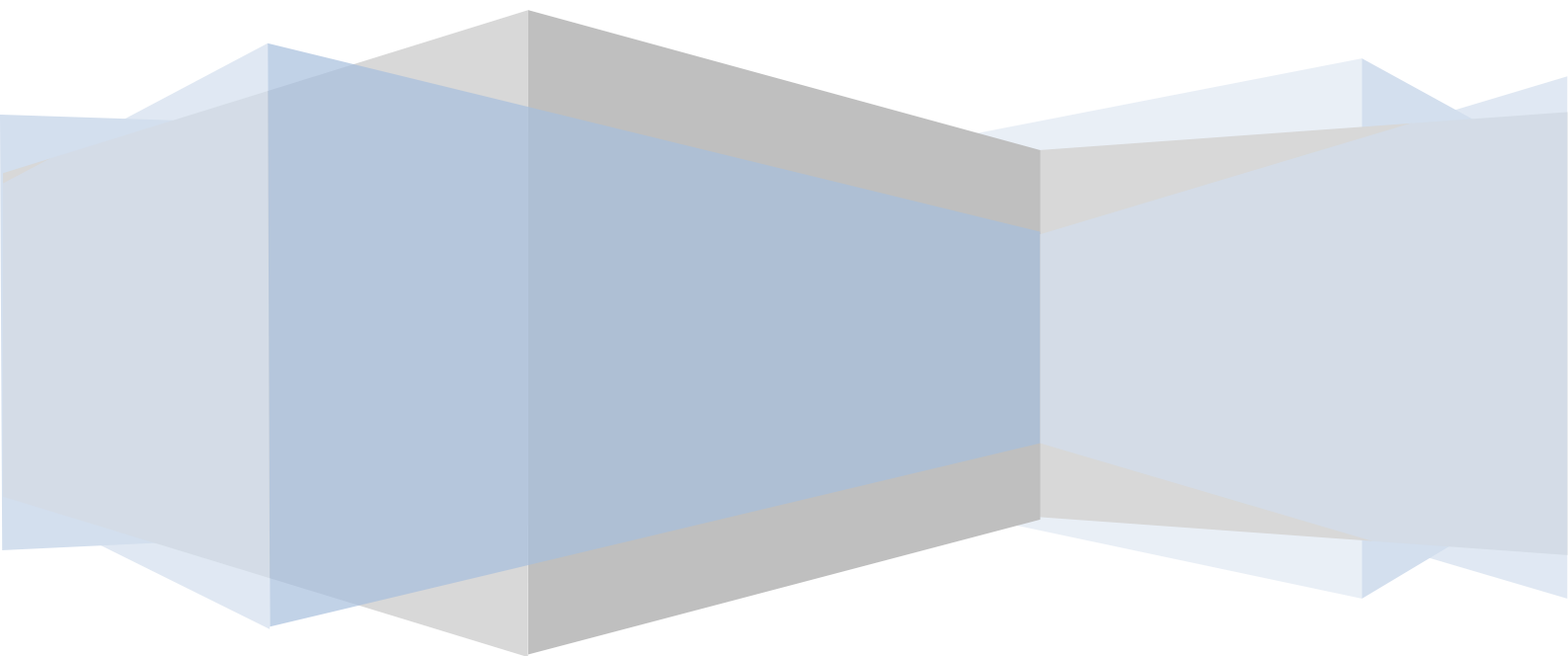


Agile-metoder.dk

World Class Test Management

6. Kvalitet er ikke en styringsparameter

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 6 om WCTM: Kvalitet er ikke en styringsparameter

Indhold

World Class Test Management.....	2
Artikel 6 om WCTM: Kvalitet er ikke en styringsparameter.....	2
1. Kvalitet er ikke en styringsparameter.....	3
2. Ukendt kvalitet	3
3. Delvis test	4
4. Prioritering af testen.	4
5. Kendte fejl og mangler.	4
6. Økonomi	5
7. Kritikalitet.....	5
8. Lean og Agile.....	6
9. Bundlinie.....	6

1. Kvalitet er ikke en styringsparameter

Denne artikel uddyber det femte punkt i definitionen på WorldClassTestManagement.

Citat fra den første artikel:

1. Ikke betragte kvalitet som en styringsparameter der finder sit leje når en deadline er nået. Et udsagn som ”Test så meget I kan, og release når I når deadline”, er ikke styring. Det giver en ukendt kvalitet, og er i den modsatte retning af WCTM. Og kan i øvrigt være ansvarspådragende. Målet for kvaliteten kan sættes lavt. Men det skal stadig beskrives hvad der forstås ved lavt.

