Ruim drie jaar geleden ging de webservices missie van start. Het eerste boek ("The Fellowship"?) is inmiddels geschreven en is een groot succes. Maar er zijn nog twee boeken nodig om webservices definitief bedrijfsbreed en wereldwijd te laten doorbreken. Boek twee is al aardig op streek en krijgt in dit artikel een voorbespreking. Aan het slot een kleine vooruitblik op het derde en laatste (?) boek dat nog maar net in de steigers staat.

Webservices: Boek 2

Voorbespreking van het tweede deel uit een trilogie

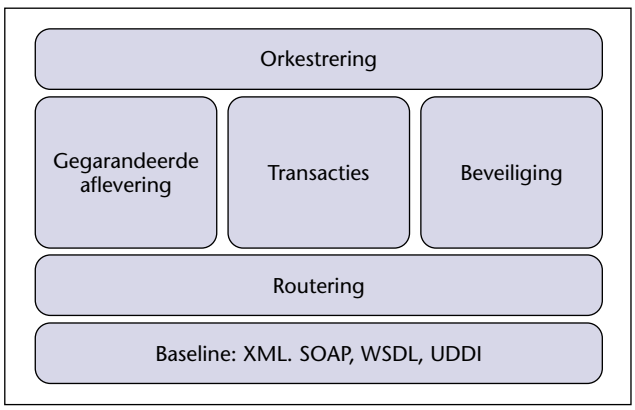
Boek 1 van webservices gaat over het voor de lezers van SRM overbekende trio SOAP, WSDL en UDDI. Het succes van dit eerste boek ligt niet eens zozeer in het feit dat het echt werkt. Nee, het gaat vooral om de enorm brede acceptatie van webservices als standaard voor de interoperabiliteit tussen applicaties, zowel binnen de muren van de onderneming (EAI) als daarbuiten (B2B/B2C). Hoewel Gartner al in 1996 het begrip Service Oriented Architecture (SOA) definieerde, is het pas met de komst van webservices verheven tot de A-status: het universum van de informatieverwerking bestaat uit een stelsel van onafhankelijke webservices die ontsloten en losjes gekoppeld worden door continue stromen van standaard SOAP-berichten.

SUCCESFACTOREN Het succes van Webservices heeft vooral twee oorzaken. Op technisch vlak is men erin geslaagd een relatief eenvoudig en universeel "programmeermodel" te verzinnen dat voor applicatie-naar-applicatie communicatie doet wat het wereldwijde web deed voor mens-naar-informatie/applicatie: het ontsluiten van informatie en functionaliteit door gebruik te maken van universele internetstandaarden zoals TCP/IP, HTTP, HTML, XML en SOAP. Waarbij het dan er volstrekt niet toe doet hoe die webservices zelf geïmplementeerd zijn: met welke taal dan ook en op welk platform dan ook.

De tweede oorzaak van het succes is niet technisch van aard. Het feit dat het webservices model gezamenlijk werd getrokken door twee van de grootste softwarebedrijven - Microsoft en IBM, tevens grote concurrenten van elkaar - heeft niet alleen voor verwondering en bewondering (en jaloezie!) gezorgd, maar vooral ook voor vertrouwen dat het om iets belangwekkends ging. Een nadeel van al die aandacht was wel dat er hier en daar wat overspannen verwachtingen ontstonden.

Want hoewel de eerste fase van webservices aantoonde dat het concept werkte en er ook echt – zij het soms met enige moeite – sprake was van interoperabiliteit tussen implementaties van verschillende leveranciers, werd al wel duidelijk dat er nog het een en ander te doen viel voordat webservices gereed waren voor enterprise 'prime time'. Want hoe zat het met de beveiliging, transacties, orkestratie en routering van berichten? En hoe kunnen webservices worden uitgebreid met SLA's en 'policy's'? Eenvoudige webservice applicaties werkten wel, maar complexe, bedrijfskritische? Het werd tijd voor Boek Twee.

BOEK TWEE Het startpunt van boek twee ligt alweer bijna twee jaar achter ons, toen Microsoft en IBM gezamenlijk aan de W3C een white paper aanboden waarin een duidelijke aanzet gegeven werd tot een veel bredere webservices architectuur (http://www.



