Sander Schutten

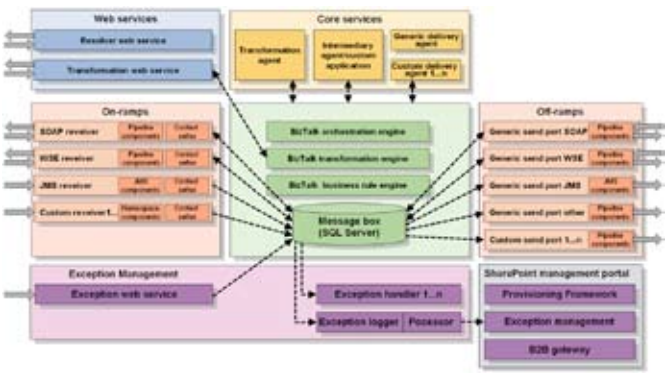
ESB Guidance: meer tijd, minder kopzorgen

ENTERPRISE SERVICE BUS MET BIZTALK SERVER 2006 R2

Microsoft BizTalk Server is sinds 2000 een belangrijk product voor het oplossen van applicatie-integratievraagstukken. De belangrijkste kenmerken van Microsoft BizTalk Server zijn de business procesmanagement-, Service Oriented Architecture- en Enterprise Service Bus-eigenschappen. SOA en ESB zijn ruime begrippen waar de afgelopen jaren veel verschillende definities aan zijn toegekend. Hierdoor zijn tal van verschillende oplossingen ontstaan die alle op hun eigen unieke manier invulling geven aan deze definities.

Het uitbrengen van de ESB Guidance door Microsoft Patterns & Practices heeft als doel bedrijven te helpen een ESB te bouwen volgens de definitie die Microsoft hanteert. De ESB Guidance kan je echter ook helpen als je andere applicaties met BizTalk Server bouwt. ESB is de afgelopen jaren door verschillende bedrijven voornamelijk gepositioneerd als een product, terwijl het eigenlijk om de onderliggende architectuur gaat. Hoewel er geen industriestandaard bestaat, is er een algemene set van karakteristieken die je zult tegenkomen in de meeste ESB-producten:

- Centraal gecoördineerde communicatie. De basisfunctionaliteit van een ESB is het versturen van gegevens tussen processen op dezelfde of verschillende computers. Er hoeft dus niet een grote centrale ESB te zijn, maar het kunnen ook meerdere kleinere ESB's zijn. De ESB gebruikt software als bemiddelaar tussen de zendende en ontvangende partij, waardoor de partijen kunnen communiceren zonder elkaar te kennen.
- Indirecte adressering en intelligente routering. Vaak maken ESB's gebruik van een database om in run-time serviceadressen te achterhalen. Ze zijn ook in staat berichten te routeren op basis van een vastgestelde set van criteria.
- Ondersteuning voor webservices. Een groeiend aantal ESB's ondersteunt basis webservicestandaarden zoals SOAP en WSDL, maar ook fundamentele standaarden zoals TCP/IP en XML.
- Endpoint metadata. ESB's bevatten metadata die de service-interfaces en berichtdefinities beschrijven. Een endpoint is de combinatie van adres, protocol en contract van een service.



Afbeelding 1. De architectuur van de ESB Guidance

ESB Guidance

De ESB Guidance is bedoeld voor architecten en ontwikkelaars die werken met Microsoft BizTalk Server 2006 R2 en die service oriented oplossingen bouwen. De ESB Guidance biedt sturing op het gebied van architectuur, patterns, practices en biedt een set van BizTalk Server- en .NET-componenten voor het vereenvoudigen van het bouwen van een ESB op het Microsoft-platform. Om het helemaal af te maken wordt alle broncode meegeleverd. Dit stelt je in staat om wijzigingen toe te passen, maar het betekent ook dat je zelf de bugs moet oplossen. De architectuur van de ESB Guidance bestaat uit de volgende onderdelen die in de volgende paragrafen uitgebreider uitgelegd worden; zie ook afbeelding 1:

- On/Off-ramps
- Webservices
- Core services
- Exception management
- ESB management portal

On-/Off-ramps

Als je met de auto rijdt, heb je een vertrekpunt en een bestemming. Vanaf jouw vertrekpunt zoek je de meest dichtstbijzijnde oprit voor de snelweg. Om de snelweg op te mogen, moet je aan een heleboel regels voldoen. Zo moet je in een motorvoertuig rijden, je moet een minimale snelheid hebben voordat je mag invoegen en je moet richting aangeven. Op de snelweg mag je de maximumsnelheid niet overschrijden en mag je alleen links inhalen. Eenmaal in de buurt van jouw bestemming gebruik je de afrit om de snelweg weer te verlaten. Je had al geraden dat we met on-/off-ramps de op- en afritten van een snelweg bedoelen. Een ESB kun je vergelijken met een autosnelweg. De auto's op de weg zijn de berichten. Berichten hebben, net als de auto's, een vertrekpunt en een bestemming. Een bericht heeft een oprit nodig om op de ESB te komen en een afrit om er weer af te gaan als hij zijn bestemming bereikt heeft. Ook moeten de berichten aan bepaalde regels voldoen. De on-/off-ramps uit de ESB Guidance zorgen voor een gestructureerde en gestandaardiseerde manier waarmee berichten op de ESB kunnen komen. De meegeleverde on-ramp van de ESB Guidance is een SOAP-/WSE-service. Wil je een ander protocol gebruiken, dan kun je dit bereiken door een andere BizTalk-adapter te kiezen. De on-/off-ramps zorgen uiteindelijk voor het verrijken en/of transformeren van het bericht, het bepalen van het adres van de bestemming en de aflevering van het bericht bij de bestemming. De on-/off-ramps uit de ESB Guidance maken gebruik van een aantal pipeline-componenten.

Met behulp van de pipeline-componenten zijn de on/off-ramps in staat om aan de hand van de metadata in de SOAP-headers van het bericht dynamisch het endpoint van het bericht te bepalen. De metadata kunnen ook worden gebruikt om te bepalen hoe het bericht getransformeerd moet worden. Verderop in het artikel bekijken we deze functionaliteit gedetailleerder.

Namespace normalization

Zoals je wellicht weet is het voor BizTalk Server belangrijk om het type van het bericht te weten. Het type van een bericht wordt bepaald door de combinatie van de target-namespace en de naam van de root-node, bijvoorbeeld: http://schemas.avanade.com/ESBGuidance/V1#MyRootNode. Je kunt je voorstellen dat zonder namespace de kans groot is dat hetzelfde type vaker voorkomt. Dit is niet toegestaan. In dit geval zou je een namespace willen toevoegen. Daarnaast zijn er services die juist niet met namespaces overweg kunnen, zodat je hier het tegenovergestelde wilt doen. Het toevoegen en verwijderen van namespaces op berichten is vervelend werk en kost tijd die je liever aan nuttige zaken wilt besteden. Een onderdeel van de on/off-ramps is namespace normalization. Met behulp van namespace normalization kun je dynamisch namespaces toevoegen aan en verwijderen van berichten. De namespace normalization gebeurt door een speciale pipeline-component. Door de component te configureren kun je bepalen hoe de namespace opgebouwd moet worden. Je kunt een vast deel opgeven, maar ook dynamische delen, die bijvoorbeeld met behulp van xpath uit het bericht gehaald moeten worden.

