

# Agile Testing

Introduktion til begreberne  
Af Poul Staal Vinje

Certified Scrum Master  
Certified Scrum Practitioner

[poul.vinje@gmail.com](mailto:poul.vinje@gmail.com)  
[www.agile-metoder.dk](http://www.agile-metoder.dk)

## Indhold

Agile Testing.....	1
Agile Manifesto.....	2
Agile Testing.....	2
Kernen.....	2
TestDrivenDevelopment (TDD).....	2
Continuous Integration.....	3
Accepttest.....	3
Unittest.....	3
Refactoring.....	3
Conversational Test Creation.....	3
Exploratory Testing.....	4
Bug Hunt.....	4
Pair Testing.....	4
Iterative testforløb.....	4
Automatiseret test.....	4
eXtreme Programming (XP).....	4
Scrum.....	4
Ikke-Agile projekter.....	4
Værktøjer.....	5
Links.....	5

## **Agile Manifesto**

Her er den centrale tekst i 'Manifesto for Agile Software Development'

**Individuals and interactions** over processes and tools

**Working software** over comprehensive documentation

**Customer collaboration** over contract negotiation

**Responding to change** over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

## **Agile Testing**

Siden manifestet har der været arbejdet intenst med at udvikle og udbrede Agile metoder. Et af resultaterne er Agile Testing.

I denne artikel skitserer jeg nogle af de væsentlige elementer i Agile Testing.

## **Kernen**

Kernen i Agile Testing er selvfølgelig de værdier der udtrykkes i manifestet. Men de praktiske konsekvenser er:

Testen deles op i en række mindre dele. Hver del af testen er knyttet tæt til udviklingen. Der tilsvarende leverer mindre dele, løbende. Testere arbejder tæt, og konkret, sammen med udviklere og brugere. Udvikling og test er to sider af samme sag.

Testfaserne fra vandfaldsmodellen kendes ikke. Testen er integreret med udviklingen, og gennemføres ikke i længere faser. Selv udviklingsorganisationer der har forladt vandfaldsmodellen til fordel for V-modellen, W-modellen, og andre modeller der søger at integrere test og udvikling, vil opleve at integrationen mellem disse, i Agile Testing, er endnu mere udpræget.

## **TestDrivenDevelopment (TDD)**

Testen følger tre trin.

1. Skriv testen til den næste del af funktionaliteten
2. Skriv kode indtil testen er OK
3. Refactoring af kode og testmateriale

Og bemærk:

Skriv kun kode til en test der fejler.

Skriv kun en enkelt ny testcase ad gangen.

Skriv kun så meget kode at det får den fejlende test til at passere.

Dette er TestFirst princippet. TDD bruges meget ofte i Agile forløb, og er for mange en 'obligatorisk' del. Når princippet udvides til at omfatte hele specifikationen af systemet, bruges en selvstændig proces til dette, AcceptanceTestDrivenDevelopment. Det vil altså sige TestFirst for udviklere der unittester (TDD), og TestFirst for brugere der accepttester (ATDD).

### **Continuous Integration**

Resultaterne integreres løbende. Dagligt, eller oftere.

### **Accepttest**

Sammen med unittest, er dette en hovedform for test i Agile Testing. Den dækker alle testaktiviteter der ser systemet med forretningsmæssige øjne. "Udefra og ind".

### **Unittest**

Sammen med accepttest, er dette en hovedform for test i Agile Testing. Den dækker alle testaktiviteter der ser systemet med udviklernes øjne. "Indefra og ud".

### **Refactoring**

Agilt princip om at starte simpelt, derefter løbende tilføje kompleksitet, men under vejs fastholde kvaliteten i produktet. Både udviklere og testerer benytter sig af Refactoring, til henholdsvis kode og testmateriale.

### **Conversational Test Creation**

Agilt princip om at skrive tests i et samspil mellem mennesker. Det kan være brugere, testere og udviklere.

## **Exploratory Testing**

Udforskende test. Kan bruges til Agile såvel som ikke-Agile projekter. Bygger på at formulere tests på grundlag af resultatet af den foregående test, snarere end at planlægge det samlede testforløb, før testen starter. Exploratory Testing er beskrevet i min bog om softwaretest.

## **Bug Hunt**

Den mest dynamiske form for udforskende test. Med dynamisk menes selve formen på udførelsen af den.

## **Pair Testing**

Agilt princip om at arbejde i par, Ikke i faste par, men i skiftende sammensætninger. MTB (Minimum Two Brains).

## **Iterative testforløb**

Iterative projekter bruger ofte Agile Testing. Det kan kræve regressionstest på tværs af iterationer.

## **Automatiseret test**

Unittest er altid automatiseret, og accepttest er det i det mest mulige omfang. Automatiseret test er velkendt i mange sammenhænge, men i et Agilt forløb hvor ændringer og udvidelser foretages løbende, er automatiseret unittest nødvendigt. Uundværligt.

## **eXtreme Programming (XP)**

Agile Testing er central for XP, og mange af erfaringerne med Agile Testing stammer fra XP projekter. Der er en introduktion til XP i begge mine bøger.

## **Scrum**

Scrum er en god måde at styre XP projekter på. Agile Testing er ikke defineret i Scrum, men passer som hånd i handske. Scrum er beskrevet i min bog om projektledelse af systemudvikling, og i en selvstændig artikel.

## **Ikke-Agile projekter**

Agile Testing kan bruges i enhver type af udviklingsforløb. Uanset om der i øvrigt bruges Agile metoder.

## Værktøjer

Værktøjer er væsentlige på flere områder. xUnit Framework til unit-test. Accepttesten kan bruge regneark eller wikier, med en efterfølgende driver til at gennemføre testen. Se Fit/Fitness, Watir og Selenium.

## Links

Der mangler ikke information om mulighederne i Agile metoder. Her er nogle få vigtige links, der kan vise videre.

[www.agilealliance.org](http://www.agilealliance.org)

[www.xprogramming.com](http://www.xprogramming.com)

[www.testdriven.com](http://www.testdriven.com)