De enterprise omgeving van Java kent sinds lange tijd al verscheidene frameworks voor het ontwikkelen van de webapplicaties. Deze zijn meestal open source of vendorspecifieke oplossingen die het werken met het standaardmodel van servlets vereenvoudigen. Menig Java-ontwikkelaar zal namen als Struts, Tapestry, Spring MVC en WebWork bijvoorbeeld bekend in de oren klinken. Het probleem met deze veelheid aan frameworks was dat er geen echte standaard was waarmee ontwikkeld kon worden. In de afgelopen jaren heeft Struts[4] die rol in zekere mate vervuld.

Java Server Faces

Nieuwe standaard in Java EE 5

Het bestaan van verscheidene andere frameworks naast Struts, geeft aan dat er meerdere oplossingen zijn voor dit type applicaties. Daarnaast was bij het bouwen van een webapplicatie het aantal uren dat besteed werd aan het implementeren van een interface te hoog. Veel van de frameworks boden hier geen afdoende oplossing voor. Om deze situatie op te lossen is sinds mei 2004 een nieuwe standaard toegevoegd aan het Java Enterprise Edition platform: Java Server Faces (JSR 127[5]).

JSF is ontwikkeld om het gebrek aan een framework om component en event gebaseerd ontwikkelen van webapplicaties op te lossen. Hierbij hebben verschillende groepen in de markt, waaronder de ontwikkelaar van Struts Craig McClanahan, samengewerkt om tot een goed werkend geheel te komen. In de afgelopen tijd is JSF door aantal kleine updates (JSR 252) verder uitgekristalliseerd. De tijd is dan ook rijp om JSF kritisch te bekijken en te analyseren in hoeverre het aan te raden is deze standaard te gaan gebruiken.

Anatomie Voordat deze analyse gemaakt kan worden zal eerst kort ingegaan worden op hetgeen JSF concreet inhoudt. Hierbij zullen we ons beperken tot hetgeen JSF tracht te bereiken voor de ontwikkelaar. Hoe JSF dit realiseert wordt in het midden gelaten. Voor de geïnteresseerden wordt verwezen naar de specificatie en verschillende artikelen die dieper hierop ingaan.

Zoals de specificatie van JSF zelf beschrijft heeft dit framework de doelstelling om het werk van de ontwikkelaar van webapplicaties te vereenvoudigen. Daarnaast wil JSF ook het werken met componenten introduceren in de ontwikkeling van Java-webapplicaties. Tenslotte bestaat de doelstelling om in webapplicaties meer event-gebaseerd te gaan werken, zoals ontwikkelaars ook gewend zijn bij het maken van cliëntapplicaties. Het verbergen van de specifieke en vooral standaardproblemen, waarmee een ontwikkelaar te maken krijgt bij webapplicaties, is natuurlijk ook een taak van het framework.

Aan deze manier van werken zitten een aantal voordelen. Het belangrijkste gegeven is het componentgebaseerd werken, dat hergebruik mogelijk maakt. Ook zorgt het ervoor dat componenten gebruikt kunnen worden, die door externe bedrijven aangeleverd worden. JSF definieert zelf een aantal standaardcomponenten. Deze set is dan bijvoorbeeld uit te breiden met een component om grafieken te tonen. Hierdoor is sneller software van hogere kwaliteit te ontwikkelen.

> JSF beoogt het werk van de ontwikkelaar van webapplicaties te vereenvoudigen

Tools! Tools! Tools! Java-ontwikkelaars geven het niet graag toe, maar Steve Balmer had gelijk met zijn oerkreet “Developers! Developers! Developers!”. Ontwikkelaars willen graag efficiënt kunnen werken en de mogelijkheid hebben om standaardproblemen met de klik van een muis op te lossen. Het maken van een systeem bestaat voor een groot deel uit standaardprogrammatuur, waarvan de creatie goed geautomatiseerd kan worden.

Hierom is JSF duidelijk ontworpen met support voor ontwikkelomgevingen. Deze ondersteuning en de resulterende tools die ontwikkeld zijn voor JSF zijn ook direct een van de sterke punten. De productiviteit van

Advertentie

de developers stijgt hierdoor sterk. Daarnaast is het hierdoor mogelijk, dat de maker van de user interface een echte designer is, die nauwelijks verstand hoeft te hebben van het schrijven van Java-code. De meeste ontwikkelaars hebben niet heel veel verstand van het maken van een mooie interface, waardoor dit duidelijk een voordeel is.

De ondersteuning voor JSF in de verschillende tools is goed te noemen. Grote namen binnen de Java-wereld – IBM, Sun en Oracle – hebben ondersteuning voor JSF ingebouwd in hun producten. Dat JSF onderdeel gaat vormen van de nieuwe J2EE-standaard draagt hier zeker aan bij. Naast de commerciële oplossingen bestaat er ook een open source implementatie[3] van JSF vanuit het Apache project. Qua aanbieders van implementaties is er dus voldoende concurrentie, wat de kwaliteit ten goede komt.

