Object bus en objecten zijn de toekomst van het Internet

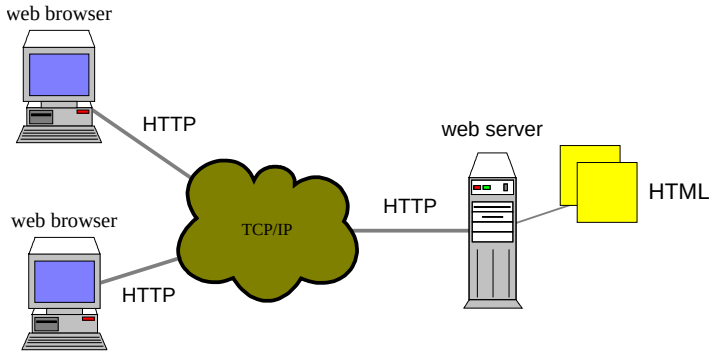
Tot voor kort kende het gebruik van internettechnologie een aantal problemen met het beschikbaar stellen van zakelijke toepassingen. De simpele gebruikersinterfaces, de stateless interactie met de eindgebruiker, de geringe schaalbaarheid, het ontbreken van een mechanisme voor robuuste transacties en de lastige integratie met legacy systemen.

Het zijn problemen die al een geruime tijd bestaan en hierdoor het breed uitrollen van zakelijke toepassingen op het Internet belemmerde. Tot voor kort was er geen concrete oplossing voor. Tot voor kort, want, met het volwassen worden van state-of-the-art technologieën zoals Corba, DCom (object bus technologie), Java en ActiveX (objecttechnologie) is het nu mogelijk om zakelijke toepassingen via het Internet beschikbaar te stellen.

Het Internet

In de jaren vijftig hield de RAND Corporation zich bezig met het vraagstuk hoe de Amerikaanse overheid kon blijven communiceren na een nucleaire aanval [1]. De oplossing, bedacht door Paul Baran van RAND, werd in 1964 openbaar gemaakt en voorzag in een netwerk zonder centrale autoriteit. Het netwerk zou moeten bestaan uit autonome knooppunten die volgens een standaard protocol op gelijke voet met elkaar zouden communiceren. Zo ontstond het Internet, het wereldwijde netwerk van computernetwerken. Communicatie op Internet geschiedde via het tcp/ip-protocol (*transmission control protocol/internet protocol*).

De eerste generatie toepassingen die op het Internet ontstonden, betrof het opvragen en transporteren van statische informatie. Voor de communicatie tussen de web browser en de web server werd het *hypertext transfer protocol* (http) ontwikkeld (zie figuur 1). Dit protocol draait bovenop het tcp/ip-protocol. De codering van de statische informatie gebeurde met behulp van de *hypertext markup language* (html).



Figuur 1 De eerste generatie: http en html.

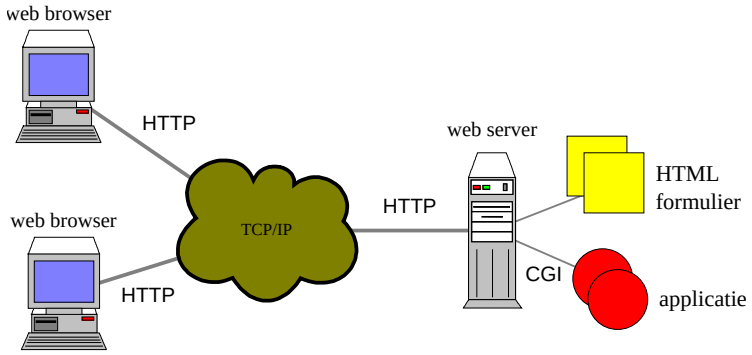
Het grote voordeel dat het gebruik van deze technologie bood, was dat het mogelijk werd om toepassingen eenmalig te ontwikkelen en deze vervolgens middels automatische distributie op verschillende platformen te laten draaien. Doordat de toepassing zich op de web server bevond, was een bijkomend voordeel dat het onderhoud van deze toepassing centraal plaats kon vinden.

