Inleiding in projectmethodes

Waterval & Agile/Scrum

Graduaatsopleidingen – PXL Digital

2020-2021

Inleiding in projectmethodes waterval & agile/scrum

Om projecten tot een goed einde te brengen, zijn er projectmethodes ontwikkeld. Zo'n methode toepassen, vergroot de kans op het slagen van een project. Het zijn afspraken en richtlijnen over hoe je op een goede en gestructureerde manier projecten kan aanpakken.

Wij behandelen in dit hoofdstuk enkele soorten projectmethodes: de klassieke watervalmethode (die bestaat al het langst), de scrum methode en de kanban methode.

Op het einde van deze cursus over projectmethodes, zal je begrijpen wat de typische kenmerken en grote verschillen zijn van waterval t.o.v. scrum, wat de voor- en nadelen zijn van beide, en welke rollen medewerkers hebben in dergelijke projecten en projectteams. Ook zal je weten hoe kanban verschilt t.o.v. scrum.

INHOUDSTAFEL:

1. Klassieke ontwikkeling via de watervalmethode

In deze methode voer je een project stap na stap uit in verschillende fases, van analyse tot in productiestelling en onderhoud. Een volgende fase start maar als de vorige fase is afgelopen, goed gedocumenteerd, en liefst ook officieel afgesloten is.

1. **Analyse:** Hier wordt nagedacht en vastgelegd WAT je gaat automatiseren of gaat uitvoeren in het project (bv. wat je gaat maken, installeren, of configureren). Je spreekt met de klant af wat van het project verwacht wordt. De **scope** is het resultaat van een eerste analyse, het is een verzameling van *requirements* (dat zijn eisen) die aangeven wat je wil uitvoeren, implementeren of oplossen in het project. Het beschrijft alles wat de klant wil om het doel (=objectief) van het project te bereiken.

2. **Ontwerp**: In deze fase wordt nagedacht HOE je de oplossing zal maken

3. **Implementatie**: In deze fase wordt het project uitgevoerd, je maakt de oplossing die je tijdens de ontwerpfase bedacht hebt

4. **Testen**: De oplossing moet getest worden. Er wordt gekeken of die voldoet aan de verwachting. Als het goed is en de oplossing aanvaard wordt door de klant, wordt de IT-oplossing in productie genomen of het eindproduct overhandigd aan de klant

5. **Uitrol en onderhoud**: Nadat het product wordt uitgerold (in productie of in gebruik genomen), wordt het eindproduct nog onderhouden, d.w.z. dat fouten of problemen opgelost (kunnen) worden.

De watervalmethode toepassen, betekent dat deze fases van analyse t.e.m. onderhoud fase na fase plaatsvinden, na elkaar, als een “vloeiende waterval”. Een nieuwe fase kan maar starten als de vorige is afgelopen.

Gedurende al die fases wordt aan de oplossing, het eindproduct gewerkt. Dit eindproduct is afgesproken aan het begin van het project.

In het begin van een project wordt er immers afgesproken wat je wil bereiken, m.a.w. het doel (of objectief) van het project.

Voorbeeld : bouw van een huis

Doel/objectief van het project: we willen een nieuwbouwwoning op onze gekochte grond bouwen, een huis waar wij de eigenaar van zijn en waar we met ons gezin minstens 20 jaar kunnen wonen.

Scope: een moderne nieuwbouwwoning met minstens 4 slaapkamers, een bureau, een dubbele garage, een groene tuin.

Timing: we hebben anderhalf jaar, tegen dec 2021 moet het huis instapklaar zijn.

We passen nu op dit project een watervalmethode toe:

Er wordt dus fase na fase gewerkt:

- Analyse: Eerst denken we na over wat we verwachten in het huis en zo definiëren we de *scope*: het huis heeft minstens 4 slaapkamers, een grote woonkamer met open keuken, een bureau voor minstens 2 personen, een muziekkamer waar een vleugelpiano kan staan, een verdieping met minstens 4 slaapkamers en een badkamer.

- Ontwerp: We spreken af met een architect die een tekening/ontwerp maakt van het huis. Hij tekent uit hoe het huis er zal uitzien. Ook een grondplan van de kamers wordt uitgetekend. We keuren het ontwerp goed alvorens er gestart kan worden met de bouw.

- Implementatie: Het huis wordt gebouwd zoals is afgesproken in het ontwerp.

- Test: Met de architect en veiligheidscoordinator wordt er nagekeken of alles veilig en correct verloopt en of alles klopt met de afspraken. Heeft de trap de juiste hellingsgraad? Alles wordt goed nagekeken.

- In gebruik nemen en onderhoud: Als alles klaar is, kan je verhuizen. Er wordt normaal gezien nog een periode van garantie afgesproken met de aannemer of architect.

Dit is duidelijk een watervalmethode. Stap voor stap wordt het huis gebouwd en in orde gebracht. Op een bepaald moment is het volledige einddoel bereikt: het huis is instapklaar.

2. Agile ontwikkelmethode

2.1. Waarom agile?

De watervalmethode heeft als nadeel dat het zou kunnen dat je verwachtingen anders zijn tegen dat je project klaar is, en dat je moeilijk(er) op die veranderingen hebt kunnen inspelen.

Agile organisaties spelen gemakkelijker in op veranderingen rondom zich, bij organisaties of op de markt. Veranderingen gebeuren altijd maar sneller en komen meer en meer voor. Als je agile bent, pas je je snel aan aan de wensen van de klant.

Agile werken op projecten betekent niet harder werken maar slimmer werken. Het gaat niet om meer werk te doen in minder tijd maar om meer waarde te creëren met minder werk.

De nadruk ligt op het efficiënt en snel opleveren van producten en/of services die waarde leveren voor de klant.

Als je zou projecten doen waarbij je je baseert op technologie “omdat die nieuw is” of “omdat de medewerkers het tof vinden”, riskeer je dat je eindproduct iets is wat klanten niet willen of waarvoor ze niet willen betalen.

2.2. Wat is Agile? De ervaring

We leggen op een praktische manier uit wat “agile” is aan de hand van een balspel of een Corona-proof variant. Zo ervaar je de principes van agile in de prakijk.

Het balspel:

Dit is een gekende werkvorm die ook bij bedrijven of in opleidingen wordt gebruikt.

We verdelen in teams. Het doel is om als team zoveel mogelijk balletjes door een “proces” te krijgen op 2 minuten tijd.

