Eenvoud werkt

Zeven vuistregels voor softwareontwikkeling

Een belangrijke vraag bij het implementeren van een software ontwikkelmethode is altijd, hoe de methode concreet gemaakt moet worden voor de organisatie en het project waarvoor hij is bedoeld. In onze drang om het goed te doen wordt deze stap nog al eens vergeten en ziet men het volledig en zo zuiver mogelijk toepassen van de methode als doel op zich. Veel ontwikkelmethoden (DSDM, Scrum, XP) maken deze stap niet expliciet maar veronderstellen hem wel degelijk. In het Rational Unified Process (RUP) is deze stap wel expliciet verwoord in het eerste uitgangspunt: "Adapt the Process" (pas RUP aan). Het is zeker geen makkelijke stap omdat hiervoor kennis van en ervaring met de methode nodig is.

Welke regels hanteer je nu om deze stap te zetten en daarmee te komen tot een duidelijke en vooral concrete inrichting van het ontwikkelproces te komen, die jouw project optimaal ondersteunt? Welke tooling richt je hiervoor in en hoe en door wie laat je deze gebruiken? Hieronder volgen een paar situaties waarin deze stap niet bewust genoeg is genomen:

1. Een bedrijf heeft een dure tool aangeschaft maar schrikt zo van de hoeveelheid tijd die het kost om deze in te richten en up-to-date te houden dat het gebruik ervan nooit van de grond komt.
2. Er zijn reviewformulieren ontwikkeld waarvan het invullen zoveel tijd vergt dat iedereen probeert om reviews te vermijden.
3. RUP definieert 128 werkproducten die allemaal gemaakt moeten worden waardoor het project tot stilstand komt.
4. Het project moet snel af dus er wordt een groot ontwikkelteam neergezet, maar de klant blijkt niet in staat om voldoende input te leveren.
5. De goedkeuringsprocedure voor wijzigingen duurt minimaal 3 maanden waardoor een groot deel van de aangevraagde wijzigingen niet meer relevant is tegen de tijd dat ze zijn gerealiseerd.
6. De mensen die de requirements hebben opgesteld zijn tijdens de bouw van het systeem niet meer beschikbaar waardoor het systeem niet aansluit bij de wensen van de klant.
7. We hebben heel veel tijd geïnvesteerd in het inrichten en bijhouden van traceability tussen requirements en code, maar het inschatten van wijzigingen gaat nauwelijks sneller.

In dit artikel geven we zeven vuistregels die je helpen om de ontwikkelmethode van je keuze effectief te maken. Het volgen van deze vuistregels had elk van de bovengenoemde situaties kunnen voorkomen.

Kosten en baten

Overkoepelend voor de zeven vuistregels geldt: weeg kosten en baten tegen elkaar af. Een maatregel (document, procedure of inzet van een tool) is effectief als hij het juiste effect bereikt: een optimale balans tussen kosten en baten. Het gaat bij kosten en baten niet alleen over de periode van het software ontwikkelproject, maar over de gehele levenscyclus van een applicatie en de context waarbinnen deze ontwikkeld en gebruikt wordt. Een belangrijke vraag daarbij is wat de verwachte levensduur van de te bouwen applicatie is en hoeveel wijzigingen te verwachten zijn. Een korte levensduur geeft heel andere overwegingen dan een lange.
Vaak zien we dat er bij een project enthousiasme is voor de te verwachten baten van een maatregel, maar dat daarbij de kosten uit het oog worden verloren. Ook andersom komt voor: vanwege de kosten wordt een 'slimme' oplossing niet gekozen, terwijl er op langere termijn veel baten tegenover zouden staan. Kosten of baten op zichzelf zijn geen criterium, maar het resultaat van de afweging tussen beide levert inzicht op.

Vuistregel 1: Ceremonieel niveau

Ceremonieel niveau kan worden gedefinieerd als de hoeveelheid toevoegingen aan het ontwikkelproces t.b.v. controle over het proces en de resulterende werkproducten. Een hoger ceremonieel niveau impliceert dus meer formele vastlegging, review en goedkeuring. Concreet betekent dit meer

werkproducten en procedurebeschrijvingen en strakkere sturingsmogelijkheden hierop. Elke organisatie kent haar eigen 'ceremonies'. Meestal geldt: hoe groter, hoe formeler. Enkele voorbeelden van lager en hogere ceremonieel niveau zijn te vinden in de onderstaande tabel:

Lager ceremonieel niveau	Hoger ceremonieel niveau
Informatie is beschikbaar via een wiki, iedereen kan hierop kijken.	Er wordt expliciet vastgelegd wie moet worden geïnformeerd, en op welke manier.
De werkproducten zelf weerspiegelen wat er is afgesproken.	Elke bijeenkomst wordt genotuleerd, de notulen worden ter goedkeuring voorgelegd.
Informele uitwisseling van informatie.	Informatie wordt vooral schriftelijk uitgewisseld.
Kwaliteit wordt door het team zelf onderkend en bewaakt.	Er is een afdeling Quality Assurance.
Wijzigingen kunnen soepel worden verwerkt.	Er is een commissie belast met het goedkeuren van wijzigingen.
Reviews vinden informeel plaats.	Alle reviewresultaten moeten worden vastgelegd.
Standaards en richtlijnen zijn impliciet.	Er zijn uitgebreid beschreven standaards en richtlijnen.

