E-BusinessSuite 12.1 SOA Gateway

Een paradox opgelost?

Integratie en E-Business Suite lijkt een paradox: Dat was het oorspronkelijk in elk geval. De hele gedachte achter E-BuisinessSuite was dat als je alles wat je nodig hebt in een pakket stopt, je geen "mannen met lijmpistolen" meer nodig hebt. Integratie vanuit de doos. Maar Oracle had gelukkig vlot door dat klanten vaak heel goede redenen hebben om voor andere producten te kiezen. Bijvoorbeeld Siebel voor CRM. Bekend is dat de strategie van Oracle is veranderd. En daarmee is de noodzaak om de verschillende pakketten van Oracle (waaronder inmiddels datzelfde Siebel) te integreren vergroot.

Een interface maken op E-BuisnessSuite is geen "raket-technologie". Het datamodel is vrij transparant, en EBS ondersteunt verschillende interfacetechnologieën, gebaseerd op SQL en Pl/Sql: API's, Open Interfaces, XML Gateway.

Voor het opnemen van E-BusinessSuite in een echte integratie oplossing moest naar externe tools worden uitgeweken. Het feit dat tot en met 11i EBS geen echte J2EE applicatie server in zich had hielp ook niet. Voor een zelfbouw Webservice had je een aparte applicatie server nodig, met alle beheer van dien. En wilde je het meer declaratief aanpakken: een BPEL Process Manager of een Service Bus.

De Apps Adapter

Al heel lang heeft Oracle een aparte adapter voor E-BusinessSuite in de SOA Suite (en daarvoor ook voor BPEL ProcessManager en InterConnect). Die Apps-adapter maakt het mogelijk om te bladeren in de Integration Repository (iRep) van EBS. Daar kan dan een te benaderen interface worden gekozen waarop dan de adapter-definities worden gegenereerd. Staat een interface niet in de iRep, bijvoorbeeld een custom-package, dan is daar ook naar te bladeren. Afgezien van de introspectie-mogelijkheden op de iRep is de Apps adapter (in mijn ogen) niet veel meer dan een sausje op de Database-Adapter (voor SQL en Pl/Sql interfaces) en de AQ-Adapter (voor business events). In dat sausje zit met name de zorg voor het zetten van de Apps-Responsibility bij het aanroepen van de interface. Dat wil zeggen: voor welke authorisatie de interface wordt aangeroepen.
Hoewel de Apps-adapter-wizard gewoon beschikbaar is in JDeveloper is de prijs van Apps-adapter dusdanig dat ik altijd koos voor de Database- en AQ-Adapters Want deze zijn gratis en meestal maakten we toch zelf complexere wrappers om de EBS-API's. Ook had ik altijd wat moeite manier waarop de EBS-Adapter de responsibility regelde (nl. in de WSDL van de service), zodat ik dat net zo lief zelf in de wrapper oploste.

De Apps-adapter, of de andere 2 adapters, maakten het in combinatie met SoaSuite mogelijk om op een declaratieve manier EBS aan te spreken in een SOA-omgeving. Maar het blijft buiten EBS. Foutafhandeling moet je dus ook buiten EBS, in je SOA omgeving, regelen. En wat als dat nu eens een niet-Oracle-Stack is?

De SOA Gateway

In EBS 12 is er een heel belangrijke architectuur wijziging doorgevoerd: de introductie van OC4J als applicatie server. OC4J (Oracle Containers for Java of J2EE) was de J2EE applicatie server van Oracle voordat Weblogic werd omarmd. De applicatie server waar SoaSuite10.1.3 op was gebaseerd. En dit op het oog simpele gegeven maakt iets anders ineens mogelijk: wat nu als de SoaSuite Apps-Adapter nu eens geïmplementeerd wordt in EBS? De JDeveloper-wizards zijn dan te vervangen door HTML schermen op de iRep in EBS zelf. En dan kun je met een paar simpele muisklikken een webservice

