INLEIDING

HET DOEL VAN AGILE WERKEN

Agile werken werkt, en daarom zijn er ook zo veel mensen die het hebben omarmd. Helaas hebben steeds meer mensen moeite met agile werken. Het is in de praktijk niet zo effectief meer, omdat er veel bureaucratische methoden voor zijn ontwikkeld die slagvaardig, daadkrachtig en wendbaar werken in de weg zitten. In dit boek probeer ik uit te leggen hoe agile eigenlijk bedoeld is, en hoe je het kunt inzetten zonder het gedoe.

In dit boek lees je veel over agile werken tijdens softwareontwikkeling. Daarvoor zijn twee redenen. De eerste reden is dat software nu eenmaal de wereld is waarin ik carrière heb gemaakt. De tweede en belangrijkste reden is dat agile werken in hoge mate is ontwikkeld in de computerwereld. Daarin wordt nu eenmaal veel met projecten gewerkt, en mensen in deze wereld waren en zijn continu op zoek naar manieren om die projecten beter en sneller uit te voeren. Agile-zoals-we-het-nu-kennen heeft dan ook veel wortels

in de computerwereld. De problemen en oplossingen in deze wereld zijn natuurlijk universeel, en ik nodig je dan ook uit om een ander product of dienst in gedachten te nemen als ik schrijf over IT-projecten.

De duivelsdriehoek

Agile werken heeft drie grote voordelen:

- het levert een product op met hoge kwaliteit;
- de kosten op de lange termijn zijn laag;
- een project kan met grote snelheid gerealiseerd worden.

Volgens sommigen kun je deze drie doelen niet tegelijk bereiken. Daarom wordt deze 'project management triangle' ook wel de 'duivelsdriehoek' genoemd, en daar zit wel wat in. Een hoge kwaliteit en een grote snelheid leiden vanzelf tot hoge kosten. Lage kosten en een grote snelheid leiden tot een lage kwaliteit. Met agile werkwijzen en methoden kun je de drie doelen toch bereiken.
Een hoge kwaliteit kun je namelijk bereiken door deze in het werkproces in te bedden. Als je bijvoorbeeld een auto bouwt, zorg je ervoor dat elk onderdeel van hoge kwaliteit is en dat elke handeling bij de montage perfect wordt uitgevoerd. Dat lukt als je personeel voldoende opgeleid en goed gemotiveerd is.
Lage kosten op langere termijn bereik je bijvoorbeeld door de onderhoudbaarheid van je systemen of producten groot

te maken. Een auto die nooit stukgaat en slechts zelden een beurt nodig heeft, is goedkoop in onderhoud, zeker als de auto van hoge kwaliteit is en lage afschrijving heeft. Lage kosten bereik je ook door met goede mensen te werken. In de softwareontwikkeling geldt dat een goede programmeur twintig keer meer produceert dan de middelmatige programmeur, maar niet twintig keer zoveel kost. Het inhuren van goede mensen loont.

Een snelle realisatie kun je bereiken door het hele project in korte tijd uit te voeren, en je kunt het ook bereiken door onderdelen van je product tussentijds op te leveren. Een auto kun je niet in stukjes opleveren, maar een trein kun je wel wagon voor wagon opleveren – als je maar begint met de locomotief. Software kun je stukje bij beetje opleveren, je kunt elke feature die je maakt publiceren alvorens met de volgende feature te beginnen. Snelle oplevering van stukjes van je product kan echter leiden tot een latere oplevering van het geheel: als een nieuw stukje niet past in de oude stukjes moet je oude stukjes opnieuw maken. Je moet dus van tevoren bedenken wat je beter vindt: snelle oplevering van kleine onderdelen of snelle oplevering van het geheel.

