Software Architecture – written exam 2016

Written by Rasmus Løbner Christensen (rloc@itu.dk)

CPR number: 190687-2109

Class code: MSAR

Date: 06-01-2015

Disclaimer:

I was unable to actually connect to the internet at the exam. Even the IT-department couldn't solve the problem, even though I was at premises one hour before the exam. This is not meant as an excuse but simply underlining that I did not have access to a variety of the course literature (especially the slides), since I only had some parts of this stored locally on my machine, meaning that some of my answers might be a bit lackluster.

Perhaps you guys could underline for the coming students of this course, that saving all the course literature locally could be a good idea, in case of no internet.

Indhold

Part 1

Question 1

As stated, a general model of Software Architecture has three main activities; Architectural analysis, Architectural synthesis and Architectural evaluation. In the following I will for each of these parts give a short description of the main task involved in each part.

Architectural analysis

Architectural analysis serves to define the problems the architecture must solve. The architect(s) must look at architectural concerns, and in the end come up with a set of ASR's. An ASR being an *Architectural Significant Requirement* (requirement which influences its architecture). A note here is that not all ASR's may be stated as requirements, but may have come from concerns or system context.

Architectural synthesis

Architectural synthesis is more or less the main activity of architecture design (!). With the architectural analysis at hand, the architect has a set of ASR's. The main task in the synthesis part is then to come up with solutions to these ASR's, and therefore moving from the problem to the solution space.

Architectural evaluation

To put it short, the architectural evaluation is about making sure that the architectural design decisions are the correct ones. The proposed or candidate architecture is matched up against the ASR's repeatedly, and in the end the architect(s) is left with a validated architecture. A way of doing this ongoing evaluation could be with an ATAM evaluation.

The above three parts (analysis, synthesis and evaluation) are a general way of putting the activities involved in different approaches to designing a software architecture. Obviously different approaches exist, with some of the key ones being: ADD, S4V, RUP 4+1 and ASC.

Question 2

In the following I will provide a brief explanation of the following concepts.

Sensitivity point

Sensitivity points are a property of one or more components that are critical for achieving a particular quality attribute Trade-off point.

Trade-off point

Tradeoff points is key architectural decisions which impacts multiple components and sensitivity points, and as the name implies, is a consequence of some kind of tradeoff design decision.

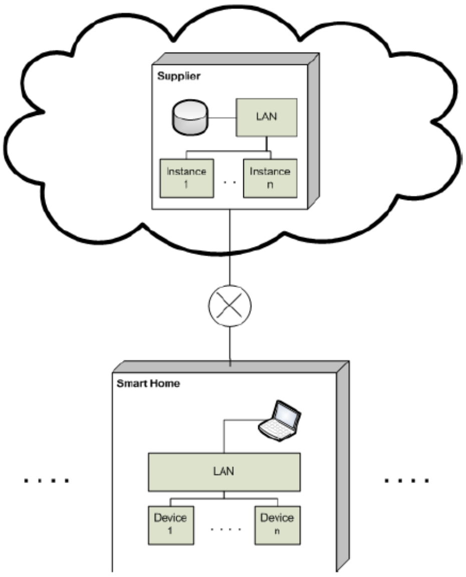
Risk/Non-risk

If an attribute of interest for the architecture/system is conflicting with some design decision, this is said to be a risk. For instance if the architecture of an IoT Smart Home solution states that all communication should run through a router in the household, it would be a reliability risk that this router could face a power failure. Non-risk is design decisions which do not impact the attributes of interest.

Part 2

Question 1

Cloud computing is an architectural way of designing a system. For instance in a IoT Smart Home one could imagine a solution where system was "stored" in the cloud at, and all communication would therefore go through this cloud, leaving the given household with no need to have some kind of server or terminal located in their household. The below is taken from my groups submission in this course, and gives a nice and understandable view of how a cloud solution in general is though out to be.



A main service for a cloud solution could be massage handling or interoperability. For instance in a IoT Smart Home the cloud solution must be able to process messages from all the connected households in a reliable and persistent way. Imaging if the notification of, e.g. lack of milk in the fridge, got prioritized above a potential fire alarm, this could have terrible consequences. A way of dealing with this could be via a broker pattern (with some prioritized queue). The Deployment (view) in this context (IoT) would be how the users register new devices and/or configure the system. This should ideally happen more or less automated, with of course potential options for the user to configure.

Question 2

The three main security challenges involved in designing modern web-based mobile application are *encryption*¸ *digital signatures* and *authentication*.

Encryption should shortly protect messages from eavesdroppers, digital signatures should ensure information integrity and last authentication should provide some strong cryptographic mechanism to identify callers.

The solution to the above mentioned challenges could be the use of *WS-Security* .

Question 3

The core principles of designing a service-oriented architecture are the following four[1]:

- Boundaries are explicit
- Services are autonomous
- Share schemas and contracts, not implementations
- Service compatibility is based on policy

To give examples of the above I will again take base in the earlier mentioned IoT Smart Home application.

In order to reach explicit boundaries, the architect needs to understand how to make services with simple interfaces and furthermore, services that do not share a lot of abstractions. In this understanding of what a given service is capable of and how to actually communicate with this, is explicit. In a IoT Smart Home this would make the creation of different sensors and microchips a lot easier for the given vendors, since they all needed to communicate through the same contracts.

Autonomous services are more or less a continuation of the above explicit boundaries. Services need to know nothing about any other than themselves – no knowledge about other services or some client application. In the IoT example, this means that our *Fire Alarm* service do not know anything about others than itself and the needed logic in order to resolve this situation.

Since the whole focal point of service oriented architectures is simplicity, a service should only be an application able to receive and send messages. Therefore clients and services should share nothing but the definition of these messages. This also means the implementation of these services might look completely different.

Obviously the clients using a service must be compatible with the given service. This means that the message formats should be correct, but most importantly other potential security requirements stated by the given services policy.

Question 4

A pattern consists of the three parts: context, problem and solution.

In the given example I think that the problem part is left out. We have the context given in the first two lines and then the rest is part of solution. We then learn from the solution part that some of the problems are inefficient use of aircrafts, aircrafts not flying at full capacity or aircrafts not flying frequently enough.

The following is an attempt to write the missing part (obviously put in after the context description):

The problem with doing this in an efficient way is that such a solution should always ensure that aircrafts are being used in an optimal way, meaning that aircrafts should always take off with (or close too) full capacity as well as flying as frequently as possible per day.

[1] Essential Software Architecture 2nd Edition by Ian Gorten (page 68)

Part 3

Question 1

[Write something about ATAM activities: Presentation of the system from different stakeholders? Looking at SRA's whether these actually are met (do we even have SRA's?).]

Question 2

Looking through the different User stories and personas in the architecture material, the following two quality attributes have been identified as most important:

Security

Since the financial organizations are required to look into customers personal credit card histories, it is at most important that the security of these services is secured, following all standards within banking services. The financial organizations have to be correctly authorized, as well as all the communication going in/out the system should be encrypted in a correct way. Further in case of dispute, the Law enforcement agencies must be able to access the communication history between the customer and the financial organization in a secure way.

Scalability

The system is already being used by millions across the globe, meaning that need for scalability is core. Since this system potentially could be a dominant factor within car rental/leasing, it is at most importance that the system can guarantee some high level of customer threshold, as well as low and measurable respond times.

[Write about the different patterns that might've been used. Layered pattern perhaps?]