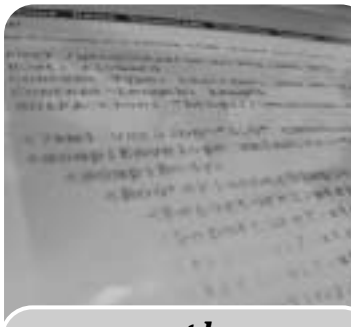
Business Process Execution Language for Web Services (BPEL4WS of BPEL) is op dit moment de leidende technologie om web services volledig te integreren in bestaande business processen. Bas van de Sande schetst in dit artikel een historisch overzicht over de totstandkoming van BPEL. Voorts beschrijft hij de technische en functionele vereisten, en wanneer en waarom BPEL ingezet kan worden.Tenslotte geeft hij een introductie in de taal BPEL.



BPEL for Web Services

Een nieuwe manier van applicatie integratie

Voordat BPEL tot stand kwam gebruikten de diverse leveranciers met verschillende standaarden om "executable business processes" te definiëren. Microsoft zette hiervoor XLANG in, IBM gebruikte hiervoor WSFL en BPMI.org combineerde deze twee methoden in BMPL. BPEL is door IBM, BEA en Microsoft gelanceerd in augustus 2002. In April 2003 werd BPEL aangeboden aan OASIS (Organisation for Advancement of Structured Information) om op deze manier een nog bredere acceptatie te krijgen als een open standaard. OASIS is een non-profit organisatie die een standaardisatie nastreeft op het gebied van e-business standaarden.Vandaag de dag wordt BPEL door veel analisten gezien als de standaard.

SAMENSMELTING Naast BPEL bestaat er nog een andere standaard op de markt, te weten WSCI (Web Services Choreography Interface) die door SUN en Oracle wordt gepromoot. Het grootste verschil tussen WSCI en BPEL is dat WSCI zich uitsluitend focust op choreografie - dit zijn de gedefinieerde interacties tussen onafhankelijke processen (in BPEL terminologie "abstract processes"). Terwijl BPEL ook nog eens "executable processes" ondersteunt (die dus afwezig zijn in WSCI).

BPEL wordt op dit moment ondersteund door Microsoft, IBM, SAP, Siebel Systems, BEA, OASIS. Recentelijk hebben SUN en Oracle aangekondigd om zich ook aan te sluiten bij het OASIS comité. Als gevolg hiervan zullen er waarschijnlijk twee standaarden ontstaan BPEL (OASIS) en WSCI (W3C).

