KAN DE IT-AUDITOR EEN MEDEDELING AFGEVEN?

Risico's van Service Oriented Architecture

Service Oriented Architecture (SOA) is een IT-strategie die de functies in de ondernemingsbrede toepassingen realiseert via services. Een SOA bestaat uit een complex en dynamisch netwerk van services die tezamen de bedrijfsprocessen moeten ondersteunen en onderling informatie uitwisselen, mogelijk zelfs via internet. De IT-auditor zal in staat moeten zijn deze dynamiek en complexiteit te doorgronden om voldoende onderbouwing te vinden voor een goedkeurende mededeling.

KOOS ZIERE

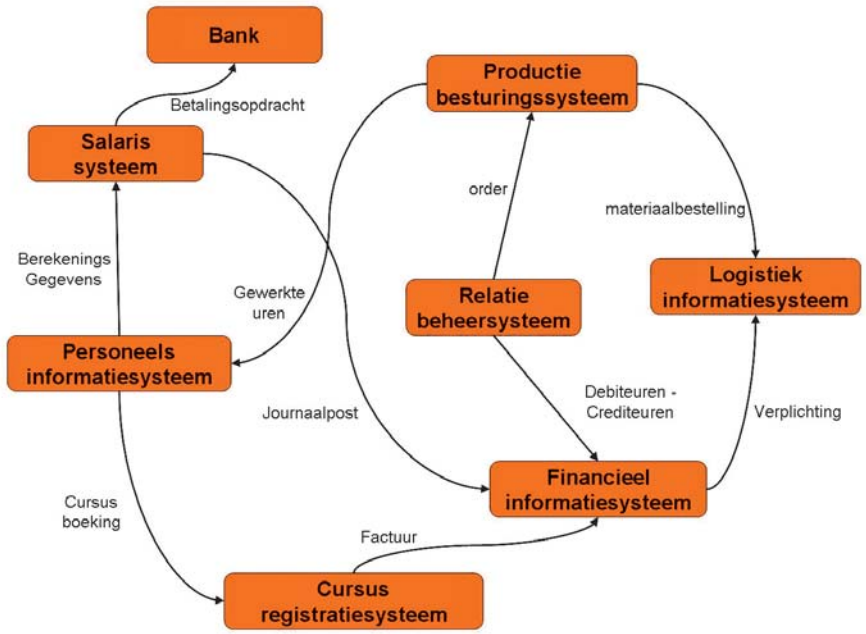
SOA is een IT-strategie gebaseerd op het ondersteunen van de bedrijfsprocessen door zogenaamde services. Deze services kunnen als bouwstenen worden gebruikt om IT-ondersteuning snel aan de behoeften van de bedrijfsprocessen te kunnen aanpassen. Organisaties die in staat zijn hun IT te laten aansluiten bij hun business doelstellingen, hebben een competitief voordeel. SOA lijkt hiermee de panacee voor het flexibel door IT ondersteunen van processen in de organisatie.
Vanuit oogpunt van flexibiliteit en aanpasbaarheid is een bouwsteenmodel te verkiezen boven de monolithische legacy-systemen. In theorie kunnen de services, in de meeste gevallen gebaseerd op webtechnologie, zich overal ter wereld bevinden. De grenzen liggen niet bij die van het bedrijfsnetwerk, maar pas daar waar geen netwerkaansluiting meer beschikbaar is.
Netwerken (zoals) internet, intranet, extranet vormen de basis waarover de services van een SOA communiceren. De risico's van een www (web-) toepassing hebben we nog maar nauwelijks onder controle, laat staan een complex netwerk van services, die tezamen de bedrijfsprocessen moeten ondersteunen en hun informatie uitwisselen via internet.

SERVICE ORIENTED ARCHITECTURE

In de meest uitgebreide vorm bestaat een toepassing op basis van SOA uit een dynamisch aantal services, gekoppeld via één of meerdere netwerken, die met elkaar het bedrijfsproces ondersteunen. Services zijn modules met een gedefinieerde functie, input en output, die via een communicatie-interface worden aangeroepen. Het zijn dus geen subroutines of objecten in de traditionele zin van modules. Tussen services is sprake van transacties.
In een IBM White Paper geeft Kishore Channabasavaiah [CHAN03-1] de volgende definitie van een SOA:

An application architecture within which all functions are defined as independent services with well-defined invokable interfaces which can be called in defined sequences to form business processes

Een belangrijk aspect in deze definitie is dat er geen sprake is van systemen, programma's, et cetera, maar dat de architectuur is bedoeld om bedrijfsprocessen te ondersteunen. De onafhankelijke services, met goed gedefinieerde interfaces vormen zogezegd de legostenen, waarmee, binnen een architectuur, de bedrijfsprocessen kunnen worden ondersteund. Veranderen de bedrijfsprocessen, dan kan een rode legosteen eenvoudig door een blauwe worden vervangen, mits hiermee de architectuur geen geweld wordt aangedaan.
Een tweede aspect is dat de interface goed en eenduidig gedefinieerd moet zijn. Ook hier gaat de analogie met de legosteen weer op: als de rondjes op de ene steen niet passen in de onderkant van de andere steen, ▸



Figuur 1: Relaties tussen systemen

kan er geen gebouw tot stand komen, laat staan onder architectuur worden gebouwd. De architectuur legt ons echter geen beperkingen op als het gaat om de bij het modelleren van de businessprocessen te gebruiken 'bouwstenen' (de services). Dit betekent dat de services niet alleen in eigen huis hoeven te worden aangeboden en afgenomen. Het is zo dat netwerken en communicatiemogelijkheden het toelaten dat services worden gebruikt die worden aangeboden door derden.
Is de situatie van een SOA met alle services in eigen huis (misschien) nog te overzien, bij het afnemen van services van derden nemen complexiteit en risico exponentieel toe. In de ultieme variant van een SOA gaat de afnemer van een dienst via een directory op zoek naar de gewenste service, gebruikt hem en rekent (desnoods) af.

Vergelijking met Architectuur in de bouw

Het vergelijken van de architectuur van een SOA met de architectuur van een gebouw ligt voor de hand, maar is niet juist. Een gebouw is bedoeld om gedurende een langere tijd te worden gebruikt en de basisstructuur van het gebouw zal voor de totale levensduur vast liggen. Een SOA dient flexibel te zijn om aan de dynamiek van de veranderende organisaties te kunnen voldoen. Als je dat vergelijkt met een gebouw, zou dat betekenen dat de plek van de liften en trappenhuizen naar believen kan worden gevarieerd en dat er verdiepingen kunnen worden weggehaald en toegevoegd of zelfs de hele lay-out van het gebouw kan worden gewijzigd. [BARR03] De starre structuur van een gebouw kan dus niet vergeleken worden met de dynamische structuur van een SOA.

