AGILE PROJECTKOSTEN EN TIJDMANAGEMENT

**Bron**: <https://www.interactivated.nl/blog/agile-projectkosten-en-tijdmanagement/>

Agile-softwareontwikkeling gaat over het leveren van waarde voor de klant. Doordat al het werk is gericht op de waarde voor de klant en een werkend product vanaf de start, is het van nature efficiënter en minder verspillend. Een goed lopende agile ontwikkelingsmethodiek is behendig omdat het zich aanpast aan en reageert op veranderingen. Als gelden zijn uitgeput of voorwaarden grote veranderingen ondergaan, zal de klant nog steeds zonder werkend product zitten. Agile ontwikkeling is zo logisch dat het zelfs breder wordt toegepast dan softwareontwikkeling.

Geweldig, maar betekent dit dat de kosten haar schatting, waar softwareprojecten om  bekend staan, niet gaan overschrijden? Hoe beheer je de kosten en schat je deze in?

Een bekend flauw antwoord is: Agile software projecten hebben een budget nodig, geen schatting. Budget vervangen voor schatting zou dingen zo logischer en het leven zo veel makkelijker maken. Helaas hebben dingen een manier om nog gecompliceerder te worden dan dat ze lijken, en we staan op het punt om ons hier in te verdiepen.

**Hebben alle Agile Projecten Een Schatting Nodig?**

Het antwoord op deze vraag is een andere vraag: Waarom denkt de klant dat een schatting gepast is?

Als het antwoord voornamelijk is om de ontwikkelaar zich te laten houden aan een toewijding tot het bedrijf of als de ontwikkelaar niet aan de verwachtingen voldeed, dan zijn we geneigd nee te zeggen. Aan de andere kant, als het maken van grote keuzes of plannen op een schatting gebaseerd zijn, moeten we ons er in verdiepen.

Je zult soms zien dat een project zo dringend is dat, zolang de klant en de leverancier tot een akkoord kunnen komen, het op de juiste manier moet worden uitgewerkt, zonder schatting. Een project in de ontdekkende fase heeft slechts een ruwe schatting nodig voor de klant om de benodigde gelden toe te wijzen, of om te bepalen of ze het kunnen betalen of kunnen wachten tot voltooiing.

Het hele doel van agile ontwikkeling is niet om vast te zitten aan de oorspronkelijke verwachtingen, die wellicht niet eens haalbaar zijn, dus de meeste schattingen zijn van nature aan de ruwe kant. Hoe dan ook zijn er altijd tijden wanneer we een meer stabiele schatting moeten ontwerpen.

**De 5 Stappen van Schatting en Planning**

Schatting en planning zijn grondig met elkaar verbonden in agile ontwikkeling. Middelen zijn beschikbaar om de efficiëntie van het proces bij te houden. Een ander kenmerk van agile ontwikkeling is dat de schatting zo vaak kan worden geüpdatet als mogelijk, dus het is geen probleem als zelfs de details van een project aan het begin onbekend zijn.

We zullen het Scrum raamwerk van agile-softwareontwikkeling vanaf het begin gebruiken. Er zijn 3 rollen in een Scrum project: Producteigenaar, ScrumMaster, en Team. Een backlog is een lijst van kenmerken en een backlog item is een kenmerk. Een backlog item wordt opgenomen als verhaal genaamd een gebruikersverhaal, dat verder kan worden afgebroken in verhaalpunten, waarvan de som de hoeveelheid moeite benodigd voor het afronden van het gebruikersverhaal van een kenmerk vertegenwoordigt. Een sprint is een herhaling. De rest zal worden geïntroduceerd terwijl we bezig zijn. (De Scrum nomenclatuur is bruikbaar buiten haar elegantie om. Waarom geen sprint in plaats van een herhaling die mensen in slaap brengt? Wil je liever worden toegewezen tot een verhaal of werk?

**1) Maak Backlog Aan:**

Een backlog van een product is een lijst met alle productkenmerken. Eerst breken we het aanvankelijke idee in kleinere kenmerken of backlog items. Het aanvankelijke idee kan abstract zijn, maar backlog items moeten te definiëren zijn. Deze backlog items worden opgenomen en verteld in gebruikersverhalen. Een agile gebruikersverhaal beschrijft het kenmerk tot in de details, die constant kunnen worden aangepast. De backlog van het product wordt vooraf gepland, hoewel het verandert naarmate een gebruikersverhaal wordt gewijzigd, toegevoegd, of verwijderd. Een backlog kan tientallen of zelfs honderden gebruikersverhalen bevatten.

**2) Prioriteer Backlog Items:**

Sorteer gebruikersverhalen om de backlog op waarde van prioriteit te houden (te worden uitgevoerd door de Producteigenaar). Om het principe van vroege waardevoorziening te behouden, begint Agile-Softwareontwikkeling altijd met items met hoogste prioriteit.