Je moet daarbij wel de afgesproken spelregels hanteren:

- Elke deelnemer moet elke bal tenminste 1 keer aangeraakt hebben
- De bal moet “air time” hebben bij het wisselen (dus even los zijn van je handen)
- Als de bal op de grond valt, moet hij terug naar de eerste speler
- Je mag een bal niet aan de persoon direct links of rechts van je geven
- Elke bal moet uiteindelijk terugkomen bij de eerste speler

Als team zorg je zelf dat je op een juiste manier het aantal balletjes telt dat je op 2 minuten rond kan krijgen in het team.

Het team doet dit spel een paar keren.

- Het team bepaalt zelf wie de eerste speler is.
- Het team krijgt eerst 2 minuten om je strategie te bespreken.
- Dan start een eerste ronde van 2 minuten ballen doorgeven.
- Het resultaat wordt genoteerd.
- Vervolgens krijg je opnieuw de tijd om met het team de strategie te bespreken om het een volgende ronde beter te doen.
- Deze ronde wordt 3x gespeeld, 3x moet het team zich verbeterd hebben.

Door het spel krijg je inzichten, in wat net met agile te maken heeft:

- **Zelforganisatie** werkt. Je hebt niet altijd een leider nodig die vertelt wat je moet doen. De spelregels waren duidelijk en binnen die spelregels heeft het team zelf bepaald HOE ze het gingen doen.

- Als iedereen zich betrokken voelt en zich engageert, werkt het het best. Zogauw teamleden minder betrokken zijn, is het resultaat een pak minder goed. Iedereen betrokken houden en vertrouwen hebben in mekaar dat het goed komt, is belangrijk.

- Lang plannen vooraf heeft niet altijd zin, gewoon DOEN en PROBEREN, opnieuw proberen (in iteraties) en jezelf VERBETEREN (=> Leren en aanpassen)

- Samen met het team zet je je beste beentje voor om het beste resultaat te halen voor wat gevraagd werd -> focus op waarde voor de klant.

2.3. En scrum dan? Wat is scrum?

Agile verwijst naar het "wendbare" van bv. een organisatie of een projectaanpak.

Scrum is een methode om agile te werken, een methodologie die de agile principes volgt.

Als je een scrum werking toepast, is het de bedoeling dat je je aan een aantal principes houdt die scrum voorschrijft. Als je in een "scrum project" terechtkomt, zal je normaal gezien deze principes herkennen (tenzij de organisatie het niet helemaal of anders toepast).

Historiek:

Scrum is een term die afkomstig is uit het rugby. Het is in die sport een manier om een spel te hervatten na een kleine, technische overtreding. Bij een scrum gaan 2 groepen spelers in een voorover gebogen houding tegen elkaar induwen. Het belang van het teamwerk wordt hiermee benadrukt.

De grondleggers van scrum waren Ikujiro Nonaka en Hirotaka Takeuchi. Zij beschreven een productontwikkelraamwerk dat vergeleken wordt met de manier waarop een rugby-team als groep de achterlijn van de tegenstander probeert te bereiken.

Samenwerking, aanpassingsvermogen, snelheid en zelfsturing zijn kenmerken van multidisciplinaire teams die overeenkomen met zo'n rugbyteam.

Begin 1986 werd hierover een artikel gepubliceerd in de Harvard Business Review, dat is een invloedrijk vaktijdschrift over bedrijfseconomie dat publiceerd wordt door de Harvard Business School (in Harvard, nabij Boston).

Naar aanleiding van dit onderzoek ontwikkelde Jeff Sutherland rond 1993 het scrumproces. Ken Schwaber paste dan weer een eigen benadering bij zijn bedrijf toe. Samen werkten ze dit verder uit en in 1995 werd scrum officieel een software ontwikkelmethode.

Scrum is dus ontwikkeld in de USA en het werd eerst vooral gebruikt in Engelstalige projecten, vandaar de Engelstalige terminologie.

2.4. Vaste gekende termen in scrum

Werk je later in een scrumteam in een organisatie, dan zullen normaal gezien de principes toegepast worden die hieronder beschreven staan. Het voordeel van scrum te gebruiken is, dat waar je ook werkt, je altijd dezelfde manier van werken en termen zal horen als het team scrum toepast.

Enkele belangrijke termen zijn:

Sprint: Een sprint heeft een vaste lengte van 1 tot 4 weken (elke organisatie of projectorganisatie kan dat zelf kiezen). In die sprints is het de bedoeling om werkende (software) producten op te leveren. Een sprint is dus een periode waarbinnen je een stukje werkende software of een werkend product maakt.

Voorbeeld van het balspel:

Elke poging om zoveel mogelijk balletjes door te geven is een iteratie of een sprint.

In software ontwikkeling is dit het ontwikkelen van een eerste stukje software.

Sprint review: Een vergadering of afspraak met het team waarbinnen je samen bekijkt welke sprintdoelstellingen gehaald zijn (en welke niet). Dit gebeurt op het einde van elke sprint.

Voorbeeld van het balspel:

Na elke poging werd er gekeken hoeveel balletjes er doorgegeven zijn.

In software ontwikkeling: na een sprint kijken wat er gehaald is.

Sprint retrospective

In een sprint retrospective bespreek je de leerpunten: wat hebben we als team goed gedaan, wat hebben we geleerd, wat moeten we in de toekomst anders doen?

Dit is anders dan een sprint review. In een sprint review kijk je naar WAT je gehaald hebt, in een sprint retrospective blik je terug op HOE je het hebt gehaald en wat je daarbij beter kan doen.

Voorbeeld van het balspel:

Na elke poging besprak het team hoe ze het beter konden. Hoe kon je je manier van werken aanpassen opdat je nog betere resultaten zou behalen?

In software ontwikkeling: leren uit fouten, of hoe iets sneller kan of beter kan en dat bespreken als een stukje klaar is, zodat je ermee rekening kan houden in het vervolg.

Daily standup

Een dagelijkse afspraak met het team waarbij iedereen vertelt

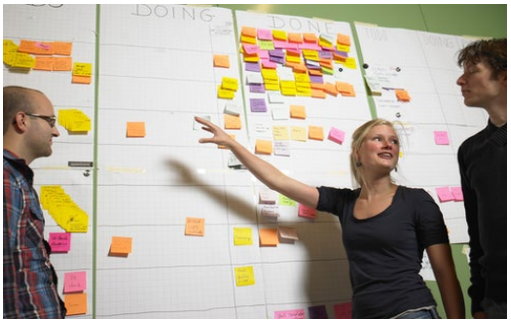
- wat hij/zij heeft gedaan (sinds de vorige daily standup)
- wat je vandaag nog gaat doen (en ev. de volgende dagen)
- wat je uitdagingen zijn hierbij, welke obstakels verwacht worden en hoe het team je daarbij kan helpen

De daily standup wordt georganiseerd door een scrum master. Hij/zij hoeft zelf niet verplicht aanwezig te zijn maar zorgt ervoor dat die daily standups plaatsvinden (en op een efficiënte manier verlopen).