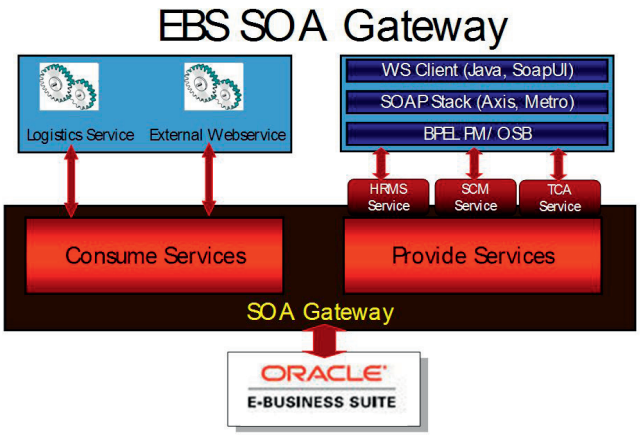
Oracle had vlot door dat klanten vaak heel goede redenen hebben om voor andere producten te kiezen.

Afbeelding 1: SOA Gateway Architectuur

genereren en implementeren op elke willekeurig interface die in de iRep zit. En werkelijk: het is niet veel ingewikkelder dan dat. Dit onderdeel wordt de SOA Provider genoemd.

Naast de SOA Provider is het in de SOA Gateway ook mogelijk gemaakt om op een declaratieve manier externe services toegankelijk te maken in EBS. Het mechanisme daarvan is gebaseerd op het Business Event System van Oracle Workflow. Naast op Pl/Sql of Java "Rulefunctions", Oracle Workflow Processen gebaseerde Event Subscriptions (Gebeurtenis Abonnementen) zijn nu ook Webservice gebaseerde abonnementen te definiëren. Voor de responses op webservice aanroepen kunnen ook business events worden gedefinieerd. Deze events worden dan afgevuurd met de payload van het response bericht. Op die events kunnen dan ook weer abonnementen worden gedefinieerd om de response te verwerken. Denk aan bijvoorbeeld een rule function om de data in de database bij te werken. Of een event-activiteit in een workflowprocess om het process na terugkomst van het event weer verder te laten gaan. De architectuur van de SOA Gateway ziet er dan als volgt uit:

De SOA Gateway is beschikbaar vanaf EBS 12.1. Let daarbij op dat er nog wel wat extra patching en configuratie nodig is.



Afbeelding 2: Integration Repository

SOA Provider

De SOA Provider is het in het oogspringende onderdeel van de SOA Gateway. Het leunt zwaar op de Integration Repository. Dat is de centrale repository van EBS waarin alle publieke services in een hiërarchische structuur zijn geregistreerd. De iRep is ook aan te vullen met "Custom" en "Compound" services. Vanuit de iRep zijn van "Native Interfaces" en "Custom Interfaces" webservices te genereren. Dat is te zien in bijgaande screendump.

Het begint met het genereren van de WSDL. Als die eenmaal is gegenereerd dan biedt de pagina de mogelijkheid om authorisatie en authenticatie op Webservice niveau en op EBS-niveau te configureren. Vervolgens is met andere knoppen de service te (her-) implementeren (deploy/redeploy) naar de interne OC4J server. Daarna is de service meteen extern beschikbaar.

Dit is vrij eenvoudig te testen met een tool als SoapUI. Wel is het nog wel puzzelen om het request bericht met de juiste data te vullen zodat de aanroep voor EBS juist uitvoerbaar is. Ook blijkt dat wanneer een aanroep tot meerdere fouten leidt, de lijst van fouten niet altijd correct wordt terug gestuurd. Dan is het handig om de aanroep eerst na te spelen in Pl/Sql.

Overigens kunnen ook zogenaamde "Compound Services" worden geregistreerd in de iRep. Dit zijn eigenlijk uitgewerkte BPEL processen die een of meerdere andere EBS native interfaces aanroepen. EBS bevat echter geen interne BPEL server en is dus niet in staat deze processen zelf uit te voeren. Geregistreerde "compound services" kunnen daarom alleen worden gedownload. Daarna zullen ze moeten worden aangepast om de end-points goed te zetten. Dit kan eventueel geautomatiseerd worden met behulp van Ant. Daarna zullen ze, al dan niet geautomatiseerd, op een externe BPEL server geïmplementeerd worden.

