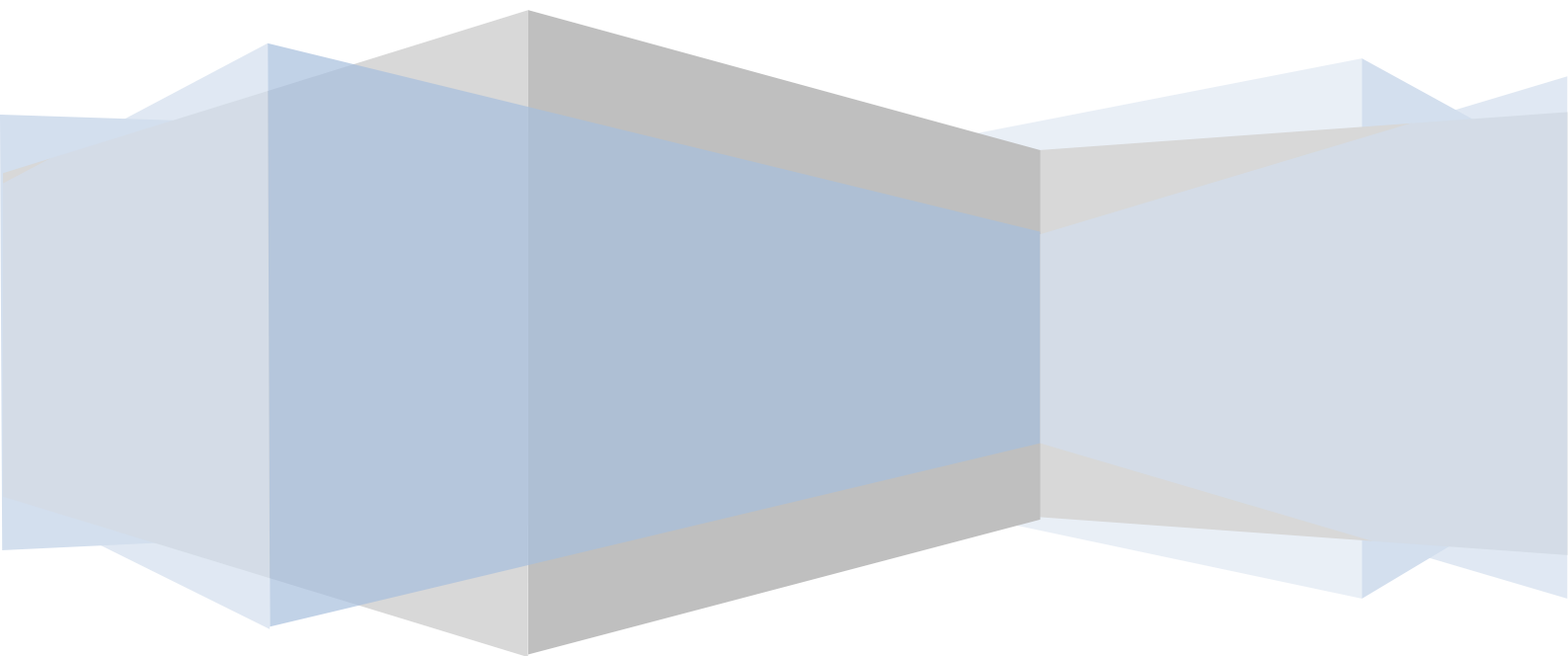


Agile-metoder.dk

World Class Test Management

4. Test ER udvikling

Poul Staal Vinje



World Class Test Management

Af Poul Staal Vinje

Certified Scrum Master

Certified Scrum Practitioner

poul.vinje@gmail.com

www.agile-metoder.dk

www.softwaretest.dk

Artikel 4 om WCTM: Test er udvikling

Indhold

World Class Test Management.....	2
Artikel 4 om WCTM: Test er udvikling.....	2
1. Test ER udvikling	3
2. Teste hvornår?	3
3. To primære processer	4
4. Teknikker og færdigheder	5
5. Samarbejde og kommunikation	5
6. Mindre sluttest, mere automatiseret test	6
7. Fælles produkter	7

1. Test ER udvikling

Denne artikel uddyber det tredje punkt i definitionen på WorldClassTestManagement.

Citat fra den første artikel:

Betragte test som en del af udviklingen. Test bidrager til udvikling. Test og udvikling er to sider af samme sag. Testscenarier bidrager til specifikationen. De præciserer funktionalitet og omfang. Testcases bidrager til kodningen. De detaljerer udviklernes grundlag for at skrive kode.

Citat slut.

Denne artikel handler om at udvikling og test ikke er to forskellige aktiviteter. I et traditionelt forløb udvikler man softwaren, og tester den bagefter. Mere eller mindre vandfald. I et moderne forløb er udvikling og test i større grad integreret. I denne artikel trækkes det ud i det ekstreme. Nemlig at test ER udvikling.

