vraag de specificaties van de gebruikte tools op van www.radrace.nl

TOOLS & TEAMS

thema

Sybase

'Ik geloof heilig in dit product'

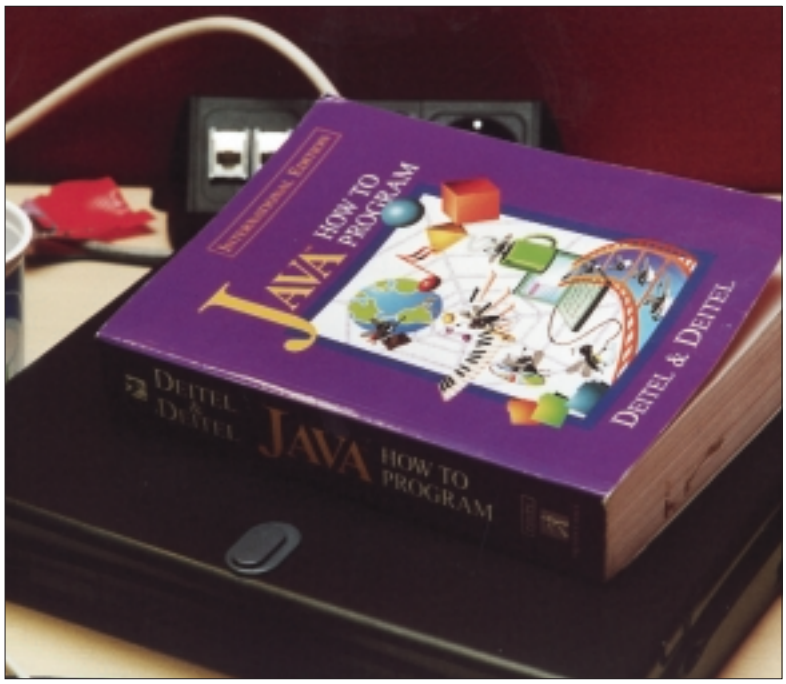
Hoe kijken jullie terug op de RAD Race? De opdracht viel mee, het was een hoop werk, maar het was te doen, mits je je goed voorbereid hebt. Waarbij ik dan wel in mijn achterhoofd had, dat onmogelijk alles door ons afgemaakt had kunnen worden. Het had prima opgelost kunnen worden met onze producten en het had ook nog wel sneller gekund, in die zin valt het mee, het viel een beetje tegen in de uitwerking: het was gewoon veel.

Hebben jullie gebruik gemaakt van speciale bibliotheken, of van standaard componenten? Alleen van de training voor de portal waarin alle belangrijke componenten aan de orde komen, hoe benader ik een database, hoe maakt ik de business logica en hoe zorg ik dat ik het spul presenteer. Die wordt standaard met de portal meegeleverd.

Hoe was de taakverdeling binnen jullie team? We hebben de taken verdeeld, redenerend vanuit de producten. Je hebt de database, een applicatieserver in het midden en een presentatielaag daarboven. In die applicatieserver zitten bij ons ook allemaal security componenten. Als je kijkt naar de verdeling van het werk: ik weet genoeg van de database af om daar allerlei dingen in te schrijven, ik ken de presentatielaag en de security componenten die erin zitten. Ik ben echter geen echte Java-programmeur. Dus ben ik naar een partner van ons gegaan en heb gevraagd: 'hebben jullie een Java-krasser, iemand die goed en snel Java-code kan schrijven?'. Helaas zijn we er niet meer toe gekomen om van tevoren met elkaar te oefenen, eigenlijk waren we dan ook geen team, in de gebruikelijke zin. Tijdens de RAD Race heb ik hem het framework gegeven: zo en zo ziet het eruit, en zo gaan we het ontwikkelen. Vanuit die gedachte hebben we een soort splitsing gemaakt: alles wat portal en database gerelateerd is, zou ik doen en alles wat Java en businesslogica gerelateerd was, dat zou hij doen. Later bleek dat hij problemen had met het implementeren van bepaalde businesslogica. Omdat dat te ingewikkeld zou worden heb ik het op een gegeven moment opgepakt en in triggers opgelost.

Aan het eind van dag één was er eigenlijk vrij weinig klaar. Hoe kwam dat? De grap is: zonder business logica kan ik niks presenteren. We hadden redelijk wat af onder de motorkap maar we hadden geen business logica om het te presenteren. Daar is het uiteindelijk een beetje fout op gelopen. Dat had ook te maken met de voorbereidingstijd van ons beiden. Waar het dan ook een beetje fout is gegaan door gebrek aan communicatie vooraf, is dat mijn collega de tijd nam om een mooi Javadesign te maken, dat echter qua interface niet bleek aan te sluiten bij onze portal. Het leek eerder alsof de interface-eisen duidelijk waren, dus dat werd pas heel laat duidelijk, tegen vier uur 's middags. Dat heeft te maken met de voorbereidingstijd. Wij hebben elkaar de avond van tevoren voor het eerst gezien en het leek toen alsof alles duidelijk was, maar dat bleek in de praktijk niet zo te zijn.