FIGUUR 1. Voor alle lagen uit de webservices stack zijn specificaties gepubliceerd

w3.org/2001/03/WSWS-popa/paper51). Inmiddels heeft die aanzet, met veel aanvullingen onderweg, geleid tot een webservices stack die is weergegeven in figuur 1. Voor alle lagen zijn en worden specificaties geschreven en gepubliceerd (over dat proces later wat meer). De benaming van de specificaties start telkens 'ws-' (bijvoorbeeld ws-security) met uitzondering van de orkestratielaag die de naam BPEL4WS (Business Process Execution Language for Webservices) heeft meegekregen.

De onderste laag wordt gevormd door de 'baseline': XML, SOAP, WSDL en UDDI. De toegevoegde waarde ligt in de lagen erboven. Het uitgebreid beschrijven van de diverse lagen is binnen het bestek van een overzichtsartikel niet mogelijk; daarvoor is een reeks van aanvullende artikelen nodig. Daarom hier alleen een korte beschrijving van de lagen.

Intermediairs Bij de routering van berichten gaat het om de abstractie van het werkelijke eindpunt dat de webservice implementeert. De berichten gaan dan niet rechtstreeks van zender naar de webservice leverancier, maar passeren onderweg een of meer 'intermediairs'. Denk daarbij aan proxy's en firewalls die hun eigen authenticatie-controle en herroutering willen doen. Of denk aan 'load balancers' die de binnenkomende berichten verdelen over verschillende instanties van de webservice, misschien wel op basis van afgesproken SLA's ('goldcard' houders versus gewone zenders). Of een hoog beschikbaarheidsmechanisme, waarbij naar een 'fail over' implementatie van de webservice gerouterd wordt zodra de primaire is uitgevallen. De betreffende specificaties hier heten ws-routing en ws-referral.

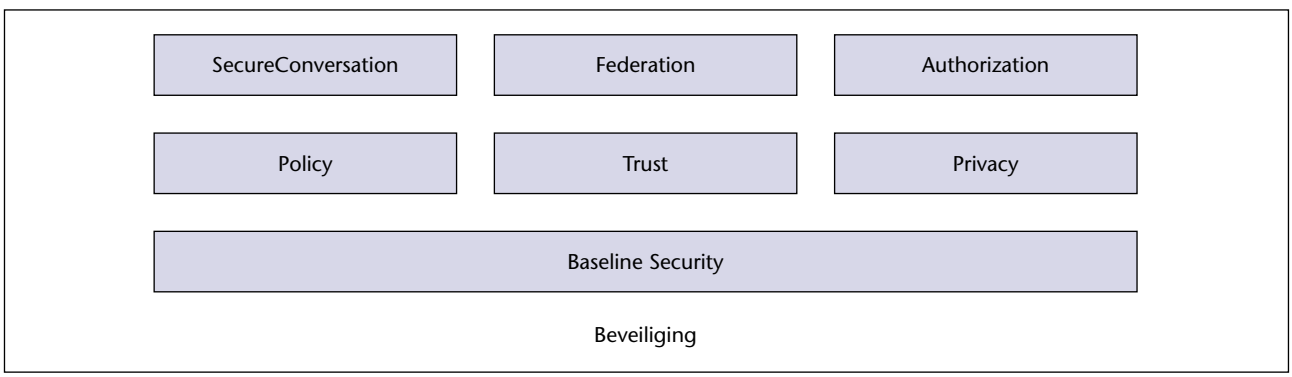
Beveiliging Een goede beveiliging van berichten is waarschijnlijk de belangrijkste voorwaarde voor een bedrijfsbrede en wereldwijde doorbraak van webservices. In april 2002 presenteerden Microsoft en IBM een white paper aan de W3C waarin de architectuur van de beveiliging van webservices uit de doeken werd gedaan. De beveiligingslaag bestaat niet uit één maar uit zeven verschillende specificaties (figuur 2).

De 'baseline' van de beveiligingsspecificatie bestaat uit ws-security en betreft de authenticatie, integriteit en (partiële) encryptie van SOAP-berichten. De aanvullende specificaties zijn pas recentelijk (december 2002) in eerste versie gepubliceerd. Een voorbeeld daarvan is ws-policy waarin het beleid rondom het gebruik van webservices centraal staat. In zo'n policy-bericht wordt bijvoorbeeld opgenomen dat de betreffende web service alleen digitale certificaten als authenticatie accepteert die minder dan vijf minuten oud zijn.