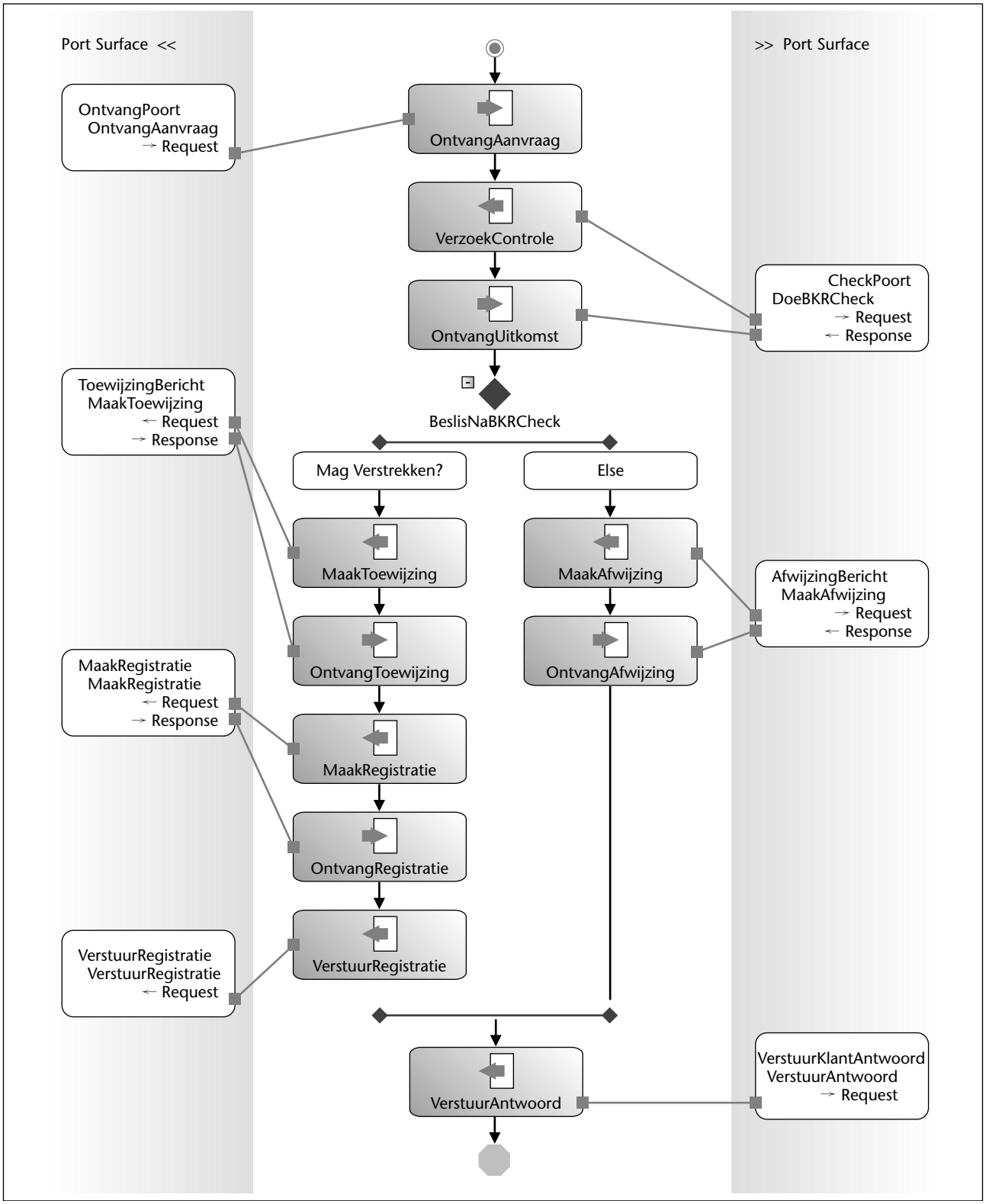
In feite representeert BPEL een samensmelting van taal eigenschappen van IBM's Web Service Flow Language (WSFL) en Microsoft's XLANG (zoals gebruikt in Biztalk Server 2000 en 2002). Zowel WSFL als XLANG worden aan de kant gezet voor de BPEL specificatie, zo zal Biztalk Server 2004 gebruik maken van BPEL. Er is inmiddels al een werkende BPEL implementatie gereed in de eerste bèta van BizTalk 2004, volledig geïntegreerd met Visual Studio.NET.

BPEL ORCHESTRATION In de hedendaagse IT omgevingen worden processen veelal aan elkaar geknoopt middels maatwerk oplossingen of middels functioneel georiënteerde applicaties (denk hierbij aan ERP, CRM, supply chain applicaties). Deze manier van proces integratie is vaak tijdrovend en bovendien kostbaar, omdat:

- productspecialisten meestal schaars zijn,
- systemen vaak op andere platformen draaien,
- er geen eenduidige standaarden zijn vastgelegd over de manier hoe systemen met elkaar samenwerken.

Door gebruik te maken van open web standaarden verwacht men juist deze kosten te kunnen drukken en bovendien de ontwikkeltijden sterk terug te dringen en kosten-besparingen te realiseren in de onderhoudssfeer. BPEL kan er voor zorgen dat systemen op een meer flexibele manier met elkaar kunnen samenwerken. Juist door het gebruik maken van een vooraf gedefinieerde manier van samenwerken is het voor organisaties gemakkelijk om systemen te upgraden of te vervangen zonder dat het totale proces beïnvloed wordt.