Agile werken heeft nog een voordeel: het kan het welzijn van de medewerkers bevorderen. Inherent aan agile methoden is het verminderen van bureaucratie, het vereenvoudigen van beslisstructuren, het verhogen van kennisniveaus van mensen, het meer inzichtelijk maken van processen, mensen meer eigen verantwoordelijkheden geven, het ontwikkelproces in kleine stukjes opdelen en

regelmatig deelproducten opleveren. Medewerkers vinden dit prettig, waardoor ze hun werk met meer plezier doen.
Een belangrijke valkuil van agile methoden is de volgende. Een methode impliceert dat er voorschriften zijn en zodra er voorschriften zijn is het voor mensen verleidelijk om die voorschriften dan maar letterlijk te volgen in plaats van een eigen plan te trekken. Kijk naar de daily standup in scrum: meestal is het een ritueel dat geen praktisch doel dient omdat het praktische doel, namelijk kennis-, idee- en informatie-uitwisseling tussen teamleden, al 's ochtends bij de koffiemachine of de dag ervoor tijdens het werk heeft plaatsgevonden. Als je teams wilt verleiden tot echt slagvaardig en wendbaar werken dan moet je je teams de juiste manier van denken en de juiste houding en instelling bijbrengen, en ze dus geen methode in de maag splitsen.

In dit boek zet ik eerst mijn bezwaren uiteen tegen bestaande agile methoden als scrum en daarna schets ik de manier van denken die mijns inziens leidt tot een werkelijk daadkrachtige, slagvaardige en wendbare uitvoering van projecten. Tussendoor geef ik korte anekdotes en praktijkverhalen om mijn betoog te illustreren.

INTERMEZZO 1

MIJN EERSTE BAAN

Ik ging na mijn studie werken bij een kleine bank. Ik werd naar een training gestuurd om de nieuwste methoden en technieken voor softwareontwerp en -ontwikkelingen te leren. Het was de pre-agile tijd: softwareontwikkeling gebeurde in fasen. Eerst de analyse, dan het ontwerp, dan het technisch ontwerp, dan het programmeren en dan het testen. Zo'n fase kon best maanden of jaren duren. Nu noemen we zo'n model de 'watervalmethode' en vinden we het onproductief, maar ooit was het gestructureerd ontwikkelen van software in goed gedefinieerde fasen met goed gedefinieerde methoden revolutionair. Het was immers beter dan de 'doe-maar-wat-aanpak' die men voor die tijd hanteerde.

Ik kwam in een team met ervaren programmeurs te werken en aanvankelijk vond ik het leuk omdat het allemaal nieuw was. We vergaderden niet of nauwelijks, dat was wel mooi. Als er iets moest gebeuren, zoals het maken van een stuk

nieuwe software, of als er aanpassingen gedaan moesten worden in bestaande systemen, dan had de teamleider dat overlegd met het hoofd IT en vroeg dan een van ons om het te gaan doen. We zaten er allemaal bij als dat besproken werd en het hele team bleef dus op de hoogte van wat we allemaal deden. Soms was er een grotere klus en dan gingen daar twee of drie mensen aan werken, en een enkele keer was het dan nodig om te vergaderen in een van de schaarse vergaderkamers die we hadden. Er werd hard gewerkt en er werd veel geproduceerd.

Op een dag werd ik ontboden bij het hoofd IT. Er was een groot project gestart om alle systemen van de bank te upgraden en te vernieuwen, en hij vroeg of ik daarbinnen een nieuw project wilde doen. Er moest een nieuwe, centrale transactiecomputer komen voor de verwerking van financiële transacties in de hele bank. Nu doe je dat via internet, maar toen was dat er nog niet. Die transactiecomputer zou een centrale spil in de IT van de bank worden en mocht dus nooit down gaan. Er was besloten een kostbare computer te kopen die zo was gebouwd dat hij het altijd zou doen. Ik werkte net een jaar in mijn eerste baan en nu was ik ineens projectleider van het meest innovatieve project bij de bank.

De keuze voor de hardware was al gemaakt, maar welke software we nodig hadden, moesten we nog bekijken. We gingen dus praten met de leverancier van de computers, we praatten met diverse softwareleveranciers en -consultants en we gingen op bezoek bij een andere bank waar ze met eenzelfde project bezig waren. Uiteindelijk zaten we

met een paar mensen bij elkaar om besluiten te nemen. Het hoofd IT vond de softwareleveranciers erg arrogant en wat ze aanboden was veel te duur. Ik dacht dat we best zelf konden bouwen wat we nodig hadden, gebaseerd op de standaardsoftware die de hardwareleverancier aanbood. En dus gingen we dat doen.

