Een prioriteitsstelling op knelpuntsbasis voor de ontwikkeling van informatiesystemen

Rik Veldhuizen

Augustus 1995
Scriptiebegeleider: H.A. Huis in't Veld
Doctoraalscriptie Bedrijfsgerichte Informatica
Katholieke Universiteit Nijmegen
Scriptie-n° 349

Een prioriteitsstelling op knelpuntsbasis voor de ontwikkeling van informatiesystemen

Rik Veldhuizen

Inhoudsopgave

Inleiding.....4

Hoofdstuk 1 - Methoden Voor Informatieplanning.....6
Inleiding.....6
1.1 Information Strategy Planning.....7
1.1.1 Inleiding.....7
1.1.2 De afbakening.....8
1.1.3 De prioriteitsstelling.....13
1.1.4 Conclusie.....14
1.2 Handboek Informatieplanning.....16
1.2.1 Inleiding.....16
1.2.2 De afbakening.....17
1.2.3 De prioriteitsstelling.....20
1.2.4 Conclusie.....22
1.3 System Development Methodology.....23
1.3.1 Inleiding.....23
1.3.2 De afbakening.....24
1.3.3 De prioriteitsstelling.....26
1.3.4 Conclusie.....28
1.4 ARTIST.....29
1.4.1 Inleiding.....29
1.4.2 De afbakening.....30
1.4.3 De prioriteitsstelling.....33
1.4.4 Conclusie.....34
Conclusie.....36

Hoofdstuk 2 - De Prioriteitsstelling.....37
Inleiding.....37
2.1 De opbouw van de theorie.....39
2.2 Algoritmen voor de prioriteitsstelling.....49
2.2.1 De verzamelingen.....49
2.2.2 De functies.....51
2.2.3 De algoritmen.....55
2.3 Een knelpunten-matrix bij het algoritme.....68
2.4 Een voorbeeld uitgewerkt.....71
Conclusie.....75

Hoofdstuk 3 - Workflow Management.....76
Inleiding.....76
3.1 Workflow Management nader bekeken.....76
3.2 Een vergelijking.....80
Conclusie.....81
Bijlage I.....niet aanwezig (te groot voor postscript)
Bijlage II.....82
Verwijzingen.....86
Bibliografie.....87

Inleiding

Business Process Redesign, Strategic Alignment en Informatieplanning zijn alle activiteiten waarbij binnen een organisatie bedrijfsprocessen onder de loep worden genomen en mogelijk worden herontworpen. Een van de achterliggende bedoelingen daarbij is na te gaan of en in hoeverre de toepassing van Informatie Technologie ondersteunend kan zijn aan deze bedrijfsprocessen. De feitelijke ondersteuning komt dan van informatiesystemen. In hun functionele betekenis, dus zonder direkt aan computers en netwerken te denken, worden informatiesystemen onder andere bij informatieplanning bepaald. Bij het analyseren van de organisatie kan de behoefte aan informatiesystemen naar voren komen: er is een informatiesysteem nodig voor de marketing dat op voorhand inzicht geeft in bepaalde zaken; er is een informatiesysteem nodig zodat het onderhoud van het wagenpark beter geregeld wordt; er is een grote behoefte aan een informatiesysteem waarmee de orderafhandeling geregeld wordt; etc. De informatieplanner zal een grote lijst met dit soort behoeften opstellen. Daarbij zal niet direkt gelet worden op de haalbaarheid van deze behoeften, die onder meer bepaald wordt door middelenschaarste, organisatorische tegenwerking en het niet beschikken over voldoende IT-deskundigheid. Nadat de informatieplanner de theoretische behoefte aan informatiesystemen in kaart heeft gebracht, zal immers toch de selectiefase komen waarin wordt besloten welke systemen wel kunnen worden ontwikkeld of aangeschaft en welke niet. Dit vanwege het feit dat er altijd en overal sprake is van financiële en organisatorische beperkingen.

Maar wie selecteert er en welke criteria worden gehanteerd? Waarom het ene systeem wel en het andere niet? Is de ontwikkeling of aanschaf van een marketing information system belangrijker dan het snel kunnen beschikken over een goed human resource mangement system? Voor de marketing manager wel, voor het hoofd P&O niet. De ervaring leert dat hier veel subjectieve meningen in het spel zijn, hetgeen nog wel eens uitmondt in het onterecht stellen van persoonlijke voorkeuren. Vele beslissingen worden gevoelsmatig genomen. Maar al te vaak zijn genomen beslissingen afgedwaald van het doel: optimale inpassing van IT voor ondersteuning aan bedrijfsprocessen in de organisatie. Omdat de meest bekende onderzoeks- en planningsmethoden tekort schieten bij de toekenning van prioriteiten aan de ontwikkeling van informatiesystemen, rijst de vraag of er geen formele beslissingscriteria kunnen worden gevonden.

Veel van de binnen de organisatie gebruikte informatiesystemen hangen met elkaar samen, al is het alleen vanwege het veelal gebruiken van dezelfde gegevens. Maar ook zullen bepaalde informatiesystemen niet of niet goed kunnen functioneren wanneer zij afhankelijk zijn van de

uitvoer van ‘voorliggende’ systemen. Zo leidt het gebruik van een klantinformatiesysteem door de marketingafdeling tot een averechtse werking wanneer het voor zijn voeding afhankelijk is van inaccurate, niet actuele of onvolledige informatie die wordt afgegeven door een slecht werkend order- of factureringssysteem. Het zijn deze onderlinge verbanden en beïnvloedingen waarnaar ik in deze scriptie onderzoek heb gedaan. Daarbij is gebleken dat door de gebruikers gesignaleerde knelpunten tot op zekere hoogte gebruikt kunnen worden voor de toekenning van prioriteiten aan de ontwikkeling van informatiesystemen. Door gebruik te maken van deze kennis wordt enerzijds volledige willekeur bij prioriteitsbeslissingen ingedamd en worden anderzijds onterecht uitgesproken persoonlijke voorkeuren min of meer aanwijsbaar teruggedrongen. Gevoelsmatigheid zal er altijd in een zekere mate bijhoren.

Het doel van mijn scriptie is het vinden van een methode voor de prioriteitstoekenning op knelpuntsbasis, waarbij subjectiviteit zoveel mogelijk wordt teruggedrongen. Hiertoe heb ik eerst een viertal veel gebruikte informatieplanningsmethoden nader bekeken, daarbij vooral lettend op de manier van afbakening van informatiesystemen en het toekennen van prioriteiten. Vervolgens breid ik één van deze bestaande methoden uit met een methode voor het toekennen van prioriteiten. Deze is gebaseerd op de bestaande onderlinge verbanden tussen knelpunten en informatiesystemen. Als laatste maak ik nog een kleine zijstap naar het momenteel zeer in zwang zijnde workflow management, waarbij ik beschrijf wat workflow management is en wat de relatie is met mijn ontwikkelde methode.

Tot slot wil ik graag een aantal mensen bedanken, zonder wie deze scriptie nooit deze vorm gehad zou hebben: Herman Huis in’t Veld, voor de aanzet en de enthousiaste begeleiding; dr ir P.W.P.J. Grefen, voor de informatie over workflow management; mijn ouders, zonder wie ik deze studie niet had kunnen doen en vooral Claudia, die mij altijd met veel geduld geholpen en gesteund heeft. Mijn dank voor haar is grenzeloos.

Hoofdstuk 1
Methoden Voor Informatieplanning

Inleiding

In dit hoofdstuk worden vier methoden voor informatieplanning nader bekeken, te weten ISP, HIP, SDM en ARTIST. Ik kijk bij deze methoden vooral naar de afbakening en de prioriteitsstelling, omdat deze aspecten voor mijn onderwerp het belangrijkste zijn. Zonder al te gedetailleerd te worden, zal beschreven worden hoe elk van de methoden de afbakening en de prioriteitsstelling van informatiesystemen aanpakt. Ter afsluiting van elke paragraaf zal mijn eigen mening gegeven worden.

Het doel van dit hoofdstuk is voornamelijk het verkrijgen van inzicht in de wijze waarop verscheidene, veelgebruikte methoden voor informatieplanning de planning van de ontwikkeling van informatiesystemen beschrijven. Er zal blijken dat eigenlijk geen enkele van de beschreven methoden echt voldoet.

Een opmerking dient gemaakt te worden over de gebruikte benamingen bij het beschrijven van de methoden. Elke methode kent z’n eigen definities en soms wordt met hetzelfde woord bij de ene methode iets heel anders bedoeld dan bij de andere methode. Belangrijke benamingen worden binnen de methode gedefinieerd en gelden alleen binnen die omgeving. Dit zal duidelijk aangegeven worden.

Als laatste paragraaf van dit hoofdstuk wordt een conclusie gegeven. Hierin worden elementen weergegeven van de bij de paragrafen gegeven conclusies, zonder opnieuw elke methode te beoordelen op de sterke en zwakke kanten. Daarvoor verwijs ik naar de desbetreffende conclusies.

De laatst beschreven methode, de methode ARTIST, vormt de basis van de prioriteitsstelling uit hoofdstuk 2. Voor een goed begrip van de inhoud van het volgend hoofdstuk dient deze methode bekend te zijn.

1.1 Information Strategy Planning

1.1.1 Inleiding

De methode Information Strategy Planning, kortweg ISP, is grotendeels gebaseerd op het welbekende BSP, IBM's Business Systems Planning (zie bijvoorbeeld [SPR1986] en [JMS1982]). Bij het laatste echter werden door James Martin ([JMS1982]) enkele tekortkomingen geconstateerd, zoals bijvoorbeeld het tekortschieten in een goede ondersteuning van gegevensverzamelingen. Hiervoor is binnen James Martin Associates een ontwikkeling gaande geweest om een nieuwe methode te creëren, die in de praktijk zou voldoen. Sinds 1983 is deze methode bekend onder de naam Information Strategy Planning. Zij maakt deel uit van The Information Engineering Methodology (TM), een 'complete methode voor de ontwikkeling van informatiesystemen vanaf een strategisch plan tot en met de realisatie van operationele systemen' ([BUS1987], p.228) en is daarvan de eerste fase. ISP dient als hulpmiddel bij het bepalen van een richting die de organisatie zal inslaan ten aanzien van informatievoorziening en automatisering (strategieplanning), bij het definiëren van een globaal ontwerp van de toekomstige informatievoorziening (architectuurplanning) en bij het creëren van een plan ter realisatie. Hierna kan de tweede fase van TM, zijnde Business Area Analysis, het genoemde inzicht verder uitwerken.[1]

Het gaat bij ISP vooral om het bepalen van een viertal globale ontwerpen (architecturen), die tezamen de toekomstige informatievoorziening vormen. Deze zijn ([BUS1987], p.229):

- informatie-architectuur (samenhang tussen ondernemingsactiviteiten en gegevens)
- informatiesysteem-architectuur (informatiesystemen, databanken en de relatie daartussen)
- technische architectuur (infrastructuur van technische voorzieningen)
- organisatie-architectuur (gevolgen voor de ondernemingsorganisatie en de structuur van de informatiemanagement functie)

Het bepalen van deze architecturen wordt binnen ISP gezien als een aparte fase. In totaal zijn er vijf fasen binnen ISP te onderscheiden ([BUS1987], p.229):

1. planning van ISP-project.
2. analyse van bestaande plannen, strategieën en beleidslijnen.
3. bepalen van eisen aan de informatievoorziening vanuit het gezichtspunt van de ondernemingsstrategie.
4. bepalen van de vier architecturen.
5. bepalen van de strategie en planning voor de overgang van de huidige naar de gewenste situatie.

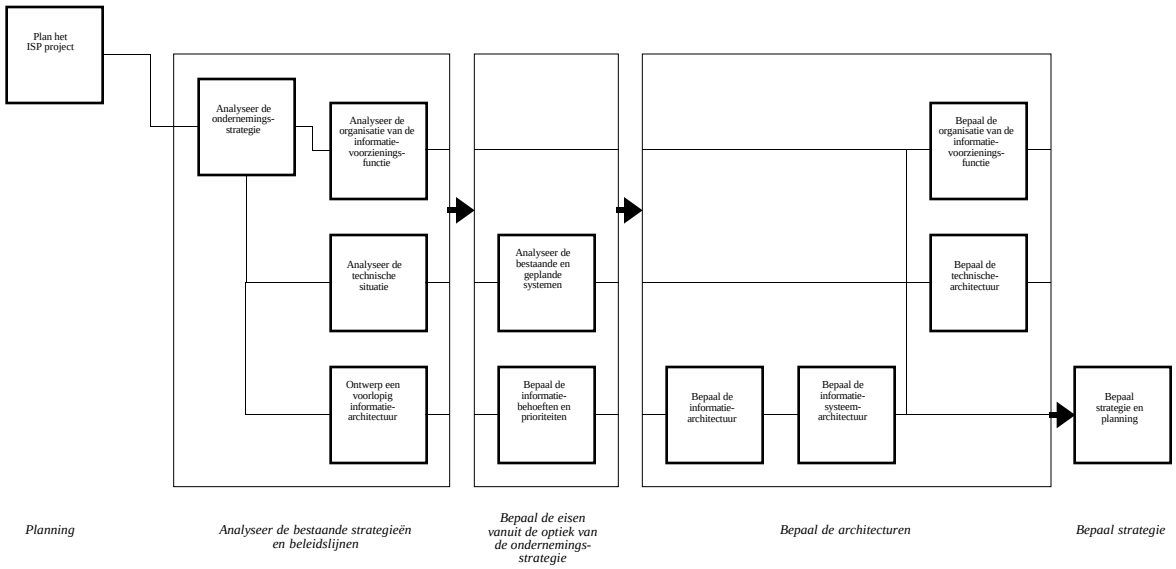
---

1 Zie voor een uitgebreid verslag van TM [MAC1986].

Binnen deze fasen is dan weer een aantal taken te onderscheiden. Figuur 1.1 geeft dit weer. In deze paragraaf zal niet elke fase of taak apart beschreven worden. Alleen de meest relevante aspecten van een ISP-project worden uitvoerig besproken. De overige aspecten worden slechts kort vermeld. Voor een uitgebreide beschrijving van ISP verwijs ik naar [BUS1987].

1.1.2 De afbakening

De afbakening van de te ontwikkelen informatiesystemen gebeurt pas in fase vier (bepaal de architecturen). De benodigde informatie hiervoor komt uit de voorgaande fasen. Uit de eerste fase (planning) volgen zaken als de team-samenstelling en de structuur van de rapportage; voor het project belangrijk, maar voor de afbakening niet. De tweede fase, analyseer de bestaande strategieën en beleidslijnen, is bedoeld om ‘de juiste uitgangspunten te definiëren voor het vaststellen van informatie-eisen en de daarvoor in te richten informatievoorziening’ ([BUS1987], p.230). Uit figuur 1.1 blijkt dat deze fase bestaat uit vier taken. De eerste drie, de analyseer-taken, zijn vooral bedoeld om een uitgangspunt te creëren voor de toekomstige informatievoorziening. Hierbij worden de doelstellingen en de strategie van de onderneming geanalyseerd, bijvoorbeeld door middel van interviews met het top-management. Dus vragen als ‘wat wil de onderneming bereiken met de toekomstige informatievoorziening?’ en ‘wat zijn de voorziene problemen of de kritieke succes factoren hierbij?’ dienen hier beantwoord te worden. Bij ISP worden kritieke



Figuur 1.1 - De ISP-taakstructuur ([BUS1987], p.230)

succes factoren (Critical Success Factors) overigens gedefinieerd als ‘factoren die een belangrijke invloed zullen hebben op het vermogen van de onderneming om haar doelstelling te bereiken’ ([BUS1987], p.238). In deze fase wordt ook onderzocht welke personen of groepen personen aan de huidige informatievoorziening gerelateerd kunnen worden. Zo zal bijvoorbeeld blijken dat het Informatiesturingscomité verantwoordelijk is voor de ontwikkelingscoördinatie van de informatievoorziening (voorbeeld uit [BUS1987], p.231). Een andere taak is het analyseren van de huidige technische voorzieningen binnen de organisatie en van het ondernemingsbeleid ten aanzien van de technologische ontwikkelingen. In de vierde taak, het ontwerpen van een voorlopige informatie-architectuur, worden de aandachtsgebieden en de bedrijfsfuncties bepaald (een bedrijfsfunctie wordt gedefinieerd als een groep bedrijfsactiviteiten die tezamen één aspect van het nastreven door de onderneming van haar doelstelling volledig ondersteunen ([BUS1987], p.238)). Hierbij wordt gekeken wie bij welke bedrijfsfuncties betrokken zijn en deze taak bepaalt daarmee de keuze voor personen die geïnterviewd moeten worden met betrekking tot de informatievoorziening.

De derde fase, bepaal de eisen vanuit de optiek van de ondernemingsstrategie, vormt de eigenlijke basis voor de toekomstige informatievoorziening. Men wil vaststellen aan welke eisen de toekomstige informatievoorziening moet voldoen en in hoeverre reeds bestaande faciliteiten aan deze eisen voldoen ([BUS1987], p.233). Zo wordt de in de voorgaande fase ontworpen voorlopige informatie-architectuur gebruikt om een relatie te leggen tussen de huidige en de toekomstige situatie. Hierbij worden de huidige gegevensverzamelingen en informatie-systemen vergeleken met de toekomstige. Men moet natuurlijk bedenken dat de voorspelde informatie-architectuur slechts een globale indruk is. Ook hier zullen weer interviews gehouden worden om de bedrijfsfuncties goed in kaart te brengen, tezamen met de daarbij gestelde informatie-eisen en bijbehorende prioriteiten. Hier kom ik later nog op terug.

In fase vier worden dan de systemen afgebakend en wel in de eerste twee taken, het bepalen van de informatie- en informatiesysteem-architectuur. De overige twee taken worden slechts kort toegelicht, omdat ze voor mijn aandachtsgebied niet uitvoerig beschreven hoeven worden.

Bepaal de informatie-architectuur

Zoals reeds vermeld werden bij het opstellen van de voorlopige informatie-architectuur bedrijfsfuncties en aandachtsgebieden onderkend. Deze zullen hier verder uitgewerkt worden. De aandachtsgebieden worden in entiteittypen (een entiteittype wordt gedefinieerd als een verzameling entiteiten waarvoor een bepaalde definitie en algemene eigenschappen gelden ([BUS1987], p.238)) opgedeeld en de bedrijfsfuncties worden top-down geanalyseerd tot op het niveau van bedrijfsprocessen (een bedrijfsproces wordt gedefinieerd als een bedrijfsactiviteit waarvan de uitvoering geïdentificeerd kan worden in termen van input en/of output van entiteiten van bepaalde

C = create / modify
U = use

bedrijfs functie	entiteittype / bedrijfs proces	e1	e2	e3	e4	e5	e6	e7
f1	p1	U				C	C	C
	p2			U		U		
	p3		C					
	p4				U			
f2	p5	C					U	
	p6			C			U	
	p7				C			U

Figuur 1.2 - Matrix entiteittypen en bedrijfsprocessen

typen of van gegevens over entiteiten van bepaalde typen ([BUS1987], p.238)). Als er belangrijke relaties bestaan tussen verschillende entiteittypen, dan moeten deze weergegeven worden. De volgende stap is het onderling relateren van de bedrijfsfuncties en -processen. Hierbij moet analyseerd worden welke informatie door een bepaalde bedrijfsfunctie of -proces aan een andere bedrijfsfunctie of -proces geleverd wordt. Tevens dient de relatie tussen de verschillende bedrijfsfuncties en -processen en de entiteittypen zichtbaar gemaakt worden. In figuur 1.2 is een matrix te zien met daarin de relatie tussen bedrijfsprocessen en entiteitypen. Hierin is ook goed te zien dat bedrijfsfuncties opgedeeld kunnen worden in bedrijfsprocessen. Een *C* betekent dat een bedrijfsproces een bepaald entiteittype kan creëren, verwijderen en/of wijzigen. Een *U* staat voor gebruiken. Een entiteittype moet in ieder geval door één bedrijfsproces gecreëerd worden en elke bedrijfsproces dient bij tenminste één entiteittype betrokken te zijn. Tevens wordt gekeken of elk entiteittype nog door een ander proces dan het creërende proces gebruikt wordt (in figuur 1.2 voldoet entiteittype *e2* hier niet aan). Nu vereist ISP nog dat de betrokkenheid van de verschillende afdelingen met de entiteittypen en bedrijfsfuncties en -processen geanalyseerd wordt. Onderling vergelijk van de uitkomst van de verschillende analyses moet leiden tot consistente informatie. Als bijvoorbeeld blijkt dat een bepaalde afdeling betrokken is bij een bepaald entiteittype, dan dient deze afdeling ook betrokken te zijn bij het bedrijfsproces dat het entiteittype creëert.

