Thema Audit & Agile
Tekst Drs. Jacob Witvoet RE EMIA RO
Beeld Adobe Stock

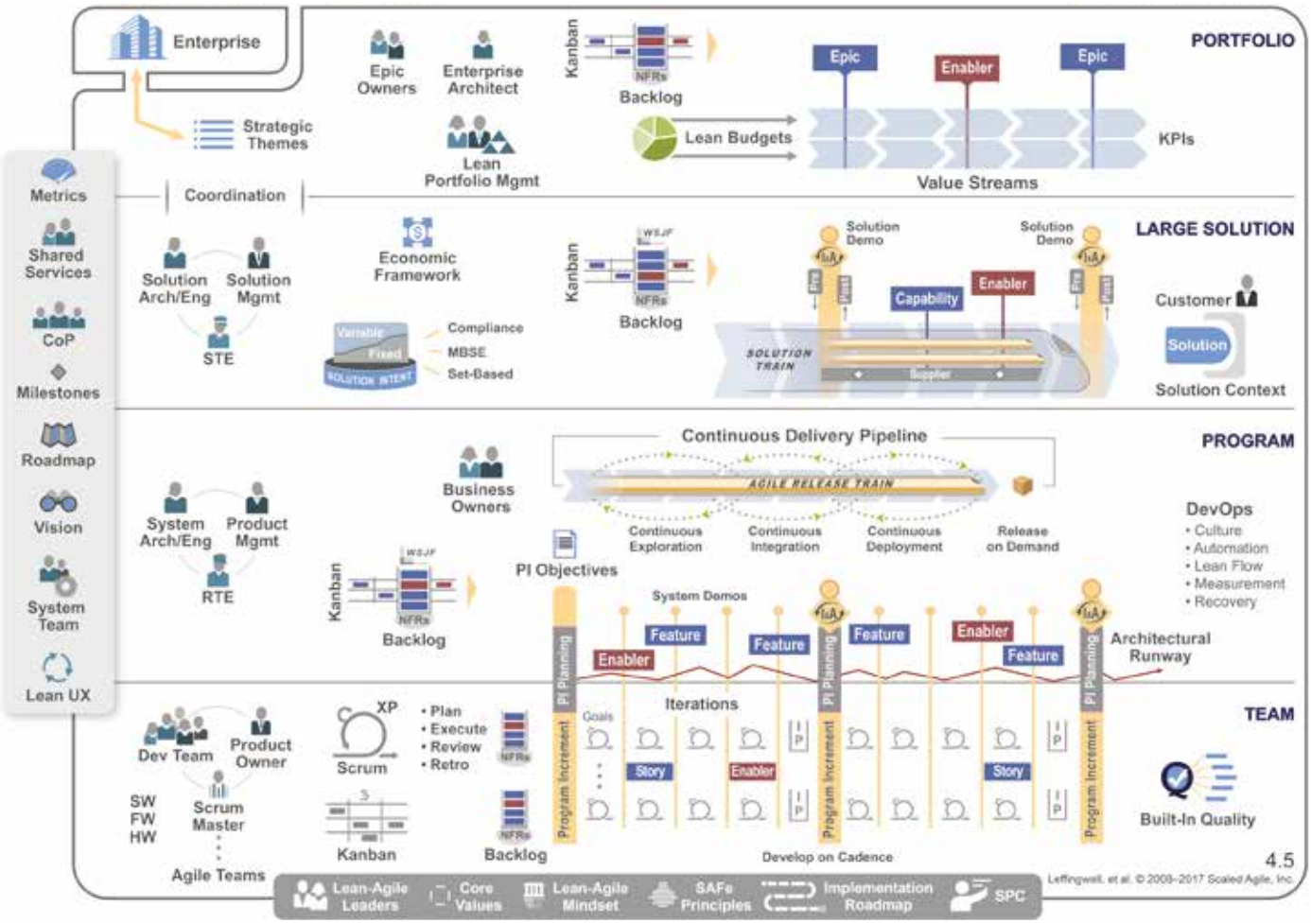
Agile opgeschaald

Het beoordelen van de kwaliteit van een IT-systeem (applicatie), op het moment dat het nog in ontwikkeling is, is lastig voor auditors. Zeker in agile-ingerichte omgevingen. In dit artikel mijn ervaringen met dit probleem, in een organisatie ingericht volgens het scaled agile framework.

