Interview

Spring is een van de meest bekende open source frameworks. Het werd bekend in de jaren dat de kritiek op J2EE grote vormen aannam en bood daar een eenvoudiger alternatief voor. In de loop der jaren nam Spring in omvang en populariteit toe. Daarmee groeide echter ook de kritiek op Spring. Java Magazine sprak met Alef Arendsen over Spring én over zijn lange geschiedenis ermee.

'Open source is middel, geen doel'

Alef Arendsen en Spring

Alef Arendsen begon ooit met een klein beetje sterrenkunde gecombineerd met een HBO-studie industriële automatisering. Daarna werd het informatica, maar al studerend startte hij met huisgenoten een bedrijfje en in het begin van 2000 – vlak voor de crash – begon hij bij Smart Haven een DotCom startup.
Arendsen: "We gebruikten daar Java om met kunstmatige intelligentie via internet profielen op te stellen van mensen op basis van hun zoektermen. Een van de mogelijke toepassingen zou een recommendations engine als die van Amazon kunnen zijn. We wilden daar een generieke engine voor bouwen, die we niet alleen voor zoeken, maar ook voor allerlei andere dingen als on line shopping en dating kunnen inzetten."
"Ik heb daar in vrij korte tijd heel veel geleerd. Maar op een gegeven moment stortte de Business-to-Consumer markt in, de Business-to-Business markt bleek ook niet voldoende houvast te bieden en vervolgens is de zaak overgenomen door Quote Media. Begin 2002 heb ik met Joost van de Wijgert JTeam opgezet met als eerste klant SmartHaven. We werkten zowel bij JTeam als bij SmartHaven vrij veel met open source technologieën, daar waren we vrij vroeg mee in de markt. Op een gegeven moment las ik over een technologie die net was omgedoopt tot Spring. Ik heb een soort vijfminutentest: als ik het in vijf minuten kan downloaden, drie regels documentatie kan lezen en het snap – en ik zie de potentie ervan – dan ga ik er verder naar kijken. Al heel snel bleek dat het een heel interessante technologie was en toen ben ik gaan helpen met de ontwikkeling ervan. Toentertijd hadden we een testcoverage van vijftig procent en veel te weinig documentatie en ik ben een aantal dingen gaan oppakken. Ik heb een groot gedeelte van de documentatie zelf geschreven, de Unit Test codebase op peil gebracht en zo ben ik bij het Springframework betrokken geraakt."

Veel gebruikt

Bij JTeam werd Spring veel gebruikt en dat leidde ertoe dat vanuit Nederland veel werd bijgedragen aan het Spring framework, ook door Arendsens collega Joost van de Wijgerd. Arendsen schreef veel documentatie en waarschijnlijk is nu nog steeds de helft van de documentatie van zijn hand.
"Daar heb ik vele avonden achter elkaar aan gewerkt. Buiten het feit dat ik technisch gezien wel een kikker van de kant kan duwen, zoals men dat zegt, was mijn insteek ook: als je aan een open source-project meewerkt en je wilt er ook voor zorgen dat het succesvol wordt, dan moet je niet alleen kijken naar de technische aspecten. Je moet ook ernaar kijken hoe je een community opbouwt die blij is met je product, goede documentatie heeft, het makkelijk kan gebruiken en vertrouwen heeft in het product."

JTeam was begin 2003 al bij Spring betrokken, dus voordat de code final werd. Maart 2004 kwam Spring 1.0.

Dré de Man
is freelance journalist

"Als ik iets in vijf minuten kan downloaden, drie regels documentatie kan lezen en het snap en ik zie de potentie ervan, dan ga ik er verder naar kijken."

"Halverwege 2004 had ik ondermeer Rod Johnson, Jürgen Höller, Colin Sampaleanu en Keith Donald veel gesproken op conferenties. We kwamen tot de conclusie dat er gezien de snel groeiende populariteit van Spring in Nederland en in Amerika behoefte was aan wat diensten eromheen. Wij deden al workshops in Nederland over Spring en daarnaast hadden we consulting op het gebied van Java en J(2)EE. Begin 2005 zijn we Interface 21 gestart, met het idee om consulting en training en additionele dienstverlening rondom Spring te verlenen. Het was een vrij internationaal gezelschap. In Nederland zijn Steven Schuurman, Arjen Poutsma en ik van JTeam overgegaan naar Interface21."