SOA Monitor

Zowel tijdens de generatie/implementatie van de services als tijdens de uitvoer van de services kan er van alles mis gaan. Prettig is dan dat een en ander wordt gelogd om de problemen te kunnen analyseren. Wat er gelogd wordt en voor wie dat beschikbaar is kan worden ingesteld.

Van Service aanroepen worden naast de stappen in de uitvoer (log) ook de request en response documenten opgeslagen.

De opslag van de request en response documenten is erg prettig voor ontwikkel en test omgevingen. Maar mogelijk noodzakelijk voor regelgeving. Het vind echter alleen plaats

wanneer "auditing" aan staat. Dit is ook uit te zetten en te "purgen". Zeker iets om rekening mee te houden bij grote hoeveelheden aanroepen. Immers die opslag van documenten en logging kost wel database ruimte.

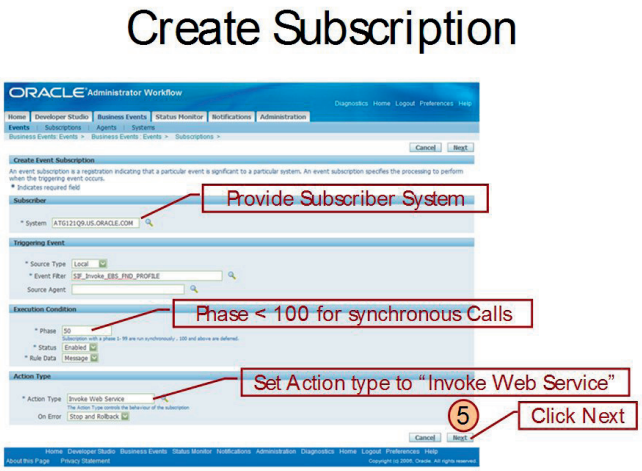
Security

Hier ga ik niet al te diep op in. Naast de gelaagde authenticatie en authorisatie gebaseerd op de frameworks in EBS is er beperkte ondersteuning voor WS-Security. Voor uitgebreidere ondersteuning van policy gebaseerde beveiliging en Encryptie en Signing moet worden uitgeweken naar tools als Oracle Webservice Manager. De security in SOA Gateway is met name bedoeld om te voorkomen dat intern elke aanroep zonder meer wordt toegestaan. Het is niet verstandig om de gegenereerde services direct op internet toegankelijk te maken.

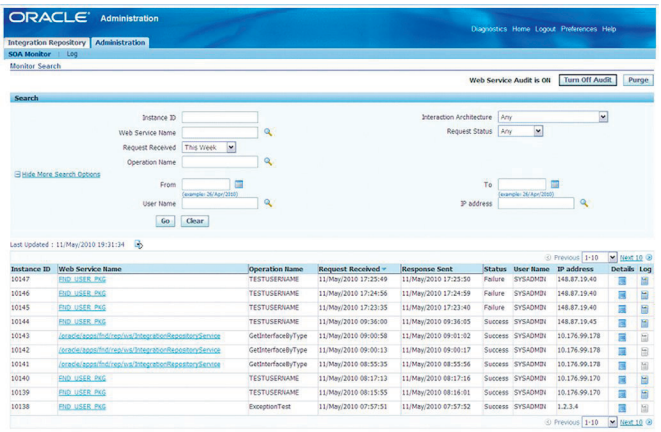
Service Invocation Framework

De SOA Provider is in zich zelf al een hele mooie toevoeging. Maar tijdens een EBS Implementatie traject vindt ook het nodige maatwerk plaats. Het zal derhalve ook regelmatig voor komen dat er op basis van een gebeurtenis in EBS externe services moeten worden aangeroepen.
Nu is het al een tijdje mogelijk om vanuit de database webservices aan te roepen. Dat kan met behulp van de packages UTL_HTTP of UTL_DBWS. In Oracle 8i laadden we wel eens een Java SOAP Stack en maakten daar een java+pl/sql wrapper op. Ook kun je met JPublisher aan de slag. Vervelend is daarbij dat het toch allemaal helemaal moet uitgeprogrammeerd moet worden. Inclusief het opbouwen van de soap-envelope en het afhandelen van fouten. En dat is allemaal geen sinecure.

