Final Assignment

Software Design & Architecture

Submitted By: Muhammad Zain Ul Abideen
ID# 13740
BSSE

Submitted To: Respected Ma'am Aasma Khan (Lecturure)

Final Assignment

Question 1:

a) What is Software Architecture? Why is software architecture design so important?

Ans: **Software Architecture:**

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.

i. 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 −

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

Importance:

There are three main points of software architecture and design which are as follow:

1. **Communication among stakeholders:** Software architecture represents a common abstraction of a system that most if not all of the system's stakeholders can use as a basis for mutual understanding, negotiation, consensus, and communication.
2. **Early design decisions:** Software architecture manifests the earliest design decisions about a system, and these early bindings carry weight far out of proportion to their individual gravity with respect to the system's remaining development, its deployment, and its maintenance life. It is also the earliest point at which design decisions governing the system to be built can be analyzed.
3. **Transferable abstraction of a system:** Software architecture constitutes a relatively small, intellectually graspable model for how a system is structured and how its elements work together, and this model is transferable across systems. In particular, it can be applied to other systems exhibiting similar quality attribute and functional requirements and can promote large-scale re-use.

b. Explain any four tasks of architect.

Ans: The four basic tasks of an architect are:

1. An architect performs static partition and decomposition of a system into subsystems and communications among subsystems.
   - A software element can be configured, delivered, developed, and deployed, and is replaceable in the future.
   - Each element's interface encapsulates details and provides loose coupling with other elements or subsystems.

2. An architect perform tradeoff analysis on quality attributes and other nonfunctional requirements during the selection of architecture styles.
   - For example, in order to increase a distributed system's extensibility, portability, or maintainability, software components and Web services may be the best choice of element types, and a loose connection among these elements may be most appropriate.
3. Architect establish dynamic control relationships among different subsystems in terms of data flow, control flow orchestration, or message dispatching.
   - An architect control the flow of data in a software system and also explain the path for data flow.
   - If there is any problem in data flow or control flow orchestration a message has to be dispatch.
4. Consider and evaluate alternative architecture styles that suit the problem domain at hand.
   - An architect has to keep an alternative style for that software designing if any problem arises he has to be completely ready for solving that problem without any further delay.

Question No: 02

Explain Architecture Business Cycle (ABC) in detail with figure.

Definition: **Architecture Business Cycle (ABC)**: "Architecture Business Cycle (ABC) is description of a system, used to represent relationship among structures/ components of the system to the environment in which the system is developed and implemented."