Naast de voordelen van platformonafhankelijkheid, automatische distributie en centraal onderhoud kende het ontwikkelen van eerste generatie internettoepassingen ook een aantal nadelen. Zo was het niet mogelijk om dynamische toepassingen te ontwikkelen, die op basis van invoer van een eindgebruiker bepaalde uitvoer genereren. Een voorbeeld van een dynamische internettoepassing is de

zoekmachine Alta Vista die op basis van een zoekstring de uitvoer, het resultaat van de zoekstring, genereert. Tevens was interactie met de eindgebruiker niet mogelijk doordat de html-standaard geen elementen bevatte om dit te realiseren. Een ander groot nadeel was dat het ontbreken van de mogelijkheid om internettoepassingen te koppelen aan traditionele omgevingen, zoals bijvoorbeeld een relationele database. Hierdoor was men genoodzaakt om bepaalde gegevens op meerdere plaatsen te bewaren.

Op basis van de hierboven genoemde nadelen vonden er twee ontwikkelingen plaats. Allereerst werd de html-standaard uitgebreid met formulieren (*forms*). Hierdoor was het mogelijk om eenvoudige interactie met de gebruiker middels invulvelden en knoppen te realiseren. Tevens werd *het common gateway interface protocol* (cgi) ontwikkeld. Met dit protocol was het mogelijk om internettoepassingen te koppelen aan een traditionele omgeving (legacy integratie) en dynamische internettoepassingen te creëren.

Naast de web browser, web server, htpp en html uit de eerste generatie waren er nu ook formulieren op de web server beschikbaar. De communicatie tussen applicaties en de webserver vond plaats middels het cgi-protocol (zie figuur 2).



Figuur 2 De tweede generatie: http, html, forms en cgi.

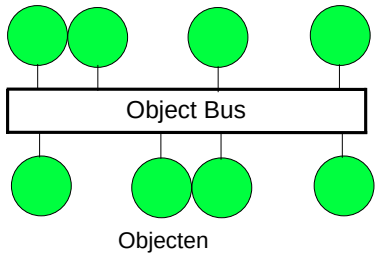
Ondanks de grote vooruitgang die deze nieuwe technologieën met zich meebrachten, kleefden er toch ook een aantal problemen aan. Deze problemen stonden het gebruik van het Internet voor zakelijke toepassingen in de weg. Zo was het niet mogelijk om met behulp van invulvelden en knoppen geavanceerde gebruikersinterfaces te ontwikkelen. Tevens ontstond het zogenaamde *stateless* probleem, de toestand van de interactie tussen de web browser en de web server werd niet bewaard. Hierdoor was het niet mogelijk om meer geavanceerde internettoepassingen te ontwikkelen.

Een ander probleem dat ontstond was dat de schaalbaarheid van deze generatie internettoepassingen gering was. De performance werd negatief beïnvloed bij een toename van het aantal web browsers. Dit omdat er elke keer een proces op de web server opgestart moest worden wanneer een web browser informatie uit een traditionele omgeving opvroeg. Verder was het definiëren van robuuste transacties niet of nauwelijks mogelijk en bleef het ontsluiten van legacy systemen lastig. Kortom, in het gebruik van internettechnologie voor het beschikbaar stellen van zakelijke toepassingen werd niet of nauwelijks voorzien.

De toekomst van het Internet

Doordat de combinatie van het Internet met een object bus en objecten deze problemen oplost, zal het Internet in de toekomst door zakelijke toepassingen overspoeld worden. Hierbij vormen de object bus en objecten de toekomst van het Internet.

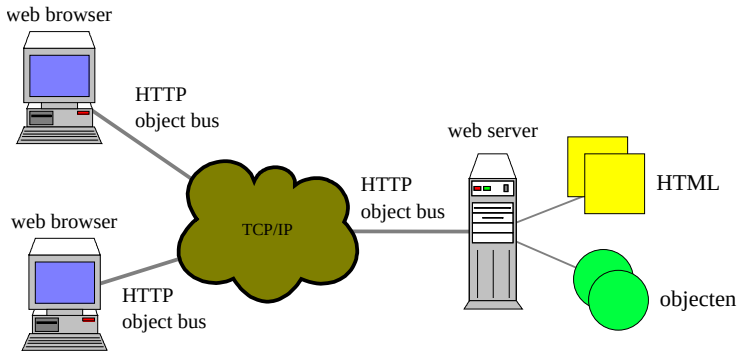
Een object bus [1][2][3] biedt de infrastructurele voorziening zodat objecten mogelijk onafhankelijk van hardware, besturingssysteem, programmeertaal en netwerkprotocol met elkaar kunnen communiceren. Een object bus biedt dus een transparante vorm van communicatie tussen objecten (zie figuur 3).