TERUGBLIK

Vanaf de jaren vijftig van de vorige eeuw vond de geautomatiseerde gegevensverwerking plaats op grote, monolithische en centraal opgesteld computersystemen, de mainframes. Integratie tussen systemen is historisch gegroeid en iedere interface is specifiek geprogrammeerd. Het is duidelijk dat het vervangen van één van de systemen, bijvoorbeeld het financiële informatiesysteem, een enorme inspanning vergt om de specifieke interfaces met de overige systemen weer werkend te krijgen. Een globaal overzicht van zo'n historische omgeving is opgenomen in figuur 1.

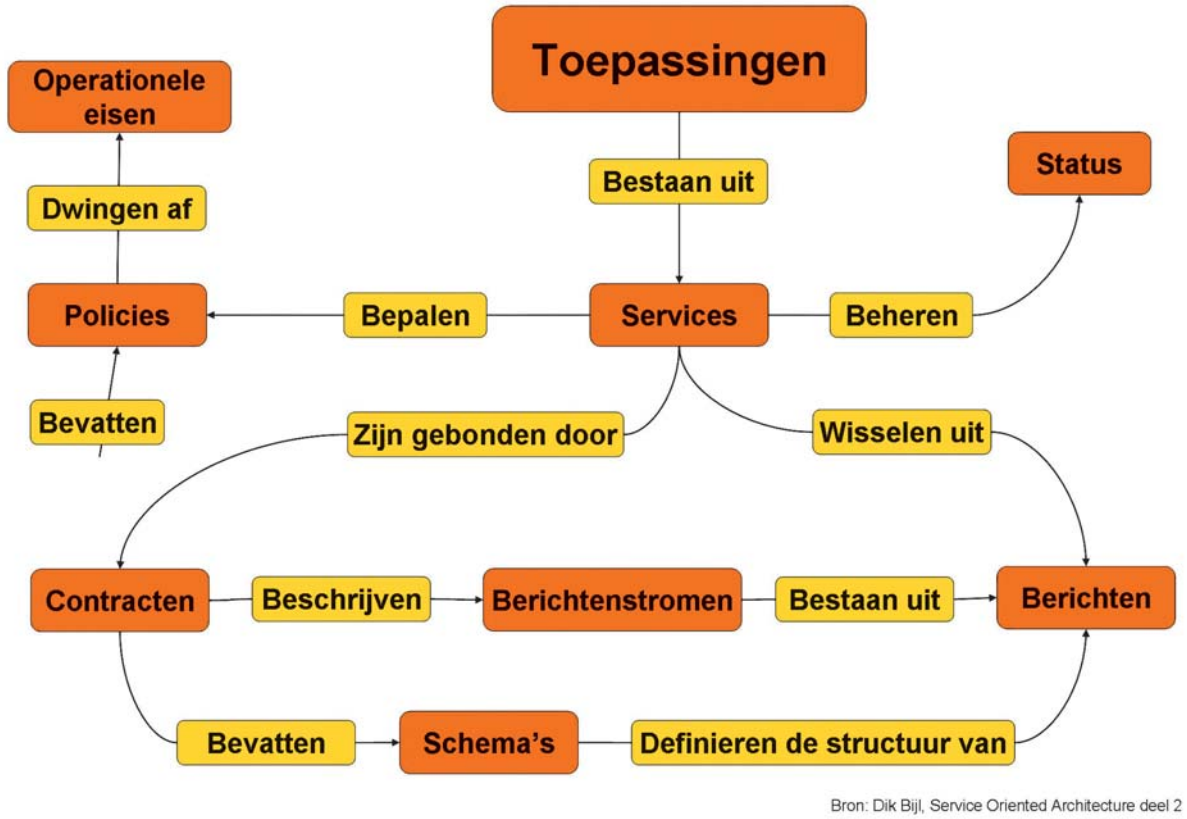
Deze vorm van interfacing wordt ook wel aangeduid met de term *tightly coupled*. De uitwisseling van gegevens tussen twee systemen is zeer strak gekoppeld aan de twee systemen. Dergelijke koppelingen hebben meestal een synchrone verschijningsvorm: als het ene systeem iets verstuurt, zal het op het antwoord van het andere systeem wachten.

SOA EN WEBSERVICES

In het afgelopen decennium heeft het World Wide Web (WWW) bewezen bestaansrecht te hebben en zijn er rond internet en zijn toepassingen vele nieuwe standaarden geboren. Na een aarzelende start, waarin alleen 'platte informatie' werd aangeboden, is het web nu een plaats geworden waar informatie wordt uitgewisseld en handel wordt gedreven. Het ontwikkelen van webgebaseerde toepassingen is voortgekomen uit het gebruik van de reeds beschikbare technologie. Zo is HTTP op alle systemen te genereren, maar veel belangrijker: ook te interpreteren. Hiermee is een universeel middel geboren om gegevens tussen niet gelijke systemen uit te wisselen.
Een belangrijke stap voorwaarts in webservices, de bouwstenen voor SOA, is de ontwikkeling van XML (eXtensible Markup Language). Als subset van het veel complexere SGML (Standard Generalized Markup Language) is een uitbreidbare beschrijvende taal ontwikkeld, waarmee uitwisseling van berichten (gegevens) tussen heterogene systemen mogelijk is.

Services als bouwstenen

De webservices zijn inmiddels zover volwassen geworden dat zij autonoom berichten kunnen ontvangen, verwerken en verzenden. Deze services kunnen als bouwstenen fungeren om toepassingen te bouwen voor het ondersteunen van bedrijfsprocessen. Doordat de interactie tussen services bestaat uit het uitwisselen



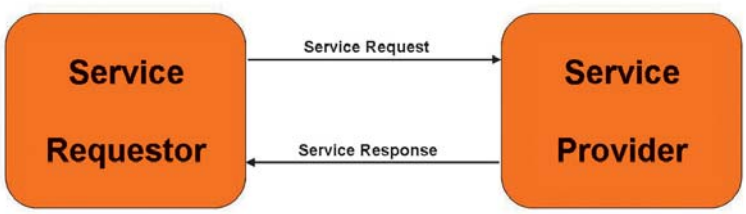
Figuur 2: Definitie van Services volgens Microsoft

van berichten, is het eenvoudig om een service of bouwsteen te vervangen door een andere service met een vergelijkbare (maar niet noodzakelijkerwijs dezelfde) interface. Hierbij is de beschrijvende XML van grote betekenis, omdat hiermee de starre interfaces op basis van uitwisselingsrecords kunnen worden doorbroken.