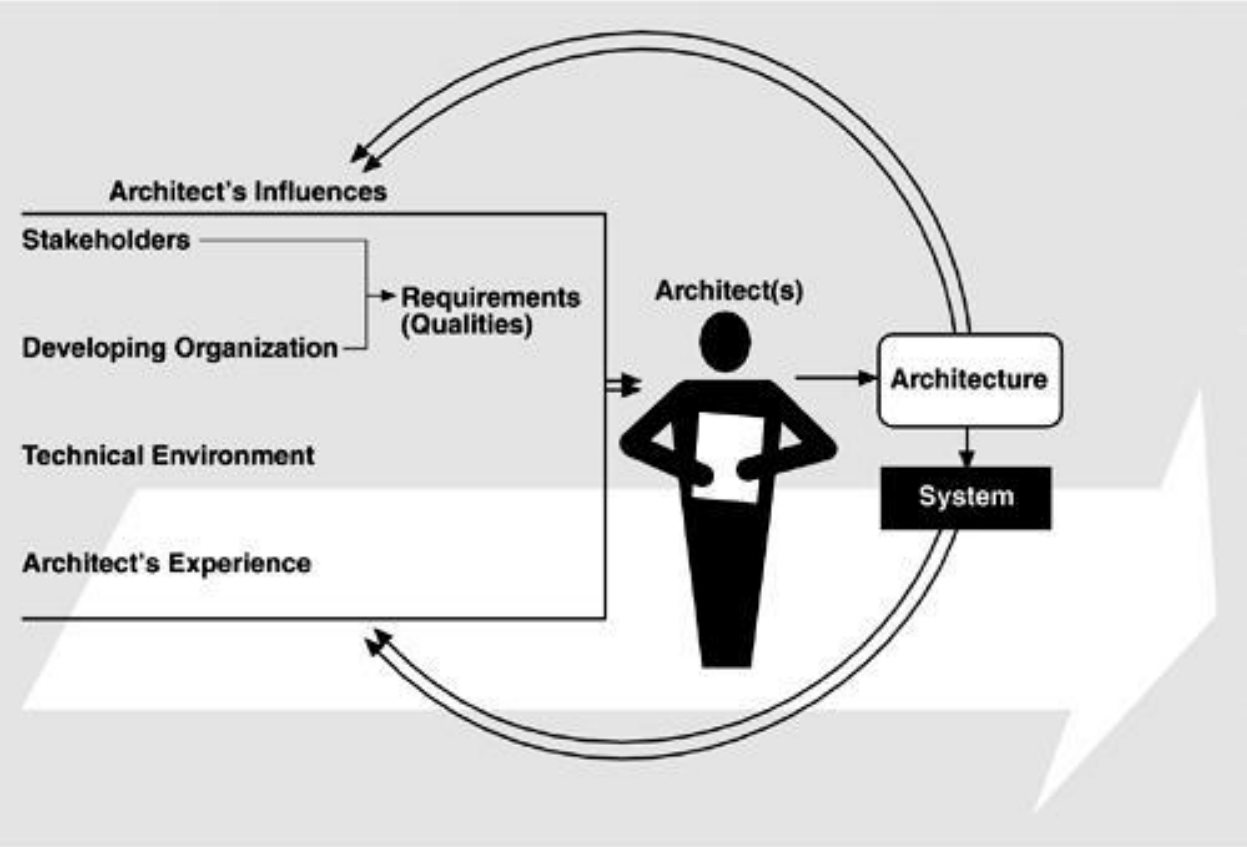


Fig: Architecture Business Cycle (ABC)

The organization goals of **Architecture Business Cycle** effect requirements, which effect an architecture, which effects a system. The architecture flows from the architect's experience and the technical environment of the day.

Main three requirements for Architecture Business Cycle (ABC) are as follow:

i. Case study

ii. Methods

iii. Techniques

i. Case studies: of successful architectures crafted to satisfy demanding requirements, so as to help set the technical playing field of the day.

ii. Methods to assess an architecture before any system is built from it, so as to mitigate the risks associated with launching unprecedented designs.

iii. Techniques for incremental architecture-based development, so as to uncover design flaws before it is too late to correct them.

Building the ABC: To build an ABC we have to identify the influences to and from architectures

1. Stakeholders influences the architecture:

Stakeholders are:

· The customer,
· the end users,
· the developers,
· the project manager,
· the maintainers, and
· even those who market the system.

Stakeholders have different concerns that they wish the system to guarantee or optimize, including things as diverse as providing a certain behavior at runtime, performing well on a particular piece of hardware, being easy to customize, achieving short time to market or low cost of development, gainfully employing programmers who have a particular specialty, or providing a broad range of functions.

. Influence of stakeholders on the architect

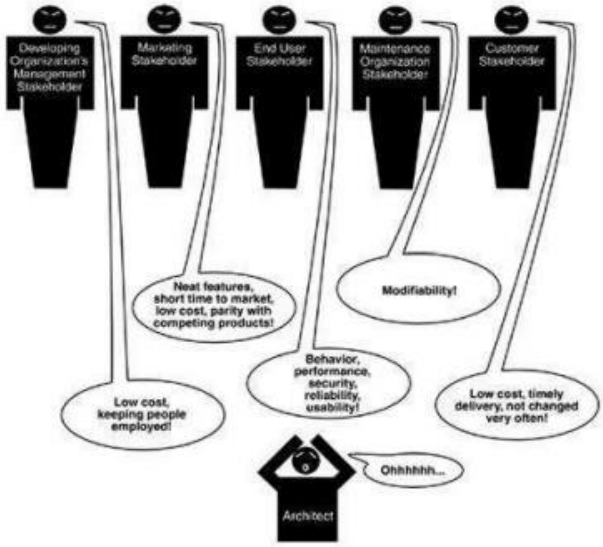


Fig: Stakeholders giving suggestions to the architect

2. Architecture are influenced by the developing organization:

1. In addition to the organizational goals expressed through requirements, an architecture is influenced by the structure or nature of the development organization.