Business transacties draaien zelden geïsoleerd. Sterker nog deze transacties gaan vaak over systemen heen (zowel interne systemen als systemen bij externe



FIGUUR 1. Voorbeeld van Orchestration in Microsoft Biztalk Server 2004

partijen). Deze transacties zijn dan ook vaak onderdeel van langlopende processen. Processen zijn meestal opgebouwd uit een aantal transacties, acties en/of beslissingen die genomen moeten worden. Dit wordt ook wel een automatische 'workflow' genoemd. Orchestration is dan ook het opzetten en beheren van de automatische 'workflow'. Een workflow kan op zijn beurt weer opgebouwd zijn uit andere workflows en kan ook weer over organisaties heen gaan. Denk hierbij aan communicatie tussen systemen van organisaties.

FOUTAFHANDELING Om aan al deze eisen tegemoet te komen zijn de eerste web services ontstaan. De eerste generatie web services richtte zich met name op het

beschikbaar maken van systemen. Alleen het beschikbaar maken van systemen is niet voldoende, het gaat er juist om hoe systemen met elkaar samenwerken en hoe beslissingen en eventuele fouten worden afgehandeld. Juist dit is hetgene waar "Orchestration" zich op richt. Orchestration kan gezien worden als het coördineren van interacties tussen webservices binnen langlopende processen. Orchestration bestaat uit een drietal pijlers.

1. *Correlation*
   Met name als de communicatie over systemen tussen organisaties of afdelingen gaan is het mogelijk dat er geen realtime communicatie plaatsvindt. Er moet dan ook een mechanisme zijn, waarbij de ontvangende partij een soort van 'reply-to-adress' krijgt om een berichtje terug te sturen indien een proces gereed is.
2. *Flow coördinatie*
   Afhankelijk van terugmeldingen of van bepaalde waarden in berichten kan het systeem andere beslissingen moeten nemen. Denk hierbij aan een kredietwaardigheidcontrole, waarbij afhankelijk van de uitkomst een andere actie moet worden ondernomen.
3. *Exception management*
   Wanneer processen "fout" lopen moeten er soms bepaalde acties worden ondernomen om correcties door te voeren. Dit kan zijn het terugdraaien van een database transactie of het uitvoeren van een annuleringsboeking. Dit soort acties wordt compensating transactions genoemd.

Deze drie pijlers tezamen vormen de basis van end-to-end business processen. Maar om orchestration te implementeren is er ook een infrastructuur nodig. De basis hiervoor is een BPEL server. Deze BPEL server moet een aantal eigenschappen hebben, te weten:

1. *BPEL Server*
   De BPEL server vormt de basis van de infrastructuur. Deze server moet instaat zijn om de proces flows uit te kunnen voeren en moet kunnen communiceren (middels web services) met andere BPEL servers. Verder moeten de business flows over een langere periode asynchroon kunnen draaien.
2. *BPEL console*
   De BPEL console zorgt voor eenvoudig onderhoudbare omgeving waarbinnen de business flows beheerd kunnen worden. In een grafische omgeving worden de workflows getekend, de console zorgt ervoor dat het BPEL script wordt gegenereerd.
3. *Integratie*
   De BPEL server moet zijn voorzien van een mogelijkheid waarbij gemakkelijk andere systemen kunnen worden geïntegreerd. Bijvoorbeeld er zijn voor Microsoft Biztalk een groot aantal connectors beschikbaar om koppelingen te maken met externe systemen (SAP, Siebel, Navision etc.)

BPEL TAAL Zo op het eerste gezicht ziet BPEL eruit als een op XML gebaseerde script taal; een BPEL execution engine kan door het script heenlopen en deze ook uitvoeren. Buiten dit om zal BPEL een beetje vreemd lijken voor ontwikkelaars die meer bekend zijn met omgevingen als C, C#, Java of zelfs Cobol. Dit artikel gaat via een voorbeeld scenario de taal BPEL toelichten. In de toelichting wordt gebruik gemaakt van Microsoft Biztalk Server 2004. Vervolgens zal de gegenereerde BPEL code worden behandeld.