Udsagnet er dels et filosofisk synspunkt, som kan stå alene. Som en måde at anskue verden på. Og dels en overskrift for hvad der kan gøres i praksis for at få tiden der investeres i test til at bidrage aktivt til udviklingen af produktet.

Test og udvikling er to sider af samme sag. De kan ikke skilles ad. Det er som vand der består af H₂O. Hvis man fjerner en af delene har man noget andet. Det er ikke delvist vand. Udvikling uden test er ikke udvikling, det er noget andet. Det er ikke delvis udvikling.

Sammenligningen med vand er langt ude? I hvert fald set med traditionelle briller. Men konsekvenserne af at se dem som to adskilte dele er at begrebet "fejl" opstår. Vi har vænnet os til at arbejde med fejl og fejlrapporter. Defect Management. Selvom det jo ret beset er spild. Det er en fejl at have fejl. Der er noget galt i måden at arbejde på.

WCTM har derfor til formål at forhindre fejl i at opstå. En testcase har ikke til formål at finde en fejl. En testcase har til formål at kaste lys. At afklare noget. Detaljere noget.

2. Teste hvornår?

Tidspunktet er afgørende for om test er udvikling. Testen skal jo drive udviklingen.

Rent praktisk kan man enten bruge en testdrevet model, hvor testen flyttes op før udvikling. Eller som minimum kan man se udvikling og test som to parallelle aktiviteter. Ofte er det en blanding af 'før' og 'parallelt', men aldrig som afsluttende testfaser.

3. To primære processer

Det er ikke alle de traditionelle testfaser der er centrale for WCTM, når testen skal drive udviklingen. Det er primært accepttest og unittest. WCTM er afhængig af at få løbet de to i gang. De skal blot ikke ses som testfaser, men som udviklingsprocesser. Når de to testfaser er omlagt fra test til udvikling, er meget på plads. De øvrige kan trækkes igennem ved hjælp af de to. Herunder integrationstest, systemtest (funktionel test), og brugertest.

Det er nemmest at starte med at omlægge unittest. Den kan drive udviklingen når udviklerne skriver testcases til en komponent, før koden skrives. Man kan vælge at skrive færrest mulige testcases ad gangen. Måske kun en enkelt. Efterfulgt af tilsvarende mindst mulig kode. Derefter fortsætte med en ny testcase og tilsvarende kode, indtil en komponent er færdig. Eller man kan vælge at skrive en lidt større mængde testcases, for eksempel til en hel metode ad gangen. For derefter at skrive koden efter den facitliste som testcases udgør. Men det er teknikaliteter om metode. Princippet er at se test som et aktivt middel til at udvikle, i modsætning til et passivt middel til at finde fejl. Så det første skridt i at få test til at blive en del af udvikling, er at involvere udviklerne i testen. For mange udviklere er det i strid med deres ønsker. For at sige det på dansk, med disse udvikleres øje: kodning er sjovt, test er kedeligt. Men det er så langt det nemmeste sted at starte når vi ser på teknikker og værktøjer.

Unittest driver udviklingen indefra og ud mod omverdenen. Ved at tage afsæt i den enkelte komponent og integrere med andre unittestede komponenter.

Accepttesten kan også drive udviklingen. Ved at skrive testscenarier der er forankret i den forretningsmæssige brug og i forretningsmæssig domæneviden. Der er brug for at skrive eksempler på hvert krav til det nye system. For eksempel testscenarier til konkrete forekomster af Usecases og Userstories. Hvis man nøjes med at skrive selve Usecase eller Userstorien, tjener aktiviteten som kravspecifikation. Ved at tage skridtet videre og skrive eksempler samtidig med kravet, tjener de som indhold i accepttesten, og bidrager både til udvikling og test af kravspecifikationen.

Testscenarier kan blive en del af produktbackloggen, allerede før Sprint Planning. Eller de kan skrives under vejs i Sprint Planning. I begge tilfælde kan de drive den efterfølgende analyse og design.

Det kan være de samme scenarier som testere i et traditionelt projekt ville skrive, når systemet er færdigt, og hvor scenarierne bruges til at finde fejl, og til at afgøre af om systemet lever op til forventningerne om funktionalitetens omfang.

Testscenarierne driver udviklingen, ude fra og ind mod systemet. Billedligt talt, fra forretningen, gennem grænsefladen, i retning mod systemets indre.

4. Teknikker og færdigheder

Selvom test er en del af udvikling, er test ikke identisk med udvikling. Opdeling i ækvivalensklasser, identifikation af grænseværdier, Pairwise Testing, klassifikationstræer. Ingen af disse teknikker er magen til de teknikker en udvikler bruger til at skrive kode. Det er teknikker som en tester bruger til at skrive testcases.

Testteknikkerne der skal bruges i WCTM er i vidt omfang de samme som bruges i et traditionelt forløb. Men tidspunktet og formålet er anderledes. En tester kan bruge opdeling i ækvivalensklasser som teknik til at dække alle legale og illegale kombinationer af forretningsdata. I et traditionelt forløb bruges de til at teste om der er fejl i det færdige system. I et WCTM forløb bruges de til at producere en facitliste. Testerne stiller deres viden om testteknikker til rådighed for udviklerne, og de kan arbejde sammen om at udarbejde facitlisten.