Bijvoorbeeld, om een Magento eCommerce site te bouwen zijn inventarisbeheer en een winkelwagentje hogere prioriteit van reviews en blogs. De site zou nog functioneel zijn, zelfs als het project niet tot reviews en blogs kwam.

**3) Schat de Omvang van het Project:**

Schat de omvang van het project door het totaal aantal verhaalpunten op te tellen. Een Scrum Team van ontwikkelaars wijst het aantal verhaalpunten toe aan ieder gebruikersverhaal en het grote totaal aantal verhaalpunten is een beoordeling van de omvang en complexiteit van het project (hoger is complexer).

Iedere gebruiker wordt een Fibonacci getal toegewezen, laten we zeggen tussen 1 en 21. Het getal in opnemende volgorde weerspiegelt de complexiteit van een gebruikersverhaal, en dus de relatieve hoeveelheid moeite die nodig is om te voltooien. Dit getal zou de verhaalpunten van een genoemd gebruikersverhaal worden. Let op dat ieder verhaalpunt van 8 niet betekent dat het verhaal 8 uur (of zoiets als dat) in een sprint nodig heeft – het is slechts een voorlopig aantal verhaalpunten, en een populaire manier om deze voorlopige schatting te bepalen is Scrum Poker.

Scrum poker schijnt de cognitieve vooroordelen van verankering (teveel afhankelijkheid van het eerste stukje informatie) en het Dunning-Kruger Effect (een illusie van grootsheid) verminderen.

**4) Schat de Benodigde Projecttijd**

Deze schatting is geen precieze voltooiingdatum. Begin met een routekaart en releaseplan van het product. De routekaart van een agile ontwikkeling weergeeft de evolutie van het product door een aantal grote releases. Het releaseplan is een voorlopig schema van grote releases gebaseerd op de backlog van het product.

De benodigde projecttijd wordt geschat voor iedere sprint. Het wordt gemeten in snelheid, wat bij Scrum het aantal voltooide gebruikersverhalen in iedere sprint is. Een goede richtlijn voor het voorspellen van de voorlopige snelheid kan worden gevonden in de [blog](https://www.mountaingoatsoftware.com/blog/predicting-velocity-when-teams-change-frequently) van Mike Cohn. De schatting van de benodigde tijd is een iteratief proces doordat de snelheid fluctueert voordat het na enkele sprints stabiliseert.

De incrementele aard van agile ontwikkeling betekent dat het product in gedeelten kan worden geleverd, ieder stuk als werkend en te vervoeren product als het nog niet compleet is. Hoeveel en welke functionaliteiten je dient te releasen kan worden beheerd en gecontroleerd op andere manieren dan deadlines.

**5) Schat de Projectkosten**

Het berekenen van de schatting van projectkosten is logischer zodra de omvang en de snelheid van het project bekend zijn. Er zijn twee standaard benaderingen: via gemengd tarief of specifieke arbeidscategorieën. Controleer dit [stuk](https://www.leadingagile.com/2012/11/calculate-budgets-agile-team/) van Jesse Fewell omdat de details hier verder dan omvang gaan. Beide methoden incorporeren Team leden en sprints zoals genoemd in het releaseplan.

Een snellere schatting kan worden bereikt door het teamverbrandingspercentage per sprint te berekenen en ervan uit te gaan dat het gemaakt is. Het totaal aantal benodigde sprints kan worden berekend door het aantal gebruikersverhalen te delen door de snelheid.

De meeste dienstbedrijven hebben een blad met uurtarieven voor iedere arbeidscategorie. Ieder teamlids (inclusief Producteigenaar en ScrumMaster) verbrandingspercentage per sprint is: Uurtarief x Werkuren/Dag van # Dagen in een Sprint.

En het verbrandingspercentage per sprint van het team is de som van het verbrandingspercentage per sprint van ieder lid, zoals hierboven berekend.

Hoewel gezien als afgerond, kan het geschatte teamverbrandingspercentage per sprint ook constant wijzigen, doordat de eigenlijke snelheid bekend is na afronding van iedere sprint.

**Bijhouden van Workflow Efficiëntie**

Agile-softwareontwikkeling is inherent aan meer efficiëntie. Daarnaast zijn er kant-en-klare workflow efficiëntie oplossingen ingebouwd in Scrum en andere agile ontwikkelraamwerken. Aan het einde van de dag zijn er echter inzichten en geschikt management nodig om absoluut zeker te zijn dat middelen niet op enig moment tijdens de ontwikkeling verspild worden.