Bepaal de informatiesysteem-architectuur

Op basis van de uitkomsten van de vorige taak worden nu informatiesystemen en gegevensverzamelingen afgebakend. Er wordt in [BUS1987] slechts summier vermeld hoe dit dient te geschieden. Zo wordt vermeld dat informatiesystemen worden gedefinieerd die ‘één of meer samenhangende bedrijfsfuncties of -processen van informatie voorzien’ ([BUS1987], p.234) en gegevensverzamelingen groepen van ‘sterk gerelateerde entiteittypen’ ([BUS1987], p.234) moeten zijn. Dit alles zou moeten gebeuren door middel van ‘een grondige analyse van de informatie-architectuur met als doel de omvang en complexiteit van de gegevensstromen te minimaliseren’ ([BUS1987], p.234), waarna tevens de informatiesystemen en de gegevensverzamelingen geanalyseerd moeten worden. Gezien het feit dat ISP grotendeels gebaseerd is op BSP, mag men aannemen dat het definiëren en analyseren zoals hierboven is

beschreven op de BSP-manier gaat. Echter, ISP claimt een verbetering ten opzichte van BSP bij het definiëren van gegevensverzamelingen. In [BUS1987] wordt dat echter niet beschreven, maar [JMS1982] beschrijft een nauwkeurige methode. Gaarne verwijs ik daarnaar ([JMS1982], H.7 & H.8), er van uitgaande dat dit de bedoelde verbetering is. Hier zal ik verder niet op ingaan. Hoewel het definiëren van gegevensverzamelingen summier beschreven wordt bij BSP, wordt wel stap voor stap beschreven hoe uit een matrix als in figuur 1.2 informatiesystemen afgeleid kunnen worden ([JMS1982], H.5 & H.6). De eerste stap is de bedrijfsprocessen (op dit niveau wordt niet met bedrijfsfuncties gewerkt, maar met de niet meer verder op te delen bedrijfsprocessen) in de juiste volgorde in de matrix-rijen te zetten en wel zo, dat van boven naar beneden het verloop van een produkt te lezen is. Zo zal het eerste proces bijvoorbeeld een markt-analyse kunnen zijn, terwijl een van de laatste processen in de matrix een transport-proces kan zijn. In de tweede stap worden de entiteittypen die door het eerste bedrijfsproces gecreëerd worden naar links geschoven (de gehele entiteittype-kolom). Natuurlijk alleen als het eerste proces een entiteittype creëert. Zo worden achter elkaar de processen afgelopen en waar nodig worden entiteittype-kolommen naar links geschoven. Voor de matrix uit figuur 1.2 zal dit resulteren in de matrix als in figuur 1.3. Nu kunnen in de volgende stap al de grenzen van de toekomstige informatiesystemen aangegeven worden door processen en entiteittypen te groeperen als in figuur 1.4a. Er wordt toegegeven dat dit enige willekeur met zich meebrengt ([JMS1982], p.69). Zo valt proces *p4* een beetje buiten de boot. Dit proces zou zowel bij het tweede

bedrijfs proces \ entiteittype	e5	e6	e7	e2	e1	e3	e4
p1	C	C	C		U		
p2	U					U	
p3				C			
p4							U
p5		U			C		
p6		U				C	
p7			U				C

Figuur 1.3 - Matrix na de tweede stap

bedrijfs proces \ entiteittype	e5	e6	e7	e2	e1	e3	e4
p1	C	C	C		U		
p2	U					U	
p3				C			
p4							U
p5		U			C		
p6		U				C	
p7			U				C

Figuur 1.4a - Matrix na de derde stap

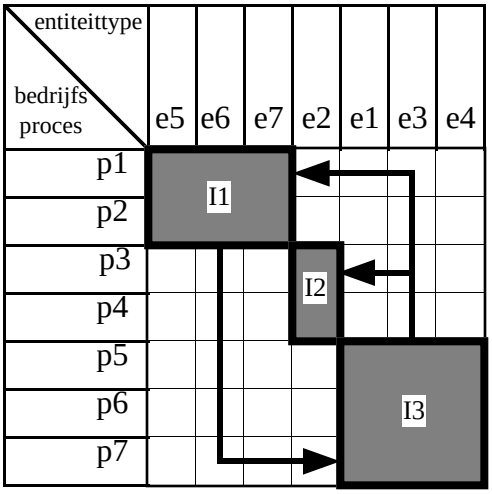
bedrijfs proces \ entiteittype	e5	e6	e7	e2	e1	e3	e4
p1	C	C	C		U		
p2	U					U	
p3				C			
p4							U
p5		U			C		
p6		U				C	
p7			U				C

Figuur 1.4b - Mogelijke afbakening

informatiesysteem (samen met proces *p3*) als bij het derde informatiesysteem (samen met processen *p5*, *p6* en *p7*) kunnen horen. Hier lijkt het derde informatiesysteem logischer, omdat proces *p4* entiteittype *e4* gebruikt en die wordt in het derde informatiesysteem gecreëerd. Deze mogelijke afbakening is zichtbaar in figuur 1.4b. Ik heb proces *p4* echter bij het tweede informatiesysteem laten horen. Figuur 1.4a dient slechts ter illustratie.

We hebben nu een aardig overzicht van de afbakening van informatiesystemen. In het voorbeeld zijn drie potentiële informatiesystemen afgebakend. De *U*'s die nog buiten de aangegeven informatiesystemen vallen, duiden op dataflow van het ene informatiesysteem naar het andere. Zo betekent de *U* op positie {*p5*, *e6*} dat er een dataflow bestaat van het eerste informatiesysteem (daar waar *e6* gecreëerd wordt) naar het derde (aan welke *p5* toebehoort). Een dataflow heeft dus als beginpunt het informatiesysteem waar het desbetreffende entiteittype gecreëerd wordt en als eindpunt het informatiesysteem waarin het proces zich bevindt die het entiteittype gebruikt.

Als laatste stap kunnen nu de informatiesystemen benoemd worden, samen met de verschillende dataflows. Zie hiervoor figuur 1.5. De pijlen duiden slechts op het feit dat er dataflows bestaan van het ene informatiesysteem naar het andere; zij zeggen niets over de inhoud van de dataflows. Hiermee sluit ik de BSP-sessie af.



Figuur 1.5 - Matrix na de laatste stap

ISP gaat in deze taak overigens wel diep in op 'het gewenste gebruik van informatiesystemen en gegevensverzamelingen op de bedrijfslokaties' ([BUS1987], p.234). Hierbij worden twee matrices gecreëerd, waarin verschillende mogelijkheden van informatiesysteem- of gegevensverzameling-gebruik door de bedrijfslokaties worden aangegeven. Zo kunnen er van een bepaald informatiesysteem verschillende versies gecreëerd worden, die op verschillende bedrijfslokaties gebruikt worden. Dit zal hier niet bekeken worden.

De laatste twee taken van deze fase bepalen de technische architectuur en de organisatie van de informatievoorziening. Hierin worden de mogelijke technische architectuur en de veranderingen voor de organisatie bepaald. Daarop zal niet verder worden ingaan.

1.1.3 De prioriteitsstelling

De laatste fase van het ISP-project is, zoals reeds eerder vermeld, het bepalen van de strategie en planning voor de overgang van de huidige naar de gewenste situatie. Hierbij wordt een aantal alternatieve overgangen gepresenteerd, die geëvalueerd worden op grond van de baten voor de onderneming en de succesfactoren ([BUS1987], p.234). Men kan hier echter niet van een prioriteitsstelling ten aanzien van de te ontwikkelen informatiesystemen spreken, aangezien de aandacht zich te weinig op de individuele informatiesystemen richt. Er is echter eerder in het project al sprake van het geven van prioriteiten aan informatiebehoeften. Door middel van interviews worden prioriteiten bepaald die gesteld worden aan informatieondersteuning bij de verschillende bedrijfsfuncties. Er wordt echter niet verteld hoe dit precies in zijn werk zou moeten gaan. We mogen aannemen, wederom vanwege het feit dat ISP grotendeels op BSP is gebaseerd, dat ISP op dit punt vergelijkbaar is met BSP en dezelfde methode gebruikt. BSP bepaalt een prioriteitsstelling met behulp van een aantal criteria. Deze zijn in vier categorieën ondergebracht en het is de bedoeling dat voor elk afgebakend informatiesysteem een waarde aan elke categorie wordt toegekend ([JMS1982], p.98), variërend van één tot en met tien. De categorieën zijn als volgt ([JMS1982], p.97-98):

1. Potentiële voordelen
   - tastbaar
   - niet tastbaar
   - opbrengsten van investeringen

2. Invloed op organisatie
   - aantal getroffen organisaties en personen
   - kwalitatieve effect
   - effect op het volbrengen van doelstellingen

3. Aannemelijk succes
   - mate van zakelijke acceptatie
   - waarschijnlijkheid van volledige implementatie
   - eerste vereisten
   - duur van implementatie
   - risico
   - beschikbare hulpbronnen

4. Eisen

waarde van bestaande systemen
relatie met bestaande systemen
politieke bijkomstigheden
noodzaak

Aan de hand hiervan zal het top-management moeten kunnen besluiten welke informatiesystemen wel en welke niet ontwikkeld zullen worden. Ik zal de verschillende criteria niet verder toelichten, ondanks het feit dat zulke vage termen vele vragen kunnen oproepen bij toepassing in de praktijk.

1.1.4 Conclusie

Net als bij BSP wordt het top-management bij veel zaken direkt betrokken, tenminste, dat claimt de methode. De ervaring leert dat het top-management het meestal te druk heeft om echt actief bij de ontwikkeling betrokken te zijn. Het middenkader van de organisatie wordt volgens [BUS1987] ook goed bij het project betrokken (onderdeel van het ISP-team, analyse van informatie-eisen). Wat ik echter mis is de betrokkenheid van de eigenlijke eindgebruikers, de mensen op de werkvloer. Zij lopen het risico aan het eind van het project een systeem onder ogen te krijgen waaraan zij zelf niet hebben meegewerkt, terwijl juist die mensen heel goed weten wat er precies in de organisatie gebeurt en daarom van grote waarde zijn voor de informatieplanners. Interviews met de mensen op de werkvloer is daarom aanbevelenswaardig.

Tevens wordt de prioriteitsstelling niet voldoende belicht. Er moet maar vanuit gegaan worden dat de BSP-methode op dat punt weer bijspringt. Gesteld dat de BSP-methode hierbij bedoeld wordt, dan valt daar op aan te merken dat het toch moeilijk zal zijn de juiste waarden toe te kennen. Er wordt gesteld dat de prioriteiten aan de hand van interviews bepaald worden, maar een ieder vindt het informatiesysteem dat voor zijn of haar afdeling/omgeving geschikt is het belangrijkste. Voor een objectieve prioriteitsstelling is deze methode niet geschikt.

Het afbakenen van de informatiesystemen gebeurt mijns inziens op een redelijk arbitraire wijze en vooral deze willekeur is een tekortkoming. De methode van afbakening is echter wel geschikt voor algoritmische bewerkingen en daardoor een basis voor automatische ondersteuning van ISP, als hulpmiddeltje bij het informatieplannings-project. Wat ik mis, is het niet toekennen van actualiteitswaarden aan verschillende informatiesystemen. Misschien worden er nu wel gegevensverzamelingen en processen in één informatiesysteem gegroepeerd, waarvan sommige elke minuut gebruikt worden en andere een keer in het jaar. Onderdelen van informatiesystemen met uiteenlopende actualiteitswaarden kunnen beter niet in hetzelfde informatiesysteem gestopt worden.

Als ISP uitgewerkt zal worden op de onvolledige punten, dan lijkt het mij een goede informatieplannings-methode. Op het gebied van de prioriteitsstelling dienen verbeteringen aangebracht te worden en op het gebied van de afbakening zal men nauwkeuriger te werk moeten gaan. Het is jammer dat de presentatie van ISP, namelijk [BUS1987], te weinig gedetailleerd is en dat er veel zaken bijgezocht moeten worden.

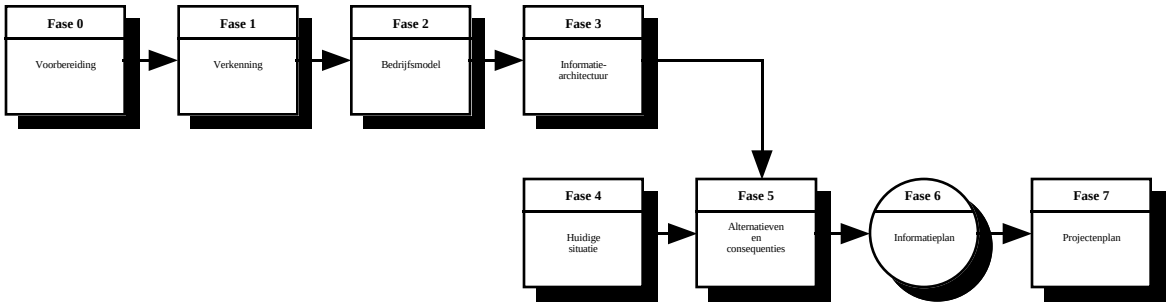
1.2 Handboek Informatieplanning

1.2.1 Inleiding

Het Handboek Informatieplanning (kortweg HIP) is gebaseerd op de ervaringen die zijn opgedaan bij verschillende informatieplanningsprojecten in het Ministerie van Defensie ([AAR1989], p.5). De beschreven methode dient als gids bij het opstellen van een informatieplan, waarbij getracht wordt zoveel mogelijk aan te sluiten bij bestaande systeem-ontwikkelingsmethodieken, zoals bijvoorbeeld System Development Methodology (SDM). HIP is vooral bedoeld voor het management; zij ziet informatieplanning als een instrument voor het management om 'informatievoorziening beheersbaar te maken in termen van tijd, kwaliteit en kosten' ([AAR1989], p.9).

Informatieplanning met behulp van HIP verloopt via acht fasen. Deze zijn ([AAR1989], p.29):
0. Voorbereiding.
1. Verkenning.
2. Bedrijfsmodel.
3. Informatie-architectuur.
4. Huidige situatie.
5. Alternatieven en consequenties.
6. Afronden informatieplan.
7. Projectenplan.

Figuur 1.6 geeft deze fasen schematisch weer. Ik zal niet alle fasen toelichten. Mijn blikveld richt zich op fase 3 (het vierde element uit figuur 1.6), waarin het afbakenen van informatiesystemen aan de orde komt, en op fase 5 (het zesde element uit figuur 1.6), waarvan de prioriteitsstelling een onderdeel is. Wanneer informatie uit voorgaande fasen nodig is, zal dit aangegeven worden. Hoe



Figuur 1.6 - Fasen van informatieplanning ([AAR1989], p.29)

die specifieke informatie tot stand is gekomen, zal niet of slechts summier beschreven worden. Voor een gedetailleerde beschrijving van HIP verwijs ik naar [AAR1989]. Overigens zal veelvuldig naar de vorige paragraaf verwezen worden, direkt of indirekt.

1.2.2 De afbakening

In de voorgaande fasen van HIP, fase 0 tot en met fase 2, zijn aspecten gedefinieerd die belangrijk zijn voor de afbakening van informatiesystemen, namelijk de processen, de entiteittypen en de verantwoordelijkheden. Deze drie aspecten worden met behulp van matrices onderling gerelateerd, hetgeen een basis moet zijn voor de afbakening. Een grondige analyse van de matrices moet leiden tot consistente informatie. Dit wordt nog uitgelegd.

De eerste stap is het creëren van een zogenaamde P/G-matrix ([AAR1989], p.310), ofwel het onderling relateren van de processen (P) en de entiteittypen (G van gegevens, dat bij HIP staat voor entiteittypen). Een proces wordt bij HIP gedefinieerd als ‘Een reeks van activiteiten en de beheersing (planning en controle) daarvan, gericht op een bepaald doel’ ([AAR1989], p.424) en een entiteittype wordt gedefinieerd als ‘een afzonderlijk herkenbare en wezenlijke eenheid van informatie betrekking hebbende op (een deel van) een object- of associatietype’ ([AAR1989], p.427). Dit laatste klinkt moeilijk, maar een objecttype is niets meer dan een karakteristieke beschrijving van een aantal elementen uit de werkelijkheid, die aan een aantal eigenschappen voldoen. Een associatietype is een karakteristieke beschrijving van een aantal associaties, waarbij deze associaties aan een aantal specifieke eigenschappen voldoen. Associaties kunnen tussen twee of meer objecten bestaan, of tussen objecten en andere associaties ([AAR1989], p.427). De P/G-matrix vertoont veel overeenkomsten met de matrices zoals gebruikt bij ISP. Hieruit blijkt dat ook HIP deels op BSP gebaseerd is. Een P/G-matrix ziet eruit als de matrix uit figuur 1.2, waarbij de daargenoemde bedrijfsfuncties buiten beschouwing gelaten dienen te worden. De volgorde van de processen wordt op dezelfde manier bepaald als bij ISP, namelijk in de volgorde van het verloop van een produkt. Het invullen van de matrix gebeurt ook hier op de Create/Use manier, maar hier staat een *C* slechts voor het creëren van een entiteittype en een *U* voor zowel het wijzigen als het gebruiken (lezen) van een entiteittype door een proces. Een extra voorwaarde die door HIP gesteld wordt aan het invullen van de matrix is dat

entiteittype / bedrijfs proces	e5	e6	e7	e1	e3	e2	e4
p1	C	C	C	U			
p5		U		C			
p2	U				U		
p6		U			C		
p3						C	
p4							U
p7			U				C

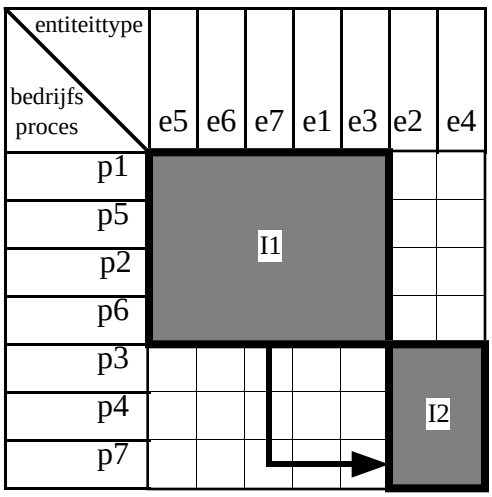
Figuur 1.7 - Matrix na de rangschikking

een entiteittype slechts door één proces gecreëerd mag worden ([AAR1989], p.311).
De transformatie van de P/G-matrix naar een matrix waarin de verschillende informatiesystemen zich reeds afbakenen (vergelijk transformatie van figuur 1.2 naar figuur 1.3), gebeurt met behulp van de volgende stappen ([AAR1989], p.312):

- Het rangschikken van de entiteittypen door het zodanig verschuiven van de kolommen, dat de *C*'s en de *U*'s diagonaal komen te staan, beginnend bij het eerste proces.
- Het rangschikken van de bedrijfsprocessen door het zodanig verschuiven van de rijen, dat de buiten de diagonaal vallende *C*'s en *U*'s nog zo goed mogelijk op de diagonaal komen. Dit is slechts tot op zekere hoogte mogelijk.

Voor de matrix uit figuur 1.2 zou dit kunnen resulteren in de matrix uit figuur 1.7. Nu is het 'zou kunnen' natuurlijk een slechte zaak en duidt dan ook op de mogelijke willekeur die hanteerbaar is bij deze methode. Als elk proces een bepaald entiteittype zou creëren, zouden zich weinig problemen voordoen. Het rangschikken gaat dan vrij stapsgewijs en door de mogelijkheid van het verschuiven van de rijen (wat bij bijvoorbeeld ISP niet mogelijk is), komen de *C*'s altijd wel op de diagonaal terecht. Echter, in het voorbeeld van figuur 1.2 komen processen voor die alleen entiteittypen wijzigen en/of gebruiken (een *U* in de matrix) en dan wordt het rangschikken enigszins arbitrair. Ik had proces *p2* net zo goed onder proces *p6* kunnen zetten, of zelfs geheel buiten het eerste informatiesysteem kunnen laten vallen (zie figuur 1.8, waarin *p2* bij het eerste informatiesysteem hoort). Er zal bij deze transformatie dan ook goed gekeken moeten worden naar de omstandigheden. Uit vele interviews zal moeten blijken wat precies de bedoeling is. Overigens betekent een *U* die buiten de afbakeningen valt ook hier een dataflow.

