Agile requirements: user stories, use cases of beide?

Wat is de beste manier om requirements te beschrijven? De user story is op dit moment erg populair. Maar die heeft ook haken en ogen. Volgens Dennis Geluk voldoet de use case in bepaalde situaties beter. In dit artikel geeft hij handvatten voor het kiezen tussen beide en vertelt hij hoe je een combinatie van user stories en use cases kunt gebruiken.

door Dennis Geluk

Agile requirements

Er is al veel geschreven over agile, requirements en agile requirements maar wat mij betreft bestaan ‘agile requirements’ niet. Agile werken wel. Agile is immers een denkwijze, geen voorschrift. Het legt de nadruk op principes zoals ‘Just in Time’ en ‘Just Enough’ en wil voorkomen dat er tijdens een ontwikkelproces ‘verspilling’ ontstaat.

Het agile manifesto is wat dat betreft duidelijk: *“Working software over comprehensive documentation”*. De nuance zit hier natuurlijk in het gebruik van het woord: “over”. De essentie is niet dat je geen documentatie meer maakt maar dat je kritisch nadenkt of, wanneer en hoe je documentatie maakt.

Voor de agile revolutie bestonden requirements en documentatie vaak uit lijvige documenten met soms eindeloze lijsten van eisen en wensen of een verzameling use cases met een veelvoud aan flows. Aan het begin van het agile tijdperk werd dit vervolgens gereduceerd tot een schoenendoos met post-its (stories). Inmiddels hebben we ontdekt dat beide werkwijzen niet ideaal zijn en zoeken we naar de gulden middenweg.

> De meeste organisaties zijn agile gaan werken met een schoenendoos vol post-its en hebben het kind met het badwater weggegooid.

De gulden middenweg

Het vinden van deze gulden middenweg is vaak lastiger dan je denkt. De meeste organisaties zijn agile gaan werken met een schoenendoos vol post-its en hebben *het kind met het badwater weggegooid*. Alle oude vertrouwde pre-agile standaarden voor het opzetten van documentatie zijn hierdoor vaak zwaar verouderd. Organisaties schipperen nu tussen de oude, verouderde werkwijze en de nieuwe ‘laten-we-niets-opschrijven’ werkwijze.

Als ik dan toch zou moeten kiezen, dan kies ik de oude pre-agile werkwijze. Vaak was deze goed doordacht en voorzien van tooling en standaards. Daarnaast geldt ook meestal dat de essentie van deze werkwijze nog steeds goed is maar de uitvoering en de details niet agile en dus niet meer van deze tijd zijn.

Wist je bijvoorbeeld dat je use cases en user stories prima naast elkaar kunt gebruiken? Use cases hebben vooral een meerwaarde wanneer er behoefte is om vanuit een totaalbeeld de requirements en documentatie op een gestructureerde manier vast te leggen. Als de behoefte aan structuur en documentatie minder is, dan kan volstaan worden met user stories. User stories zijn immers perfect om kleinschalig zaken vast te leggen maar minder geschikt als structurele documentatie.

Ondanks de verschillen tussen use cases en user stories blijkt in de praktijk dat veel analisten use cases als een uitgebreide vorm van user stories beschouwen. Sterker nog, in sommige gevallen worden use cases en user stories zelfs als synoniemen gezien.

In dit artikel ga ik in op de belangrijkste overeenkomsten en verschillen tussen use cases en user stories en hoe beide samen gebruikt kunnen worden. Hierbij zal ik duidelijk maken waarom en met welk doel we bij DiVetro use cases en user stories gebruiken, want use cases en user stories hebben allebei (pas) hun nut wanneer ze op het juiste moment voor het juiste doel gebruikt worden.

Drie overeenkomsten tussen use cases en user stories

1. *Beschrijving vanuit het perspectief van de gebruiker*

Eén van de belangrijkste overeenkomsten tussen user stories en use cases is dat beide de gewenste functionaliteit van het systeem beschrijven vanuit het perspectief van de gebruiker (user). De focus ligt hierbij op de business value zonder in te gaan op de technische details van het systeem.

2. *Hulpmiddelen om de scope van een systeem te bepalen*

Zowel use cases als user stories kunnen gebruikt worden om de functionele en technische grenzen (scope) van een systeem vast te stellen. Voor een scope bepaling is het immers voldoende om een globale beschrijving van de functionaliteit en verantwoordelijkheden te hebben.

In het geval van use cases gebruiken we een use case model en een korte beschrijving per use case. In het geval van user stories beschrijven we per user story scherp wie welke toegevoegde waarde zal ondervinden.

> Als we alleen user stories gebruiken dan kunnen we de grenzen van een systeem niet direct uit de individuele stories afleiden.

Door het gebruik van actoren binnen een use case model is het ook relatief eenvoudig om de technische grenzen van een systeem te beschrijven. Als we daarentegen alleen user stories gebruiken dan kunnen we de grenzen van een systeem niet direct uit de individuele stories afleiden.

