Sander Hoogendoorn

Alhoewel ik het liefst code schrijf, heb ik een zwak voor het schatten van projecten. En dan niet een academische functiepuntenanalyse, maar elke pragmatische aanpak verdient mijn steun. Tijdens mijn loopbaan heb ik aardig wat schattingen gemaakt en gezien. Van tenenkrommend tot waar-halen-ze-die-getallen-vandaan.

Contingency en bierviltjes

Waarom rekenen veel organisaties voor projectmanagement een toeslag van twintig procent? Waarom geen 14.81%? Zouden schriftgeleerden in die organisaties dat hebben vastgesteld aan de hand van ruim honderd gelijksoortig uitgevoerde projecten? Of komt het van een bierviltje waar de accountmanager de prijs van de concurrent al op had genoteerd? Ik heb beide meegemaakt.
Een van de leukste schattingen maakte ik toen mijn toenmalige manager me vroeg een schatting op te leveren aan de hand van een ontwerp dat de klant al had geschreven. Natuurlijk was de termijn voor het uitbrengen van de offerte al bijna verstreken, zoals dat hoort bij het moment waarop developers worden benaderd om mee te werken aan offertes. Laat ik een lang verhaal heel kort maken. Ik kreeg een half uur. Geen tijd meer om het tweehonderd pagina's dikke document door te spitten. Op dat moment – 1996 – bedacht ik de ultieme metriek: het aantal uur dat nodig is voor het bouwen van de software is gelijk aan het aantal pagina's van het ontwerp, vermenigvuldigd met het aantal entiteiten in het datamodel. Cool toch?

Het verhaal heeft nog een staartje. Zonder blikken of blozen nam mijn manager het cijfer over, zonder naar de herkomst te vragen. Maar hij telde er nog wel tien procent bij op. Voor contingency. Toen wist ik niet precies wat dat was - en nu nog niet. Mijn werkgever won de offerte, en wonder boven wonder kwam ik in het project terecht. En de clou was: het project werd op tijd en binnen budget opgeleverd. Sterker nog, we hadden tien procent van het budget over.
Vorige maand viel er een interessant boek in mijn postvak. *Agile portfolio management* heet het. Uitgegeven in de *Best Practices* reeks van Microsoft Press. Op zich geen onverdienstelijk werk over hoe je je portfolio (een generic collection van projecten) agile kunt beheren. Daarbij blijven alle principes van agile netjes overeind, zoals verwacht. Auteur Jochen Krebs geeft netjes aan hoe je de projecten in je portfolio prioriteert en hoe je het succes ervan kunt meten – tijdens een retrospective. En alhoewel het boek tweehonderd pagina's voortkabbelt, staan er toch her en der interessante statements in. *Balance your portfolio between risky and rewarding projects* bijvoorbeeld. Die moest ik maar onthouden.

Maar dan tref ik het hoofdstuk met de veelzeggende titel *Metrics* aan. Daarin gaat de auteur in op het schatten van use cases - geen smart use cases trouwens. Eens zien: is Krebs een contingency-teller of een bierviltjes-voorspeller? Zonder met zijn ogen te knipperen presenteert hij fraaie tabellen met factoren die de complexiteit van een project beïnvloeden. Zo leren we dat *Distributed Systems* een factor 6 toevoegen en *Easy To Use* een factor 0.5. Dat wist ik nog niet. Uit hoeveel projecten zouden die factoren zijn gedestilleerd?
Het allermooist vind ik echter de formules. Ik zal ze u niet onthouden. Volgens Krebs is de complexiteit van software gelijk aan 0.6 + (0.01 x de som van een rij van factoren zoals hierboven). Vervolgens rekent hij de complexiteit van de omgeving uit via 1.4 - (0.03 x de som van een lijst van omgevingsfactoren, zoals ervaring met UML en de capaciteit van de analysten). En tenslotte wordt het aantal *use case points* bepaald door beide uitkomsten te vermenigvuldigen met het aantal ongewogen use case points. Bent u er nog? Benieuwd wie deze getallen op een bierviltje heeft gekrast. Jammer voor de geloofwaardigheid van een niet onaardig boek. «

Sander Hoogendoorn
blog.sanderhoogendoorn.org