Hosam Abdeltawab

Bin-Othmane, Lotfi

Assignment 3

November 30, 2017

***Introduction.***

The objective of this paper is to assess an architecture design. The architecture assessment process is used to aid in development of enterprise, component-based, web applications. The client for this service typically has to build a complex software system and, in an early stage of the development, use the architecture assessment to validate the decisions being taken in regard to the architecture of the system.

This assessment takes place when the functional requirements are already identified, and the client has at least a primary view of the architecture. World-class companies go to the client’s site and work close to the lead architect of the system to evaluate and evolve the architecture. Very often, non-functional aspects of the system, such as performance, security, availability, are not considered until final stages of the development process. Aware of that, this architecture assessment leverages the discussion of these non-functional requirements in the form of quality attributes. The architecture assessment process has the following objectives:

* **Clarify quality attribute requirements for the system.**
* **Review the architecture of the system and its documentation to provide a solution that can meet functional and non-functional requirements.**
* **Document architectural decisions and rationale.**
* **Identify risks early in the life-cycle.**

***Architecture Evaluation***

The architecture does a good job in explaining multiple views of the team as well as the client. The multiple iterations help in mapping the different components from one view to the other. In arranging the complexity of the system, the team did a great hob in providing different levels of abstraction. They have managed to establish those levels of complexities on which a person can visually interact with the system.

The architecture helped the team to provide a solid foundation for the project, make their platform scalable, increase the performance of their platform, reduce costs, avoid code duplicity, identify areas for cost savings, better maintain their code, increase the quality of their platform, manage complexity, identify risks, reduce market time, and prioritize project goals. The textual description in providing a better understanding for the diagrams.

The architecture does a good job in the description of the SmartSync system by providing an architecture of the existing software as well as the extended one, providing a textual description of their custom library, and providing a textual description of their services breakdown (e.g. API Gteway, Config Service, User Service, Household Service, Invite Service, Service Service, Todo List Service, Weather Service, Zipkin Service, and Eureka Service).

***Summary***

Overall, this team’s architecture was stunning. What was good was the pre-identification of terms as well as concepts. They presented their ideas perfectly by providing a project scope and multiple views of the architecture. Also, they provided well-organized diagrams that aided in providing a solid foundation for the project (e.g. Use case diagram, Module view diagram, and Component diagram). Finally, the team’s architecture designs were well formatted helped in providing a well-developed explanation that aided in the overall understanding of the project.

***Architectural Risks***

In evaluating this architecture, I identified some risks that could make the project vulnerable was *time;* due to the nature of this course and that the team have only two weeks to implement this project, time will be a very important risk. Another risk would be *productivity,* on projects involving short timelines, developers tend to take things easy to begin with. As a result, sometimes, they lose significant time to complete the project. Set a realistic schedule, and stick to it.

Another risk would be *technical risk,* sometimes software development teams reduce the functionality of the software to compensate for overruns pertaining to high budgets and scheduling. There is always a conflict between achieving maximum functionality of the software and peak performance. In order to compensate for excessive budget and schedule overruns, teams sometimes reduce the functionality of the software.

Moreover, other risks include, *procedural risks, and Breakdown of specification;* Day-to-day operational activities might hamper due to improper process implementation, conflicting priorities, or a lack of clarity in responsibilities, and during the initial phases of integration and coding, requirements might conflict. Moreover, developers may find that even the specification is unclear or incomplete.

Finally, there are the unavoidable risks, these include changes in government policy, the finality of software or other risks that cannot be controlled or estimated. As the field of software development becomes more and more complex, the risks associated with it have intensified. It is vital that development firms focus on strategic planning to mitigate such risks.

***Conclusion***

In conclusion, in assessing the architecture, I have gained a multitude of effective tools and methods that will help me in my future encounters with software architecture. The team’s architecture and thoughts were well defined as well as organized. The use case diagram helped in understanding all the use cases for the project, the component diagram aided in understand all the components for this project. The module diagram aided in understanding how the components and interfaces of the pre-defined smart sync and the current project interacts with each other. The identified risks should be looked and taken care of by team to help their project prosper forward. Finally, the team’s architecture designs were well formatted and developed that aided and helped in providing a well-developed and easy to understand explanation that aided in the overall strength of the project.