Het wat, hoe en waarom van applicatieservers

Strategisch gebruik van applicatieservers in multi-product frontend toepassingen bij (bank)verzekeraars

Een veranderende wereld …

De wereld is sterk aan verandering onderhevig. De algemene trend van de laatste jaren is dat alles groter, sneller en beter moet. Ook in het bedrijfsleven is deze trend duidelijk zichtbaar. Dit heeft direct invloed op de eisen die door het bedrijfsleven aan de informatisering gesteld worden. Nu geeft de genoemde trend *groter, sneller en beter* ons uiteraard weinig houvast voor het bepalen van deze eisen. Binnen het artikel worden een achttal concreet waarneembare trends benoemd, en op basis van deze trends worden de nieuwe eisen voor de informatisering bepaald (het wat?). Vervolgens zal blijken dat de architectuur van de huidige systemen niet voldoet aan de nieuwe eisen. Tevens zal aangetoond worden dat het gebruik van applicatieservers *wel* bijdraagt aan het invullen van deze nieuwe eisen (het waarom?). Met het inzetten van applicatieservers kan worden overgestapt op een nieuw architectuur-model, waarbij een deel van de huidige systemen hergebruikt kan worden en waarbij tevens wordt voldaan aan de nieuwe eisen. Van dit architectuurmodel worden binnen dit artikel een tweetal te onderkennen stromingen beschreven: Microsoft en de NOIS-alliantie (het hoe?). Het artikel wordt afgesloten met de beschrijving van een intern project dat momenteel binnen de Cap Gemini divisie Insurance & Social Security wordt uitgevoerd. Dit project – getiteld *multi-product frontend* – heeft als doel om een standaard oplossing te bieden voor de elektronische communicatie tussen het intermediair en de verzekeringsmaatschappijen. Binnen het project spelen de eisen zoals ze in dit artikel benoemd zijn een grote rol, ook de inzet van applicatieservers komt derhalve ruimschoots aan bod.

Trends binnen bedrijfsleven

Als gekeken wordt naar het bedrijfsleven kunnen een aantal duidelijke trends onderscheiden worden.

- *Globalisatie* – de afzetmarkt van het bedrijfsleven wordt steeds groter en laat zich steeds minder beperken door regio- en landgrenzen. Een goed voorbeeld hiervan is de Europese eenwording waardoor bedrijven in Nederland de mogelijkheid krijgen om hun producten eenvoudiger in de andere Europese landen af te zetten. Hierdoor wordt globalisatie van de afzetmarkt steeds meer gemeengoed en blijft deze niet meer beperkt tot de grote multinational bedrijven.
- *Distributiekanalen* – waarbij vroeger elk product of dienst één of meerdere standaard distributiekanalen kende, worden tegenwoordig steeds meer en fundamenteel verschillende distributiekanalen gebruikt. Hierbij kan gedacht worden aan de recente mededeling van Ahold dat ze bank- en verzekeringsproducten in de supermarkten gaan aanbieden.
- *Investeringen in Y2K en Euro* – het bedrijfsleven heeft de afgelopen jaren massaal geïnvesteerd in het aanpassen van hun huidige informatiesystemen. Gezien de investeringen die gedaan zijn voor het 2000- en Euro-proof maken van de systemen, zullen deze de komende jaren nog niet vervangen worden.
- *Time-to-market* – in de huidige sterk concurrerende markt is het essentieel om nieuwe producten snel op de markt te kunnen zetten. Tevens neemt de levenscyclus van producten af waardoor bedrijven steeds snellere en vaker nieuwe producten en diensten moeten lanceren.
- *Technology push* – de huidige stand van de techniek biedt het bedrijfsleven tal van nieuwe mogelijkheden. De opkomst van Internet is wellicht één van de meest sprekende voorbeelden van technology push van de afgelopen jaren waardoor een compleet nieuw medium beschikbaar is voor de ondersteuning van de diverse disitributiekanalen. Een meer recent voorbeeld is ubiquitous computing, een ontwikkeling die zich de komende jaren moet gaan bewijzen.
- *Samenwerkingsverbanden* – binnen het bedrijfsleven wordt steeds meer business-to-business samenwerkingsverbanden aangegaan om nieuwe producten en nieuwe distributiekanalen te kunnen exploiteren. Daarnaast kan kostenbesparing gerealiseerd worden door middel van het integreren van de bedrijfsprocessen (supply chain integration) van de samenwerkende bedrijven.
- *24-uurs economie* – de huidige maatschappij verschuift steeds meer richting een 24-uurs economie. Hierdoor is de beschikbaarheid van de dienstverlening de afgelopen jaren drastisch toegenomen. Simpele voorbeelden hiervan zijn openingstijden van winkels en supermarkten, een ander wellicht meer sprekend voorbeeld is het dag en nacht kunnen afsluiten van verzekeringen middels het Internet.

