Prof. dr. J. P. C. Kleijnen*

Decision Support Systems (DSS), en de kleren van de keizer...

1. Inleiding

'Decision Support Systems' (DSS) staan sterk in de belangstelling. Wat DSS echter precies zijn, is onduidelijk. Behoren 'spreadsheets' tot de DSS? Is de term DSS een nieuw etiket voor Management Information Systems (MIS) en/of Operations Research (OR)? Bestaan DSS echt, of zijn ze als de kleren van de keizer van China: ieder sprak er over, maar...? (zie ook Sol, 1985). In mijn verhaal heb ik de volgende begrippen nodig. Een bedrijf heeft een primair doel, bijv. staalproduktie. Het geheel van produktiemiddelen ten behoeve van dat primaire doel heet in de informatiekunde het *objectsysteem* of fysiek systeem. Om dit systeem te doen functioneren, moeten allerlei gegevens worden vastgelegd; denk aan facturen. Deze registratie in het kader van de primaire bedrijfsprocessen is ook nodig in verband met bureaucratische eisen bijv. ten behoeve van de fiscus. Deze ex-post registratie is de primaire functie van de klassieke boekhouding en de moderne *Electronic Data Processing* (EDP). Om het objectsysteem te kunnen *besturen* moeten bovengenoemde brongegevens en bepaalde externe gegevens (bijv. over de concurrenten) worden bewerkt tot informatie. De kerntaak van de manager is het nemen van beslissingen, d.w.z. het kiezen uit alternatieven. Derhalve is informatie toekomstgericht (ex-ante) (zie Kleijnen, 1980, blz. 9-11).

* Prof. dr. J. P. C. Kleijnen is hoogleraar Simulatie en Informatiesystemen aan de Faculteit der Economische Wetenschappen van de Katholieke Universiteit Brabant.

2. Een voorbeeld: 'GADS'

Ik wil een voorbeeld bespreken dat populair is in de DSS-literatuur (zie Keen en Morton, 1978, blz. 147) en waarmee ikzelf toevallig vertrouwd ben (doordat ik in 1974 een jaar doorbracht bij IBM Research in San Jose, Californië, waar het te bespreken voorbeeld werd ontwikkeld). Het systeem heet *GADS*, oftewel Geodata Analysis and Display System. In GADS vormen geografische gegevens (zoals een stadsplattegrond) een essentieel subsysteem. De eerste toepassing van GADS werd ontwikkeld ten behoeve van de politie van San Jose. Procesverbalen vormen de *brongegevens*. Uit die massale brongegevens worden bepaalde gegevens 'geëxtraheerd' en vastgelegd in tabellen [deze tabellen vormen een 'relationele' database; tussen vierkante haken plaats ik computertechnische opmerkingen]. De gegevens uit de diverse tabellen worden gecombineerd en op een stadsplattegrond gesuperponeerd; deze informatie wordt op een beeldscherm aan de gebruikers gepresenteerd. Die gebruikers communiceren met het computersysteem door middel van een lichtpen.
De eerste gebruikers van GADS waren politiefunctionarissen die probeerden een oplossing te vinden voor een aantal concrete *klachten* (vanuit eigen gelederen) over de indeling van de stad San Jose in politiewijken ofte wel 'beats'. Op het beeldscherm wordt daarom een stadsplattegrond gepresenteerd [dus is digitalisatie van een plattegrond nodig]. Via de lichtpen tekent een agent een bepaalde wijkindeling; de gebruiker bedenkt dus een oplossing.

Vervolgens berekent de computer de gevolgen van die oplossing; de computer berekent dus de waarden van een aantal *criteria*, zoals de werklast per 'beat' gekwantificeerd door (bijv.) het aantal oproepen per dag. Deze criteria zijn expliciet gemaakt (zodat de computer ermee kan rekenen) in een dialoog tussen de verschillende gebruikers, te weten vertegenwoordigers van de wijkagenten en van hun superieuren. Naast deze criteria hebben de gebruikers ook een aantal *nevenvoorwaarden* expliciet gemaakt, bijv. een wijk mag niet doorsneden worden door een grote verkeersweg; immers zo'n weg zou het moeizaam maken om van het ene deel van de wijk naar het andere deel te rijden. (Nevenvoorwaarden betreffen ook vaak de beperkte beschikbaarheid van produktiemiddelen zoals auto's; zie Lineaire Programmering of LP in par. 3.) In de computer zit ook een aantal *procesregels* zodat er geen onzin in de gegevens ontstaat, bijv. een agent komt pas aan bij het misdrijf nadat het misdrijf is gebeurd, resp. gemeld. [Deze procesregels zorgen voor de 'integriteit' van de gegevensbank.] De gebruikers communiceren met elkaar, mede dank zij GADS, d.w.z. er staan meerdere agenten rondom het scherm; een agent formuleert een wijkindeling via de lichtpen (eventueel geholpen door een computerdeskundige); te zamen bekijken de agenten de resultaten (criteria en nevenvoorwaarden), waarna iemand een alternatieve oplossing probeert enz. [Grotere beeldschermen zouden deze dialoog aanzienlijk vergemakkelijken.]