Enkele tips hierbij:

- Je houdt een daily standup kort.
- Begin niet uit te wijden over problemen maar kaart ze aan en maak een afspraak om ze op een ander moment te bespreken met de juiste personen.
- Vertel het aan mekaar, aan alle teamleden, niet aan de scrummaster

Visueel management



Dit is ook heel typisch bij scrum. Taken worden visueel gemaakt, met post-its (op papier of electronisch). De klant, het management, het team, de product owner en scrum master (rollen waarover je meer leert in de verdieping) hebben allemaal toegang tot een scrumbord met de taken die opgenomen moeten worden (**todo**), waarmee iedereen iedereen bezig is (**ongoing),** en die klaar zijn (**done)**.

Visueel management is een krachtige tool bij het toepassen van scrum. Je ziet in één oogopslag hoeveel taken er al klaar zijn of nog lopende zijn. Men heeft ook gemerkt dat het voldoening geeft aan teamleden als ze in een daily standup een taak kunnen verhangen als die klaar is (van “ongoing” naar “done”). Teamleden hebben zo ook een heel visuele manier om aan te duiden waarmee ze klaar zijn. Je ziet als team dat er resultaat en vooruitgang wordt geboekt, en wat er nog moet gebeuren om alles klaar te krijgen.

Product backlog

De product backlog geeft een overzicht wat je allemaal nog moet realiseren. Het is een lijst van geprioritiseerde punten die moeten ontwikkeld worden om het product te maken en op te leveren.

De punten staan in volgorde van prioriteit. In voorbereiding van een sprint wordt er altijd afgesproken welke punten in een sprint opgenomen zullen worden.

Voorbeeld : consultatie van je lessenroosters

1. Als student wil je je rooster kunnen inzien zodat je weet welke lessen je de komende tijd hebt
2. Als lector wil je je rooster kunnen inzien zodat je weet welk vak je wanneer geeft aan welke klassen.
3. Als student wil je zien van welke lectoren je les hebt.
4. Als student en als lector wil je je lessenrooster kunnen integreren in je Outlook kalender.

Sprint backlog

Dit is de verzameling van taken die moeten gerealiseerd worden in 1 sprint. Dit wordt aan het begin van een sprint afgesproken in een sprintplanningsmeeting.

In het bovenstaande voorbeeld zou het kunnen dat je enkel punten 1 en 2 opneemt in de eerste sprint. Je maakt dan voor die 2 punten gedetailleerde taken die er moeten gebeuren.

Bijvoorbeeld

Bij het eerste punt uit de product backlog:

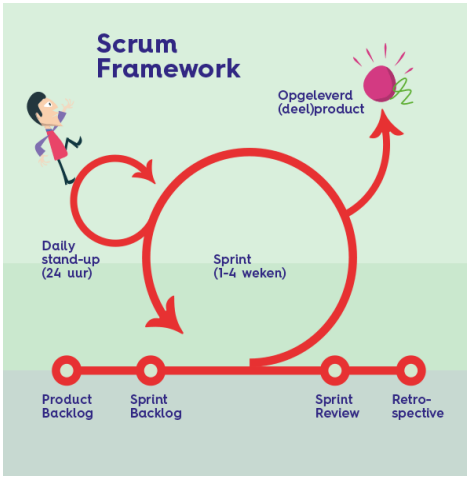
1.1. Als student moet je kunnen aanloggen op de applicatie
1.2. Het rooster moet getoond worden in een gebruiksvriendelijke kalender
1.3. Je kan de kalender bekijken per dag, week of maand.
1.4. Je moet kunnen filteren per les, per student of per klas.

Misschien zijn er nog andere taken bv. Als de ontwikkelsoftware nog moet geïnstalleerd worden. Of misschien een infrastructuurtaak dat er nog een development computer moet geïnstalleerd worden met Windows en een programmeersoftware voor de programmeur.

Het tweede punt uit de product backlog kan verdeeld worden in volgende programmeertaken:

2.1. Als lector wil je kunnen aanloggen op de applicatie
2.2. Als lector wil je het rooster kunnen raadplegen in een gebruiksvriendelijke kalender.
2.3. In het rooster staan naam van het vak, klaslokaal en klasgroep vermeld.

Het volgende schema vat de cyclus in scrum samen om een product op te leveren:



Het product dat je wil opleveren wordt in een product backlog beschreven (verschillende “puntjes” (dat noemen we eigenlijk *product backlog items*)

Het (software) product wordt in verschillende iteraties gemaakt. Elke iteratie is een sprint en duurt 1 à 4 weken.

Voor elke sprint neem je enkele punten uit de product backlog die je gaat realiseren in de sprint met het team. Zo maak je een sprint backlog door die punten nog eens uit te splitsen in taken. De taken worden verdeeld onder de teamleden (dat doen ze zelf). Elke dag in een daily standup wordt overlopen hoe ver ze staan (wat is er klaar, waarmee is men nog bezig, waar gaat men aan werken, wat zijn issues om er te geraken).

Op het einde van de sprint houdt het team een sprint review en kijken ze naar wat er gerealiseerd is van de sprint doelstellingen. Ook vindt een retrospective plaats waarin de leerpunten worden besproken.

Na elke sprint kan een deelproduct klaar zijn; na enkele sprints zal het product opgeleverd kunnen worden.

Tot zover de introductie tot scrum (en vergelijking met waterval) en enkele begrippen in scrum.

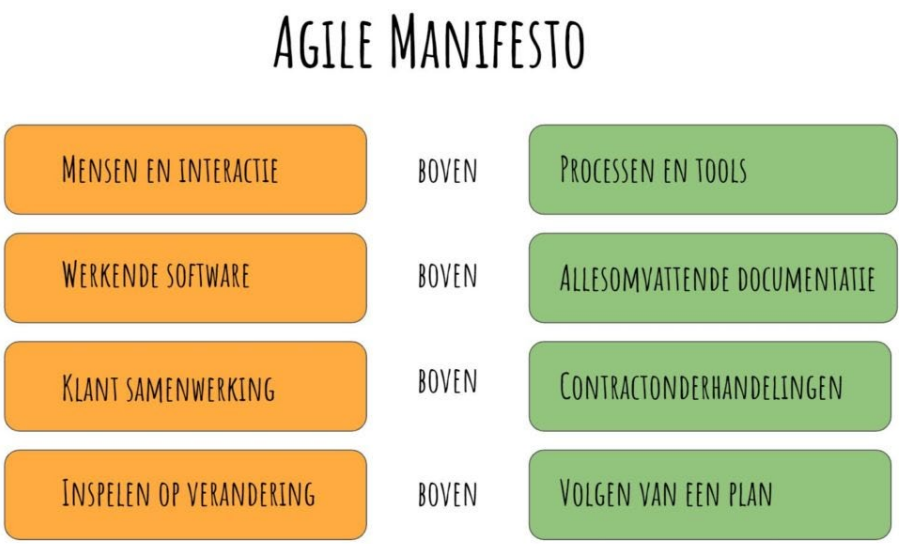
In een tweede deel zullen we scrum nog verder verdieping door in te gaan op de voor- en nadelen, de rollen in scrum en nog meer.

