In de praktijk zien we sterk uiteenlopende toepassingen van use cases. Sommigen beschrijven er bedrijfsprocessen mee, anderen proberen ze objectgeoriënteerd te maken en weer anderen zien er mogelijkheden voor functionele decompositie in of beschrijven er de user interface mee. En allen lopen daar op de een of andere manier in vast. De literatuur is ook al niet eenduidig. Dit artikel beoogt een bijdrage te leveren aan het gebruik van use cases op de manier zoals oorspronkelijk bedoeld: het weergeven van de scope en functionaliteit van een te realiseren systeem.

Correct gebruik van use cases

In kaart brengen van scope en functionaliteit

Met de opkomst van objectgeoriënteerde methoden en technieken hebben use cases zich een vaste plaats verworven bij het identificeren en beschrijven van systeemeisen. Merkwaardig, als we bedenken dat use cases niets met objectoriëntatie te maken hebben. Ook niet met functionele decompositie. Maar daarom biedt de use case techniek juist de mogelijkheid om het identificeren en beschrijven van systeemeisen los te maken van de aanpak die gevolgd wordt bij het ontwerpen van systemen. Use cases zijn bij uitstek geschikt om, op het grensvlak van vooronderzoek en functioneel ontwerp, de scope en functionaliteit van een te realiseren systeem eenduidig vast te leggen. Dit kan gerealiseerd worden op een manier die alle betrokkenen (ook gebruikers!) begrijpen: doelgericht en weergegeven in natuurlijke taal met zo weinig mogelijk schema's.

ONTWIKKELDOMEINEN Bij het analyseren, ontwerpen en bouwen van informatiesystemen onderscheiden we een aantal ontwikkeldomeinen:

- die van de gebruikers (gericht op verbetering van de bedrijfsvoering),
- de informatieanalisten (gericht op het aanreiken van oplossingsrichtingen voor een verbeterde bedrijfsvoering, het vooronderzoek),
- de ontwerpers (gericht op ICT-ondersteuning, het systeemontwerp) en
- de bouwers (de realisatie).

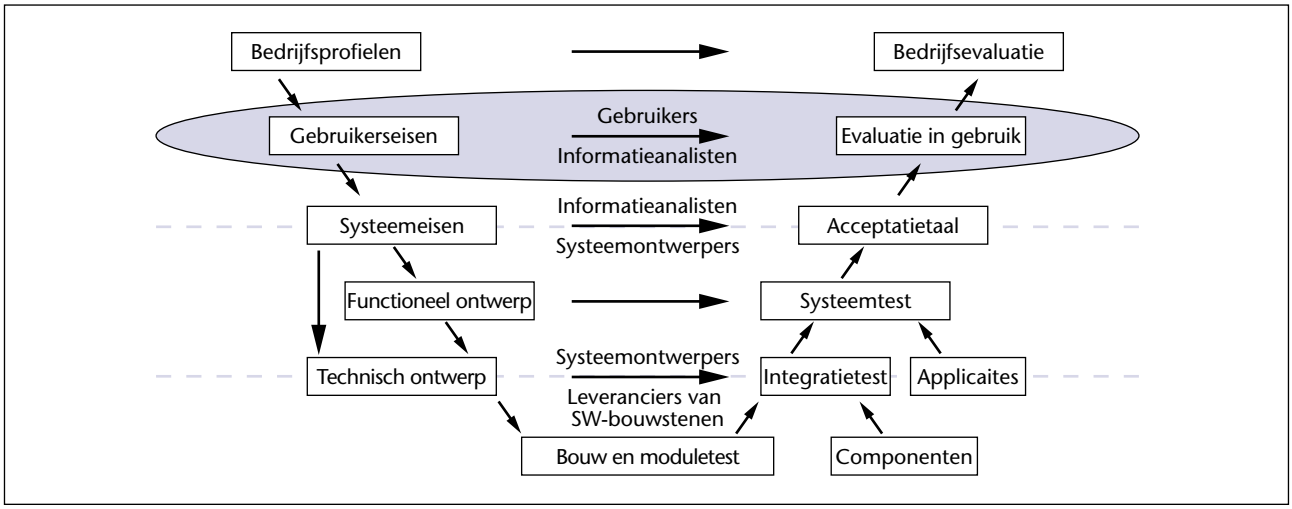
In figuur 1 hebben we de domeinen door middel van stippellijnen van elkaar gescheiden. Telkens als we een domeingrens passeren krijgen we in formele zin te maken met een klant-leverancierrelatie. Boven de stippellijn staat telkens de klantrol, onder de stippellijn de leveranciersrol. Links worden eisen en specificaties doorgegeven, rechts worden softwareproducten getest, geassembleerd en geïntegreerd. Het proces dat daarbij gevolgd wordt kan lineair, incrementeel of evolutionair zijn.