BIJSCHRIFT: **Het Sybase-team: Arjaan Peree en René Ouwehand (r).**



BIJSCHRIFT: Bij sommige teams behoefde de Java-kennis kennelijk nog enige verdieping.

Wat hadden we af: we konden inloggen, er waren wat schermen klaar waarmee je wat gegevens kon opvragen, die schermen waren af, ik kon ze alleen niet laten zien omdat de business logica nog niet af was. Er was ook nog businesslogica in de triggers af, dus we hadden meer af dan we lieten blijken, alleen zag ik er ook niet echt een gat in dat er de volgende dag businesslogica zou komen.

Zou je het een volgende keer weer zo aanpakken? Ik zou in ieder geval weer direct dezelfde aanpak kiezen, maar alleen nu wel eisen dat we de volgende keer wel een week van tevoren oefenen. Qua aanpak zou ik nog steeds die scheiding aanhouden, want je hebt allebei je tijd hard nodig. Degene die de business logica maakt, want dat is een hele kluif, moet bepalen wat waar komt, in de database of in de applicatieserver, dat is typisch zijn werk. De ander kan zich bezig houden met alles wat daar omheen hangt. Op die manier houd je een redelijke focus waarbij de een vliegende keeper is en de ander redelijk kan focussen op de business logica.

Ik geloof heilig in dit product, ik zou weer met dit product de race ingaan. Met een goede Javaprogrammeur moet het mogelijk zijn goed te scoren, al zullen we misschien niet nummer één worden. Daar is het product uiteindelijk ook niet voor bedoeld, het is geen RAD tool. Maar een kunt er wel binnen een dag een goede applicatie mee bouwen en binnen twee dagen een complete omgeving.

Borland

Viel de deelname aan de RAD Race mee of tegen? Het viel zowel tegen als mee. Dat heeft ermee te maken dat je geen voorstelling hebt van wat je moet verwachten. Het was ook de eerste keer. Ik heb het als heel leuk ervaren, zeer uitdagend. Dat is misschien waar je je op de eerste dag een beetje in verslikt. Je ziet de omvang van de opgave en dan moet je echt de discipline hebben om een goede indeling van de dag te maken en te focussen op het resultaat van de eerste dag. Je moet goed beseffen: ok, we gaan eerst de basisfunctionaliteit neerzetten en dan langzaam aan die leuke optionele dingen werken. Ik denk dat we dat wel goed aangepakt hadden. Wat tegenviel is dat het niet zo liep zoals we dat graag hadden gewild. We zaten er zo naar uit te kijken en op de dagen zelf lopen de dingen niet zoals je gepland hebt, en dat valt wel een beetje tegen.

Jullie werkten met een van de laatste bèta's, terwijl de officiële versie bijna tegelijkertijd uitkwam. Ja, we hebben net geen gelegenheid gehad om de definitieve versie te installeren. We hebben met een vrij late fieldtest gewerkt, tegen de productieversie aan, maar nog niet af. We waren redelijk gewend aan deze versie, maar we hadden nog niet een dergelijke exercitie uitgevoerd. In dat opzicht is de RAD Race heel goed geweest. In een grote applicatie komen er direct dingen naar boven, omdat je behoorlijk veel tegelijkertijd gaat gebruiken. Dan wordt het wel snel duidelijk wat er mis was.

Waren die problemen in de definitieve versie opgelost? Nee, niet alles. Inmiddels wel, er waren nog een paar kleine probleempjes waar wij ook tegenaan gelopen zijn. Sinds de versie die wij gebruikt hebben, die dateerde van 19 november, zijn er nog behoorlijk wat zaken gesignaleerd en opgelost door het team. Als wij met de officiële release hadden gewerkt hadden we minder problemen gehad, maar dat wisten we van tevoren niet. Dat is altijd de gok als je met een bèta aan de slag gaat.