Transacties De transacties specificatie (ws-transaction) is in twee delen opgesplitst: 'Atomic Transaction' en 'Business Activity'. Bij het eerste gaat het om standaard (ACID-) transacties inclusief 'two-phase commit' protocol. Het tweede deel draait om (langdurige) 'business' transacties. In beide gevallen gaat het om het bewaren van de integriteit en consistentie van de transactie. De standaard transactie maakt daarbij gebruik van isolatie (via 'locking') van alle in de transactie participerende webservices; iets wat de beschikbaarheid en doorvoersnelheid van webservices ernstig kan belemmeren. Dit moet dan ook alleen bij transacties worden toegepast die volledig binnen de grenzen van de onderneming plaatsvinden. De business transacties gaan uit van compenserende maatregelen als de consistentie in het geding komt en zijn daardoor beter geschikt om te worden gebruikt bij langdurige transacties of voor transacties die deels buiten de ondernemingsgrenzen plaatsvinden.

De specificaties rondom gegarandeerde aflevering zijn nog in de maak en zullen binnenkort verschijnen. De gegarandeerde aflevering moet daarbij losstaan van het onderliggende transportmechanisme.

Bij de orkestratie tenslotte gaat het om de samenvoeging van webservices tot bedrijfsprocessen. Het bedrijfsproces is daarbij leidend en sturend. Vanuit het bedrijfsproces worden webservices aangeroepen. Wel kunnen webservices op hun beurt nieuwe bedrijfsprocessen initiëren of bestaande, maar 'slapende' activeren.



Figuur 2. In het whitepaper van de W3C wordt de beveiligingsarchitectuur van webservices nader gespecificeerd

ONTWERPPRINCIPES Maar hoe passen al die afzonderlijke specificaties nu in de overkoepelende webservices architectuur? Drie fundamentele ontwerpprincipes liggen aan die architectuur ten grondslag: decentraal, modulair en transport neutraal.

Decentraal: er is geen centraal regelmechanisme zoals bij klassieke EAI/B2B-gebaseerde 'message brokers' waarbinnen zaken als routering, autorisatie en transformatie vaak centraal in de 'hub' plaatsvinden. Bij webservices hoeven beide partijen - respectievelijk de consument en leverancier van de web service – het alleen maar eens te zijn over de syntax en semantiek van de berichten die uitgewisseld worden. Centrale regulering ontbreekt en dit komt onder meer de schaalbaarheid ten goede.

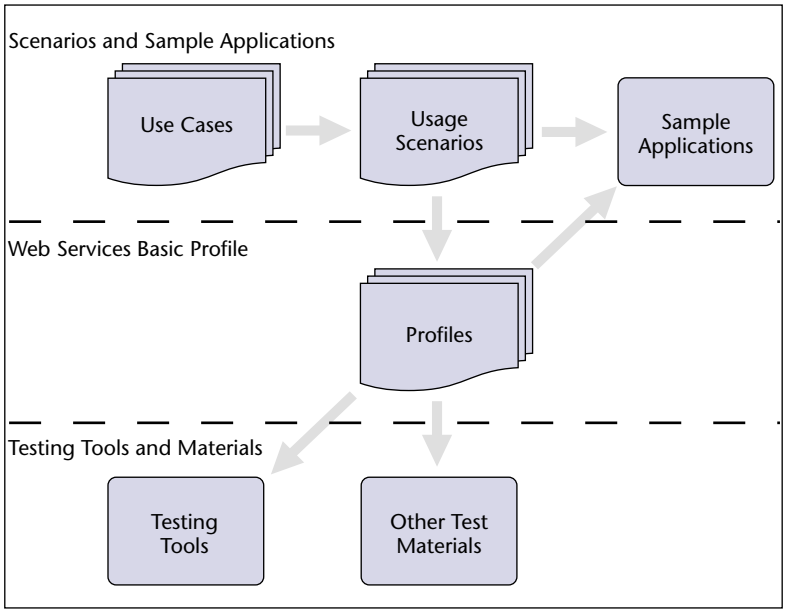
Modulair: het credo is: 'gebruik alleen wat je nodig hebt'. Niet elk bericht hoeft indirect gerouteerd, beveiligd én transactioneel te zijn. De lagen uit figuur 1 zijn volledig onafhankelijk ('orthogonaal') van elkaar te gebruiken.

