JHeadstart: tussen traditioneel en Java

De kracht van twee ontwikkelstraten samen

Al geruime tijd wordt gediscussieerd over de vraag of maatwerksysteem-ontwikkeling met traditionele Oracle ontwikkeltools voortgezet moet worden of dat het tijd is om voor een Java-platform te kiezen. Een belangrijk aspect van dit vraagstuk is het grote verschil tussen de beide omgevingen qua systeemontwikkelmethodiek. Om toch zinvol een brug te slaan tussen de traditionele ontwikkelmethode en Java kan slim gebruik van JHeadstart als productieversneller veel waarde toevoegen. In dit artikel wordt hierop ingegaan en zal tevens een praktijkcase worden behandeld waar dit vraagstuk centraal stond.

Er zijn inmiddels in de tussentijd technische hulpmiddelen op de markt verschenen om Java-integratie in een traditionele ontwikkelstraat gemakkelijker te maken. Ook bestaan er tools om op de traditionele Oracle ontwikkelmethode systemen te maken voor een J2EE-platform. Vaak zijn deze hulpmiddelen echter functioneel oude wijn in nieuwe zakken. Een voorbeeld van een technisch hulpmiddel om Java te integreren in de traditionele Oracle ontwikkelomgeving is Oracle 9i Webforms. Met de komst van 9i Webforms zijn de integratie mogelijkheden van Java met forms veel verder gefaciliteerd dan voorheen.

Procesgeoriënteerd

Indien de wens is om de traditionele Oracle ontwikkelmethode in een Java omgeving in stand te houden, is JHeadstart beschikbaar. JHeadstart biedt de mogelijkheid om een Java applicatie, op basis van het BC4J-framework of een bestaande repository, volledig gegenereerd en vrijwel zonder programmeren in een J2EE architectuur te zetten. Ook kunnen met JHeadstart bestaande webforms gemigreerd worden naar een J2EE omgeving. Kernredenen om voor een Java-architectuur te kiezen zijn: de gebruikersinterface, het dynamische en flexibele karakter van de programmatuur, mogelijkheden voor webservices, platformonafhankelijkheid en het hergebruik. De voordelen van de traditionele Oracle ontwikkelomgeving zijn een hoge productiviteit in het ontwikkelproces, database georiënteerd programmeren en reeds veel aanwezige kennis van deze ontwikkelmethodiek binnen organisaties. Indien Oracle 9i Webforms gebruikt wordt om Java te integreren in een traditionele omgeving, dan is er nog steeds een beperking in de deployment van de applicatie. Het is relatief zwaarder voor de client en er zal een download plaats moeten vinden van JInitiator. Een ander aandachtspunt is dat de ontwikkelaars in de traditionele stijl blijven ontwikkelen, waardoor het niet zelden voorkomt dat interface requirements niet gehaald kunnen worden. En last but not least: de typerende generiekheid van (Java) Object Oriented ontwikkelen komt zeer magertjes tot zijn recht. Men geniet dus niet van de voordelen van een Java-omgeving. JHeadstart kent dit applicatie deployment-probleem niet. Tevens versnelt het de productiviteit van de Java applicatie-ontwikkeling naar het niveau, dat we in de traditionele Oracle straat gewend zijn. Hier staat weer tegenover dat er geen verbetering optreedt in de generiekheid van de applicatie en de gebruikersinterface. Hoewel de interface anders is, zitten we nog steeds met een systeemgeoriënteerde interface, terwijl een vernieuwende kracht van Java juist de mogelijkheid is om een procesgeoriënteerde interface op te bouwen.
Dit in ogenschouw nemende lijkt JHeadstart meer een tool voor migratie dan voor systeemontwikkeling. Immers, wie zal ervoor kiezen in een nieuwe architectuur te gaan ontwikkelen wanneer de gekozen ontwikkeltool de voordelen van deze architectuur niet benut. Voor uitsluitend het deployment-verschil zal zelden een architectuur omgegooid worden.