Use cases treffen we aan op het grensvlak van vooronderzoek en systeemontwerp (zie ovaal). De informatieanalist die het vooronderzoek verricht reikt de organisatie een aantal oplossingsalternatieven aan voor behoeften die er bestaan. Per alternatief zal hij een beeld schetsen van de toekomstige situatie en de veranderingen die daarvoor nodig zijn, in termen van bedrijf of organisatie, bedrijfsprocessen, informatiebehoeften, ICT-systemen en infrastructuur, met daarbij een plan van aanpak.

Use cases worden daarbij gebruikt voor het doorgeven van de scope en de functionaliteit van een te ontwikkelen ICT-systeem. De informatieanalist treedt op als klant, de ontwerpers van het systeem vervullen de leveranciersrol.

USE CASES, DE TECHNIEK De belangrijkste elementen die in de use case techniek worden toegepast zijn in figuur 2 samengevat.

We zien een Ordersysteem met daarin twee functies (use cases): "Order plaatsen" en "Overzicht openstaande orders maken". De actoren zijn Klant en Verkoper. De use cases zijn de gebruiksmogelijkheden waarin het



FIGUUR 1. Ontwikkel-domeinen

Ordersysteem moet voorzien om de actoren Klant en Verkoper ermee te laten werken. De aanleiding om een use case uit te voeren is een prikkel uit de omgeving van het systeem waarop een response moet plaatsvinden, bijvoorbeeld "Klant plaatst order". Die prikkel noemen we een "event". We komen daar nog op terug. Klant initieert de use case "Order plaatsen" en is tevens een bron van informatie voor het systeem. Er wordt dus niet aangegeven dat een klant rechtstreeks met het systeem zou kunnen werken.

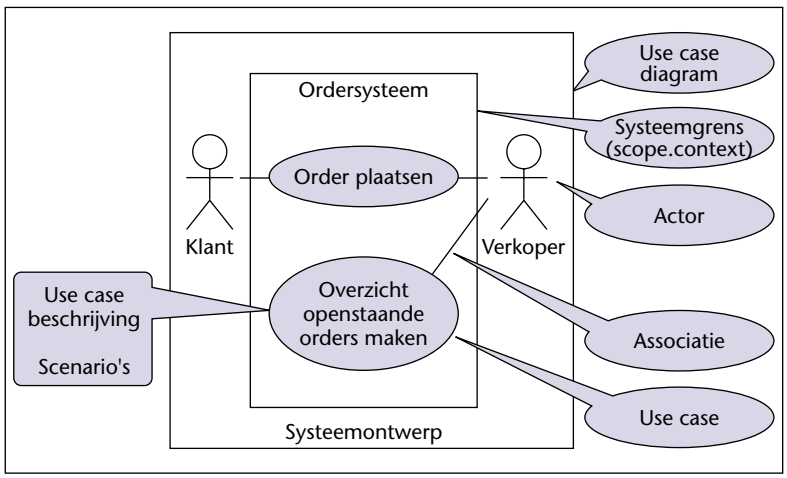
HET BESCHRIJVEN VAN USE CASES Natuurlijk is een grafische afbeelding alleen niet voldoende. Voor het beschrijven van een use case reiken we een template aan (figuur 3).