"Zelf ben ik toen vooral bezig geweest met Interface21 in Nederland en Europa, met name Scandinavië, met training en trainingmateriaal, veel meer ondernemen dan techniek. Maar ondernemen is een van de dingen die ik ook heel graag doe. Op het moment dat het dan staat, verlies ik mijn aandacht echter al gauw en dan moet er weer wat nieuws gebeuren. Ook heb ik gesproken op heel veel conferenties. De afgelopen twee, tweeënhalf jaar heb ik de halve wereld wel gezien."

En dat verveelt nog niet?

"Het toeval wil dat ik net gestopt ben bij SpringSource (zoals Interface21 tegenwoordig heet). Ik ben daarvan één van de aandeelhouders, dus ik blijf wel actief betrokken, maar mijn activiteiten voor het bedrijf ga ik afbouwen. Ik heb er lang over gedacht. Nee, het verveelt nog niet, maar het bloed kruipt waar het niet gaan kan. Ik mis een stukje ondernemen binnen de context van SpringSource. Ik werk vanuit huis en als ik daar niet ben, dan zit ik in het buitenland. Dat kan heel erg prettig zijn, maar iets op een langere termijn opbouwen met een team heeft toch ook wel wat. Ik ga weer parttime bij JTeam aan de slag, als strategisch adviseur. Dus vervelen, nee, dat niet, maar het is denk ik wel beter om nu te stoppen, nu het nog leuk is."

Spring is een van de meest succesvolle open source-projecten. Waarom?

"Ik denk dat je ook moet kijken naar wat voor projecten het zijn. Log4J en Struts 1.0 waren natuurlijk extreem succesvol, JBoss ook, maar dat het zijn toch weer heel andere projecten. Wat Spring onnderschiedt, is dat we probleemoplossend gericht werken. Er is een aantal problemen of uitdagingen binnen de Enterprise Java markt; laten we daar een framework voor neerzetten. Bij Spring luisteren we naar wat mensen willen en we destilleren de vragen uit de markt. Als je kijkt naar Google Guice, de implementatie is echt prima. We hebben daar ook inspiratie door opgedaan. Wij gaan zelf iets doen à la Google Guice. Maar je ziet dat het een klein aspect van het hele probleem aanpakt. De hoeveelheid frameworks in Java is zo groot dat het voor de wat meer mainstreamontwikkelaar heel moeilijk is om door de bomen het bos te zien. Wij proberen wel op allerlei manieren aan de ene kant de community levend te houden en aan de andere kant toch een brand neer te zetten met een platform dat consistent is waarmee we ontwikkelaars iets vertrouwds kunnen bieden. Op het moment dat je Spring zelf kent, is het vrij eenvoudig om met Spring Integration aan de slag te gaan, net zoals met een ander product uit de Spring projectlijst. Dat is zeker een

“We sluiten niet uit dat we ooit een EJB 3-implementatie gaan bouwen, of dat we een van de profiles van JEE 6 gaan implementeren.”

onderdeel van het succes. Google Guice en Struts 2 zijn goede implementaties maar ze lossen een deelprobleem op. We willen niet alles oplossen, absoluut niet. Maar ik denk dat je uiteindelijk wel een vrij consistent coherent geheel nodig hebt, wel modulair opgezet, wel los van elkaar te gebruiken, waarmee je ook een soort van one stop shop kunt zijn.”

Dat roept bij mij twee vragen op: als je zoveel dingen wilt doen, kun je dan niet beter maar deel proberen te worden van de Java-specificatie? En hoe ver zijn jullie nu verwijderd van het idee dat je het in vijf minuten kunt downloaden en uitproberen?

“Als je naar individuele onderdelen kijkt, denk ik zeker dat het nog kan. We zijn nu op de blog een serie artikelen aan het releasen: … in tien minuten, te beginnen met Spring Integration in ten minutes. Volgens Mark Fischer is het Venus-tijd, dus eerder vijftien of twintig minuten, maar binnen heel korte tijd legt het uit wat het is, met codevoorbeelden etcetera. Maar Spring is wel groter geworden. JEE-ontwikkeling is echter gewoon complex, daar moeten we niet voor weglopen en ten tweede is er natuurlijk in de technologie altijd een wet van de remmende voorsprong, die je zo veel mogelijk moet proberen te voorkomen. Dat doen we ook. In Spring 3 gaan we een heleboel dingen depreciaten en die gaan we in latere releases echt weghalen. Grappig voorbeeld: we hebben een goede sample app die redelijk representatief allerlei features van Spring gebruikt en daarin meten we het aantal regels code en de complexiteit. Je ziet bij iedere versie het aantal regels code en de complexiteit nog steeds naar beneden gaan. Maar het is zeker complexer dan het was, ook al proberen we dingen zo simpel mogelijk te houden.”