Traditioneel werden grote IT-systemen vaak ontwikkeld via de zogeheten watervalmethode. Dat is een methode die uitgaat van heldere specificaties en volgordelijke ontwikkelstappen. Pas als het helemaal klaar is levert het systeem (economische) waarde. Momenteel is 'agile ontwikkelen' een hype, mede als reactie op de tekortkomingen die worden ervaren bij het watervalontwikkelen. Of misschien beter, als doorontwikkeling vanuit tekortkomingen van onder meer de watervalmethode. Voor het inrichten van een IT-ontwikkelproces of -organisatie op basis van agile-methoden zijn meerdere raamwerken bedacht. Een van de meest gebruikte raamwerken is SAFe®, het scaled agile framework. Een belangrijk uitgangspunt van dit raamwerk is dat, anders dan bij de watervalmethode, stukken werkende software eerder worden opgeleverd en daardoor eerder (economische) waarde wordt geleverd. Een ander typisch verschil met de watervalmethode is dat agile-methoden gericht zijn op het omgaan met wijzigingen (in specificaties). Bij waterval is dat juist een van de moeilijkheden.
Vanuit mijn eigen beleving ga ik in dit artikel in op enkele aspecten van het beoordelen van kwaliteit van eindproducten in een conform SAFe® ingerichte organisatie. Hierbij ga ik uit van de situatie dat de organisatie een nieuw systeem (applicatie) moet bouwen. Het is bedoeld als een eerste introductie, ik stip meer punten niet aan dan wel.

First time SAFe®
Voor het eerst geconfronteerd met een auditvraag in een agile IT-ontwikkelomgeving, ingericht volgens het scaled agile framework, was het even alsof ik in een andere wereld was beland. Hoe moest ik hierin, opgegroeid met auditen in watervalingerichte projecten, iets vinden over de kwaliteit van producten in wording in een (wat?) 'continuous development omgeving'? Hoe doe je dat als de makers praten in termen van epics, features, stories, in plaats van procesontwerp, functioneel ontwerp, technisch ontwerp? En de mensen waarmee je te maken krijgt product owners, scrum masters, developers met een T-profile en release train engineers zijn in plaats van projectleider, ontwerper en programmeur? Wat is dat nu weer: continuous deployment, continuous integration, WSJF, kanban, backlog, enabler, en, oh, de topper: architectural runway. Moet ik naar al die daily stand-ups, demo's, inspect and adapts en PI-events? Moet ik meesprinten? En spelen ze nu echt een spelletje: 'planning poker'? Ik had eerst een cursus én een woordenboek nodig.
Ik heb meerdere cursussen gedaan. Woorden vertaald. Alles gelezen wat ik kon lezen over agile en het scaled agile framework (SAFe®). Hoe meer ik er bekend mee werd, hoe meer ik herkende. Niet zo gek bleek, want SAFe® is ontworpen op basis van best practices en bekende organisatiedenkramen.

Waterval en audit op productkwaliteit
Vroeger, lang geleden, werkte ik als programmeur. De functie programmeur was de eerste stap op de IT-carrièreladder. Het was een tijd van Cobol programmeren met alleen een technisch ontwerp. Verder hoefde je niets te weten. 'Maak nu maar wat er op papier staat', zei de technisch ontwerper. Hij had het papier van de functioneel ontwerper in hapklare technische brokken vertaald en vooral opgeschreven. Wie die functioneel ontwerper was, ik wist het niet. Dat was ook niet nodig. Mijn vragen mochten alleen gesteld worden aan de



Figuur 1. Het scaled agile framework (SAFe®)

technisch ontwerper. Maar te veel vragen kwam je carrière niet ten goede, papier was immers de communicatieweg. En denk erom, geen fouten maken, want dat komt terug in je beoordeling. Dat was waterval, eerst een fase helemaal afronden dan de volgende beginnen. Eerst eisen opstellen, dan ontwerpen, dan bouwen, dan testen, dan klaar. Vaak voldeed het toch niet helemaal en had het veel te veel gekost. De wereld was immers al weer twee jaar verder. En het was even lang geleden dat de gebruiker de eisen mee had opgesteld en daarna grotendeels uit beeld was verdwenen. Gelukkig veranderde de wereld toen nog niet zo snel (en gelukkig overdrijf ik voor het verhaal een beetje).