EVENTS ALS VERTREKPUNT Een event is een prikkel waarop het systeem een response moet geven. Die prikkel betreft een enkelvoudig feit, komend vanuit de omgeving, vanaf één plaats (actor) en op één punt in de tijd. Het event initieert een proces dat in zijn geheel moet worden uitgevoerd om een nieuwe consistente toestand in het systeem te bereiken. De functionaliteit die de response naar aanleiding van een event verzorgt noemen we een use case.

Alle mogelijke events waarop het systeem een response moet geven worden verzameld in een event list (figuren 4 en 5). Die bevat twee soorten events: stroomevents en tijd-events.

Een stroomevent is een prikkel uit de omgeving van het informatiesysteem, afkomstig van een actor, waar het informatiesysteem op moet reageren. Bijvoorbeeld: "Klant plaatst order". De klant initieert de use case, daarna komt ook de verkoper als actor in beeld.

Een tijd-event is een op tijd gebaseerde afspraak met de omgeving. Periodiek moet het systeem in actie komen zonder prikkel uit de omgeving. Bijvoorbeeld: "Het is tijd om een overzicht van openstaande orders te publiceren". Er is geen actor die de use case initieert (tenzij we de tijd als "actor" in het model meenemen).



FIGUUR 2. Use cases; de techniek

OMGEVINGSMODELLEN Een omgevingsmodel drukt uit wat de omgeving van een systeem, de gebruikerswereld dus, van het systeem verwacht.

Het omgevingsmodel volgens SA/SD (Yourdon) omvat (zie figuur 4):

- Een omschrijving van het doel van het systeem
- Een event list (met omschrijvingen)
- Een informatiemodel
- Een context diagram

Het overeenkomstige omgevingsmodel volgens de objectgeoriënteerde aanpak (UML) omvat (zie figuur 5):

- Een omschrijving van het doel van het systeem
- Een event list (met omschrijvingen)
- Een use case diagram.

vervolg op pagina 49.

vervolg van pagina 8.

Zo op het eerste gezicht lijken de verschillen in het vak "Systeemontwerp" groter dan ze in werkelijkheid zijn. Bij nadere beschouwing moeten we constateren dat beide omgevingsmodellen elkaar in het vak "Systeemontwerp" aanvullen. Het context diagram in figuur 4 verduidelijkt de associaties van het use case diagram in figuur 5, terwijl het use case diagram de onder het context diagram liggende event-responseprocessen laat zien. Het informatiemodel (figuur 4) kan het objectgeoriënteerde systeemontwerp een eerste aanzet geven bij het opzetten van een class diagram.

DE VERDERE DETAILLERING Met het vastleggen van het omgevingsmodel is de grens getrokken tussen vooronderzoek en ontwerp. Als de ontwerpers werken volgens de gestructureerde aanpak, dan zullen ze, met het omgevingsmodel van figuur 4 als vertrekpunt, verder detailleren met behulp van een hiërarchische set van dataflow diagrams, een informatiemodel, procesSpecificaties en een data dictionary. De event-responseprocessen, de use cases dus, staan daarbij aan de basis.

Werken de ontwerpers volgens de objectgeoriënteerde aanpak, dan is het omgevingsmodel van figuur 5 vertrekpunt. Men zal het procesverloop in de use cases vertalen naar sequence diagrams, en daarbij tevens de functionaliteit verdelen over objecten. Die krijgen daarmee de verantwoordelijkheid voor het bewaren van gegevens en het uitvoeren van processen. Het class diagram is daarin het centrale deel. Verder worden ook in de objectgeoriënteerde aanpak tekstuele specificaties toegevoegd voor het beschrijven van processen (operations) en gegevens.