Ook het toevoegen van eigen features door de verschillende leveranciers van een JSF-implementatie lijkt voorspoedig te gaan, waardoor een legio aan mogelijkheden ontstaat voor een applicatieontwikkelaar. Hij kan eenvoudigweg kiezen voor de implementatie die features biedt, welke het beste aansluiten bij de eisen van de te maken applicatie. Vanuit de ontwikkelaars van de JSF-implementaties is het toevoegen van een uitgebreide set aan complexe componenten natuurlijk een van de zaken waarop zij kunnen concurreren.

JSF vergeleken met Struts Struts is sinds enige tijd al een van de *de facto*-standaarden voor de ontwikkeling van webapplicaties. Daarmee is Struts een van de uitgelezen kandidaten voor een vergelijking. De ontwikkelaar van Struts Craig McClanahan was spec lead voor JSF. De verwachting is dus dat JSF beter zal presteren.

Een van de nadelen van Struts is dat het zich altijd heel erg heeft gericht op de implementatie van de controller van het Model-View-Controller pattern en minder op de visuele kant. Het maken van attractieve webinterfaces was hierdoor bij Struts altijd het ondergeschoven kindje. Bij JSF is de focus van het begin af aan gericht geweest op het mogelijk maken van de ontwikkeling van de interface door een designer, met behoud van de uitgebreide mogelijkheden van Struts. Zoals beschreven, ondersteund JSF bijvoorbeeld de ontwikkeling van tools waarmee de designer de webpagina's kan ontwerpen.

Daarnaast is een ander groot nadeel van Struts aangepast: de business logica raakte verweven met het gebruikte framework. Struts stelt eisen aan de wijze van implementatie, hetgeen geen wensbare situatie is. Het wisselen van een framework is hierdoor vaak lastig of zelfs niet goed(koop) uit te voeren. Bij JSF is deze afhankelijkheid in duidelijk mindere mate tot niet aanwezig. Best practices, zoals hergebruik van de code, zijn hierdoor beter mogelijk.

Waar Struts duidelijk op dit moment het voordeel heeft, is de volwassenheid van het framework. Struts is veelvuldig gebruikt bij het ontwikkelen van webapplicaties en daardoor uitontwikkeld. JSF-implementaties zijn op dit moment ongeveer een jaar oud. Begin volgend jaar zal de nieuwe versie van Java Enterprise Edition gelanceerd worden, waarvan JSF een standaardonderdeel vormt. De verwachting is dan ook dat dit voordeel voor Struts snel zal verdwijnen.

Na het publiceren van de JSF specificatie in 2004 is in de Struts-wereld dan ook de discussie ontstaan waar Struts zich in de toekomst op gaat richten. Besloten is om Struts te splitsen in twee frameworks. Het ene framework – Struts Core – zal gebaseerd blijven op de bestaande Struts technologie en deze verder ontwikkelen. Struts Shale zal daarentegen gebaseerd worden op de JSF-specificatie en hiervoor technologie ontwikkelen waarmee de ontwikkeling van applicaties met JSF nog beter uit te voeren is.

Daarnaast wordt veel aandacht besteedt aan het combineren van Struts en JSF, waardoor een eenvoudig migratiepad voor bestaande applicaties geboden wordt. Hierbij is het mogelijk om nieuwe features toe te voegen met behulp van de voordelen die JSF biedt, terwijl de bestaande zaken niet direct gemigreerd hoeven te worden. Dit kan vervolgens in de toekomst alsnog gebeuren, op het moment dat dit bijvoorbeeld het beste past binnen het project.

> Grote namen binnen de Java-wereld hebben ondersteuning voor JSF ingebouwd in hun producten

JSF en AJAX Binnen de wereld van webapplicaties is AJAX een van de hot items van dit moment. Met behulp van deze techniek – die in feite een combinatie van een aantal bestaande technieken is zoals JavaScript, XML en DOM – is het mogelijk om webapplicaties een meer dynamisch karakter te geven. Normaal kan wordt de inhoud van een webapplicatie alleen ge-update met nieuwe gegevens vanuit de server bij het klikken op een link bijvoorbeeld. Het nadeel hiervan is dat de pagina dan ververst moet worden en de gebruiker dus niet kan doorwerken. Met AJAX is het mogelijk om het opvragen van de gegevens asynchroon – terwijl de gebruiker doorwerkt – te doen en deze te tonen op het moment dat deze binnen gekomen zijn.