- *Fusies & Overnames* – met name in het bank- en verzekeringswezen zijn fusies en overnames de laatste jaren aan de orde van dag. Het resultaat hiervan zijn steeds grotere, maar ook complexere organisaties.

Voor het bedrijfsleven is het van groot belang om zich bovenstaande trends te realiseren en hiermee rekening te houden bij de inrichting van hun informatisering.

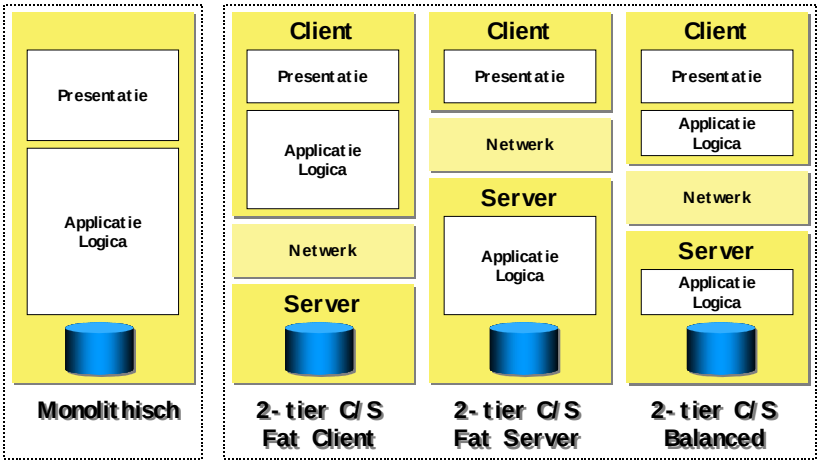
Nieuwe eisen voor de informatisering

Kijkend naar bovenstaande trends en de invloed hiervan op de informatisering kunnen de volgende eisen onderkend worden:

- *Schaalbaar* – door trends als globalisatie, samenwerkingsverbanden en fusies & overnames wordt het steeds moeilijker om het aantal potentiële gebruikers van applicaties in te schatten. Hierdoor is het noodzakelijk om schaalbare applicaties te ontwikkelen zodat de performance niet negatief beïnvloed wordt bij een toename van het aantal gebruikers. Schaalbare applicaties stellen bedrijven in staat om vandaag enkele honderden en morgen enkele duizenden gebruikers te ondersteunen.
- *Hergebruik* – door investeringen in Y2K en Euro eisen bedrijven dat nieuwe applicaties kunnen aansluiten op bestaande systemen. Daarnaast is het vanuit het oogpunt van time-to-market van belang om bij de bouw van nieuwe applicaties rekening te houden met toekomstig hergebruik.
- *Flexibiliteit* – nieuwe applicaties dienen dusdanig flexibel te zijn dat ingespeeld kan worden op toekomstige ontwikkelingen binnen het bedrijf, denk aan nieuwe distributiekanalen, samenwerkingsverbanden en fusies & overnames, alsmede het beschikbaar komen van nieuwe technologieën op termijn.
- *Beschikbaarheid* – door de opkomst van de 24-uurs economie en het ondersteunen van een globale markt via diverse distributiekanalen is het noodzakelijk dat applicaties ten minste 24x7 uur per week beschikbaar zijn.
- *Openheid* – door de samenwerkingsverbanden en fusies & overnames is het noodzakelijk dat systemen van diverse bedrijven geïntegreerd kunnen worden. Deze integratie kan vergemakkelijkt worden door het gebruik van standaarden en open technologieën. Tevens ondersteunt deze openheid zowel de flexibiliteit en als het hergebruik van applicaties.
- *Beveiliging* – door het aangaan van samenwerkingsverbanden, fusies & overnames en het gebruik van Internet vervagen de fysieke grenzen van bedrijven. Doordat bedrijfsnetwerken zowel onderling als aan het Internet gekoppeld worden, is het noodzakelijk om het aspect beveiliging bij de ontwikkeling van nieuwe applicaties te betrekken.