Zowel de kosten- als de batenkant van het gehanteerde ceremonieel niveau is niet altijd even duidelijk. Bij zaken als formele reviews, traceability van requirements, een uitgebreide goedkeuringsprocedure en detailplanningen met Gantt charts wordt ervan uit gegaan dat ze onmisbaar zijn. Wat echter ontbreekt, is een gedegen inschatting van de kosten om deze zaken op te pakken en een kwantificering van wat ze opleveren. Door hier expliciet aandacht aan te besteden worden mensen in de organisatie zich opnieuw bewust van de overhead die het ceremoniële niveau met zich mee brengt en kunnen hier afgewogen keuzes in gemaakt worden.

Vuistregel 2: Het team

Het meest efficiënt qua overdracht is een ontwikkelteam van precies 1 ervaren persoon, die alle rollen in zichzelf kan verenigen. Vaak staan echter niet alleen de doorlooptijd en beschikbaarheid van één persoon met alle benodigde kennis en vaardigheden deze optie in de weg. Ook overwegingen als ontwikkelsnelheid, kwetsbaarheid (wat gebeurt er als deze persoon wegvalt) en kennisborging spelen mee.

Stem het aantal teamleden af op de hoeveelheid input die de opdrachtgeverorganisatie op wekelijkse basis kan leveren. Het heeft weinig zin, een team van 3 ontwerpers en 8 ontwikkelaars te hebben zitten als de maximale input van de gebruikersorganisatie slechts toereikend is voor 1 ontwerper en 2 ontwikkelaars.

Verder levert een groter team niet een evenredig grotere productiviteit vanwege de grotere overhead in planning, afstemming en afhankelijkheden. Het kan zelfs zijn dat een goed functionerend team van 3 ontwikkelaars, een architect, een ontwerper en een tester meer productie levert dan hetzelfde team uitgebreid met nog 3 ontwikkelaars, een ontwerper en een tester, omdat dit voor de doorlooptijd noodzakelijk zou zijn. Als een dergelijke opschaling dan ook nog eens tegen het einde van het project plaatsvindt om toch maar de deadline te halen is dit eerder blussen met benzine dan een oplossing.

Vuistregel 3: Werkproducten

Bij het opstellen van een lijst met te maken werkproducten zijn de belangrijkste vragen: wie zal het werkproduct missen als je het niet maakt, en wat zal er dan mislopen. Deze vraagstelling levert ook inzicht op in het doel waarvoor het werkproduct wordt gemaakt: is dit alleen ten behoeve van het ontwikkelproces (tussenproduct) of zal het ook bij toekomstig beheer en onderhoud een rol spelen (onderdeel van het eindproduct). Als je weet voor wie je iets schrijft kun je die belanghebbende een rol laten spelen in de totstandkoming en zo valideren dat de inhoud voldoet en de inspanning om het te maken opweegt tegen de te verwachten baten.

Als je als enige werkproduct de applicatie oplevert en de opdrachtgeverorganisatie kan deze gebruiken en beheren, dan is dat op de korte termijn voldoende. Dit geldt met name als er echt Agile gewerkt is en dus alle code is voorzien van unit tests en codedocumentatie en door refactoring zo eenvoudig mogelijk is gehouden. Voor de langere termijn is een breder gedocumenteerde applicatie beter.

Vuistregel 4: Werkproducten goedkeuren

Niet voor alle werkproducten is formele goedkeuring vereist. Ook voor de formele acceptatie kun je steeds de vraag stellen wat er misgaat als een werkproduct niet formeel wordt geaccepteerd. Is het antwoord 'niets', dan is duidelijk dat formele goedkeuring achterwege kan blijven. Vaak zijn werkproducten die de scope van het project afbakenen kandidaten voor formele acceptatie. Ook use case specificaties zullen vaak worden goedgekeurd om een gemeenschappelijk referentiepunt voor acceptatie van de oplossing te hebben.

Formele acceptatie kost tijd en inspanning. Het levert helderheid, baselines en daarmee terugvalmomenten op. Indien er een duidelijk mandaat van één persoon aanwezig is voor goedkeuring, vereenvoudigt dit het proces. Zo kan bijvoorbeeld een

gemandateerd domeindeskundige heel snel beslissen of een use case specificatie goed is. Hij kan de use case dan zelf formeel accepteren. Het heeft weinig toegevoegde waarde om het formele acceptatiepunt een trede hoger in de hiërarchie te leggen.

