Imhotep
Vidensformidling indenfor
Softwareudvikling

Test-Driven Development og Design Patterns for Test.

Overblik:

I kurset vil jeg fokusere på hvorledes pålideligt og fleksibelt software i høj grad fremkommer naturligt, hvis man benytter nogle bestemte teknikker og en bestemt vinkel på, hvordan man designer software. De primære teknikker er *test-dreven udvikling (TDD)* som fokuserer på hurtig fremdrift og høj pålidelighed, *refaktorisering* som fokuserer på at sikre stadig høj kvalitet i software arkitekturen, *test doubles* som definerer teknikker til at teste software uafhængigt af hinanden og af eksterne ressourcer, samt *principperne for fleksibelt design* og *3-1-2 processen* som nærmest automatisk indfører gode *design patterns* i ens arkitektur og sikrer at designet får lav kobling og høj samhørighed. Der vil yderligere blive langt vægt på arkitektur og på design patterns til adskillelse af brugergrænseflade og domæne kode, samt til at lette testen af dem individuelt.

Teknikkerne vil blive demonstreret på en konkret case, nemlig en betalingsautomat til en parkeringsplads. Dette software starter som en simpel applikation men i præsentationerne gennemgår jeg en proces hvor nye kundekrav og arkitektur-refaktoriseringer afløser hinanden og demonstrerer agil og test-dreven software udvikling, og hvorledes en mere og mere kompliceret kodebase forbliver pålidelig, overskuelig og fleksibel. Denne proces er også udgangspunktet for hand-on øvelserne i kurset. Jeg vil bevidst fokusere på teknikkernes kode-nære aspekter fordi alle fordelene i et design pattern baseret design alt for let kan ødelægges ved simple fejl på implementationsniveauet. Case er baseret på Java, Ant og vil demonstrere TestNG.

Format:

Kurset vil skifte mellem præsentationer og dialog med kursisterne, hands-on sessioner hvor der løses opgaver i direkte forlængelse af de gennemgåede teknikker i grupper af 2-3 personer, og diskussioner af erfaringerne fra disse øvelser.

Forudsætninger:

Jeg forudsætter at deltagerne kan læse gængs UML 2.0 notation (specielt klasse diagrammer og sekvens diagrammer). Jeg forudsætter at deltagerne kender et objekt-orienteret programmeringssprog og kender standard objekt-orienteret terminologi. Opgaverne i kurset vil være i Java, baseret på Ant og TestNG. Kursisterne forventes at have adgang til et sådant miljø på en PC'er (en per gruppe af 2-3 personer). Det forudsættes at der maksimalt er 15 kursister på kurset for at sikre at jeg får diskuteret med hver enkelt gruppe under øvelserne.

Det forventes, at der er slide projektor til laptop og gerne en tavle i undervisningslokalet.

Imhotep

Vidensformidling indenfor
Softwareudvikling

Litteratur:

Slides. Dele af eksemplerne vil være baseret på JUnit.

Uddrag af min bog "Flexible, Reliable Software: Using Patterns and Agile Development." Bogen beskriver de i kurset dækkede principper og teknikker, men eksemplerne i bogen er dog baseret på JUnit.

Rettigheder:

Slides, uddrag af bogen, og opgavemateriale må frit bruges internt i Cego Aps men ikke videregives til tredje-part. Jeg ejer rettighederne til alt materialet og må anvende det i anden sammenhæng. CRC Press ejer rettighederne til uddraget af min bog.

Dagsplan (udkast):

Undervisningen foregår to dage i træk, fra kl. 9.00 til 16.00. Der beregnes ca. 5 ½ time undervisning pr dag, afbrudt af frokost, samt to pauser af ca. et kvarter i løbet af undervisningsdagen.

Dag 1:

Introduktion af Case, en betalingsautomat for en parkeringsplads. Test-dreven udvikling: TDD patterns og TDD rytmen. Håndtering af en ny produktvariant: Analyse af variabilitetsteknikkerne "source code copy", "parametrisk-", "polymorfisk-", og "kompositionelt" design. Udledning af Strategy pattern igennem en test-dreven process med fokus på refactoring, kompositionelt design og 3-1-2 processen. Ny produktvariant og efterfølgende udledning af State pattern.

Dag 2:

Teknikker til test med kontrol af eksterne resurser: Test Doubles. Pattern skrøbelighed. Rolle begrebet og principperne for fleksibelt design. Design patterns og arkitektur med fokus på afkobling og individuel aftestning af brugergrænseflader og forretningslogik (spillogik)/domæne kode. System, integrations- og unit test i Agil udvikling.