Alvorens ingegaan wordt op de wijze waarop deze nieuwe eisen aan de informatisering ondersteund kunnen worden, door de inzet van applicatieservers, wordt allereerst ingegaan op de architectuur van de huidige applicaties. Hierbij wordt aangetoond dat deze niet aan deze eisen kan voldoen. De huidige applicaties zijn ontwikkeld volgens een tweetal architectuurmodellen (zie Figuur 1):

- monolithische model – in het monolithische model zijn presentatie, applicatielogica en gegevens sterk afhankelijk van elkaar geïmplementeerd en bevinden zich op één fysieke machine. Bekende voorbeelden van monolithische applicaties zijn de Cobol/CICS applicaties op een OS/390 main-frame of Linc/DMSII applicaties op een Unisys mainframe.



Figuur 1 Architectuurmodellen van huidige applicaties

- 2-tier client/server model – in het 2-tier client/server model is een tweetal onderdelen losgekoppeld, namelijk de client en de server. Van dit model is een drietal varianten te onderkennen op basis van de plaatsing van presentatie, applicatielogica en gegevens:
  - fat client – in de *fat client* variant zijn presentatie en applicatielogica in de client en bevinden de gegevens zich op de (database)server. Een voorbeeld van dit model is een applicatie ontwikkeld met Oracle Designer/Developer waarbij de presentatie (Oracle Forms) en applicatielogica (PL/SQL) op de client draaien en de gegevens zich op de databaseserver (Oracle database) bevinden. De client en de server communiceren middels PL/SQL.
  - fat server – in de *fat server* variant bevinden zich de applicatielogica en de gegevens in de server en de presentatie op de client.
  - balanced – in de balanced 2-tier client/server variant bevindt een deel van de applicatielogica zich op de client terwijl een ander deel zich op de server bevindt.

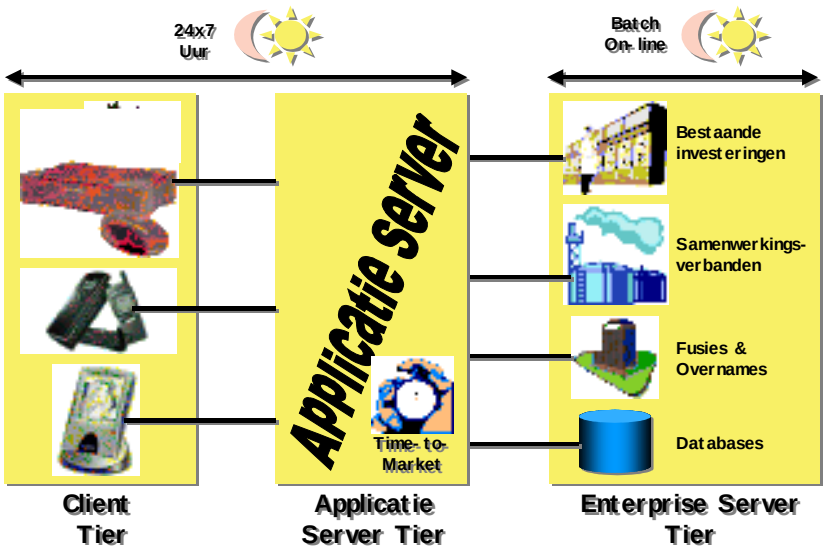
Het grote nadeel van bovengenoemde modellen is de sterke koppeling van presentatie, applicatielogica en gegevens waardoor met name de *flexibiliteit* voor de huidige tijd niet meer voldoet. Daarnaast zijn de gebruikte implementatietechnologieën te gesloten, is er geen ontkoppeling tussen batch en on-line systemen, is er geen rekening gehouden met internettechnologie en zijn er onvoldoende handvatten voor hergebruik. Verder geldt specifiek voor de 2-tier client/server varianten dat invulling van voldoende schaalbaarheid en hoge beschikbaarheid een groot probleem zijn.

