Architecture Evaluation Model

QOS Capture & Architecture Evaluation Model
QOS capture & Architecture evaluation model

Published: June 2010

This article talks about mechanism to capture customer requirements related to Quality of Service and to evaluate a software architecture vis-à-vis these requirements.

Problem Description

Software professionals are often stuck with the problem of either undefined or inadequately defined architecture requirements. While project team will often have exact functional requirements, problem with non-functional requirements is that they are almost never stated explicitly. For example, team developing loan account origination software would know that there are various steps involved e.g. check if customer exists, if not, create customer, open the account, accept initial cash deposit, send a message to card management system to print and emboss a debit card and so on. However, how often do you hear that requirement is to open 20,000 accounts per day or that the account opening software should comply with DDA (Disability Discrimination Act)? Similarly, for a distribution setup, you will know that there are so and so use cases to be developed as a part of application that needs to be rolled out to various stores. However, customers will rarely inform us of an expectation that if the application changes, it should be distributed to different stores within three hours.

Even if there is a mechanism to capture non-functional requirements, challenge is in prioritizing several requirements all of which look important. For example, for a large enterprise level application developed on a legacy platform, it may be important to revamp UI it to improve usability. It may also be important to separate business and presentation logic to move towards Service Oriented Architecture to offer functionalities through different channels. Besides prioritization, project teams may need inputs as to how to address some of the architecture issues using best practices which have been followed by architects even working in other domains over the years. Finally, there may be a
requirement to measure compliance of a given deliverable from the point of view of given architectural requirements.
Solutions Overview

Current Problems

- Architecture related requirements are generally not explicitly stated
- Need for a checklist for possible solutions to architecture problems
- Need to get a prioritized checklist of actions for architecture improvement based on criticality of requirement, cost and impact of the resolution
- To measure the maturity of architecture

Proposed System

- Template to capture non-functional requirements with priorities
- Template to suggest design alternatives
- Template to capture cost/impact and criticality to arrive at prioritized list of architecture improvements

Solution

The QoS capture and Architecture Evaluation model that we have developed assists teams in capturing and meeting Quality of Service requirements of various projects. This model can be used at the requirements stage of a green field development project to capture the Quality of Service requirements in a structured manner. For this purpose, the model will take user through a structured mechanism to ask questions on various aspects on availability, modifiability, performance and scalability, usability, security and testability. These have been used since there is ample literature available for ATAM from CMU. Terminologies used within ISO 9126 and relevant parts of TOGAF can be mapped to these terminologies. What we have done beyond the existing literature is to take a stock of commonly asked non-functional requirements and put these within structured template. We have kept the model technology agnostic and hence it has questions like “resilience at presentation tier” or “resilience at application tier” or “DDA compliance” which has nothing to do with any specific technology.
During the design phase, architects and project teams can use the same model to assist architect and the project teams to evaluate various design alternatives to meet non-functional requirements. These alternatives fall in two different categories. Firstly architect can take a decision to implement a certain design pattern to meet a non-functional requirement. For example, for better modifiability architect can ensure that the code is developed in a modular fashion with well defined interfaces so that impact of change in one module on another one is minimized. Alternatively, architect can recommend usage of a rule engine to keep the code clean and externalize decision making within well defined rules. Second category of alternatives is related to deployment. These alternatives would call for addition of certain infrastructure components to meet certain non functional requirements. For example, active-active clustering at database tier would address the problem of availability at database tier. Thus, architect can address the non-functional requirements by using design patterns
within application or deployment or both. These design alternatives are from years and years of learning that architects have developed over multiple projects and so on.

Final problem that we are addressing is the problem of architecture audits or detailed evaluation. We often encounter a situation where in there is a piece of software that is being acquired from other organizations. One organization could be taking over maintenance of software from another one and so on. Similarly, a client organization may want to carry out acceptance testing of software that has been delivered to them. This calls for a “check” or “evaluation” as to whether the software meets the architecture requirements or not. If not, then to what extent is the software non-compliant? Another important question to answer is if there are multiple corrections to be applied to given software to meet the entire set of requirements, what would be the prioritization of these. While CBAM methodology from CMU addresses prioritization of action items on architecture side, we have provided a template to capture these. Template allows evaluator to express things like whether a requirement is in “must be met” category or is in “important” or in “desirable” category. Now, if a certain approach is taken in meeting a certain requirement, what is the cost of this alternative? Whether the cost is “prohibitive” or “significant” or “not significant” and so on? It also
allows evaluator to express as to what is the impact of a certain approach on meeting of a non-functional requirement. Evaluator can choose as to whether a design alternative as “high impact” or “low impact” or “insignificant impact” and so on. Based on all these, we have come up with a rigorous measurement of architectural compliance index and a prioritized list of improvements to be carried out to a given piece of software.

**Future plans**

This model is designed to be self learning. New alternatives that are found over the coming years to meet non functional requirements could be easily added as either design alternatives or deployment alternatives. Newer requirements that would come in future could be added within templates for requirements capture e.g. in coming years, if retina based credentials become mandatory to log in to the system that would find its way into the requirements section.

**Business Benefits**

To summarize, this model offers several business benefits e.g.
1. As a tool to be used during architecture consulting assignments
2. As a framework to assess architecture compliance for acquisition of software products
3. To arrive at roadmap for architecture improvement of existing products

This model can be applied during:

- Pre-sales or Requirements stage to elicit Non Functional Requirements from clients
- Design stage To provide suggestions on application design
- Deployment stage to check Architecture compliance and robustness and to provide system integrators with a possible deployment strategy
- Post deployment stage to come up with the improvement strategy of the existing system as a long term plan
Authors

Deepak Hoshing heads Finacle architecture and has been architecting variety of solutions for over two decades. He has been with Infosys about 14 years.
Sanjita Bohidar works with Infosys Quality department and has been with Infosys for about 13 years.

For More Information

For more information about Infosys and its capabilities visit the World Wide Web at http://www.infosys.com

© 2010 Infosys. All rights reserved.

This document is for informational purposes only. Infosys MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS SUMMARY.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.