Later is een verwante toepassing van GADS ontwikkeld, nl. de indeling van de stad Palo Alto in *schoolwijken*. De start van het project is wederom een concreet probleem, nl. er komen minder kinderen, zodat de scholen leeg raken. De gebruikers formuleren weer criteria en nevenvoorwaarden. Vergeleken bij de politiewijken zijn er nu grotere tegenstellingen tussen de verschillende soorten gebruikers, te weten ouders en schoolfunctionarissen. Toch wordt GADS uiteindelijk geaccepteerd door alle gebruikers, vooral door de hoge kwaliteit van de gegevens. In deze meer recente toepassing is GADS uitgebreid met geavanceerde algoritmen uit de OR. Dit soort problemen wordt namelijk ook onderzocht met behulp van LP e.d. Maar in de OR moeten alle nevenvoorwaarden expliciet worden gemaakt, terwijl in interactieve DSS de gebruikers bij het formuleren van mogelijke oplossingen impliciet met bepaalde nevenvoorwaarden rekening houden; in het probleem van de schoolwijken bleek men rekening te houden met toezeggingen aan sommige wijkbewoners in het verleden.

3. Beschrijving van DSS

Na het voorbeeld van de vorige paragraaf lijkt de volgende beschrijving redelijk. DSS omvatten als subsystemen:
1. modellen,
2. gegevens.

3.1. Modellen

Bij DSS vinden wij de volgende soorten modellen.

a. Spreadsheets
De werkelijkheid wordt weergegeven in de vorm van een of meer tabellen, elk met twee ingangen (dimensies) waarvan één ingang normaal de tijd weergeeft. Er is veel programmatuur voor microcomputers: Visicalc, Multiplan enz.

b. Simulatie
Voordat spreadsheets op micro's werden ontwikkeld, was er al soortgelijke programmatuur voor grote computers, die diende voor financiële simulaties en bedrijfsmodellen ('corporate models'): IFPS, SIMPLAN enz. (zie Gray, 1984). Naast deze strategische modellen bestaan er (al sinds de jaren vijftig) operationele modellen voor het beheren van voorraden en wachttij-

den. Deze modellen zitten ingebed in programmatuur ten behoeve van het functioneren en beheren van voorraden: IBM's IMPACT en COPICS (zie ook Kleijnen en Rens, 1978). En er is een veelheid van hogere talen voor het simuleren van complexe wachttijdsystemen: GPSS, Simscript, Simula, SLAM, SIMAN, enz.

c. Lineaire Programmering
Bij LP gaat het om statische modellen waarvoor een algoritme automatisch een optimale oplossing vindt; het criterium en alle nevenvoorwaarden moeten lineaire vergelijkingen vormen (zoals $a_1x_1+a_2x_2+...+a_nx_n=b$). (Bij simulatie ging het om dynamische modellen, d.w.z. de ontwikkeling van een systeem over de tijd wordt nagebootst; het model is niet lineair; er zit geen algoritme in dat optimaliseert; de gebruiker formuleert 'what-if'-vragen en het simulatieprogramma rekent de ontwikkeling van criteria voor het systeem na.) Er zijn diverse pakketten die helpen bij het formuleren, oplossen en analyseren van het LP-model: MPSX, LINDO enz.

d. Statistiek
Een veelheid aan brongegevens kan worden geanalyseerd via de wiskundige waarschijnlijkheidsrekening en statistiek. Het gaat dan om wetmatigheden (modellen) zoals de normale verdeling en regressiemodellen. De statistiek moet gegevens verwerken tot informatie. Er zijn veel statistische pakketten beschikbaar, voor mainframes en micro's, bijv. SAS, SPSS (zie ook Kleijnen, 1984*b*).