SCENARIO KREDIETAANVRAAG Een klant van een bank wil een krediet aanvragen. In hoofdlijnen gebeurt tijdens het proces van kredietaanvraag het volgende:

1. Eerst vindt er een controle plaats bij het Bureau Krediet Registratie (BKR) of de klant kredietwaardig is.
2. Is de klant niet kredietwaardig dan volgt er een afwijzing. Een bericht wordt naar de klant gestuurd dat hij niet kredietwaardig is. Het proces is hiermee afgelopen.
3. Is de klant wel kredietwaardig dan wordt een krediet aangemaakt en wordt tegelijkertijd bij het BKR het krediet geregistreerd op naam van de klant.
4. Zodra zowel het krediet is aangemaakt en de registratie bij het BKR is voltooid, dan wordt er een terugmelding gedaan naar de klant. Het proces is hiermee afgelopen.

In de Orchestration module van BizTalk Server 2004 is het scenario weergegeven zoals in Figuur 1.

SCENARIO In figuur 1 is het scherm opgedeeld in een drietal gedeelten. In het midden staat de workflow weergegeven. De workflow geeft de proces stappen weer die achtereenvolgens worden doorlopen, en beslissingen die gemaakt worden tijdens het proces. De workflow wordt gelezen van boven naar beneden. Aan de zijkanten staat de port surface, hier wordt de koppeling gemaakt tussen de workflow en de diverse systemen. Dit worden ook wel de message poorten genoemd. Per

> Met het 'receive' en 'invoke' statement wordt een partnerlink geactiveerd om berichten te ontvangen en te versturen

```
<?xml version="1.0"?>
<process
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
   xmlns:q2="http://KredietAanvraagSchema.Property-
   Schema"
   xmlns:q1="http://KredietAanvraag.Modules"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   name="AanvraagProcess.Orchestration_1"
   targetNamespace="http://KredietAanvraag.Proces"
   xmlns="http://schemas.xmlsoap.org/ws/2003/03/
   business-process/">

<partnerLinks>
   <partnerLink name="OntvangPoort" partnerLink-
   Type="q1:OntvangPoortType"
   myRole="portRole" />
   <partnerLink name="CheckPoort" partnerLink-
   Type="q1:CheckPoortType"
   partnerRole="portRole" />
   <partnerLink name="VerstuurKlantAntwoord"
   partnerLinkType="q1:VerstuurKlantAntwoordType"
   partnerRole="portRole" />
   <partnerLink name="VerstuurRegistratiePoort"
   partnerLinkType="q1:VerstuurRegistratiePoortType"
   partnerRole="portRole" />
   <partnerLink name="MaakRegistratieBerichtPoort"
   partnerLinkType="q1:MaakRegistratieBerichtPoort-
   Type" partnerRole="portRole"
   />
   <partnerLink name="ToewijzingBerichtPoort"
   partnerLinkType="q1:ToewijzingBerichtPoortType"
   partnerRole="portRole" />
   <partnerLink name="AfwijzingBerichtPoort"
   partnerLinkType="q1:AfwijzingBerichtPoortType"
   partnerRole="portRole" />
   </partnerLinks>

<variables>
   <variable name="KlantAanvraag"
   messageType="q1:__messagetype_KredietAanvraag-
   Schema_KredietAanvraagKlant" />
   <variable name="KlantAanvraagAntwoord"
   messageType="q1:__messagetype_KredietAanvraag-
   Schema_KredietControleAntwoord"
   />
   <variable name="KredietRegistratie"
   messageType="q1:__messagetype_KredietAanvraag-
   Schema_RegistreerKrediet" />
   <variable name="KlantAntwoord"
   messageType="q1:__messagetype_KredietAanvraag-
   Schema_AanvraagAntwoord" />
</variables>

<sequence>
   <receive partnerLink="OntvangPoort"
   portType="q1:PortType_1"
   operation="OntvangAanvraag"
   variable="KlantAanvraag"
   createInstance="yes" />

   <invoke partnerLink="CheckPoort"
   portType="q1:PortType_2"
   operation="DoeBKRCheck"
   inputVariable="KlantAanvraag"
   outputVariable="KlantAanvraagAntwoord" />

   <switch>
   <case condition=" bpel:getVariableProperty(
   'KlantAanvraagAntwoord',
   'q2:ControleResultaatCode')  = "toegestaan"">
      <sequence>
         <invoke partnerLink="ToewijzingBericht-
         Poort"
         portType="q1:PortType_8"
         operation="MaakToewijzingBericht"
         inputVariable="KlantAanvraagAntwoord"
         outputVariable="KlantAntwoord" />

         <invoke partnerLink="MaakRegistratie-
         BerichtPoort"
         portType="q1:PortType_5"
         operation="MaakRegistratieAanvraagBericht"
         inputVariable="KlantAanvraag"
         outputVariable="KredietRegistratie"
         />

         <invoke partnerLink="VerstuurRegistratie-
         Poort"
         portType="q1:PortType_6"
         operation="VerstuurRegistratieAanvraag"
         inputVariable="KredietRegistratie" />
      </sequence>
   </case>
   <otherwise>
         <invoke partnerLink="AfwijzingBerichtPoort"
         portType="q1:PortType_9"
         operation="MaakAfwijzingBericht"
         inputVariable="KlantAanvraagAntwoord"
         outputVariable="KlantAntwoord"
         />
      </otherwise>
      </switch>

      <invoke partnerLink="VerstuurKlantAntwoord"
      portType="q1:PortType_7"
      operation="VerstuurAntwoordNaarKlant"
      inputVariable="KlantAntwoord" />
   </sequence>
</process>
```