Vuistregel 5: Toolset

Het wordt meestal onderschat wat een goed draaiend team aan de requirements en management kant allemaal kan doen met een groot whiteboard, een brown paper en geeltjes, tekstverwerker, spreadsheet, een simpele modelleertool en een Wiki. Voeg alleen extra tools toe vanuit een duidelijke behoefte of een duidelijk knelpunt. Zorg dat je een concreet beeld van de oplossing voor zo'n knelpunt hebt en deze zich in de praktijk heeft bewezen voordat je een tool gaat kiezen. Vanuit de geprioriteerde eisen kijk je naar de eenvoudigste tool die aan die specifieke eisen voldoet. Vaak komen een paar losse gespecialiseerde tools beter tegemoet aan de eisen dan een complete geïntegreerde toolset.

Vuistregel 6: Traceability werkproducten

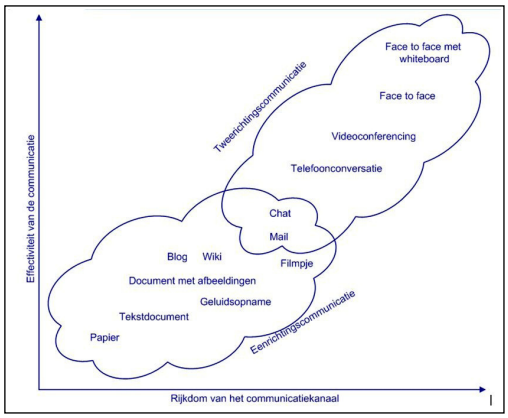
Traceability of traceerbaarheid betekent grofweg twee dingen: er is samenhang aangebracht tussen beslissingen in de tijd, of tussen bij elkaar horende werkproducten in één baseline. Het kan expliciet worden aangebracht (in een tool of spreadsheet) of impliciet door het volgen van design regels of naamgevingsconventies. Traceability is gewenst omdat het kan helpen bij:

- Het in lijn houden van business requirements, high-level en detail software requirements, testgevallen en gerealiseerde functionaliteit.
- Het bepalen van de impact van wijzigingen, ook als het oorspronkelijke ontwikkelteam niet meer beschikbaar is.
- Het herleiden waarom bepaalde beslissingen zijn genomen, door wie en wanneer.

Expliciete traceability van high-level naar detail software requirements en gerealiseerde functionaliteit is bijvoorbeeld zinvol in situaties waarbij gewijzigde wet- en regelgeving tot aanpassingen in de applicatie kan leiden. De aangebrachte traceability kan helpen bij het bepalen van de impact van deze wijzigingen. Bedenk dat traceability nooit als vervanging voor een intelligente (menselijke) inspectie kan dienen. Traceability wordt ook aangebracht om te zorgen dat aan alle high-level requirements tegemoet gekomen wordt. Bij het detailleren van requirements treedt voortschrijdend inzicht op waardoor de eerder geformuleerde high-level requirements mogelijk niet meer relevant zijn. Ook hier is een menselijke inspectie nodig. Overweeg of het alléén inzetten van de menselijke factor niet goedkoper is. In veel gevallen kan traceability ook impliciet worden aangebracht en is dit goedkoper. Hierbij kunnen bijvoorbeeld naamgevingsconventies en objectgeoriënteerd werken helpen. Hoe inzichtelijker een applicatie is opgebouwd, hoe beter de impliciete traceability.

Vuistregel 7: Communicatie

Communicatie verloopt effectiever naarmate de bedoeling van de ene partij sneller en beter begrepen wordt door de andere partij. De beste manier om dit te bereiken is door de partijen live met elkaar te laten spreken, waarbij ze kunnen visualiseren wat ze bedoelen. In de onderstaande figuur zijn de verschillende communicatiekanalen en hun effectiviteit afgebeeld.



Het is wel van belang te bedenken waarvoor een bepaald medium wordt gebruikt. Om een bedoeling over te brengen staat geschreven tekst onderaan. Echter, als vastlegging van een stuk communicatie of als ondersteuning hiervan is een papieren document prima. Het is dan een ondersteuning van de eigenlijke communicatie.

Conclusie

Nog even de zeven vuistregels op een rij:

1. Houd het ceremonieel niveau zo laag mogelijk
2. Houd het team zo klein mogelijk
3. Maak zo weinig mogelijk werkproducten
4. Laat zo min mogelijk werkproducten goedkeuren in een zo simpel mogelijke goedkeuringsprocedure
5. Neem de simpelste toolset die voldoet
6. Maak zo weinig mogelijk werkproducten (expliciet) traceable
7. Kies de effectiefste vorm van communicatie

Voor de invulling van elk van deze vuistregels is het essentieel om kosten en baten tegen elkaar af te wegen. Met name voor de baten op de lange termijn is dat nogal eens lastig maar wel uiterst zinvol. Het expliciet maken van de te verwachten baten is namelijk onmisbaar om te kunnen meten of de gekozen maatregelen wel effectief zijn. De resultaten van deze metingen zijn op hun beurt onmisbaar om het gehanteerde ontwikkelproces effectief verder te kunnen optimaliseren.


Remi-Armand Collaris (l) en Eef Dekker zijn beide werkzaam bij Ordina.