Nu verschijnen er van JBuilder zo vaak nieuwe versies, dat zo'n kersverse versie ook een zeker risico in zich draagt. Ja, dat klopt. Het heeft te maken met de keuze die wij gemaakt hebben door de insteek van het project. Wij hebben ervoor gekozen om te werken volgens een bepaalde architectuur, volgens de laatste J2EE release. Dan hebben we met de nieuwste features te maken, dus willen we EJB volgens de 2.0 en er was op dit moment nog geen IDE die dat

ondersteunde. JBuilder is de eerste die op dat vlak een aantal andere belangrijke features ondersteunt. Je neemt dus een risico, maar wij vonden dat er in deze aanpak veel voordelen zaten. Dat waren voornamelijk de server components en de functionaliteit die ze bieden, darnaast het gemak waarmee je database tabellen kunt mappen en daar snel verandering in kunt aanbrengen. We hielden er rekening mee dat je met een change project te maken zou krijgen vanuit de RAD Race. Daarom wilden we een architectuur neerzetten waarin we zo flexibel mogelijk zouden zijn, en zo snel mogelijk dat soort veranderingen door te voeren. Daarom hebben we ook gekozen voor EJB 2.0.

Daarvan hadden jullie er nogal wat geschreven tot de crash. Om die daarna terug te brengen naar versie 1.1 bleek geen haalbare kaart. Ja, dan moet je het eigenlijk opnieuw gaan doen. De aanpak is wezenlijk veranderd omdat tussen EJB 1.1 en 2.0 drastische veranderingen zijn opgetreden. Alle database gerelateerde code die in de componenten zit, is eigenlijk niet meer aanwezig. Dat betekent ook dat zo'n component vrijwel onafhankelijk van een databasemodel wordt. Daardoor ben je inderdaad in staat zeer snel te reageren op een veranderende omgeving. Als alles gewerkt had, hadden we aan de serverzijde vrij snel op veranderingen kunnen reageren. Aan de andere kant hebben we natuurlijk de presentatielogica, en het grote voordeel is dat je tegen een set van componenten praat. Wij hadden duidelijk het voornemen om een API te definiëren tussen representatie en componenten die eigenlijk helemaal los van het hele databasemodel stonden, zodat veranderingen op databaseniveau en op de presentatielaag niet zoveel impact zouden hebben.

Tegen twaalf uur begon het al een beetje mis te gaan. Toen kreeg Krishnan last van dat deployment script, de XML files raakten corrupt. Hij is aan de slag gegaan om dat met de hand te herstellen, dat is tot op een zeker niveau redelijk gelukt want om een uur of één had hij weer een compileerbaar project, maar ja toen moesten we weer dingen gaan toevoegen en iedere keer als hij dat deed betekende dat, dat zijn deployment descriptors scripts weer corrupt raakten. Zo kwamen we in een vicieuze cirkel, want hoe ga je dat herstellen, dat gaat dan teveel tijd kosten. We hebben nog overwogen om terug te gaan naar de vorige versie maar daarvoor was het toch te laat.

Doen jullie de volgende keer weer mee? Dan zijn we zeker van de partij. Een ding heb ik ervan geleerd. Wij hadden de insteek: hoever kunnen we komen met de basistools, out-of-the-box, met geen enkel additioneel framework erbij applicaties maken. Dat gaat redelijk, maar zeker bij zo'n applicatie als deze heb je een framework nodig waarbij je al snel op de businessfunctionaliteit kunt gaan focussen. Als je een webapplicatie maakt, is dat voornamelijk op de presentatiekant. Teams die in de praktijk zitten, die nemen dat soort zaken mee van project naar project terwijl wij meer in de rol van consultants zitten en elk project is weer anders.

Op zich had ik om één of twee uur ook de uitdaging nog wel aan willen gaan om met de vorige versie opnieuw te beginnen. Het punt is: hoe ver kom je? Achteraf denk ik, dat we nog een eind hadden kunnen komen. Zeker als ik zie wat het merendeel van de teams aan het einde van de dag gerealiseerd hadden. Maar ja, als je op dat moment in die situatie zit… We dachten dat we zo'n grote achterstand hadden opgelopen, dat die niet meer in te halen was. Achteraf gezien viel dat wel mee.

> 'We dachten dat de achterstand zo groot was dat die niet meer was in te halen. Achteraf viel dat wel mee.'

De jury heeft aan ieder team gevraagd: hoever denken jullie te komen? Mede op basis daarvan is bepaald wie door mocht gaan. Ik kan me voorstellen dat je na zo'n crash minder optimistisch bent. Er waren ook wel teams die misschien wat te optimistisch waren op dat moment. Dat is een vervelend moment, ja. Aan het einde van de dag was ik met veel plezier de tweede dag in gegaan zijn, maar het mocht niet zo zijn.

BIJSCHRIFT: Crisisoverleg bij het Borland-team: Krishnan Subramanian en Gerard van der Pol (r).

IC Group

Werken met een kale JDK

Robert Bor: Het resultaat is tegengevallen, omdat we niet gedacht hadden dat we tegen die vaste RAD-tools moesten opboxen. Het evenement op zich is hartstikke meegevallen, we hebben het in ieder geval heel erg naar onze zin gehad. De opgave was te doen: alleen met de methode waar wij voor hadden gekozen heb je gewoon veel meer tijd nodig om er te komen, wij gingen niet met RAD-tools aan de slag, maar simpelweg met een kale JDK.