Waterval was een mooie methode voor de auditor merkte ik toen ik later auditor werd. De gestelde eisen als uitgangspunt nemende, kon je tot het eindproduct de ontwikkellijn volgen en vaststellen dat het eindproduct al dan niet aan de eisen voldeed. Kon je dat niet, dan zat er iets fout in (de beheersmaatregelen van) het ontwikkelproces. Daar kwam geen soft control aan te pas. Stellige uitspraken, harde bewijzen. Vaststellen dat de gestelde eisen juist en volledig waren was wat lastiger, maar evident ontbrekende (niet-)functionele eisen waren vanuit de algemene normatiek en organisatiekaders nog wel te signaleren. Je kon altijd beginnen met heel globale eisen en nagaan hoe die navolgbaar waren uitgewerkt naar concretere eisen. Op globaal niveau is het formuleren van eisen immers niet zo moeilijk. En niet-navolgbaar is niet goed. Hoe je die eisen moet invullen en wanneer je met alle details een globale eis voldoende hebt ingevuld, dát kost hoofdbrekens.

Niet zó nieuw

Later werden mijn collega's en ik 'systeemontwikkelaars' en werkten we volgens de RAD-methodiek: rapid application development. Tweewekelijks met de gebruiker om tafel, specificaties afstemmen en laten zien wat je van die de vorige keer gemaakt had: beschreven, geprogrammeerd en getest. Hé, dat lijkt wel een sprint, een demo en een multifunctionele ontwikkelaar (developer met T-profile!). En 's ochtends stonden we altijd met zijn allen, op een vaste tijd, spontaan dat wel, een kwartiertje rond het koffieapparaat het werk door te nemen (en meer). Zowaar een dagelijkse stand-up.

Kortom, vanuit historie gezien zijn agile-methoden en -raamwerken niet uit de lucht komen vallen, maar is het een (achteraf) logische voortzetting van al lang lopende ontwikkelingen: mensen doen iets op een bepaalde manier, constateren dat het niet de resultaten oplevert die gewenst zijn en passen de werkwijze aan. Een van de mooie dingen is dat het zich aanpassen nu in de methode/wijze van organiseren zelf zit ingebakken. Het SAFe® Framework hangt samen van kleine en grote Deming-cirkels (Plan Do Check Act). Constante verbetering is een van de belangrijke basiswaarden in SAFe®, geformuleerd in het onderdeel de SAFe® House of LEAN.

Het SAFe® raamwerk

Het scaled agile framework (SAFe®) is een veelgebruikt raamwerk voor het inrichten van (IT-)organisaties die op grote

schaal op agile-wijze software ontwikkelen. SAFe® is gebouwd op de ideeën uit systeemdenken, agile-ontwikkelen en Lean-productontwikkeling. Het is volgens de makers opgebouwd uit verzamelde best practices uit de softwareontwikkelwereld. Het omvat meerdere organisatielagen en is, qua omvang, schaalbaar. Dit is beschreven in verschillende configuraties: full SAFe®, large solution SAFe®, portfolio SAFe®, essential SAFe®. In *figuur 1* is de configuratie portfolio SAFe® weergegeven. Voor de helderheid: dit betreft het ontwikkelen van IT volgens een continuous-developmentinrichting ofwel, ontwikkelen in een lijnorganisatie. Het omvat niet het agile-ontwikkelen in een (tijdelijke) projectorganisatie.

Een van de twaalf agile-principes uit het Agile Manifesto is: 'Welcome changing requirements, even late in development....'. Dit is onder meer verwerkt in SAFe® principe #3 'Assume variability, preserve options', het zo lang mogelijk openhouden van ontwerpopties in plaats van al vroeg in het proces dé optie kiezen en daaraan vast houden. Een belangrijk verschil met een watervalinrichting is dat wijzigingen (in prioriteiten, eisen, et cetera) 'omarmd' worden. Een belangrijk uitgangspunt dat ook in de mindset van de deelnemers opgenomen moet zijn. Maar wijzigingen in de teambemensing worden niet van harte omarmd, want dat komt in beginsel het presteren als team niet ten goede.
Van boven naar beneden in het framework worden de te ontwikkelen producten steeds concreter en in kleinere eenheden opgedeeld en beschreven. Van globaal naar specifiek. Beschrijvingen moeten net voldoende zijn voor het niveau waarop ze gebruikt worden. En vooral niet meer, dat is verspilling. Bovenin zijn het zogenaamde epics: een (beschrijving van een) initiatief voor een te realiseren product, service of systeem. Een epic wordt opgedeeld in meerdere features, een opleverbaar deel, klein genoeg om door een zogenaamde agile release train (een aantal samenwerkende teams) in één program increment (een vast aantal sprints) opgeleverd te worden. Een feature is weer opgedeeld in stories; een klein stukje functionaliteit dat door een team in één sprint kan worden opgeleverd.
Teams in een trein sprinten samen, in dezelfde cadans, en zijn met elkaar gesynchroniseerd. De daarvoor benodigde afstemmingen en coördinatie gebeuren in gezamenlijke bijeenkomsten (events) op teamniveau (de scrum events) en op treinniveau. Individuen en interacties zijn belangrijker dan processen en hulpmiddelen is immers het eerste statement in het Agile Manifesto. De verzameling events, op team of op treinniveau, vormen, in het klein en in het groot, PDCA-cycli. Kenmerkend voor het geheel is dat er wordt uitgegaan van min of meer vaststaande capaciteit en dat het werk van onderaf door de teams door de pijplijn wordt 'getrokken' (pull in plaats van push).
Voor meer informatie over dit framework verwijs ik naar de website www.scaledagileframework.com. Voor elk onderdeel uit het framework is daar een toelichting te vinden.

