de versie daarvoor beschikbaar zal zijn van open source groepen. MySQL werkt ook zo, de commerciële versie is n, en n -1 is GPO.'

Het gevaar van open source lijkt mij dat er afscheidingen ontstaan (forking) die elkaar in de weg staan.

Phipps: 'Forking is geen probleem, de vrijheid om te forken is de kracht van open source, als ik het fout doe, ben jij altijd vrij om het goed te doen. Het probleem is niet community-forking, het probleem ontstaat wanneer een monopolist ervoor kiest om te forken uit marktoverwegingen. Binnen de Java-ruimte zijn er hele grote ondernemingen die een fork zouden kunnen creëren zonder dat de gemeenschap daartoe noodzaak ziet. Daarom moeten wij voorzichtig zijn. Jini, Juxta, open Solaris, we hebben niet eens gepraat over Sun-grid. Dat is gebaseerd op de open source grid engine, de grid engine draait bovenop open Solaris of bovenop Linux. Onze gehele business-strategie is gebaseerd op open source. Sun begrijpt open source, het gaat om het werken met community's en partnerschappen met andere slimme mensen om een win/win situatie te creëren. Sun is gecommitteerd aan partnerships en community's want het profiteert door het verkopen van de services en de apparatuur aan die gemeenschappen en partner-side die nodig hebben om te slagen, dat is ons business-model.'

Spring maakt J2EE eenvoudiger

Gesprek met Rod Johnson en Alev Arends

Spring is een open source Java-framework, dat beoogt J(2)EE programmeren eenvoudiger te maken. Het ontstond uit de code die gepubliceerd werd met het boek Expert One-on-One J2EE Design and Development geschreven door Rod Johnson. De auteur van het framework en het boek legt samen met zijn Nederlandse interface21 collega Alev Arends uit waarom Spring zo populair geworden is.

Expert One-on-One J2EE Design and Development werd gepubliceerd in oktober 2002. Het werd geleverd met een sample-applicatie die geschraagd werd door 14.000 regels code. Deze bood oplossingen voor veel voorkomende problemen die Johnson keer op keer tegenkwam in J2EE-projecten. Er was dus veel belangstelling in het opensourcen ervan en in het begin van 2003 begon het leven van Spring als open source- project. In maart 2004 werd de Spring 1.0 final release uitgegeven, in mei 2005 de 1.2 final release. Welke zijn nu de voordelen van Spring?

Johnson: 'Een belangrijk probleem in veel J2EE-projecten was dat men steeds opnieuw dezelfde functionaliteit bouwde. In de beginjaren bouwden mensen bijvoorbeeld hun eigen logging infrastructuur en geleidelijk is er een beweging ontstaan naar goede generieke frameworks die generieke oplossingen voor generieke problemen aanbieden. Natuurlijk levert dat een heel groot productiviteitsdividend op. Het betekent namelijk dat developers in ondernemingen zich concentreren op de kernactiviteiten van dat bedrijf en de business-logica die specifiek is voor dat bedrijf, in plaats van het oplossen van problemen die ieder ander heeft en die maar een keer opgelost zouden hoeven te worden op een effectieve manier. Natuurlijk krijg je, wanneer je je als community focust op het leveren van goede oplossingen voor veel voorkomende problemen, veel meer ervaring en feedback rond die oplossing dan wanneer ieder bedrijf zijn eigen oplossing schrijft. Spring levert bedrijfsoplossingen die onafhankelijk zijn van het bedrijf of de branche in kwestie. Daardoor kunnen developers zich concentreren op kernproblemen en de infrastructuur overlaten aan het framework. Dat is waarschijnlijk het eerste belangrijke voordeel.

Het tweede is, dat Spring software-ontwikkeling in feite een stuk simpeler maakt. Spring is ontworpen om nogal wat complexiteit binnen het framework op te nemen. Traditioneel in standaard-J2EE, ben je verplicht je transactiemanagement te doen, je bent verplicht het J2EE-model dat een behoorlijk niveau van complexiteit oplegt op het programmeermodel en ook het development-proces. Met Spring ligt de nadruk op het mogelijk maken de applicatie op te bouwen uit Pojo's. Spring levert die voordelen op zonder dat je mogelijkheden prijs moet geven. Het levert een complete oplossing die simpel is in zijn traditionele aanpak.'

Testen Johnson: 'Een derde groot voordeel is dat Spring een veelomvattende oplossing biedt om de testmogelijkheden te vergroten. Allereerst wordt door een Pojo-gebaseerd programmeermodel te leveren, unit-testing erg vergemakkelijkt, zodat het bijvoorbeeld mogelijk wordt om effectief testdriven development te doen. Spring gaat echter nog een stap verder door krachtige integratie testondersteuning te bieden. Je kunt bijvoorbeeld tests schrijven die een development staging-database gebruiken, je code uitvoeren tegen de echte enterprise services die het moeten doen, je hele Spring-configuratie testen, zonder de noodzaak op een applicatieserver te deployen. Dat kan veel sneller zijn dan een traditionele aanpak, waar een heel groot risico van incorrecte code was omdat het meestal te langzaam was om het

gedurende development te testen.