Dat heeft te veel tijd gekost? Je hebt een behoorlijk voorbereiding nodig. Je moet je RAD-tools goed selecteren om met deze wedstrijd mee te kunnen doen. Je hebt er echt voordeel van wanneer je ze gebruikt.

Als jullie volgend jaar weer zouden meedoen, zouden jullie het dan weer zo doen? Wat we in ieder geval niet doen is een standaard tool gebruiken, simpelweg omdat we dat in het dagelijks werk ook niet gebruiken. Wat we zouden kunnen doen is Visual Age inzetten, die gebruiken we wel. Maar waar we wel heel erg naar gaan kijken is: oké, hoe kunnen wij hier zelf alvast componenten bouwen die de opdracht kunnen vergemakkelijken? Daar hebben we van tevoren heel weinig tijd aan besteed. Met deze wedstrijd hebben we daar een heel goede indruk van gekregen. Mijn collega Erik heeft zelfs hele goede ideeën hoe we dat kunnen gaan aanpakken. Waar je met name aan kunt denken is een persistency framework. Dat is efficiënter, zodat je die tabellen vanuit de database in objecten kunt omzetten en omgekeerd hoe je die objecten weer kunt persisten naar de database.

Hoe was de samenwerking, waren jullie al voor de wedstijd een team? Absoluut, die samenwerking gaat heel goed. Saillant detail is dat we door die enorme werkdruk tot een verdeling kwamen dat hij min of meer de leiding nam in het project en dat ik volgde. Normaal werk je veel minder onder druk dus dan is er meer tijd om tot een consensus te komen. Nu moest onder de tijdsdruk een van de twee de leiding nemen. Anders kom je in een poldermodelstructuur vast te zitten en dat kun je je niet veroorloven.

En de verdere taakverdeling? Jullie hebben geen functionaliteit op de database gelegd, want jullie hebben met Microsoft Access gewerkt? Erik heeft zich met name gericht op het opzetten van de GUI en het formuleren van de interfaces, en ik heb me met name gericht op het bouwen van de backend.

Hoe vond je de opdracht? Ik vond de opdracht heel mooi, een heel goede opdracht. Het was heel leuk om te maken. Wat wel opviel is dat hij heel erg toegespitst is op rapid application development. Niet dat dat een probleem is, want dat was natuurlijk ook de assignment, maar dat sloot daar heel erg goed bij aan en dat uit zich dan met name dat de opdracht heel omvangrijk is: de diverse subassigments gaan niet diep, maar wel heel breed. Terwijl als je wel zou zien voor een opdracht die wel heel diep gaat, dat moet je veel meer logica inbouwen op de backend, en dan zit daar een groot gedeelte van het werk en dat leent zich helemaal niet voor rapid application development. Met een goede RAD tool kun je dan ook uitstekend iedere opdracht voltooien en als je onze aanpak hierbij kiest – wij zijn meer mensen van de backend – dan blijkt dat je er heel erg over struikelt. Dat is wat we in de toekomst willen vermijden, dan gaan we ons richten op het snel ontwikkelen van kleine componenten.

Pink Roccade

'Het zijn ook niet de tools die het hem doen'

Jullie werkten met Active Serve Pages en SQL server. 'Ja, alleen met Notepad red je het niet in twee dagen dus we hebben geprobeerd een hoop stored procedures in SQLServer te maken, dat is sowieso belangrijk. Bovendien hadden we van tevoren al een kleine website gemaakt met een paar pagina's waarbij elke pagina een overzicht van een tabel uit een database kon laten zien, waarmee je gemakkelijk informatie in de database kon zetten of aan kon passen in de database. De voorbeelden van pagina's waren al af, en bovendien hadden we allerlei componentjes die we zelf gebruiken om snel en op een goede manier dingen in te zetten. Die hebben we eigenlijk constant gekopieerd, maar dan steeds met een andere tabelwaarde.'

Is het jullie meegevallen? 'Het viel niet echt mee, maar het viel ook niet tegen. Je verwacht wel dat je het heel druk krijgt. Ik denk dat het zelfs nog meer was dan dat we hadden gedacht maar we waren ook sneller dan we dachten. Het was redelijk wat we er van verwacht hebben. Het is een duidelijk gedefinieerde case zoals je hem

op school ook kunt krijgen, bij wijze van spreken, een systeem met een hoop businessrules erin en maak het maar. Het was niet heel ingewikkeld of zo. Het was wel veel, maar niet irreëel veel- gewoon een goede case.'