Figuur 3 Objecten communiceren middels een object bus.

De object bus treedt op als een soort makelaar. Hij redigeert vragen van objecten naar andere objecten en verzorgt de terugkoppeling van het antwoord. De objecten bieden diensten aan elkaar aan.

Naast de al bekende elementen uit de eerste en tweede generatie bestaat de toekomstige (derde) generatie internettechnologie ook uit een object bus en objecten (zie figuur 4).

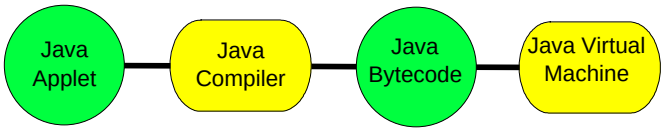


Figuur 4 De toekomst: http, html, object bus en objecten.

Het opvragen en versturen van de html-code en objecten van de web server naar de web browser vindt plaats met behulp van het http-protocol. De communicatie tussen objecten onderling echter wordt verzorgd door de object bus. Ondanks deze onderliggende principes zijn er twee duidelijk verschillende uitwerkingen van een object bus en objecten waarneembaar. Eén uitwerking wordt gevormd door Netscape: Corba als object bus en Java Applets als objecten. De andere uitwerking door Microsoft: DCom als object bus en ActiveX componenten als objecten.

Netscape - Java en Corba

Java is een krachtige platform onafhankelijke object georiënteerde programmeertaal voor derde generatie internettoepassingen. Met Java worden objecten gecreëerd, zogenaamde Java Applets. Om platform onafhankelijkheid te creëren, vindt compilatie van een Java Applet in uitvoerbare code in twee stappen plaats.



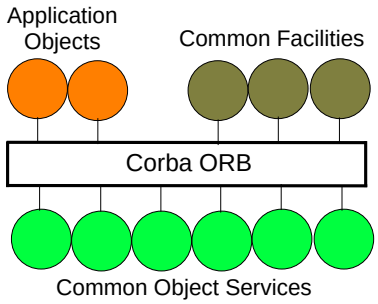
Figuur 5 Compilatie van een Java Applet

Allereerst wordt de Java Applet gecompileerd tot platform onafhankelijk Java Bytecode. Deze Java Bytecode bevindt zich op de webserver. Nadat een webbrowser een verzoek doet voor een Java Applet, verstuurt de webserver de Java Bytecode met behulp van het http-protocol. In de webbrowser wordt vervolgens de Java Bytecode geïnterpreteerd door de zogenaamde Java Virtual Machine. De platform onafhankelijkheid wordt gecreëerd doordat er diverse implementaties beschikbaar zijn van de Java Virtual Machine.

Om met behulp van hergebruik van Java Applets applicaties te bouwen, het zogenaamde *Component Based Application 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. Bedrijven als JScape (o.a. Java Beans voor de bouw van GUI's), Corel, EnterpriseSoft, Gemstone, IBM (o.a. CICS Gateway), K&A Software, KL Group, Lotus Development, Novell, ProtoView Development, Rogue Wave, en Stingray Software hebben aangekondigd standaard Java Beans te ontwikkelen.

De Java programmeertaal is een platform onafhankelijk object georiënteerde programmeertaal. Echter, doordat Java alleen de communicatie tussen Java Applets onderling ondersteunt, is een infrastructuur die alleen gebaseerd is op Java niet open. Door het gebruik van Corba als object bus voor de communicatie tussen Java Applets en andere objecten wordt een open infrastructuur gecreëerd.

Corba staat voor *Common Object Request Broker Architecture* (zie figuur 6). Deze object bus middleware standaard is ontwikkeld (en wordt nog steeds verder 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. Medio 1997 zijn meer dan 600 bedrijven en instellingen uit de softwarebranche lid van de OMG [1].