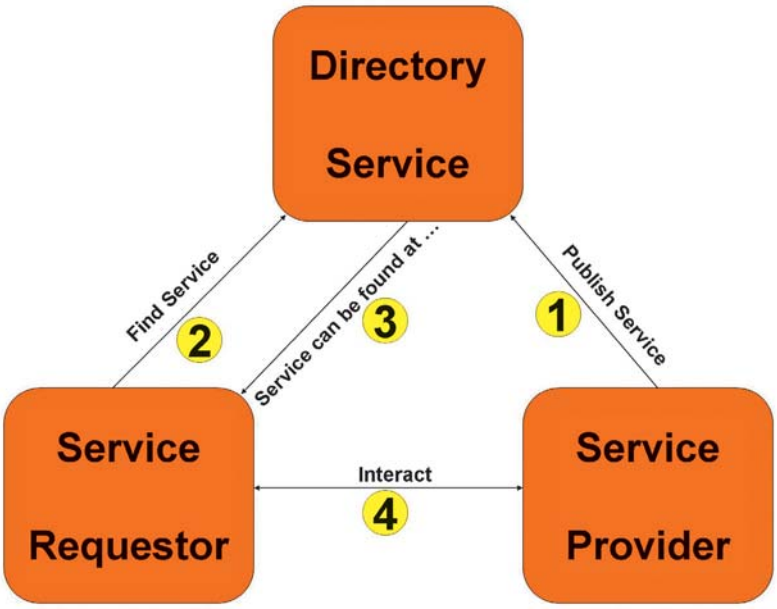
Microsoft gaat bij het gebruik van services binnen een SOA uit van de in figuur 2 grafisch weergegeven definitie [BIJL05]. In de figuur staat de toepassing midden bovenin. Deze bestaat uit een aantal services die zelf hun data en status bijhouden (rechtsboven), berichten uitwisselen en gebonden zijn aan contracten met afspraken (linkerzijde). Microsoft gaat uit van het bestaan van een formele afspraak (contract) tussen de gebruiker en aanbieder van services voordat van de service gebruik gemaakt wordt. Dit is een flinke beperking van de meer open definitie van Channabasavaiah [CHAN03-1], waar alleen sprake is van uitwisseling van berichten op een goed gedefinieerde wijze.
Sun Microsystems [SUN04] gaat bij webservices ook uit van een contract, maar dat kan realtime (op runtime) worden afgesloten tussen servicevrager en service aanbieder.

BASISMODEL VAN EEN SOA

Alle toepassingen op basis van SOA bestaan in de basis uit dienstenaanbieders (Service Providers) en afnemers van de aangeboden services, de Service Requestors (figuur 3). In dit basismodel verzoekt de afnemer de dienstenverlener een dienst te verlenen en de dienstverlener zal daarop een antwoord zenden.
Dit basismodel gaat ervan uit dat de aanbieder van de dienst bekend is, de dienst exact beschreven is, de aanbieder vertrouwd kan worden, dat er altijd tijdig een respons komt op een verzoek, et cetera. In werkelijkheid is de communicatie over en weer echter veel gecompliceerder, bijvoorbeeld omdat er met de aanbieder van de dienst nog autorisatie en authenti- ▸



Figuur 3: Basismodel van een SOA



Figuur 4: Gebruik van een registry bij het vinden van services

catie plaats moet vinden of dat er met de aanbieder een beveiligde verbinding (encryptie) moet worden opgezet.
Uitwisseling van request en response zal in de meeste gevallen gebaseerd zijn op SOAP (Simple Object Access Protocol, WWW consortium) [W3C03-1]. SOAP is een op XML gebaseerd protocol voor het uitwisselen van berichten.

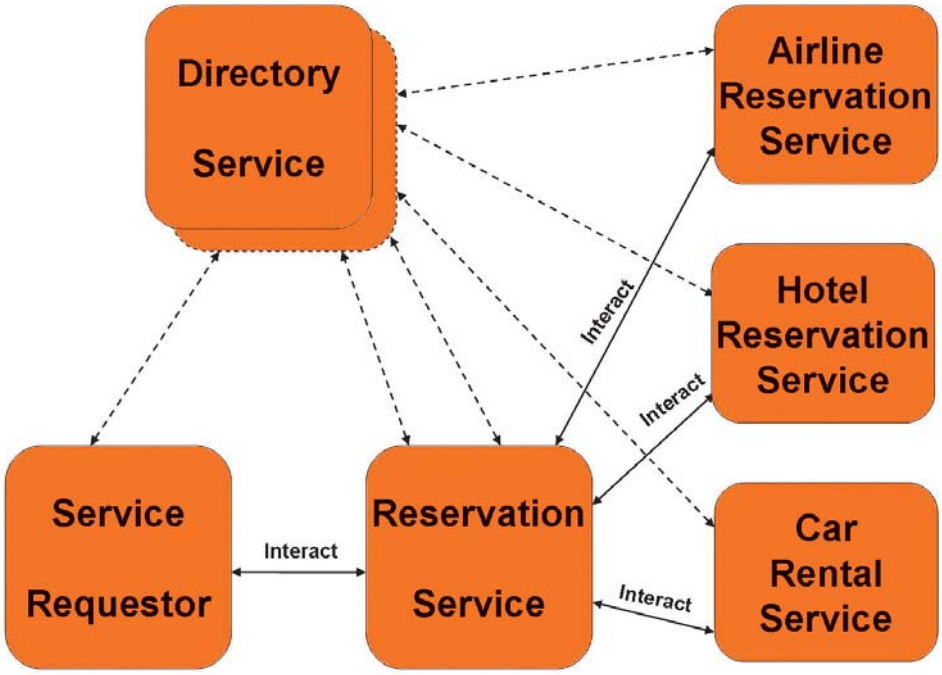
Gebruik van een registry - UDDI
Het basismodel gaat ervan uit dat de aanbieder van de dienst bekend is. In de praktijk zal dit echter lang niet altijd het geval zijn. De afnemer moet dan op zoek naar een aanbieder van de gewenste dienst. De afnemer zal daarvoor een registry (of directory) raadplegen en daarin de gewenste dienst opzoeken. In het antwoord van de directory staat een beschrijving van de dienst, waar de dienst gevonden kan worden en hoe daarmee gecommuniceerd kan worden. Ook zijn daarin de voorwaarden (contract) opgenomen waaronder gebruik kan worden gemaakt van de betreffende service.
Eén van de standaarden van OASIS (Organization for the Advancement of Structured Information Standards) is Universal Description, Discovery and Integration (UDDI) standaard. Deze beschrijft het publiceren en opzoeken van services in een registry. [OASI05-1, OASI05-2]

Het proces gaat werkt als volgt (zie figuur 4):

1. De Service Provider is de webservice die via het netwerk bereikbaar is en services levert. De provider publiceert zijn services en de voorwaarden waaronder deze gebruikt kunnen worden (contract in de registry).
2. De Service Requestor is een toepassing die een bepaalde service nodig heeft. De Service Requestor initieert een zoekactie in de registry.
3. De registry retourneert aan de Service Requestor een bericht met de resultaten van de zoekopdracht. In dit bericht staat ook vermeld hoe en onder welke voorwaarden de service wordt verleend.
4. De Service Requestor verzoekt vervolgens de Service Provider van zijn keuze om de opdracht uit te voeren. Dit gebeurt door het sturen van een verzoek (bericht), dat is geformatteerd conform de voorwaarden uit het zoekresultaat (contract). Dit is de start van de transactie.

