OK kursusdag: ”Testbar arkitektur”

Testbarhed / ”Testability” er en central kvalitetsattribut af en arkitektur, som via automatiske test understøtter kontinuert integration og idriftsættelse. En testbar arkitektur sikrer, at der kun går kort tid fra fejlretning/feature opdatering til at kunden kan nyde godt af den.

En testbar arkitektur er imidlertid også et stort aktiv for udviklerne selv, idet eksperimenter og redesign i kodebasen kan udføres med høj sikkerhed for, at det ikke vil lede til uforudsete fejl i andre dele af systemet. En testbar arkitektur er derfor selve grundlaget for at kunne ”rydde op”, refaktorisere, og øge kodekvaliteten. Det gamle programmør udsagn: ”If it ain’t broke, don’t fix it” er gælder ikke i nævneværdig grad i en testbar arkitektur.

Men – hvordan kommer vi så frem til en testbar arkitektur? Hvad er ”do’s and don’ts”? I dette kursus vil jeg komme ind på taktikker, patterns, metoder, og konkrete programmeringsteknikker, som medvirker til at gøre kode og arkitektur testbart, samt beskrive teknikker til at bringe legacy systemer i en testbar tilstand. De overordnede guidelines vil blive konkretiseret igennem eksempler fra et case, som også danner rammen omkring hands-on øvelser for kursisterne.

Indhold

- Hvad er testbarhed/testabilty?
- Kompositionelt/Rollebaseret software design og dependency injection principper
- Test doubles – stubs, saboteurs, spies, mocks, fakes
- Taktikker/patterns til at gøre arkitektur nær 100% testbar *uden* at produktionskoden indeholder test orienteret kode
- Migrering af legacy systemer til testbare systemer: *Application Modernization*

Kurset vil veksle mellem præsentationer, diskussioner, og hands-on øvelser. De central programmeringsteknikker/patterns bliver der arbejdet med igennem konkrete øvelser, demonstrationer, og hands-on arbejde på et konkret case study. Kodebase og udviklingsmiljø ligger på en virtuel maskine, som bliver udleveret før kursusstart.

Kursus materiale og case er i Java, men alle teknikker og metoder er anvendelige i ethvert moderne objekt-orienteret sprog såsom C#.

Forudsætninger

Case-study og øvelser er baseret på en Java 8 og Gradle stack. Jeg udleverer en VMWare virtuel maskine (VM), der indeholder hele teknologi-stakken i et Lubuntu Linux miljø, samt det case study som opgaverne baserer sig på, så der er ingen opsætningsvanskeligheder. VM’en indeholder desuden IntelliJ som foreslået udviklingsmiljø. Der vil blive en del udvikling i Java, men der vil være stor guidening i opgaverne, så selve

programmingssprogsdelen vil være mindre krævende, det primære fokus er på bredt anvendelige teknikker.

Det er derfor et krav at kursisterne har adgang til en moderne 64-bit laptop med nok hestekræfter til at drive en VM på fornuftig vis. Det er en fordel at kursisterne er nogenlunde rutinerede i almindelige Linux shell-kommandoer.

Materiale

Slides og VM udleveres ca. en uge før kursusstart inklusiv en guide til installation og start.

Slides vil tage udgangspunkt i materiale fra bogen *"Flexible, Reliable Software – Using Patterns and Agile Development"*, af Henrik Bærbak Christensen, CRC Press 2010, så det vil være en stor fordel at kursisterne har adgang til et mindre antal af disse bøger igennem kurset.

OK må benytte materialet internt i organisationen men intet af materialet må distribueres til 3.part eller gøres til genstand for offentliggørelse.

Overordnet program for kursusdagen

Formiddag

- Arkitektur testbarhed og hvorfor testbarheden kan være lav
- Introduktion til Case Study
- Øvelse: Introduktion til kodebasen samt 'bug hunting'

Eftermiddag

- Taktikker og patterns for testbar arkitektur: Kompositionelt design, test doubles, dependency injection og automatiserede test.
- Øvelse: Indførelse af automatiserede tests, test stubs og spies, og dependency injection
- Applikationsmodernisering: taktikker til at få legacy systemer refaktoriseret til testbare arkitekturer

Kursusholder

Henrik Bærbak Christensen har en ph.d. indenfor softwarearkitektur fra Århus Universitet, og har været ansat som lektor selvsamme sted siden 2003. Før hans ph.d. arbejdede han en årrække i industrien, og har igennem hele sit akademiske virke fastholdt en interesse og fokus på industriel softwareudvikling og at *teorier er til for at blive omsat til praksis*. Han har arbejdet sammen med utallige danske it-virksomheder igennem årenes løb såsom Uber, Google, Systematic, B&O, Danfoss, Grundfos, KMD, Jyske Bank, TDC, Kamstrup, Mjølner og mange andre. Dette tætte parløb med industrien ses også i, at Henrik er leder af Efter- og Videreuddannelsen *Master i IT i Softwarekonstruktion*, en del af IT-Vest, hvor fokus er på opkvalificering af ansatte i den danske IT industri.

DevOps og drift er et af de aspekter af softwarekonstruktion, som Henrik har kastet sig over indenfor de sidste fem år, og har i perioden 2013-2017 udviklet og været driftsansvarlig for en Big Data arkitektur, der

opsamlede data fra dels Grundfos kollegiet på Århus Havn, samt fra mobile apps til trafik og kørselsregistrering. Arkitekturen er en micro service arkitektur, bygget til drift 24/7/365, rullende opgradering og deployment, og fuld geografisk redundans af de omkring 18 TB data, der var opsamlet da projektet sluttede primo 2017.

Henrik er meget optaget af undervisning og pædagogisk formidling, og er flere gange blevet nomineret som *Årets Underviser* af de studerende på Datalogisk Institut - og vundet prisen to gange. Han er forfatter af lærebogen *Flexible, Reliable Software - Using Patterns and Agile Development* udgivet af CRC Press, USA, som anvendes på en lang række universiteter verden over. Igennem sin privatejede virksomhed *Imhotep* har han holdt mere end 30 kurser og foredrag for danske virksomheder igennem de sidste 15 år.

Og - så elsker Henrik at kode! *The truth is in the code…*