Daarnaast zijn er vele andere modellen, zoals niet-lineaire programmering, netwerk-planning (PERT/CPM), wachttijd-netwerken. Sommige auteurs spreken over een *modellenbank,* d.w.z. in de computer zit een verzameling van modellen waaruit de gebruiker naar believen kiest. Momenteel is het echter nog uiterst moeilijk om programmatuur (voor simulatie, LP, statistiek enz.) te koppelen!

3.2. Gegevens

Modellen moeten gevoed worden met goede gegevens ('garbage in, garbage out'). Terwijl veel wetenschappers zich uitleven in het ontwikkelen van nieuwe algoritmen, blijkt in de praktijk het knelpunt te zijn het gebrek aan gegevens (zie de American Production and Inventory Control Society, APICS, 1974). Dit knelpunt kan worden bestreden door moderne *gegevensbanken.* Het idee is dat alle gegevens over de transacties in de primaire bedrijfsprocessen (zie par. 1) worden opgeslagen in één centraal computergeheugen. Die gegevens worden bij de bron onmiddellijk ingevoerd ['on-line data capture at the source', bijv. 'Point of Sale'-systemen, in plaats van 'batch'-gewijze verwerking]. Tegenwoordig is het technisch mogelijk een zekere decentralisatie te realiseren: micro's worden periodiek (bijv. dagelijks) voorzien van uittreksels (extracten) vanuit de centrale computer (zie Sprague en Carlson, 1982). En bij grote bedrijven kunnen meerdere 'mainframes' gekoppeld worden in een netwerk. Bovendien kan het bedrijf externe gegevens verkrijgen vanuit honderden *publieke* gegevensbanken. De beschikbaarheid van publieke gegevensbanken betekent niet dat er geen problemen meer zijn: enerzijds zijn er te veel externe gegevens; anderzijds zullen de concurrenten hun gegevens afschermen.

Het voorraadbeheer illustreert de relatie tussen modellen en gegevens. Sinds het begin van de twintigste eeuw zijn er allerlei voorraadmodellen ontwikkeld, te beginnen met de klassieke wortel-formule en eindigend (?) bij modellen voor voorraden op verschillende niveaus in een organisatie ('multi-echelon'-systemen) op te lossen via moeilijke technieken zoals Dynamische Programmering. Met de komst van computers ontstond er programmatuur voor het voorraadbeheer geïntegreerd met het produktiebeheer. Daarbij staan niet de modellen centraal, maar de gegevens over stuklijsten

('Bill of Materials', BOM). Deze gegevens zijn opgeslagen in een 'database', beheerd door een Data Base Management Systeem (DBMS). De vraag naar het eindprodukt wordt via die stuklijsten omgerekend in vraag naar de diverse onderdelen ('dependent orders'). Confrontatie met de gegevens over beschikbare voorraden (eindprodukten, tussenprodukten, grondstoffen) leidt tot produktie- en inkooporders: *'Materials Requirements Planning'* of MRP. Een voorbeeld van programmatuur voor MRP is IBM's COPICS.

Met name bij DSS moeten wij onverwachte (ad-hoc-)vragen kunnen beantwoorden (zie ook par. 4). Daarom moeten we gegevens naar allerlei gezichtspunten kunnen terugzoeken ('data retrieval'). Bij een databank zorgt het DBMS ervoor dat gegevens volgens een bepaalde structuur [boom, netwerk, relationeel] worden opgeborgen: IMS, Oracle enz. Dan kunnen wij die gegevens vervolgens terugzoeken via speciale vraagtalen ('query' talen): IMS-DML, SQL enz. De *presentatie* kan gebruiksvriendelijker worden door grafieken, kleuren enz. Ook op micro's is het opslaan, terugvinden, en presenteren van gegevens mogelijk, en wel via 'geïntegreerde' pakketten zoals Lotus 1,2,3 en nieuwe systemen zoals Framework en Symphony.