Transacties en hun integriteit
Een transactie zal in veel gevallen betekenen dat er meerdere berichten worden uitgewisseld, totdat het gewenste resultaat is bereikt. Indien de transactie niet tot een positief en gewenst resultaat leidt, dan is het noodzakelijk dat er een mechanisme in werking treedt dat de oorspronkelijke situatie herstelt.

Veronderstel dat je een vliegticket naar New York en een hotelkamer wilt boeken. In dat geval wil je alleen boeken als je beiden kunt krijgen. Als je de vluchtreservering bevestigt en geen hotelkamer kunt boeken, moet je de vlucht vervolgens weer annuleren. Omgekeerd blijf je met een nutteloze hotelkamer zitten als er geen vlucht beschikbaar blijkt te zijn.
Dit probleem bestond al lang voor de introductie van elektronische gegevensuitwisseling. Kaye [KAYE03] signaleert in zijn boek dat er nog geen standaarden zijn om complexe transacties tussen verschillende organisaties (en systemen) gecontroleerd te laten verlopen en juist en volledig af te handelen. Ook in 2009 is daarvoor nog geen oplossing vastgelegd.
Het voorbeeld met de reis naar New York levert meteen een mogelijkheid op om het basismodel van een SOA uit te breiden. Stel dat er een Reservation Service is die er voor zorgt dat je naast de hotelkamer en het vliegticket ook nog een huurauto krijgt. In figuur 5 is weergegeven dat de Reservation Service zelf weer een aantal services aanroept en er voor zorgt dat alleen een positief resultaat wordt teruggegeven als zowel het ticket, de hotelkamer èn de auto zijn geboekt.
Met het toenemen van het aantal services dat betrokken is bij een proces, neemt het aantal transacties ook sterk toe. Indien één van de transacties niet het gewenste resultaat ople-



Figuur 5: Samengestelde service

vert, zal de gehele reservering van de reis naar New York niet slagen.

Communicatie tussen services: loosely coupled

Bij uitwisseling van gegevens tussen services is sprake van loosely coupled-communicatie. De historische tight-koppeling is hier ongeschikt voor. Anders dan bij een tight-koppeling, zal de gevraagde service in een SOA niet altijd direct de gewenste actie kunnen uitvoeren. In dat geval kan de service een bevestiging van de aanvraag retourneren en op een later moment de gevraagde bewerking uitvoeren en eventueel de resultaten opleveren. Door dit gedrag zullen de services niet wachten op de antwoorden, maar verdergaan en de antwoorden op een later moment relateren aan de eerdere vraag. De reactietijd ligt mogelijk in de orde van minuten of zelfs uren. Loose coupling biedt deze mogelijkheid en kan zelfs worden beschouwd als een voorwaarde.

Kenmerk is ook dat SOA-services zich weliswaar van elkaar 'bewust' zijn, maar tevens een zo gering mogelijke onderlinge afhankelijkheid hebben; SOA-modules worden niet gelinkt of geassembleerd.

Koppeling van services bestaat uit het uitwisselen van SOAP-berichten. Door het asynchrone karakter van de berichtenuitwisseling, zal de ontvangende service de berichten afhandelen op een passend moment en pas daarna het resultaat retourneren. In de praktijk zien we dat berichten tussen services ook wel worden verstuurd via een message broker (message queueing), die er voor zorgt dat berichten worden afgeleverd bij de juiste service, op het moment dat die beschikbaar is.

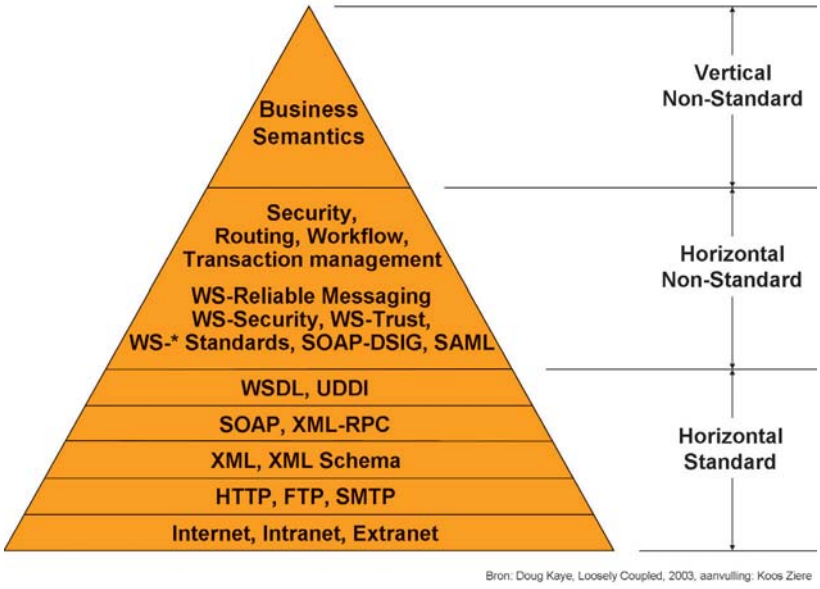
Data-Centric Architecture versus Process Centric Architecture

Een praktische vraag bij de inzet van een SOA is: waar zijn de gegevens opgeslagen? Barry onderscheidt hierbij een tweetal uitgangspunten: de Data-centric Architectuur en de Process-centric Architectuur [BARR-03].

In de Data-centric Architectuur worden alle gegevens in een centrale database ondergebracht en zal het bijhouden daarvan op een consistente manier kunnen plaatsvinden. Het voordeel van één database is dat er slechts op één plaats beschikbaarheid, integriteit en exclusiviteit hoeft te worden gegarandeerd. Een nadeel is echter dat alle services hun gegevens via het netwerk uit de centrale database moeten ophalen.

In architecturen met een process-centric benadering heeft ieder proces de gegevens die voor de uitvoering nodig zijn onder zijn hoede. Het voordeel hiervan is dat de service snelle responsetijden kan garanderen. Het nadeel is echter dat de bewaking van beschikbaarheid, integriteit en exclusiviteit een stuk complexer is.