Transport neutraal: de webservices architectuur is volledig gespecificeerd op het niveau van SOAP-berichten (of specifieker: SOAP-headers) en staat los van het onderliggende transportmechanisme. Hoewel in de praktijk veelal http gebruikt wordt, kan het SOAP-bericht net zo goed direct over TCP/IP, message queues of SMTP (e-mail) transporten gaan. In theorie kan men een volledig valide, veilig en transactioneel SOAP-bericht zelfs per postduif versturen; zelfs met gegarandeerde aflevering (ondanks de kiekendieven en katten onderweg; mits zender en ontvanger over een onuitputtelijke hoeveelheid duiven beschikken).

HET PROCES Microsoft en IBM zijn de trekkers van de specificaties van de webservices architectuur en gebruiken daarbij het volgende proces.

Allereerst wordt van een domeingebied een 'draft' specificatie gemaakt. Daarbij wordt een beroep gedaan op bedrijven die hun sporen in dat domein verdiend hebben. Voor de specificaties rondom beveiliging hebben Verisign en RSA Security meegewerkt. BEA heeft onder meer meegedaan met de specificaties rondom transacties en orkestratie. Daarna wordt vrij snel een eerste (vaak: interne) productimplementatie gemaakt van de draft specificaties, waarin een en ander getest kan worden, wat weer aanleiding kan geven tot een aanpassing op de specificaties.

Ook wordt snel een officiële standaardisatie organisatie gezocht die het best past bij de specificaties. Voor de basisniveau specificaties van SOAP en WSDL is een beroep gedaan op de W3C (www.w3c.org), voor de specificaties rondom beveiliging en coördinatie & transacties op OASIS (www.oasis-open.org).



FIGUUR 3. De aanpak van de Webservices Interoperability Organisation (WS-I) schematisch weergegeven.

AANVULLING Als de specificaties enigszins stabiel zijn wordt een nieuw product gemaakt dat meestal als aanvulling op bestaande runtime en ontwikkelomgevingen geldt. Zo heeft Microsoft in december de Webservices Enhancement (WSE-) Toolkit gelanceerd dat een aanvulling vormt op het .Net Framework en VS.NET. IBM heeft de Webservices ToolKit (WSTK) uitgebracht. Het voordeel hiervan is dat men snel de beschikking heeft over de technologie; het nadeel is dat er vooral in de beginfasen wijzigingen in de specificaties kunnen komen en dus ook in de toolkits. De toolkits worden vaak wel ondersteund maar er wordt in verband met de mogelijke wijzigingen in specificaties geen terugwaartse compatibiliteit gegarandeerd.

Als de specificaties echt definitieve vormen aangenomen hebben zullen de aanvullingen worden opgenomen in de kernproducten en komen er nieuwe aanvullingen in de vorm van toolkits voor de meest recente specificaties.

INTEROPERABILITEIT Het succes van webservices is niet afhankelijk van goede specificaties of goede individuele productimplementaties. Neen, het succes staat of valt met de interoperabiliteit tussen die verschillende productimplementaties. Daartoe is in februari 2002 de Webservices Interoperability Organisation (www.ws-i.org) opgericht door Accenture, BEA, Fujitsu, HP, IBM, Microsoft, Oracle en SAP. De WS-I heeft niet alleen tot taak om de adoptie van webservices te promoten, maar ook om voorbeeldapplicaties en testsuites voor interoperabiliteit te maken en producten te 'certificeren' die

de tests doorstaan. De aanpak van de WS-I staat weergegeven in figuur 3. Eind oktober is het eerste 'profile' verschenen op basis waarvan men zowel testsuites als referentie-applicaties is gaan maken. Het eerste 'profile' is nog uitsluitend gebaseerd op de 'baseline' standaarden: SOAP 1.1, WSDL 1.1, UDDI 2.0, XML 1.0 en XML Schema. Latere profiles zullen de nieuwere specificaties meenemen.

Boek 3 De specificaties voor bedrijfskritische webservices zijn onderweg, de eerste producten komen mondjesmaat binnen en de broodnodige interoperabiliteit tussen die verschillende producten wordt op het juiste – onafhankelijke – niveau aangepakt. Maar is dat voldoende voor het gebruik van webservices in bedrijfskritische omgevingen?

