**Kostenbeheersing in Scrum: kiezen tussen scope en budget**

**Bron**: <https://ambrero.nl/kennis/zekerheid-bij-scrum-kiezen-tussen-scope-en-budget>

**Als er iets is dat ik de afgelopen 17 jaar als ontwikkelaar heb geleerd, is het dat er niets zo moeilijk is als het begroten van grote software projecten. Ik kan mij nog heel goed één van de eerste projecten van Ambrero herinneren, inmiddels zo’n 15 jaar geleden. We zouden een EPD (Electronisch Patienten Dossier) ontwikkelen voor Stichting CASA. Het exacte getal weet ik niet meer, maar ik kan me herinneren dat we voor de eerste fase een uur of 100 hebben geoffreerd. Uiteraard is het project vele malen groter geworden dan dat. Maar blijkbaar hebben we het goed gedaan, want CASA is lang klant gebleven. In deze blog deel ik mijn ervaring met kostenbeheersing in [Scrum](https://ambrero.nl/kennis/scrum)Scrum is een iteratief raamwerk, veel gebruikt voor de coördinatie van softwareontwikkeling. met je.**

Inmiddels zijn we een stuk beter geworden in het inschatten van werk, maar het blijft een lastig onderdeel van ons vak. Bij maatwerkprojecten ligt er geen draaiboek klaar met exacte specificaties, dus de betrokken personen, inclusief de klant, doen een hoop aannames over de te ontwikkelen functionaliteit.

## Waterval ontwikkelmethode voor meer zekerheid?

Ik hoor vaak de vergelijking met de bouwsector: een aannemer maakt toch ook een exacte prijscalculatie voor een bouwproject? Waarom is dat voor software dan zo moeilijk? Die aannemer doet dat op basis van standaardproducten (materiaal) en specificaties die vaak al vaststaan (bouwtekeningen). Wil je dit doortrekken naar softwareontwikkeling dan zal je vooraf exact moeten definiëren wat je wilt hebben en een specialist moeten inhuren om een specificatie te schrijven. Het resultaat is een hoop tijd en kosten om tot die specificatie te komen, maar daarna wel meer zekerheid over doorlooptijd en budget.

Ik zeg meer zekerheid, want uitgewerkte specificaties zijn geen garantie voor succes. Waar dingen in de bouw vaak zichtbaar en meetbaar zijn, is dit bij software ontwikkeling helaas niet het geval. De specificaties gaan over de verwachte output voor de klant: welke functionaliteit krijg ik en hoe ziet deze er ongeveer uit. Maar the devil is in the details: goede specificaties schrijven op detailniveau is ontzettend lastig, dus regelmatig lopen ontwikkelaars tegen diverse scenario’s aan waar bij het schrijven van de specificaties geen rekening mee is gehouden. Daarnaast loop je het risico op interpretatieverschillen van de specificaties: bij het [**waterval model**](https://nl.wikipedia.org/wiki/Watervalmethode) is de periode tussen specificeren en software geleverd krijgen behoorlijk lang, vooral bij grotere projecten. Je kunt in de tussentijd slecht meten of de interpretatie van het team correct is en daarnaast loop je het risico dat de behoeften van de gebruikers in die periode inmiddels zijn veranderd.

*“Hofstadter’s Law “It always takes longer than you expect, even when you take into account Hofstadter’s Law.”*

Bovenstaand proces is het waterval model dat inmiddels door de meeste IT bedrijven is losgelaten. Tegenwoordig bewegen de meeste bedrijven richting “**[Agile](https://ambrero.nl/kennis/agile)**Agile is een organisatiestructuur waarin wendbaarheid centraal staat.”. Nu begint [**Agile**](https://nl.wikipedia.org/wiki/Agile-softwareontwikkeling) als begrip door alle marketing-buzz nogal zijn waarde te verliezen, maar voor ons betekent het in ieder geval het volgende. Zet korte stappen richting je doel, waarbij je na iedere stap evalueert of de richting nog steeds de juiste is en of er zaken zijn die je kunt leren van de vorige stap die je hebt genomen. Herhaal dit proces, totdat je bent waar je wilt zijn.