Figuur 6 De Corba object bus standaard

De *Object Request Broker* (ORB) is de object bus van Corba. De ORB verzorgt de communicatie tussen objecten. Objecten kunnen tijdens uitvoer aan de ORB vragen welke diensten beschikbaar zijn en er vervolgens gebruik van maken. De *Common Object Services* zijn een verzameling van

elementaire diensten verpakt als objecten. Ze zijn een uitbreiding op de diensten van een ORB. Er zijn services gedefinieerd voor bijvoorbeeld security, transacties, concurrency en persistentie. De *Application Objects* zijn de objecten die de organisatie modelleren en de gebruikers bij hun werkzaamheden ondersteunen. Deze objecten worden ook wel aangeduid met de term *Business Objects*. De application objects maken gebruik van object services. Om de ontwikkeling van toepassingen te vereenvoudigen kent Corba de zogenaamde *Common Facilities*. Dit zijn standaard raamwerken (*frameworks*) van samenwerkende objecten die kant-en-klaar ingezet kunnen worden in specifieke situaties. Er zijn twee soorten Common Facilities: horizontale en verticale. De horizontale Common Facilities bieden domein onafhankelijke ondersteuning, zoals bijvoorbeeld werkstroombesturing. De verticale Common Facilities bevatten objecten voor de ondersteuning van specifieke bedrijfstakken, bijvoorbeeld de sociale zekerheid.

Tegenwoordig zijn er verschillende commerciële Corba implementaties beschikbaar, zoals: Orbix van Iona Technologies Ltd, Object Broker van Digital, Powerbroker van Expertsoft, ORB Plus van HP, NEO van Sun, Smalltalk Broker V. van DNS, VisiBroker van Visigenic Software Inc en CBConnector van IBM (officiële release datum: oktober ’97).

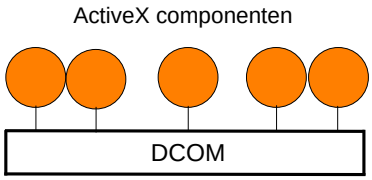
Met behulp van de *Netscape Navigator* als web browser en de *Netscape Suitespot server* als web server is het mogelijk om derde generatie internettoepassingen met behulp van Java en Corba te realiseren. Zowel de web browser als de web server van Netscape bevatten naast een Java Virtual Machine om Java Applets te kunnen draaien ook een Corba implementatie voor de communicatie tussen Java Applets onderling en met andere objecten. De Corba implementatie in zowel de web browser als web server van Netscape wordt gevormd door de VisiBroker van het bedrijf Visigenic.

Microsoft - ActiveX en DCom

Met behulp van ActiveX is het mogelijk om bestaande componenten geschreven in verschillende programmeertalen te combineren door encapsulatie. Zo is het mogelijk om Java Applets te encapsuleren in ActiveX. In tegenstelling tot Java Applets of Java Beans worden ActiveX componenten niet geïnterpreteerd maar gecompileerd. Door deze compilatie wordt een snelheidsvoordeel ten opzichte van Java verkregen, echter ten kostte van de platform onafhankelijkheid. ActiveX componenten draaien uitsluitend onder Windows 95 en Windows NT.

Ook met ActiveX componenten is het mogelijk om *Component Based Application Development* te plegen. Bedrijven als Network Imaging Corporation (o.a. gateway voor diverse databases), Wall Data (o.a. gateway naar AS/400), ChartFX (o.a. componenten om grafieken te maken), National Instruments, English Wizard (o.a. vertaling van semi-engels naar SQL), VideoSoft, The Digital Foundry en natuurlijk Microsoft zelf leveren standaard ActiveX componenten.

DCom (distributed component object model) is een object bus die de communicatie verzorgt tussen gedistribueerde objecten (zie figuur 9). DCom is gebaseerd op Com, een standaard die de communicatie verzorgt tussen objecten op één machine. Een object kan tijdens de uitvoering informatie verkrijgen over de interfaces van andere gedistribueerde objecten en vervolgens van deze objecten gebruik maken. DCom kent een aantal simpele voorzieningen voor licensing, events, life cycle management en dergelijke. Complexere voorzieningen zoals transacties worden niet geleverd. Tevens kunnen ook Windowsdiensten als bestandsbeheer en geheugenallocatie worden aangeroepen. Ook hier komt het platform afhankelijke karakter van de Microsoft uitwerking naar boven. Een windowsdienst als bestandsbeheer is namelijk niet beschikbaar op bijvoorbeeld een Apple Macintosh.