Hoe was de arbeidsverdeling binnen jullie team? Gepko en ik hebben al een paar jaar samen klusjes gedaan dus we zijn helemaal op elkaar ingespeeld. Op de dag zelf heeft Gepko zich vooral op de database gericht en ben ik wat meer met de code zelf bezig geweest, althans in grote lijnen.

Als je weer zou meedoen, hoe zou je het dan aanpakken? 'Wanneer ik zou weten dat ik volgend jaar dezelfde opdracht zou krijgen zou ik op een andere manier te werk gaan. Maar dat weet je natuurlijk niet van tevoren. Ik heb persoonlijk ook iets van: het zijn ook niet de tools die het hem doen. Met Java kun je dit net zo goed maken als met ASP, het is maar wat je fijn vindt werken. Uiteindelijk komt er allemaal code uit. Het is gewoon een kwestie van even nadenken over hoe je het systeem in elkaar gaat zetten. Ik denk ook dat het niet meer uitmaakt - als je eenmaal een snelle manier hebt gevonden om die business rules aan te passen - of je dat in Java of ASP doet, denk ik. Persoonlijk denk ik, maar dat komt omdat ik een beetje Java fan ben: goh, als ik dit nog eens had moeten doen was ik met Java Beans aan de slag gegaan. Het gaat erom dat een tool stabiel is en bepaalde dingen kan en verder gaat het erom wat je zelf fijn vindt werken en goed kent.'

BIJSCHRIFT: Uiterste concentratie bij het Pink Roccade-team: Gepko Sterringa en Tim Abeln (r).

Magic

Is het mee- of tegengevallen? Voor het Nederlandse team: 'Ik vond deze keer de RAD Race beter in elkaar steken dan de vorige keer, veel duidelijker, meer persoonlijke aandacht of uitleg te geven, waardoor je in principe beter op weg was. De case was ook duidelijk, maar omdat er ook ruimschoots de tijd was om de case door te nemen denk ik dat je beter beslagen ten ijs kwam. Ook de beschrijving in de case was vele malen duidelijker dan het jaar daarvoor.'

De Israëlische collega's van het andere Client/Server Magic-team verklaarden, dat zij de case pittig hadden gevonden. Ze hadden nog nooit zoiets meegemaakt, ook omdat het om een gecombineerde expertise vroeg, niet alleen de opdracht, maar het feit dat er analytische kennis gevraagd wordt. En dan zat er ook nog een SQL deel verwerkt.

Ze hebben het toch heel goed gedaan. Ze vonden de case van hoog niveau. Ze zijn ook van plan dit soort cases in de toekomst te gaan gebruiken om nieuwe medewerkers te beoordelen, juist vanwege de eisen aan de combinatie van expertises. Het Israëlische team werkte met een programmeur en een analist. Noam is voornamelijk programmeur, en Juval analist. Noam hield zich voornamelijk bezig met programmeren en stuurde de analist aan met wat voor werk hij voor hem moest doen.

Van het Nederlandse team had ik de indruk dat jullie zonder veel te praten van elkaar wisten wat je aan het doen was. Klopt. Wij hebben taken opgesplitst, we hebben maar een half oor nodig omdat we elkaar goed kennen. De een hield zich bezig met het datamodel en de ander ging alvast met een stuk van de programmatuur aan de gang, op een gegeven moment kom je op een bepaald niveau en werk je gelijktijdig op. Ik deed het programmeren van de business logica.

Magic lijkt me wel geschikt voor een RAD-wedstrijd. Toch bleek in de praktijk toch nog wat tijd verloren te gaan. Wij hadden als nadeel, dat wij niet echt de hulpmiddelen hebben om EAD diagrammen of zo te maken, dat zit eigenlijk min of meer in onze hoofden. Dan is het afhankelijk van hoe snel je het een en ander doet - hoe vat je het systeem in Magic samen. Dat voorgedeelte doe je zelf allemaal, waar je bij de anderen toch wat hulpmiddelen hebt om zo'n EAD-diagram te genereren en van daaruit te gaan werken, is dat bij ons een stuk werk dat aan de programmeurs en aan de analisten wordt overgelaten.

BIJSCHRIFT: De beide Magic-teams: v.l.n.r. Noam Honig, Yuval Lavi (Israëlische team), Chris de Bijl en Ronald Span (Nederlandse team).

Hoeveel tijd heeft dat gekost? Ongeveer een halve dag. Daarna gaat het heel snel. Eerst gaat het erom dat je moet begrijpen: wat is het systeem, hoe zijn de bestanden onderling gekoppeld en hoe ga je daar vervolgens mee aan het werk. Dat is het voornaamste voordeel. Wij zeggen altijd, dat de programmering niet veel voorstelt omdat het eigenlijk geen kunst is: je zet een aantal schakelaars en de tool doet eigenlijk alles voor je.