Multi-tier Client/Server Architectuur

Op basis van bovenstaande tekortkomingen van de architectuur van de huidige applicaties en de implementatietechnologieën vindt er een duidelijke verschuiving plaats naar multi-tier client/server applicaties. Terwijl monolithische applicaties fysiek uit 1 laag bestaan, en 2-tier client/server applicaties uit twee, wordt bij deze technologische visie gedefinieerd dat applicaties verdeeld zullen worden in tenminste een drietal fysieke lagen (zie Figuur 2):

- *Client*
- *Applicatieservers*
- *Enterpriseservers*

De Client-tier bevat de presentatie van de applicatie in de *client logic*. Deze presentatie kan gerealiseerd worden middels een traditionele programmeertaal en derhalve getoond worden in een traditionele client, of middels internettechnologie en derhalve getoond worden in bijvoorbeeld een webbrowser, een Network Computer, webTV, een Personal Digital Assistants, etc.



Figuur 2 Multi-tier Client/Server Architectuur

De enterpriseserver-tier bevat enerzijds de gegevens van de applicatie en anderzijds de bestaande (legacy) systemen en de systemen van samenwerkingsverbanden, fusie- en overnamepartijen. De applicatieserver-tier vormt de brug tussen de client-tier en de enterpriseserver-tier door het herbergen van de applicatielogica. Deze applicatielogica werkt zowel ontsluitend naar de enterpriseserver-tier als faciliterend aan de client-tier.

Door gebruik te maken van bovenstaand architectuurmodel wordt invulling gegeven aan een aantal zaken die in de huidige wereld van belang zijn. Zo wordt *globalisatie* afgedekt door de mogelijkheid om op wereldwijde schaal zowel klanten (via diverse *distributiekanalen*) als *samenwerkingsverbanden* (supply chain integration) en *fusies & overnames* te ontsluiten. Voor deze ontsluiting kan gebruik gemaakt worden van o.a. *Internettechnologie*. Tevens kan hergebruik gepleegd worden van *bestaande systemen*. Tenslotte is er een duidelijke ontkoppeling aangebracht tussen batch en on-line applicaties waarbij de batch applicaties zich in de Enterprise Server-tier zullen bevinden. Hierdoor kan aan de voorkant (de client- en applicatieserver-tier) *24x7 uur ondersteuning* geboden worden in de frontoffice terwijl de backoffice middels bijvoorbeeld batch bepaalde zaken later afhandelt.

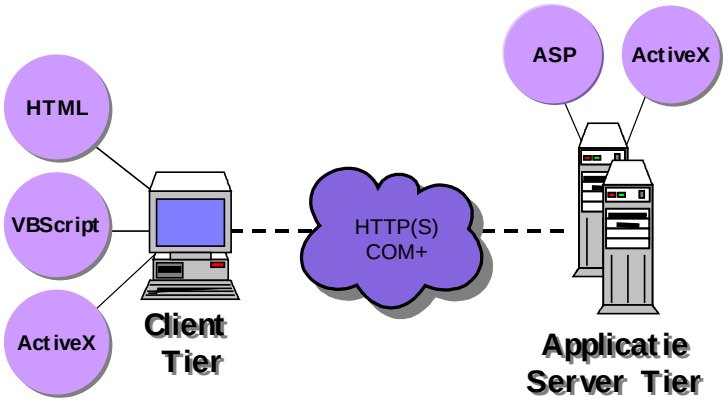
Als gekeken wordt naar de momenteel beschikbare applicatieservers kan een duidelijke tweedeling waargenomen worden: Microsoft en NOIS.

Microsoft

In de realisatie van het multi-tier client/server model door Microsoft[1] staat Windows 2000[2], die als applicatieserver te gebruiken is, centraal. Als ingezoomd wordt op de belangrijke onderdelen die Windows 2000 tot onder andere een applicatieserver maken, worden de volgende onderdelen zichtbaar:

- Internet Information Server – betreft de webserver. Deze is verantwoordelijk voor het versturen van HTML en eventueel VBScript en ActiveX componenten naar de client-tier (webbrowser). Daarnaast vormt het een executieplatform voor ASP's (Active Server Pages). Deze bieden de mogelijkheid om op basis van eenvoudige server-side scripting dynamische internettoepassingen te ontwikkelen.
- COM+ – betreft Microsoft's Object Transaction Monitor implementatie in Windows 2000. De applicatielogica ontwikkeld als ActiveX componenten draait bovenop COM+. Op deze manier kunnen ze of door de client-tier of door ASP's op de webserver aangeroepen worden.