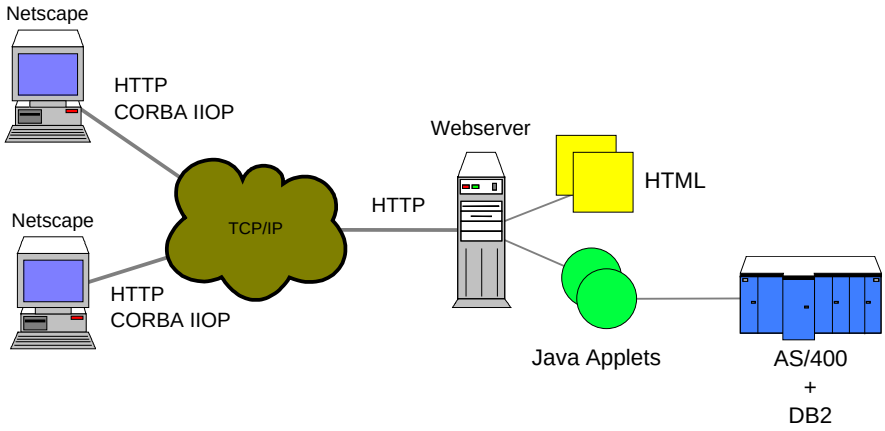
Figuur 7 De DCOM object bus

Met behulp van Windows 95 en Windows NT is het mogelijk om derde generatie internettoepassingen te realiseren. Microsoft levert hiervoor een tweetal produkten voor de Windows 95 kant: *DCom voor Windows 95* en *Internet Explorer* als web browser. Aan de Windows NT kant levert Microsoft de *Internet Information Server* als de web server en de *Transaction Server* om transacties over ActiveX componenten heen te definiëren. DCom voor de server kant zit tegenwoordig standaard in Windows NT.

Legacy integratie

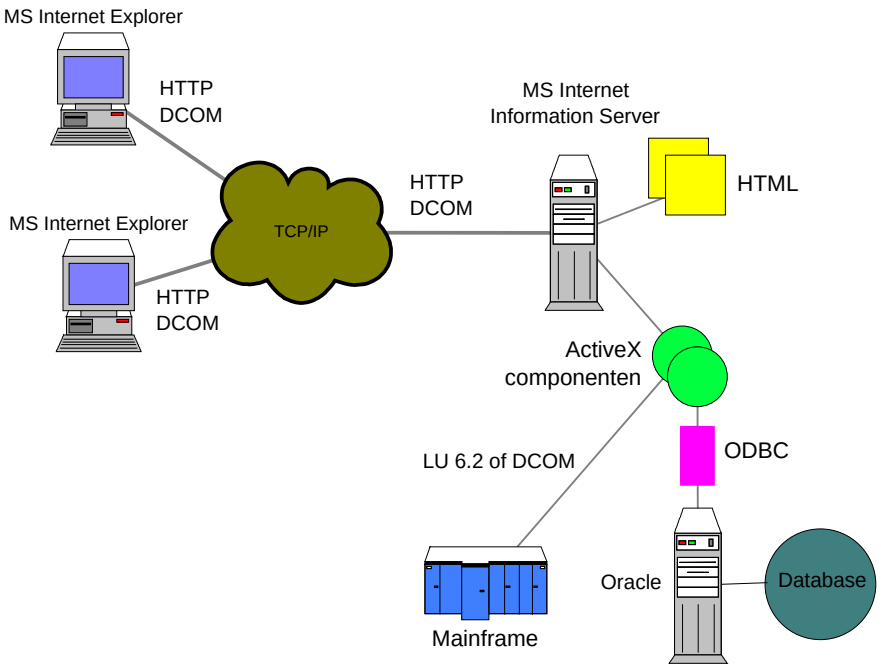
In de uitwerking van zowel Netscape als Microsoft is het mogelijk om door het gebruik van standaard componenten legacy applicaties te ontsluiten.

In de uitwerking van Netscape kan zo'n standaard component een Java Applet, Java Bean of elke in een andere taal geschreven component die met behulp van de Corba ORB aanspreekbaar is, zijn. Op deze manier is het mogelijk dat een Java Applet via de object bus communiceert met een legacy applicatie. Zo is het bijvoorbeeld mogelijk dat een Java Applet communiceert met een DB2 database op een AS/400 middels JDBC (zie figuur 8).