Deze rangschikking vormt reeds een goede basis voor de systeemafbakening ([AAR1989], p.313). Bij HIP wordt er echter nog een tweetal matrices gecreëerd. In de ene worden de processen aan de organisatie-eenheden gerelateerd (hier wordt een organisatie-eenheid gedefinieerd als 'een binnen de organisatiestructuur te onderkennen en te benoemen gedeelte, dat op zichzelf als een min of meer afgerond geheel beschouwd kan worden en als zodanig algemeen herkenbaar kan worden geacht' ([AAR1989], p.315)). In de derde matrix worden de entiteittypen en de organisatie-eenheden tegen elkaar afgezet. Het aspect verantwoordelijkheid speelt hier de belangrijkste rol. Verantwoordelijk zijn betekent hier



Figuur 1.8 - Matrix na clustering

dat men aansprakelijk is voor het resultaat van een proces of voor het beheer van een entiteittype ([AAR1989], p.315 & 317).

In de proces/organisatie-eenheid matrix (P/V-matrix, ofwel proces-verantwoordelijkheid matrix) wordt uiteengezet welke organisatie-eenheid verantwoordelijk danwel betrokken is bij een proces. Een restrictie is hier dat precies één organisatie-eenheid verantwoordelijk mag worden gesteld voor een proces ([AAR1989], p.315).

In de entiteittype-organisatie-enheid matrix (G/V-matrix, ofwel gegevens-verantwoordelijkheid matrix) wordt uiteengezet welke organisatie-eenheid beheerder danwel gebruiker is van een bepaald entiteittype. Een entiteittype moet door precies één organisatie-eenheid beheerd worden ([AAR1989], p.317).

De twee laatst genoemde matrices kunnen op dezelfde wijze als bij de P/G-matrix gerangschikt worden. Dit zal leiden tot een logische samenhang van de processen en entiteittypen met de organisatie-eenheden en de methode stelt dat dit wederom een afbakening voor eventuele informatiesystemen is. Echter, wanneer na onderzoek blijkt dat verschillende grenzen zich af hebben getekend bij de laatste twee matrices ten opzichte van de eerste matrix (de P/G-matrix), dan dient de P/G-matrix als voornaamste uitgangspunt te functioneren. Analyse van de drie matrices moet in ieder geval leiden tot consistentie. Als bijvoorbeeld proces *p* entiteittype *e* creëert en entiteittype *e* wordt beheerd door organisatie-eenheid *o*, dan dient *o* verantwoordelijk te zijn voor *p*. Overigens is dit consistentie-beleid een aanname. Uit de *tekst* van [AAR1989] lijkt dit een juiste. Uit de *voorbeeldmatrices* ([AAR1989], p.311, 316 & 317) mag dit niet geconcludeerd worden. Daarin is geen enkele vorm van consistentie te vinden. Een gesprek met generaal-majoor Aarts RA, co-auteur van HIP, verduidelijkte dit. De aanname ten aanzien van de consistentie zoals hierboven vermeld, is juist. Op de voorbeelden moest niet gelet worden.

Er dient in ieder geval met een juiste P/G-matrix gewerkt te worden voor het laatste deel van de systeemafbakening. Tezamen met het hoofdentiteitenmodel (dit is een produkt van fase 2 en wordt hier gedefinieerd als ‘een gegevensmodel waarin de belangrijkste entiteittypen van een informatiegebied met hun onderlinge relaties zijn weergegeven met een beschrijving van de bij deze entiteittypen behorende attributen’ ([AAR1989], p.428)) vormen zij het uitgangspunt voor de systeemafbakening. De uiteindelijke systeemafbakening kan anders uitpakken dan uit de P/G-matrix moet blijken. Op grond van verschillende factoren, zoals bijvoorbeeld de geografische spreiding van locaties ([AAR1989], p.319), kan voor een afwijkende systeemafbakening gekozen worden. Consequenties van een andere afbakening dienen natuurlijk goed bekeken te worden, alvorens deze door te voeren. Een goed punt van HIP is in deze het aangeven van de mogelijkheid dat de verschillende gegevensverzamelingen weleens verschillende actualiteitswaarden

(bijvoorbeeld *onmiddelijke verwerking* of *'s nachts* ([AAR1989], p.320) zouden kunnen hebben. Helaas wordt daar verder niet op ingegaan, terwijl dit toch, zoals reeds vermeld in de vorige paragraaf, een belangrijk aandachtspunt is wanneer men verschillende gegevensverzamelingen groepeert en die actualiteitswaarden kunnen zeker van invloed zijn op de keuze van afbakening.

1.2.3 De prioriteitsstelling

De tweede fase waar ik mijn aandacht op vestig, is fase 5, de fase waarin de alternatieven en consequenties bepaald worden. Hierin wordt onder andere een prioriteitsstelling bepaald, waarmee elke afgebakend informatiesysteem een prioriteitswaarde krijgt. Voor het bepalen van de prioriteitsstelling gebruikt HIP twee factoren, namelijk de noodzakelijkheid en het risico van een systeem. Elk afgebakend informatiesysteem krijgt voor elk van deze factoren een waarde toegekend aan de hand van een aantal criteria. Per factor moet men een aantal criteria nalopen en aan elk daarvan een waarde toekennen. Het aantal keuzemogelijkheden van de waarden is beperkt. De factor noodzakelijkheid hanteert acht criteria. Deze zijn ([AAR1989], p.329):

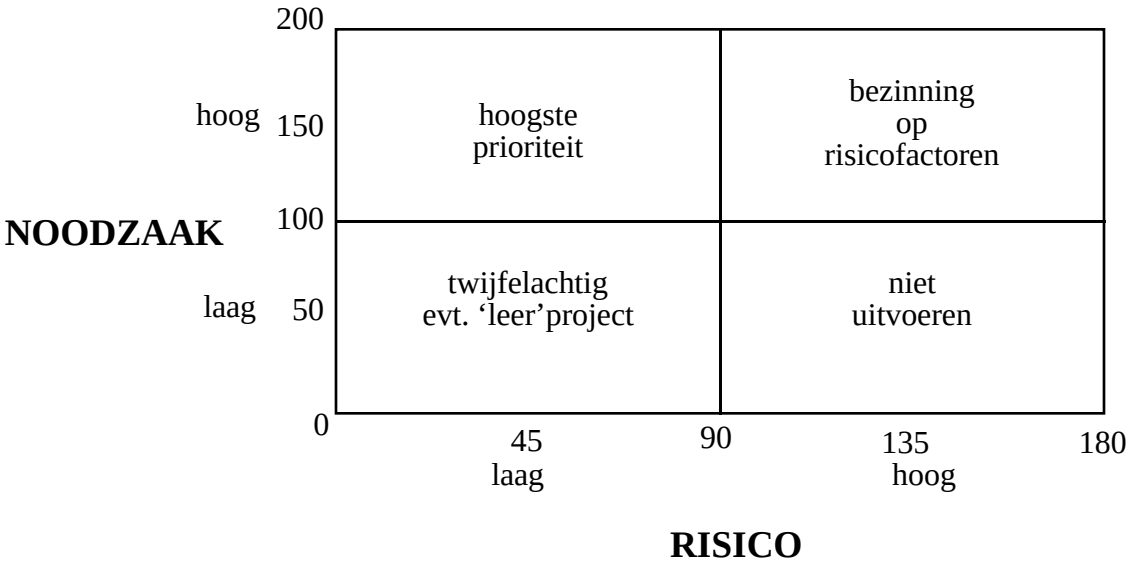
1. In welke mate sluit het informatiesysteem aan bij de voorgestelde bestuurlijke inrichting van de organisatie
2. Hoe belangrijk zijn de knelpunten die moeten worden opgelost door de implementatie van het informatiesysteem
3. In welke mate worden deze knelpunten opgelost door de implementatie van het informatiesysteem
4. Wat is de verwachte continuïteit van het bedrijfsproces/organisatieonderdeel, waarvoor het informatiesysteem wordt ontwikkeld
5. Hoe sterk is de "politieke" druk die van buitenaf wordt uitgeoefend om een dergelijk systeem te ontwikkelen
6. Hoe groot is het aantal organisatie-eenheden dat profiteert van het te ontwikkelen informatiesysteem
7. Wat is de verwachte opbrengst (baten minus kosten)
8. Hoe lang is de ingeschatte terugverdientijd

De factor risico hanteert veertien criteria. Deze zijn ([AAR1989], p.342-343):

1. Geschatte doorlooptijd van het project
2. Benodigde capaciteit voor systeemontwikkeling en programmeren
3. Aantal functionele deelgebieden dat betrokken is
4. Aantal eindgebruikers dat van het systeem gebruik gaat maken
5. Aantal functionele deelgebieden dat gebruik gaat maken
6. Aantal systemen waar het onderhavige systeem bij gebruik van afhankelijk is
7. Gaat het om modificatie of nieuwbouw
8. In hoeverre is het te modificeren systeem reeds geautomatiseerd

9. In hoeverre zullen bestaande administratieve procedures moeten wijzigen
10. Lopen er andere projecten, die afhankelijk zijn van dit project
11. Wat zal de houding van de eindgebruikers zijn
12. Hoeveel automatiseringservaring hebben de eindgebruikers
13. Kent het project een definitieve deadline
14. Zal er sprake zijn van deelprojecten en is de voortgang dan afhankelijk van de coördinatie hier tussen

Voor een lijst van gehanteerde criteria met de bijbehorende scores verwijs ik naar [AAR1989]. Elk te kiezen waarde kent een bepaalde score. Door nu de scores van alle gekozen waarden op te tellen, verkrijgt men een *noodzaak-score* en een *risico-score*. Met behulp van de matrix uit figuur 1.9 kunnen deze scores tegen elkaar afgezet worden. De positie van een informatiesysteem in de matrix geeft de prioriteit aan. Enigszins vreemd is de opbouw van de matrix wel, want de hoogst mogelijke noodzaak-score is 104 (blijkt uit criteria-lijst op p.329 van [AAR1989]) en de hoogst mogelijke risico-score is 164 (blijkt uit criteria-lijst op p.342-343 van [AAR1989]), terwijl de grenzen van figuur 1.9 voor de noodzaak-score 200 en voor de risico-score 180 is.



Figuur 1.9 - Prioriteitsmatrix ([AAR1989], p.344)

1.2.4 Conclusie

Uit de inleiding kwam al naar voren dat HIP zich vooral op het management richt. Hoe de projectgroepen precies opgebouwd worden, wordt niet vermeld. Er wordt alleen gerept over ‘het vrijmaken van materiedeskundigen’ ([AAR1989], p.45) voor de projectorganisatie. Het bepalen van de personen die geïnterviewd moeten worden, wordt aan de gebruiker van HIP overgelaten. Een aanwijsbare betrokkenheid van de mensen op de werkvloer is er dus niet.

De prioriteitsstelling wordt redelijk belicht. Hoewel het toekennen van de waarden aan de verschillende criteria niet altijd even makkelijk zal zijn, biedt HIP een vrij stapsgewijze benadering voor het bepalen van een prioriteitsstelling. Bestaande knelpunten zijn van invloed op de prioriteitsstelling en de gehanteerde methode is duidelijker dan bij ISP (of BSP zo men wil). Misschien dat meerdere criteria het bepalen van de prioriteitsstelling nauwkeuriger zullen laten verlopen. Het niet aansluiten van de prioriteitsmatrix uit figuur 1.9 bij de mogelijke scores is een tekortkoming, of gewoon een slordigheid. Volgens generaal-majoor Aarts RA was het inderdaad nogal slordig. Vreemd genoeg had hij nog nooit aanmerkingen gekregen omtrent de voorbeelden in zijn boek. Wat ik wel mis, is de samenhang tussen verschillende informatiesystemen bij het bepalen van prioriteiten. Er wordt mijns inziens teveel gekeken naar een individueel informatiesysteem bij het wegen van het belang daarvan, terwijl het mogelijk is dat een bepaald informatiesysteem voorrang moet krijgen boven een ander informatiesysteem, indien het positieve gevolg van de implementatie van het eerste informatiesysteem voor andere informatiesystemen groter is. Als door de implementatie van een bepaald informatiesysteem bijvoorbeeld knelpunten worden weggenomen die ook invloed hadden op andere informatiesystemen, dan dient dat informatiesysteem een hoge prioriteit te krijgen.
De afbakening van informatiesystemen gebeurt onnauwkeurig. Zoals reeds vermeld, is hier wederom een bepaalde mate van willekeur aanwezig. Een positieve opmerking dient wel gemaakt te worden ten aanzien van de dataflows. Bij HIP bestaat de mogelijkheid dat een proces uit een bepaald informatiesysteem gegevens uit een ander informatiesysteem kan wijzigen (een *U* in de matrix van figuur 1.7, buiten de diagonaal). Dit bevordert de flexibiliteit van de organisatie. Helaas schieten de voorbeelden bij HIP hopeloos tekort. Maar dit is, zoals reeds vermeld, gewoon een slordigheid.

Afsluitend kan ik zeggen dat HIP een redelijke methode is. Jammer is dat er zeer onnauwkeurig met de voorbeelden is omgegaan en bij een niet altijd even duidelijke tekst is dat een gemis. Helaas schiet HIP ook tekort bij de actualiteitswaarden van verschillende informatiesystemen. Het wordt slechts aangestipt, om er vervolgens niet meer op in te gaan. Een gemiste kans.

1.3 System Development Methodology

1.3.1 Inleiding

De System Development Methodology (kortweg SDM) is een faseringsmethode waar een eigen invulling aan gegeven dient te worden. Deze methode bestaat al ruim twintig jaar en is in de loop der tijd onderhevig geweest aan verscheidene veranderingen. Met de ontwikkeling van de automatisering dienden ook de methodieken aangepast te worden. De hier gebruikte versie van SDM stamt uit 1990. Zij is onder andere aangepast ten aanzien van vorige versies vanwege 'het steeds groter wordende belang van de ontwikkeling van informatiesystemen voor de totale bedrijfsplanning, (...), het toegenomen bewustzijn ten aanzien van de invloed van informatiesystemen op de organisatie' ([TUR1990], p.VII). SDM bestaat uit een aantal fasen, die een systeem-ontwikkeling stap voor stap beschrijven, van de informatieplanning tot en met het gebruik en beheer van het systeem. Voor mij is alleen de fase *Informatieplanning* van SDM interessant. De andere fasen zullen niet worden besproken. Omdat ik alleen het onderdeel *Informatieplanning* van SDM gebruik, zal in het vervolg met de benaming SDM het informatieplannings-gedeelte van SDM bedoeld worden.
Informatieplanning bij SDM (voortaan dus aangeduid met SDM) bestaat uit tien activiteiten, namelijk ([TUR1990], p.1):

1. Leg uitgangspunten vast en stel plan van aanpak op.
2. Verzamel gegevens over organisatie en analyseer situatie.
3. Selecteer interessegebieden en definieer taakstelling Informatieplanning.
4. Rapporteer over situatie-analyse en stel plan van aanpak bij.
5. Bepaal criteria toekomstige informatievoorziening.
6. Ontwikkel informatie-architectuur.
7. Bepaal voorkeuren systeemontwikkeling en realisatie-voorwaarden.
8. Maak projectenplan met kosten/batenraming.
9. Valideer Informatieplanning.
10. Stel informatieplan op en rapporteer.

Deze activiteiten zijn in figuur 1.10 weergegeven. Daaruit blijkt dat bijvoorbeeld activiteit 4 pas kan beginnen nadat activiteit 3 gedaan is, terwijl activiteit 6 parallel aan activiteit 5 uitgevoerd kan worden (als een pijl van het begin van een activiteit reeds naar het begin van de volgende activiteit loopt en van het eind van eerstgenoemde activiteit naar het eind van laatstgenoemde activiteit, dan kan deze laatste activiteit gelijktijdig met de eerste activiteit uitgevoerd worden). Wederom zal ik niet alle activiteiten beschrijven. Vele zullen door hun identificatie reeds duidelijk zijn. Voor wat betreft de afbakening van informatiesytstemen is activiteit 6 van belang. De prioriteitsstelling komt

in activiteit 7 aan de orde. Wanneer informatie uit voorgaande activiteiten nodig is, zal dit aangegeven worden, maar de totstandkoming ervan zal niet of nauwelijks beschreven worden. Voor een uitgebreide handeling over SDM verwijs ik naar [TUR1990]. Ook hier zal weer vaak aan de voorgaande paragrafen gerefereerd worden. Voor een goed begrip van de inhoud van deze paragraaf dient de lezer met die paragrafen bekend te zijn.

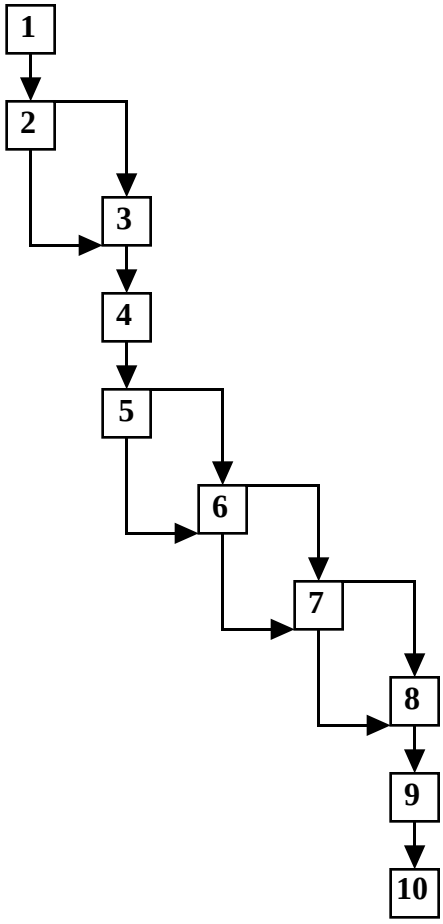
1.3.2 De afbakening

In activiteit 6 worden de informatiesystemen gedefinieerd, die in de eerder gekozen interessegebieden liggen (dit is in activiteit 3 gebeurd). Volgens SDM is dit ‘het hart van de informatieplanning’ ([TUR1990], p.65). Voor het uitvoeren van deze activiteit moet men op de hoogte zijn van het informatiebeleid van de organisatie, de uitgangspunten voor de toekomstige informatievoorziening en de documentatie over de bestaande informatiesystemen ([TUR1990], p.68). Hoewel figuur 1.10 laat zien dat activiteit 6 parallel aan activiteit 5 uitgevoerd kan worden, blijkt er toch informatie uit activiteit 5 nodig te zijn voor activiteit 6 (uitgangspunten voor toekomstige informatievoorziening). Er is dan ook eerder sprake van een soort feedback van informatie. Activiteit 6 toetst de uitgangspunten van activiteit 5 en eventuele veranderingen worden alsnog uitgevoerd, wanneer dat nodig mocht zijn.

Het afbakenen van informatiesystemen verloopt stapsgewijs. Ten eerste moeten de belangrijkste processen in de vastgestelde interessegebieden bepaald worden ([TUR1990], p.65-67). Of een proces als belangrijk wordt ervaren, dient te worden bepaald door het management. Volgens SDM spelen Critical Success Factors (CSF’s, zie §1.1.2) hierbij een belangrijke rol. Na het vaststellen van die CSF’s moeten de belangrijkste produkten of diensten van de organisatie geïdentificeerd worden ([TUR1990], p.67), waarna de daaraan gerelateerde processen vastgesteld kunnen worden. Na overleg met het management en/of deskundigen dienen de belangrijkste processen gedefinieerd te zijn.

De volgende stap is het vaststellen van de entiteitgroepen (hier gedefinieerd als ‘de met elkaar verband houdende entiteiten’ ([TUR1990], p.66), die de reeds gedefinieerde processen ondersteunen ([TUR1990], p.67). Hierbij wordt gekeken naar de entiteiten die de processen initiëren, besturen, begeleiden of daar gecreëerd worden ([TUR1990], p.67).

De derde stap is het onderling relateren van processen, entiteitgroepen, bestaande informatiesystemen en organisatie-delen. In dit verband kan men de processen vergelijken met de bedrijfsfuncties van ISP. Zij zijn dus niet elementair en kunnen nog verder opgesplitst worden. SDM wil op dit punt in haar methode niet te gedetailleerd te werk gaan. Er worden matrices



Figuur 1.10 - Fasen van informatieplanning ([TUR1990], p.1)

gecreëerd voor ([TUR1990], p.66-67):

- Processen & entiteitgroepen. Welke entiteitgroepen worden door welke processen gecreëerd, gebruikt en/of gewijzigd (vergelijk P/G-matrix uit vorige paragraaf)?
- Processen & organisatie-delen. Welke organisatie-delen zijn verantwoordelijk voor welke processen (vergelijk P/V-matrix uit vorige paragraaf)?
- Entiteitgroepen & organisatie-delen. Welke organisatie-delen zijn gebruiker van welke entiteitgroepen (vergelijk G/V-matrix uit vorige paragraaf)?
- Processen & bestaande informatiesystemen. Welke processen hebben problemen met welke bestaande informatiesystemen?
- Entiteitgroepen & bestaande informatiesystemen. Welke bestaande informatiesystemen maken gebruik van welke entiteitgroepen?