Voor de realisatie van clients zijn een aantal technologieën te gebruiken: HTML (Hypertext Markup Language) als opmaaktaal voor voornamelijk statische webpagina's, VBScript voor de implementatie van dynamische aspecten in een webpagina en ActiveX. Daar waar voornamelijk HTML en VBScript gebruikt wordt voor het ontwikkelen van Internet clients, wordt ActiveX voornamelijk gebruikt voor het ontwikkelen van traditionele clients.



Figuur 3 Microsoft's applicatieservermodel

Ter verduidelijking, bij een Internet client vindt de volgende communicatie plaats: de webbrowser vraagt aan de webserver een bepaalde webpagina op, de webpagina wordt verstuurd naar de client alwaar de webbrowser de in HTML geschreven webpagina interpreteert inclusief het eventueel aanwezige VBScript. De webpagina kan of reeds kant en klaar beschikbaar zijn op de webserver, of

[1] Microsoft noemt zijn multi-tier client/server architectuur strategie: Windows DNA (Distributed interNet Architecture).

[2] Windows NT 4.0 is eveneens als applicatieserver te gebruiken middels het installeren van een aantal Service Packs, het zogenaamde "Option Pack" en de laatste versies van de BackOffice producten.

gegenereerd worden door een ASP. Deze ASP gebruikt hiervoor bijvoorbeeld een ODBC koppeling met een relationele database of een aanroep van een ActiveX component via COM+.

In het geval van een traditionele (lees: niet Internet client) zal een op de client-tier aanwezige ActiveX component rechtstreeks communiceren met een ActiveX component op de applicatieserver via COM+.

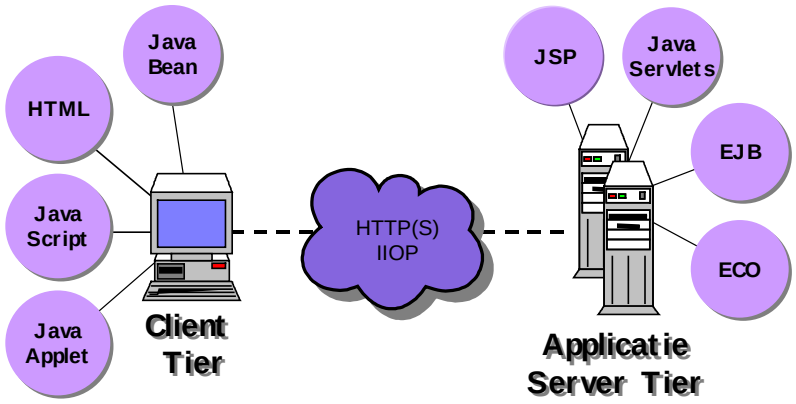
Voor de koppeling met systemen uit de Enterprise Server-tier levert Microsoft een aantal adapters:

- COM/TI – voor synchrone koppeling met CICS-mainframe applicaties eventueel in combinatie met SNA Server voor de koppeling van Windows NT (TCP/IP) met het mainframe (SNA).
- ODBC – voor de koppeling met diverse databases.
- MSMQ – voor asynchrone koppeling met applicaties. Momenteel worden er een groot aantal MSMQ Clients voor diverse platformen geleverd door Level 8, o.a. SUN Solaris, HP-UX, SCO-Unix, Linux, VMS, OS/400 en MVS.