FIGUUR 2. BPEL script van voorbeeld uit Microsoft Biztalk Server 2004

poort staat aangegeven welke functionaliteit er wordt geboden en welke berichtstromen er zijn voor de betreffende functionaliteit. Zo zijn er requests - binnenkomende berichten en responses - uitgaande berichten. Als voorbeeld in de "CheckPoort" (rechts boven in het plaatje) staat de functie "DoeBKRCheck". Deze functie

verwacht een inkomend bericht van het type "KlantAanvraag" en retourneert een bericht van het type "KlantAanvraagAntwoord".

De transformatie van de berichten vindt plaats achter de message poort. Dit wil zeggen dat de web service die wordt aangeroepen ervoor zorgt dat het inkomende berichttype wordt vertaald in het uitgaande berichttype. Het BPEL script dat aan de hand van het scenario is gegenereerd ziet er als volgt uit.

BLOKKEN Wanneer wordt gekeken naar het script dan zijn er grofweg een viertal blokken waarin het script is opgebouwd.

1. <process></process>
   Binnen het process blok wordt de gehele workflow beschreven. In de <process> tag staat beschreven welke XML name spaces er zijn en waar deze zijn geregistreerd.

2. <PartnerLinks></PartnerLinks>
   In de PartnerLinks sectie worden alle poorten beschreven. Dit zijn inkomende en uitgaande bericht poorten.

```
<partnerLink name="OntvangPoort"
partnerLinkType="q1:OntvangPoortType"
 myRole="portRole" />
```

FIGUUR 3. Voorbeeld van een PartnerLink

In de <partnerLink> tag wordt beschreven wat de naam is van de poort en welk bericht type deze poort gebruikt.Hier staat nog niets beschreven over de richting van de berichten - inkomend of uitgaand.

3. <Variables></Variables>
   In deze sectie worden de berichten beschreven die in het proces gebruikt worden.

```
<variable name="KlantAanvraag"

messageType="q1:__messagetype_KredietAanvraagS
chema_KredietAanvraagKlant" />
```

FIGUUR 4. Voorbeeld van een Variable

In de <variable > tag wordt beschreven wat de naam is van het bericht en van welk type dit is. In de messageType staat de prefix "q1", deze prefix geeft aan in welke XML name space het bericht type is gedefinieerd.

4. <Sequence></Sequence>
   In deze sectie wordt beschreven welke stappen er achtereenvolgens plaatsvinden die als een eenheid kunnen worden beschouwd. Binnen een sequence kunnen er meerdere sequences voorkomen. Een <sequence></sequence> kan dan ook worden beschouwd als een Begin...End statement.