Core services

Onder de core services valt een aantal agents. Deze agents zijn patterns voor het oplossen van een vaak voorkomend probleem. Je kunt ze vergelijken met bekende programmeer-patterns als het Factory-pattern of het Model-View-Controller-pattern. De agents zijn zo opgezet dat ze dynamisch worden aangeroepen door de ESB op basis van een filterexpressie op gegevens uit de metadata van het bericht. De verzender van het bericht kan zo bepalen welke agents in welke volgorde het bericht moeten verwerken. De on-ramp zorgt ervoor dat de metadata in het bericht terechtkomt.
We nemen als voorbeeld een bericht dat door applicatie A naar service S verstuurd moet worden. We willen dat deze berichten door de volgende agents en in de gegeven volgorde worden verwerkt; zie afbeelding 2:

- Transformation agent
- Fulfillment (custom) agent
- Generic delivery agent

Deze informatie voegen we vervolgens toe aan de Soap-header van het bericht. De Soap-header van het bericht ziet er dan uit als in codevoorbeeld 1. Als we daarna het bericht versturen naar de on-ramp worden de gegevens uit de Soap-header gekopieerd naar de context van het bericht, zodat routering naar de verschillende agents mogelijk wordt. Ter verduidelijking worden in dit artikel de transformation en generic delivery agents nader bekeken.

Transformation agent

Normaliter specificeer je in BizTalk Server de map die je wilt gebruiken om een bericht te transformeren. Er zijn echter gevallen waarbij je een bericht op een andere manier wilt transformeren, bijvoorbeeld op basis van een bepaalde waarde in het bericht of het moment van de dag. Dit betekent dat de keuze van de map die je wilt gebruiken dynamischer moet zijn. De transformation agent laat je zien hoe je dat kunt doen. De transformation agent is een orchestration die luistert naar berichten waar het ProcessingInstruction-element van het bericht de waarde 'TRANSFORM' heeft (codevoorbeeld 1). De orchestration voert vervolgens de volgende acties uit:

1. Een untyped bericht wordt ontvangen. Normaal moet in BizTalk Server een bericht altijd aan een XML-schemadefinitie voldoen. Wanneer je van tevoren de schemadefinitie niet kent, kun je het type System.Xml.XmlDocument gebruiken. Dit is vergelijkbaar met het gebruik van het type object in C#.
2. Als de naam van de map niet in het bericht zit, wordt geprobeerd de naam van de map te bepalen met behulp van de resolver-manager.
3. De map wordt toegepast op het bericht.
4. Het resultaat van het bericht wordt gepubliceerd naar de BizTalk Message Box-database.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <soap:Header>
      <EsbSoapHeaders xmlns="http://Microsoft.BizTalk.ESB.Receivers.
Receivers_SoapHeaders">
         <ProcessingInstruction>TRANSFORM</ProcessingInstruction>
         <Itinerary>TRANSFORM,FULFILLMENT,ROUTE</Itinerary>
         <MapRulesPolicy>Microsoft.BizTalk.ESB.Transformation</MapRules
Policy>
         <MapType/>
         <MapUddiLabel/>
         <MapXPath/>
         <EndpointAddress/>
         <EndpointConfigurationString/>
         <EndpointConfigurationRulesPolicy/>
         <EndpointConfigurationUddiLabel/>
         <EndpointConfigurationXPath/>
         <EndpointDeliveryAgent/>
         <EndpointRulesPolicy/>
         <EndpointUddiLabel>ENDPOINT:TheService.Out</EndpointUddiLabel>
         <EndpointXpath/>
      </EsbSoapHeaders>
   </soap:Header>
   <soap:Body>...</soap:Body>
</soap:Envelope>
```

Codevoorbeeld 1. De Soap-header van het bericht

Generic delivery agent

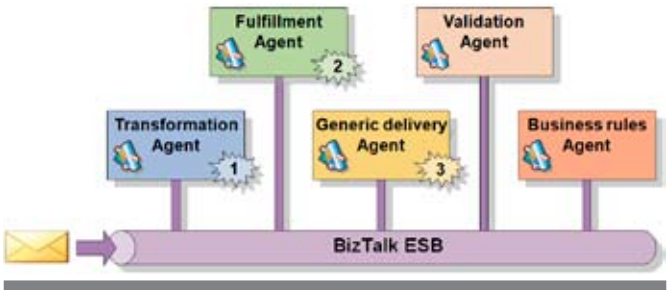
De generic delivery agent laat zien hoe je dynamisch berichten kunt routeren. Je kunt zelf andere delivery agents maken of delivery agents samenstellen die slechts bestaan uit een send-port. De generic delivery agent implementeert een orchestration die luistert naar berichten waar het ProcessingInstruction-element van het bericht de waarde 'ROUTE' heeft. De orchestration voert vervolgens de volgende acties uit:

1. Een untyped bericht wordt ontvangen.
2. Er wordt geprobeerd de endpoints te vinden met behulp van de resolver-manager.
3. De endpoint-gegevens worden op de context van het bericht en de logische dynamische poort ingesteld.
4. Het bericht wordt gepubliceerd naar de BizTalk Message Box-database via de direct-bound-methode op de logische dynamische poort.

Webservices

De ESB Guidance komt out-of-the-box met een aantal webservices. De webservices bieden generieke functionaliteit die essentieel is voor een ESB, namelijk:

- Itinerary webservice. Dit is feitelijk de on-ramp webservice voor het plaatsen van een bericht op de ESB.



Afbeelding 2. Op volgorde verwerken van berichten

