COMPUTABLE

Vijf jaar Agile: hosanna of drama? (1)

29-10-2014 10:00 | Door Leo van der Aalst | Lees meer artikelen over: Agile, Scrum

Agile is het antwoord op alles. Zo wordt alom gesproken. Hoe dan ook zou de traditionele softwareontwikkelaanpak niet meer geschikt zijn voor deze tijd. Is dat terecht? Ik heb onderzoek gedaan naar driehonderd agile projecten in de afgelopen vijf jaar. Succesvolle projecten, en helaas ook faliekante mislukkingen. De belangrijkste misvattingen en valkuilen komen aan de orde in dit verhaal. Uiteraard met het doel deze de wereld uit te helpen. In deel 2 van mijn verhaal zet ik de voordelen en succesfactoren op een rij. Drama kan worden voorkomen als we onderstaande misvattingen en valkuilen serieus nemen.

Misvattingen

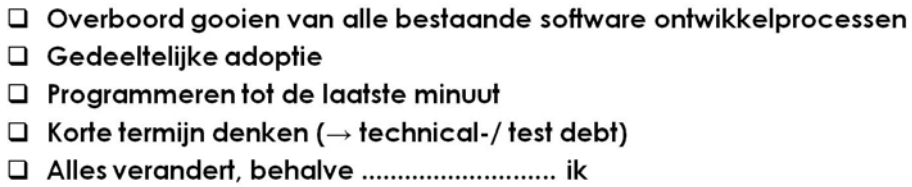
- Agile is het antwoord op alles
- In Agile ontbreken processen
- In Agile wordt niet gepland
- In Agile ontbreekt discipline
- In Agile wordt niet gedocumenteerd

Helaas, in de praktijk blijkt Agile niet het antwoord op alles te zijn. Vooral in hiërarchische omgevingen met vastomschreven procedures en processen. Of waar certificering een belangrijke rol speelt zoals in de luchtvaart- of beveiligingsindustrie. Dan is Agile minder geschikt dan een traditionele aanpak. In bijna de helft van de Agile-projecten in deze strak gestuurde organisaties is men niet tevreden over het resultaat.

In Agile-omgevingen ontbreken processen, wordt er niet gepland en is discipline ver te zoeken. Dit wordt vaak geroepen. Maar het zijn allemaal misverstanden. In een Agile-aanpak bestaan wel degelijk processen, net als in een traditionele aanpak. Dit zijn weliswaar lichtgewicht processen. Het Agile-team past processen naar eigen behoefte aan. Ook is er wel degelijk sprake van discipline. Ga maar na, hoe kun je in een iteratie van bijvoorbeeld twee weken werkende software opleveren zonder planning en discipline? Inderdaad. Niet dus. Release- en iteratieplanningen zijn een vast onderdeel van Agile. Daarbij worden scrum- of kanban-borden, burndown charts en daily scrum gebruikt om werkzaamheden transparant te maken en af te stemmen. Dit alles ondersteunt de planning om te komen tot het gewenste resultaat. Zo vraagt deze aanpak juist om een ijzeren discipline van Agile-teamleden.

Een hardnekkig vijfde misverstand is dat er in Agile niet wordt gedocumenteerd. Dit misverstand is waarschijnlijk ontstaan omdat het Agile-team alleen documenteert wat nodig is om als Agile-team het werk te kunnen doen of waar stakeholders een belang bij hebben. In het ene geval is dat minimaal, omdat de teamleden precies weten wat ze moeten doen en wat ze van elkaar kunnen verwachten. In het andere geval kan dat meer zijn. Bijvoorbeeld als er eisen zijn gesteld aan onderhoudbaarheid of overdraagbaarheid van producten. Of omdat er sprake is van wet- en regelgeving-, of certificeringseisen die meer documentatie vragen dan voor het team nodig was om het product te maken. Er wordt dus wel degelijk gedocumenteerd, maar niet meer dan nodig.

Valkuilen



Je zult maar als softwareontwikkelaar al jaren met een bewezen goedwerkende aanpak werken. En dan beslist het management op een dag dat Agile nu de aanpak wordt. En dat 'by the way' die goedwerkende aanpak de prullenbak in kan. Het overboord gooien van bestaande softwareontwikkelprocessen is een vaak voorkomende valkuil. Want wat staat er allemaal in het Agile-manifest of in de Scrum-aanpak over hoe je software moet ontwikkelen? Over hoe je requirements, functionele specificaties als user stories of use cases, technische specificaties, datamodellering, systeem- en software architectuur, software, tests, moet opstellen en maken? Niets toch? Dus dan lijkt het verstandig het kind niet met het badwater weg te gooien. Blijf daarom goedwerkende onderdelen van de software-ontwikkel- en testaanpak gebruiken, of aan te passen aan de Agile-situatie.

Vrijwel alle methoden zijn goed doordacht en werken prima als ze worden gebruikt, zoals ze zijn bedoeld. Denk bijvoorbeeld aan SDM, JSD, RUP, of PRINCE2. Bij de implementatie van een Agile-aanpak zoals Scrum is veelal een gedeeltelijke adoptie te zien. Even zo vaak blijkt dat een bewuste als onbewuste keuze te zijn. Zelfs een bewuste keuze van 'cherry picking' leidt in de meeste gevallen tot minder succes dan een volledige adoptie. Het bekende gezegde 'hinken op twee gedachten' doet hier opgeld. Dat is een notoire valkuil.

Programmeren tot de laatste minuut. Wie komt dat niet bekend voor? Dit is een diepe valkuil, want dit betekent per definitie dat er geen werkende software aan het eind van een iteratie wordt opgeleverd. In iets afwijkende bewoordingen van elkaar, zeggen alle Agile-aanpakken dat een product pas gereed ('done') is als alle stakeholders akkoord gaan met het product. Het korte termijn denken betekent meer risico op het gebied van test- en technical debt. Door wel snel 'iets' - omdat bijvoorbeeld de 'product owner' dat wil - op een 'quick and dirty' wijze aan het eind van de iteratie op te leveren, kunnen code-, architectuur- en testgerelateerde onvolkomenheden in volgende iteraties aan het licht komen. Dat leidt tot grote, kostbare of tijdrovende herstelactiviteiten.

En ineens zit je niet meer in een traditioneel ontwikkelproject, maar in een Scrum-team. Nu maar hopen dat de andere teamleden hiermee kunnen omgaan, denk je. Dit is een heel typische valkuil, alles verandert, behalve … ik. Een menselijke eigenschap is dat mensen al gauw van zichzelf vinden dat ze iets wel kunnen, terwijl ze denken dat andere mensen daar meer moeite mee hebben. In Scrum is het van belang dat je zelf ook mee verandert. Dat gaat niet vanzelf. Daar moet je echt iets aan doen.

Leo van der Aalst, senior testing consultant bij Sogeti Sogeti en lector software quality & testing aan de Fontys Hogeschool