NOIS

In de NOIS alliantie, bestaande uit Netscape, Oracle, IBM en Sun heeft zich als aanval op Microsoft een standaardisatie gemanifesteerd. Het voordeel van deze standaardisatie betreft de mogelijkheid om applicatieservers van verschillende partijen onderling te koppelen of applicatielogica van de ene applicatieserver over te zetten naar de andere. Centraal in de aanpak van NOIS staan:

- Webserver – de webserver is verantwoordelijk voor het versturen van HTML, Java Script en Java Applets naar de webbrowser. Daarnaast vormt het een executieplatform voor JSP en Java Servlets. JSP (Java Server Pages) zijn de NOIS equivalent van ASP (Active Server Pages) van Microsoft om op basis van server-side scripting dynamische internettoepassingen te ontwikkelen. Java Servlets zijn stukjes applicatie geschreven in Java.
- Object Transaction Monitor - de Object Transaction Monitor is een belangrijk onderdeel van de Application Server. Hier draait de applicatielogica ontwikkeld als EJB (Enterprise Java Bean) of ECO (Enterprise Corba Object) in zogenaamde containers. Een OTM bevat o.a. een Corba[3] compliant ORB voor de communicatie met de applicatie logica middels IIOP[4], een JVM voor de interpretatie van in de applicatielogica geschreven in Java en een aantal standaard services, o.a. *Naming service, Security service, Event service, Identity service, Life cycle service, Externalization service, Object Transaction service, Concurrency service* en *Query service.*



Figuur 4 NOIS's applicatieservermodel

[3] Corba staat voor *Common Object Request Broker Architecture* (zie figuur 4). Deze objectbus middleware standaard is ontwikkeld door de *Object Management Group* (OMG). De OMG is in mei 1989 opgericht door acht bedrijven: 3Com, American Airlines, Canon, Data General, Hewlett-Packard, Philips Telecommunications, Sun Microsystems en Unisys. Sinds oktober 1989 opereert de OMG als onafhankelijke non-profit instelling.

[4] Internet Inter ORB Procotol (IIOP) betreft OMG's standaard protocol voor de communicatie tussen verschillende objectbussen

Voor de realisatie van clients is een aantal technologieën te gebruiken: HTML (Hypertext Markup Language) als opmaaktaal voor voornamelijk statische webpagina's, Java Script en Java Applets voor de implementatie van dynamische aspecten in een webpagina en Java Beans.

Java Applets zijn kleine stukjes applicatie geschreven in Java die in een webbrowser uitgevoerd kunnen worden. Om met behulp van hergebruik van Java Applets applicaties te bouwen, het zogenaamde *Component Based Development*, is een standaard ontwikkeld: *Java Beans*. Deze standaard geeft aan hoe herbruikbare componenten in Java gebouwd moeten worden, hoe de eigenschappen van een component aan een ontwikkelomgeving beschikbaar gesteld moeten worden en hoe de communicatie tussen componenten onderling plaatsvindt. Een Java Applet dat aan de eisen van de Java Beans standaard voldoet wordt een Java Bean genoemd.

Ter verduidelijking, bij een Internet client vindt de volgende communicatie plaats: de webbrowser vraagt aan de webserver een bepaalde webpagina op, de webpagina wordt verstuurd naar de client alwaar de webbrowser de in HTML geschreven webpagina interpreteert inclusief de eventueel aanwezige Java Script en Java Appplets. De webpagina kan of reeds kant en klaar beschikbaar zijn op de webserver, of gegenereerd worden door een JSP. Deze JSP gebruikt hiervoor bijvoorbeeld een aanroep van een Java Servlet, EJB of ECO via IIOP.

In het geval van een traditionele (lees: niet Internet client) zal een op de client-tier aanwezige applicatie, geschreven in bijvoorbeeld Java of C++, rechtstreeks communiceren met een Java Servlet, EJB of ECO component op de applicatie server via IIOP.

Voor de koppeling met systemen uit de Enterprise Server-tier leveren de diverse leveranciers applicatieserver afhankelijk oplossingen. Zo levert IBM bijvoorbeeld in hun WebSphere applicatieserver mogelijkheden om CICS en IMS te koppelen middels TXSeries [1] en diverse andere applicaties middels MQSeries. Oracle levert onder andere de mogelijkheid om applicaties via de Oracle Application Server te koppelen middels Tuxedo.