Testteknikkerne bruges til at drive udviklingen af de tre centrale artefakter, Produktbackloggen, Sprintbackloggen og Working Product.

For at bruge teknikkerne behøves der færdigheder. Det er ikke nok at annoncere at, fremover skal der skrives testcases før kode. Nogen, i form af mennesker, skal have færdighederne til at gøre det på en måde der gør en forskel.

Testere i et WCTM forløb skal kunne anvende testteknikkerne i et kreativt samarbejde med udviklere. Færdighederne kommer kun ved at arbejde som et team.

Historisk er testere blevet målt på at finde fejl. Nu skal de måles på at forhindre at fejl opstår. Testerne skal kunne deltage i udviklingen fra første dag i et projekt. Det kræver samarbejdskompetencer, kommunikationsevner og kunsten at give konstruktiv feedback.

5. Samarbejde og kommunikation

I WCTM arbejder testere og udviklere arbejder sammen i et team, sammen med de øvrige der er nødvendige for at udføre alle aktiviteterne i sprinten. Samarbejdet må gerne foregå fysisk i et fælles lokale.

Planlægningen af en sprint er i høj grad præget af kommunikation. Produktejeren præsenterer sine krav for teamet, som en væsentlig del af Sprint Planning. Teamet diskuterer hvert krav indbyrdes, som en lige så væsentlig del af Sprint Planning. Udviklere og testere arbejder sammen om at producere systemet.

WCTM er i mindre grad kommunikation via artefakter. Det er en sikker kilde til fejl blot at levere en kravspecifikation til teamet. Som krav fra omverdenen til analytikere. Der i

sin tur yderligere kan øge risikoen for fejl ved at levere nye artefakter til designere, der leverer et design til udviklere, der leverer et system til testerne.

Artefakter er ikke et optimalt middel til kommunikation i et WCTM forløb der bruger test som udvikling. Artefakter kan ikke stå alene. De skal leveres i form af samarbejde og kommunikation.

6. Mindre sluttest, mere automatiseret test

Når test bruges som udvikling falder behovet for test som afsluttende fase. Der bliver mindre brug for test som slutkontrol.

Det kræver mod at fjerne den afsluttende testfase. Men i den udstrækning at testen automatiseres løbende, bliver det nemmere og billigere at gentage testen under vejs som regressionstest.

Når den traditionelle afsluttende test forsvinder, til fordel for test under vejs, styrkes det første punkt i definitionen på WCTM: Ingen testgæld.

Når test er udvikling spiller automatiseret test en lige stor rolle for testere som for udviklere. Testerne hjælper i første omgang udviklerne med at bruge test til at kaste lys over opgaven. Ved løbende at automatisere de tests der fremstilles, får både udviklere og testere et middel til regressionstest. Der er værdifuldt når nye opgaver skal løses, hvor der er brug for at sikre sig at alt eksisterende stadig fungerer.

Automatiseringen gælder alle tre niveauer af automatiseret test, hvor vi ser det blive brugt i dag:

- End-to-end test, i praksis accepttesten, med værktøjer som Selenium og Watir.
- Komponenttest, i praksis en del af kodningen, med xUnit værktøjer.
- Funktionel test, med værktøjer som Fit/Fitness. Enten i forlængelse af kodningen, med integration af komponenter der er testet hver for sig, eller som Black box test, efter en funktionel nedbrydning.

Automatiseret test er en udviklingsteknik, om man vil.

7. Fælles produkter

Når vi bruger test som udvikling, bliver de fysiske produkter der udvikles, altså artefakterne, fælles for udviklere og testere. Et produkt er ikke færdigt før det er både udviklet og testet, og testere og udviklere oplever kun at arbejde med fælles produkter.

I et agilt forløb, som WCTM i reglen vil benytte sig af, er der tre væsentlige artefakter:

- Produktbackloggen, der indeholder udestående krav og ønsker til et produkt, og dermed svarer til en kravspecifikation. Udviklere og testere hjælper produktejeren med at udvikle og vedligeholde den i produktets levetid.
- Sprintbackloggen, der indeholder detaljer og aktiviteter på den del af produktbackloggen der er valgt til den næste sprint (iteration). Sprintbackloggen er resultatet af Sprint Planning. Udviklere og testere opretter en fælles Sprintbacklog til hver sprint.
- Working Product; det færdige produkt. Resultatet af den gennemførte sprint (iteration). Og i høj grad et fælles produkt.

Testernes artefakter i form af testaktiviteter og testcases bliver udviklingsartefakter.

Alle artefakter er fælles for teamet, og kan først erklæres done-done, når de er både udviklet og testet. Betragt testcases til unittest som en del af koden. Lad testcases og kode videreudvikle sig sammen, som en og samme ting. Det samme med Userstories og eksempler på dem. De hører sammen.