Ik mocht twee programmeurs uitkiezen voor mijn team en we gingen aan de slag. Ik had weinig woorden nodig om het team uit te leggen wat er moest gebeuren: 'We bouwen een systeem dat alle transacties die uit de kantoren komen verwerkt en dan doorstuurt naar de centrale mainframes en het SWIFT-netwerk.'

Ineke ging aan de slag met de nieuwe softwareomgeving, Henk ontfermde zich over de communicatie met andere systemen en ik begon met een architectuur. In de IT-wereld is een 'architectuur' een soort globaal ontwerp van de te gebruiken techniek. In een architectuurontwerp staan dingen als 'we gebruiken microservices' of 'we gebruiken no-sql-databases', er kan ook in staan wat het totaalplaatje is van de gebruikte IT-systemen binnen een project of binnen een organisatie. We praatten veel heen en weer, keken met elkaar mee en het duurde niet heel lang of we hadden de eerste testjes draaien en de eerste versie van een architectuur geïmplementeerd. Sinds de eerste dag van ons project hadden we niet meer vergaderd. Regelmatig kwam Dirk binnenstormen, hij was de projectleider van de software die op de minicomputers in de kantoren draaide waarmee wij moesten communiceren. Dan kaartte hij een issue aan en ter plekke bedachten we een oplossing en gingen we

weer verder. Het hoofd IT kwam regelmatig buurten, ons project was nu eenmaal superzichtbaar in de organisatie, hij kon zich niet veroorloven dat het niet goed ging. Als ik extra budget nodig had voor een onvoorzien softwarepakket, dan was dat geen probleem. Vlakbij zaten de system engineers en de operators, en daar hielden we ook goed contact mee. Regelmatig zat mijn collega Henk bij hen om dingen met hen uit te proberen, te testen en hun te leren hoe ze de nieuwe software en hardware het beste konden beheren. Ook zag ik regelmatig de router-mensen. Via hun routers communiceerden wij met de mainframes en de SWIFT-systemen.

We wisten niet wat agile was, dus hadden we ook geen sprints. Wel praatte het hoofd IT elke maandagochtend met alle projectleiders in een meeting van hooguit een half-uur. Het gesprek was simpel: 'Wat heb je afgelopen week gedaan, wat ga je deze week doen en lig je op schema?' Zolang je er met je team maar voor zorgde iets voor te lopen op je schema, had je nooit problemen. Soms achterlopen was ook niet erg. Altijd achterlopen, dat werd niet op prijs gesteld.

Eigenlijk was dit de perfecte agile aanpak. Er waren deadlines, natuurlijk. Klanten en gebruikers moesten van tevoren weten wanneer ze de grote switch moesten maken. En hoewel de term 'minimum viable product' toen nog niet in zwang was, was dat wel wat we maakten. We maakten het minimale product waarmee we klanten en gebruikers (en het hoofd IT natuurlijk) tevreden konden stellen.

Een daily standup of daily scrum bestond niet. Zo'n overleg

ontstond vanzelf als we 's ochtends binnenkwamen en de computers aanzetten. Henk haalde koffie voor ons en voor de andere collega's die 's ochtends kwamen buurten.

Er was net een nieuwe laserprinter aangeschaft – denk dan niet aan de desktopprinters die je kent, die waren er toen nog niet. Denk aan een gigantisch apparaat waarmee honderden bankafschriften per minuut konden worden geprint. Wij hadden op slinkse wijze onze systemen verbonden met die printer, en zo konden we mooie documentatie printen. Daar maakten we vrienden mee: steeds meer ontwikkelaars in andere teams leverden prachtig geprinte documentatie op. We waren een beetje een agile organisatie avant la lettre. Wij maken iets nuttigs, jullie maken iets nuttigs en samen werken we efficiënter en plezieriger.

We waren op tijd klaar met de software. Al snel hadden we iets wat werkte, dat haalde de druk van het project. Daarna hebben we alle features toegevoegd die nodig of wenselijk waren. Omdat we met het team van Dirk nauw samenwerkten, met de operators, de system engineers en met de router-mensen, werd alles bij elkaar een goed werkend efficiënt systeem en waren ook de andere teams ruim op tijd klaar. Achteraf gezien was dit mijn eerste agile project, alleen wist ik dat toen nog niet.