“Het specificatieverhaal is interessanter. We hebben nooit gezegd dat we geen onderdeel willen zijn van een specificatie of iets dergelijks. We willen echter zeker geen compromissen aangaan. Volgens ons en volgens een deel van onze community is er een aantal specificaties waarin we vinden dat er compromissen gemaakt zijn. Kijk maar naar EJB 3.0. We hadden daar op zich graag zelf een rol in gespeeld.”

Je zou ook kunnen zeggen dat Spring niet meer zo nodig is nu EJB 3 er is.

“Daar kun je een hele lange discussie over voeren, maar de scope van Spring is veel breder dan EJB 3. Met EJB 3 kun je geen integratie doen, het biedt je niet de functionaliteit van een ESB, je kunt er nauwelijks batchprocessen, en offline processen mee implementeren. Maar het belangrijkste is dat we zoveel mogelijk compromisloos te werk willen gaan. Aan de andere kant zie je dat we wel meewerken aan een aantal specificaties, bijvoorbeeld aan OSGI en we zijn ook betrokken bij een aantal specificaties die wel bij de JCP zitten. Rod Johnson is voor SpringSource sinds kort zelfs lid van het JCP Executive Committee for Java SE/EE. Maar voor ons is Spring meer dan dat, want Spring ligt ook ten gronde aan grote delen van de basisfunctionaliteit van Weblogic. Als we Spring in de standards body stoppen, dan wordt het waarschijnlijk een technologie die dermate veranderd is dat hij niet meer bruikbaar is in bijvoorbeeld WebLogic. We sluiten ook niet uit dat we ooit een EJB 3.0-implementatie gaan bouwen of een van de profiles van JEE 6 gaan implementeren. Als we daar waarde in zien en de Springcommunity daarom vraagt, dan is het een probleem dat om een oplossing vraagt en dat probleem zien we op dit moment nog niet sterk genoeg.”

Sprekend over WebLogic, hoeveel samenwerking is er nu met Oracle op dat gebied?
"Er is absoluut vrij veel samenwerking tussen een aantal belangrijke mensen van Oracle en het vroegere BEA en ons. Er zijn nog steeds committers in bepaalde projecten die daar vandaan komen. Ook op commerciële basis werken we af en toe samen."

Maar verder nog niets?
"Ja, jij denkt natuurlijk net als iedereen: wanneer worden ze nu eens opgekocht. Ik denk dat het nu helemaal niet aan de orde is. Spring is op dit moment zo groot gegroeid, dat we op het moment dat wij ons met een vendor zouden verbinden we onszelf ook heel erg zouden gaan limiteren. Uiteindelijk hebben we aandeelhouders en investeerders, dus ik kan niet zeggen wat de toekomst brengt. Maar we zijn op dit moment bezig om zelf een gezond bedrijf neer te zetten en het gaat erg goed."

Hoe zit het licentietechnisch eigenlijk met al die onderdelen?
"De core frameworks, Spring, Spring integration, Spring web services, Spring Webflow, vallen onder de Apache-license, dat is de meest business-vriendelijke licentie die er is. Dat laten we zo. Daarnaast hebben we een aantal producten neergezet die wat meer opinionated zijn. Er is een heel simpele regel: als er Spring voor staat is het Apache-licensed of een andere businessvriendelijke licentie. Als er SpringSource voor staat, moet je even kijken wat de licentie is. Dan kan het GPL zijn of een commerciële licentie. De commerciële producten zijn zo ingericht dat op het moment dat je die weg zou halen, je nog steeds zonder zou kunnen. Wanneer je op dm-server deployt, dan zijn de dingen die je deployt gewoon OSGi-bundles. Dat moet je op zich op elke OSGi-container kunnen deployen. Dat wil niet zeggen dat dm-server geen extra features biedt, maar er ontstaat een minimale vendor lockin."