Geïnspireerd door de mogelijkheden die Google biedt met zijn webapplicaties zijn veel ontwikkelaars bezig om zelf ook AJAX te gaan toepassen. Hoewel AJAX zeker niet de heilige graal is die alle problemen met

webapplicaties oplost, bewijst het zich steeds meer als een techniek die voor bepaalde problemen een goede oplossing kan vormen. Hierbij kan vooral gedacht worden aan applicaties die een interactie vereisen die qua mogelijkheden dicht in de buurt komt van thick-client applicaties op de computer van de gebruiker. Een concreet voorbeeld hiervan is uitgebreide validatie van gegevens of het tonen van selectielijsten met gegevens, bijvoorbeeld mogelijke postcodes nadat een stad is ingevuld. Hierbij is het wel zo dat het ontstaan van frameworks de productiviteit waarmee dergelijke applicaties gebouwd worden nog sterk kan verhogen aangezien de ontwikkeling hiervan wel enige tijd kost.

> Buiten Nederland wordt JSF al in een aantal implementaties succesvol toegepast

Implementatie De vraag is natuurlijk of en hoe AJAX gebruikt kan worden met JSF. Vanuit de werkwijze van JSF zijn er duidelijk voordelen om de AJAX-implementatie niet zelf toe te voegen, maar deze te integreren met de verschillende componenten. Op verschillende websites[6] wordt beschreven hoe dit mogelijk is. Hieruit wordt ook duidelijk dat de integratie van AJAX binnen het lifecycle management van JSF goed in te passen is.

Vanuit het oogpunt van de ontwikkelaar van een JSF-applicatie zijn de voordelen van de integratie van AJAX in een JSF-component duidelijk. Hierdoor hoeft hij zich nauwelijks druk te maken over de wijze waarop de AJAX geïmplementeerd wordt, terwijl hij wel volledig gebruik kan maken van de geavanceerde mogelijkheden hiervan. Vanuit het oogpunt van de JSF-ontwikkelaar blijft de integratie met AJAX beperkt tot het gebruik van een custom component, waarvan bepaalde eigenschappen ingesteld moeten worden en het schrijven van een Java methode die aangeroepen wordt vanuit de AJAX-implementatie. Het toevoegen van een standaardregel aan het JSF-configuratiebestand completeert het geheel.

Natuurlijk is het niet altijd zinnig AJAX in te zetten. De techniek biedt componentontwikkelaars echter wel de mogelijkheid om complexe componenten te implementeren, die de gebruiker van een website het gevoel geeft met een standaardapplicatie bezig te zijn. Door de architectuur van JSF en de bijbehorende ondersteuning voor custom componenten is deze integratie grotendeels transparant uit te voeren voor de JSF-ontwikkelaar.

Gezichtsverlies? JSF wordt een nieuwe standaard in Java EE 5. Het verhoogt de productiviteit en daarmee verlaagt het de bestede uren aan het bouwen van de interfaces, die ook nog eens door designers ontwikkeld kunnen worden. Hierdoor is de verwachting dat verschillende bedrijven zich de komende tijd gaan standaardiseren op JSF. Buiten Nederland wordt JSF al in een aantal implementaties succesvol toegepast. Daarnaast is in ons eigen land een groeiende support te zien.

Als een van de grote spelers op de markt spreekt Atos Origin zich duidelijk uit als voorstander van het gebruik van JSF en positioneert het als dé keuze voor nieuwbouw trajecten. JSF bleek na interne research duidelijke voordelen op te leveren bij het ontwikkelen van webgebaseerde applicaties. Op dit moment zijn binnen Atos Origin ongeveer twintig man opgeleid in het ontwikkelen van applicaties met behulp van JSF.

De komende tijd zal van JSF een nieuwe update van de specificatie uitkomen welke een aantal zaken in de integratie met verschillende andere componenten in het Java EE platform moet versterken en vereenvoudigen. Eén van de laatste conceptversies is in augustus van dit jaar gepubliceerd. Ook voor integratie met geavanceerde technieken zoals AJAX is veel interesse. De toekomst van JSF als het beste ontwikkelplatform van webapplicaties ziet er dan ook rooskleurig uit.

Referenties

1. http://websphere.sys-con.com/read/46516.htm - Vergelijking JSF - Struts
2. http://www-128.ibm.com/developerworks/library/j-jsf4/index.html?ca=drs-tp3005 - Component ontwikkeling in JSF
3. http://myfaces.apache.org - Apache MyFaces project
4. http://struts.apache.org - Struts project
5. http://www.jcp.org - Java Community Process en Java Specification Request
6. https://bpcatalog.dev.java.net/ajax/textfield-jsf/design.html - AJAX en JSF

Jeroen Benckhuijsen (Application Developer) en Tijs Rademakers (Software Architect) zijn beiden werkzaam bij ATOS Origin.