3. Agile en scrum: principes

<u>Merk op</u>: IT-Organisatie 2 (verdieping agile/scrum) start vanaf dit hoofdstuk. Je zal wel best agile/scrum in zijn geheel kennen op het einde van het jaar. Jullie zullen het ook volop toepassen bij Werkplekleren.

Agile werken bestaat al sinds het einde van de 20ste eeuw / begin van de 21ste eeuw. In ie eerste jaren dat Agile ontstond werd een "Manifesto for Agile Software Development" opgesteld door 17 developers. Dit bevat de belangrijkste principes van Agile werken. De developers wilden hiermee laten zien dat er ook andere manieren zijn om software te ontwikkelen dan de vroegere typische watervalmethode, nl. door meer op een flexibele en wendbare manier software te ontwikkelen en vlugger in praktijk te tonen dat iets werkt en zo sneller anderen te helpen.

De belangrijkste principes werden samengevat in dit "Agile Manifesto".



In dit manifesto wordt gesteld dat we het volgende verkiezen:

- Mensen en hun onderlinge interacties boven processen en tools
- Werkende software boven allesomvattende documentatie
- Samenwerking met de klant boven onderhandelingen over contracten
- Inspelen op verandering boven het volgen van een plan

Dit betekent het volgende:

- Business, ontwikkelaars en andere profielen die een belangrijke rol vervullen in het project moeten dagelijks intensief samenwerken. Die nauwe samenwerking is belangrijker dan het gebruiken van uitgebreide processen en tools voor samenwerking. Goed en zeer regelmatig met mekaar communiceren is belangrijk.
- De klant tevreden stellen is de hoogste prioriteit. Hiervoor wil men vroegtijdig en voortdurend waardevolle software opleveren. De bedoeling is om regelmatig werkende software op te leveren (in scrum: per sprint) en dit na korte periodes (enkele weken of

maximum enkele maanden). Je kan beter starten van een eerste werkend product dat al waarde levert voor de klant (het MVP = Minimal Viable Product) en dat in volgende iteraties uitbreiden (rekening houdend met feedback).

- Een nauwe samenwerking met de klant waarbij je tussen iteraties flexibel inspeelt op nieuwe behoeften, is belangrijker dan zware contracten met de klant waarbij je aan vroeger gemaakte afspraken blijft vasthouden.
- Tussen iteraties (in scrum: sprints) door, laat je de mogelijkheid om in te spelen op veranderingen en nieuwe prioriteiten. Daarmee rekening houden is belangrijker dan je strict aan initieel afgesproken planningen vasthouden.

4. Voor- en nadelen van agile/scrum t.o.v. waterval

We nemen als voorbeeldproject het ontwikkelen van een webapplicatie voor een bibliotheek. De webapplicatie moet het mogelijk maken om als lid van de bibliotheek aan te loggen, boeken op te zoeken, te reserveren, je uitgeleende boeken te bekijken, de terugbrengdatum te raadplegen en je uitleenperiode te verlengen.

In een **watervalmethode** doorloop je fase na fase in het ontwikkelproces:

- In een **analysefase** **o**nderzoek je wat de behoeften zijn van de klant en schrijf je de requirements neer. WAT wil men via de website mogelijk maken? Zaken zoals het vinden van boeken, reserveringen kunnen maken, uitleningen kunnen verlengen enz.. Deze requirements in kaart brengen is een taak die thuishoort in de analysefase.
- Een volgende fase kan gestart worden als de analyse klaar is. In deze fase wordt gewerkt aan een design of **ontwerp**. IT analisten of software engineers werken verder uit hoe de applicatie opgebouwd zal worden, welke gegevens moeten bewaard worden en hoe die gestructureerd zullen bijgehouden worden. Websdesigners gaan aan de slag om een idee te geven hoe de webapplicatie eruit zal zien.
- Vervolgens kan de webapplicatie gebouwd worden. Dit gebeurt in de **implementatiefase**. De programma's met schermen om de opzoekingen te doen of boeken te tonen enz, worden geschreven door de developers.
- Als de implementatie klaar is, gebeuren **testen** om te kijken of de applicatie foutloos werkt. Als de eerste technische fouten eruit gehaald zijn, wordt het resultaat met de klant ook overlopen of wordt die betrokken bij de testen zelf.
- Hebben de testen voldoende kwaliteit van de applicatie aangetoond, dan kan de applicatie gebruikt worden en gaat ze "live". Het **onderhoud** wordt afgesproken met IT.

Je ziet dat elke fase sequentieel verloopt en mekaar opvolgt. Dit is typisch voor een watervalmethode. In principe kan een volgende fase maar starten als de vorige klaar is (al zal er in praktijk soms wel wat overlapping zijn). Toch is het zo dat de klant niet zo veel betrokken wordt: voornamelijk in de beginfase om de requirements op te stellen, en tijdens of na de testfase.

In **agile/scrum** is die betrokkenheid van de klant veel hoger en frequenter.

Je probeert zo snel mogelijk een eerste werkend product op te leveren waar misschien nog niet alles in zit wat nodig is. In dit voorbeeld zou men er bv. voor kunnen kiezen om in een eerste

iteratie/sprint een eerste deel van de website klaar te maken die het mogelijk maakt om boeken op te kunnen zoeken die in de bibliotheek ter beschikking zijn, niet meer dan dat. Dit is al iets met waarde voor de gebruiker maar de website (het volledig gewenste eindproduct) is dan nog niet klaar.

Is die eerste fase klaar (opzoekingen van boeken uit de bib), dan wordt het resultaat al getoond aan de eindgebruiker en feedback gevraagd. Die feedback zal in een volgende sprint in rekening gebracht worden, maar ondertussen kan de eindgebruiker al starten met het gebruik van een eerste deel van de webapplicatie van de bibliotheek.

In een volgende sprint zou de website uitgebreid kunnen worden met een functionaliteit om boeken te kunnen verlengen.

