Bericht uit Java

Voor wie het nog niet weet (foei!), de hipste programmeertaal is momenteel Ruby on Rails. Net als Pokémon-kaartjes op de lagere school, willen alle trendgevoelig programmeurs tegenwoordig aan de Ruby on Rails. Ruby on Rails ís inderdaad geweldig. Alleen die naam al. 'on Rails' klinkt zo ontzettend cool!

Java on Rails

De naam 'Java' was ooit ook een goeie zet, toen de meeste namen van programmeertalen gewoon afkortingen waren (gelukkig kwamen er na A, B en C niet nog meer talen in deze reeks). Java deed denken aan verre oorden, aan kruiden, aan mooie mensen en aan dampende koffie. 'On Rails' doet denken aan snelheid, aan kracht. 'On Sterioids' is nog wat krachtiger – maar heeft ook de wat negatieve bijklank van valsspelen en dommekracht.

Ruby zelf is een vergeten programmeertaal – zoals Smalltalk een vergeten programmeertaal is. Ruby is – net als Smalltalk – een dynamische en zeer objectgeoriënteerde taal. Dat zijn talen waar niet veel mensen raad mee weten. De makers van Ruby on Rails excuseren zich zelfs een beetje voor Ruby. Ze schrijven ergens iets in de trant van: "Het is even wennen – zo'n raar taaltje als Ruby". Dat nou net Ruby 'on Rails' is gezet, dat is wat mij betreft een beetje toeval. Je zou bijna kunnen zeggen dat ze het om de naam gedaan hebben. 'Ruby on Rails' klinkt beter dan 'Smalltalk on Rails', 'Java on Rails' en helemaal 'C++ on Rails'. Ik ga er van uit dat de keuze voor Ruby verder redelijk arbitrair is. Het dynamische karakter van Ruby was vast handig bij het implementeren van de Rails-aspecten, maar volgens mij had dat ook wel zonder gekund. Laten we er van uit gaan dat we net zo goed 'Java on Rails' hadden kunnen hebben.

Wat is Ruby on Rails dan eigenlijk? De makers van Ruby on Rails hebben als belangrijkste statement dat conventies cruciaal zijn voor goed programmeren. Dus hebben ze conventies gewoon heel handig uitgeprogrammeerd. Alles waar een conventie voor is – is een stukje 'on Rails' voor gemaakt. Bij conventies moet je denken aan de manier waarop een tabel in een database gerelateerd is aan een domein-object of aan de manier waarop een domein-object gekoppeld wordt aan een webpagina. Ze vinden ook dat het onzin is om te gaan kiezen tussen verschillende conventies. Het consequent kiezen voor één conventie is belangrijker dan het eeuwig doordiscussiëren over welke conventie optimaal is. Java-programmeurs kennen voor elk onderdeel vele conventies en raamwerken. Java programmeurs moeten telkens kiezen tussen al die conventies (en bijbehorende raamwerken) en moeten die conventies daarna tot een geheel weten te smeden. Ruby on Rails zorgt er dus voor dat er heel veel duidelijkheid is en dat er maar minimaal geprogrammeerd hoeft te worden – de rest wordt gegenereerd.

Dat genereren is wat mij betreft wel prettig maar toch een beetje raar. Als die conventies (blijkbaar) zo goed zijn – waarom zie ik ze dan nog? Wanneer alle code behalve de meest cruciale domeinlogica gegenereerd wordt, waarom krijg ik die gegenereerde code dan nog te zien? Moet ik er iets mee? Waarom krijg ik een database te zien als er inderdaad maar één manier is om een database te gebruiken? Het is bijna alsof ze me proberen te imponeren met al die gegenereerde code. Waarom maken we niet een platform waar we alleen maar die domeinlogica aan toe hoeven te voegen? Laat het platform maar al die conventies regelen. Ik wil die conventies helemaal niet zien! Met J2EE hebben we zoiets geprobeerd, maar daar hadden we (ze) totaal verkeerde conventies gekozen. Het wordt tijd om te leren van Ruby on Rails maar vooral geen Java on Rails te gaan maken. Het wordt tijd voor een nieuw Java-platform. Eentje waar conventies (zoals die van Ruby on Rails) onderwater toegepast worden – zonder dat we er over na hoeven te denken – zonder één letter code!

Daan Kalmeijer
is docent consultant bij DNV-CIBIT
(e-mail: Daan.Kalmeijer@DNV.com)