Citat slut.

Den citerede tekst er upædagogisk fordi den definerer noget, ved at beskrive hvad det ikke er. Men i dette tilfælde er det på sin plads.

Den klassiske trekant mellem ”Scope”, ”Ressourcer” og ”Tid” burde ikke give mulighed for at overholde en presset deadline, ved at sænke kvaliteten. Og når Test Management er verdensklasse er det udelukket, fordi WCTM ikke omfatter at springe noget test over for at nå en deadline. Det ville betyde at den pågældende del havde en ukendt kvalitet.

WCTM bygger på at kravene til kvalitet er kendte og er en del af kravene. Hvis der er behov for at sænke kvaliteten i et testforløb, betyder det en ny kravspecifikation. Som er andet og mere end at tilføje en tekst om at en del af testen droppes.

2. Ukendt kvalitet

Hvornår er det OK at levere en ukendt kvalitet? Det er det vel, hvis det er en ligegyldig funktionalitet. Eller at det er ligegyldigt om der sker fejl hos brugeren. I så fald var det måske bedre at lade være med at levere den overhovedet.

Det er sjældent at kunder/brugere er ligeglade. Og i en efterfølgende situation hvor man eventuelt diskuterer ansvar, er det for en leverandører af software vigtigt at kunne bevise hvad der er testet, hvornår det blev gjort, og hvad resultatet var. Hvis man simpelt hen dropper noget test her og der, kan det blive vanskeligt. Også selvom det ”kun” er regressionstest.

Kravene kan specificere at det er OK med en ukendt kvalitet, og i det tilfælde kan man selvfølgelig teste så meget eller så lidt man kan nå.

3. Delvis test

Men kan man så forestille sig at software testes til en vis grad? En eller anden grad af overfladisk test. Hvad er forskellen til den ukendte kvalitet? Kan man teste så man sikrer sig mod et vist niveau af fejl? Med en risikoanalyse eventuelt. Det kræver at man kender og kan beskrive alle de uacceptable fejl. At man ikke bare beskriver nogle generelle risici, men opstiller en komplet liste over de fejl der ikke må forekomme. Fordi det ikke er nok at beskrive den minimumsfunktionalitet det skal indeholde. Man er nødt til at vide om man skader kundens forretning, eller brugerens hverdag, på en uacceptabel måde. Der er brug for både positive og negative tests.

Det kræver en fintunet styring at teste præcist så langt at softwaret kan bruges til noget, samtidig med at der er sparet noget test. Medmindre det som med ukendt kvalitet, er en ligegyldig funktionalitet.

4. Prioritering af testen.

Men alligevel taler vi ofte om at prioritere testen. Hvad menes der så med det? Hvad forstås der ved det, og hvad er formålet? En det en prioritering af rækkefølgen af testen? Eller er det en prioritering af testindsatsen, således at nogle områder testes bedre end andre. Hvis det er en rækkefølge, så er det fint nok. Hvis det er omfangen af testen, så er vi tilbage ved den ukendte kvalitet.

Men en prioritering af rækkefølgen, bliver primært bestemt af udviklingsrækkefølgen. Ellers risikerer man at få det vigtigste udviklet sidst. Selvom vi måske gerne vil teste det som det første. En prioritering af testens rækkefølge, der ikke slår igennem på en prioritering af udviklingsrækkefølgen, behøver faktisk ikke at være særlig meget værd.

Risikobaseret skal sørge for, at de dele af et system, der enten indeholder væsentlige risici for kundens forretning, eller som er afgørende for om vi kan nå vores testproces, bliver udviklet først. Og at vi tester dem så tidligt som muligt, enten passivt med reviews og inspektioner fra og med kravspecifikationstidspunktet, eller aktivt, ved at bruge testdrevet udvikling.

5. Kendte fejl og mangler.

Hvad så med at spare på rettelserne. Den er klassisk, og kan umiddelbart accepteres som en styringsparameter. Det er acceptabelt i WCTM at overholde en deadline, ved at lade være med at rette fejl.

Men risikoen ved at bruge systemet, skal stå klar for dem der bruger det.

Vi kan endda undgå ansvarspådragelse, hvis beskrivelsen af konsekvenserne er klare for brugeren af systemet. At eventuelle skader man kan pådrage sig, er tydelige.

Men der er grænser for hvor mange der kan være af dem. Uanset om man så forærer softwaret væk. På et tidspunkt vil man blive til grin.