2. For example, if the organization has an abundance of idle programmers skilled in client-server communications, then a client-server architecture might be the approach supported by management.

3. If not, it may well be rejected. Staff skills are one additional influence, but so are the development schedule and budget.

There are three classes of influence that come from the developing organization: **immediate business, long-term business, and organizational structure**.

3. Architecture are influenced by the background and experience of the architects:

1. If the architects for a system have had good results using a particular architectural approach, such as distributed objects or implicit invocation, chances are that they will try that same approach on a new development effort.

2. Conversely, if their prior experience with this approach was disastrous, the architects may be reluctant to try it again.

3. Architectural choices may also come from an architect's education and training, exposure to successful architectural patterns, or exposure to systems that have worked particularly poorly or particularly well.

4. The architects may also wish to experiment with an architectural pattern or technique learned from a book (such as this one) or a course.

Architectures are influenced by the technical environment:

1. A special case of the architect's background and experience is reflected by the technical environment.

3. The environment that is current when an architecture is designed will influence that architecture.

4. It might include standard industry practices or software engineering techniques prevalent in the architect's professional community.

5. It is a brave architect who, in today's environment, does not at least consider a Web-based, object-oriented, middleware-supported design for an information system.

Ramifications of influences on an architecture:

1. Influences on an architecture come from a wide variety of sources. Some are only implied, while others are explicitly in conflict.

2. Almost never are the properties required by the business and organizational goals consciously understood, let alone fully articulated.

3. Indeed, even customer requirements are seldom documented completely, which means that the inevitable conflict among different stakeholders' goals has not been resolved.

4. However, architects need to know and understand the nature, source, and priority of constraints on the project as early as possible.

5. Therefore, they must identify and actively engage the stakeholders to solicit their needs and expectations.

6. Without such engagement, the stakeholders will, at some point, demand that the architects explain why each proposed architecture is unacceptable, thus delaying the project and idling workers.

7. Early engagement of stakeholders allows the architects to understand the constraints of the task, manage expectations, negotiate priorities, and make tradeoffs.

8. Architecture reviews and iterative prototyping are two means for achieving it.

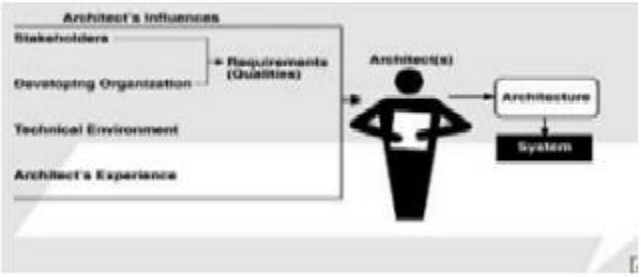
9. It should be apparent that the architects need more than just technical skills.

10. Explanations to one stakeholder or another will be required regarding the chosen priorities of different properties and why particular stakeholders are not having all of their expectations satisfied.

11. For an effective architect, then, diplomacy, negotiation, and communication skills are essential.

12 .The influences on the architect, and hence on the architecture, are shown in figure. Architects are influenced by the requirements for the product as derived from its stakeholders, the structure and goals of the developing organization, the available technical environment, and their own background and experience.

Influences on the architecture



The architectures affect the factors that influence them:

1. The main message of this book is that the relationships among business goals, product requirements, architects' experience, architectures, and fielded systems form a cycle with feedback loops that a business can manage.

2. A business manages this cycle to handle growth, to expand its enterprise area, and to take advantage of previous investments in architecture and system building.

3. Some of the feedback comes from the architecture itself, and some comes from the system built from it.

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.

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).