3. *Hulpmiddelen om de omvang van het systeem te bepalen*

Use cases en user stories kunnen allebei gebruikt worden om een schatting te maken van de omvang van een systeem of release. Wanneer we als basis voor een schatting alleen een use case model gebruiken met use cases met een zeer beperkte diepgang dan hebben we een schatting op een abstracter (hoger) niveau dan wanneer we een uitgebreide verzameling user stories als basis gebruiken.

Zoals gezegd kunnen use cases en user stories ook in combinatie met elkaar gebruikt worden. Meestal wordt het use case model dan gebruikt om een eerste (grove) schatting van de omvang van een systeem of release te maken en worden de user stories in een later stadium gebruikt voor een meer onderbouwde schatting.

Drie verschillen tussen use cases en user stories

1. *User stories worden primair voor planning gebruikt*

User stories zijn ontstaan binnen de Extreme Programming (XP) methodiek waarbij ze primair gebruikt werden als een ‘speelstuk’ in de planning game. Alle user stories komen in principe op de backlog en bieden daarmee een geprioriteerde lijst waarmee de product owner en het development team een volgende iteratie of release kunnen plannen.

Per individuele story worden de benodigde taken (development, test, requirements, etc…) bepaald, die nodig zijn om de gewenste functionaliteit op te kunnen leveren.

We zouden dus kunnen stellen dat een user story voor een significant deel een planningseenheid is. Uiteraard bevat een user story ook primaire requirements om de acceptatiecriteria van de story scherp te maken. Deze acceptatiecriteria kunnen vervolgens gebruikt worden om de story te valideren op volledigheid en correctheid.

Merk op dat, hoewel use cases een volstrekt ander doel hebben (zie hieronder), bovenstaande ook bereikt kan worden door het gebruik van use case slices! Use case slices komen voort uit de UC2.0 methode van Ivar Jacobson International en worden gebruikt om use cases op te knippen in kleinere, veelal functionele, bouwbare eenheden (slices). Een use case slice is daarmee nagenoeg gelijk aan een user story en is, net zoals een user story, primair een planningseenheid.

2. *Use cases worden primair gebruikt voor het gestructureerd vastleggen van requirements*

Use cases zijn ontstaan om alle mogelijke manieren waarop een gebruiker (actor) een doel kan bereiken gestructureerd te beschrijven. We kunnen veel eisen en wensen in de vorm van user requirements in één use case vastleggen.

Zoals hiervoor benoemd start iedere use case op een abstract niveau en kan deze indien nodig verder gedetailleerd worden. Volledig uitgewerkte use cases beschrijven, ondersteund door een termenlijst, domeinmodel, bedrijfsregels en diverse aanvullende eisen, de volledige (functionele) werking van een systeem.

Ongeacht de toegepaste diepgang is het primaire doel van use cases dus het gestructureerd vastleggen van eisen en wensen die aan een systeem gesteld worden.

Dennis Geluk

Dennis Geluk is een senior informatieanalist en partner van DiVetro. Hij heeft meer dan 15 jaar IT ervaring. Dennis is een gecertificeerd Use-Case 2.0 coach. Hij heeft bij verschillende organisaties een spilfunctie vervuld bij het implementeren van UC2.0, het invoeren van agile werken en het verbeteren van requirements standaarden.

Dennis is momenteel werkzaam binnen het Digitizing KLM programma van de KLM. Binnen dit programma is Dennis verantwoordelijk voor het implementeren van een vernieuwde agile werkwijze voor business- en informatie analisten.

https://www.linkedin.com/in/dennis-geluk/

3. Use cases hebben een waarde als naslagwerk

Bij het gebruik van user stories wordt meestal voor iedere wijziging een nieuwe user story beschreven zonder de oude aan te passen. Hierdoor kan het lastig worden om na oplevering een scherp beeld te hebben van de actuele functionaliteit.

Omdat use cases primair bedoeld zijn voor het gestructureerd vastleggen van requirements hebben zij, mits bijgehouden, een meerwaarde als naslagwerk. Een use case beschrijft immers de volledige functionele flow en een user story vaak slechts een specifieke requirement of onderdeel van een flow op het moment van schrijven. Daarmee zijn individuele stories minder geschikt als naslagwerk.

Use cases en user stories: de combinatie in de praktijk

Als we de overeenkomsten en verschillen tussen use cases en user stories in het achterhoofd houden dan is het niet moeilijk om typische situaties te identificeren waarin we user stories en use cases samen willen gebruiken.