TOEPASSING VAN USE CASES IN HET VOORONDERZOEK Bij het in kaart brengen van bedrijfsprocessen willen we aangeven of er afhankelijkheden bestaan tussen use cases onderling. We denken daarbij aan:

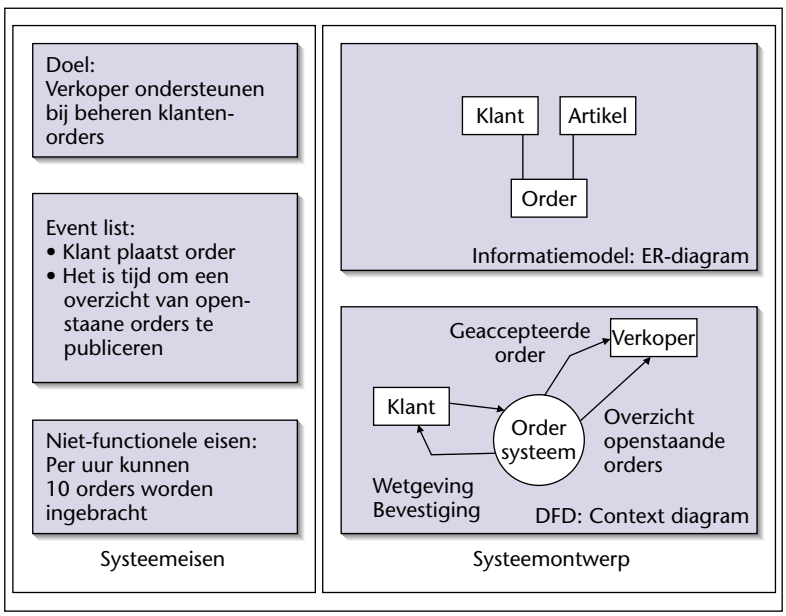
- volgorde
- keuze
- herhaling
- concurrency
- synchronisatie.

Omdat een use case diagram geen afhankelijkheden kan weergeven, kunnen we onze toevlucht nemen tot pre- en postconditions in de beschrijvingen van use cases. Maar daarmee verliezen we snel het overzicht over het geheel. Een activity diagram is een beter alternatief. Ter illustratie volgt hier een voorbeeld van een orderverwerking (Figuur 6). Het activity diagram (rechts) maakt onder andere duidelijk dat cancelen van een order niet meer mogelijk is na leveren of factureren. Leveren enerzijds, en factureren en betalen anderzijds, zijn concurrent, maar de order wordt pas gearchiveerd als beide wegen zijn afgelegd.

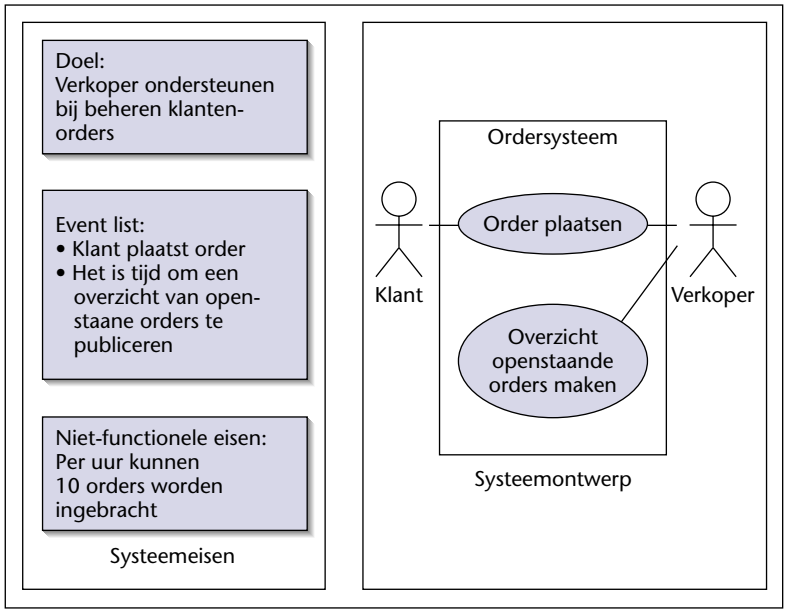
Naam	Een unieke naam waarin kernachtig wordt uitgedrukt welke service de use case verleent ten behoeve van een of meer actoren *Vb: Order plaatsen*
Omschrijving	Een korte omschrijving van de service *Vb: De klant wordt getoetst op kredietpositie. Afhankelijk van de uitkomst wordt de order geaccepteerd of geweigerd*
Precondition	Een aanduiding voor de toestand waarin een systeem zich moet bevinden als de use case geïnitieerd wordt, veelal uitgedrukt in termen van randvoorwaarden *Vb: Alleen geregistreerde klanten kunnen orders plaatsen*
Postcondition	Een aanduiding voor de toestand waarin een systeem zich bevindt nadat de use case de verlangde service verleend heeft *Vb: De order is geaccepteerd en bevestigd, of geweigerd*
Actoren	De actoren die actief of passief betrokken zijn bij de use case *Vb: Klant, Verkoper*
Main course of action	De normale processtroom binnen de use case waarbij zich geen exceptionele gevallen voordoen *Vb: De orderwaarde wordt berekend en opgeteld bij het kredietniveau van de klant. Als daarbij de kredietlimiet met niet meer dan 10% wordt overschreden, wordt de order onder voorbehoud geaccepteerd; bij overschrijding met meer dan 10% wordt de order geweigerd met een bericht aan de klant*
Exceptional course(s)	Een beschrijving van alle mogelijke exceptionele processtromen