Toch blijft het altijd de vraag bij open source frameworks, waar je naartoe wilt.
"We hebben wel heel duidelijk gezegd dat we een plaats willen zijn waar we samen met de community de infrastructuurprovider kunnen zijn waarmee je eenvoudig enterprise Java-applicaties kunt bouwen. Op dit moment doen we dat onafhankelijk, maar ons doel is een zo simpel mogelijk technologieplatform te ontwikkelen waarmee je eigenlijk alles op dat gebied kunt coveren. Dat doen we met behulp van de community en dat zijn onze twee belangrijkste dingen."

Wat is de technische toekomst van Spring?
"Laatst gaf John Rymer van Forrester een keynote op een conferentie van ons en zei dat lean en lightweight software wel heel erg belangrijk wordt. Zijn stelling was: applicatieontwikkeling gaat om complexiteit en hoe lager de complexiteit, hoe beter het is. Niet alleen in code, maar ook qua resource-consumptie, in termen van de footprint etcetera. De tijd dat we zeiden, gooi er maar meer hardware ertegenaan, die is voorbij."

'De tijd dat we zeiden, gooi er maar meer hardware ertegenaan, die is voorbij.'

Maar veel mensen denken nog zo.
"Ultiem voorbeeld: Amazone cloud services. Dat betekent gewoon dat je afrekent per resource-consumptie. Beheer en operatie kosten veel geld."

Waar het einde van Moore's law nog bij komt, met één kern kun je niet meer veel sneller.
"Ja, je moet dus op een andere manier gaan programmeren. Dat lighweight lean verhaal past daar heel goed bij. Op het moment dat je een simpele ESB wilt neerzetten met in memory asynchronous communication, dan kun je dat met Spring Integration in een heel kleine applicatiecontext gewoon doen. Daar hoef je geen server voor op te starten. We richten ons erop om die filosofie - lean, minder complexe software, minder complexe infrastructuur - in al onze projecten en al onze producten door te voeren. De basis daarvan zijn alle Spring-projecten, Spring zelf en Spring webservices. Daar zie je dat allemaal terug en wat we daar bovenop bouwen – de SpringSource-producten – zijn platformen die het gemakkelijk maken dat soort dingen te deployen. Dat doen we me de Toolsuite ook. Daarmee kun je bijvoorbeeld heel gemakkelijk deployen naar TomCat. We denken erover na hoe we daar meer opties aan toe kunnen voegen, dus ik kan me zomaar voorstellen dat we in de toekomst heel gemakkelijk gaan deployen naar Amazon EC2. We zijn dus bezig om naar het volgende niveau te kijken, daarom hebben we ook een partnerhip met VMWare gesloten."

Spring heeft heel enthousiaste aanhangers, maar er ook veel fanatieke tegenstanders. Een van de redenen is dat binnen Spring veel opgelost wordt met configuratie en XML in plaats van met code. Een ander is het feit dat het geen deel uitmaakt van de Java-specificatie.
"Er zijn verschillende groepen die iets tegen Spring hebben. Er zijn mensen die iets hebben tegen configuratie in XML. Die vinden dat niet handig. Daar hebben we zelf al van gezegd: naarmate meer mensen dat zeggen, gaan we ernaar kijken. In Spring3 gaan we daarom een Java-gebaseerde configuratie-oplossing bieden, Javaconfig. Straks heb je dus nog maar een heel klein XML-document nodig en de rest kun je dan in Java doen met dezelfde feature-set die je in XML ook hebt. Het heeft wel heel lang geduurd voordat we het onderkend hebben, maar dat komt omdat we er geen compromissen in willen sluiten. Niet alles zal in Java kunnen, want sommige dingen zijn er niet voor geschikt, maar het grootste gedeelte wel. Daardoor is het veel meer refactory-vriendelijk en veel meer type safe. Dan zijn er nog mensen die om allerlei andere redenen Spring niet mogen, omdat het te groot is, omdat het geen standaard is, omdat het gerund wordt door een partij die ook geld wil verdienen, maar daar kunnen we niets aan doen."

