Architecture Assessment

Client

<Client Name>

Input

Vision / Target

1. What is the vision of the required architecture?
2. What is the short-term target/goals to be achieved?
3. What is the long-term target/goals to be achieved?

Business requirements

1. Who is the target audience?
2. What is the expected load?
    a. The number of customers?
    b. The number of back-office users?
    c. The number of daily transactions?
3. What are the current issues you are trying to solve?
4. For non-functional requirements, what
    a. 24 x 7?
    b. Best suitable planned downtime timing?
    c. RPO (Return Point Objective)? … Maximum amount of data (Measured by time) that can be lost after recovery from disaster or failure.
    d. RTO (Return Time Objective)? … min

Project Timeline

1. What is the target go-live date?
2. Is the phased approach accepted?
3. What are suggested prioritized phases from the client's perspective?

Global compliances

1. Is there any country-based compliance required?
    a. The Federal Risk and Authorization Management Program (FedRAMP) and Department of Defense Cloud Computing Security Requirements Guide (DoDSRG) for the USA
    b. The General Data Protection Regulation (GDPR) for Europe
    c. The Information Security Registered Assessors Program (IRAP) for Australia
    d. The Center for Financial Industry Information Systems (FISC) for Japan
    e. The Multi-Tier Cloud Security (MTCS) standard for Singapore
    f. The G-Cloud for the UK
    g. The IT-Grundschutz for Germany
    h. The Multi-Level Protection Scheme (MLPS) Level 3 for China
2. Is there any industry compliance required?
    a. Finance: Payment Card Industry Data Security Standard (PCI DSS).
    b. Healthcare: Health Insurance Portability and Accountability Act (HIPAA).

3. Is Quality compliance required (ISO 9001)?
4. Is Organizational Security (SOC2) compliance required?

Budget

1. What is the CAPEX?
2. What is the OPEX?

Constraint

1. Cost?
2. Timeline?
3. Scope?
4. Resources?
5. Technology?
6. Cloud accepted?
7. On-premise is a must?

Teams

1. Is the team local/remote?
2. Is it ok to outsource / Managed Services?
3. What is the existing skillset for the team?

Solution Architect Responsibilities

Analyze functional requirements

ASIS Assessment

1. Technology assessment:
    a. Existing technology stack.
    b. Is there a complete outdated technology that needs to get replaced?
    c. Is there a newer version that is backward comptabile with existing technology?
2. Architecture assessment:
    a. Audit the architecture in the aspects of scalability, availability, performance, and security.
3. Code and dependency assessment:
    a. Is there documentation for the code?
    b. If the code upgradable?
    c. If the code maintainable?
    d. Is there dependencies across various modules and code files?
    e. If UI upgrades are required to make the user experience better.

Define modernization strategy

1. If part of the application is not used anymore, consider retire.
2. Consider replacing by SaaS solution as quick way to replace providing out of the box functionality.
3. Focus only on modernizing applications that are true differentiators to your business.

Modernization approach

1. Migration tools evaluation.
2. POC for the migration to identify the gaps.
    a. Architecture-driven modernization: Incase you need significant architectural changes. Start implementing the most critical feature first and then build POC to highlight the gaps and required effort.

    b. System re-engineering: If the legacy system is over complicated and requires long-term projects. It requires indepth understanding and perform reverse engineering to build modernized application.
        i. Start with application modernization first.
        ii. Build a mechanism where legacy and upgraded modules co-exist with the ability to communicate in a hybrid manner.
        iii. DB upgrade is last cutover.
    c. Migration and minor enhancments: If existing system technology works relatively well but is restricted due to hardware limitations and cost.
3. Consider other IT domains that require substantial redesign and modernization:
    a. Operating system.
    b. DB.
    c. Security.
    d. Identity.
    e. Network.
    f. Storage.
    g. Backup.
    h. DB tools.
    i. Monitoring and management tools.

Modernization techniques

1. Encapsulation: API wrapper for your application. To be used if the code is well written and maintained. No benefit from technology advancements and hardware flexibility.
2. Rehosting: Migrate your application to another hardware provider without code changes. To be used if there is an urgent need to move from an existing contract quickly.
3. Replatforming: If the server/platform reaches EoL, you might need to rebuild the binaries with the new operating system or platform changes, it requires testing, and should not include so many code changes.
4. Refactoring: If the technology is still relevant and can accommodate business needs with code changes, refactoring the code to accommodate the new system will be the way forward, while the overall architecture will be the same.
5. Rearchitecting: reutilize existing code as much as you can and change the architecture in a phased approach.
6. Redesigning/replacing: If the existing system is completely outdated, and can't accommodate business requirements. You are building the whole system from scratch implementing the overall scope. This is a long-term project with lots of effort and increased cost. Before taking this approach it is better to check if any SaaS or COTS products can handle your business at a lower cost. It is essential to do a cost-benefit analysis (CBA) between redesign and purchase before proceeding with this decision.

Define non-functional requirements

Scalability & Elasticity
High Availability & Resiliency
Fault tolerance & Redundancy
Disaster Recovery & Business Continuity
Extensibility & Reusability
Usability & Accessibility
Portability & Interoperability
Operational Excellence & Maintainability
Security & Compliance
Cost optimization & budgets
Performance
Reliability
Recoverability
Usability

Understand and engage stakeholders

Understand constraint

Prototyping and proof of concept (POC)

The objective to develop a POC is to evaluate technology with a subset of critical functional implementations, which can help us to decide on a technology stack based on their capabilities.

Output

Solution Overview

Technology Selection

Solution Implementation

Infra-structure requirements

Solution Maintenance