Dilemma

Van de eerder genoemde redenen die ten grondslag liggen aan het vraagstuk welke ontwikkelomgeving het best gebruikt kan worden is een belangrijke en gecompliceerde reden: de historie van de organisatie en haar ontwikkelomgeving. Op dit gebied kunnen organisaties min of meer in twee categorieën ingedeeld worden.
Aan de ene kant vind je de organisatie die veel ervaring heeft met de traditionele Oracle ontwikkeltools. Dergelijke organisaties worden gekenmerkt door een grote aanwezigheid van kennis over de traditionele ontwikkeltools en methoden, geen

Het kenmerk van een systeem-georiënteerde applicatie: de gebruiker moet zelf uit het menu kiezen wat hij gaat doen. Als hij meerdere stappen moet doen om zijn proces te voltooien zal hij zelf de bijbehorende schermen moeten kiezen en gebruiken.

Een proces-georiënteerde interface geeft de gebruiker sturing en begeleidt de gebruiker in zijn proces door zelf de benodigde stappen uit het proces in de juiste volgorde aan te bieden. Een voorbeeld hiervan is de welbekende wizard.

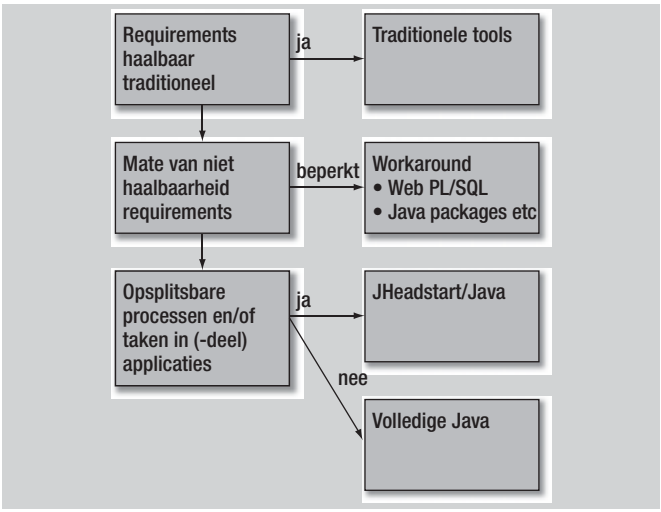
Figuur 1. Het verschil: systeemgeoriënteerd versus procesgeoriënteerd

of beperkte Java-kennis en ervaring, gebruikers die gewend zijn aan de Oracle webforms interface en een ontwikkelteam dat uitblinkt in de traditionele ontwikkelstraat en zich daar het best bij voelt. Hoewel deze organisaties op vele vlakken tevreden zijn met de huidige ontwikkelmethode, voelen zij toch vaak de druk van nieuwe requirements, die de Java-discussie onvermijdelijk doen opwaaien. Applicaties moeten 'gelikter' zijn, klantgerichter, robuuster en zij moeten functies verwezenlijken die oneigenlijk zijn voor de huidige ontwikkelmethode zoals bijvoorbeeld gebruik maken van webservices. Maar de sterke mate van productiviteitsverlies in systeemontwikkeling wordt niet als acceptabel en als moeilijk organisatorisch verkoopbaar ervaren, mede vanwege de uitstekende ervaringen met de traditionele ontwikkelstraat. Ook het verlies van de waarde van de ervaringen en capaciteiten van de huidige personele bezetting bij switchen worden als erg moeilijk ervaren.

Aan de andere kant zijn er de organisaties waar men al langere tijd werkt met de webarchitectuur. Productiviteitsverlies zien zij niet, want organisatorisch wordt een langere doorlooptijd als vanzelfsprekend gezien. Dit komt ondermeer doordat zij veel minder ervaring hebben met krachtige productiviteitsversnellers zoals generatietools en frameworks die men in de traditionele Oracle ontwikkelomgeving tegenkomt. Ook wordt er minder gebruik gemaakt van de toegevoegde waarde die een Oracle Database te bieden heeft, daar zij niet de ervaring hebben om met de Oracle tools het maximale uit een database te halen. Men staat er vaak niet bij stil staat dat packages ten behoeve van schermen en applicaties direct tegen de database gegenereerd kunnen worden.