Die laatste kritiek doet me denken aan de discussie op theserverside van een paar maanden geleden, nadat SpringSource een wijziging aangekondigd had met betrekking tot maintenance policies.
"Ik zat zelf op dat moment in Italië en ik zag op vrijdagavond die discussie en had geen internettoegang. Op dinsdag moest ik in Stockholm een lezing geven en dat was de eerste openbare presentatie van iemand van SpringSource na die wijziging. Bijna alle vragen gingen over de maintenance policy. We hebben de reactie van de community onderschat. Ik heb gezegd: ik kan nu niet beloven dat de dingen anders gaan, maar de community is voor ons belangrijk, dus uit je bezwaren en doe dat op een reële manier. Dat was ook iets waar Rod Johnson, ik en anderen pissig over waren. Dan komen er meteen weer de verhalen van: 'ze willen alleen geld verdienen'. Mensen doen onmid-

“Ik denk dat je uiteindelijk wel een vrij consistent coherent geheel nodig hebt, waarmee je een soort van one stop shop kunt zijn.”

dellijk heel veel aannames en dat is op zich ook begrijpelijk. We hebben toen uitgelegd dat we het onderhoud niet meer kunnen blijven doen tot in lengte van dagen. Twee weken later hebben we de maintenance policy nog ene keer grondig aangepast. Het is nu als volgt: we ondersteunen de majorversies, totdat de eerste release-kandidaat van de volgende majorversie uit komt. We onderhouden het dan nog wel, maar we leveren de builds niet meer uit. Dat geeft mensen een incentive om steeds naar de laatste versie te upgraden. Willen mensen dat niet en willen ze gegarandeerde bugfixes support, dan moet je gaan denken aan het nemen van een support-contract. Dat lijkt ons ook alleszins reëel.”

Dat denk ik ook wel. Alleen denk ik dat Rod Johnson met zijn argumenten dat sommige gebruikers alleen willen profiteren van open source, olie op het vuur gegooid heeft.

“Mogelijk, maar open source is ook niet per definitie gratis. Ergens moet het onderhoud, de ontwikkeling en de ondersteuning van betaald worden. Het gaat om het samen met de community ontwikkelen van software die beter is dan closed source. Kijk, als het allemaal gratis is en nergens iets oplevert dan komt er een moment dat je denkt: ‘waar doe ik het allemaal voor?’. Ik werkte negentig tot honderd uur per week toen we Spring aan het bouwen waren. Niet alleen aan Spring, maar toch. Er moet wel ergens een bedrijfsmodel achter zitten. In Amerika heeft men daar minder moeite mee dan in Europa. Voor ons is het ook een middel, geen doel. Het is voor ons de enige manier om goede software te bouwen: met de community.”

‘De enige manier om goede software te bouwen, is samen met de community.’

Gebruiken jullie één bepaalde methodologie?

“Nee, die hebben we eigenlijk niet echt. Ik ga niet zeggen dat als je goede mensen hebt, je geen methodologie nodig hebt, maar daar neigt het wel heel sterk naar. De noodzaak van een heel formeel proces wordt wel minder naarmate je meer met cracks werkt.”

Maar hoe gaat het opstellen van requirements bijvoorbeeld?

“Dat zit allemaal in het bug-trackingsysteem. Verder heeft elk project zijn eigen projectorganisatie en die bepalen zelf hoe ze werken. Het dm-server team is heel erg gebaseerd op Scrum, terwijl onze mensen heel soms bij elkaar komen en verder overleggen via Skype. Wel werken we sterk testdriven, dus alles wat er gemaakt wordt, wordt meteen getest met unit tests en integratie-tests en daar komt de community ook weer bij kijken. Die gaat de software meteen gebruiken.”

Dus de community is het proces?

“Ja, ik denk ook niet dat je er een formeel proces van kunt maken. Dat hebben we ook niet heel erg nodig, omdat die mensen die aan de software werken donders goed weten hoe ze dat moeten doen. We maken ook geen heel formele procesbeschrijving. Het enige dat van belang is, is dat de ontwikkelaars de community gebruiken en er snel en veel over communiceren.”

“Maar het hangt ook in hoge mate samen met de kwaliteit van de mensen. We hebben echt heel goede mensen in dienst en de noodzaak voor een heel formeel proces wordt wel minder naarmate je echt met cracks werkt. Lees de mythical man month er maar op na. Er zijn programmeurs die honderd keer zo effectief zijn als de gemiddelde programmeur en daar hebben wij er wel een aantal van in dienst. Daarbij werken we niet met veel mensen. Spring 3 wordt gebouwd met tweeënhalve man. Plus de community uiteraard. Die is echt heel belangrijk.” «