Middels het 'receive' en 'invoke' statement wordt een partnerlink geactiveerd om berichten te ontvangen en te versturen. Hierbij geldt dat de uitkomst (outputVariable) van het ene proces stap de invoer (inputVariable) is voor een andere proces stap.

```
   <invoke partnerLink="CheckPoort"
portType="q1:PortType_2"
operation="DoeBKRCheck"
inputVariable="KlantAanvraag"
outputVariable="KlantAanvraagAntwoord" />
```

FIGUUR 5. Invoke statement

In bovenstaand invoke statement wordt in de processtap "DoeBKRCheck" een partnerlink geactiveerd. Het binnenkomend bericht "KlantAanvraag" wordt doorgestuurd naar de poort "CheckPoort".

Verder worden hier de keuzes gemaakt over welke stappen uit te voeren aan de hand van een processtap uitkomst (dit gebeurt in het <switch></switch> blok).

```
   <switch>
<case condition="
bpel:getVariableProperty( 'KlantAanvraag-
Antwoord',
'q2:ControleResultaatCode')  = "toege-
staan"">
   <sequence>
           ...
           ...
   </sequence>
</case>
<otherwise>
...
</otherwise>
</switch>
```

FIGUUR 6. Switch statement

Binnen het switch statement wordt gecontroleerd of aan bepaalde condities voldaan wordt. Hiervoor maakt BPEL gebruik van interne functies waarmee attributen van een bericht kunnen worden uitgelezen. In bovenstaand voorbeeld wordt middels de functie "getVariableProperty" uit het bericht "KlantAanvraag-Antwoord" het attribuut "ControleResultaatCode" uitgelezen.

Referenties

Meer informatie over BPEL en haar specificatie:

Organisatie	URL
BEA	http://dev2dev.bea.com/technologies/webservices/BPEL4WS.jsp
IBM	http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/
Microsoft	http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnbiz2k2/html/bpel1-1.asp
SAP	http://ifr.sap.com/bpel4ws
Siebel Systems	http://www.siebel.com/bpel

Voorbeelden van werkende implementaties van BEPL Orchestration Servers:

Product	URL
Microsoft Biztalk Server 2004 beta	http://www.microsoft.com/biztalk/
Collaxa 2.0	http://www.collaxa.com/

Meer informatie over OASIS: www.oasis-open.org/

Het switch statement is te zien als een Select Case...End Select of als een If...Then...Else...End If statement te beschouwen.

TENSLOTTE Helaas kunnen in het voorbeeld niet alle mogelijkheden BPEL worden toegelicht. Het voorbeeld laat echter wel de mogelijkheden en kracht van BPEL zien. Aspecten die nog niet aan de orde zijn gekomen zijn onder andere exception handling (met compenserende acties), conditional looping, het voortijdig afbreken van 'workflows' en het toekennen van waarden aan variabelen. Verder wordt in het artikel ook niet ingegaan op zaken als security. Op internet is inmiddels al de nodige informatie over BPEL te vinden, een aanrader is het document "Business Process Execution Language for Web Services" - version 1.1.

Op dit moment kan het inzetten van BPEL zinvol zijn in omgevingen waarbij een geautomatiseerde workflow wordt opgezet of reeds is geïmplementeerd. BPEL biedt in dit geval een grote flexibiliteit om de diverse systemen te vervangen of te updaten zonder dat hiermee het proces wordt verstoord. In omgevingen waar slechts enkele webservices wordt gebruikt is de inzet van BPEL niet meteen zinvol. Zodra het aantal web services begint toe te nemen en er een geautomatiseerde workflow begint te ontstaan dan is het inzetten van BPEL te adviseren, omdat in dit geval vooraf wordt gedefinieerd hoe processtappen samenwerken. BizTalk Server voegt derhalve ook een aantal zaken toe aan orchestration om ook 'workflow' te automatiseren die geen gebruik maakt van web services, bijvoorbeeld de directe aanroep van .NET componenten.

> De specificatie is aan het rijpen en de eerste tools en runtime omgevingen verschijnen thans op de markt

Ondanks dat BPEL nog een nieuwe technologie is, lijkt het een mooie toekomst tegemoet te gaan; de specificatie is op dit moment nog aan het rijpen en de eerste tools en runtime omgevingen verschijnen op de markt. BPEL zal naar verwachting een belangrijke rol gaan spelen in de manier hoe systemen van organisaties met elkaar gaan integreren.

Bas van de Sande
Van de Sande is werkzaam als software architect bij Accenture Technology Solutions.