Doorlooptijd

Het hiervoor beschreven dilemma heeft niet alleen betrekking op de organisaties, maar evenzeer op de individuele ontwikkelaar. Een Java-ontwikkelaar heeft onbegrip voor het data-georïenteerde karakter van de Oracle-applicatie en de Oracle ontwikkelaar vraagt zich af wat de rol van de database in een Java applicatie is. Een vaak gehoorde eerste reactie van een Oracle-ontwikkelaar op een Java ontwikkelomgeving luidt: "Is dit niet gewoon een stap terug in de tijd waarbij je alles weer zelf moet doen?". Hij ziet een stap terug in de tijd naar een derde generatie-omgeving zonder de kracht ervan in te zien. Daarom vraagt hij zich af waarom hij de productiviteit op zou offeren om een applicatie in Java te bouwen. Een samenvatting van de misverstanden tussen de twee verschillende ontwikkelaars komt erop neer dat de Oracle ontwikkelaars goed zijn in het afstemmen van het databasemodel op de applicatieontwikkeling. Anderzijds blinken de Java-ontwikkelaars uit in de ontwikkeling van procesgestuurde applicaties, dynamiek bouwen in applicaties en applicatieontwikkeling databaseonafhankelijk te zien.



Figuur 2. Beslissingsmodel ontwikkelmethodiek ten bate van productiviteitsbehoud

De oplossing

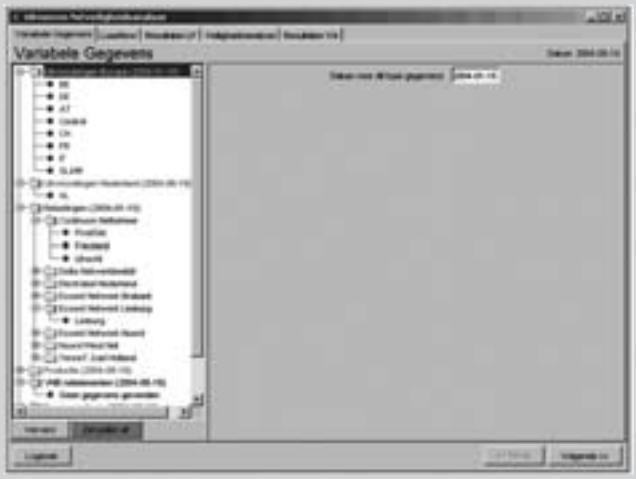
Deze twee verschillende werelden kunnen versmolten worden. JHeadstart biedt daarvoor de mogelijkheden. Momenteel zie je interesse in JHeadstart met name komen van de organisaties die historisch gezien met de traditionele Oracle tools ontwikkelen. Deze vraag is vaak gericht op het migreren van forms naar een webomgeving of om te bekijken of Java zich leent voor het relatief eenvoudig opbouwen van forms. Vanuit het andere beschreven type organisatie, de traditionele Java webontwikkelaars, kan ook de vraag komen naar JHeadstart en eigenlijk lijkt het nu juist hier een uiterst krachtige tool te kunnen zijn.
Stel namelijk dat een Java-georïenteerde ontwikkelomgeving requirements krijgt die lijken op de businessvragen waar de traditionele Oracle ontwikkelaars door en door mee bekend zijn. Bijvoorbeeld een databasegeoriënteerde functie te realiseren. Dan kan JHeadstart juist ook hier uitkomst brengen.
Anderzijds kan het de organisaties die hun ontwikkelstraat aan het omzetten zijn in meer J2EE-gerichte architectuur helpen om bij de functies die requirements hebben die in de traditionele omgeving gemakkelijk opgelost werden nog steeds snel en gemakkelijk op te lossen met JHeadstart. Dit betekent dat de transitie naar Java-gerichte systeemontwikkeling veel gemakkelijker en acceptabeler wordt. Men blijft profiteren van de kennis van database-georïenteerd programmeren en van de snelle en volwassen generatietools. Wanneer JHeadstart in organisaties of in teams gebruikt wordt die gewend zijn aan de (Java-)webarchitectuur en onbekend met de traditionele Oracle-ontwikkelmethode, dan biedt het zelfs onverwachte versnelling van ontwikkeling en nieuw inzicht in gebruik van de database voor applicatieontwikkeling.