Zo wordt de webapplicatie iteratie na iteratie (in scrum: sprint na sprint) verder uitgebreid. Elke iteratie bevat een afgewerkt nieuwe functionaliteit (feature) die waarde biedt voor de klant.

Denk over een voorbeeld na en bespreek met mekaar wat jullie zien als voordelen en nadelen van agile/scrum t.o.v. waterval.

=>

Voor- en nadelen van agile/scrum:

Voordelen:

- Het projectteam krijgt eerder feedback (over een tussenresultaat). Zo kan je sneller reageren op verandering.
- Je bouwt het product iteratie na iteratie op, met voortschrijdend inzicht.
- Doordat je het product sneller gebruikt, is het sneller duidelijk wat werkt en niet werkt.
- Je betrekt de klant sneller, die dus meer betrokken is bij de ontwikkeling van het product.
- Er is veel sneller en meer communicatie zowel in het team als met de klant.
- Het team leert van mekaar. Ze werken heel nauw samen.
- Je realiseert eerst de belangrijkste prioriteiten van de business (geprioritiseerde backlog).

Nadelen:

- Het team is als geheel verantwoordelijk en zelfsturend, waardoor het minder duidelijk is wat iemands individuele bijdrage is.
- Er is minder structuur en minder documentatie; bij grote projecten kan dat een probleem vormen. Het aanvoelen van hoeveel structuur en documentatie nodig is, is de kunst.
- De cultuur in het bedrijf moet openstaan voor het gebruik van agile en scrum. Als een bedrijf blijft denken zoals in de watervalmethode, dan kan dit tot problemen leiden.
- Een opdrachtgever is nauw(er) betrokken, wat goed is om snel feedback te vragen en krijgen maar het vraagt dan ook meer tijd van de opdrachtgever.

Voor- en nadelen van de watervalmethode:

Voordelen:

- Je start met een analyse en rondt die helemaal af, je hebt dus uitgebreid nagedacht over de gewenste functionaliteiten en de aanpak.
- Je kan meer stilstaan bij wat een goede en duurzame oplossing is.
- Je hebt goede documentatie, dat betekent dat nieuwe teamleden zich gemakkelijker kunnen inwerken, of dat je een goede basis hebt om aan andere betrokken in het team door te geven (bv. van analisten naar developers), of voor de gebruikersdocumentatie.
- De scope wordt in het begin vastgezet. Het is de bedoeling dat die zo weinig mogelijk wijzigt zodat het project binnen scope, budget en timing opgeleverd kan worden.
- Voor de afgesproken scope wordt een planning vastgelegd. Omdat de scope vastligt, kan men een goede planning maken met de fases van implementatie en de mijlpalen.

Nadelen:

- Een project in watervalmethode is niet zo flexibel als er iets moet aangepast worden. Er moeten change management procedures (met goedkeuringsprocedures) gevolgd worden.
- Meestal hebben projecten die met de watervalmethode gebeuren een langere doorlooptijd. Ondertussen kan de wereld veranderd zijn. Er is dus meer kans dat je eindproduct niet (meer) aan de verwachting van de klant voldoet.
- Doordat teams minder nauw samenwerken, is er meer kans op misverstanden tussen fases of teams. Er is minder communicatie.
- Doordat een fase maar kan starten als de vorige fase (grotendeels) afgerond is, heb je meer risico dat mensen moeten wachten op anderen uit het team.

5. Rollen in scrum

In scrum maken we onderscheid tussen 3 belangrijke rollen.

De product owner:

Dit is de persoon die de klant vertegenwoordigt naar het team toe. Het kan ook de klant zelf zijn.

De product owner bepaalt de visie van het product en kent de prioriteiten om dit product te realiseren.

Hij is in nauw contact met het scrumteam en motiveert het team om de prioriteiten te realiseren.

Praktisch gezien worden de behoeften weergegeven in de product backlog. Om de prioriteiten en de backlog uit te leggen aan het scrumteam wordt elke sprint een sprintplanningsmeeting georganiseerd door de product owner. Hij organiseert ook de demo op het einde van elke sprint, waar hij zelf, en/of klanten die hij vertegenwoordigt, feedback geeft.

De scrum master:

De scrum master is geen echte “leider” maar eerder een “facilitator” van het team. Het team is zelfsturend.

De scrummaster heeft wel de verantwoordelijkheid om ervoor te zorgen dat het scrumproces goed verloopt. Hij kent alle scrum principes en moet ervoor zorgen dat ze toegepast worden.

Hij organiseert de daily standups, de sprint reviews en retrospectives.

De scrum master zorgt ervoor dat het team focus kan houden op wat ze in de sprint moeten realiseren. Als er storende elementen zijn voor het team (men spreekt van “impediments”), moet de scrummaster deze wegnemen en zo snel mogelijk oplossen (bv. het ontbreken van licenties voor de developers).

De scrum master bewaakt de performantie van het team en coacht het team om productiever te werken.

Het scrumteam:

Een scrumteam heeft een aantal typische eigenschappen. Om goed te kunnen functioneren is het niet te groot, tussen 5 en max. 9 personen is ideaal. Het team is multidisciplinair samengesteld: verschillende profielen in het team zijn samengenomen zodat het team end-to-end kan werken en een werkend product kan opleveren. Mensen met verschillende kennis werken samen om het product op te kunnen leveren.

De teamleden zorgen ervoor dat het product voldoet aan de wensen van de klant en dat er op het einde van elke sprint een werkend product (bv. software en/of hardware) kan opgeleverd worden, dat nieuwe waarde levert voor de klant.

Heel belangrijk is dat het team zelfsturend is, het organiseert zichzelf. Het team doet al het werk om het product klaar te krijgen en verdeelt zelf het werk onder mekaar.

Als je dit vergelijkt met de watervalmethode in klassieke ontwikkeling, zijn er enkele belangrijke verschilpunten:

- Bij een klassieke ontwikkeling heb je altijd een projectmanager die het project leidt. De projectmanager maakt de planning en verdeelt zelf het werk onder de teamleden. Teamleden rapporteren aan hem. Bij een scrumteam is dat niet zo. Ze verdelen zelf het werk onder mekaar. Ze rapporteren niet aan de scrummaster maar werken nauw samen om de product backlog te realiseren. Ze gebruiken wel een aantal processen om zichzelf efficiënt te organiseren en steeds te verbeteren.
- Bij klassieke ontwikkeling bestaat een projectteam meestal uit mensen met eenzelfde profiel, of zijn er alleszins minder soorten profielen in het team. Meestal ben je nog van andere teams afhankelijk om je eindproduct op te kunnen leveren.

6. User stories

Om dieper in te gaan op hoe de product backlog aangepakt wordt, leggen we eerst het concept van “user stories” uit.