```
UDDI:\\serverUrl=http://localhost/uddi;serviceName=TheService;
serviceProvider=Microsoft.Practices.ESB
```

Codevoorbeeld 2. Een voorbeeld van een UDDI-resolver-connectionstring

- Resolver-webservice. Deze webservice zorgt voor het dynamisch bepalen van het adres van een service.
- Transformation-webservice. Deze webservice zorgt voor het dynamisch uitvoeren van een transformatie op een bericht.
- Exception handling-webservice. Via deze webservice kunnen externe applicaties een foutbericht publiceren, zodat deze zichtbaar is in de ESB-managementportal.
- UDDI-webservice. Deze webservice maakt het mogelijk via een aantal verschillende methodes endpoints uit de UDDI-database op te zoeken.
- BizTalk operations-webservice. Via deze webservice is het mogelijk informatie op te vragen over de objecten en berichten in BizTalk Server.

Het gaat te ver om alle webservices te behandelen. Er zijn echter twee webservices die extra aandacht verdienen, omdat ze ook nuttig zijn als je andere applicaties dan een ESB bouwt. Deze webservices, de resolver- en transformation-webservice, zullen we daarom in meer detail behandelen.

Resolver-webservice

De resolver-webservice stelt externe applicaties in staat endpoints op te zoeken met behulp van het ESB Guidance Resolver Framework. Dit framework ondersteunt verschillende methodes voor het opzoeken van de serviceadressen. De resolver-service wordt aangeboden als SOAP- en WCF-WShttp-service. De mogelijke opzoekmethodes zijn: business rules policies, UDDI-registraties, statische aanroepen, WS-MetadataExchange-interface, inhoud van het bericht (xpath) en elk willekeurig .NET-component dat de IResolveProvider-interface implementeert. De resolver-webservice bestaat uit twee operaties: Resolve en ResolveMessage. De Resolve-operatie verwacht een enkele string-parameter voor de resolver-connectionstring. Een resolver- connectionstring lijkt veel op een SQL-connectionstring met als verschil dat een resolver-connectionstring wordt voorafgegaan door een pseudoniem gevolgd door :\\. Een voorbeeld van een UDDI-resolver-connectionstring zie je in codevoorbeeld 2. De overige pseudoniemen zijn: WSMEX:\\, XPATH:\\, STATIC:\\ en BRE:\\. De operatie geeft een verzameling met instanties van de Resolver-class. De Resolver-class bevat informatie als de transportdetails, endpoint-configuratie en contextinformatie. De ResolveMessage-operatie is grotendeels gelijk aan de Resolver-operatie. Het verschil is dat de ResolverMessage-operatie een extra string-parameter verwacht voor een bericht. Deze operatie gebruik je als je een endpoint wilt opzoeken aan de hand van waarden uit het bericht.

Afbeelding 3. Dashboard met foutstatistieken

Transformation-webservice

De transformation-webservice stelt externe applicaties in staat een bericht te transformeren met behulp van een BizTalk-map. In tegenstelling tot het uitvoeren van maps in de send-/receive-ports of de orchestration, gaat bij het gebruik van de transformation-webservice het bericht niet langs de BizTalk Message Box-database. De eigenschap van BizTalk Server om alle berichten in deze database te persisteren, heeft natuurlijk invloed op de performance. Daarbij maakt de transformation- webservice wel gebruik van de mogelijkheden van BizTalk Server om de maps te cachen, waardoor dit een snelle methode is om een bericht te transformeren. De transformation-webservice kent alleen de Transform-operatie. Deze operatie verwacht twee string-parameters: het te transformeren bericht en de volledig gekwalificeerde naam van de gedeployde BizTalk-map. De operatie geeft het getransformeerde bericht als een string terug.