Barry geeft aan dat in de praktijk een hybride tussenvorm zal worden toegepast, met zowel data-centric als process-centric onderdelen. Hierbij kunnen data meervoudig voorkomen in de services die tezamen de toepassing vormen. Uit beheersmatig oogpunt een nog grotere uitdaging.



Figuur 6: De webservices standaarden piramide

Business Process Management (Orchestration)
De eerder genoemde definitie van SOA van Channabasavaiah geeft aan dat het gaat om het ondersteunen van bedrijfsprocessen. De inzet van services, die samen de bedrijfsprocessen ondersteunen, moet het mogelijk maken om veranderingen in de bedrijfsprocessen eenvoudig te ondersteunen door het vervangen of toevoegen van services. Door gebruik van (web-)services binnen een structuur- of architectuurmodel is dit (in ieder geval in theorie) te realiseren.
Het ondersteunen van bedrijfsprocessen in een servicegeoriënteerde architectuur wordt ook wel aangeduid met Business Process Management of Orchestration. Dit is het samenstellen van de voor de bedrijfsprocessen benodigde functionaliteit met behulp van services in een niethiërarchische structuur.

De webservices piramide en de standaarden
Kaye groepeert de componenten van webservices in een gelaagde piramide (figuur 6) [KAYE03]. De onderste laag bestaat uit (min of meer) gestandaardiseerde onderdelen, die op de markt beschikbaar zijn. Hoe lager in de piramide de componenten zich bevinden, des te meer zijn ze volwassen en als 'standaardpakket' leverbaar. HTTP is inmiddels standaard als communicatieprotocol op ieder systeem beschikbaar. Ook SOAP is voor veel systemen al beschikbaar of maakt zelfs al deel uit van het operating system. Deze componenten worden ook wel 'horizontaal' genoemd omdat ze niet specifiek zijn voor een bepaalde bedrijfstak of toepassing.
In de middelste laag bevinden zich ook 'horizontale' componenten die bedrijfstak of toepassingonafhankelijk zijn, maar nog niet gestandaardiseerd en/of als optie beschikbaar zijn voor systemen. Kaye positioneert in deze laag ook de security, waarmee hij aangeeft dat op dit gebied expliciet inspanning nodig is. De diverse standaarden voor webservices (ook aangeduid met WS-…, waaronder WS-Security) SOAP-DSIG (Digital Signature) en SAML (Security Assertion Markup Language) zijn inmiddels als standaard gepubliceerd door W3C en OASIS.
De bovenste laag is 'vertikaal', omdat hier de bedrijfsspecifieke componenten een plaats hebben gevonden. Het gaat hierbij bijvoorbeeld om het formaat van berichten, transacties, en bedrijfsprocessen.
Op de hoogste laag zien we langzaam standaarden ontstaan op het gebied van procesbeschrijving en uitvoering, zoals BPEL (Business Process Execution Language, oorspronkelijk ontwikkeld door IBM, SAP, BEA, Siebel en Microsoft, nu beheerd door OASIS).

INHERENTE RISICO'S VAN SOA

Bij de toepassing van webservices bij SOA liggen vele inherente risico's op de loer. Binnen de kaders van dit artikel worden de risico's van de onderliggende technologie (de onderste lagen in figuur 6), zoals HTTP(S), TCP/IP, FTP, SMTP, XML, et cetera verondersteld bekend te zijn. Het vervolg gaat meer in op de SOA specifieke risico's.

Transactionele integriteit: ACID
Kaye refereert aan Theo Häerder en Andres Reuter die in 1983 over Principles of Transaction-Oriented Database Recovery hebben gepubliceerd [KAYE03]. In deze publicatie gaan zij in op systemen die meerdere transacties tegelijkertijd kunnen verwerken en daarbij niet gecorrumpeerd mogen worden door een storing in de hardware, het operating system of de databaseprogrammatuur.
In de publicatie van Häerder en Reuter wordt de inmiddels beroemde term ACID geïntroduceerd, die staat voor: Atomicity, Consistency, Isolation en Durability. Hoewel deze term in hun publicaties betrekking heeft op synchrone en kortdurende transacties, gelden de ACID-eisen onverkort voor transacties via webservices.

Webservices
Webservices zijn van nature asynchroon en bevinden zich niet op hetzelfde systeem en binnen het zelfde 'tijdsdomein'. Webservices

zijn veel complexer als we letten op de aspecten tijd en ruimte. Dit betekent dat, hoewel de ACID-eisen blijven bestaan, de realisatie van deze eisen veel gecompliceerder is. Het feit dat transacties veel langer kunnen leven in een loosely coupled omgeving heeft ook nog een ander nadeel. Indien al wordt gestart met het verwerken van de resultaten van services en de langlopende transactie nog niet geheel is afgerond, of wordt geannuleerd, dan is het niet meer mogelijk de transactie terug te draaien. Voor het oplossen van dergelijke problemen moeten zogenaamde compenserende transacties worden gecreëerd.

Loosely coupled

Hoe groter de afstand tussen services, gemeten in beheersing (control) en de relatieve begrippen tijd en afstand, hoe groter de kans dat een transactie niet slaagt. Daar waar in een tightly coupled omgeving de transactie (bijna) altijd slaagt, zal een loosely coupled omgeving noodzakelijkerwijs moeten uitgaan van een pessimistischer veronderstelling. De kans dat een service geen of niet tijdig (positief) positief antwoord oplevert, is in een loosely coupled omgeving aanzienlijk groter dan bij de tight coupling. Dit heeft tot gevolg dat ontwerpers van services er ook van moeten uitgaan dat een verzoek aan een andere service niet het verlangde resultaat oplevert. Door het kiezen van een pessimistisch uitgangspunt, zal in de logica van de service rekening gehouden worden met een niet succesvolle afloop en met compenserende transacties.

Een ander aspect van loose coupling is het gebruik van services vanuit een directory of registry. De zekerheid dat een in een directory gevonden service ook exact doet wat de vragende service wenst, is minder dan bij het gebruik van vaste services waarvoor een SLA is afgesloten. Een aanvragende service kan daarom moeilijk vaststellen of het gewenste resultaat daadwerkelijk wordt opgeleverd.

Syntax en semantiek

Voor veel berichten tussen services is de te gebruike syntax inmiddels in standaarden vastgelegd. Een belangrijke standaard bij het dynamisch bepalen van de te gebruiken webservice is UDDI. Met een op UDDI gebaseerde directory kan een service de benodigde dienstverlenende service opzoeken. In een formele syntax wordt de aangeboden dienst beschreven. De betekenis voor de vragende service is echter afhankelijk van de interpretatie van hetgeen is beschreven (semantiek). Hieruit blijkt dat de beschrijving van de gewenste service in de directory slechts voor één uitleg vatbaar moet zijn. Evenzeer geldt dit voor de interpretatie van de inhoud van SOAP/XML-berichten.