In Oracle Workflow 2.6 (Standalone) is het Business Event System (BES) geïntroduceerd. Het BES is werkelijk een



Afbeelding 3: SOA Monitor



Afbeelding 4: Definitie van een Event Subscription

prachtig mechanisme. Oorspronkelijk gebaseerd op Pl/Sql en AQ, later in 2.6.4 (Oracle 10g DB) is er een op OC4J gebaseerde event listener/handler gebouwd. Via HTML pagina's zijn gebeurtenissen (events) te definieren. Elke gebeurtenis kan meerdere abonnementen hebben. Een gebeurtenis kan worden afgevuurd met een API of vanuit een Oracle Workflow activiteit. In Pl/Sql is dat heel simpel wf_event.raise. Bij het afvuren geef je de naam van de gebeurtenis, een gebeurtenis sleutel en een payload mee.
De Event Handler handelt vervolgens de geregistreerde abonnementen af. Dat kan op de achtergrond (deferred) of op de voorgrond (synchroon) plaats vinden. En in EBS 12 kan dat in de database tier (Pl/Sql) of in de Java tier (OC4J).
Een abonnement kan de volgende technologiën aan:

- Oracle Workflow Process
- Pl/Sql Rulefunction
- Java Rulefunction
- En nu ook: "Invoke Webservice"

Het EBS is uiterst flexibel en volledig declaratief. Het maakt het mogelijk om aan te roepen functionaliteit volledig technisch te scheiden van de aanroep. Dat betekent dat de aanroepen de sessie geen last heeft wanneer de aangeroepen functionaliteit:

- fout loopt
- traag is
- een lange doorlooptijd heeft
- onbeschikbaar is

In al die gevallen vangt het EBS dat op en zorgt eventueel voor het aanroepen van een error workflow met een notificatie naar de Sysadmin. De sessie is na het afvuren van de gebeurtenis meteen vrij.
Daarom is het BES ook de voorkeurstechnologie voor het aftrappen van maatwerk code in BES. EBS heeft al voor de meeste entiteiten events. Mocht voor een bepaalde entiteit nog geen voorgedefinieerd event zijn maak er dan een aan en roep in de database trigger alleen wf_event.raise aan, in plaats

van de code direct aan te roepen van uit de trigger (of nog erger: de complete code in de trigger programmeren).

Wanneer je nu kiest voor "Invoke Webservice", voert de wizard je in een aantal stappen door het process van:

- selecteren en laden van de WSDL, waarna de WSDL "geparsed" wordt
- selecteren van de aan te roepen Service uit de WSDL
- selecteren van de betreffende Service Port
- selecteren van de Operation
- en het opgeven van de benodigde parameters als
- Agent en of event voor het verwerken van de response
- WSSE security gegevens

De event pagina geeft een mogelijkheid om het event te testen. De payload van de service aanroep kan dan worden meegegeven waarna het event in de Pl/Sql of Java tier kan worden afgevuurd. Stoerder is natuurlijk wanneer het event vanuit Pl/Sql, Java of Oracle Workflow wordt afgevuurd.

Conclusie

De SOA Gateway maakt wat mij betreft de Apps Adapter in eens weer interessant. Het is heel sterk hoe met een paar klikken een api is te publiceren als Webservice. Zo zien we dat graag in een applicatie suite. Ook het gebruik van het BES als manier om op een declaratieve manier externe services beschikbaar te krijgen in EBS is sterk. Merk overigens wel op dat de Native Interfaces veelal gebaseerd zijn op entiteiten in EBS. Deze zijn over het algemeen te granulair om als Business Objects te zien. Voor het opvoeren van een klant of een order zijn meerdere aanroepen nodig. Dus voor het implementeren van een Business Service heb je toch een orchestratie tool als BPEL PM/SOA Suite nodig.

Dit artikel is een spin-off van een workshop die ik heb gegeven over de SOA Gateway. Het enthousiastme dat ontstond door het stoeien met de SOA Gateway tijdens de voorbereiding en het geven van de workshop is misschien te lezen in het artikel. Met plezier geef ik de workshop nogmaals voor geïnteresseerde bedrijven.

Martien van den Akker is Technisch Architect Integratie bij Darwin-IT Professionals en blogt ook op darwin-it.blogspot.com