Tevens moeten de bestaande gegevensstromen onderling in verband gebracht worden. Hoe dit precies in zijn werk moet gaan, is helaas niet duidelijk. In ieder geval moeten de matrices geanalyseerd worden. Hierdoor worden eventuele tegenstrijdigheden er uit gehaald, analoog aan de

consistentie van informatie uit de voorgaande paragrafen.

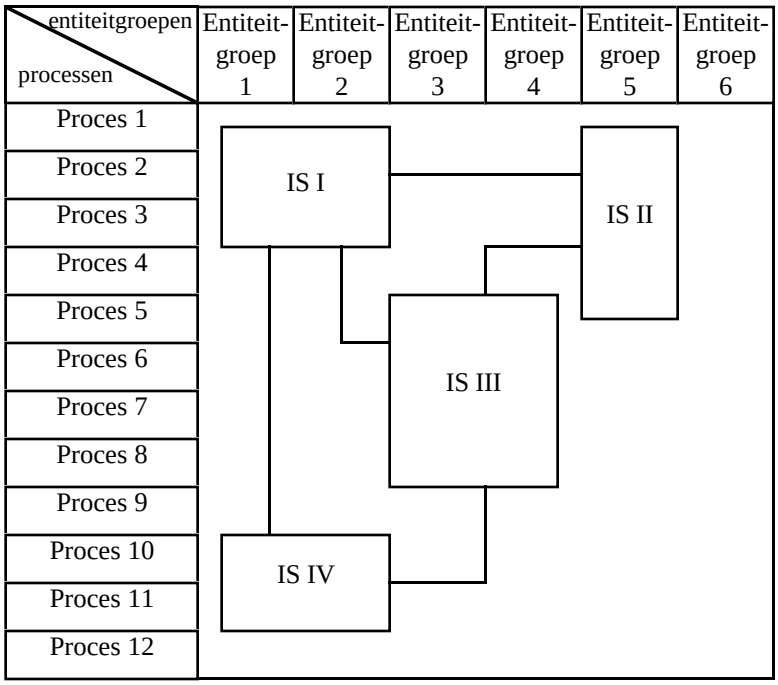
Ik zal hier niet weer een voorbeeld van een matrix geven. De voorgaande paragrafen geven voldoende inzicht in dit soort matrices en er wordt bij SDM nergens aangegeven dat de matrices er hier anders uitzien. Er wordt beschreven wat er gedaan moet worden, maar hoe dat precies moet gebeuren, wordt niet vermeld. Dit wordt ter invulling aan de toepasser gelaten. Met SDM vergeleken, zijn ISP en HIP dan ook zeer nauwkeurig in hun omschrijvingen. We mogen aannemen dat SDM de BSP-methode in gedachten heeft voor het afbakenen van informatiesystemen, gezien het voorkomen van op BSP gerichte literatuur (zie p.703-726 van [TUR1990]).

De laatste stap is het ontwerpen van de informatie-architectuur (hier gedefinieerd als ‘De onderlinge structuur tussen basisprocessen, entiteitsgroepen en informatiesystemen binnen een organisatie’ ([TUR1990], p.732)). Basisprocessen zijn de belangrijkste processen in het te beschouwen deel van de organisatie. Na de analyse van eerder genoemde matrices dient er een dusdanig duidelijke samenhang tussen de processen en de entiteitsgroepen naar voren gekomen te zijn, dat deze in een matrix gerangschikt kunnen worden, zodanig dat er clusters ontstaan van onderlinge verbanden ([TUR1990], p.66). Die clusters zijn dan de afgebakende informatiesystemen. Hoe men moet rangschikken, wordt niet uitgelegd. Hier zal er door de toepasser van SDM een eigen techniek aangedragen moeten worden. Figuur 1.11 geeft een voorbeeld van een informatie-architectuur weer. Een opvallend verschil met ISP en HIP is dat de informatiesystemen zich niet zozeer op de diagonaal aftekenen, maar overal in het schema voor kunnen komen. Hoewel er niets over vermeld wordt, neem ik aan dat de lijnen die informatiesystemen onderling verbinden (zie figuur 1.11, de lijnen zonder pijlpunten), duiden op dataflows.

1.3.3 De prioriteitsstelling

In activiteit 7 komt de prioriteitsstelling aan de orde. Aan de verschillende geïdentificeerde informatiesystemen worden prioriteiten gegeven uitgaande van de informatie-architectuur, de bestaande informatiesystemen en de daarmee samenhangende problematiek ([TUR1990], p.70). Er wordt hierbij een aantal criteria gebruikt, waarbij echter niet (zoals bijvoorbeeld bij HIP het geval is) stapsgewijs waarden aan die criteria worden toegekend. De voorkeur die men heeft voor de implementatie van bepaalde informatiesystemen dient slechts onderbouwd te worden met een aantal redenen. Deze redenen kunnen zijn ([TUR1990], p.72):

- Kans op succes
- Bestaande problemen
- Verwachte ontwikkelingen



Figuur 1.11 - Informatie-architectuur (vrij naar [TUR1990], p.69)

- Mogelijke voordelen
- Veranderingsbehoefte
- Technische ontwikkelingen en mogelijkheden
- Aanwezige kennis en middelen
- Wettelijke noodzaak

Enigszins te vergelijken met de criteria die gebruikt worden bij ISP (zie §1.1.3), echter zonder de waardetoekenning zoals ook ISP die kent. De redenen dienen slechts ter motivatie. Waarschijnlijk komt men tot een voorkeur met behulp van vele interviews met het management en grondige analyses van de bestaande problematiek. SDM staat weinig staat bij de echte inhoudelijke zaken omtrent het komen tot een prioriteitsstelling. De afgebakende informatiesystemen worden voorzien van een prioriteit gebaseerd op enkele criteria, maar mijns inziens is dat een te complexe aangelegenheid. Er zouden meer gedetailleerde factoren moeten bestaan die het geheel overzichtelijker maken. Dit lijkt paradoxaal, meer details en toch overzichtelijker, maar ik bedoel dat de informatieplanner beter begeleid moet worden door het nemen van kleinere stappen. De door SDM gehanteerde manier leunt teveel op de ervaring van de informatieplanner, of misschien wordt er ook hier van uitgegaan dat andere methoden bijspringen.

In ieder geval ziet een prioriteitsstelling volgens SDM er uit als in figuur 1.12. Hieruit kan men bijvoorbeeld aflezen dat het noodzakelijk is dat informatiesysteem 2 (IS2) geautomatiseerd wordt.

- -	onmogelijk
-	mogelijk doch niet gewenst
0	mogelijk
+	mogelijk en gewenst
+ +	noodzakelijk

	Handmatig	Formalisering	Automatisering
IS1	-	++	++
IS2	-	+	++
IS3	-	+	+
IS4	0	++	0

Figuur 1.12 - Voorkeuren (vrij naar [TUR1990], p.74)

1.3.4 Conclusie

Het is duidelijk dat SDM een methode is voor de gehele systeemontwikkeling. De informatieplanner wordt te weinig begeleid in zijn of haar activiteiten. De grootste tekortkoming van SDM is het simpelweg ontbreken van echte methodieken voor de afbakening van en het geven van prioriteiten aan verschillende informatiesystemen. Er wordt gezegd wat er moet gebeuren, maar niet hoe dit precies dient te gebeuren. Hierdoor kan ik stellen dat zowel de afbakening als de prioriteitsstelling niet door SDM voldoende worden ondersteund. Als informatieplanningsmethode is SDM ongeschikt. Het meer bij de hand nemen van de informatieplanner zou een hele verbetering zijn. Wellicht dat HIP als methode de eerste fase van SDM (de informatieplanning dus) beter kan invullen.

1.4 ARTIST

1.4.1 Inleiding

De vierde en laatste methode die ik beschrijf, is de methode ARTIST[2], zoals beschreven in [HUI1989]. Aanleiding voor de ontwikkeling van deze methode was de ontevredenheid over bestaande informatieplannings-methoden. Deze hadden voldoende aandacht voor het identificeren van informatiesystemen binnen een organisatie, maar de planning van de ontwikkeling daarvan bleef onderbelicht, terwijl zo'n planning toch erg belangrijk is ([HUI1989], p.1). ARTIST gaat wel dieper in op de planning van de ontwikkeling van informatiesytstemen. Hoewel de basis van de beschrijving van ARTIST [HUI1989] is, zal waar nodig verwezen worden naar [GRE1985], omdat daarin enkele begrippen van ARTIST uitgebreid beschreven worden.

Bij ARTIST zijn elf fasen te ondescheiden. Deze zijn ([HUI1989], p.2-21):

1. Initialisering (kennismaking, doel en afbakening, werkwijze, etc.)
2. Inventarisatie huidige informatievoorziening (vergaren van informatiebehoeften, wensen en knelpunten via interviews)
3. Bedrijfsinformatiemodel (het komen tot een informatiemodel van het bedrijf met behulp van de interviewresultaten uit fase 2 en analyse van het bedrijf)
4. Volledigheids- en juistheidscontroles uitvoeren en het creëren van modelrapporten (controleren op fouten en in kaart brengen van samenhang tussen verschillende elementen van het bedrijf)
5. Globaal entiteitenmodel (in kaart brengen van de belangrijkste bedrijfsgegevens en de onderlinge samenhang)
6. Database benadering, oplossingsvoorstellen voor knelpunten en pre-adviezen (onderlinge samenhang tussen gegevensverzamelingen en de structuur van de gegevens, in kaart brengen van belangrijke knelpunten en oplossingen hiervoor aandragen, het nemen van voorlopige beslissingen naar aanleiding van de knelpunts-oplossingen)
7. Architectuurmodel (voorlopig model afgeleid van bedrijfsinformatiemodel, waarin resultaten uit fase 6 zijn verwerkt, zoals bijvoorbeeld een bepaalde groepering van gegevensverzamelingen)
8. Van hoofdinformatiestromen naar subsystemen (onderscheiden van hoofdinformatie-stromen binnen het bedrijfsinformatiemodel, wat de afbakening van informatiesystemen impliceert; dit aftekenen in het architectuurmodel)
9. Prioriteitsstelling van de subsystemen (informatiesysteemplanning op basis van knelpunten)
10. Informatieplan (beschrijving van de afbakening, prioriteitsstelling en andere zaken betreffende de informatievoorziening; een "wat-plan")

[2] ARTIST staat voor *Assyst RAET Tool for Information and Software Technology* en is eigenlijk een door RAET geautomatiseerd ondersteunend gereedschap. De heer Huis in 't Veld, naast auteur van [HUI1989] ook ontwikkelaar van het gereedschap ARTIST, heeft de in deze paragraaf te beschrijven methode dezelfde naam gegeven.

11. Automatiseringsplan (beschrijving van hoe het beschrevene in het informatieplan gerealiseerd moet worden; een “hoe-plan”)

Ik zal niet alle fasen verder toelichten. Het belangrijkste deel van de afbakening van informatiesystemen gebeurt in fase 8 en dat van de prioriteitsstelling in fase 9. Er wordt daarbij echter veel informatie uit voorgaande fasen gebruikt en dit zal, waar nodig, aangegeven worden. Het ontstaan van bepaalde informatie in voorafgaande fasen zal dan beschreven worden, nu eens summier, dan weer uitgebreider. Voor een meer uitgebreide beschrijving van ARTIST verwijs ik naar [HUI1989].

1.4.2 De afbakening

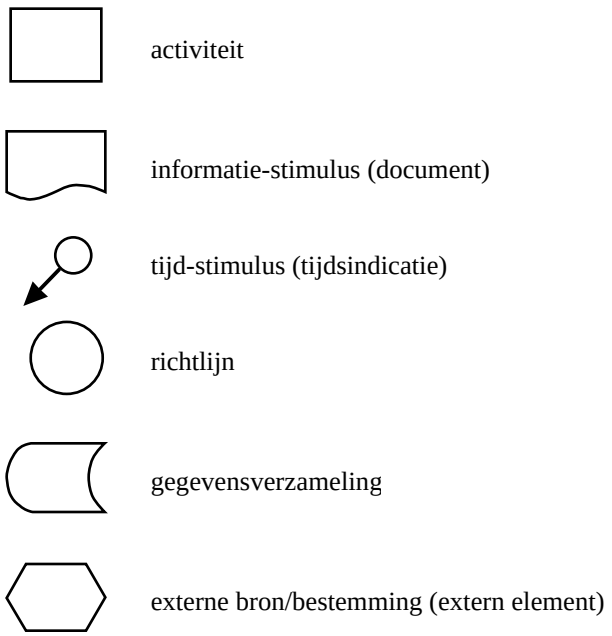
Zoals reeds vermeld, vindt de afbakening van informatiestromen voornamelijk plaats in fase 8. Voorbijgaan aan de voorafgaande fasen is in deze niet mogelijk. Essentieel voor de afbakening van informatiesystemen is het architectuurmodel uit fase 7. Dit model vormt de basis voor de afbakening. ARTIST gebruikt zogenaamde hoofdinformatiestromen, ofwel HISsen. Deze zijn af te leiden uit het bedrijfsinformatiemodel en worden ingeast in het architectuurmodel. Voordat uitgelegd wordt hoe dat in zijn werk gaat, dient eerst het architectuurmodel aan een nader onderzoek onderworpen te worden.

Het architectuurmodel is een bedrijfsinformatiemodel waarin oplossingen van bestaande informatieproblemen zijn verwerkt. Het bedrijfsinformatiemodel is een model van de organisatie waarin het aspect informatie centraal staat. Een dergelijk model is opgebouwd uit bedrijfsactiviteiten, gegevensverzamelingen, informatiestromen, stimuli die activiteiten activeren, richtlijnen (normen voor de activiteiten), externe bestemmingen, functionele eenheden van het bedrijf en de onderlinge samenhang van deze elementen. Figuur 1.13 toont de bouwstenen van zo’n model en figuur 1.14 laat een mogelijk deel van een bedrijfsinformatiemodel zien. Eigenlijk moet er tussen een activiteit en een gegevensverzameling een document aangegeven worden, maar voor de overzichtelijkheid wordt deze weggelaten. Een functionele eenheid kan gewoon aangegeven worden door de elementen die onderdeel zijn van dezelfde functionele eenheid af te bakenen.

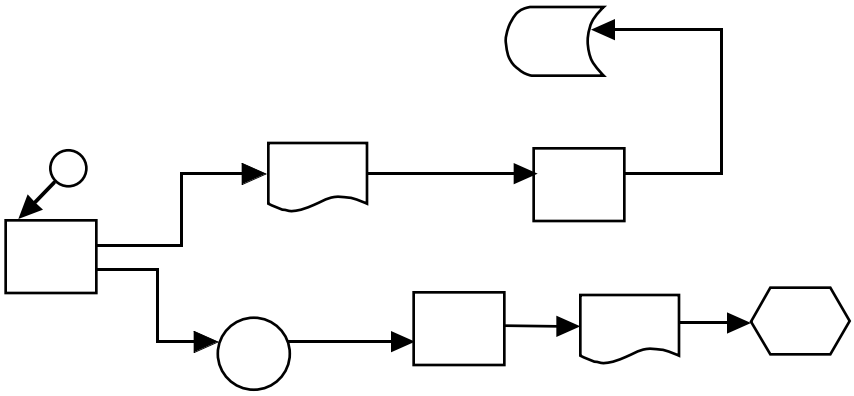
In fase 3 komt het bedrijfsinformatiemodel tot stand uit de gehouden interviews. Tevens komen in die interviews knelpunten naar voren, waarvoor oplossingen aangedragen dienen te worden. Deze oplossingen worden dan in het architectuurmodel verwerkt, wat eigenlijk neerkomt op het niet meer weergeven van de opgeloste knelpunten. Daarnaast wordt er nog een ander belangrijk aspect in het architectuurmodel weergegeven, namelijk de samenhang tussen de verschillende gegevens.

In fase 6 wordt dit inzicht verkregen. In de fasen 4 en 5 werd reeds het gegevensgebruik van de organisatie duidelijk door middel van het opstellen en analyseren van matrices. Deze matrices relateren de activiteiten aan de bestaande gegevensverzamelingen en de activiteiten aan de gegevenselementen (een gegevensverzameling is opgebouwd uit gegevenselementen). Door een nadere bestudering van de matrix activiteiten/gegevensver-zamelingen kunnen in fase 6 bepaalde gegevensverzamlingen gegroepeerd worden ([HUI1989], p.12), bijvoorbeeld door gelijksoortig gebruik.

Nu het bedrijfsinformatiemodel uitgekristalliseerd is (gecontroleerd op volledigheid en juistheid), kan aan de identificatie van de HISsen begonnen worden. Hoofdinformatiestromen worden gedefinieerd als informatieproces-modellen (informatieprocessen zijn processen die gegevens(stromen) behandelen ([HUI1989], p.14)). Een proces is hier een ‘geheel van samenhangende activiteiten, die noodzakelijk zijn voor de transformatie van een invoer tot de door de organisatie of omgeving gewenste uitvoer’ ([HUI1989], p.14). Het is dus van belang deze informatieprocessen te onderscheiden in het architectuurmodel, aangezien dat de basis is voor de afbakening van informatiesystemen. Een HIS kan ontstaan als een extern element een activiteit binnen het model triggert (het triggeren van een activiteit is het in werking zetten ervan), bijvoorbeeld de plaatsing van een order door een klant ([HUI1989], p.14). Ook kan een HIS ontstaan bij het triggeren door een tijd-stimulus, waarbij een activiteit op een bepaald moment



Figuur 1.13 - Elementen van een bedrijfsinformatiemodel ([HUI1989], p.7)



Figuur 1.14 - Mogelijke samenhang van elementen

uitgevoerd wordt (bijvoorbeeld een salarisprogramma dat één keer per maand uitgevoerd wordt). De volgende stap in het identificeren van HISsen is het bepalen van de activiteiten die als gevolg van de eerste activiteit ook getriggerd worden. Zo ontstaan zogenaamde *stimuli-afhankelijkheidstabellen*, tabellen waarin de eerst getriggerde activiteit zich links bovenin bevindt en de daarna getriggerde activiteiten in volgorde van links naar rechts en van boven naar beneden staan. Zo'n tabel kan op twee manieren eindigen, namelijk doordat een interne stimulus een extern element bereikt (dus het model verlaat) of doordat een interne stimulus in een gegevensverzameling wordt opgenomen ([HUI1989], p.15; "archivering").

Het zal duidelijk zijn dat vele tabellen overeenkomstige elementen hebben. Het komt bijvoorbeeld vaak voor dat een activiteit twee of meerdere stimuli kan hebben (bijvoorbeeld activiteit *a* die van zowel activiteit *b* als van activiteit *c* een document (informatie-stimulus) ontvangt). Zo zal het voorkomen dat bij het genereren van een bepaalde HIS *h1* een activiteit bereikt wordt die reeds onderdeel is van een eerder gegenereerde HIS *h2*. Na deze activiteit zal het pad hetzelfde zijn bij HIS *h1* als bij HIS *h2*. Het is natuurlijk niet wenselijk dat de activiteiten op dat pad bij zowel *h1* als bij *h2* horen. Er zal dus besloten moeten worden hoe de tabellen afgekapt moeten worden. Dit proces wordt het *zuiveren der stimuli-afhankelijkheidstabellen* genoemd ([HUI1989], p.15). De gegenereerde tabellen worden dus sterk afgeslankt. Dit vereist nogal wat ervaring. Er wordt wel een formele aanpak gegeven ([HUI1989], p.16-17), maar deze voldoet niet, althans niet voor elke situatie. Iedere praktijksituatie is weer anders en daarom is er geen algemene aanpak voor te geven. Logica speelt een grote rol bij het zuiveren van de tabellen. Komt men bij het genereren van een HIS een activiteit tegen die logisch gezien bij een andere HIS hoort, dan moet deze activiteit niet opgenomen worden in de tabel voor de eerste HIS.

De reeks activiteiten in een tabel vormen een definitieve HIS. Een HIS is dus een logische uitvoer ten gevolge van een bepaalde stimulus ([HUI1989], p.17). In het architectuurmodel kunnen nu alle gevonden HISsen aangegeven worden. Dit is de afbakening van de toekomstige informatiesystemen. Om de relatie aan te geven tussen de gevonden HISsen en de in het architectuurmodel aangegeven gegevensverzamelingen wordt er een matrix gecreëerd waarin de HISsen tegen de gegevensverzamelingen afgezet worden. Deze matrix geeft aan welke gegegevensverzamelingen bij welke HISsen betrokken zijn, dus welke gegevensverzamelingen door de toekomstige informatiesystemen gebruikt zullen worden.