Een item op de product backlog is dikwijls geschreven in de vorm van een “user story”.

Een “user story” is een korte, eenvoudige beschrijving van een behoefte van een eindgebruiker of klant. Dit kan een “kort verhaaltje” zijn, geschreven vanuit het oogpunt van de eindgebruiker.

Een “user story” maakt duidelijk wat een eindgebruiker wil of nodig heeft en waarom dit nodig is. Het beschrijft de waarde voor de eindgebruiker (niet de oplossing).

Het wordt geschreven in dit formaat:

Formaat: “**Als** <wie>, **wil ik** <wat>, **zodat ik** <waarom>.”

Voorbeeld1:

ALS zorgverlener **WIL IK** inzicht in de medicatiehistoriek van de patiënt,

ZODAT ik bij twijfel de nieuw voorgeschreven dosering gemakkelijk kan verifiëren.

Voorbeeld2:

ALS klant van bol.com, WIL IK mijn vorige bestellingen online kunnen consulteren, zodat ik kan zien of ik alles ontvangen heb;

Als klant, **wil** ik zien wanneer mijn bestelling geleverd wordt, **zodat** ik weet of iemand thuis moet zijn.

Als bezoeker, **wil** ik producten zoeken op basis van populariteit, **zodat** ik mijn keuze kan maken op basis van wat anderen goede producten vinden.

Als medewerker van de klantendienst, **wil** ik een lijst van openstaande bestellingen zien op basis van het klantennummer, **zodat** ik de status kan meedelen aan de klant die mij belt.

Door de manier waarop het geschreven is (“ALS …") zie je dat men probeert duidelijk te maken wat de toegevoegde waarde is voor een gebruiker (in het voorbeeld hierboven de zorgverlener of de klant, bezoeker of medewerker van bol.com).

Een user story wordt normaal geschreven door een product owner en genoteerd op de product backlog. Alle user stories samen op de product backlog vormen de totale waarde van het project.

In de sprintplanningsmeeting aan het begin van de sprint neemt het scrumteam user stories uit de product backlog en splitsen ze deze op in taken en in te plannen in de sprint.

7. Werkinschattingen in scrum

De product backlog items, user stories, moeten ingeschat worden om een goed beeld te krijgen van hoeveel werk het is om ze te realiseren.

Het werk dat moet gedaan worden in een sprint, wordt ingepland tijdens een **Sprint planningsmeeting** aan het begin van een sprint. De scrummaster zorgt dat deze meeting plaatsvindt met het hele projectteam.

Het volgende wordt in dit overleg besproken:

- **Waarom is de komende sprint waardevol?**
- **WAT kan er in de aankomende sprint gedaan worden?**
    - Aan de hand van de product backlog : de eerstvolgende prioriteiten nemen.
    - Het team bespreekt welke taken ze in de komende sprint op kunnen nemen.
    - De product owner geeft extra uitleg bij items waar nodig.
    - Wat moet er af zijn op het einde van de sprint? Het sprintdoelt wordt vastgelegd.
- **HOE zal het geselecteerde werk uitgevoerd worden?**
    - De product owner is nu vaak vertrokken uit de bijeenkomst.
    - Er wordt in detail besproken hoe de items uitgevoerd worden en de sprint backlog wordt vastgelegd.
    - Voor elk item op de sprint backlog wordt een schatting gemaakt hoe lang men erover zal doen.

In een **klassieke ontwikkeling** wordt het **werk ingeschat** in **mandagen, manuren of Euro.**

Bijvoorbeeld: Als een project is ingeschat op 200 md (md = afkorting van mandagen), wil dat zeggen dat als er maar 1 persoon zou werken op het project, deze persoon 200 dagen werk heeft om het project af te ronden. Als er 4 personen zouden meewerken, hebben ze bijvoorbeeld elk 50 dagen werk. De 200 md werk kunnen ook anders verdeeld worden, bv. de projectleider besteedt 20 md, de analisten 50 md, de programmeurs 70 md, de testers 40 md, infrastructuur 20 md. Deze verdeling en inplanning gebeurt bij klassieke ontwikkeling door de projectleider.

Men spreekt ook wel van een “absolute inschatting”, omdat je met een cijfer direct de hoeveelheid werk weergeeft.

Als je rekent dat een IT-medewerker bv. 500 Eur per dag kost (dergelijke bedragen worden afgesproken in een bedrijf, en zijn niet het loon van die IT-medewerker maar bevatten ook kosten van bureau, materiaal, het gebouw e.d.), dan zou het project van 200 md, 100 000 Eur aan werk kosten. Als er nog kosten moeten gebeuren qua hardware, infrastructuur, of licenties bijvoorbeeld, zal de totale kost van het project nog hoger liggen.

Zo worden projectkosten bij klassieke ontwikkeling ingeschat.

In scrum daarentegen schat men de “**relatieve waardes**”, dat wil zeggen dat je ***vergelijkt met andere inschattingen om een inschatting te maken***.

Neem het voorbeeld van de gebouwen rondom PXL. Je kan gebouwen zien van verschillende hoogtes: kleine gebouwen zoals het gebouw van Smart ICT, middelmatig hoge gebouwen zoals de B-blok, en hoge gebouwen zoals het nieuwe G-gebouw.

Met inschattingen op de klassieke manier zou je die hoogtes uitdrukken in meters.

Met scrum druk je die hoogtes "relatief" uit.

Bv. de "T-shirt size" (S/M/L/XL) is één van de manieren waarop inschattingen gebeuren in scrum.

Met "relatieve" inschattingen neem je alvast één gebouw als referentie, bv. Het SmartICT gebouw, wat je bijvoorbeeld ziet als het "klein" gebouw", maat "S" (small). Het B-gebouw is ongeveer het dubbele, een medium size ("M"), en het G-gebouw is nog eens 2 of 3 keer zo hoog, maat "L". Deze maten zijn relatief (t.o.v. mekaar) aangegeven. Je brengt alle gebouwen dan bv. onder in de categorieën S/M/L zonder de details te geven hoeveel meter ze effectief hoog zijn.

Een ander voorbeeld van relatieve inschattingen die gebruikt worden in scrum, **zijn inschattingen met behulp van de "reeks van Fibonacci".**

Dit wordt veel gebruikt in scrum om user stories in te schatten. Hierbij wordt gebruik gemaakt van **storypoints** die gebaseerd zijn op een verdeling volgens de reeks van Fibonacci. De reeks van Fibonacci begint normaal met 0 en 1, en daarna is elk volgend element van de rij altijd de som van de twee vorige elementen. De eerste elementen van de rij zijn dan de volgende: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, … De rij begint soms met 1 en 1in scrum (om ze waardevol te kunnen gebruiken), of met ½ en 1.