De aanwezigheid van positieve synergie en de gelijktijdige afwezigheid van negatieve omstandigheden zijn bepalend voor een efficiënte workflow. Positieve synergie bereiken houdt in dat je een omgeving creëert waarin iedere individuele inspanningen bedoeld zijn voor maximale impact. Negatieve omstandigheden in softwareontwikkeling zijn (maar zijn niet beperkt tot) het tegenkomen van knelpunten en het opstapelen van schulden.

Veel activiteiten in agile ontwikkeling kunnen worden geautomatiseerd met het ulitieme doel van workflow efficiëntie. Een bekende en bewezen middel is JIRA Software, een aan te passen tool voor agile projectbeheer met ondersteuning van alle agile raamwerken, waaronder Scrum en Kanban en hun populaire variaties.

**Burndown Grafiek**

Een burndown grafiek houdt de overblijvende nog in een sprint af te ronden taken bij. Het is een grafiek met de geplande hoeveelheid werk (gebruikersverhalen, verhaalpunten, of hun equivalent in dagen) op de y-as en de tijdlijn van de sprint op de x-as. De plot begint op het geplande aantal gebruikersverhalen in een sprint en een “burndown” tot nul, hopelijk op of voor de laatste dag van de sprintcyclus.

Twee lijnen of krommen worden geplot op de grafiek. Één is de geprojecteerde of ideale burndown tarief en de andere is het daadwerkelijke burndown tarief geplot aan het einde van de dag.

De burndown grafiek van een sprint houdt de progressie bij en geeft aan of het team ewl of niet op schema ligt om het geplande werk van de sprint af te ronden. Teamleden weten wanneer ze moeten compenseren door een oog te houden op de burndown grafiek van een sprint.

De burndown grafiek kan worden verlengd naar het hele project door de x-as naar de verwachte releasedatum en de y-as naar het geplande aantal gebruikersverhalen of verhaalpunten voor het project te verlengen. Dit staat bekend als de release burndown grafiek, in tegenstelling  tot de sprint burndown grafiek.

**Earned Value Management (EVM)**

Earned Value Management (EVM) is uitgegroeid tot een extreme populaire method om agile projectprestaties te meten sinds het werd geïntroduceerd tot agile raamwerken in 2006. EVM wordt het meest toegepast op het releaseniveau in plaats van individuele sprints.

De diverse metrische waarden van EVM worden berekend van het geplande aantal gebruikersverhalen en het aantal dat daadwerkelijk is afgerond. Nog specifieker kijken we naar slechts 3 waarden: daadwerkelijke kosten (AC), verdiende waarde (EV) en geplande waarde (PV). AC zijn de daadwerkelijke kosten van het uitgevoerde werk, EV zijn de gebudgetteerde kosten van het uitgevoerde werk en PV zijn de gebudgetteerde kosten van het geplande werk.

AC, EV en PV kunnen worden geïntegreerd in de release burndown grafiek om de efficiëntie van een agile project op welk moment dan ook te illustreren. De totale kosten en duur van het project zoals voorspeld door de burndown grafiek met EVM bieden de klant en leverancier waardevolle tijd om weloverwogen keuzes te maken of de nodige of gewenste veranderingen.

EV ie een belangrijke waarde van kostenefficiëntie maar vaker dan niet staat het niet gelijk aan AC en PV. Dit wordt veroorzaakt door efficiëntie-gerelateerde zaken zoals de workflow inefficiëntie en het fout inschatten van prioriteiten en schattingen, of door andere variabelen zoals aanvankelijke stijgingen en wijzigingen in benodigdheden en prioriteiten.

EVM data is van nature historisch, actueel en voorspellend. Het leidt tot het gebruik van historische data voor iteratieve schattingen, diagnoses van actuele data in het tracken, en de voorspelling van toekomstige data voor tijdelijke keuzes om klantenwaarde te optimaliseren.

De EVM data duiden op de aan- of afwezigheid van inefficiëntie en niet veel meer. Het is aan het Team om de oorzaak van de inefficiëntie te vinden. Zou een ScrumMaster zijn of haar reputatie op het spel zetten om de echte oorzaak te vinden?

**Echte Waarden van Agile Ontwikkeling**

Agile-softwareontwikkeling levert haar klantenwaarde snel en vaak. Na slechts enkele sprints zou een agile software product klaar moeten zijn voor vroege release met haar core features op zijn plaats en zullen haar uiteindelijke gebruikers niet veel beter weten.

Agile ontwikkeling bloeit op veranderingen dus gedetailleerde planning behoort tot het verleden. Het beste is om de ontwikkeling te beginnen met een voorlopige volgorde van prioriteiten. Houd de progressie stap voor stap bij om veranderingen te bedenken en geïnformeerde keuzes te maken.

By interactivated • on May 3, 2017