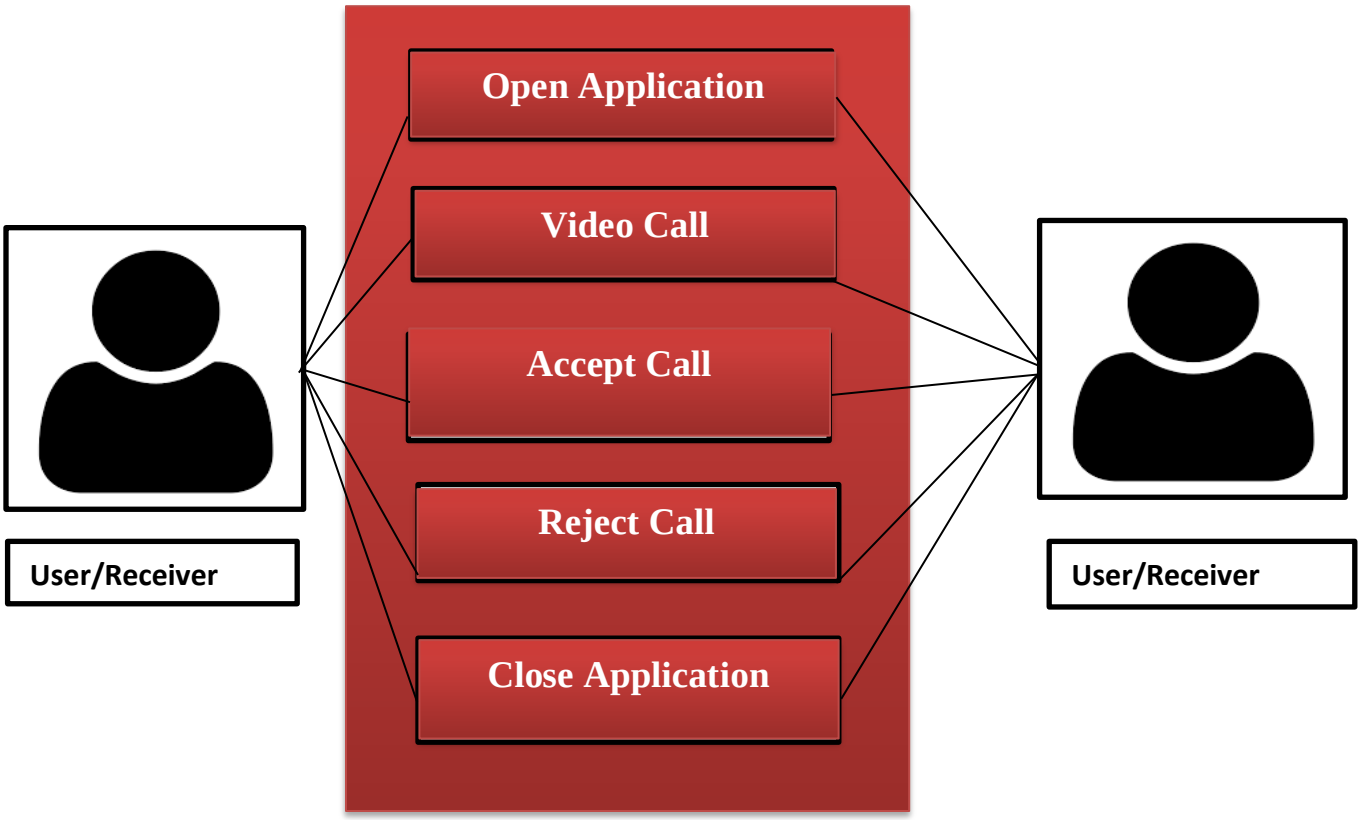
Suppose that you are asked to build a system that allows Remote Pair Programming. That is, the system should allow the driver and the observer to be in remote locations, but both can view a single desktop in real-time. The driver should be able to edit code and the observer should be able

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

Use case diagram of all functionality:

Answer



For Non-Functional requirements for the system:

Ease of Use: The front-end interface must be simple and easy to use.
Real-time performance: The Observer should be able to see the changes made by the Driver immediately without delay; the video chat should be smooth without delay also.
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

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