Ik zag daarbij wel een verschil tussen de beide teams. Klopt, omdat wij voor een deel nog met een webinterface te maken hebben. Bij Magic eDeveloper is nog steeds niet alles geautomatiseerd is en zit je toch met een stuk HTML. Dat wordt wel gegenereerd, maar om die interfaces dezelfde look-and-feel te geven moet toch nog zaken aanpassen. Dat is zal overigens wel verbeterd worden aan de tool: dat we meer met templates gaan werken richting het web. Eigenlijk kun je pas met de introductie van eDeveloper component based gaan ontwikkelen. Het Israëlische team had ook een een aantal templates meegenomen, componenten voor het GUI pad. Op dit moment is het eigenlijk zo dat je voor het web pad echter alleen de batchverwerking in een component kunt onderbrengen. Met ingang van versie 9.3, die voor maart staat gepland, wordt dat uitgebreid.

Maar ook dat zou je kunnen voorbereiden. Klopt, maar dat hadden wij alleen voor printing. Dat was het enige wat wij hadden: het kunnen spoolen in de queuën, het rechtstreeks afdrukken. Dat was als een klein stukje onderdeel van de case.

Beide teams hadden te kampen met een crash, hoe ziet dat er nu uit? Wat Magic met ingang van de komende release gaat doen, is volledig afstappen van het versie controlesysteem en het gaan integreren met een derde partij als PVCS en MVCS, externe partijen die version control en check in- en checkout-mechanismen leveren. Daarbij stappen we volledig ervan af dat zelf gaan doen. Er wordt een interface gebouwd en die zal de stabiliteit moeten gaan brengen van een professionele omgeving. De oplossing die door Magic gemaakt is, werkt in het algemeen in kleine omgevingen tot vijf programmeurs goed, alleen in dit geval hadden we met een heel specifieke situatie te maken waardoor hij crashte. Die zijn we niet eerder tegengekomen, alleen in deze race allebei tegelijkertijd. Hoe het ook zij, Magic is nu overgestapt is naar een third party mechanisme, wat eigenlijk in alle derde generatie-omgevingen gebruikt wordt om dit soort problemen van in- en uitchecken versioncontrol op te lossen binnen Magic. Het komt op korte termijn, waarschijnlijk met versie 9.5, waarschijnlijk tegen het einde van het jaar.

De client server versie heeft duidelijk beter gescoord dan de web-versie. De overgang van versie 7 naar 8 is geen schokkende overgang geweest. Magic heeft toen echter wel de hele architectuur op orde gebracht. Wat je nu met eDeveloper ziet, is dat het gemak voor de ontwikkelaar nu is teruggebracht op een bepaald niveau, de architectuur is op orde en nu is men de kleine gaten aan het vullen om een volledig compleet product te maken. Er mist bijvoorbeeld nog een menustructuur zoals die bij Magic zit. Daarbij moet je denken aan additionele functionaliteiten als templates inbrengen, het kunnen koppelen door middel van Java-functionaliteit en ASP-functionaliteit en een volledig platform worden wat ook bij de .Net strategie van Microsoft past. Volgend jaar moeten de twee versies dan ook eigenlijk even snel kunnen werken.

BIJSCHRIFT: De winnaars van de eerste en tweede prijs feliciteren elkaar. Op de achtergrond Peter Hinssen.

Radventure

'Op het moment dat je moet gaan programmeren, heb je de facto verloren.'

Jullie hebben gewonnen, dus je mag aannemen dat het is meegevallen? Het is ons inderdaad wel meegevallen. De case was heel veelomvattend, maar wel erg goed. Ik vond de case heel leuk, een echte uitdaging om ermee te beginnen. Je weet we dat we ook bijna alles, of alles afhadden, dus dan is het meegevallen.

Er zaten wel een aantal elementen in waardoor je een beetje overdonderd werd. Daardoor werd je gedwongen accenten te leggen en prioriteiten te stellen. Dat vind ik juist wel leuk.

Dachten jullie het eerste uur niet: als we dat maar afkrijgen? Dat wel, maar dan begin je eraan. In het begin moet je de case goed door krijgen. Daar hebben we misschien een foutje gemaakt, want wij zijn echt direct begonnen. We hadden globaal een idee. Later dachten we, dat we die en die vraag hadden moeten stellen, maar we zijn met eigen aannames redelijk ver gekomen. Wij keken elkaar aan en zeiden: kiezen we voor detail of voor alles afkrijgen. We hebben gekozen voor het laatste en we streefden ernaar om de indruk achter te laten, dat je veel kan. Dat is goed gelukt. Die strategie hebben we op die morgen gekozen, omdat we van tevoren niet wisten of het veel of weinig zou zijn. Op die morgen moesten we snel beslissen hoe we het zouden aanpakken. Het is toch de goede strategie gebleken.