Beslissingsmodel

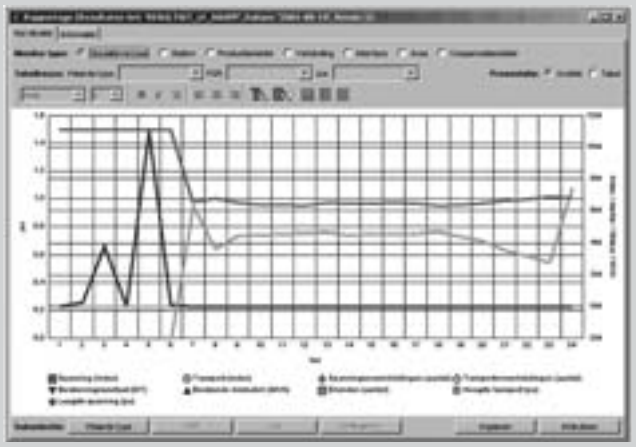
Hoe wordt met JHeadstart de optimale mix behaald van de versmelting van een Java / Oracle ontwikkelomgeving? Het gaat erom de applicatieontwikkeling te doen met de traditionele ontwikkelmethode daar waar de applicatie zich ervoor leent, en daar waar de requirements niet passen in de traditionele methode met Java alle voordelen van de Java-omgeving te realiseren. Bij de analyse van de requirements wordt afgewogen welke combinaties van taken en processen zich lenen voor de traditionele ontwikkelmethode en welke schermen onvermijdelijk niet met de traditionele methode opgebouwd kunnen worden. Criteria hiervoor zijn uiteraard organisatie en projectafhankelijkheid, maar als algemene criteria kunnen procesorientatie van de applicatie, opleidingskosten en -mogelijkheden, gebruikersinterface en (on)afhankelijkheid van functies. Op basis hiervan vindt een opsplitsing van de applicatie plaats in een gedeelte dat ontwikkelt wordt met JHeadstart en een deel wat met Java gemaakt wordt. Het database-ontwerp en realisatie van het JHeadstart-applicatiegedeelte kan direct richting de ontwikkelaars met de Oracle ervaring doorgeschoven worden.

Hier wordt dan direct geprofiteerd van de traditionele Oracle ontwikkelkennis in het database-ontwerp voor de JHeadstart

3a. Gebruikersprocessen



3b. Opgedeeld in stapsgewijze taken



3c. T.b.v. complexe grafieken

Figuur 3: De applicatie heeft een proces- en taakgestuurde gebruikersinterface en moet complexe grafische modules ondersteunen (Javaclient).

Figuur 4. Beheersapplicatie ligt min of meer '1-op-1' op definition (_def) tabellen

applicatie, dit applicatie gedeelte kan zeer snel ontwikkeld worden. Tevens worden de uitgebreidere requirements gehaald daar waar dat nodig is door 'handwerk'. Het resultaat: twee applicaties die elkaar aanvullen en een veel snellere doorlooptijd van systeemontwikkeling met een benutting van de traditioneel opgebouwde kennis. Een voorbeeld van zo'n besluitproces is weergegeven in Figuur 1.