## Project agility met Scrum

Klinkt allemaal vrij logisch, en dat is het ook. De methodiek die we daarvoor gebruiken is Scrum. Scrum werkt niet op basis van uitgebreide specificaties die bij aanvang van het project in beton staan gegoten. In plaats daarvan ligt er een soort silhouet van functionaliteit, het [**Product Backlog**](https://ambrero.nl/kennis/kennis/product-backlog), waarbij de functionaliteiten die op korte termijn gemaakt moeten worden vrij helder zijn, maar naar mate je richting de stip op de horizon kijkt steeds vager.

Dit geeft je de flexibiliteit om gedurende het project de functionaliteit te ontwikkelen die op dat moment de meeste waarde creëert voor je product. Daarnaast kun je makkelijk inspringen op actualiteiten binnen je bedrijf. Dankzij deze flexibiliteit is het mogelijk om in kortere tijd een productierijp systeem op te leveren, die wellicht nog niet bij de stip op de horizon is aanbeland, maar in ieder geval wel gebruikt kan worden, wat twee grote voordelen heeft:

* Testresultaten van echte gebruikers
* Het product verdient zichzelf sneller terug (opbrengsten/besparingen)

Echter het grootste voordeel van Scrum is de samenwerking tussen opdrachtgever (Product Owner) en het ontwikkelteam. Er is een korte feedback-loop, de stakeholders kunnen iedere 2 weken een resultaat zien en hier feedback op geven etc. Al met al zal dit de kwaliteit van de functionaliteit en de acceptatie van het product ten goede komen.

## Projectfactoren Goed, Snel, Goedkoop – kies er twee

Maar deze werkwijze kent ook nadelen. De stip op de horizon blijft vaag en het is heel moeilijk (en risicovol) om hier een datum of budget aan te koppelen. Hierdoor krijgt een opdrachtgever soms het gevoel dat het project een bodemloze put is, zonder duidelijke deadline, waar iedere Sprint weer een stuk van zijn budget wordt verbrand. En dus rijst de vraag: kun je niet meer zekerheid krijgen?

**Jazeker! Software ontwikkeling heeft een aantal factoren:**

* **Scope**; welke functionaliteit gaan we ontwikkelen?
* **Budget**; hoeveel gaat het kosten? Oftewel, hoeveel mensen werken aan het project.
* **Tijd**; wanneer is het af?

**Ik geef een paar scenario’s van fixeren van deze factoren:**

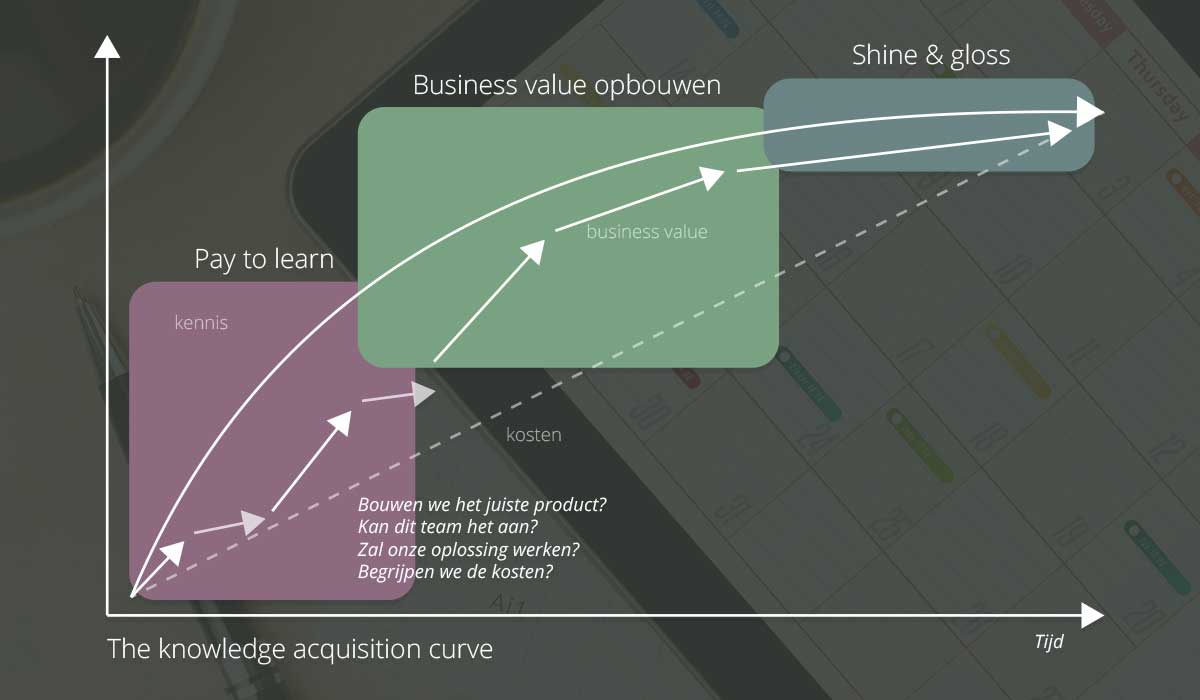
* **Scope is fixed, tijd is fixed**; de leverancier garandeert dat hij op een bepaalde datum een bepaalde hoeveelheid functionaliteit af heeft. De enige manier waarop hij dit kan garanderen is als hij flexibel mag zijn in de inzet die hij op het project zet. **Budget zal dus variabel moeten zijn.**
* **Budget fixed, tijd is fixed**; de leverancier garandeert een bepaalde inzet (budget) over een bepaalde periode (tijd), maar kan geen garanties geven over de hoeveelheid functionaliteit die ontwikkeld kan worden.
* **Budget is fixed, scope is fixed; de enige variabele is tijd, oftewel snelheid van ontwikkelen**. Door sneller te ontwikkelen zal de software meer fouten bevatten en zal de kwaliteit naar beneden gaan. Daarnaast zal bij iedere tegenslag de kwaliteit verder onder druk komen te staan..