Figuur 8 Ontsluiting van een DB2 database via het Internet

In de uitwerking van Microsoft is het mogelijk dat standaard ActiveX componenten communiceren met legacy applicaties middels cgi, ODBC of de SNA Server van Microsoft. Op deze manier is het bijvoorbeeld mogelijk dat een ActiveX component communiceert met een Oracle database middels ODBC of met een MVS Mainframe applicatie middel de SNA Server (zie figuur 9).



Figuur 9 Ontsluiting van een Oracle database en een mainframe applicatie middels Internet

Een andere ontwikkeling van Microsoft (uitgevoerd door het bedrijf Software AG) is de DCom implementatie voor MVS. Met deze implementatie is het theoretisch mogelijk dat ActiveX componenten op Windows NT communiceren met ActiveX componenten op een MVS Mainframe.

De vergelijking

Als de uitwerkingen van Netscape en Microsoft vergeleken worden dan is in het algemeen te stellen dat Netscape een *open* en *platform onafhankelijk* Internet infrastructuur biedt, gebaseerd op Corba en objecten geschreven in Java, Java Beans en alle andere voor Corba geïmplementeerde implementatietalen. Microsoft daarentegen biedt een *gesloten* en *platform afhankelijke* Internet infrastructuur, gebaseerd op DCom en ActiveX componenten.

Bij de keuze voor het volgen van één van de gepresenteerde uitwerkingen is dan ook de bestaande infrastructuur binnen de eigen organisatie van doorslaggevend belang. Grofweg kan gekozen worden voor de Microsoft oplossing indien de basis van de eigen infrastructuur gevormd wordt door Windows 95 en Windows NT, en er weinig legacy integratie nodig is. Indien er een diverse infrastructuur aanwezig is en legacy integratie van groot belang is, ligt de keuze voor de open en platform onafhankelijk uitwerking van de Netscape voor de hand.

Als we kijken naar het aantal beschikbare componenten valt op dat Microsoft hier vooruit loopt daar de ActiveX component standaard na een aantal jaren nu volwassen genoemd mag worden. Het principe van Java Beans daarentegen is een recent ontwikkelde standaard. Doordat ActiveX vooruit loopt op Java Beans zijn er op dit moment meer ActiveX componenten verkrijgbaar dan Java Beans. De voorsprong van Microsoft met betrekking tot het aantal beschikbare ActiveX componenten is echter van minder groot belang daar het slechts een kwestie van tijd is tot dit gat door de standaard Java Beans overbrugd is.

De voordelen

Het beschikbaar stellen van toepassingen met behulp van derde generatie internettechnologie biedt een aantal bijkomende voordelen naast de voordelen van de eerste twee generaties: platformonafhankelijkheid, automatische distributie, centraal onderhoud en integratie met een traditionele omgeving. Doordat objecten zowel op web browser als op de web server uitgevoerd kunnen worden is het mogelijk om geavanceerde interactie met eindgebruikers aan te kunnen, dynamische internettoepassingen te creëren en integratie met een traditionele omgeving te realiseren. Omdat een object bus transparante communicatie verzorgt wordt het mogelijk om objecten te dupliceren over meerdere web servers. Hierdoor is het mogelijk om door middel van load balancing een goede performance te realiseren. De schaalbaarheid van de toepassing neemt toe. Tevens kent het gebruik van een object bus geen *stateless* probleem doordat de toestand van de communicatie tussen objecten onderling behouden blijft in Corba en DCom. Ook het definiëren van transacties over objecten heen is mogelijk doordat Corba standaard transaction services en Microsoft de Transaction Server levert.

Kortom, door gebruik te maken van state-of-the-art internettechnologie is het mogelijk om zakelijke toepassingen via het Internet beschikbaar te stellen.

Auteur

- *Drs. P.J. Koning* is werkzaam bij de Technology Consulting Group van Cap Gemini Nederland b.v.

Literatuur

1. Gedistribueerde Objecten: Corba, OLE en OpenDoc, *G.M.A. van der Harst*, Personal Computer Gids, 1996.
2. The Essential Client/Server Survival Guide - Second Edition, *R. Orfali, D. Harkey en J. Edwards*, John Wiley & Sons, 1996.
3. The Essential Distributed Objects Survival Guide - Second Edition, *R. Orfali, D. Harkey en J. Edwards*, John Wiley & Sons, 1996.