Exception management framework

Als ontwikkelaar zul je weten dat foutafhandeling met BizTalk Server altijd een heet hangijzer is. Er zijn weinig patterns en practices die je kunt gebruiken, waardoor veel bedrijven hun eigen specifieke oplossing hebben bedacht. BizTalk Server 2006 gaf hoop met de toevoeging van de nieuwe Failed Message Routing-feature (zie Suspended Message Routing eenvoudig toe te passen, Dick Dijkstra, .Net Magazine #15). Helaas is deze feature alleen bedoeld voor de receive- en send-ports. Met de komst van de ESB Guidance is er eindelijk de gewilde sturing voor het opzetten van de juiste foutafhandeling binnen alle componenten van BizTalk Server.
Het ESB-exception management framework bestaat uit drie hoofdcomponenten. De eerste component is het ESB Failed Orchestration Exception Routing Mechanism. Hiermee worden fouten in orchestrations op een gelijke manier afgehandeld als bij Failed message-routing. De tweede component is de ESB Fault Processor Pipeline. Deze pipeline zorgt voor het converteren van message-routing- en orchestration-exceptions naar de ESB-standaard, BAM-tracking en conversie van het bericht conform het SQL-adapterschema. De pipeline geeft het bericht door aan de SQL-adapter die het foutbericht opslaat in de database.
Tot slot is er de derde component: de ESB Management Portal and Fault Message Viewer. ESB Guidance komt met een managementportal waarin onder andere een dashboard wordt getoond met allerlei foutstatistieken; zie afbeelding 3. In het dashboard kun je de foutdetails van een bericht bekijken en eventueel het bericht repareren en opnieuw aan de ESB aanbieden.

Meer tijd, minder kopzorgen

Wat je tot dusver hebt kunnen lezen, is slechts het topje van de ijsberg. Het is niet mogelijk alles wat je met de ESB Guidance in handen hebt in een paar pagina's te beschrijven. Gelukkig heeft Microsoft Patterns & Practices op de ESB Guidance-website een goed en compleet helpbestand beschikbaar. Hierin staat duidelijk uitgelegd wat er in de ESB Guidance zit en hoe je het kunt gebruiken. Verder zitten er ook veel voorbeelden in de ESB Guidance die je op weg kunnen helpen. En vergeet niet dat je alle broncode bij de ESB Guidance krijgt geleverd! Als je een architect of ontwikkelaar bent die zich bezighoudt met ESB, SOA en BizTalk, kijk dan eens naar de ESB Guidance. Je kunt er jezelf en anderen een hoop tijd en kopzorgen mee besparen.

Sander Schutten is Enterprise Application Integration specialist bij Avanade (www.avanade.com), een samenwerkingsverband van Microsoft en Accenture. Voor vragen en opmerkingen is Sander te bereiken op sanders@avanade.com of via zijn blog.

Referenties

ESB Guidance-website: http://www.codeplex.com/esb
Microsoft patterns & practices: http://msdn.microsoft.com/practices
Microsoft ESB-website: http://www.microsoft.com/biztalk/solutions/soa/esb.mspx
Sander's blog: http://www.afanaat.nl