4. DSS: wat zijn het niet?

DSS zijn systemen en bestaan derhalve uit *sub*systemen (zoals wij weten uit de Algemene Systeem Theorie). Die subsystemen heb ik in de vorige paragraaf beschreven. Wat DSS zijn, is ook nog te verhelderen door aan te geven wat DSS stellig *niet* zijn.
Een subsysteem zonder de andere subsystemen, is geen DSS, maar slechts een bouwsteen of een aanzet tot DSS, bijv., een enkel OR-model is geen volwaardig DSS. En in de inleiding (par. 1) heb ik al een onderscheid gemaakt tussen DSS en EDP.
Het onderscheid met *Management Information Systems* (MIS) is subtieler. Sommigen beweren dat DSS het nieuwe etiket is voor de MIS van de jaren zestig. Anderen zeggen dat MIS gericht zijn op standaardvragen van het management, terwijl DSS *ad-hoc*vragen beantwoorden. Bijvoorbeeld, in het voorraadbeheer moeten wij de vraag 'wanneer bestellen wij en hoeveel?' elke keer opnieuw beantwoorden; dit probleem kunnen wij door een OR-model weergeven; welke invoergegevens voor dit model nodig zijn, is dus bekend; de uitvoergegevens zijn ook standaard. DSS moeten echter onvoorziene vragen beantwoorden, en daarom hebben wij een gegevensbank nodig.
Dit onderscheid tussen MIS en DSS is verwant aan Simons onderscheid tussen gestructureerde (programmeerbare) en ongestructureerde problemen. En de mens kan eerder structuur onderscheiden, indien bepaalde problemen vaker voorkomen. Dus zijn gestructureerde problemen vaak korte-termijnproblemen. Korte-termijnproblemen heten ook operationele, niet strategische problemen (zie Kleijnen, 1980, blz. 6).

Klassieke *Operations Research* modellen zijn optimaliserend en bedoeld voor niet-interactief gebruik. Deze modellen optimaliseren, mits voldaan is aan alle vooronderstellingen (expliciete en impliciete).
Simulatie is niet optimaliserend, maar beantwoordt *what-if*-vragen. Die vragen moeten vanzelfsprekend door de gebruikers worden gesteld. *Interactief* gebruik is moeizaam, indien de computer veel tijd nodig heeft per vraag (d.w.z. per 'simulatie-run'), of indien de computer zijn resultaten niet gebruiksvriendelijk presenteert. Recente computertechnische ontwikkelingen verbeteren het interactief gebruik van simulatiemodellen. Voor de presentatie van simulatiemodellen noem ik animatie, d.w.z. een soort tekenfilm geeft de werking van het gesimuleerde systeem weer.
Computertijd per simulatie-run vermindert drastisch, indien wij supercomputers gebrui-

ken; voor micro's is simulatie vaak te rekenintensief.

OR-modellen kunnen nooit de mens vervangen, want modellen berusten op vooronderstellingen, die door de gebruiker moeten worden getoetst (zie ook Van Hee, 1985).
De uitkomst van een OR-model is dus slechts een *advies* aan de gebruiker, bijv. een bestel-advies. Wel kunnen OR-modellen de gebruiker veel werk uit handen nemen, zodat hij zich op belangrijkere zaken kan concentreren: 'management by exception'.

Modellen (ingewikkelde OR-modellen of simpele spreadsheets) zijn nuttig doordat zij de *intuitie* van de beslisser kunnen verifiëren, en zij bij gebruik door een team van beslissers bijdragen tot de *communicatie* tussen de gebruikers (zie Kleijnen, 1980, 1982). Bij sommige schrijvers bestaat het idee dat de gebruikers kunnen kiezen uit een voorraad modellen (zie de 'modellenbank' in par. 3). In de praktijk bestaan er alleen bouwstenen (programma-modules) voor het modelleren. De praktijk wijst ook uit dat het bouwen van een model al zo bijdraagt tot de communicatie tussen de gebruikers, dat het draaien en analyseren van de modeluitkomsten vaak bijkomstig wordt. [Het bouwen van een 'quick and dirty' model lijkt verwant aan 'prototyping' bij klassieke informatiesystemen.]
DSS worden vaak in één adem genoemd met Expert Systemen en Artificiële Intelligentie.
Wat zijn de verschillen? *Artificiële Intelligentie* (AI) probeert de intelligentie van de mens na te bootsen, in die zin dat niet primair de denkprocessen van de mens worden geïmiteerd, maar zijn denkprodukten d.w.z. zijn uitkomsten. Het bekendste voorbeeld zijn schaakprogramma's. Aanvankelijk meende men dat dank zij het fenomenale rekenvermogen van computers, het schaakprogramma domweg allerlei mogelijke zetten kon narekenen. AI snel bleek dat het aantal combinaties van zetten zo astronomisch groot is, dat zelfs een supercomputer niet genoeg rekencapaciteit heeft.
Daarom is men in de AI toch gaan kijken naar *hoe* de mens problemen oplost, en probeert men tegenwoordig programma's te ontwikkelen die kunnen *leren* (zie ook de volgende paragraaf).

