

Praktijkervaringen met Agile in een watervalomgeving

Agile tegen de stroom in

De laatste tijd is er vrij veel aandacht voor Agile- en Lean-werkwijzen. Maar hoe zit het met de toepasbaarheid in de praktijk, als je niet alle randvoorwaarden zelf mag kiezen? In dit artikel delen Adriaan van den Brand (softwarearchitect) en John van Spaandonk (softwareprojectmanager/adviseur) hun niet alledaagse ervaringen met Agile in een groot project dat is gebaseerd op het bekende watervalmodel. Roeien tegen de stroom in?

**Adriaan van den Brand
John van Spaandonk**

Als u ooit de Niagara-watervallen van dichtbij hebt gezien, moet u onder de indruk zijn geweest van de imposante kracht. In de softwareproductontwikkeling kennen we een andere waterval die zeker niet minder krachtig is. Veel kwaliteitssystemen zijn gebaseerd op het V-model voor gefaseerde productontwikkeling. In de basisversie van deze aanpak wordt software ontwikkeld in opeenvolgende, strikt van elkaar gescheiden fases. Is er binnen een project met zo'n sterke watervalaanpak ruimte voor de Agile-manier van werken? Wij vinden van wel, maar uit ervaring kunnen we ook melden dat dit niet vanzelf gaat. Hopelijk trekken de beschreven ervaringen anderen over de streep om zich ook eens aan Agile te wagen.

We werkten in een innovatief multidisciplinair project van ongeveer honderd personen met een looptijd van circa twee jaar. Het kwaliteitssysteem volgde een strikt watervalmodel. Voor een vlotte start werd de bemensing al vroeg in de conceptfase geregeld. Tijdens die fase groeide het project snel in omvang, doordat elders vrijgekomen mensen vervroegd aan het project werden toegekend. Uiteindelijk was het project verdeeld over drie landen, met India als lagelonencomponent. Terwijl de systeemspecificaties nog fluctueerden op scope, performance en prijs, waren er al veel ontwikkelaars beschikbaar. Hun inputdocumentatie moest echter nog langs de linkerpoot van het V-model naar beneden druppelen.

John had in het project de rol van software-integratiemanager en Adriaan was verantwoordelijk voor de softwarearchitectuur.

John wilde de mensen in zijn integratieteam snel en zinvol aan het werk krijgen en ook Adriaan zag geen heil in wachten: vele vragen zouden beantwoord moeten worden voordat de softwarearchitectuur kon worden opgeleverd. Hoe moesten we dat doen?

We vinden het van vitaal belang dat alle softwaregroepen (inclusief toeleveranciers) regelmatig leveringen doen die minstens maandelijks worden geïntegreerd en getest. Zo weet je tijdens het project waar je staat en voorkom je een langdurige periode van testen en problemen oplossen aan het einde. Een strakke intakeprocedure laat aangeleverde software alleen door voor integratie als de kwaliteit goed is. Bij deze intake en het eigenlijke integreren voert het software-integratieteam grote hoeveelheden tests uit. Om dit mogelijk te maken, is het zeer belangrijk dat we het eigenlijke testen zo veel mogelijk automatiseren. De ontwikkelgroepen leverden de testcases samen met de software op, wat ervoor zorgt dat we bijvoorbeeld unittests kunnen hergebruiken op systeemniveau. Dit vereist wel een stuk uniformiteit, niet alleen in werkwijze, maar ook in testen.

Voordat we testcases konden schrijven, hadden we een testframework nodig. Dit zou in India worden gemaakt. De mensen daar hadden echter geen specificaties van wat er moest komen en zaten te ver van het vuur om zelf een voorstel te kunnen maken. Daarom zette John een proces op van iteratief verzamelen van wensen, specificeren en bespreken in wekelijkse teleconferenties. De schrijver van de testspecificatie in Bangalore was de *product owner* van het

testframework. De testers uit de diverse softwaregroepen leverden als toekomstige gebruikers hun wensen en ideeën. Dit is al een vorm van werken volgens Agile: regelmatige leveringen, veel klantcontact en directe feedback die leidt tot verbeteringen.

Na een maand bevatte de specificatie ons eerste beeld van het testframework. Dit duurde relatief lang omdat we over verschillende vestigingen verspreid zaten. Bovendien was veel van de materie voor ons nog nieuw, waardoor veel discussie en uitzoekwerk nodig was. Voor de eigenlijke implementatie in Python zijn we begonnen vanuit het integratieteam, omdat daar de meeste Python-kennis aanwezig was. We hebben hier Scrum toegepast.

Na drie sprints (iteraties) van twee weken stond de basis voor het testframework. Deze hebben we overgedragen aan het team in Bangalore, die het in enkele maandelijkse releases opwerkte tot de eerste bètaversie. Specificatie en testframework evolueerden vanaf de overdracht gelijk op. Met de bètaversie konden we de uitgebreide handmatige regressietest van een vorig gelijkaardig product overzetten naar automatische tests in het nieuwe testframework.

Dit is een interessant voorbeeld van het gebruik van de Agile-principes. Ondanks ons nog deels ongedefinieerde project konden we in onze eigen organisatorische context toch gecontroleerd voortgang maken. De truc is om het enthousiasme van de mensen te bundelen door ze als (virtueel) team te laten samenwerken. Een ander belangrijk punt is om zo snel mogelijk tot een eerste bruikbare versie te komen, zodat iedereen snel feedback kan geven. En met de leveringen van het testframework konden we onze intakeprocedure voor software al vroeg testen.