Zo deden wij een project in de financiële sector waar we bijna tweehonderd integratietests hadden, dat een vrij complex datamodel in Oracle betrof, waarin stored procedures, triggers en views gebruikt werden. Met behulp van Spring waren wij in staat al die integratietests te doen in minder dan een minuut. Dat betekende dat voortdurende integratie vergemakkelijkt werd en ondersteund werd, zodat ontwikkelaars in staat waren alle unit-integratietests door te voeren voordat ze code aan het version control system gaven. Dat lost echt een van de kernproblemen op die ik in het verleden gezien heb, in J2EE-applicaties waar in wezen bouwen en testen veel te veel tijd kost.

Dat betekent dat er vaak incorrecte code is die het werk erg vertraagt, omdat er geen manier is om de implicaties van het invoeren van een stuk code helemaal te testen. Natuurlijk is dat nog belangrijker wanneer code in productie gaat en onderhouden wordt, omdat het van levensbelang is om een sterk snel regressie-testsysteem te hebben. Je kunt onongelofelijk grondige tests hebben, maar als de uitvoering ervan drie of vier uur duurt dan wordt de waarde ervan erg beperkt, omdat het te lang duurt voordat de feedback komt. Wij geloven dat met Spring een testmanier mogelijk wordt waarbij *'you can have your cake and eat it'*, je kunt hele grondige tests doen en je kunt ze heel snel uitvoeren.'

Arends: 'Dat is niet alleen belangrijk tijdens onderhoud, maar ook tijdens de ontwikkelfase, waarbij je snel kunt reageren op requirements-wijzigingen.'

CONSISTENTIE Johnson: 'Een ander belangrijk voordeel van Spring is dat het een consistente aanpak mogelijk maakt. In typische Spring-projecten zien we een veel grotere consistentie dan in typische J2EE-projecten. Er is een consistentie in configuratiemanagement, er is ook een consistentie in lagen-architectuur waar Spring best practice faciliteert. Wij hebben veel nadruk gelegd op consistentie in het framework zelf, wat betekent dat als je kijkt naar de manier waarop Spring abstracties biedt over een aantal belangrijke api's, zoals voor persistentie, JDBC, Hibernate, JDO en TopLink, zelfs wanneer we verder kijken dan persistentie naar gebieden als JMS, JTA, JNDI, JavaMail, zien we consistentie in de manier waarop Spring klassen en interfaces gebruikt en omgaat met exception handling over al die gebieden. Dat betekent dat ontwikkelaars de maximale waarde uit de leercurve halen.

Een vijfde voordeel van Spring is dat het geneigd is je code van zijn omgeving los te koppelen. Een goed voordeel daarvan is remoting. Traditioneel ben je daarmee in J2EE sterk afhankelijk van api's als de EJB api. Als je dus een remote client voor een EJB wilde schrijven, waren er dus afhankelijkheden van de JNDI en de EJB api's. Omdat het dependency-injectieconcept look-up compleet externaliseert van je eigen code, ben je alleen afhankelijk van de service-interface waarin je geïnteresseerd bent. Dat betekent dat nu zonder enige verandering in je Java-code, je toegang hebt tot je EJB, je webservice, een lokaal object, en test-stub gedurende unit testing, een remote object via een van de vele andere remoting-protocollen die Spring ondersteunt, en je code is minder bewust van zijn runtime omgeving en dus minder kwetsbaar voor veranderingen daarin. Als je een aankoopservice hebt, en de manier waarop dingen gekocht worden verandert, dan verandert je code ook. Maar als alleen de manier waarop je die service aanspreekt verandert, dan niet. Dus het idee van het abstraheren van de specifieke implementatie aspecten is een van de kernvoordelen van Spring.

Een zesde voordeel van Spring is dat het een uitstekend integratieplatform biedt. Er zijn daarvan twee hele goede illustraties. De ene is de groei van ander open source projecten rond Spring. De integratie van Lucene, OSWorkflow en tevens de integratie van Spring binnen JasperReports. Een ander voorbeeld is wat Oracle recentelijk heeft gedaan met Toplink. Spring voorziet in een data-abstractielaag die voor verschillende api's geïmplementeerd wordt. Oracle heeft een implementatie voor Toplink gemaakt en die gedoneerd aan het Spring framework wat wij als een zeer aardig gebaar naar de open source gemeenschap zien. Het is ook belangrijk

omdat belangrijke vendors het belang van Spring als integratieplatform inzien.'