Men muligheden er der. Det betyder fuld test, men ikke fuld rettelse. Eller for den sags skyld, rettelser overhovedet.

6. Økonomi

Hvad er den økonomiske effekt af at levere med fejl og mangler? På kort sigt kan produktiviteten øges ved at release softwaret, med en vis mængde fejl. Uanset om de er kendte eller ej, og om de beskrives som kendte eller ej. Spørgsmålet er om de skal rettes senere. Og om konsekvenserne af at de har levet et stykke tid, skal rettes op. Så reduceres produktiviteten på lang sigt. Defineret som Throughput på et år; den samlede mængde brugbart system vi kan levere pr. år.

Fokus på kortsigtet produktivitet kan retfærdiggøres af Time-to-Market. Selvom produktiviteten på lang sigt forringes. Men organisationer der leverer ukendt kvalitet, spiller hasard med deres langsigtede produktivitet.

Alt er økonomi, siges det. Og alt bør afgøres på om det er økonomisk forsvarligt. Enten på kort eller lang sigt. Men passer det nu også helt. Personlig prestige, organisatorisk prestige, kortsigtet imagepleje og tro og håb på at en kontrakt kan se ud som om den overholdes, spiller også ind. Det er set.

7. Kritikalitet

Ideen om at vi retter og kører om, er gammeldags. Den stammer fra dengang de enkelte systemer kørte hver for sig. I vore dage er der allerede sket skader 17 andre steder på jorden, fordi systemet kommunikerer med andre systemer, 24/7. Plus at man dengang havde et driftsvindue der gjorde det muligt at fixe databaserne og køre om. Mange systemer arbejdede isolerede, og kunne rettes op lang tid efter at en fejl var gået i daglig drift.

I dag hænger systemerne sammen. Det svarer til at åbne for en ny vejbane på en motorvejsbro. Selvom alt ser godt på vores videreudvikling (vejbanen), så kan der ske fatale fejl andre steder i systemet, hvis ikke alt er afprøvet og godkendt. Man skal også passe på med for mange kendte fejl og mangler. Det hjælper ikke at hænge en seddel op med det som trafikanterne lige skal være opmærksomme på.

8. Lean og Agile

WCTM låner elementer fra såvel agile metoder som Lean principper. I ingen af dem leveres ukendt kvalitet for at nå en deadline.

Lean principper arbejder med Built-in Quality, og åbner ikke rum for utestede dele, eller mangler i det der leveres. Built-in Quality giver kun mening hvis der arbejdes med kendt kvalitet. Lean arbejder ikke med kvalitet som en styringsparameter der kan skrues op eller ned for.

Agile metoder gør heller ikke. Hver lille del kan kun erklæres færdig, når den er testet. Kode der ikke er testet, er ikke kodet færdig. Utestet kode eksisterer ikke som begreb. Det er kun gammeldags programmører der ser kode som noget i sig selv. I en moderne synsvinkel, herunder WorldClassTestManagement, er unittest en del af kodningen. Gammeldags programmører forstår ikke engang udsagnet. Agile metoder arbejder desuden ofte med testdrevet udvikling, der sætter test før udvikling.

9. Bundlinie

Ukendt eller overfladisk kvalitet er ikke en del af WorldClassTestManagement.

Dette er verdensklasse: Testen gennemføres altid i sin helhed, og den bygger på en aftalt kvalitet. Testen er så tidligt på banen at den forudsiger om tidsplanen holder. Konsekvensen kan blive at en del af funktionaliteten fjernes, baseret på en prioritering af udviklingsrækkefølgen. Der kan desuden releases med et mindre antal (velbeskrevne) kendte fejl og mangler.

Selvfølgelig kan man i en presset situation vælge at gøre noget der ikke er WCTM. Men det er vigtigt at vide at det er inkompetent på et helt nyt niveau, at fortsætte med at blive bedre til en inkompetent proces.

Der er brug for en læreproces, i sin mest almindelige betydning. Det er kun verdensklasse at bruge en proces det er værd at blive endnu bedre til. Gennem en kontinuerlig forbedring.

Hvad er muligt at bruge som styringsparameter i WCTM:

Ukendt kvalitet, helt eller delvis	Nej
Overfladisk kvalitet, ved at teste delvis	Nej
Fuldt planlagt test, men kun delvis gennemførelse	Nej
Delvis planlagt test, og delvist gennemført	Nej
Release med kendte fejl og mangler	Ja
Fjerne funktionalitet fra den kommende release	Ja