In mijn dagelijkse praktijk herken ik dergelijke situaties aan de volgende kenmerken:

- Er wordt agile gewerkt (veelal Scrum);
- Er zijn meerdere agile teams die binnen hetzelfde domein werken (veelal SAFe of Nexus);
- Er wordt gewerkt aan “een systeem” dat bedrijfskritisch is en meestal een verwachte levensduur heeft van meer dan 3 jaar;
- De werking van het systeem is vaak niet triviaal en bevat hier en daar complexe logica.

Vanuit de agile gedachte is het niet meer dan logisch dat er gewerkt wordt met Epics, Features en Stories om de teams aan de slag te houden. Door de grote hoeveelheid stories over meerdere teams en de complexiteit zijn features en stories meestal onvoldoende om goed en snel een beeld van de geïmplementeerde requirements te vormen. Wat staat er nu eigenlijk daadwerkelijk buiten? Hier komen de “vertrouwde” use cases weer om de hoek kijken. Use cases zijn namelijk zeer geschikt om gestructureerd je requirements vast te leggen vanuit een breder perspectief. Het risico is hier echter dat we door het gebruik van use cases ook kunnen terugvallen in een oude valkuil: te veel en te uitgebreid documenteren. Om dit te voorkomen heeft Ivar Jacobson (de geestelijk vader van de use cases) use case 2.0 beschreven. Use case 2.0 kan ons helpen om just-enough en just-in-time informatie in use cases te verwerken.

Hoe UC2.0 invulling kan geven aan just-enough en just-in-time

Het grote verschil tussen traditionele use cases en use case 2.0 is dat de just-enough gedachte heel nadrukkelijk in use case 2.0 is ingevoegd. Om goed en duidelijk afspraken te maken over just-enough zijn er bijvoorbeeld statussen gedefinieerd waarin een use case

tekst zich kan bevinden. Deze statussen maken het makkelijk en eenduidig om als team afspraken te maken over de gewenste diepgang van de documentatie.

De essentie van just-enough en just-in-time binnen UC2.0 is om altijd te starten op een abstract niveau zonder veel details (briefly described). Vanuit dit abstracte niveau bepalen we per story die we in de sprint oppakken of deze story een reden is om de documentatie verder te verdiepen.

De essentie van just-enough en just-in-time binnen UC2.0 is om altijd te starten op een abstract niveau zonder veel details (briefly described). Vanuit dit abstracte niveau bepalen we per story die we in de sprint oppakken of deze story een reden is om de documentatie verder te verdiepen.

De essentie van just-enough en just-in-time binnen UC2.0 is om altijd te starten op een abstract niveau zonder veel details (briefly described). Vanuit dit abstracte niveau bepalen we per story die we in de sprint oppakken of deze story een reden is om de documentatie verder te verdiepen. Het oppakken van een zeer complexe user story zou bijvoorbeeld een reden kunnen zijn om een deel van de use case tekst verder te detailleren tot bijvoorbeeld het niveau “fully described”.

Het is uitermate belangrijk om iedere keer per user story die we oppakken een afgewogen keuze te maken! Per story bepalen we of het nodig is om meer details aan de use case toe te voegen. Het resultaat van deze werkwijze is een use case waarbij bepaalde stukken in meer en andere stukken in minder detail zijn uitgewerkt.

Voor de meeste analisten is dit even wennen. Als je echter eenmaal los kunt komen van de “use case is nog niet af” valkuil dan kun je goed wisselen in de diepgang van je document. Het just-in-time principe is veelal projectspecifiek. Zo ken ik organisaties waar iedere user story minimaal 3 sprints doorloopt (analyse, bouw en test) en organisaties waar alles in 1 sprint doorlopen wordt. Het is vooral belangrijk dat je als analist probeert zo min mogelijk vooruit te werken en pas details toevoegt als ze ook echt gerealiseerd zijn (of worden).

Conclusie:

Use cases en User stories hebben duidelijke overeenkomsten en verschillen. Afhankelijk van de specifieke projectsituatie kun je ervoor kiezen om use cases naast user stories te gebruiken. Houd hierbij altijd rekening met het primaire doel van zowel de use case als de user story en op welk vlak zij een meerwaarde hebben.

De ervaring leert ons dat je binnen een agile project kunt kiezen voor het gebruik van:

1) alleen user stories of
2) een combinatie van user stories en use cases.

In een agile setting is het gebruik van alleen use cases zeker onvoldoende; je heb immers stories nodig om het proces te sturen. In onze visie zou iedere niet technische user story zijn oorsprong moeten vinden in een use case.

Afhankelijk van een top-down of bottom-up benadering ontstaat een user story uit een use case (top-down) of ontstaat de use case (achteraf) uit een verzameling user stories (bottom-up). De use case wordt dan gebruikt om alsnog structuur aan te brengen in de verzameling van stories en als basis voor de functionele documentatie.

De ‘verantwoordelijkheid’ van user stories en use cases is dus duidelijk: een user story faciliteert primair de planning en de scope van het werkpakket terwijl een use case een middel is voor het structureel vastleggen van requirements.