Duidelijk zal zijn dat [HUI1989] de HISsen definieert vanuit de invoerkant. Er wordt gekeken naar het gevolg van een bepaalde stimulus. Een andere manier om tot een HIS te komen, is te beginnen bij de uitvoer. Hoofdgedachte hierbij is dat de uitvoer het doel is van een informatiesysteem ([GRE1985], p.67-74). Door nu bij de verschillende uitvoer-elementen terug te gaan in het model, kan men tot verschillende HISsen komen. Men kijkt dan naar de informatie die nodig is voor het tot stand komen van een bepaalde uitvoer. Omdat deze manier buiten het blikveld van [HUI1989] valt, zal ik er hier niet verder op ingaan. Voor een uitgebreide beschrijving van deze benadering verwijs ik naar [GRE1985].

1.4.3 De prioriteitsstelling

In fase 9 vindt de prioriteitsstelling plaats. Deze is gebaseerd op de overgebleven knelpunten. Tijdens de interviews in fase 2 kwamen deze al naar voren, waarna zij in fase 6 onderling werden gerelateerd door het creëren van zogenaamde knelpunt-afhankelijkheidsketens. Knelpunt-afhankelijkheidsketens zijn gebaseerd op het idee dat de activiteiten van een informatiemodel op een bepaalde manier samen kunnen hangen. Een knelpunt treedt bij een activiteit op en heeft daarom invloed op andere activiteiten, die met de knelpunt-houdende activiteit samenhangen. Zo kan in kaart worden gebracht hoe en in welke mate knelpunten elkaar onderling beïnvloeden ([HUI1989], p.18). Er zal dus ook duidelijk worden welk knelpunt de aanstichter van een knelpunt-afhankelijkheidsketen is. Het idee achter de prioriteitsstelling van ARTIST is het feit dat ook de HISsen reeds gedefinieerd zijn (zie de vorige paragraaf) en dat deze bestaan uit samenhangende activiteiten. Als nu de gevonden knelpunt-afhankelijkheidsketens over de HISsen heen worden gelegd, moet blijken met welke prioriteit de informatiesystemen (de HISsen) ontwikkeld dienen te worden. Er zal dan immers blijken welke HISsen de belangrijkste knelpunten (de zogenaamde “bottle-necks”) bevatten en daarmee een hoge prioriteit toegedicht krijgen. Met belangrijkste knelpunten bedoel ik hier de knelpunten die vooraan in de knelpunt-afhankelijkheidsketens staan. Bij oplossing daarvan kunnen ook vele andere knelpunten opgelost worden. Stapsgewijs gaat dit proces als volgt ([HUI1989], p.18-19):

1. *Bepalen knelpuntenmatrix en knelpunt-afhankelijkheidsketens*
   Er wordt een matrix gecreëerd waarin alle activiteiten van het informatiemodel tegen elkaar zijn afgezet. Als een activiteit nu een bepaalde invoer als knelpunt ervaart, dan wordt in deze matrix aangegeven welke activiteit het knelpunt ervaart (activiteit op de rij), welke activiteit deze invoer verzorgt (activiteit op de kolom) en welke invoer dit betreft (inhoud van matrix op de positie aangegeven door de beide activiteiten). Een voorbeeld van zo'n matrix wordt gegeven in §2.1. Tevens worden knelpunt-afhankelijkheidsketens gecreëerd, waarin de activiteiten met de tussenliggende in- en uitvoer worden aangegeven. De voorste activiteit kent geen knelpunten, maar de daarop volgende activiteiten wel en deze knelpunten worden mogelijkerwijs veroorzaakt door de eerste activiteit ([HUI1989], p.18).

2. *Relateren HISsen en knelpunt-afhankelijkheidsketens*
   Door het behoren van knelpunten aan bepaalde activiteiten en het behoren van activiteiten aan bepaalde HISsen, kan een verband worden gelegd tussen de knelpunten en de HISsen. Dit verband maakt duidelijk welke HISsen knelpunten bevatten en welke knelpunten dat dan zijn.

3. *Toekennen prioriteiten*
   Door de relatie tussen HISsen en knelpunten te analyseren, kan een prioriteitsstelling vervaardigd worden. Als een bepaalde HIS een belangrijk knelpunt bevat, dat invloed heeft op knelpunten in andere HISsen, dan dient deze HIS een hoge prioriteit te krijgen.

ARTIST kent dus een prioriteitsstelling op knelpuntsbasis. In [HUI1989] wordt niet verder ingegaan op hoe die toekenning van prioriteiten precies tot stand komt. In het volgende hoofdstuk zal getracht worden hier een antwoord op te geven. Er zal met andere woorden een verband gelegd worden tussen HISsen, knelpunten en prioriteiten.

1.4.4 Conclusie

Vergeleken met de methoden uit de vorige drie paragrafen is ARTIST zeker anders. Eindelijk worden informatiesystemen vanuit een andere invalshoek afgebakend. Deze afbakening is namelijk gebaseerd op de informatiestromen binnen een organisatie. Ook sluiten de HISsen beter aan op de welbekende workflows, omdat beide uitgaan van informatiestromen. Deze aansluiting zal in een later hoofdstuk beschreven worden.

ARTIST wordt echter te summier beschreven in [HUI1989] om een goed beeld te krijgen van deze methode. Behalve dat is het duidelijk een methode die veel leunt op de ervaring van de toepassers

ervan. Tevens vind ik dat het belangrijkste doel van ARTIST, namelijk het bieden van een betere planning voor de informatiesystemen dan andere methoden, slechts ten dele bereikt is. Zeker wordt er een goede manier aangereikt, maar zij wordt niet verder uitgewerkt. Ik zal daar in het volgende hoofdstuk wel dieper op ingaan.

Een opmerking over gegevensverzamelingen is hier op z'n plaats. Er kan namelijk onderscheid gemaakt worden tussen gegevensverzamelingen die bij één HIS horen en die door meerdere HISsen gebruikt worden. Deze laatste vereisen een aparte aanpak, daarom dienen zij voor aanvang van de ontwerp-projecten apart genomen en ontwikkeld te worden. [HUI1989] zegt hier niets over.

De afbakening van informatiesystemen is bij ARTIST, zoals reeds eerder vermeld, verfrissend. Ook is een belangrijk probleem van andere methoden, namelijk het slecht in ogenschouw nemen van de mogelijke verschillen in actualiteitswaarden van processen, bij ARTIST opgelost door te kijken naar de verschillende tijdstimuli die een proces kunnen initiëren. Logica zal daarbij moeten uitwijzen dat een activiteit met een lage actualiteitswaarde niet in een proces (een HIS in dit geval) thuishoort dat constant uitgevoerd wordt. Het is echter wel jammer dat er voor het zuiveren van de stimuli-afhankelijkheidstabellen geen eenduidige methode aan te geven is.

De prioriteitsstelling is interessant, maar slechts summier belicht. Verder dan een vermelding dat er naar de relatie tussen knelpunten en HISsen gekeken moet worden, gaat [HUI1989] eigenlijk niet. Dit nodigt natuurlijk uit tot een nadere beschouwing (zie het volgende hoofdstuk).

ARTIST is een interessante methode, die echter nog verder uitgewerkt dient te worden. Hoewel het niet de bedoeling zal zijn van [HUI1989] om een volledige beschrijving van een methode te geven, zoals bijvoorbeeld [AAR1989] duidelijk wel is, is het jammer dat de karakteristieken van ARTIST niet gedetailleerd zijn uitgewerkt.

Conclusie

Elke beschreven methode kent z’n tekortkomingen op het gebied van de afbakening en de prioriteitsstelling van informatiesystemen.

ISP is te onnauwkeurig en laat teveel in het midden omtrent de te volgen stappen.

De methode HIP is ook onnauwkeurig in de afbakening van informatiesystemen. De prioriteitsstelling wordt redelijk belicht, maar de invloed van bepaalde informatiesystemen op andere blijft onbelicht.

SDM valt een beetje buiten de boot, omdat het een faseringsmethode is. Hierdoor blijft een goede beschrijving van de werkwijze omtrent het afbakenen en het geven van prioriteiten achterwege. Dat is tevens het grootste punt van kritiek. Een methode die al zolang gebruikt wordt en zoveel veranderingen heeft ondergaan, mag onderhand best een nauwkeurige inhoud aan de fasen geven.

ARTIST is verfrissend en redelijk nauwkeurig in het afbakenen van informatiesystemen. Echter, er wordt nogal wat ervaring geëist van de toepasser van deze methode. Van de prioriteitsstelling wordt beschreven wat er moet gebeuren, zonder een invulling te geven. Het volgende hoofdstuk beschrijft een manier om prioriteiten toe te kennen, waarbij ARTIST als ondergrond wordt gebruikt.

Gebleken is dat geen enkele methode een voldoende objectieve manier aandraagt voor het geven van prioriteiten aan de ontwikkeling van informatiesystemen. Het invullen van de criteria blijft een subjectieve bezigheid.

Hoofdstuk 2
De Prioriteitsstelling

Inleiding

In dit hoofdstuk presenteer ik een methode waarmee men een uitspraak kan doen over de prioriteiten van informatiesystemen ten aanzien van de ontwikkeling ervan. Deze prioriteitsstelling is een aanvulling op de methode ARTIST (zie §1.4). Volgens deze methode moet een prioriteitsstelling voortkomen uit de HISsen en de daaraan gerelateerde knelpunten (zie §1.4), maar gaat niet in op hoe dit precies dient te gebeuren. Dit hoofdstuk zal trachten aan te geven hoe men op basis van knelpunten en HISsen een prioriteitsstelling kan bepalen.

Om tot een methode te komen voor het bepalen van de prioriteiten, wordt een aantal aspecten van ARTIST gebruikt, namelijk de knelpunten en de HISsen. Uit interviews kwamen knelpunten naar voren, waarbij voor enkele reeds oplossingen werden aangedragen bij de pre-adviezen. Het gaat echter om de overgebleven knelpunten. Deze zijn gerelateerd aan activiteiten, want een knelpunt wordt bij een activiteit gesignaleerd. Doordat knelpunten verbonden zijn met activiteiten en deze activiteiten zich in een bepaalde onderlinge samenhang bevinden, kunnen de knelpunten ook onderling gerelateerd worden. Hierdoor ontstaan zogenaamde knelpuntsketens (zie §1.4.3). ARTIST beschrijft ook een manier om verschillende HISsen binnen een organisatie te onderscheiden. Aangezien HISsen uit activiteiten bestaan en activiteiten knelpunten kunnen bevatten, bestaat er een relatie tussen HISsen en knelpuntsketens. Deze relatie is de basis van mijn theorie. Om te komen tot een goede beschrijving van mijn theorie moet ik een aantal aannames doen. Zo neem ik aan dat de HISsen en knelpunten reeds zijn gedefinieerd. Ook zijn de knelpuntsketens reeds bekend zoals zij ontstaan uit de knelpuntenmatrices (zie §1.4.3). Ik zal niet ingaan op de manier waarop deze aspecten tot stand zijn gekomen. Voor een uitgebreide beschrijving van ARTIST verwijs ik naar §1.4 en [HUI1989].

Op dit punt zijn enkele belangrijke opmerkingen over knelpunten op z’n plaats. Er bestaan verschillende soorten knelpunten. Een activiteit kan bijvoorbeeld een knelpunt ervaren doordat de benodigde informatie te laat binnenkomt (tijdigheid-knelpunt) of omdat bepaalde informatie slecht leesbaar is (lees- en interpreteerbaarheid-knelpunt). Zo kunnen knelpunten verschillende oorzaken hebben. Daarnaast is er ook nog een scheiding aan te geven tussen knelpunten met betrekking tot

de invoer en knelpunten met betrekking tot de activiteit zelf. Een voorbeeld van een knelpunt met betrekking tot de activiteit zelf is een capaciteit-knelpunt. Zo'n knelpunt slaat op de verwerking van de activiteit. De informatie die bij de activiteit binnenkomt is dan misschien wel juist en volledig, maar het kan dan eenvoudigweg niet binnen een bepaalde tijd verwerkt worden. Dit ligt niet aan de invoer, maar aan de activiteit zelf. De activiteit veroorzaakt hierdoor waarschijnlijk knelpunten voor andere activiteiten, maar ervaart zelf ook een knelpunt, zonder dat zij daar een bepaalde invoer de schuld van kan geven. Wellicht betreft het hier een organisatorische fout, met als gevolg dat een activiteit overbelast is. Dit soort knelpunten, waarbij er dus wel een knelpunt wordt ervaren door een bepaalde activiteit zonder dat daar een andere activiteit als schuldige bij aangewezen kan worden, zijn weliswaar zeer belangrijk, maar zij vallen buiten het blikveld van deze prioriteitsstelling. De oorzaak van zo'n knelpunt ligt namelijk niet in een activiteit die onderdeel is van een bepaalde HIS. Uit de interviews zal zo'n knelpunt wel naar voren komen en genoteerd worden in de knelpuntennota. Hierbij kan dan als veroorzaker bijvoorbeeld de organisatie (in ieder geval iets anders dan een (combinatie van) activiteit(en)) aangegeven worden. Zij dienen door de informatieplanner als aparte knelpunten beschouwd te worden en niet in de te behandelen knelpuntsketens opgenomen te worden. Echter, het is mogelijk indien er een nieuwe activiteit wordt gecreëerd (in dit geval bijvoorbeeld de activiteit 'organisatie'), zonder deze aan een HIS te laten behoren, zodat het algoritme van de prioriteitsstelling daar niet op zal vastlopen wanneer er . Bij de prioriteitsstelling wordt namelijk gekeken bij welke HIS de in behandeling zijnde activiteit hoort. Wordt er geen HIS gevonden, dan wordt er ook geen prioriteit toegekend aan de keten in kwestie. Overigens wil dit alles niet zeggen dat knelpunten met betrekking tot de activiteit nooit een andere activiteit als schuldige kunnen aanwijzen. Als bijvoorbeeld een activiteit een knelpunt ervaart doordat hij verkeerd aangestuurd wordt (besturing-knelpunt) door bepaalde richtlijnen, dan kan de activiteit die de richtlijnen creëert als veroorzaker gezien worden. Zo'n activiteit is bijvoorbeeld onderdeel van de HIS 'bedrijfsplanning'. Het is ook mogelijk dat het ene knelpunt zwaarder weegt dan het andere. Een organisatorisch knelpunt is bijvoorbeeld moeilijker op te lossen dan een lees- en interpreteerbaarheid-knelpunt.

Op basis van de zwaarte en de geaardheid van knelpunten kunnen bepaalde toegewezen prioriteiten bijgesteld worden, maar dit dient pas achteraf te gebeuren. Bij het toekennen van prioriteiten abstraheer ik van de zwaarte en de geaardheid van knelpunten. Ik beschouw een knelpunt als een knelpunt, zoals die in de knelpuntennota is opgenomen. Ik beoog een voldoende geabstraheerde theorie te presenteren, zonder te verzanden in details.

2.1 De opbouw van de theorie

In deze paragraaf beschrijf ik de aspecten die mijns inziens de prioriteitsstelling bepalen. De verschillende aspecten zullen ter verduidelijking begeleid worden met voorbeelden.[3] Hiertoe heb ik een casus ontworpen. Bijlage I beschrijft drie mogelijke functionele eenheden van een bedrijf. In bijlage II wordt een aantal HISsen gegeven die uit de functionele eenheden zijn af te leiden met behulp van ARTIST. Hoe ik stapsgewijs tot die HISsen gekomen ben, is niet relevant. Zij dienen slechts ter illustratie. De voorbeelden die ik in deze paragraaf gebruik, zijn alle direkt gerelateerd aan deze functionele eenheden en HISsen. De definities van de in deze paragraaf genoemde activiteiten zijn bijvoorbeeld te vinden in bijlage I.

Allereerst zal ik een voorbeeld presenteren, dat later gebruikt zal worden.

Uit de interviews zijn de volgende knelpunten naar voren gekomen:

activiteit	knelpunt (met beschrijving)	veroorzaker
VA02	K01: gegevens zijn niet actueel van GV01	VA01
VA04	K02: gegevens zijn niet actueel van GV10	VA09
VA05	K03: gegevens zijn niet actueel van GV10	VA09
VA08	K04: slecht hanteerbaar verzendadvies VD06	VA04
VA02	K05: gegevens zijn niet actueel van GV07	FA05
FA05	K06: faktuur VD14 niet op tijd beschikbaar	VA08
VA07	K07: gegevens zijn niet actueel van GV07	FA05
AA01	K08: gegevens zijn niet actueel van GV01	VA01
VA01	K09: opgave VD03 niet op tijd beschikbaar	VA05

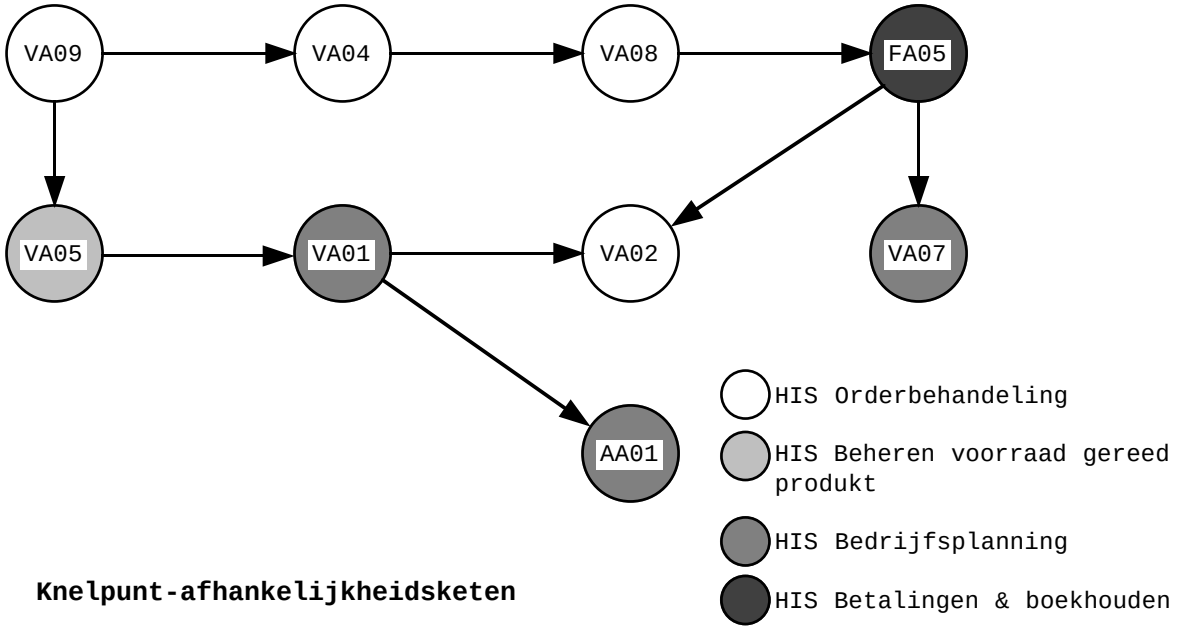
Knelpuntennota

Met behulp van een knelpunt-afhankelijkheidsmatrix zijn de onderlinge relaties in kaart te brengen:

[3] Uiteraard bewijzen de voorbeelden mijn theorie niet. Zij dienen uitsluitend ter illustratie van de verschillende stappen die ik neem. Of deze theorie geschikt is, moet blijken uit de praktijk.

	AA01	FA05	VA01	VA02	VA04	VA05	VA07	VA08	VA09
AA01			K08						
FA05								K06	
VA01						K09			
VA02		K05	K01						
VA04									K02
VA05									K03
VA07		K07							
VA08					K04				
VA09									

In een knelpunt-afhankelijkheidsmatrix is de activiteit op de rij de activiteit die een knelpunt ervaart. De activiteit op de kolom is de

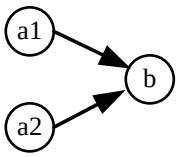


Knelpunt-afhankelijkheidsketen

activiteit die het knelpunt veroorzaakt en het knelpunt staat op de positie die aangegeven wordt door de twee activiteiten. Activiteit VA02 ervaart bijvoorbeeld het knelpunt K01, dat door activiteit VA01 veroorzaakt wordt. Deze matrix laat zich vertalen naar een knelpuntafhankelijkheids-keten. In deze keten geef ik ook aan bij welke HISsen de verschillende activiteiten horen. De HIS *Beheren voorraad gereed produkt* wordt niet in bijlage II vermeld, omdat deze voornamelijk op de functionele eenheid *Produktie* betrekking heeft en deze heb ik niet in beschouwing genomen. Er dient van uit gegaan te worden dat deze bestaat.

