Klomps & Snels Klomps & Snels Klomps & Snels

De zoete smaak van Java

Ons professionele leven staat bol van het maken van keuzes. Bijvoorbeeld tussen Java of .NET of tussen maatwerk of een pakket. Wie denkt dat dit lastig is, moet zich eens verdiepen in de geschiedenis van koloniaal Java, want ons koloniale verleden blijkt één grote 'aha-erlebnis' te zijn...

In de negentiende eeuw was Java befaamd om zijn succesvolle suikerrietteelt. In de concurrentieslag met Cuba, de wereldleider, zocht Java continu naar een suikerrietsoort met een hoger suikergehalte. De productie werd op allerlei manieren steeds verder opgekrikt. Jarenlang was dit een succesvolle strategie, totdat tegen het einde van de negentiende eeuw de seré-ziekte de sawahs van Java trof. Op dramatische wijze kwam men tot de conclusie dat bestendigheid tegen verschillende ziektes ook belangrijk was. Nog steeds zien we dat functional requirements voor de hand liggen en dat non-funcional requirements zoals security en toekomstvastheid gemakkelijk vergeten worden...

Koortsachtig werd onderzocht hoe de seré-ziekte het hoofd geboden kon worden. Bestond er een sublieme rietsoort die goed bestand was tegen alle mogelijke ziektes, en ook een hoog suikergehalte had – of waren er andere mogelijkheden? Een theoretische optie was het planten van meerdere soorten suikerriet door elkaar heen, elk met zijn eigen specifieke kenmerken en weerstand. Zo zou het geïntegreerde geheel letterlijk tot een 'best of breed'-oogst leiden, maar het gevolg zou wel zijn dat elke soort op zijn eigen, unieke manier onder handen genomen moest worden. De bijbehorende dilemma's voor de toenmalige suikerrietteler, zijn te 'vertalen' naar een situatie waar de hedendaagse softwarearchitect zich vaak in bevindt:

- Hoe moeten we de soorten mengen? Dwars door elkaar heen planten of in gescheiden 'sub-sawahs'? Hoe combineren we al die verschillende programmeertalen en platformen goed met elkaar?
- Waar halen we snel de kennis vandaan zodat het hele landschap in het gareel gehouden wordt en het doet wat wij willen dat het doet? Waar halen we de specialistische generalisten vandaan die meerdere platformen beheersen? Of de echte specialisten die volledig los zijn op één van de platformen?
- En welke zekerheid hebben we dat we volgende jaren ook een goede oogst hebben? Hoe zorgen we voor die toekomstvaste architectuur en hoe houden we de kennis op peil?

Net als vroeger op Java geldt nu voor IT-projecten dat het oplossen van integratievraagstukken draait om het maken van keuzes. Keuzes die gemaakt worden op basis van objectieve feiten. Volgens Microsoft verliest Java het van .NET. Volgens Forrester is juist Java de populairste taal voor zware bedrijfsapplicaties en is .NET bezig met een langzame start... Kennelijk is het heel eenvoudig om, gebaseerd op *dezelfde* feiten een eigen werkelijkheid te vormen.

Een softwarearchitect moet kiezen voor een platform dat aansluit bij de natuurlijke omstandigheden; de functionele én niet-functionele eisen en wensen. Een projectmanager moet kiezen voor een aanpak die aansluit bij de bevolking en omgeving maar die ook beantwoordt aan de tijdsdruk en gewenste projectmanagementstandaarden. En de software-engineer moet rekening houden met de gekozen oogst- en drainagetechnieken; de standaards en best practices van het platform. Als regisseur van dit alles moet de opdrachtgever keuzes maken, gebaseerd op de meerjarige business-case, de visie van de organisatie en tegelijkertijd het beperken van de impact op 'de operatien'. Allemaal keuzes die met elkaar in overeenstemming moeten zijn – gebaseerd op begrijpelijke en objectieve argumentatie! Het kan dan ook niet anders dan dat ze samen worden gemaakt, waarbij argumenten over en weer worden gedeeld. Waarbij elkaar inzicht wordt geboden in de fundering van de keuzes en er gezamenlijk in een stimulerend, effectief en goed begeleid proces een gewogen en weloverwogen richting wordt gekozen. Of het nu .NET wordt, Cobol, of eh... Java.

Birgit Klomps en Tommes Snels, beiden managing consultants bij Capgemini, schrijven over het werk in projecten, workshops en alles wat daarbij komt kijken.