Het antwoord is: noodzakelijk, maar niet voldoende. Er is nog een derde en wellicht laatste boek nodig dat de werktitel 'patterns en practices' draagt. Wat velen zich nog niet echt realiseren is dat er een nieuw paradigma op komst is, namelijk "message & service based computing". Het omgaan met berichtuitwisseling in een service-georiënteerde omgeving is iets geheel anders dan het traditionele, meest synchrone client/server computing waarop de meeste ontwikkelaars getraind en door de wol geverfd zijn. Men kan webservices wel op die manier ontwikkelen, maar dat is eigenlijk tegennatuurlijk en contraproductief om meerdere redenen. In de eerste plaats is messaging van nature asynchroon en in principe eenrichtingsverkeer. Er wordt een bericht verstuurd (een verzoek), maar dat hoeft niet noodzakelijkerwijs tot een antwoordbericht te leiden. En als er al een antwoord verwacht wordt, dan is de 'latency' daarvan onbekend. Op die onzekerheid zal men de programmalogica van de web service consument moeten inrichten. Een ander aspect is dat de webservice leverancier alle binnenkomende berichten in principe moet wantrouwen omdat ze van buiten komen; elk bericht is een potentiële indringer. Een goede beveiliging op berichtniveau is cruciaal en moet ook aan de consument gecommuniceerd worden. Tenslotte zijn de mate van fijn- of grofmazigheid van de webservices en de wijze waarop webservices tot business transacties worden georkestreerd van belang.

Net begonnen Dit alles vereist ondersteuning vanuit architectuur en van daaruit meer gedetailleerde beschrijvingen van patterns en best practices. Binnen Microsoft is hier op conceptueel niveau al over nagedacht (www.microsoft.net/architecture). Maar een verdere detaillering en operationalisering naar patterns en best practices is nodig. En dat is het derde boek waaraan nu begonnen wordt.

Dik Bijl is werkzaam bij Microsoft als enterprise architect (e-mail: dikbijl@microsoft.com)

PATCHES Patches PATCHES Patches PATCHES Patches PATCHES

SAP annonceert SAP Enterprise Services Architecture en SAP NetWeaver

SAP noemt de release van zijn SAP Enterprise Services Architecture (ESA) en SAP NetWeaver, afgelopen januari, 'de belangrijkste aankondigingen op het gebied van de platform- en applicatiearchitectuur van SAP, sinds de introductie van de schaalbare drielaags client/server-architectuur in 1992'. SAP Enterprise Services Architecture (ESA) maakt het mogelijk om individuen, informatie en systemen met elkaar te verbinden. ESA is een blauwdruk voor volledig op webservices gebaseerde bedrijfsapplicaties die het bedrijven mogelijk maakt hun bestaande applicaties en systemen beter te benutten doordat zij hun bedrijfsprocessen makkelijker op elkaar kunnen afstemmen. Met ESA is het voor het eerst mogelijk om op bedrijfsniveau gebruik te maken van webservices. Deze webservices dragen zorg voor de informatie-uitwisseling tussen verschillende applicaties, zowel binnen als buiten de onderneming, via het bedrijfsnetwerk of via Internet. Vanaf nu zullen alle SAP-oplossingen worden ontwikkeld op basis van deze ESA-blauwdruk.

SAP NetWeaver, een product van mySAP Technology, is het integratieplatform voor bedrijfsapplicaties die gebaseerd zijn op webservices. NetWeaver is de basis van de SAP-oplossingen en biedt een volledig open infrastructuur, die ook andere niet-SAP applicaties kan omvatten. SAP NetWeaver werkt volledig samen met de webservices-architecturen van Microsoft (.NET) en van IBM (Java 2 Platform, Enterprise Edition), waardoor organisaties hun huidige applicaties beter kunnen integreren. Door gebruik te maken van SAP NetWeaver is geen afzonderlijke integratiesoftware nodig voor elke applicatie.

SAP NetWeaver kent twee nieuwe onderdelen: Composite Application Framework en Master Data Management (MDM). Het Composite Application Framework maakt het voor klanten mogelijk om applicaties te ontwikkelen voor meerdere bedrijfsprocessen of onderdelen daarvan. Dit raamwerk biedt onder meer tools en methodologieën om tot een homogene ontwikkel- en implementatie-omgeving te komen, die de software-ontwikkelaar afschermt van verschillende onderliggende systemen en applicaties. Master Data Management is de eerste gestandaardiseerde service die het mogelijk maakt om via het bedrijfsnetwerk data uit verschillende systemen te integreren. Hierdoor verbeteren de mogelijkheden voor organisaties om hun gegevens te consolideren en centraal te beheren, ook wanneer zij werken met systemen en applicaties van verschillende leveranciers.