De inschattingen in scrum worden door een team gemaakt.

Eén van de veelgebruikte manieren is "**planning poker**".

Bij planning poker krijgt elk teamlid een set kaarten met de waardes 0, ½, 1, 2, 3, 5, 8, 13, 20, 40 en 100. Ook een kaart met een ? en een kaart met een koffie icoontje voor "koffiepauze" horen erbij.

In de planningsmeeting wordt vervolgens een user story uit de product backlog genomen die moet ingeschat worden. Elk teamlid legt een kaart neer met een inschatting hoe groot of complex het volgens hem of haar is om deze user story te implementeren, t.o.v. de voorgaande items. Er zitten regelmatig grote verschillen in tussen wat de teamleden inschatten. De personen met de hoogste en de laatste inschatting lichten dan hun keuze toe. Zo krijgen anderen inzicht in de redenen waarom het niet zoveel werk of juist veel werk zou kunnen zijn en de activiteiten van de taak worden concreter. Daarna start een nieuwe ronde. Iedereen gooit terug een kaart op, rekening houdend met de nieuwe informatie die men heeft sinds de laatste bespreking.

Dit herhaalt zich tot het team erover eens is wat de inschatting wordt. Dikwijls is dit niet het gemiddelde, maar ligt het resultaat dicht bij één van de uitersten.

We doen een oefening met planning poker tijdens de les.

8. Uitdagingen en moeilijkheden in scrum

Uiteraard loopt het werken met scrum niet altijd van een leien dakje, zeker niet in het begin als je de omschakeling maakt van het werken met de klassieke methode naar scrum.

- Er is allereerst de leercurve om met scrum om te gaan, zowel voor het team als het management van de organisatie. Het team moet het hele proces leren en het management moet aanvaarden dat niet de volledige scope volgens een exact budget is ingeschat. Ook moeten ze aanvaarden dat tussentijdse producten zinvol kunnen zijn om al te releasen (en zo feedback te kunnen krijgen voor volgende sprints). Een bedrijf dat gewoon te werken is met een klassieke methode, verwacht dikwijls dat een project in totaliteit.

  Het team moet scrum leren kennen en toepassen.
  In het begin gaat een introductie van scrum gepaard met problemen, maar die moet je als team overwinnen, en zo gaat het met de tijd beter.

- Je moet als team groeien. Scrum werkt met zelfsturende teams. Een zelfsturend team ben je niet van vandaag op morgen. Het team moet op de juiste manier leren omgaan met de vrijheden, en ook het management moet aanvaarden dat het team zelf initiatief neemt in oplossing en werkverdeling.

- Als je een scrumteam bent dat een project volgens de scrum methode wil opleveren, maar de rest van de organisatie werkt niet met scrum, dan kan dat problemen geven. Als je iemand nodig hebt uit een ander team bijvoorbeeld, die op een andere manier werken (analyse/development/testing…), of iemand van infrastructuur die andere afspraken heeft over wanneer hij werk mag doen en wat zijn prioriteiten zijn, kan dat problemen geven.

- Managers moeten dikwijls ook leren of aanvaarden zich niet te mengen met waar het team zich mee bezig houdt. Ze moeten aanvaarden dat een team niet gestoord wordt tijdens de sprint en niet tussendoor met nieuwe “dringende” vragen komen. Als er nieuwe dringende requirements zijn, moeten die altijd via de product owner afgesproken worden. Het scrumteam moet “focus” kunnen houden tijdens een sprint.

- Een uitdaging met scrum is aandacht te hebben voor het grotere plaatje. Door alles op te splitsen in sprints heb je een focus op wat er op korte termijn klaar moet zijn, en minder op wat men wil bereiken op termijn. Als je ook denkt aan de lange termijn, ga je meer nadenken over herbruikbaarheid in je oplossingen. Je moet proberen ook aandacht te hebben voor het geheel van het project en waar je op termijn naartoe werkt.

9. Wat maakt scrum succesvol?

Er zijn verschillende zaken die maken dat scrum succesvol is, zowel op het vlak van samenwerking tussen **mensen**, de **organisatie** als het **proces**.

Wat **mensen** betreft, is het opmerkelijke dat je met de rol van de product owner iemand hebt die heel dicht bij de business staat. De product owner bespreekt de prioriteiten met de business en onderzoekt wat het meeste waarde heeft. Dat maakt scrum erg krachtig.

Het feit dat je met relatief kleine scrumteams werkt, zorgt dat er efficiënt kan samengewerkt en gecommuniceerd worden. Typisch werkt een scrumteam ook fysiek samen in 1 ruimte (al dan niet een “digitale ruimte” / virtueel).

Op het vlak van **organisatie** kan het heel goed werken als de organisatie meegaat in de scrum-gedachte. Scrum stimuleert dat er open, proactief en veel gecommuniceerd wordt. Er zijn zo minder misverstanden. De organisatie is nauw betrokken bij feedback momenten maar bijvoorbeeld ook als er iets live gaat en gevierd kan worden. Elke sprint kan er iets live gaan in scrum, dus dat creëert vele succesmomenten en vermijdt dat projecten te langdradig worden voor ze iets opleveren.

Tenslotte op het vlak van **proces** is het goed dat een project opgedeeld wordt in kleinere sprints. Na elke sprint wordt er tijd gemaakt om feedback te ontvangen (demo) en te leren (retrospectives). Als je die goed gebruikt, kan je meer waarde leveren voor de klant.

10. Scrum toepassingen

Scrum wordt zowel gebruikt binnen IT als buiten IT.

Binnen IT wordt het veel toegepast in software ontwikkeling, in het bijzonder bij teams die digitale ontwikkelingen doen bv. voor het maken van websites of apps. Ook in een samenwerking tussen IT development en IT infrastructuur heeft het gebruik van scrum zijn waarde. In teams die met DevOps werken, ziet men ook veel toepassingen van scrum.

Ook buiten IT wordt scrum veel gebruikt, bv. bij productontwikkelingen. Ook merken zoals Fitbit (smartwatches/stappentellers) en Lego hebben scrum toegepast bij het ontwikkelen van nieuwe producten.

Ook voor marketingcampagnes kan scrum nuttig zijn. Je kan een eerste campagne starten (na een eerste sprint) en direct feedback verwerken zodat een tweede campagne beter en doeltreffender wordt.

Een derde voorbeeld van het toepassen van scrum buiten IT is bij HR afdelingen. Met scrum kan je bekijken hoe je HR services (zoals loonadministratie, support voor management bij evaluatiegesprekken, opleiding, …) in verschillende sprints uitrolt in plaats van te wachten tot de hele HR afdeling georganiseerd hebt en pas dan de diensten openstelt naar de rest van de organisatie.