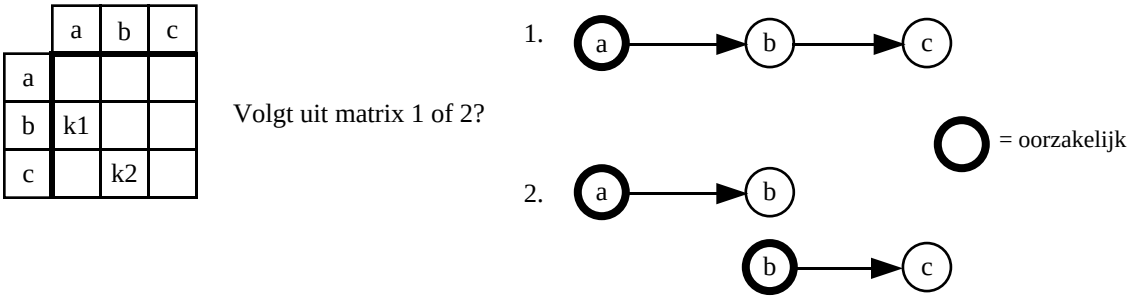
Tot zover de basis van het voorbeeld, dat na de beschrijving van de algoritmen gebruikt zal worden.

Elke knelpunt-afhankelijkheidsketen (voortaan knelpuntsketen) heeft één of meer beginpunten. Een beginpunt is een activiteit die als aanstichter van een reeks knelpunten beschouwd wordt. Een volgende activiteit heeft een knelpunt als gevolg van de eerste activiteit, waardoor daaropvolgende activiteiten eveneens een knelpunt ervaren. Nu kan het zijn dat meerdere activiteiten verantwoordelijk zijn voor een bepaald knelpunt. Stel dat activiteit *b* een knelpunt ervaart door twee activiteiten *a1* en *a2* (figuur 2.1). Stel daarnaast dat bij oplossing van het probleem bij *a1* het gehele knelpunt voor *b* verdwijnt, dan spreek ik van een AND-relatie, dus *a1* AND *a2* is verantwoordelijk voor het knelpunt van *b*. Als de oplossing van *a1* echter niet automatisch de oplossing van het knelpunt van *b* impliceert, dan spreek ik van een OR-relatie, dus *a1* OR *a2* is verantwoordelijk voor het knelpunt van *b*. Bij deze laatste relatie kan *b* immers ook nog gehinderd worden door *a2*. De keuze voor de AND- en de OR-relatie is logisch. Stel dat een activiteit de waarde 1 krijgt als het een knelpunt is en 0 als er geen knelpunt is, dan is de keuze al duidelijk. Immers, bij invulling van waarde 0 voor *a1* in het vorige voorbeeld (*a1* is opgelost), krijgt *a1* AND *a2* de waarde 0, dus geen knelpunt meer voor *b*. Evenzo voor de OR-relatie. Als *a1* de waarde 0 krijgt, wil dat nog niet zeggen dat *a1* OR *a2* ook de waarde 0 krijgt. Activiteit *a2* kan immers nog veroorzaker zijn voor het knelpunt bij *b*. Het aantal combinatie-mogelijkheden van AND- en OR-relaties is natuurlijk oneindig, hoewel het in de praktijk waarschijnlijk zeer beperkt blijft. Zelfs de AND-relatie op zichzelf komt bijna nooit voor. Een algemeen toepasbare theorie dient hier echter wel rekening mee te houden.



Figuur 2.1

Bij het analyseren van een organisatie kunnen er verscheidene knelpuntsketens naar voren komen. Als een bepaalde HIS een oorzakelijk knelpunt bevat, i.e. een knelpunt waardoor bij oplossing ervan een aantal andere knelpunten ook opgelost worden, dan dient de ontwikkeling van deze HIS voorrang te krijgen. Andere HISsen, die knelpunten bevatten als gevolg van het oorzakelijk knelpunt, zullen daar baat bij hebben. Hierdoor lijkt het logisch te onderzoeken welke HISsen oorzakelijke knelpunten bevatten en op basis daarvan een prioriteit toe te kennen. Er doet zich



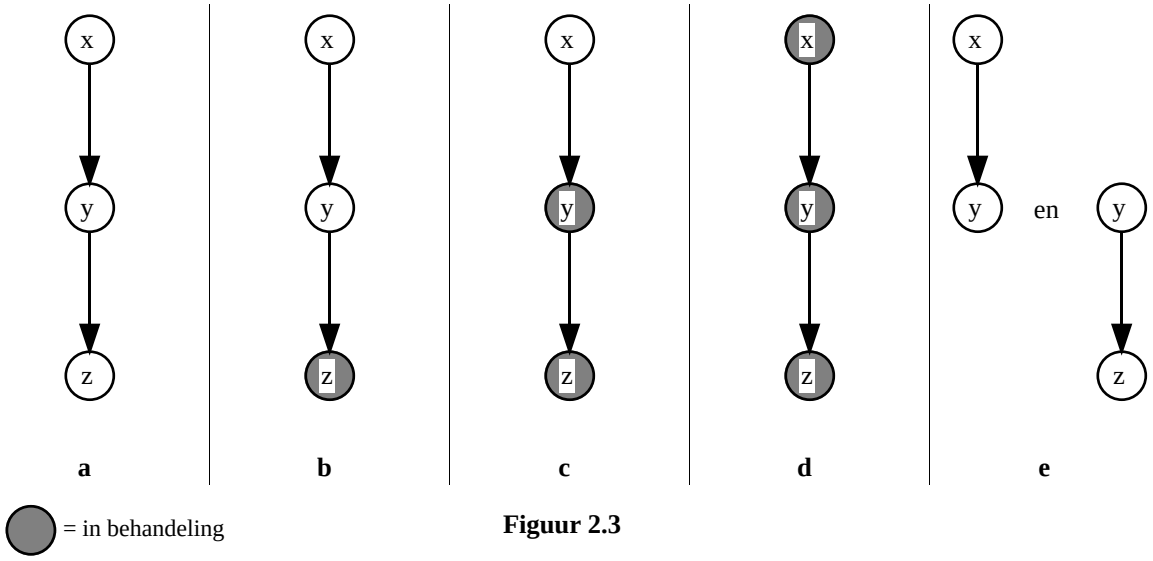
Figuur 2.2

echter een probleem voor als men alleen maar kijkt naar de beginpunten van de gemaakte knelpuntsketens. Doordat deze ketens zijn ontstaan uit de knelpunt-afhankelijkheidsmatrices, kan het voorkomen dat bepaalde activiteiten met de daarbij behorende knelpunten met elkaar in verband zijn gebracht, terwijl er in feite geen relatie hoeft te zijn. Als uit de matrix bijvoorbeeld blijkt dat activiteit *b* last heeft van activiteit *a* en dat activiteit *c* last heeft van activiteit *b*, dan vertaalt zich dat in de keten als a→b→c. Hierdoor lijkt het alsof oplossing van *a* ook oplossing voor *c* inhoudt, maar daar is misschien geen sprake van. Misschien is activiteit *b* wel alleen verantwoordelijk voor het knelpunt van activiteit *c*. Dit betekent dus dat het mogelijk is dat in de reeds gemaakte knelpuntsketens oorzakelijke knelpunten zich niet alleen aan het begin bevinden (zoals activiteit *a*), maar ook middenin (zoals activiteit *b*). Zie bijvoorbeeld figuur 2.2, waarbij uit de knelpunt-afhankelijkheidsmatrix niet duidelijk volgt of het geval 1 of geval 2 moet zijn. Bij het toekennen van prioriteiten wil ik graag alle oorzakelijke knelpunten kennen. De oplossing schuilt dan in het opsplitsen van de reeds gemaakte ketens in nieuwe ketens, waarbij de beginpunten oorzakelijk zijn voor alle knelpunten in zo'n nieuwe keten. Opsplitsing geschiedt in een aantal stappen, waarbij de keten uit figuur 2.3a als voorbeeld gebruikt wordt:

1. Pak een eindpunt van een bestaande keten (noem deze *z*; zie figuur 2.3b).
2. Stel het soort knelpunt vast van deze activiteit.
3. Loop tegen de richting van de keten in naar de veroorzaker van dit knelpunt (noem deze *y*; zie figuur 2.3c). Let op, als een activiteit meerdere knelpunten ervaart, dan dient per knelpunt de voorganger bepaald te worden. Elk knelpunt hoort in een andere keten thuis.
4. Bepaal van *y* de (waarschijnlijke) veroorzaker van het knelpunt dat *z* van *y* ervaart. Is er geen veroorzaker die er aan vooraf gaat, dan is *y* een oorzakelijk knelpunt. Is er wel een veroorzaker, dan dient van daaruit verder gegaan te worden met stappen 2, 3 en 4. Als *y* een beginpunt is van een reeds bestaande keten, dan heeft deze activiteit zeker geen veroorzaker. Als *y* geen beginpunt is van een reeds bestaande keten, dan ervaart *y* dus een knelpunt van een bepaalde activiteit (noem deze *x*; zie figuur 2.3d). Echter, het knelpunt dat *y* van *x* ervaart,

hoeft niet de oorzaak te zijn van het knelpunt dat *z* van *y* ervaart. Is dat het geval, dan zorgt deze stap er voor dat activiteit *y* het beginpunt wordt van een nieuwe keten en dat automatisch de oude keten afgekapt wordt bij activiteit *y*, i.e. activiteit *y* is een eindpunt geworden van de oude keten (en zal dus bij herhaling van deze stappen ook als zodanig beschouwd worden). Zie figuur 2.3e ter illustratie.

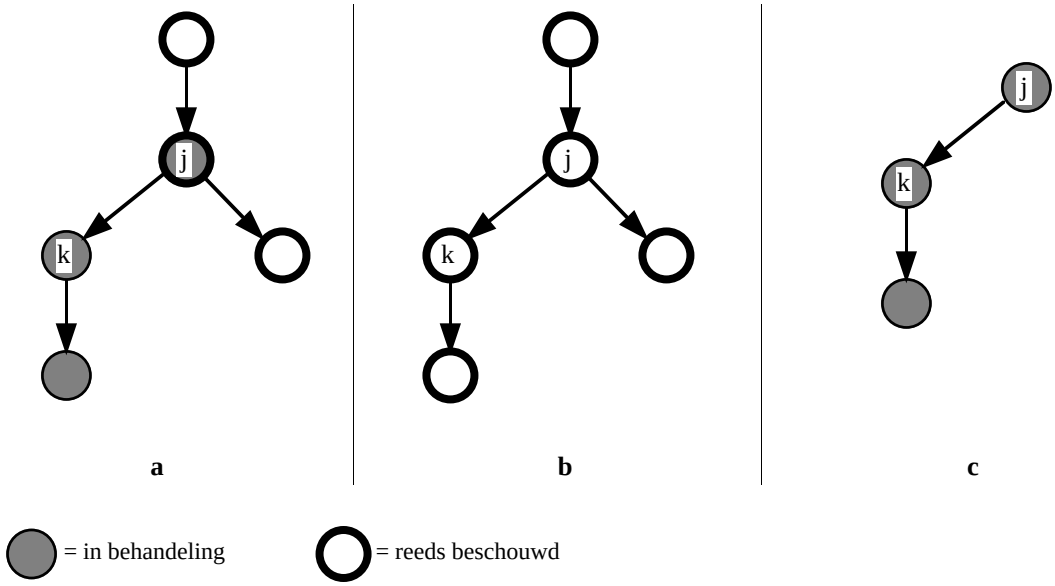
5. Herhaal deze stappen voor alle eindpunten van de oude ketens. Let wel, ook de keten die overblijft nadat er een stuk van afgesplitst is (en dus een nieuw eindpunt heeft gekregen), wordt als oude keten beschouwd.



Figuur 2.3

Zo te werk gaande heeft men dus een aantal ketens waarbij de oorzakelijke knelpunten veroorzakers zijn van alle knelpunten in zo'n keten. Voor alle ketens die gebruikt worden voor de prioriteitentoekenning moet gelden dat zij betrekking hebben op meerdere HISsen (zogenaamde inter-analyse). Dat wil dus zeggen dat niet alle activiteiten, die onderdeel zijn van een keten, aan één en dezelfde HIS mogen behoren. Het gaat namelijk om de invloed van een HIS op andere HISsen. Een HIS *h* krijgt dus alleen een hogere prioriteitswaarde als andere HISsen baat hebben bij oplossing van knelpunten behorende bij *h*. De knelpuntsketens die betrekking hebben op één HIS (zogenaamde intra-analyse) komen later in dit hoofdstuk nog aan bod. De knelpuntsketens waar nu mee gewerkt wordt, zijn de ketens uit de inter-analyse.

Aan de HISsen die de oorzakelijke knelpunten bevatten, kan nu een prioriteit toegekend worden. Bij elke HIS wordt simpelweg geteld hoeveel oorzakelijke knelpunten deze heeft. Voor elk oorzakelijk knelpunt gaat de waarde van de prioriteit van de bijbehorende HIS met één omhoog. Er



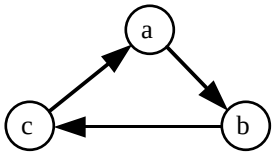
Figuur 2.4

doet zich echter een aantal problemen voor bij het bepalen van de nieuwe ketens en het toekennen van de waarden.

Ten eerste kan het voorkomen dat een veroorzaker (noem deze *j*) van een bepaald knelpunt van een activiteit (noem deze *k*), die tijdens stap 3 of stap 4 naar voren is gekomen, zich in een keten bevindt die reeds beschouwd is (zie figuur 2.4a). Blijkt dan dat het soort knelpunt van activiteit *k* overeenkomt met het soort knelpunt dat die keten kenmerkt en waardoor dus ook het zelfde pad gevolgd zou zijn, dan voegt men deze activiteit *k* en de reeks activiteiten die het gevolg zijn van *k* gewoon toe aan die keten (zie figuur 2.4b). Immers, het oorzakelijk knelpunt van die keten heeft dan ook invloed op het knelpunt dat activiteit *k* ervaart. Komt het soort knelpunt echter niet overeen, dan ontstaat er een nieuwe keten, waarvan de veroorzaker *j*, activiteit *k* en de reeks activiteiten als gevolg van *k* onderdeel worden (zie figuur 2.4c). Tot het moment dat er geen activiteiten meer zijn waarvoor de vijf stappen uitgevoerd kunnen worden, blijven de nieuwe ketens dus aan verandering onderhevig. Hier dient het begrip ‘veroorzaker’ breed beschouwd te worden. Een veroorzaker kan ook een aantal activiteiten in een AND- of een OR-relatie zijn. De rol van een veroorzaker zoals die voor een in behandeling zijnde activiteit geldt (in het voorbeeld de rol van *j* voor *k*), moet hetzelfde zijn als de rol die de veroorzaker in de reeds beschouwde knelpuntsketen speelt. Hiermee bedoel ik dat als de veroorzaker in de keten onderdeel is van bijvoorbeeld een OR-relatie, maar alleen verantwoordelijk is voor het knelpunt dat de in behandeling zijnde activiteit ervaart, er een nieuwe keten zal ontstaan (zoals bijvoorbeeld in figuur 2.4c), ongeacht de aard van het knelpunt. Andersom geldt dat als de in behandeling zijnde activiteit

een knelpunt ervaart door een aantal activiteiten in bijvoorbeeld een OR-relatie en één of meerdere, maar niet alle activiteiten van die relatie bevinden zich in een reeds beschouwde keten, er een nieuwe keten ontstaat. De mogelijkheid van opname van de in behandeling zijnde activiteit in een reeds beschouwde keten bestaat wel als deze activiteit een knelpunt ervaart door een aantal activiteiten in een bepaalde relatie en deze activiteiten spelen dezelfde rol, dus zitten in dezelfde relatie, in een reeds beschouwde keten.

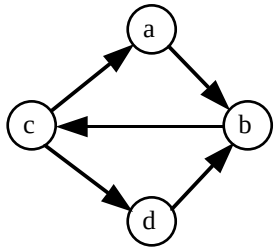
Ten tweede kunnen er cykels voorkomen in de knelpuntsketens. Wanneer bijvoorbeeld uit de knelpuntennota blijkt dat activiteit *c* een knelpunt ervaart door activiteit *b* en *b* ervaart een knelpunt door activiteit *a* en de laatste heeft op zijn beurt last van *c*, dan vertaalt zich dat in de keten van figuur 2.5. Voor het opsplitsen van de ketens moeten deze cykels reeds opgemerkt en doorbroken zijn. Het maakt op zich niet uit op welk punt zo'n cykel doorbroken wordt, want elk van de



Figuur 2.5

activiteiten die daarbij betrokken zijn, kan als aanstichter gezien worden. In een cykel zijn alle activiteiten aanstichter van een knelpunt en lijkt er geen oorzakelijke te zijn. Daarom moet er één aangewezen worden als aanstichter. In het voorbeeld van figuur 2.5 kan bijvoorbeeld activiteit *b* als aanstichter van de cykel beschouwd worden en dient de cykel verbroken te worden op dat punt, dus de verbinding tussen *a* en *b* verdwijnt. Bij het verbreken van een cykel moet rekening gehouden worden met de knelpunten die de cykel bevat. De verbinding verdwijnt namelijk alleen als voor elke activiteit die onderdeel is van de cykel geldt dat het knelpunt dat hij ervaart oorzakelijk is voor het knelpunt dat hij veroorzaakt. Is dat niet het geval, dan wordt de cykel niet als cykel opgemerkt bij het opsplitsen van de ketens, omdat dan de keten bewandeld wordt via activiteiten die telkens oorzakelijk zijn voor een bepaald knelpunt. De verbinding bij een punt waar dat het geval is, mag dan niet verdwijnen, omdat zij op een ander knelpunt betrekking heeft en bij het opsplitsen van de ketens in een andere keten ondergebracht wordt. Het eerste geval noem ik een **éénsoortige cykel** en het tweede geval een **meersoortige cykel**. In beide gevallen wordt er een eindpunt gedefinieerd, namelijk de activiteit in de cykel die zich voor de activiteit bevindt die op het reeds beschouwde pad ligt (wanneer *b* als beginpunt wordt beschouwd, zal dit *a* zijn).
Een cykel kan opgemerkt worden door telkens te kijken of een veroorzaker (of een activiteit die onderdeel is van een OR- of een AND-relatie) op het reeds bewandelde pad ligt, dus onderdeel is van reeds beschouwde activiteiten. Is dat het geval, dan ligt het voor de hand om op dat punt de

cykel te verbreken. Om cykels op te speuren en te elimineren, dient op alle ketens een algoritme losgelaten te worden dat vanaf elk beginpunt van een keten in de richting van de pijlen de keten doorloopt en vanaf elk eindpunt de keten tegen de richting van de pijlen in doorloopt, waarbij telkens gekeken wordt of een activiteit niet op het reeds bewandelde pad ligt. Elke keten moet een begin- en een eindpunt hebben. Is dat niet het geval, dan bevindt er zich een cykel aan het begin of het einde van de keten. Als een keten geen beginpunt kent, dus die keten bevat geen enkele activiteit zonder voorganger kent, maar wel een eindpunt, dan wordt de cykel opgemerkt door vanaf het eindpunt de keten te doorlopen. Bij het verbreken van de cykel wordt er een beginpunt gedefinieerd. Kent een keten geen eindpunt, maar wel een beginpunt, dan wordt de cykel opgemerkt door de keten te doorlopen vanaf het beginpunt en wordt er een eindpunt gedefinieerd. Wanneer een keten echter noch een begin- noch een eindpunt heeft, dan bestaat de hele keten uit een cykel (zoals bijvoorbeeld de keten uit figuur 2.5). In dat geval kan elke willekeurige activiteit als beginpunt beschouwd worden, waarvandaan de keten bewandeld kan worden. Daarmee is echter nog niet voldaan aan de eis dat elke keten zowel een begin- als een eindpunt moet hebben. Een beginpunt mag slechts gedefinieerd worden bij het opmerken van een cykel vanuit een eindpunt. Je kunt bij het verbreken van de cykel vanuit een eindpunt wel de activiteit waar dat gebeurt als beginpunt beschouwen. Deze kan dan nog onderdeel zijn van een andere cykel in de keten, hetgeen zeker opgemerkt zal worden bij een volgende ronde van het aflopen van de keten. Bij het opmerken van een cykel moet eerst gekeken worden of deze niet reeds is beschouwd, anders wordt er bij het aflopen van de keten vanuit een bepaald punt telkens een cykel gevonden (steeds dezelfde). Dit kan alleen voorkomen wanneer het een meersoortige cykel betreft, want dan verdwijnt er geen verbinding. Overigens kan een keten ook meerdere cykels hebben. Zie als voorbeeld figuur 2.6. Deze keten heeft geen begin- en geen eindpunt. Stel we beschouwen *a* als beginpunt, dan vervalt de verbinding tussen *c* en *a* (wanneer het een éénsoortige cykel betreft). Daarmee heeft de keten activiteit *c* als eindpunt, terwijl *c* nog onderdeel is van een andere cykel. Bij het opnieuw aflopen van de keten vanaf het nieuw gecreëerde beginpunt (in dit geval is dat activiteit *a*) wordt die tweede cykel opgemerkt. Hierbij wordt ook een breuk geforceerd (in dit geval zal dat zijn wanneer vanuit *d* naar *b* gelopen wordt, terwijl de laatste zich op het reeds beschouwde pad bevindt) indien het een éénsoortige cykel betreft, waarbij de activiteit waar dat gebeurt als eindpunt



Figuur 2.6

aangemerkt wordt (in dit geval zal dat *d* zijn). Het aflopen van de keten vanaf het begin- en het eindpunt herhaalt zich tot er geen cykels meer gevonden worden. Vervolgens kan met de opsplitsing van de ketens begonnen worden. Het feit dat een keten meerdere eindpunten heeft, die zich misschien midden in de keten bevinden (bijvoorbeeld *c*, die als eindpunt gedefinieerd is bij het aflopen van de keten, maar zich midden in de keten bevindt) vormt geen probleem bij het opsplitsen van de ketens. Vanuit elk eindpunt wordt de keten afgelopen. Als er een veroorzaker gevonden wordt die onderdeel is van een reeds beschouwde keten en het betreft hetzelfde knelpunt, dan worden de twee ketens gewoon aan elkaar geplakt (zie het voorbeeld van figuur 2.4).