Multi-product frontend

De Cap Gemini divisie Insurance & Social Security heeft over de afgelopen jaren bij een aantal (bank)verzekeraars in Nederland projecten uitgevoerd met als overeenkomstig strategisch doel de communicatie tussen de (bank)verzekeraar en het intermediair te verbeteren. Bij de realisatie van dit strategische doel speelde het gebruik van applicatieservermodellen een belangrijke rol. Dit heeft de Cap Gemini divisie Insurance & Social Security doen besluiten tot het definiëren van het project *multi-product frontend*. De doelstelling van dit project is om een generieke oplossing te ontwerpen voor de communicatie tussen het intermediair en de (bank)verzekeraar. Deze generieke oplossing dient enerzijds te voorzien in een flexibele frontend, waardoor het lanceren van nieuwe verzekeringsproducten versneld kan worden. Anderzijds dient de multi-product frontend toepasbaar te zijn voor *plug & play* koppelingen met de bestaande backoffice en intermediair systemen.

Een aantal uitgangspunten van het multi-product frontend project zijn afgeleid van de binnen dit artikel beschreven trends:

- *Globalisatie* – door de toepassing van internettechnologie wordt de mogelijkheid gecreëerd om (op termijn) op eenvoudige wijze ook intermediairs in het buitenland te bedienen. Hierdoor kan de (bank)verzekeraar zijn afzetmarkt globaliseren.
- *Distributiekanalen* – door de opzet van het multi-product frontend wordt het mogelijk om naast het afzetten van producten via het intermediairkanaal ook directe verkoop (direct writing) toe te passen.
- *Investeringen in Y2K en Euro* – het ontsluiten van de bestaande backoffice functionaliteit. Hierdoor kan bestaande productgerichte functionaliteit, zoals polisadministratiesystemen voor leven-, zorg en schadeproducten, door het multi-product frontend worden hergebruikt.
- *Time-to-market* – door het flexibel ontwerpen en bouwen van de multi-product frontend kan het toevoegen van nieuwe (bank)verzekeringsproducten op een eenvoudig en snelle manier worden uitgevoerd. Hiermee wordt de time-to-market van nieuwe verzekeringsproducten versneld.
- *Technology push* – het gebruik van Internet als compleet nieuw medium voor het ondersteunen van het intermediair.
- *Samenwerkingsverbanden* – de multi-product frontend voorziet in het integreren van de bedrijfsprocessen (supply chain integration) van het intermediair en de (bank)verzekeraar. Dit

levert ondermeer een tijd- en kostenbesparing op, zodat een betere concurrentiepositie ten opzichte van de direct writers verkregen wordt.
- *24-uurs economie* – het 24x7 uur kunnen aanbieden van offerte- en polisaanvraagfunctionaliteit aan de intermediair middels het multi-product frontend terwijl de afhandeling van de polisaanvraag tijdens werktijden wordt uitgevoerd door de backoffice.
- *Fusies & Overnames* – het multi-product frontend voorziet in openheid en flexibiliteit, waardoor toekomstige integratie met andere systemen relatief eenvoudig is uit te voeren.

De architectuur en het ontwerp van de multi-product frontend zijn technologie-onafhankelijk opgesteld, waarbij gebruik is gemaakt van Cap Gemini's *Integrated Architecture Framework* methodiek. Deze architectuur en het ontwerp zijn gebaseerd op de hier beschreven multi-tier client/server architectuur. Daarentegen wordt het detailontwerp en de realisatie van de multi-product frontend binnen één technologische omgeving uitgevoerd. Deze uitvoering vindt plaats op basis van de eerder beschreven applicatieservermodellen van Microsoft en de NOIS-alliantie, waarbij het Microsoft applicatieservermodel (Windows DNA) als eerste wordt gehanteerd.

Auteurs

Drs. P.J. Koning adviseert in zijn rol van Technology Consultant bij Cap Gemini klanten uit diverse branches in het gebruik van state-of-the-art technologie voor het ontwikkelen van informatiesystemen. Centraal in zijn aanpak staat het gebruik van flexibele applicatie-architecturen, ontwikkelomgevingen internettechnologie, middleware en legacy wrapping.

Ir. F. Groen is werkzaam als projectleider bij Warp11, de IAD/RAD-unit van Cap Gemini Insurance & Social Security. Hij is vanuit een team/projectleider rol betrokken geweest bij een grootschallig multi-product frontend project, wat bij een grote verzekeraar is uitgevoerd.

Literatuur

[1] Component Broker – modern en bedrijfskritisch hand in hand, drs. P.J. Koning, maart '99, Software Release Magazine.