Integriteit van gegevens

In paragraaf Data-centric Architecture versus Process Centric architecture zijn de verschillende mogelijkheden voor de opslag van gegevens beschreven. Door de loose-koppeling van services uit een directory en de gedistribueerde opslag van gegevens is de kans op inconsistente gegevens groot. Dit heeft een negatief effect op de integriteit van de gegevens in zijn totaliteit.

Beveiliging

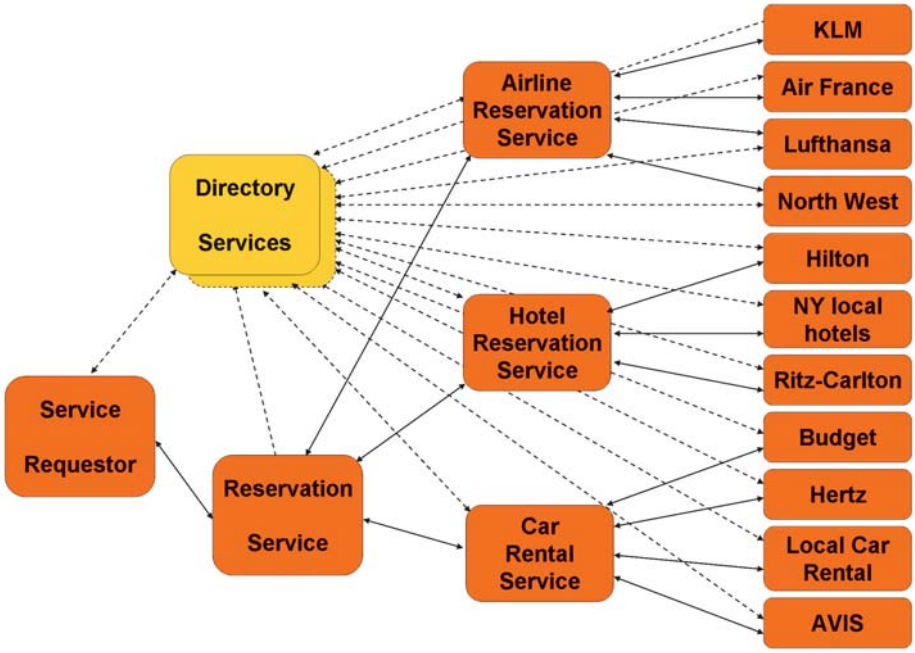
In de traditionele systemen is er sprake van een gecontroleerde omgeving, waarbinnen de IT-middelen en de gegevens kunnen worden beveiligd. In veel gevallen is er sprake van een (fysieke) beveiligingsschil [CHOP04]. Bij webservices (buiten de eigen invloedsfeer) is er geen sprake meer van een centrale beveiligingsschil. SOA is een nieuwe vorm van gedistribueerde verwerking van gegevens. In een dergelijke open omgeving is het lastig om te bepalen of aanroepen van een webservice legitiem is.

In een SOA kunnen de volgende gebieden worden onderkend, waarop eisen aan de beveiliging moeten worden gesteld:

- Berichtenbeveiliging
  Het is essentieel dat de integriteit en vertrouwelijkheid van het bericht gewaarborgd blijft tot aan de uiteindelijke ontvanger.
  Het is tevens van belang dat de authenticiteit van het bericht door de ontvanger kan worden vastgesteld en dat verzending en ontvangst niet kunnen worden ontkend (non-repudiation).
- Vertrouwen tussen services (trust)
  SOA bestaat uit loosely coupled services. Wederzijds vertrouwen is hierbij noodzakelijk. WS-Security omschrijft vertrouwen als volgt: 'Het feit dat een entiteit bereid is te steunen op een andere entiteit voor het uitvoeren van een aantal (trans) acties'. Vertrouwen kan ondermeer worden gebaseerd op elektronische certificaten.
- Identity Management
  Een gebruiker, maar ook een service, heeft een identiteit nodig om toegang te krijgen tot een service. In een SOA omgeving zijn de services per definitie verspreid over het gehele netwerk. Hierbij kan het noodzakelijk zijn dat toegang verkregen wordt tot een groot aantal verschillende (beveiligings)domeinen waar de services zich bevinden. Een consistent beheer van de identiteit, met de bijbehorende rechten, over de gehele SOA is dan van groot belang.

Foutdetectie

Een fout in één van de webservices van de SOA kan worden veroorzaakt door onjuiste functionaliteit, maar evengoed door een fout in de beveiliging. Bij het manifest worden van de fout, zal het resultaat uit de transactie (het 'antwoord') ten behoeve van het bedrijfsproces niet meer betrouwbaar zijn. Een belangrijke vraag die daarbij rijst is: hoe is de detectie van eventuele fouten nog mogelijk? Detectie kan dan alleen nog plaatsvinden door 'om het systeem heen' te controleren. Controle in het totale samenstel van services is niet meer praktisch uitvoerbaar. ◘



Figuur 7: Samengestelde service met een extra niveau

Toename complexiteit

Als we het voorbeeld van de reis naar New York uitbreiden met de ondernemingen die de respectievelijke reserveringsservices raadplegen, dan komt dat er ongeveer uit te zien als in figuur 7. Zeer waarschijnlijk zal er in dit geval gebruikgemaakt worden van een aantal verschillende directories.

Als we een bedrijfsproces ondersteunen met behulp van services, dan zal het plaatje er nog veel complexer uit zien. Als we daarbij verder rekening houden met het feit dat bepaalde services dynamisch, uit een directory, worden geselecteerd op het moment dat deze nodig zijn, gaat de grip op de structuur goeddeels verloren. De service-requestor heeft dus geen zicht meer op de services die in het voorbeeld door de Reservation Service verder nog worden ingezet.

Op basis van de bovenstaande gesignaleerde toename van de complexiteit kunnen een tweetal typen omgevingen worden geïdentificeerd:

- Een SOA waar alle services en directories binnen de eigen directe invloedssfeer liggen;
- Een SOA waar ook services van buiten de eigen invloedssfeer worden gebruikt en waar services dynamisch worden ingezet via directories binnen en buiten de eigen invloedssfeer.

Deze typen omgevingen hebben ieder een eigen risicoprofiel. Het is duidelijk dat het potentiële risico bij het type dat gebruikmaakt van externe services en directories beduidend hoger is dan bij het type dat geheel binnen de eigen invloedssfeer ligt.

DE IT-AUDITOR EN SOA

Voor de IT-auditor, die over deze complexe omgeving een mededeling gaat afgeven, is het essentieel om te kunnen vaststellen welke services op enig moment deel uitmaken of hebben uitgemaakt van zijn object van onderzoek. Om dit te kunnen vaststellen is een uitgebreide audit-trail noodzakelijk met daarin opgenomen de gegevens over iedere service die op enig moment is gebruikt. Indien deze audit-trail voldoende uitputtend is, kan de IT-auditor bijvoorbeeld door middel van het gebruik van audit software een beeld krijgen van de ingezette services en de eigenschappen daarvan.