Webcam

Van architectuur en requirements heb je volgens Agile-goeroes *just as much as needed, just in time* nodig. Dat botste echter met de organisatie: die wilde graag vooraf in detail weten wat er hoe gemaakt zou worden en hoeveel dat zou kosten. Architectuur en requirementsdocumenten moesten we dus gewoon opleveren. Onze vrijheidsgraden



waren de indeling en het prioriteren van delen van de documenten.

We formeerden een Scrum-team uit de groep softwarearchitecten. Adriaan had als lead-softwarearchitect de rol van product owner en vormde de brug naar het team dat de systeemspecificaties uitwerkte. John fungeerde als Scrummaster. Veel aandacht ging uit naar communicatie en coördinatie. Enkele gezamenlijke workshops hebben wonderen gedaan voor de teamband. We wilden hiermee invulling geven aan de Agile-principes colocatie en zelfsturendheid van het team.

Drie maal per week was er een voortgangsteleconferentie (als alternatief voor de *daily stand-up*) en we gebruikten het gratis tool XPlanner voor het volgen van de *deliverables* en taken. XPlanner voegde veel waarde toe omdat het ons in staat stelde per sprint exact te definiëren wat er gemaakt moest worden. Tegelijkertijd was al onze informatie zichtbaar op iedere locatie. Ieder lid van het team kon zijn eigen taken beheeren en zijn uren bijhouden. Deze continue blik op het gezamenlijke werk heeft in belangrijke mate bijgedragen aan het succes van dit team. Dit is bij multisiteontwikkeling net zozeer een must als wanneer een team in één ruimte zit. We wilden nog een stapje verder met het verlagen van de communicatiedrempels, bijvoorbeeld door het gebruik van videoconferencing of webcams. Vanwege bandbreedte en beveiliging in de organisatie was dit echter niet haalbaar.

Binnen de softwarearchitectuurgroep bleken de standaard documenttemplates een incrementele werkwijze niet erg te ondersteunen. De gevraagde architectuurview per hoofdstuk was moeilijk te verdelen in kleine stapjes. Beter was het om de verdeling op basis van aspecten te maken, bijvoorbeeld de communicatie tussen componenten of een model van de CPU-belasting. Zo'n aspect omvat veelal meerdere architectuurviews en kan in enkele pagina's volledig worden beschreven en gereviewd. Bijkomend voordeel is dat de reviewlast goed wordt verdeeld over de tijd en er meer to the point feedback wordt gegeven tijdens de reviews.

Voor het werk aan requirements was de database Doors voorhanden. Hierin heeft iedere specificatie een eigen status en is het mogelijk een hiërarchie te bouwen van steeds hoger detailniveau. Het 'document' hoeft dus niet af te zijn om aan te geven dat een specificatie is goedgekeurd. Ideaal om incrementeel te werken in Scrum sprints. Doordat het kwaliteitssysteem requirements toch als een aantal documenten zag, konden we dit maar deels uitbuiten. Er ging veel tijd zitten in zaken als het wachten op een document en discussies over de juiste verdeling van requirements over documenten.

Nukken

Hoe kijken we terug op onze ervaringen? Ten eerste botst Scrum in een klassiek project regelmatig met de normale

werkwijze. Het in teamverband werken in *time-boxed* sprints haalt sommige mensen uit hun *comfort zone*. Zo waren veel auteurs van specificaties gewend aan een aantal weken 'rust' vanuit het project wanneer ze aan een document werkten. Met Scrum wordt binnen het team van auteurs meer geschakeld tussen onderwerpen op basis van belang en afhankelijkheden en werkt iedereen aan korte taken. Er is ook meer druk op levertijd. En het werk moet na iedere sprint af zijn, zodat anderen het kunnen reviewen. Er worden dus telkens aspecten aan het document toegevoegd en gereviewd. Voordeel: een snel groeiende set *approved* documentatie.

Verder hebben we met virtuele teams gewerkt waarbij mensen verdeeld waren over meerdere landen. Ondanks gebruik van tools en telecommunicatiemiddelen heeft dit een veel lager rendement gehad dan een team in één ruimte. We denken wel dat we het beste uit de situatie hebben gehaald.

Ten slotte valt natuurlijk op dat we Agile hebben toegepast op architectuur en requirements. Puristen zullen benadrukken dat je dat niet moet doen, maar zo snel mogelijk moet beginnen met het eigenlijke coderen en testen. Daar zijn wij het mee eens, met dien verstande dat architectuur en requirements nog steeds belangrijk zijn als middel, maar niet als doel.

Als we terugkijken naar onze aanpak, zijn we redelijk tevreden. Scrum heeft ons geholpen de beschikbare capaciteit zo goed mogelijk te gebruiken. In dezelfde tijd hebben we meer en met hogere kwaliteit geproduceerd. Het heeft ons echter niet immuun gemaakt voor de nukken van de organisatie. Helaas kende het project geen happy end: het werd geannuleerd na meer dan een jaar in de eerste fase van de waterval te hebben doorgebracht. Markten, klantwensen en budgetten bleken sterk te veranderen, zaken die het watervalmodel als constanten beschouwt. Toch adviseren we andere mensen de kracht van Agile te ervaren en dagen we organisaties uit om de randvoorwaarden te creëren om Scrum maximaal tot zijn recht te laten komen.

Adriaan van den Brand werkt bij Uphantis High Tech Solutions als systeemarchitect op detachingsbasis. Daarbij geeft hij advies op het gebied van productontwikkeling. John van Spaandonk is werkzaam als zelfstandig projectmanager. Ook geeft hij advies en trainingen op het gebied van (de invoering van) Agile/Scrum en Prince2.

Redactie Alexander Pil