Toepassing

In het voorjaar van 2003 werd bij TenneT B.V., de netbeheerder van het Nederlandse elektriciteitsnet, de knoop doorgehakt. Er wordt besloten een applicatie te ontwikkelen met Java. Dit betekent organisatorisch een grote stap, omdat TenneT historisch gewend is aan gebruik en ontwikkeling van applicaties die in de Oracle Designer / Developer straat ontwikkeld worden. Dit besluit is gefundeerd op de volgende uitgangspunten: men wil innovatief zijn, meer mogelijkheden met applicaties creëren en ervaring in Java opdoen. Die ervaring werd voor de IT-afdeling als een must gezien, aangezien het moeilijk bleef om vanuit de literatuur, adviezen en het afwegen van voors en tegens een goede, onderbouwde keuze voor ontwikkelmethode te maken. Aan dit project werkte TenneT samen met Capgemini en Ordina.
Lucien Okkes, projectleider en informatieanalist bij TenneT, werd bij dit project betrokken. Hij geeft aan waarom TenneT al snel geïnteresseerd was in JHeadstart: "Toen de keuze voor Java eenmaal gemaakt was, kwam JHeadstart automatisch in beeld. Het concept van genereren tegen een database en minimale programmeerinspanningen werd immers al jaren succesvol toegepast.". Toch realiseerde hij zich al snel dat de webinterface van de applicatie met uitsluitend JHeadstart ontwikkelen oneigenlijk gebruik van deze ontwikkeltool zou betekenen.

Gouden greep

Bij het afstemmen van de tools op de requirements werd besloten de gebruikersprocessen en de beheerstaken op te splitsen. De gebruikersprocessen worden met een Java client opgebouwd in JDeveloper en de beheerstaken worden met JHeadstart gegenereerd en in een aparte gebruikersapplicatie ondergebracht. "Een gouden greep" aldus Lucien Okkes. "Van de 75 gedefinieerde use cases hebben we er 23 toegeschreven aan de beheerstaken. Deze beheerstaken zijn vervolgens gerealiseerd met behulp van JHeadstart waardoor de productiviteit met een ruime factor 2 omhoog ging. Dit betekende een daling van een gemiddelde 60 uur per use case tot minder dan 30 uur per use case in geval van gebruik van JHeadstart. Door deze productiviteitswinst hebben we meer use cases gerealiseerd dan gepland, meer aandacht kunnen besteden aan use cases die niet met JHeadstart ontwikkeld werden en een kwalitatief betere applicatie ontwikkeld." Ook de gebruikers zijn enthousiast. De Java client voldoet aan hun eisen en het beheer is een krachtige tool geworden om de Java client te configureren.

Logica

Uiteraard heeft het gebruik van deze nieuwe technologie ook de nodige aandachtspunten opgeleverd. Zo blijkt uit de projectervaringen dat hergebruik van de BC4J-componenten alleen mogelijk is indien de JHeadstart files in hetzelfde Java project staan als de BC4J-componenten. En zelfs dan is de applicatiemodule nog niet herbruikbaar, deze moet twee keer aangemaakt worden. In dit project is daarom gekozen voor twee aparte BC4J-lagen.
Verder dient het ontwikkelteam bij de interface ontwikkeling met JHeadstart het verschil in architectuur met de Oracle Forms-omgeving te realiseren. Daar JHeadstart is gebaseerd op het MVC-model, zit de business logica in de modellaag. De logica is daardoor op één plaats verzameld. Tevens wordt deze meegenomen in de gebruikersinterface bij generatie. Het kan voor een Forms-ontwikkelaar echter wennen zijn, dat handmatig coderen aan de clientzijde nauwelijks ondersteund wordt.

Conclusie

JHeadstart, vaak in combinatie met het BC4J-framework, biedt een interessante tussenweg om de kracht van zowel de Oracle Designer / Developer ontwikkelstraat als de Java ontwikkelstraat te combineren. Er hoeft niet volledig voor één ontwikkelmethode gekozen te worden. De voordelen van de ene methode gaan niet meer volledig verloren wanneer gekozen wordt voor nadruk op de andere methode. Opgedane organisatorische ontwikkelkennis blijft deels van waarde en herbruikbaar. Dit resultaat is maximaal bij opsplitsing van use cases en applicatieontwikkeling.

drs. Marcel van Oers is consultant bij Capgemini.