FIGUUR 3. Template

Aan de techniek zelf kunnen we niet zien of het bedrijfsniveau dan wel systeemniveau in kaart wordt gebracht. We moeten daar geen verwarring over stichten, maar door middel van tekst expliciet duidelijk maken waar we het over hebben. Onze ervaring is dat het in kaart brengen van bedrijfsprocessen ook uitste-



FIGUUR 4. Omgevingsmodel volgens SA/SD (Yourdon).



FIGUUR 5. Omgevingsmodel volgens OO (UML)

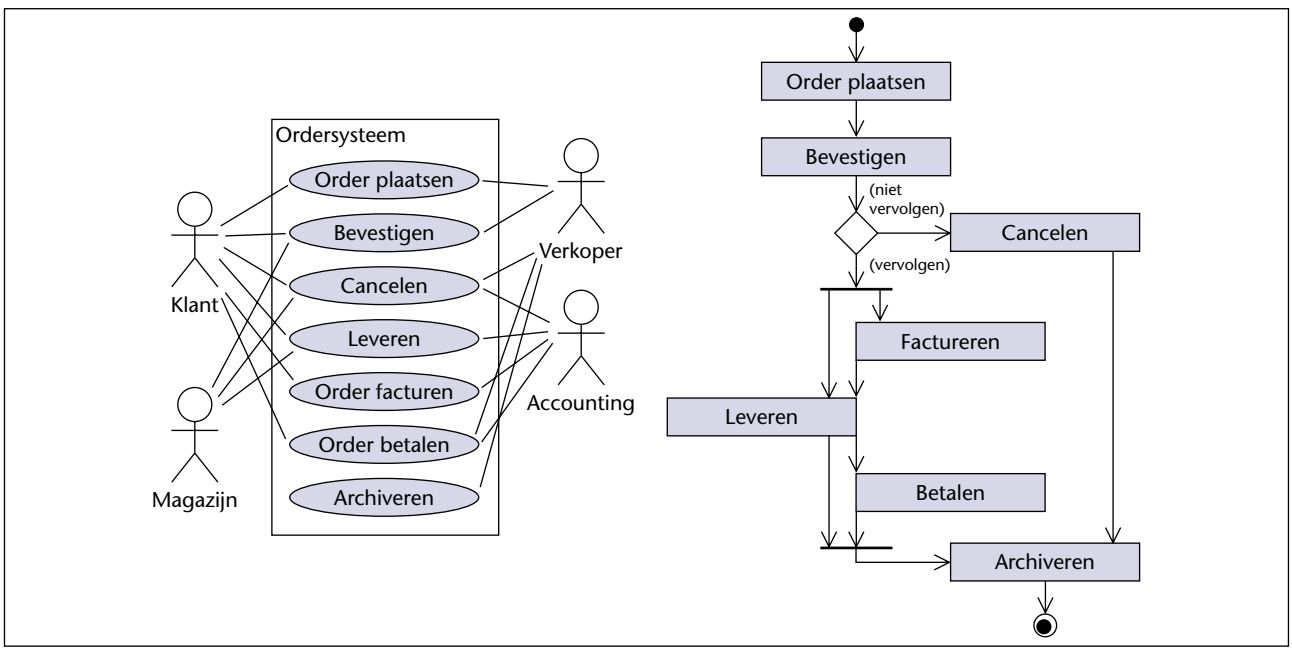
kend kan met behulp van technieken als IDEF0. Met behulp van deze technieken stellen we een model op van processen met in- en uitgaande stromen en besturingsstromen die worden uitgewisseld met de omgeving en tussen processen onderling. In figuur 7 zien we een eenvoudig voorbeeld. We hoeven nu ook niet meer expliciet te maken waar we het over hebben: IDEF0 is bedrijfsniveau, Use case diagram is systeemniveau.