SAFe® en audit op productkwaliteit

Van belang om op deze plek even in herinnering te roepen is dat er in SAFe® idealiter kleine eenheden werkende software worden opgeleverd, niet een heel systeem in één keer. Wat niet per se hoeft te betekenen dat al die kleine deeltjes direct individueel in gebruik worden genomen. Kleine deeltjes kunnen worden samengevoegd alvorens beschikbaar te worden gesteld voor gebruik.
Ten opzichte van de watervalmethode heeft de auditor in een SAFe®/agile-omgeving een uitdaging als gevraagd wordt om vast te stellen of een product voldoet aan 'de daaraan te stellen' eisen. Als de daadwerkelijk gestelde eisen ten aanzien van het complete eindproduct al volledig zijn, dan zijn ze globaal. Opties worden zo lang mogelijk opengehouden. Keuzen worden zo laat mogelijk gemaakt. Pas op het moment dat het echt nodig is worden eisen concreet in detail uitgewerkt. Die uitwerking heeft dan uiteindelijk betrekking op een klein deeltje van het uiteindelijke geheel, zeg een user story. Hoe en in welke mate een globale eis uiteindelijk ingevuld gaat worden is daarmee inherent ongewis, behalve als alles klaar is. Uitzonderingen daargelaten.
Of een gerealiseerd deeltje van het geheel in voldoende mate aan de juiste eisen voldoet, is in deze ontwikkelopzet mede afhankelijk van hoe nog te ontwikkelen delen ingericht gaan worden. En als spiegelbeeld daarvan: de eisen die later gesteld moeten worden aan de nog te ontwikkelen delen zijn mede afhankelijk van de mate van invulling van een eis voor zover die al in andere delen is gerealiseerd. De toekomst is dus niet alleen afhankelijk van het heden, het heden is ook afhankelijk van de toekomst! En we hebben geen glazen bol. Stellige uitspraken en harde bewijzen worden waarschuwingen en onderwerpen voor discussie. Zeker als delen, en dat is wel het streven, gebruikt gaan worden voordat het geheel klaar is, vergt dit een parallel lopende betrokkenheid van de auditor bij de ontwikkeling van systemen.

Een eis die in zijn algemeenheid aan het ontwikkelen van met name nieuwe processen/systemen in een SAFe®-omgeving moet worden gesteld is dat de ontwikkelorganisatie te allen tijde in beeld heeft welke globale eisen zijn gesteld en in welke mate daar door welke onderdelen al invulling aan is gegeven. Zo is ook transparant waar (nog niet) aan voldaan is.
Het risico dat het geheel niet aan de eisen voldoet terwijl alle onderdelen wel aan de eisen voldoen, ligt op de loer. Alle gebouwde delen voldoen volgens de teams en de ontvangers aan de van toepassing zijnde eisen. Zij hebben ze immers gebouwd en goed bevonden. Wat echter niet gebouwd is, is niet beoordeeld. Op deze wijze kun je een goedgekeurde auto bouwen zonder wielen. En daarom moet je een auto als geheel keuren en niet alleen de auto-onderdelen.
Het is dus van belang dat op alle niveaus in het SAFe®-model opgeleverde applicatie(delen) worden beoordeeld op het voldoen aan de eisen. Van klein naar groot, van onder naar boven in het model. Daarom is het ook aan te bevelen dat elke laag zijn eigen 'definition of done' heeft als handvat om te bepalen wanneer een deeltje, deels of geheel naar wens en eis als afgerond kan worden geaccepteerd.