HOOFDSTUK 1

DE GESCHIEDENIS VAN AGILE WERKEN

Ooit maakten we producten voor een markt die maar langzaam veranderde. Je ontwierp een schoen en dan wist je dat schoenen de komende eeuwen wel ongeveer hetzelfde zouden blijven. Je ontwierp een nieuw model auto en dat model ging moeiteloos twintig jaar mee. Dat is niet meer zo. Mensen willen vandaag elektrische auto's, en morgen graag met zonnecellen op het dak, maar een autofabriek kan niet volgend jaar al in plaats van benzineauto's plotseling elektrische auto's leveren, laat staan met een dak vol zonnecellen. De klassieke autofabriek is niet agile: het is een moloch die slechts langzaam tot iets nieuws te bewegen is. Veranderingen gaan überhaupt steeds sneller. Tienduizenden jaren geleden leefden we in een nomadische samenleving. We plukten ons eten, we jaagden op ons eten, en dan trokken we verder. Tot we de landbouw uitvonden. Toen konden we ons in een dorpje vestigen en daar ons eten

telen in plaats van het elders te vangen. Die verandering voltrok zich over een periode van vele duizenden jaren, in alle delen van de wereld in verschillende perioden. Het was een ingrijpende verandering: de samenleving werd fundamenteel anders. De volgende verandering voltrok zich daarna: in plaats van dat we allemaal boeren waren die vee hielden, graan kweekten en onze eigen klompen maakten, gingen we ons specialiseren. Er ontstonden ambachten; klompenmakers die de hele dag klompen maakten konden dat beter en efficiënter dan de boeren die 's avonds ook wel eens een klomp sneden. Deze overgang vond plaats in de middeleeuwen, in minder dan duizend jaar. De volgende overgang duurde nog korter. Na de uitvinding van de stoommachine werd het werk gemechaniseerd. Spierkracht werd vervangen door stoomkracht, in een periode van honderd jaar.

De volgende revolutie vond plaats in de eerste helft van de vorige eeuw. We gingen ons makkelijker verplaatsen met auto's en vliegtuigen, we gingen over lange afstanden communiceren met radio en telefoon, en zelfs met televisie. Televisie veroverde de wereld in minder dan tien jaar, tien keer sneller dan de stoommachine dat deed. Tussen 1950 en 1960 werd tv van iets heel bijzonders tot iets wat in de meeste huiskamers aanwezig was. Daarna kwamen internet en mobiele telefoons. Internet is uitgevonden in de jaren zestig, maar werd pas maatschappelijk relevant begin jaren negentig. En toen ging het snel: in 1994 had bijna niemand internet, twee jaar later was al duidelijk dat het gemeengoed ging worden. Mobiele telefoons maakten

AGILE
ZOALS HET BEDOELD IS

Ben je enthousiast over agile werken, maar vraag je je af hoe agile te rijmen is met al die vergaderingen? Of wil je gewoon weten hoe het werkt? In dit boek kun je lezen hoe je het beste uit agile haalt.

Nu technologie en inzichten steeds sneller veranderen, is agile een onmisbaar concept. Maar de bureaucratie die methoden als scrum met zich meebrengen, heeft weinig te maken met het oorspronkelijke gedachtegoed. Je wilt in je projecten en in je organisatie niet nog meer checklists en vergaderingen, je wilt gewoon snel inspelen op marktkansen.

In dit boek kun je lezen wat er mis is met de manier waarop we agile werken, hoe het bedoeld is is en hoe je dit succesvol kunt toepassen.

Christine Karman is ondernemer, uitvinder en software-ontwikkelaar. Ze maakte naam als oprichter van vernieuwende IT-bedrijven en was Technology Pioneer van het World Economic Forum. Ze ontwikkelde nieuwe technologie voor ondernemingen en was verantwoordelijk voor projecten bij uiteenlopende organisaties, van attractieparken tot overheidsinstanties.

9 789461 263575