DE USER INTERFACE Voor het beschrijven van de user interface zijn use cases niet erg geschikt, omdat navigatiemogelijkheden en validaties moeilijk te traceren zijn. In de praktijk worden daarom andere technieken toegepast zoals schermlayouts, menubomen, dialoogstructuurdiagrammen enzovoort. Onze ervaring is dat men, in verband met de dynamiek van user interfaces, de documentatie daarover zoveel mogelijk moet zien te beperken. Men kan beter een bedrijfsbrede standaard opzetten en alleen de afwijkingen daarvan in kaart brengen.

OPTIONELE UITBREIDING UML ondersteunt het in kaart brengen van structurele relaties die een use case met andere use cases onderhoudt, in termen van <<extend>> (optionele uitbreiding) en <<include>> (verplichte insluiting). Toepassing van deze technieken leidt soms tot onoverzichtelijke schema's. Er is niets tegen het gebruik van beide features voor het doel waarvoor ze bestemd zijn: optionele uitbreiding met extra functionaliteit (bijvoorbeeld het uitvoeren van extra controles bij het plaatsen van orders waarvan de orderwaarde boven een vastgesteld bedrag uitkomt) of gemeenschappelijke functionaliteit (bijvoorbeeld het controleren van de gegevens van het ponskaartje van een patiënt bij zowel opname als bij het maken van een poliklinische afspraak). Het probleem met <<extend>> en <<include>> zit hem vaak in de overdreven toepassing, waarbij men probeert de use cases op een objectgeoriënteerde manier te modelleren (bijvoorbeeld het "includen" van "functies" die niet méér inhouden dan create, retrieve, update of delete van een gegevensgroep).

SAMENVATTING We hebben opgemerkt dat:

- use cases tijdens het vooronderzoek vaak aanvulling behoeven met activity diagrams, en dat er andere technieken zijn die ook goed toepasbaar zijn voor het beschrijven van bedrijfsprocessen
- use cases een belangrijke rol vervullen op het grens-

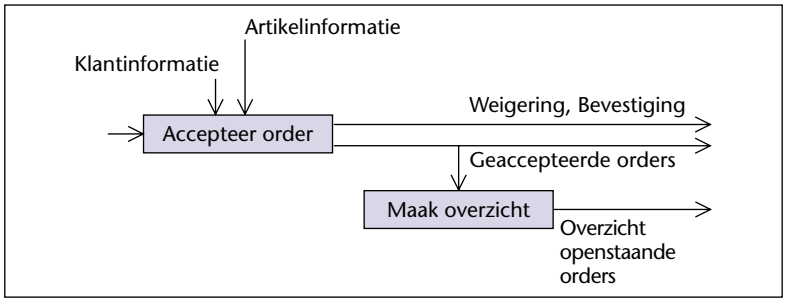


FIGUUR 6. Use cases en Activity diagram

vlak van vooronderzoek en functioneel ontwerp waarbij:

- de scope en de functionaliteit van een te realiseren systeem eenduidig wordt vastgelegd en overgedragen van informatieanalist naar functioneel ontwerper
- de acceptatietest tevens gestuurd wordt
- de use case techniek steunt op eventanalyse, en slechts een eenvoudige, maar daardoor beperkte schematechniek gebruikt voor het visualiseren van de hoofdlijnen van de functionaliteit
- de use case techniek vrij is van de aanpak die gevolgd wordt bij het ontwerpen van systemen: gestructureerd of objectgeoriënteerd
- na het passeren van het grensvlak in zowel de gestructureerde als in de objectgeoriënteerde aanpak andere technieken dan use cases nodig zijn voor het verder detailleren van het functioneel ontwerp:
- dataflow diagrams, informatiemodel, processpecificaties en data dictionary in de gestructureerde aanpak
- sequence diagrams, class diagram, operations en attributes specificaties in de objectgeoriënteerde aanpak
- menubomen en dialoogstructuurdiagrammen voor het in kaart brengen van user interfaces.

Conclusie De use case techniek is eenvoudig en doeltreffend voor het doel waarvoor deze bestemd is: het in kaart brengen van de scope en de functionaliteit van een systeem. Problemen bij de toepassing ontstaan bij verkeerd gebruik door bijvoorbeeld:



Figuur 7. IDEF0 diagram

- use cases "objectgeoriörienteerd" te maken
- een functionele decompositie uit te voeren met use cases
- de user interface te beschrijven met behulp van use cases.

Literatuur

[1]: Pollaert, W, Ruigrok, K, *Informatieanalyse, De brug van bedrijfsdoelen naar ICT-oplossingen*, Thema, 2002

[2]: Jacobson, I., *The Object Advantage*, Addison Wesley, (1994)

[3]: Fowler, M., *UML Distilled (Second Edition)*, Addison Wesley, (2000)

[4]: Schneider, G., *Applying Use Cases (Second Edition)*, Addison Wesley, (2001).

Wiel Pollaert is werkzaam bij ISES International BV

In memoriam: Petra van Krugten

Op 9 mei 2003 overleed op 46-jarige leeftijd volkomen onverwacht Petra van Krugten-Elgersma. Samen met Mark Hoogenboom, met wie zij het Web-Enabled Team (www.we-team.com) vormde, verzorgde zij vanaf 1999 de column 'Softskills' voor Software Release Magazine. Petra was (mede-)auteur van een zestal boeken, columniste en werkzaam als principal consultant bij Cap Gemini Ernst & Young. In 1974 begon zij bij Philips als 'computerdeskundige' en heeft daarna alle stadia doorlopen die een gedegen ICT carrière kenmerken: van programmeren tot strategisch advies, van ponskaart tot de mobiele werkplaats. Petra had een brede kijk op het ICT-vakgebied. Een onderdeel van haar expertise betrof het implementeren van de Iteratieve Applicatie Development methode (IAD). Daarnaast gaf Petra colleges aan de Haagse Hogeschool, Hogeschool INHolland en ook de Rijksuniversiteit Groningen en was zij regelmatig spreker op verschillende seminars. De laatste jaren was zij bezig met het effectief faciliteren van groepsprocessen. Petra vond het, in een tijd van ongekende technologische mogelijkheden en snelle verandering, niet alleen van belang te beschikken over technologische kennis. Evenzeer is de juiste 'menselijke' begeleiding in veranderingstrajecten noodzakelijk, stelde Petra: "Zodat men met plezier de techniek van nu gaat uitproberen en toepassen en men vaardig wordt in wat over twee jaar noodzakelijk is om te overleven".

Het plotselinge overlijden van Petra van Krugten was ook voor de redactie van Software Release Magazine een grote schok. Haar echtgenoot, familie, vrienden en collega's wensen wij veel sterkte bij het dragen van dit verlies. Wellicht kan de volgende uitspraak van Petra zelf een steun zijn voor hen, die nu zonder haar verder moeten: "*We moeten de durf hebben het leven dat we hebben uitgestippeld los te laten om het leven te leiden dat op ons wacht.*"