Afhankelijk van het aantal (externe) services waarvan gebruik is gemaakt kan de IT-auditor meer of minder goed en eenduidig vaststellen welke services zijn gebruikt. Indien niet met voldoende zekerheid kan worden vastgesteld dat de services voldoen aan de daaraan te stellen (of gestelde) kwaliteitscriteria, kan de IT-auditor ook geen positieve mededeling afgeven.

Om het hierboven geschetste probleem te voorkomen kan natuurlijk al voor het gebruik van externe services als criterium worden gesteld dat zij, voor te worden gebruikt, moeten voldoen aan de gestelde kwaliteitscriteria. Hiervoor is een aantal mogelijkheden in (concept) standaarden vastgelegd. Twee belangrijke standaarden worden hierna kort toegelicht.

UDDI Security policy

In versie 3.02 van de UDDI-standaard voor service-registries is een hoofdstuk opgenomen over security. In dit hoofdstuk staat een aantal eisen waaraan een directory moet voldoen. Het beveiligingsmodel van een UDDI-registry bestaat uit een verzameling policies en de implementatie daarvan in de registry. Een aantal van deze policies is:

- Autorisatie
  *A registry **MUST** have a policy on access to the information registered in it.*
- Integriteit van gegevens
  *A registry **MUST** specify how it maintains the information registered in it. The nodes MUST enforce this policy.*
- Vertrouwelijkheid van gegevens
  *A registry **MAY** have a policy for protecting the information under the custody of its nodes from unauthorized access.*
  UDDI also supports XML Digital Signatures on UDDI data to enable inquirers to verify the integrity of the data with respect to the publisher.

Het gebruik van een directory die de optionele policy voor de vertrouwelijkheid van gegevens heeft geïmplementeerd zou de voorkeur moeten

krijgen bij het zoeken naar een webservice.

UDDI Audit Policy
In versie 3.02 van de UDDI-standaard is een mogelijkheid opgenomen voor het vastleggen van de transacties die door de registry zijn afgehandeld. Deze audit-trail moet wel met voldoende waarborgen zijn omgeven en het moet aan de gebruikers bekend worden gemaakt welke gegevens worden vastgelegd. De standaard geeft aan dat deze audit-trail ook gebruikt kan worden als bewijsmateriaal in rechtszaken.

Service Level Agreements
De complexiteit van een SOA kan worden beheerst door de dynamiek te beperken. Een mogelijke beperking is dat er alleen gebruik gemaakt wordt van services waarvoor expliciet met de aanbieder een Service Level Agreement (SLA) is overeengekomen [BIJL05]. Hierdoor wordt zowel de complexiteit als de dynamiek van de SOA beperkt tot die services waarvan vooraf expliciet is vastgesteld dat zij voldoen aan de eisen van functionaliteit en betrouwbaarheid.
In de met de dienstenaanbieder af te sluiten SLA kan ook worden bepaald dat bij de service(s) periodiek een TPM wordt verstrekt door een onafhankelijke IT-auditor.

De dynamiek in de ultieme SOA
Een toepassing op basis van SOA, in de meeste uitgebreide zin van de definitie, bestaat uit dynamisch geselecteerde services. De beslissing om een bepaalde service te gebruiken wordt bepaald door bijvoorbeeld de volgende criteria:

- Is de service passend, dus is de te leveren dienst conform mijn eisen?
- Wat kost de service?
- Is de service gecertificeerd?
- Kan de berichtenstroom worden vercijferd?
- Welk serviceniveau bieden de geselecteerde services?

In deze dynamische omgeving is het de vraag hoe je als auditor kunt vaststellen welke services, sub-services, sub-sub-services, et cetera zijn/worden gebruikt en hoe betrouwbaar deze zijn. Eigenlijk is het antwoord hierop simpel: alleen de direct door eigen services opgeroepen 'externe' service is zichtbaar en kan worden gelogd in de audit-trail. De 'legitimatie' van deze service kan aan de directory worden ontleend, de gegevens van de door deze service gebruikte services en de service zelf blijven echter verborgen. De gebruiker moet dus vertrouwen op de eigenschappen van de eerste externe service die wordt gebruikt.
UDDI heeft hoofdstuk 1.7 van de standaard gewijd aan een introductie van beveiligingsaspecten. UDDI gaat niet verder dan beveiliging op het gebied van autorisatie en authenticatie, en integriteit van de gegevens die bij de gebruiker worden afgeleverd. Het is dus aan de gebruikers van de directory om na te gaan of een resultaat van de zoekopdracht in de directory wordt vertrouwd.
UDDI biedt geen mechanisme om te voorkomen dat kwaadwillenden een dummy service op het netwerk aanbieden, waarmee de mogelijkheid bestaat om gegevens van gebruikers te ontvreemden, te manipuleren of anderszins te misbruiken. Voor services met een hoog risico is het daarom aan te bevelen om de service in eigen beheer te nemen en niet via een openbare directory op te zoeken.

Beveiliging in de standaarden
In de vele standaarden op het gebied van (web)services wordt primair aandacht besteed aan het realiseren van functionaliteit en weinig aan risico's en beveiliging. In het beste geval wordt een paragraaf of hoofdstuk gewijd aan beveiligingsaspecten. Voor beveiliging zijn en worden gescheiden standaarden ontwikkeld. Het gevaar hiervan is dat in eerste instantie naar de functionele standaard wordt gekeken en pas in tweede instantie naar de beveiligingsstandaard. Het risico is dan dat aan de beveiligingsaspecten minder of in het geheel geen aandacht zal worden besteed.

SAMENVATTING

Service Oriented Architecture is een model waarbij toepassingen worden opgebouwd uit services. Het is flexibel en biedt de IT-omgeving de mogelijkheid snel op wijzigingen in de business in te spelen. In het SOA-model kunnen services eenvoudig door andere, met een gewijzigde functionaliteit, worden vervangen.
SOA biedt ook de mogelijkheid om services dynamisch te koppelen tot toepassingen door de benodigde service in een (al dan niet openbare) directory op te zoeken. Deze dynamiek brengt risico's met zich mee. Naast de reeds aanwezige risico's van het gebruik van internet komt er nog een aantal risico's bij door het gebruik van services in bedrijfsbrede toepassingen.
In dit artikel is bewust niet ingegaan op de economische aspecten van SOA, bijvoorbeeld het spreiden van kosten over meer gebruikers.