Aanknopingspunten voor de auditor
Voor de auditor die voor de vraag staat om te beoordelen of eindproducten voldoen aan de 'van toepassing zijnde' eisen levert het framework een aantal aanknopingspunten op. Enkele belangrijke daarvan zet ik hierna op een rij.

Aanknopingspunten vanuit de onderliggende principes en uitgangspunten van SAFe®
SAFe kent vier core values. Ingebouwde kwaliteit en transparantie zijn er daarvan twee waar (ook) de auditor zich op kan beroepen bij het uitvoeren van zijn taken.

Ingebouwde kwaliteit (waaronder documentatie!) – Een veronderstelling die ik nogal eens heb gehoord is dat in agile-ontwikkelingen/SAFe® onvoldoende documentatie (voor de auditor) zou zijn, aangezien teams vinden dat zij die documentatie niet nodig hebben voor hun taken. Het zou dus agile zijn die documentatie niet te maken. Echter, onder de core value 'ingebouwde kwaliteit' vermeldt SAFe® in de context van compliance dat het duidelijk is dat er een software requirements specification ontwikkeld en onderhouden moet worden ter ondersteuning van verificatie en validatie van het voldoen aan (bepaalde) eisen. Een zelforganiserend team komt er dus niet onderuit om geen (bepaalde) documentatie op te leveren.

Transparantie
Een andere core value van SAFe® is transparantie. Dit staat in relatie en ten dienste aan het creëren van een omgeving waarin vertrouwen in elkaar een belangrijke basis vormt. Om dit vertrouwen te creëren is openheid nodig. Daarom moet bijvoorbeeld voortgang gemeten worden aan de hand van objectieve maatstaven van werkende oplossingen. SAFe®-principe #3: 'objective evaluation of working software' biedt ook een argument om als auditor (namens de eigenaren!) in staat te worden gesteld de producten te beoordelen.

Aanknopingspunten vanuit het model van SAFe®
In de SAFe®-inrichting/processen komen zogenaamde artifacts voor die gebruikt kunnen worden om de kwaliteit van producten te beoordelen. Enkele voorbeelden.

Backlogs – Geprioriteerde werkvoorraad op elk niveau (team, trein, portfolio). Het prioriteren van de backlogitems is een spel waarin de business owner/product owner een belangrijke rol heeft. Het team heeft hier ook een rol in. In dit spel is het belangrijk dat verschillende belangen het juiste gewicht krijgen. Daarom kan het voor de auditor van belang zijn om het teamfunctioneren te betrekken in het onderzoek: dominantie van belangen (lees: een bepaald persoon die altijd een bepaald belang doordrukt) kan negatief zijn voor de kwaliteit van producten. Dit kan zich ook voordoen bij de uitvoering.

Definition of Done – In een Definition of Done is opgenomen wat gedaan moet zijn en/of aan welke criteria voldaan moet zijn wil een stuk werk als gereed beschouwd worden.

Epics, features, stories – In of bij epics, features en stories kunnen acceptatiecriteria geformuleerd zijn. Dit zijn eisen waaraan de opgeleverde producten op dat niveau moeten voldoen.

Events als aanknopingspunten
De overweging waar de auditor ook voor komt te staan is of, en welke, events geobserveerd zouden moeten worden. Zo zijn er daily stand-ups, planningsessies, product owner syncs, demo's, et cetera. Mijn ervaring tot nu toe is dat deze, in het opzicht van het beoordelen van de kwaliteit van producten, niet direct wezenlijk zijn om bij te wonen. Wel is het bijwonen van events nuttig om een beeld van, en gevoel over, het ontwikkelproces te krijgen. Daarmee kan de aanpak van de audit beter bepaald worden. Ook met betrekking tot de factor teamfunctioneren is het nuttig, ik denk zelfs noodzakelijk, bepaalde events te observeren.

Tot slot
Ik hoop met dit artikel een inkijkje gegeven te hebben in de agile-wereld van het op grote schaal software ontwikkelen en enkele uitdagingen voor de auditor daarin. Daarbij heb ik over meer aspecten niet kunnen schrijven dan wel. In de sfeer van agile heb ik van te voren de beschikbare hoeveelheid tijd voor dit artikel begrensd en de (hoeveelheid) inhoud daarop afgestemd. <<

Jacob Witvoet is auditmanager (IT en OA) bij de Auditdienst Rijk.

Opties worden zo lang mogelijk opengehouden. Keuzen worden zo laat mogelijk gemaakt