INNOVATIE *Spring bestaat al een tijd, wordt nu zeer populair, maar dat betekent niet dat het werk af is.*

Johnson: 'We willen doorgaan met een snelle innovatie. Vanwege de uitbreidbare architectuur denken dat we goed in staat zijn om nieuwe functionaliteit op nieuwe gebieden te leveren, zonder de uistekend reputatie op het gebied van stabiliteit te verliezen. Er is een aantal nieuwe features in spring 1.2 waarvan de community voordeel zal hebben, dingen als JMX-integratie zonder de noodzaak je code aan te passen, in 1.3 kijken we naar features als extra opties voor xml-configuratie die niet alleen de config van typische applicaties zullen vereenvoudigen, maar ook ideaal zullen zijn om third party producten met Spring te integreren. Zo'n product zou een xml-schema namespace kunnen leveren, dat tags definieert die hun product configureert voor Spring. We zijn ook geïnteresseerd in het dynamisch reconfigureren van draaiende applicaties waarvan we vinden dat J2EE daar nu goed in voorziet. Na 1.3 zijn we er ook in geïnteresseerd om het Spring Pojo-model naar gridcomputing te brengen, een gebied dat van groot belang zal zijn voor de financiële wereld en verder willen we support voor andere talen dan Java binnen Spring-applicaties. We hebben al code die dit ondersteunt maar die is nog niet officieel vrijgegeven. Daarmee zouden ontwikkelaars code kunnen schrijven in, een scripttaal als Groovy met behoud van alle voordelen van Spring. In essentie hebben we in de eerste twee jaar van het Spring-project hebben we goede oplossing gegeven voor veel voorkomende problemen in J2EE in de loop daarvan hebben we best practices duidelijk vooruit gebracht. Nu gaan we verder met innovaties die verder gaan dan de grenzen van het J2EE-model. J2EE heeft veel opgeleverd, maar op het gebied van grid-computing en batch processing levert het geen bijzonder geschikte oplossing, bijvoorbeeld voor de problemen die banken hebben.'

Met JBuilder in het Eclipse peloton

JBuilder vanaf volgend jaar gebaseerd op Eclipse

JBuilder werd tot nu toe gebouwd op een eigen framework, vergelijkbaar met NetBeans en Eclipse. In de afgelopen jaren heeft Eclipse enorm aan populariteit gewonnen, ook ten koste van JBuilder. Waarschijnlijk volgens het principe ' If you can't beat them...', is nu besloten wel over te gaan naar Eclipse. Is het dan niet toch een beetje een kwestie van het moede hoofd laten hangen? Borland's VP software David Intersimone geeft een toelichting.

DavidI: 'Ik zeg altijd tegen mensen: kijk naar de capaciteiten van JBuilder, dat komt neer op alles wat we de afgelopen tweeëntwintig jaar gedaan hebben. Meer productiviteit van zowel individuele developers als van teams. De volgende versie van JBuilder zal het mogelijk maken om developers die geografisch van elkaar gescheiden zijn dezelfde blik op sourcecode te geven. Twee ontwikkelaars zouden dezelfde source-code kunnen debuggen, en eentje zou in India kunnen zijn en de andere in Nederland of in de kamer naast de jouwe. Het ontwikkelen wordt meer en meer samenwerken. Of je nu een *pair programming* aanpak volgt of wat dan ook, je kunt gewoon twee JBuilders met elkaar verbinden. De volgende versie van JBuilder zal weer meer samenwerkingsmogelijkheden hebben, de roadmap laat dat ook zien.'

Borland zal ook iets winnen, want veel code hoeft nu niet meer onderhouden te worden, die zit nu in Eclipse.

DavidI: 'Ja, die developers kunnen nu werken aan meer productiviteitswinst en anders aspecten voor ontwikkelaars.'

Kunt u een schatting geven van de hoeveelheid code van JBuilder die betrekking heeft op het framework, code die nu door Eclipse onderhouden zal worden?

DavidI: 'Ik heb geen precieze gegevens daarover, maar ik schat minstens zestig procent, waarschijnlijk nog meer. Het framework, de editing, de compiler en al die dingen, het is misschien nog wel meer. Er werkten tien engineers aan *Prime time* maar dat was alleen het kerngedeelte. Er waren andere mensen op Star Team, Together, Janeva, die allemaal die code schrijven als interface naar *prime time* (de interne naam) én naar Eclipse. We zullen ook het voordeel krijgen dat alle plug-ins voor Eclipse zullen werken voor al onze producten. Voor de third party designer side-tools en component designers is er dus ook een voordeel, die hoeven alleen nog maar te schrijven voor de Eclipse plug in interface.'

Er ontstaat nu meer competitie maar het vergroot misschien ook de markt voor JBuilder.

DavidI: 'Jazeker. Je kunt natuurlijk de standaard-omgeving van Eclipse nemen en daar iets aan toevoegen. SAP doet dat met bijvoorbeeld met Abap. Wij richten onze