Jullie wekten de indruk vaker samengewerkt te hebben Ja, we zijn redelijk goed op elkaar ingespeeld. Om het kwartier overlegden we even wie wat deed, dat ging eigenlijk automatisch. Dat het veel was is dan ook wel prettig, omdat er veel werk te verdelen is, zodat je elkaar niet in de weg zit. Dat is ook het voordeel van een grote hoeveelheid werk, dan is er altijd wel iets om te doen, als je op een gegeven moment gaat stilzitten en de een zegt, ja wat moet ik nu doen, jij bent ermee bezig, ik kan even niks doen, ja dan gaat het verkeerd.

Hoe was de taakverdeling, nadat jullie samen een strategie hadden uitgewerkt? Peter heeft in het begin vooral de conversie van de data gedaan, terwijl ik meteen begonnen ben om het menu en de bestandsgegevens te bouwen. Daarna hebben we de functionaliteit gewoon verdeeld. In de trant van: als jij nu het judge-gedeelte doet, dan doe ik het European civilian-gedeelte. Er waren duidelijke subsysteempjes identificeerbaar: er is een judge part een civilian part, een police part. Ik deel het meestal maar meteen in subsystemen op, dan kan iemand zich concentreren op de functionaliteit van het subsysteem. Dus we hebben het in het begin eigenlijk directe verdeeld.

De opgave liet het wel toe zo te werken. Ja, dat scheelt inderdaad. Dan kan waarschijnlijk niet bij alle opgaven, als je serieel moet werken, dan wordt het veel lastiger, maar dat is ook meer wat voor een programmeerolympiade dan voor een RAD Race.

Hebben jullie daar ooit aan meegedaan? Peter Rakké wel, die is daar ooit tweede geworden. Dat is een keigoede programmeur. Kijk, het is een goed tool, maar er zaten denk ik ook redelijk goede mensen. Die combinatie maakt dat je ver komt natuurlijk.

Jullie hebben eigenlijk het meest van alle teams gebruik gemaakt van een echte RAD tool, waarbij weinig regels code geschreven hoefden te worden. Volgens mij hadden jullie wel een eigen bibliotheek, maar die was niet zo groot. We hebben er wel een paar gebruikt, voor de security bijvoorbeeld. Dat kun je wel aanleggen natuurlijk, maar dan zou je moeten programmeren, we hebben daar zelf een bibliotheek en templates voor gebruikt om het te implementeren, dat was eigenlijk geen werk. Zo werkt dat in Clarion, als je een bibliotheek maakt, maak je ook de manier waarop het moet worden geïmplementeerd in de applicatie. Eigenlijk moet je dan zeggen tegen de applicatie op een heel hoog niveau: gebruik de security. Dan hoef je verder niets te programmeren of te implementeren, dat doet het template. Ja, dat is superkrachtig, om objecten te implementeren hoef je niet te programmeren, alleen maar instellingen te doen, terwijl de code die gegenereerd wordt, optimaal en heel low level is. Terwijl je op heel hoog niveau dingen kan aangeven, genereert het wel een taal die vergelijkbaar is met C++.

BIJSCHRIFT: Het winnende team met beker en bloemen; Peter Rakké en Erik Pepping (r) (Radventure).

Die kennelijk ook een behoorlijke performance heeft. En die ook alles kan. We hadden laatst op een demonstratie iemand die vroeg: ja maar kan dit, kan dit, dat heb ik gelijk afgekapt. Je kan zover gaan als je zelf wil: alles kan. Er zit gewoon een heel low level taal onder. Uiteindelijk kan ik die taal gewoon gebruiken in mijn editor en kan ik alle dingetjes maken die ik wil. In die zin is Clarion heel flexibel en kun je alles bouwen.

Wordt dat in de praktijk veel gedaan, dat je dan toch nog in speciale gevallen nog eens gaat kijken naar de code en aanpassingen gaat maken? Jawel hoor, klanten hebben altijd heel speciale wensen. We hebben bijvoorbeeld voor een borstkankerproject een screening bus applicatie gemaakt met Oracle Live op pc's. Daar moet je dan heel specifiek code voor schrijven en aanpassingen voor maken. Dat moet uiteraard wel, maar steeds minder. Hoe groter je bibliotheek wordt en hoe verder Clarion zelf ook komt met het aanleveren van standaardbibliotheken, hoe minder er geschreven hoeft te worden. Je kunt er inmiddels ook heel veel op doen op de third party market. Als je je datacommunicatie op je comport wilt lezen en bewaken, zijn daar allemaal componenten voor die je zo kunt gebruiken. Je hoeft er eigenlijk niet voor te gaan programmeren.