Je vindt op het Internet gemakkelijk nog verschillende toepassingen van scrum.

Bekijk dan zeker eens welke processen je herkent van scrum en het in het voorbeeldproject waarde levert via tussentijdse sprints.

11. Tools voor scrumteams

Er zijn enorm veel tools ter beschikking die je kan gebruiken in een project waarin je scrum wil toepassen. Vertrek altijd eerst van de functionaliteiten die je wil gebruiken als je een tool zoekt. Bv. zoek je een tool voor het maken van een product en sprint backlog, waar je ook een scrumbord in kan maken, wil je al dan niet ook rapportering van het werk enz.

We noteren hier enkele voorbeelden. Je zal in Werkplekleren ook met tools aan de slag gaan.

Scrumblr	Free shared whiteboard (postits planned/in progress/done)	http://scrumblr.ca/
TaskJunction	Plannen, inschatten en tracken van software devlp projecten met scrum	http://www.taskjunction.com/
Task Planner	Geïntegreerd in MS Teams	
Monday.com	Projectmanagement meer algemeen, ook scrum	
Trello	Intuïtief voor het gebruik van een scrumbord, Gebruikt in WPL2 SNE	
Azure DevOps	Geïntegreerd met Azure DevOps, handig voor ontwikkelaars. Gebruikt in WPL2 PRO	
Jira	Vooral gekend voor testing, incident en problem management, veel gebruikt in de bedrijfswereld. Gebruikt in WPL2 DVO.	

Je kan planning poker met kaarten spelen maar er zijn ook vele online varianten op het spel.

Voorbeelden:

http://www.planitpoker.com/	Zonder registratie	Interessante tool
https://scrumpoker.online/	Zonder registratie	
https://planningpokeronline.com/	Zonder registratie	Interessante tool
www.planningpoker.com	Met registratie	Minder handig doordat je moet registreren
https://marketplace.visualstudio.com/items?itemName=ms-devlabs.estimate en https://github.com/cschleiden/azure-boards-estimate	Azure Boards Estimate in Azure DevOps	
https://firepoker.io/	Firepoker	

Probeer zeker eens een nieuwe tool uit, toepassen helpt je om de leerstof beter te begrijpen.

12. Scrum versus…

Tot slot maken we nog een korte vergelijking tussen de scrum methode en andere methodes voor projecten. Er is bv. naast Scrum ook **Kanban**, eveneens een veel gebruikte methode om agile en iteratief iets te ontwikkelen.

Het grote verschil tussen Kanban en Scrum is dat je met **Kanban** een continue flow hebt van nieuwe taken die binnenkomen (en bij "todo" komen te staan) en afgewerkt geraken ("done"). Je werkt bij Kanban niet in sprints maar je neemt de hele tijd nieuw werk aan en probeert tussentijdse opleveringen te doen, met een demo, wanneer het past.

Wat doe je wel bij Kanban?

- Je visualiseert wat je kunt visualiseren, net als bij scrum. Het wordt als belangrijk aanzien om het werk te visualiseren, zodat je het gemakkelijk met het team kan bespreken.
- Je beperkt de hoeveelheid werk die je tegelijkertijd oppakt. Managers kunnen wel vragen om nieuwe dingen op te pakken. Dat werk wordt aangenomen zogauw er nieuwe capaciteit vrijkomt. Door minder dingen tegelijk op te pakken realiseer je een korte levertijd en betere kwaliteit, net als bij scrum. Je vergroot ook de wendbaarheid als prioriteiten wisselen. Je bent niet met honderd dingen tegelijk bezig die allemaal half af zijn.
- Je gaat ook regelmatig verbeteren en feedback integreren.

Bij Scrum zijn de sprints heel typisch, je blokt meer een stuk werk af en geeft het team alle focus geeft om dat te realiseren. Na de sprint heb je iets klaar om te releasen, geef je een demo, doe je een sprint review en retrospective. Dit alles gebeurt niet bij Kanban.

13. BIJLAGE: Woordenschat

- **Agile:** Agile / agility verwijst naar de "wendbaarheid" van een organisatie en de flexibiliteit om met veranderingen om te gaan.
- **Daily scrum:** Korte dagelijkse meeting met het team waar je bespreekt wat je gisteren hebt gedaan, wat je vandaag gaat doen en welke problemen je hebt.
- **Methodologie:** Een verzameling van methodes, een manier van werken die helemaal beschreven staat, procedures en werkwijzes die je kunnen helpen om een activiteit uit te voeren of om kennis te verwerven.
- **Multidisciplinair:** Verschillende disciplines zijn samengebracht in een team om samen een activiteit te voltooien, dat kunnen verschillende beroepen zijn of verschillende functies/rollen/profielen in een organisatie. Bv.
    - Een operatie in een ziekenhuis met een multidisciplinair team met daarin een uroloog, een darmspecialist en een gynaecoloog.
    - Een projectteam voor software ontwikkeling met een analist, java ontwikkelaar, database specialist, infrastructuur specialist en 2 testers.
- **Product backlog**: Alle functionaliteiten die moeten geïmplementeerd worden in het project.
- **Product Owner:** Vertegenwoordiger van de klant of gebruikers die de prioriteiten bepaalt.
- **Scrum:** Een methode, een framework om op een flexibele manier producten te maken. Er wordt gewerkt in multidisciplinaire teams die in korte sprints werkende producten

opleveren. Bij scrum hoort een vaste manier van werken die in scrumteams toegepast wordt.

- **Scrum Board**: Visuele voorstelling van alle sprint backlog items in enkele kolommen (Todo / Ongoing / Done).
- **Scrum Master**: Faciliteert het team met alles wat zij nodig hebben en waakt erover dat het scrum proces toegepast wordt.
- **Sprint**: Is een korte periode waarbinnen het team een stukje werkende software of een werkend product maakt.
- **Sprint Backlog**: Al het werk dat moet uitgevoerd worden tijdens één sprint.
- **Sprint Planning**: Selectie van items uit de Product Backlog die voorgesteld worden voor realisatie in een sprint, op basis van de prioriteiten van de klant/gebruikers.
- **Sprint Retrospective**: Leerpunten bespreken op het einde van een sprint: wat hebben we goed gedaan, wat hebben we geleerd, wat moeten we in de toekomst anders doen?
- **Sprint Review**: Op het einde van een sprint wordt er gekeken wat er gehaald is (realisaties).
- **User Story**: Korte, eenvoudige beschrijvingen van een product kenmerk, verteld vanuit het perspectief van de persoon die de nieuwe vaardigheid wenst.