CONCLUSIE

Indien de onderzochte omgeving aan een aantal hieronder genoemde randvoorwaarden voldoet, kan de IT-auditor door onderzoek een deugdelijke grondslag voor een (niet noodzakelijkerwijs positief) oordeel verkrijgen. Hiervoor is het echter wel noodzakelijk dat het grote aantal onzekerheden binnen een SOA met voldoende waarborgen wordt omgeven.
Het belangrijkste aspect is dat de risico's van het gebruik van services in kaart zijn gebracht en dat er, afhankelijk van het risiconiveau, voldoende maatregelen zijn getroffen. Als voorbeelden van dergelijke maatregelen kan worden gedacht aan de volgende randvoorwaarden bij de inzet van services:

- Alle services waarvan gebruik gemaakt wordt, moeten minimaal zijn gecertificeerd op de gebieden: ▶

- beveiliging;
- functionaliteit en functioneren;
- beschikbaarheid.

- Alle gebruikte services, zowel intern als extern, moeten in een audit-trail worden vastgelegd.
- Met leveranciers van services moet een Service Level Agreement worden afgesloten, voordat van services gebruik gemaakt wordt.
- Bij services op gebieden met een hoog risico moet worden besloten om de vrijheden bij de dynamische keuze van services te beperken.
- Bij inzet van services moet voldoende aandacht zijn besteed aan de beveiliging, door bij de implementatie nadrukkelijk uit te gaan van beveiligingsstandaarden.

Indien voldoende waarborgen aanwezig zijn, wordt het voor een IT-auditor mogelijk om een mededeling met een positieve strekking af te geven. Voor een dergelijke mededeling is het dan wel noodzakelijk dat aan alle randvoorwaarden wordt voldaan en dat alle waarborgen juist functioneren.

HANDREIKING AAN DE AUDITOR

Deze handreiking vormt zeker geen 'werkprogramma' maar een indicatie van de gebieden waar de auditor rekening mee moet houden.

1. Risico analyse
   Heeft een expliciete analyse plaatsgevonden van de aan de SOA verbonden risico's?
2. Compenserende maatregelen
   Zijn op basis van de risicoanalyse compenserende maatregelen getroffen?
3. Standaarden
   Is er bij de ontwikkeling van de SOA voldoende gebruik gemaakt van standaarden?
4. Dynamiek en complexiteit
   Zijn er criteria gedefinieerd waaraan dynamisch te selecteren services moeten voldoen?
   Zijn er voldoende maatregelen getroffen om vast te kunnen stellen welke services op enig moment zijn gebruikt (audit trail) en of deze services op de juiste wijze hebben gefunctioneerd (bijvoorbeeld door gebruik van certificaten en TPM's)
5. Beheersbaarheid
   Worden de bij punt 4 genoemde criteria voor dynamisch te selecteren services daadwerkelijk gehanteerd? Zijn er criteria gedefinieerd om vast te stellen welke (externe) services van een zodanig belang zijn dat daarbij een mededeling (TPM) van een auditor noodzakelijk is? Zijn er afwegingen gemaakt om bepaalde services mogelijk in het geheel niet uit te besteden?
6. Beveiliging
   Is er bij het ontwerp en het onderhoud van de SOA voldoende aandacht voor de beveiliging, door expliciet beveiligingsstandaarden te hanteren?

Is het antwoord op één van de bovenstaande vragen 'NEE', dan zal de auditor heel zorgvuldig na moeten gaan of hij/zij nog tot een deugdelijke grondslag voor een positieve mededeling kan komen.
Indien alle vragen met 'JA' kunnen worden beantwoord, dan betekent dit nog niet automatisch dat 'het wel goed zit'. De auditor zal zich ook in dit geval een beeld moeten vormen van de afdoende werking van de getroffen maatregelen. Het toepassen van een standaard alleen is nog geen garantie voor een juiste werking. Hier past de auditor (zoals gebruikelijk) een kritische grondhouding. ■

Ing. J.M. (Koos) Ziere RE CISA is als senior IT-auditor werkzaam bij Berk N.V., IT Audit & Risk Management, en heeft meer dan dertig jaar ervaring met geautomatiseerde gegevensverwerking. Dit artikel is gebaseerd op zijn afstudeerreferaat aan de Erasmus Universiteit waar hij als IT-auditor afstudeerde en is geschreven op persoonlijke titel.

Literatuur

[BARR03], Douglas K. Barry; Web Services and Service-Oriented Architectures: The savvy manager's guide; Morgan Kaufmann Publishers 2003
[BROW01], Kyle Brown e.a.; Web Services Architectures and best practices, IBM Software Services for WebSphere 2001
[BIJL05], MICROSOFT; SOA: Van business-strategie tot technologie, artikelenserie, Microsoft 2005
[CHAN03-1], Kishore Channabasavaiah e.a.; Migrating to a service-oriented architecture, Part 1, IBM Global Services, 2003
[CHAN03-2], Kishore Channabasavaiah e.a., Migrating to a service-oriented architecture, Part 2, IBM Global Services, 2003
[CHOP04], Dipak Chopra, Security for SOA and Webservices
[ERL04], Thomas Erl; Service-Oriented Architecture: A field guide to integrating XML and Web Services, Prentice Hall 2004
[GUPT03], Samudra Gupta; Service Oriented Architecture – Part 1&2, javaboutique.internet.com 2003
[HASH03], Sayed Hashimi; Service Oriented Architecture explained, ondotnet.com 2003
[HE01], Hao He; What is Service Oriented Architecture?, XML.COM 2001
[IETF02-1], Internet engineering taskforce, RFC 3275, XML-Signature Syntax and Processing, IETF 2002
[KAYE03], Doug Kaye; Loosely Coupled: The missing pieces of Web Services, RDS Press 2003
[LOOH04], Kim Loohuis, Acht artikelen in Computable met betrekking tot SOA, COMPUTABLE 2004
[OASI05-1], Organization for the Advancement of Structured Information Standards; UDDI, Executive overview: Enabling Service Oriented Architecture, OASIS 2005
[OASI05-2], Organization for the Advancement of Structured Information Standards; Introduction to UDDI, Important Features and Functional Concepts, OASIS 2005
[OASI05-3], Organization for the Advancement of Structured Information Standards; Universal Description, Discovery and Integration v3.0.2 (UDDI) specification, OASIS 2005
[OASI05-4], Organization for the Advancement of Structured Information Standards; Security Assertion Markup Language (SAML) 2.0 Technical Overview, OASIS 2005
[OASI04-1], Organization for the Advancement of Structured Information Standards; Web Services Security: SOAP Message Security 1.0 (WS-Security 2004), OASIS 2004
[SUN04], Sun Microsystems; Assessing Your SOA Readiness, A Technical White Paper, SUN 2004
[W3C03], WWW Consortium; Simple Object Access Protocol, W3C 2003