# ATAM

Meget lig QAW, men hvor der tages udgangspunk i en arkitektur design. Struktureret og relativ verbos, men der er dog mulighed for at skræddersy den til behov. Anvender scenarier, quality attributes, business drivers and architectual approaches. Business drivers er gode til at sikre at kvalitetsattributerne (og dermed arkitekturen) er stilet efter de konkrete forretningsmæssige behov, og ikke bare hvad arkitekten synes er bedst (sjovest). Quality attributes er gode til at definere de non-functional dele af kravene til arkitekturen, og kombineret med scenarier gør det både overskueligt og forståelige for stakeholders.

I forhold til TM12 …

# QAW

Minder om ATAM, men anvendes til at definere arkitekturen. Tager også udgangspunkt i business drivers, quality attributes og scenarier. Lidt mere light-weight end ATAM, men har risiko for at der ikke blive fulgt ordentlig op på analysen.

I forhold til TM12 …

# aSQA

Meget light-weight i forhold til ATAM. Fokus er her at det skal være en process der er hurtig at udføre(3-5 timer) og simpel at kommunikere. Det er tiltængt at metoden gentages ofte(1 gang pr sprint).

I aSQA starter man med at definere hvilke QA’er man ønsker at bruge og hvilken defination de skal have.

Derefter deles systemet op i bider som så bliver vurderet individuelt.

For hver del af systemet gøres følgende:

De valgte qa’er gives en værdi for: Det ønskede niveau(t), Nuværende niveau(c), Så kan Sundheds niavuet beregnes (Forskællen mellem ønsket niveau og nuværende niveau.)med formel: 5-max(0,(t-c)).

Derefter prioteres vigtigheden af QA’en (i) . Nu kan man så beregne en værdi som angiver hvor det største gab er i forhold til vigtighed (f) . Formlen er: ceil( (6-h) \* i/5 ).

Alle tildelte værdier er i skalaen 1-5 hvor 5 er best.

Metoden kommer ikke med konkrete eksempler på hvad som skal forbedres men peger mere på et område som skal forbedres eks: performance. Dette skal ses i forhold til ATAM som resultere i lister med anbefalinger (risks discovered og sensitivity points).

# Praktisk erfaring

Da vi er to i gruppen fra forskellige firmaer har vi naturligvis også forskellige fremgangsmåder.

## Anders

Da jeg arbejder i et lille firma er det nemt at tage diskussionerne i plenum når det er nødvendigt. Samtidig er der meget kort til kunderne, og sælger, projektleder, arkitekt og udvikler har derfor alle et indgående kendskab til business drivers og domænet. De største arkitektoniske udfordringer er nok at da det er så nemt at snakke sammen har man ofte en tendens til ikke at skrive noget ned. Samtidig kan det nemt ske at der justeres i prioriteringen på en dag til dag basis efter hvilken kunde der råber højest den dag, og det kan derfor være svært altid at holde business drivers for øje. Dette håndterer vi med status møder hvor vi gennemgår de brandslukningsopgaver der er, samt de arkitektoniske og udviklingsmæssige opgaver der vil blive udskudt af denne grund. Til tider kan en brandslukningsopgave udskydes ved at kunde simpelthen må leve med det, eller at der kan laves en midlertidig løsning (men her er det meget vigtigt at sikre at den senere bliver rettet – dette er en risk som man skal være bevidst om). Et andet issue er at da der er tale om high dependability SW/HW er vi meget forsigtige med at ændre i arkitekturen, og meget ofte bypasser man hellere et problem frem for at løse det korrekt, hvilket resulterer i code degeneration, hvilket sandsynligvis vil betyde et komplet redesign på et tidspunkt.

## Mogens

I Solar arbejder vi med programmer som bliver brugt i 6 forskellige lande. I landene har vi udpeget område ansvarlige som er primære kontagt til it i forbindelse med nyudvikling. Der er tæt kontagt mellem udviklere og område ansvarlige, hvilket bevirker at der er generel konsensus om QA’s, dog uden at der er nedfældet noget på skrift. I afdelingen har vi defineret for alle programmer hvilke udviklere som er primære udvilkere og seckondære udviklere. Ved nyudvikling vil der normalt altid været en primær udvikler involveret. Ved nyudvikling oprettes RFC (Request for change) dokumenter som evalueres og godkendes først af foretningen i alle lande og dernæst af en arkitekt eller primære udvikler. Normalt sker udviklingen i cykluser af 6-8 ugers perioder, hvor der er test,UAT og release. Sideløbende med dette er der i perioder store projecter som køre et paralelt spor. Vores vigtigeste metode til at sikre at Qualitets attributterne bliver overholdt er, konstant integration og et udvikler team som har 7+ års erfaring. For de fleste programmer har primære udvikler været med fra starten.

Jeg tror vi ville kunne bruge ATAM til at trykprøve vores eksisterende programmer. Jeg kunne forstille mig at man en gang om året ville bruge ATAM til at vurdere de forskellige systemer med henblik på at få belyst problemer og risiko ved valgte design. Det kunne så genere RFC’er til forbedring af systemet.

aSQA tror jeg at vi ville kunne bruges som informations kilde til styregrupperne da det er information der er let at bryde ned for ikke programøre. Det vil kunne være med til at belyse hvis et system har nogle trends gående i negativ retning.