Expert Systemen (ES) zijn een onderdeel van AI, met de volgende kenmerken.
1. ES beperken zich tot een specifiek *domein,* bijv. medische diagnose van infectieziekten (MYCIN is een bekend ES). In AI heeft men geprobeerd een General Problem Solver te ontwikkelen, die dus niet tot een bepaald domein was beperkt.
2. ES hebben verder een gegevensbank met kennis over *feiten* in dat bepaalde domein. Om deze kennis te vergaren moeten deskundigen worden 'uitgemolken'.
3. ES hebben ook een *redeneer-mechanisme* ('inference machine'), meestal een reeks regels in de vorm van 'als..., dan...'. Essentieel is, vind ik, dat ES hun conclusies kunnen verklaren, d.w.z. kunnen uitleggen hoe zij tot bepaalde aanbevelingen zijn gekomen.

Er wordt veel geschreven over ES, maar naar mijn mening zijn ES nog helemaal niet geschikt voor de praktijk.

5. Acceptatie van DSS

Bij mijn weten zijn er nog geen cijfers bekend over de acceptatie van DSS (mede doordat niet duidelijk is wat DSS zijn) (zie ook Liang, 1986). Omdat modellen een belangrijk subsysteem vormen, is het interessant dat er wel een aantal *enquêtes is uitgevoerd naar toepassingen* van *modellen,* in Nederland en in de VS (zie Kleijnen, 1982). Uit die enquêtes blijkt dat men in de praktijk werkt met de volgende technieken.
1. Statistische pakketten voor extrapolaties van cijferreeksen, filteren van omvangrijke gegevensbestanden enz.

2. Simulatie, want simulatiemodellen zijn realistisch en vragen toch weinig wiskundige kennis. Spreadsheets vallen buiten deze enquêtes. Uit andere enquêtes blijkt dat deze programmatuur uiterst populair is op micro's.
3. Eenvoudige OR-technieken, namelijk LP, Economic Order Quantity en PERT/CPM.

Een kosten-batenanalyse van DSS is erg moeilijk. Immers, zo'n analyse is al moeilijk voor klassieke automatiseringsprojecten (zie Kleijnen, 1980, 1984*a*).
In het algemeen verwacht ik dat de nieuwe gebruikersgeneratie computers – en dus DSS – eerder zullen accepteren dan de oude generatie. Een nieuwe *symbiose tussen mens en machine* staat wellicht op stapel. De machine heeft daarbij de volgende inbreng (zie ook Mertens en Kress, 1970).
1. De computer rekent onvoorstelbaar snel [uitgedrukt in MIPS].
2. Hij heeft een onvoorstelbaar groot geheugen [GIGABYTES] vooral geschikt voor de opslag van goed gestructureerde gegevens zoals tabellen.
3. Hij vergeet niks (of alles, bij een calamiteit: 'security' aspect).

De menselijke partner heeft ook een aantal goede eigenschappen.
1. De mens werkt vaak met vuistregels, d.w.z. hij hoeft niet domweg van alles na te rekenen (zoals sommige schaakprogramma's wel doen). De mens hanteert dus heuristiek; de computer hanteert optimaliserende algoritmen zodat hij soms het perfecte antwoord vindt op de verkeerde vraag.
2. De mens heeft een associatief geheugen, d.w.z. hij kan in zijn geheugen informatie terugvinden via de 'gekste verbanden' (wij weten nog niet goed hoe de mens, in zijn enorm groot geheugen, gegevens vastlegt en terugvindt; zie Hofstadter, 1979).
3. De mens kan selectief vergeten; hij kan met name dingen vergeten die niet nuttig bleken: leergedrag.

Uit bovenstaande sterkte-zwakte-analyse van de mens en de machine volgt, dat in de verschillende fasen van een probleem de mens en de computer verschillende rollen spelen. Met Simon onderscheid ik de volgende *probleemfasen.*
1. 'Intelligence': het herkennen in de omgeving van problemen en kansen. Dit is een fase waarin de mens zonder hulp van compters opereert.
2. 'Design': het bedenken van alternatieven. 'Echte' alternatieven zal de mens bedenken; de computer kan kleine, niet-structurele variaties 'bedenken'; bijv. de mens bedenkt in welke gedeelten van de stad bushaltes moeten komen; de computer onderzoekt automatisch kleine verschuivingen (zie Schneider e.a. 1972).
3. Evaluatie: het berekenen van de karakteristieken (criteria en nevenvoorwaarden) van de alternatieven. Hierin is de computer veel beter dan de mens!
4. 'Choice': aangezien de mens verantwoordelijk is, kan het kiezen niet aan de computer worden overgelaten.