Het **laatste scenario is onwenselijk**. Je hebt als opdrachtgever op de korte termijn zekerheid, maar waarschijnlijk is het resultaat een stuk software dat niet goed functioneert, vol zit met fouten en slecht onderhoudbaar is. Op de lange termijn is dit de duurste optie en omdat wij graag achter ons product willen staan raden wij dit ten zeerste af!

Alle factoren fixeren voor het hele project is dus niet realistisch. Maar er kan wel een fixed-price per Sprint worden afgesproken. Vlak voor een Sprint staan de eisen immers vast en hierdoor is er geen risico op interpretatieverschillen. Daarnaast helpt het om het [**Product Backlog**](https://ambrero.nl/kennis/product-backlog)Het Product backlog omvat een lijst met werkzaamheden die nog binnen het project moeten worden uitgevoerd. al in een vroeg stadium zo compleet mogelijk te maken. Op deze manier kan het team al een voorlopige schatting maken over de gevraagde functionaliteit en kan dit worden meegenomen in budgetoverwegingen.

## Business Value creëren

Als je op projectniveau meer zekerheid wilt creëren dan is een Fixed Budget, Fixed Time scenario een goede optie. Hierbij is het duidelijk dat het projectteam in de afgesproken periode keihard gaat werken. En dat zij samen met de Product Owner het maximaal haalbare aan Business Value gaan creëren. Dit dwingt het team om pragmatisch te werk te gaan én de klant om te zoeken naar de bedrijfswaarde in de functionaliteit die hij van het ontwikkelteam vraagt.



Bovenstaand plaatje illustreert hoe dat in zijn werk gaat. Gedurende de eerste fase van het project creëren we de waarde door risico’s weg te nemen. Visievorming over functionaliteit en technologie is in deze fase heel belangrijk. Daarnaast ontwikkelen we prototypes voor de verschillende technische uitdagingen. Als de risico’s van de baan zijn dan richten we ons op de Business Value. De functionaliteit die de meeste waarde toevoegt voor de organisatie ontwikkelen we als eerste. Naarmate het project vordert en de toegevoegde Business Value per Sprint afneemt, staat het de klant vrij om op ieder moment het project af te ronden (trimming the tail).

## Kostenbeheersing in Scrum; communicatie is het sleutelwoord

Kostenbeheersing is bij alle IT projecten een uitdaging, met of zonder Scrum. Het bewaken van een budget is een gezamenlijke verantwoordelijkheid. Met name bij projecten waar niet voldoende gecommuniceerd wordt loopt dit vaak fout. Je kunt de grootste valkuilen vermijden door tijdens het project voortdurend met elkaar in dialoog te blijven en elkaar regelmatig bij te praten over voortgang, verwachtingen en het budget.

Communicatie is en blijft het sleutelwoord!

### BART MATTHAEI