De prioriteiten worden toegekend aan de HISsen waarvan de beginpunten van de ketens onderdeel zijn. Per keten wordt bekeken bij welke HIS het beginpunt hoort en de prioriteitswaarde van die HIS wordt met waarde één verhoogd. De waarde één is de maximale waarde waarmee de prioriteitswaarde van een HIS per keten verhoogd kan worden. Het kan namelijk voorkomen dat een keten meerdere beginpunten heeft (vanwege AND- en OR-relaties; dit wordt later uitgelegd). Elk beginpunt heeft door omstandigheden een bepaalde waarde gekregen en men dient deze per HIS op te tellen. Is de uiteindelijke waarde voor de HIS groter dan één, dan wordt de echte prioriteitswaarde van die HIS met één verhoogd. Is de opgetelde waarde gelijk of lager dan één, dan wordt de prioriteitswaarde van die HIS gewoon met deze waarde verhoogd.

Bij het toekennen van prioriteiten komt men een probleem tegen als blijkt dat een activiteit niet één maar meerdere veroorzakers heeft voor een bepaald knelpunt. Zoals reeds vermeld, kan een combinatie van activiteiten veroorzaker zijn van een knelpunt. Wanneer een activiteit alleen verantwoordelijk is voor een keten van knelpunten, dan dient dat uiteraard zwaarder te wegen dan wanneer meerdere activiteiten samen veroorzaker zijn. Zoals blijkt uit de AND- en OR-relaties, is een activiteit die onderdeel is van een AND-relatie in staat de gehele keten op te lossen, terwijl een activiteit in een OR-relatie het probleem slechts ten dele oplost. Dit heeft dan ook gevolgen voor de prioriteitentoekenning aan de ketens.
Stel dat de veroorzaker van een bepaald knelpunt bestaat uit *n* activiteiten in een OR-relatie. Oplossing van ieder bijbehorend knelpunten zegt niets over oplossing van de andere knelpunten uit de OR-relatie. De waardetoekenning van de prioriteiten is dan ook geen één meer voor de bijbehorende HIS (als zo'n knelpunt uit de OR-relatie oorzakelijk is), maar 1/n, omdat oplossing van het knelpunt slechts invloed heeft op 1/n-de deel van het probleem, m.a.w. elke activiteit van de OR-relatie is slechts in staat 1/n-de deel van het knelpunt weg te nemen. Deze waardetoekenning geschiedt alleen bij oorzakelijke knelpunten. Is een activiteit wel onderdeel van een OR-relatie, maar is hij niet oorzakelijk (de activiteit ervaart een knelpunt door een andere activiteit), dan krijgt de bij deze activiteit behorende HIS geen waarde toegekend en wordt er bij die keten gewoon verder gekeken naar de veroorzakers. De uiteindelijke waardetoekenning aan die keten is dan onafhankelijk van deze OR-relatie, indien het pad van de activiteit in de OR-relatie naar het

beginpunt zich over meerdere HISsen uitstrekt. De waardetoekenning aan dat beginpunt kan dan gewoon één zijn, doordat de oorzakelijke activiteit geen onderdeel meer is van een OR-relatie en tot aan de OR-relatie invloed heeft op meerdere HISsen. Wanneer echter het pad van de activiteit in de OR-relatie naar het beginpunt slechts activiteiten van dezelfde HIS bevat, krijgt dit beginpunt een waardetoekenning die afhankelijk is van de OR-relatie (dus in dit geval waarschijnlijk 1/n).
Als de veroorzaker van een bepaald knelpunt bestaat uit *n* activiteiten in een AND-relatie, dan krijgen alle bijbehorende HISsen de waardetoekenning één, omdat zij alle in staat zijn de hele knelpuntsketen weg te nemen. Let wel, alleen als de activiteit in een AND-relatie geen veroorzaker meer kent. Is dit echter wel het geval, dan wordt er gewoon verder gegaan met het analyseren van de keten via zo'n activiteit.

Nadat de prioriteitswaarden zijn bepaald aan de hand van de knelpuntsketens die betrekking hebben op meerdere HISsen (inter-analyse), wordt hetzelfde verhaal herhaald voor de knelpuntsketens die betrekking hebben op één HIS (intra-analyse). Alle HISsen krijgen dus eveneens een prioriteitswaarde die betrekking heeft op de knelpuntsketens die binnen een HIS blijven. Aan deze waarde mag niet de zelfde zwaarte worden toegedicht als aan de prioriteitswaarde die bij de inter-analyse naar voren kwam. Wanneer twee HISsen een gelijke inter-analyse-prioriteit hebben, is de intra-analyse-prioriteit bepalend. Als bijvoorbeeld HIS *h1* een inter-analyse-prioriteit 2 heeft en HIS *h2* ook, maar *h1* heeft een intra-analyse-prioriteit 5 en *h2* heeft een intra-analyse-prioriteit van 1, dan dient *h1* logischerwijs eerder ontwikkeld te worden dan *h2*. Wanneer bijvoorbeeld HIS *h1* een inter-analyse-prioriteit 2 heeft en HIS *h2* een inter-analyse-prioriteit 1, maar *h1* heeft een intra-analyse-prioriteit 1 en *h2* heeft een intra-analyse-prioriteit van 9, dan dient *h1* ook eerder ontwikkeld te worden dan *h2*. Vergelijk het met gouden en zilveren medailles. Eén gouden en nul zilveren telt zwaarder dan nul gouden en twintig zilveren. Uiteraard staat de inter-analyse hier voor goud en de intra-analyse voor zilver.

2.2 Algoritmen voor de prioriteitsstelling

In de vorige paragraaf heb ik de opbouw beschreven van de prioriteitsstelling zoals ik die voor ogen heb. In deze paragraaf presenteer ik de theorie in de vorm van een aantal algoritmen. Deze zijn geschreven in pseudo-code en daarmee in een dusdanige vorm gegoten dat het als formele notatie beschouwd mag worden. In de aanloop naar de algoritmen beschrijf ik de benodigde verzamelingen en functies die ik gebruik. Ik beschrijf hoe de verzamelingen zijn opgebouwd en wat de inhoud van de functies is. Hierbij beschouw ik de beschreven functies dermate triviaal, dat een beschrijving van de werking volstaat, zonder de functies in pseudo-code te presenteren. Dit zou ten koste gaan van de overzichtelijkheid en het haalt de aandacht weg van de hoofdzaak, namelijk het algoritme voor de prioriteitsstelling. Een ieder met kennis van algoritmen kan zich een voorstelling maken van de invulling van deze functies.

2.2.1 De verzamelingen

Om tot een goede bepaling van de prioriteiten te komen, moet een aantal zaken reeds bekend zijn. De algoritmen kunnen pas uitgevoerd worden nadat de knelpuntennota en de daaruit af te leiden knelpuntsketens zijn opgesteld. Eveneens dienen de HISsen te zijn geïdentificeerd. Uit de knelpuntennota kan men veel informatie halen. Zo kunnen daar de knelpunten uitgehaald worden, evenals de lasthebbende en veroorzakende activiteiten. Let wel, een veroorzaker van een knelpunt is niet noodzakelijkerwijs een alleenstaande activiteit. Deze kan namelijk uit meerdere activiteiten bestaan. Daarom moet daar een aparte verzameling voor aangelegd worden. Er moeten verzamelingen gedefinieerd worden voor knelpunten, activiteiten, knelpuntsketens, veroorzakers en HISsen. De definities van de verzamelingen geven een weergave van een element van de verzameling in kwestie, waarbij '…' achter een element wil zeggen dat de verzameling meer van dit soort elementen kan hebben.

knelpunten

De verzameling P is de verzameling van alle knelpunten, zoals die uit de knelpuntennota zijn af te leiden. Een knelpunt kent een code, een beschrijving en een soort. De code is uniek.

P = {kp_code(kp_beschrijving, kp_soort), …}

activiteiten

De verzameling A is de verzameling van alle activiteiten die onderdeel zijn van de functionele

eenheden. Een activiteit kent een unieke code en een beschrijving.

A = {act_code(act_beschrijving), …}

veroorzakers

De verzameling V is de verzameling van veroorzakers, zoals die afgeleid kunnen worden uit de knelpuntennota. Zoals reeds vermeld, hoeft een veroorzaker niet per se een alleenstaande activiteit te zijn. Het kan ook een aantal activiteiten in een AND- of een OR-relatie zijn (of een combinatie van beide relaties). Hetgeen tussen '[]' staat, is optioneel. Hoewel niet expliciet genoteerd, kan een element uit de OR-relatie ook uit een op zich staande AND-relatie bestaan. Evenzo kan een element uit de AND-relatie uit een op zich staande OR-relatie bestaan. Een veroorzaker kent een unieke code.

V = {ver_code({act_code [OR act_code …] [AND act_code …]}, …}

knelpuntsketens afgeleid uit de knelpuntennota

De verzameling O is de verzameling van knelpuntsketens, zoals die afgeleid kunnen worden uit de knelpuntennota. Een keten bestaat uit een aantal verbindingen tussen een veroorzaker en een activiteit, daarbij ook het knelpunt aangevend dat door zo'n activiteit ervaren wordt. Dit wordt weergegeven door de functie 'v(veroorzaker_code, activiteit_code, knelpunt_code)'. Ook kan het begin van een keten aangegeven worden door de functie 'begin(activiteit_code)' en het einde van een keten door de functie 'einde(activiteit_code)' (zie *De functies* voor een beschrijving van de functies). Een knelpuntsketen kent een unieke code. De ketens uit deze verzameling noem ik oude ketens, omdat zij de knelpuntsketens zijn die opgesplitst zullen worden in nieuwe ketens, waarop de prioriteitsstelling gebaseerd wordt.

O = {keten_code({v(ver_code, act_code, kp_code), …, begin(act_code), …,
einde(act_code), …}), …}

knelpuntsketens na opsplitsing van oude ketens

De verzameling N is de verzameling van knelpuntsketens die ontstaan na opsplitsing van de zogenaamde oude ketens. Een keten bestaat uit een aantal verbindingen tussen een veroorzaker en een activiteit, daarbij ook het knelpunt aangevend dat door zo'n activiteit ervaren wordt. Dit wordt weergegeven door de functie 'v(veroorzaker_code, activiteit_code, knelpunt_code)'. Ook kan het begin van een keten aangegeven worden door de functie 'begin(activiteit_code)' en het einde van een keten door de functie 'einde(activiteit_code)' (zie *De functies* voor een beschrijving van de functies). Een knelpuntsketen kent een unieke code. De ketens uit deze verzameling noem ik

nieuwe ketens, omdat zij ontstaan na opsplitsing van de oude ketens. Deze verzameling is als enige van de hier gedefinieerde verzamelingen bij aanvang leeg.

N = {keten_code({v(ver_code, act_code, kp_code), …, begin(act_code), …,
einde(act_code), …}), …}

HISsen

De verzameling H is de verzameling van hoofdinformatiestromen, zoals die in een eerder stadium van informatieplanning zijn geïdentificeerd. Hoe een element van deze verzameling er uitziet, is voor mij niet interessant, omdat ik alleen aan de HIS_code zal refereren. Om een idee te geven van zo'n element, heb ik ook hier een mogelijke manier beschreven van de opbouw van een HIS in deze verzameling. De functies die daarbij gebruikt worden, zijn niet beschreven bij *De functies*, maar ik hoop dat zij voor zich spreken. Zo geeft de functie 'lezen(act_code, geg_code)' bijvoorbeeld aan dat activiteit act_code gegevens uit een bepaalde gegevensverzameling geg_code leest. Een HIS kent ook een unieke code.

H = {HIS_code({uitvoer(act_code, doc_code), …, invoer(act_code, doc_code), …,
lezen(act_code, geg_code), …, schrijven(act_code, geg_code), …,
stimulus(stim_code, doc_code), …, in(ext_elem_code, doc_code), …,
uit(ext_elem_code, doc_code), …}), …}

2.2.2 De functies

Hier zal een aantal functies beschreven worden, die als hulpmiddel dienen bij de nog te beschrijven algoritmen. Ik volsta met het beschrijven van de functie, zonder een nadere specificatie van de inhoud in pseudo-code.

element van

Nu de verzamelingen zijn beschreven, moet het natuurlijk ook mogelijk zijn om te bepalen of iets deel uitmaakt van een bepaalde verzameling. Hiertoe introduceer ik een 'is element van'-functie. Deze functie geldt voor alle bovengenoemde verzamelingen, waarbij binnen een verzameling gekeken kan worden naar unieke codes. Uit de beschrijving van de verzamelingen blijkt dat elk element een unieke code heeft. Het is niet nodig om het volledige bijvoegsel van een element (hetgeen achter de unieke code tussen haakjes staat; in de volgende beschrijving is (…) daar de notatie van en de inhoud hangt af van de beschouwde verzameling) te beschrijven bij het zoeken

binnen een verzameling, vandaar dat deze functie geïntroduceerd wordt. Daarbij wordt de gewone ‘is element van’-functie uit de verzamelingen-leer gebruikt. Het symbool ‘V’ staat voor elke verzameling zoals beschreven onder *De verzamelingen*:

$$x \in V \quad \Leftrightarrow \quad \exists_{el\_code(\ldots) \in V} [el\_code = x]$$

Ook heb ik een functie nodig die bepaalt of een bepaalde activiteit onderdeel is van een bepaalde veroorzaker. Een veroorzaker kan, zoals gezegd, uit meerdere activiteiten bestaan.

$act\_code \in_V ver\_code$ ≡ act_code is een element van ver_code als ver_code slechts bestaat uit act_code of als act_code onderdeel is van een OR- of AND-relatie binnen ver_code.

Er moet bepaald kunnen worden of een activiteit, een veroorzaker, een verbinding, een begin- of een eindpunt onderdeel is van een bepaalde knelpuntsketen.

$act\_code \in_K keten\_code$ ≡ act_code is een element van keten_code als er een $v(act_1, act_2, kp)$ bestaat die onderdeel is van keten_code, waarbij act_2=act_code of als er een $v(ver, act_1, kp)$ bestaat die onderdeel is van keten_code, zodat $act\_code \in_V ver$.

$ver\_code \in_K keten\_code$ ≡ ver_code is een element van keten_code als er een v(ver, act, kp) bestaat die onderdeel is van keten_code, waarbij ver=ver_code.

$v(a_1, a_2, kp) \in_K keten\_code$ ≡ $v(a_1, a_2, kp)$ is een element van keten_code als $v(a_1, a_2, kp)$ een element is van de verzameling die keten_code beschrijft (het bijvoegsel).

$begin(act) \in_K keten\_code$ ≡ begin(act) is een element van keten_code als begin(act) een element is van de verzameling die keten_code

beschrijft (het bijvoegsel).

eind(act) $\in_K$ keten_code	≡	eind(act) is een element van keten_code als eind(act) een element is van de verzameling die keten_code beschrijft (het bijvoegsel).

Het is ook nodig dat bepaald kan worden of een activiteit tot een bepaalde HIS behoort.

act_code $\in_H$ HIS_code	≡	act_code is een element van HIS_code als er een functie bestaat die een element is van de verzameling die HIS_code beschrijft en waarvan act_code op zijn beurt weer onderdeel van is.

De laatste functie van de ‘is element van’-functies bepaald of een activiteit onderdeel is van een OR-relatie binnen een veroorzaker.

act_code $\in_{or}$ ver_code	≡	act_code is een element van een OR-relatie binnen ver_code wanneer ver_code bestaat uit a_1OR…ORa_n zodanig dat er een a_i ($1 \leq i \leq n$) bestaat zodat act_code=a_i of zodat a_i=a_{i1}AND…ANDa_{im} zodanig dat er een a_{ij} ($1 \leq j \leq m$) bestaat zodat act_code=a_{ij}.

bepaal

Deze functie kent een waarde toe aan de variabelen zodat aan de conditie (tussen vierkante haken) wordt voldaan. De conditie is altijd een ‘is element van’-functie.

BEPAAL variabele, … [conditie]	≡	kent waarde(n) toe aan de variabele(n), zodat aan de conditie voldaan wordt.

Tevens moet de bepaal-functie het i-de knelpunt dat een bepaalde activiteit ervaart, kunnen vaststellen. Hiertoe wordt een variabele kp(i) meegegeven en de desbetreffende activiteit. Een activiteit kan meerdere knelpunten ervaren.

BEPAAL (kp(i), act_code) ≡ kent het i-de knelpunt toe aan kp(i) dat door activiteit act_code ervaren wordt, zoals het in de knelpuntennota staat vermeld.

aantal elementen

Van een veroorzaker moet vastgesteld kunnen worden hoeveel elementen er in de OR-relatie zitten van die veroorzaker. Let wel, een AND-relatie binnen een OR-relatie wordt als één element gezien. Een AND-relatie is sterker dan een OR-relatie. Zo bestaat bijvoorbeeld a1 OR (a2 AND a3) OR a4 uit drie elementen.

AANTAL_ELEMENTEN (ver_code) ≡ levert het aantal elementen uit de OR-relatie van veroorzaker ver_code.

Idem voor het aantal elementen van een AND-relatie binnen een veroorzaker. Let wel, en OR-relatie binnen een AND-relatie wordt als één element gezien.

AANTAL_ELEMENTEN_AND (ver_code) ≡ levert het aantal elementen uit de AND-rela van veroorzaker ver_code.

aantal knelpunten

Een activiteit kan één of meer knelpunten ervaren. Dit aantal moet vastgesteld kunnen worden. Dit kan uit de knelpuntennota afgeleid worden.

AANTAL_KNELPUNTEN (act_code) ≡ levert het aantal knelpunten dat activiteit act_code ervaart.

cykel

Deze functie levert een verzameling van verbindingen (een verbinding is van de vorm v(ver, act, knelpunt)) vanaf een bepaalde verooorzaker tot de activiteit op een bepaald pad. Een pad is een lijst van verbindingen. Bij gebruik van deze functie is al vastgesteld dat er een cykel is en dat de activiteit onderdeel is van de veroorzaker.

CYKEL (ver_code, act_code, pad) ≡ levert een verzameling van verbindingen vanaf de veroorzaker ver_code tot de activiteit act_code en deze verzameling is een deelverzameling van *pad*. De verzameling die opgelevert wordt, is van de vorm

$\{v(ver\_code, a_1, kp_1), \ldots, v(a_n, act\_code, kp_n)\}$.

is gelijk
Twee knelpunten zijn aan elkaar gelijk als hun beschrijving en hun soort gelijk zijn. Deze staat in het bijvoegsel van het knelpunt.

kp_1 IS_GELIJK kp_2 ≡ levert TRUE op wanneer de beschrijvingen en de soorten van kp_1 en kp_2 overeenkomen.

is gevolg
Verreweg de belangrijkste functie voor het bepalen van prioriteiten is de ‘is gevolg van’-functie. Deze bepaalt of een bepaald knelpunt het gevolg is (of tenminste kan zijn) van een ander knelpunt. Met behulp van de menselijke logica valt daar wel een uitspraak over te doen. Zo kan een actualiteit-knelpunt zeer goed het gevolg zijn van een ander actualiteit-knelpunt, maar is het onwaarschijnlijk dat een leesbaarheid-knelpunt het gevolg is van een tijdigheid-knelpunt. Het blijft afhankelijk van de ervaring en het inzicht van de informatieplanner en daardoor lastig implementeerbaar. Hoe deze functie geïmplementeerd zou kunnen worden, komt in de volgende paragraaf aan de orde.

kp_1 IS_GEVOLG kp_2 ≡ levert TRUE op wanneer knelpunt kp_1 het gevolg is van knelpunt kp_2.