Dat is wel RAD. Maar niet RAD op de manier van 'quick and dirty' en een applicatie die niets doet. Nee, het heeft daarnaast ook een heel gedegen professionele aanpak met uiteindelijk een heel gedegen resultaat. Dat is ook het krachtige van het tool, vind ik zelf. Het probleem met andere tools was altijd – want we hebben wel een tijdje gezocht – dat je een tool hebt die dingen voor je doet die je niet kunt beïnvloeden. Als ik uiteindelijk iets wil maken – ik wil naar de Windows API, om maar wat te noemen - dan kon dat niet. Onder Clarion ligt een C-achtige taal waarmee alles kan. Er zijn tweeduizend statement geloof ik.

Maar die moet je dan wel leren. In het begin niet, maar wel als je afwijkingen wilt gaan programmeren. Of als je een eigen tool wilt gaan bouwen - dan wel.

Willen jullie volgend jaar weer meedoen maar dan met een andere tool? Met dezelfde tool, maar een andere versie. Kijk, de tool genereert code. Nu is dat Clarion-code, zeer binnenkort ASP-code, en ik hoop binnen de komende zes maanden ook Java. In die zin, dat de Java serverbeans en de HTML outlook niet alleen een zeer specifieke programmeertaal kunnen genereren maar ook een algemene bredere taal. Java is dan toch wel een goed voorbeeld.

Kun je gewoon kiezen: ik wil ASP of Java in plaats van Clarion code? Ja, inderdaad. Clarion is trouwens een taal die een beetje afgekeken is van Pascal, C en Cobol, met wat objectgeoriënteerde elementen.

Wanneer wordt dat precies afgeleverd? ASP binnen enkele weken, Java in juni of juli. Er is gekozen voor de Sun J2EE-edition. Daaraan gaan we ons conformeren omdat met name Oracle, IBM en Sun zich daaraan hebben geconformeerd. Het blijft redelijk puur en we gaan geen Clarion-extensies aanmaken, bij wijze van spreken, om het mooier of anders voor te doen dan het is. Het moet zich bewijzen natuurlijk, maar als je wilt scoren en je wilt ook op Unix gaan draaien en op Mac en op Windows dan zul je het redelijk puur moeten houden natuurlijk. Daarmee verlaat je overigens ook een heleboel dingen.

Hoe vond je de overeenkomst tussen de case en werkelijkheid? Ik dacht dat het een werkelijke case was; dat vind ik echt knap. Ik vond het heel reëel. Ik had echt zoiets: ik ga de case gebruiken voor trainingsdoeleinden, want hij is zo leuk.

Je zit al lang in het vak, ben je nog begonnen met Cobol? Ja, mijn eerste opdracht was het maken van een Cobol programmagenerator, dus je kunt wel nagaan dat het genereren van code voor mij altijd al 'the way' was. Een programmeur is lui en ik haat het om routinematig werk te doen. Met Clarion haal je de routine eruit en blijft de sjeu van het programmeervak toch een beetje overeind. Overal waar je zegt dat moet ik een tweede keer doen, daar maak je een template van en dan zeg je: dat hoef ik dus nooit meer te doen. Ik heb me vanaf het begin af aan bezig gehouden met programmageneratoren. Toch heb ik nooit iets gevonden waarbij ik niet dacht: leuk bedacht, maar beperkt. Tot ik Clarion vond. Het concept staat als een paal boven water. Het is alleen jammer, dat het er nog niet van is gekomen die generator wat meer algemeen goed te maken. Het blijft een beetje een niche tool. Er zit geen Microsoft achter en geen Oracle; er zit een stel techneuten achter die ooit, een jaar of acht geleden, bij Borland zijn weggelopen. Voor de helft komt het uit de Borland-hoek, qua compiler en linker. En de andere helft, de capaciteit generator, kwam uit de hoek van oud-Top Speed. Die zijn ooit samen gegaan. Ik had heel graag gehad dat er een naam als Borland achter gezeten had of Oracle, of Microsoft puur vanwege de naam.

Tips voor de mensen die nu niet gewonnen hebben? Lachend: 'Dé tip is denk ik: Clarion gebruiken. Maar serieus: ik zag mensen toch nog veel programmeren. Als producten niet met veel componenten komen, haal ze dan uit eigen ervaringsprojecten en neem ze mee. Op het moment dat je moet gaan programmeren, heb je de facto verloren. Wij hadden aan het einde van de tweede dag driehonderd regels code, ja dat kan in twee dagen. Maar als je een inlogscherm gaat coderen, dan zit je op de verkeerde wedstrijd.

Tekst en fotografie Dré de Man