6. Conclusie

Een definitie geven van DSS is een hachelijke zaak. Wel is het mogelijk een beschrijving te geven van de subsystemen van DSS, te weten, modellen en gegevens (inclusief het terugzoeken en presenteren van gegevens). Ook is het duidelijk dat DSS, in tegenstelling tot de klassieke EDP, niet gericht zijn op het automatiseren van het (uitvoerend) werk van het kantoorpersoneel, maar op het verwerken van de vele (ex post, primaire) gegevens tot (ex ante) informatie, ten behoeve van management-beslissingen. In de OR ligt het accent op het formuleren van modellen en het afleiden van oplossingsalgoritmen. DSS gebruiken eenvoudige modellen

interactief en voeden die modellen met zinvolle gegevens vanuit de databank. In plaats van te discussiëren over de juiste definitie van DSS, lijkt het beter om computerapparatuur en -programmatuur (incl. modellen) te ontwikkelen die de manager echt ondersteunen bij het nemen van beslissingen.
De opkomst van DSS wordt gestimuleerd door de technische ontwikkeling ('technology push') zoals DBMS en 'on-line' computerverwerking, op grote en kleine computers. De acceptatie van DSS neemt toe, doordat in de verschillende opleidingen (bedrijfseconomie, bedrijfskunde enz.) een grotere rol wordt gespeeld door modellen en computers. Tevens wordt de maatschappij in het algemeen geïnformatiseerd (point-of-sale terminals, elektronisch geld).

Literatuur

- APICS, 'State-of-the-art survey: a preliminary analysis', *Production and Inventory Management,* 15, nr. 4, blz. 1-11.
- Gray, P., 'Using the Interactive Financial Planning System (IFPS) for stochastic simulation', *Simulation,* 43, nr. 6, dec. 1984, blz. 286-292.
- Hee, K. van, 'Informatiesystemen en beslissingsondersteuning', *Informatie,* 27, nr. 11, nov. 1985, blz. 978-986.
- Hofstadter, D. R., *Gödel, Escher, Bach: An eternal golden braid,* Basic Books, Inc., Publishers, New York 1979.
- Keen, P. G. W. en M. S. Scott Morton, *Decision support systems; an organizational perspective,* Addison-Wesley Publishing Company, Reading (Ma.) 1978.
- Kleijnen, J. P. C., *Computers and profits; quantifying financial benefits of information,* Addison-Wesley, Reading (MA.) 1980.
- Kleijnen, J. P. C., *'Modellen en simulatie: een inleiding', Handboek Informatieverzorging,* Alphen a.d. Rijn, Samsom mei 1982.
- Kleijnen, J. P. C., 'Quantifying the benefits of information systems, *European Journal of Operational Research,* 15, nr. 1, jan. 1984*a*, blz. 38-45.
- Kleijnen, J. P. C., 'Statistische analyse: kans op informatie', *Handboek informatieverzorging,* Samsom, Alphen a.d. Rijn nov. 1984*b*.
- Kleijnen, J. P. C. en P. J. Rens, 'IMPACT revisited: a critical analysis of IBM's inventory package 'IMPACT', *Production and Inventory Management,* 19, nr. 1, first quarter, 1978, blz. 71-90.
- Liang, T. P., 'Critical success factors of decision support systems: an experimental study'. *Data Base,* 17, nr. 2, 1986, blz. 3-16.
- Mertens, P. en H. Kress, 'Mensch-Maschinen-Kommunikation als Hilfe bei Entscheidungsvorbereitung und Planung; *Zeitschrift für betriebswirtschaftliche Forschung,* 22, no. 1, 1970, blz. 1-21.
- Schneider, J. B., J. G. Symons en M. Goldman, 'Planning transportation terminal systems in urban regions: a man-machine interactive problem-solving approach; *Transportation Research,* 6, nr. 3, sept. 1972, blz. 257-274.
- Sol, H. G., DSS: buzzword or OR challenge?' *European Journal of Operational Research,* 22, nr. 1, okt. 1985, blz. 1-8.
- Sprague, R. H. en E. D. Carlson, *Building effective decision support systems,* Prentice-Hall, Inc. Englewood Cliffs (New Jersey) 1982.

Een uitgebreidere versie van dit artikel is aan te vragen bij de auteur.