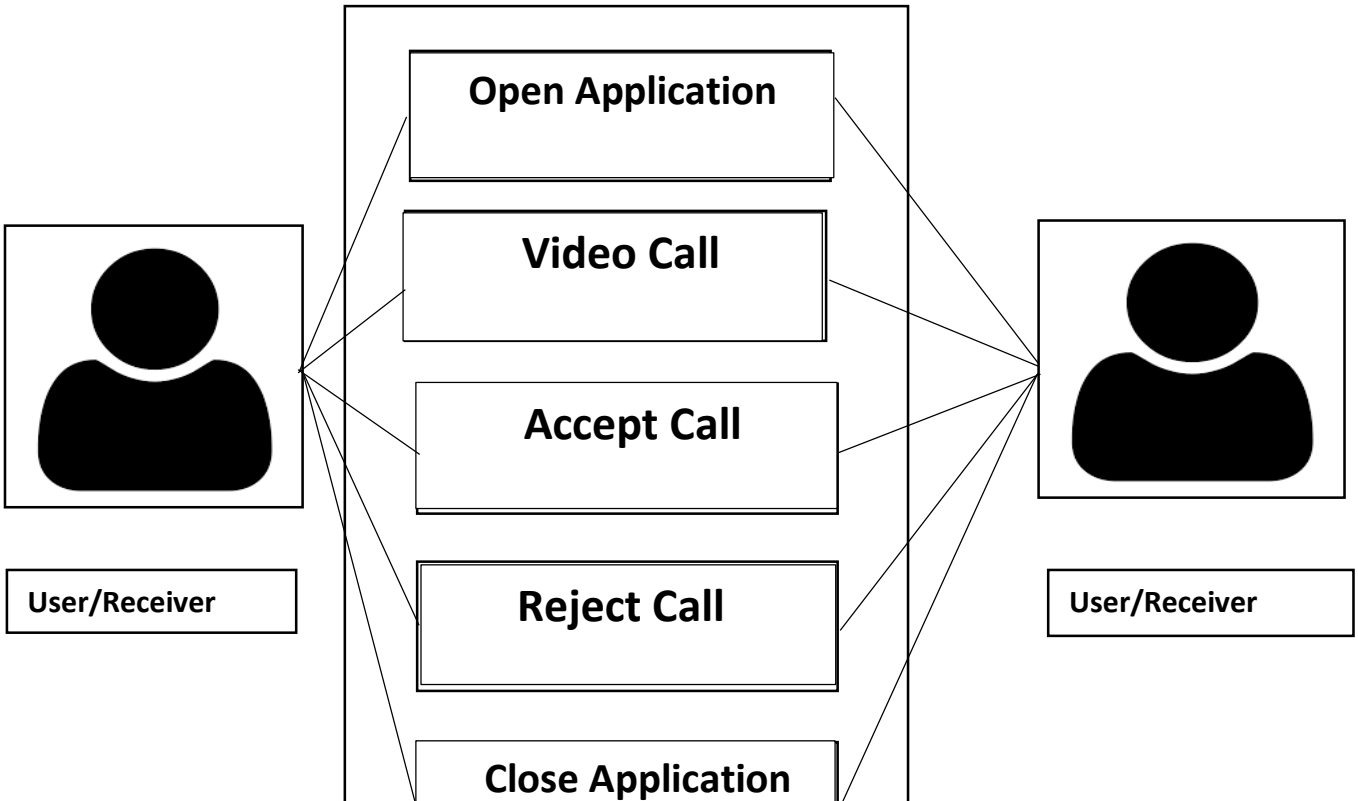
Pair programming is an agile software development technique in which two programmers work together at one work station. One types in code while the other reviews each line of code as it is typed in. The person typing is called the driver. The person reviewing the code is

called the observer. The two programmers switch roles frequently (possibly every 30 minutes or less).

Suppose that you are asked to build a system that allows Remote Pair Programming. That is, the system should allow the driver and the observer to be in remote locations, but both can view a single desktop in real-time. The driver should be able to edit code and the observer should be able to "point" to objects on the driver's desktop. In addition, there should be a video chat facility to allow the programmers to communicate. The system should allow the programmers to easily swap roles and record rationale in the form of video chats. In addition, the driver should be able to issue the system to backup old work.

- **Draw a use case diagram to show all the functionality of the system.**
- **Describe in detail four non-functional requirements for the system.**
- **Give a prioritized list of design constraints for the system and justify your list and the ordering.**
- **Propose a set of classes that could be used in your system and present them in a class diagram**

Answer



For Non-Functional requirements for the system:

Ease of Use: The front-end interface must be simple and easy to use.

Availability: The system will be available all the time.

Probability: The users will be able to use the application regardless of what their computer or system might be

Cost: User should not pay more than 5$ a month.

Give a prioritized list of design constraints for the system and justify your list and the ordering.

Reason:

We prioritized the list as follow

- Easy to use
- Availability
- Probability
- Cost

Our top priority is that the application is easy to use so every user can operate the application and will not have any kind of problem while working, the availability of the application for 24 hours is necessary but due to updates it might not provide 24 hour service , the probability of the device would be of great service regardless of the system

specifications of the user and with less cost the user can easily acquire the product.

Example" Security"

The system must be secured is NFR. The design constraints could be user authentication and it must be in place., the communication protocol must be encrypted, and the data must be stored on a server behind firewall.

Propose a set of classes that could be used in your system and present them in a class diagram