2.2.3 De algoritmen

Ik zal hier stapsgewijs de algoritmen beschrijven die op knelpuntsbasis prioriteiten toekennen aan de te ontwikkelen informatiesystemen. Het zijn drie losstaande algoritmen, zoals uit de vorige paragraaf eigenlijk al duidelijk werd. Voordat er prioriteiten kunnen worden toegekend, dienen de bestaande knelpuntsketens, i.e. de ketens zoals die uit de knelpuntennota zijn afgeleid, eerst opgesplitst te worden in ketens waarvan het beginpunt (of de beginpunten) ook echt oorzakelijk zijn voor de gehele keten. Daarvoor moeten echter de cykels uit de bestaande ketens worden gehaald. Dus het eerste algoritme beschrijft het elimineren van cykels, het tweede het opsplitsen van de ketens en het derde de eigenlijke prioriteitsbepaling. De vetgedrukte termen spreken hopelijk voor zich.

elimineren van cykels

Het verwijderen van cykels dient op elke keten toegepast te worden, daarom:

FOR ALL keten ∈ O
DO

Om te zien of een keten goed is doorlopen, dus vanaf een begin- en een eindpunt, zijn er twee variabelen nodig die dat bijhouden. Een keten wordt goedgekeurd als er vanaf een begin- en een eindpunt gezocht is en er geen cykels (meer) zijn gevonden. Voor dit laatste wordt eerst een derde variabele toegevoegd die een lijst van gevonden cykels bijhoudt. Later wordt er een vierde variabele geïntroduceerd, die TRUE wordt als er een cykel gevonden is:

BEGIN:=FALSE**;** EIND:=FALSE**;** CYKEL_LIJST:={}**;**
REPEAT
CYKEL_GEVONDEN:=FALSE**;**

Vanaf alle beginpunten moet de keten doorlopen worden, dus:

FOR ALL act ∈ A **WITH** [begin(act) $\in_K$ keten]
DO

Het algoritme moet nu een pad aflopen vanaf een beginpunt (als dit er is). Dat gebeurt met behulp van een procedure, die later beschreven wordt, waarin de globale variabele CYKEL_GEVONDEN gebruikt wordt:

PAD:={}**;** BEGIN:=TRUE**;** LOOP_VOORUIT(act, PAD)
ENDFOR;

Tevens moet de keten vanaf alle bestaande eindpunten doorlopen worden. Ook hier wordt een procedure gebruikt, die later beschreven wordt, waarin de globale variabele CYKEL_GEVONDEN gebruikt wordt:

FOR ALL act ∈ A **WITH** [eind(act) $\in_K$ keten]
DO
PAD:={}**;** EIND:=TRUE**;** LOOP_ACHTERUIT(act, PAD)
ENDFOR;

Als nu blijkt dat de keten geen begin- of eindpunt heeft (de variabelen BEGIN en EIND zijn dan nog steeds FALSE), kan er een willekeurige activiteit als voorlopig beginpunt beschouwd worden. Dit is dan geen echt beginpunt, maar gewoon het beginpunt van het zoekpad. Er zal dan vanzelf een cykel gevonden worden (en deze bestaat, anders had de keten wel een begin- en een eindpunt). Daarbij zal een eindpunt aangemerkt worden, dus bij de volgende repeat-loop zal er een eindpunt bestaan, waar vanuit wederom gezocht zal worden, daarbij een beginpunt creërend. Dit beginpunt zal bij de daaropvolgende repeat-loop gebruikt worden als startpunt voor een volgende zoektocht door de keten. Pas als er een begin- en eindpunt bestaan en er geen cykels meer gevonden worden, houdt de repeat-loop op:

IF NOT BEGIN **AND NOT** EIND
THEN
IF EXISTS act $\in$ A, ver $\in$ V, kp $\in$ P **SUCH THAT**

[v(ver, act, kp) $\in_K$ keten]

THEN
PAD:={}**;** LOOP_VOORUIT(act, PAD)
ENDIF
ENDIF

UNTIL BEGIN **AND** EIND **AND NOT** CYKEL_GEVONDEN
ENDFOR

De belangrijkste onderdelen van bovenstaand algoritme zijn natuurlijk de LOOP-procedures. Deze bepalen of een keten een cykel kent of niet. Er dient vanaf de meegegeven activiteit gezocht te worden. Deze activiteit moet in een veroorzakers-rol zitten. Dit betekent dat als zij onderdeel is van een veroorzaker met meerdere activiteiten, dan moet vanaf die oorzakelijke veroorzaker gezocht worden:

PROCEDURE LOOP_VOORUIT(act, PAD)**::**
FOR ALL x $\in$ A, ver $\in$ $V_{[act \in V\ ver]}$, kp $\in$ P **SUCH THAT**

[v(ver, x, kp) $\in_K$ keten]

DO

Pak de verbinding bij het pad (er wordt een nieuw pad gecreëerd, omdat er vanaf een beginpunt meerdere paden bewandeld kunnen worden). Kijk of er een verbinding bestaat die weer uitkomt op

het pad. Zo ja, dan is dit een cykel. De variabele C wordt de verzameling van verbindingen die tezamen de cykel vormen:

NPAD:=PAD∪{v(ver, x, kp)}**;**

IF EXISTS y ∈ A, v ∈ $V_{[x \in V\ v]}$, k ∈ P **SUCH THAT**

[v(v, y, k) ∈ NPAD]

THEN

C:=CYKEL(v, x, NPAD)**;**

Als de cykel zich nog niet in de lijst van gevonden cykels bevindt, dan wordt gekeken of voor elke activiteit die onderdeel is van de cykel geldt of het knelpunt dat hij ervaart, oorzakelijk is voor het knelpunt dat hij veroorzaakt. In dat geval wordt er een verbinding uit de keten gehaald. Is dat niet het geval, dan wordt er geen verbinding weggehaald uit de keten, omdat de cykel dan bij het opsplitsen van de ketens niet als cykel opgemerkt zal worden. Daar wordt namelijk gekeken naar de veroorzakers van één bepaald knelpunt per keer. In beide gevallen wordt er echter een eindpunt aangewezen:

IF NOT C ∈ CYKEL_LIJST

THEN

IF FOR ALL ac ∈ C **EXISTS** z ∈ A, ve ∈ V , v ∈ $V_{[ac \in V\ v]}$,

kp_1 ∈ P, kp_2 ∈ P **SUCH THAT**

[v(ve, ac, kp_1) ∈ NPAD **AND**

v(v, z, kp_2) ∈ NPAD **AND**

kp_2 IS_GEVOLG kp_1]

THEN

keten:=keten\{v(ver, x, kp)}**;**

keten:=keten∪{eind(act)}

ELSE

keten:=keten∪{eind(act)}

ENDIF;

CYKEL_LIJST:=CYKEL_LIJST∪C**;**

CYKEL_GEVONDEN:=TRUE

```
            ENDIF
        ELSE
            LOOP_VOORUIT(x, NPAD)
        ENDIF
    ENDFOR
    ENDPROC
```

De procedure LOOP_ACHTERUIT is vrijwel analoog aan LOOP_VOORUIT, vandaar dat ik deze zonder verder commentaar presenteer:

PROCEDURE LOOP_ACHTERUIT(act, PAD)**::**

FOR ALL ver ∈ V, kp ∈ P **SUCH THAT** [v(ver, act, kp) $\in_K$ keten]

DO

NPAD:=PAD∪{v(ver, act, kp)}**;**

IF EXISTS v ∈ V, y ∈ $A_{[y \in V\ ver]}$, k ∈ P **SUCH THAT** [v(v, y, k) ∈ NPAD]

THEN

C:=CYKEL(ver, y, NPAD)**;**

IF NOT C ∈ CYKEL_LIJST

THEN

IF FOR ALL ac ∈ C **EXISTS** z ∈ A, ve ∈ V , v ∈ $V_{[ac \in V\ v]}$,

$kp_1 \in P$, $kp_2 \in P$ **SUCH THAT**

[v(ve, ac, kp_1) ∈ NPAD **AND**

v(v, z, kp_2) ∈ NPAD **AND**

kp_2 IS_GEVOLG kp_1]

THEN

keten:=keten\{v(ver, act, kp)}**;**

keten:=keten∪{begin(act)}

ELSE

keten:=keten∪{begin(act)}

ENDIF;

CYKEL_LIJST:=CYKEL_LIJST∪C**;**

CYKEL_GEVONDEN:=TRUE

```
        ENDIF
    ELSE
        FOR ALL a ∈V ver
        DO
            LOOP_ACHTERUIT(a, NPAD)
        ENDFOR
    ENDIF
ENDFOR
ENDPROC
```

Nu zijn eventuele cykels uit de knelpuntsketens verwijderd en kan verder gegaan worden met het opsplitsen van de ketens.

opsplitsen van knelpuntsketens

Alle knelpuntsketens moeten opgesplitst worden, waarbij vanuit elk eindpunt de keten doorlopen wordt. Aangezien een eindpunt meerdere knelpunten kan ervaren, moet er voor elk knelpunt afzonderlijk een pad bewandeld worden. Er wordt vanuit de FOR-loop met drie variabelen gewerkt, namelijk NK(i), de i-de nieuwe keten, PAD(i), het pad dat bij het i-de knelpunt wordt gebruikt en kp(i), het i-de knelpunt van een bepaald eindpunt. Nadat vanuit een eindpunt de keten doorlopen is, wordt de nieuwe keten aan de verzameling N toegevoegd. Om te voorkomen dat er vanuit het zojuist behandelde eindpunt nogmaals door de keten gelopen wordt, verwijdert men de markering van de activiteit als eindpunt uit de keten:

```
FOR ALL keten ∈ O
DO
    FOR ALL act ∈ A SUCH THAT [eind(act) ∈K keten]
    DO
        FOR i = 1 TO AANTAL_KNELPUNTEN(act)
        DO
            NK(i):={}; PAD(i):={}; BEPAAL (kp(i), act);
            OPSPLITSEN(act, kp(i), PAD(i), NK(i));
            N:=N∪NK(i)
        ENDFOR;
        keten:=keten\{eind(act)}
    ENDFOR
ENDFOR
```

Ook hier draait het weer om de procedure in de FOR-loop, namelijk de procedure OPSPLITSEN. Van de meegegeven activiteit wordt de veroorzaker bepaald en aan het pad toegevoegd. Tevens wordt gekeken of de activiteiten die onderdeel zijn van deze veroorzaker niet als eindpunt zijn gemarkeerd. Dit kan gebeuren bij het opspeuren van cykels. Het is niet wenselijk om een eindpunt te behandelen dat eigenlijk geen eindpunt is, maar gewoon (onderdeel van) een veroorzaker is. Staat zo'n activiteit genoteerd als eindpunt, dan wordt deze markering uit de keten verwijderd:

PROCEDURE OPSPLITSEN(ac, kp, PAD, NK)**::**

BEPAAL ver $\in$ V [v(ver, ac, kp) $\in$ keten]**;**

PAD:=PAD$\cup${v(ver, ac, kp)}**;**

FOR ALL a $\in_V$ ver

DO

IF eind(a) $\in$ keten

THEN

keten:=keten\{eind(a)}

ENDIF

ENDFOR

Hierna dient gecontroleerd te worden of de veroorzaker niet al onderdeel uitmaakt van een reeds afgesplitste keten. Is dit het geval, dan moet, in het geval dat de veroorzaker in die keten een beginpunt is of uit meerdere activiteiten bestaat, het knelpunt dat de veroorzaker in die keten veroorzaakt, onderzocht worden. Is dit knelpunt gelijk aan het knelpunt dat de in behandeling zijnde activiteit ervaart door de veroorzaker, dan wordt deze activiteit met het reeds bewandelde pad toegevoegd aan die keten. Bestaat de veroorzaker uit één activiteit en is het geen beginpunt van die keten, dan moet gekeken worden naar het knelpunt dat de veroorzaker zelf ervaart in die keten. Is dat knelpunt een mogelijke veroorzaker voor het knelpunt dat de in behandeling zijnde activiteit ervaart door de veroorzaker, dan wordt de activiteit met het reeds bewandelde pad aan die keten toegevoegd. Is geen van de bovengenoemde gevallen van toepassing, dan gaat de procedure gewoon verder. Bij een toevoeging van de activiteit en het reeds bewandelde pad aan de keten waarin de veroorzaker zich reeds bevindt, wordt er een einde van de procedure geforceerd:

FOR ALL ket $\in$ N **SUCH THAT** [ver $\in_K$ ket]

DO

IF EXISTS a $\in_V$ ver **SUCH THAT** [begin(a) $\in_K$ ket] **OR**

```
        AANTAL_ELEMENTEN(ver)>1 OR
        AANTAL_ELEMENTEN_AND(ver)>1
    THEN
        BEPAAL x ∈ A, k ∈ P [v(ver, x, k) ∈K ket];
        IF kp IS_GELIJK k
        THEN
            ket:=ket∪PAD;
            keten:=keten\PAD;
            ENDPROC
        ENDIF
    ELSE
        BEPAAL x ∈ A[x ∈V ver], k ∈ P, v ∈ V [v(v, x, k) ∈K ket];
        IF kp IS_GEVOLG k
        THEN
            ket:=ket∪PAD;
            keten:=keten\PAD;
            ENDPROC
        ENDIF
    ENDIF
ENDFOR;
```

Als we op dit punt van de procedure belanden, is er blijkbaar geen toevoeging geweest aan een reeds beschouwde keten. Nu is het zaak te kijken naar de veroorzaker(s) van een mogelijk knelpunt dat door de veroorzaker van de in behandeling zijnde activiteit ervaren wordt. Laatstgenoemde veroorzaker kan uit meerdere activiteiten bestaan. Voor elk van deze activiteiten moet het knelpunt vastgesteld worden dat zij ervaart (zo deze bestaat). Wordt er een veroorzaker gevonden die een knelpunt veroorzaakt dat de oorzaak is (kan zijn) voor het knelpunt dat de in behandeling zijnde activiteit ervaart, dan wordt de keten via die veroorzaker verder afgelopen, omdat het pad dan nog steeds samenhangt. Wordt er een veroorzaker gevonden die een knelpunt veroorzaakt dat niet de oorzaak is (kan zijn) voor het knelpunt dat de in behandeling zijnde activiteit ervaart, dan wordt de keten hier afgebroken, omdat het pad dan niet meer samenhangt. Wanneer er echter geen veroorzaker blijkt te zijn, dan is er blijkbaar een beginpunt gevonden en wordt dit in de nieuwe keten genoteerd. Het gehele pad wordt onderdeel van de nieuwe keten:

FOR ALL a ∈ A[a ∈V ver]

```
DO
    IF NOT EXISTS k ∈ P, v ∈ V [v(v, a, k) ∈K keten]
    THEN
        NK:=NK∪PAD∪begin(a);
        keten:=keten\PAD
    ELSE
        FOR ALL k ∈ P, v ∈ V [v(v, a, k) ∈K keten]
        DO
            IF kp IS_GEVOLG k
            THEN
                OPSPLITSEN(a, k, PAD, NK)
            ELSE
                NK:=NK∪PAD∪begin(a);
                keten:=keten\PAD;
                FOR ALL b ∈ A[b ∈V v]
                DO
                    keten:=keten∪eind(b)
                ENDFOR
            ENDIF
        ENDFOR
    ENDIF
ENDFOR
ENDPROC
```

Nu bestaat de verzameling N uit knelpuntsketens waarvan de beginpunten oorzakelijk zijn voor de gehele keten. Dit is het gevolg van de IS_GEVOLG-functie, die bij het opsplitsen gehanteerd werd. Hierna kan met het toekennen van de prioriteiten begonnen worden.

toekennen van prioriteiten

Alle ketens uit de verzameling N moeten onderzocht worden. Een HIS kan twee soorten prioriteiten toegekend krijgen, namelijk een INTRA_PRIOTITEIT, i.e. de prioriteit die betrekking heeft op knelpuntsketens die zich geheel in de desbetreffende HIS bevinden, en een INTER_PRIORITEIT, i.e. de prioriteit die wordt toegekend aan een HIS wanneer het een keten betreft die zich over meerdere HISsen uitstrekt. Als er een keten beschouwd wordt die zich in zijn geheel binnen één

HIS bevindt, dan wordt de INTRA_PRIORITEIT van die HIS met waarde één verhoogd, omdat de HIS vanzelfsprekend alleen verantwoordelijk is voor die keten. Er kan zich echter een probleem voordoen. Bij het opsplitsen van de knelpuntsketens in ketens die geheel oorzakelijk zijn, kan het voorkomen dat een bepaalde activiteit meerdere veroorzakers kent en daardoor één knelpunt veroorzaakt. Let wel, deze veroorzakers zitten niet in een OR- of AND-relatie, omdat een veroorzaker zelf een OR- of AND-relatie is (of een alleenstaande activiteit). Met meerdere veroorzakers bedoel ik dus meerdere (groepen van) activiteiten die een aantal knelpunten veroorzaken voor een andere activiteit, zodat deze op haar beurt weer een knelpunt veroorzaakt. Dit betreft dan wel een knelpunt dat het gevolg is van die veroorzakende knelpunten, anders zou het niet in de nieuwe keten zijn opgenomen. Stel bijvoorbeeld dat activiteit *b* een lees- en interpreteerbaarheid-knelpunt ervaart door veroorzaker a_1 en een onvolledigheid-knelpunt door veroorzaker a_2. Activiteit *b* veroorzaakt op haar beurt weer een tijdigheid-knelpunt voor activiteit *c*. Dit laatste knelpunt kan zowel het gevolg zijn van het lees- en interpreteerbaarheid- als het onvolledigheid-knelpunt, vandaar dat zij alle in de nieuwe keten zijn opgenomen. De beide veroorzakers a_1 en a_2 zitten niet samen in een OR-relatie, terwijl ze toch allebei opgelost moeten worden, wil het knelpunt voor *b* en daarmee voor *c* verdwijnen. De oplossing ligt voor de hand: breng dit soort gevallen onder in een OR-relatie. Er moet dus een nieuwe veroorzaker gemaakt worden, waarvan dit soort veroorzakers (in het voorbeeld a_1 en a_2) in een OR-relatie onderdeel worden. Hoe het knelpunt genoemd wordt dat deze overlappende veroorzaker dan veroorzaakt, is niet echt van belang, omdat hier bij het toekennen van de prioriteiten niet meer gekeken wordt naar het soort knelpunt. Wil men echter achteraf de prioriteiten nog aanpassen op grond van de zwaarte en geaardheid van de knelpunten, dan dienen er copieën van de ketens gemaakt te worden alvorens er nieuwe veroorzakers aan toegevoegd worden. Deze kunnen dan gebruikt worden voor eventuele aanpassingen:

FOR ALL keten ∈ N
DO
    FOR ALL a ∈ $A_{[a \in K \text{ keten}]}$
    DO
        IF EXISTS ver ∈ V, k ∈ P [v(ver, a, k) ∈$_K$ keten]
        THEN
            nver:=Ø**;**
            FOR ALL ver ∈ V, kp ∈ P [v(ver, a, kp) ∈$_K$ keten]
            DO
                IF nver=Ø

THEN
nver:=ver
ELSE
nver:=nver OR ver
ENDIF;
keten:=keten\{v(ver, a, kp)}
ENDFOR;
keten:=keten∪{v(nver, a, k)}
ENDIF
ENDFOR;
IF EXISTS h ∈ H **SUCH THAT FOR ALL** a ∈ $A_{[a \in K \text{ keten}]}$ [a ∈ H h]
THEN
INTRA_PRIORITEIT(h):=INTRA_PRIORITEIT(h) **INCR** 1

Als het een keten betreft die zich over meerdere HISsen uitstrekt, dan moet er voor elk beginpunt een prioriteit worden toegekend aan de bijbehorende HIS, mits zo'n beginpunt tot een bepaalde HIS behoort. Is dat niet het geval, dan hoeft er geen prioriteit te worden toegekend, omdat het alleen prioriteiten betreft die betrekking hebben op de ontwikkeling van informatiesystemen en deze zijn gebaseerd op de HISsen. Wat er dan wel moet gebeuren, valt buiten de beschouwing van deze scriptie. Het betreft dan knelpuntsketens die veroorzaakt worden door activiteiten die geen onderdeel zijn van toekomstige informatiesystemen en ik wil alleen prioriteiten toekennen aan de ontwikkeling van informatiesystemen. Besloten kan worden om bij het definiëren van de HISsen de niet gebruikte activiteiten bij een bepaalde HIS onder te brengen, zodanig dat de functionaliteit van de activiteiten gehandhaafd blijft. Ik zal daar niet verder op ingaan. Voor het toekennen van prioriteiten aan activiteiten (dus eigenlijk aan HISsen) die wel bij een bepaalde HIS horen, wordt een variabele WAARDE gebruikt. Deze variabele heeft bij aanvang de waarde één, omdat de uiteindelijke prioriteit één of minder is die aan een beginpunt wordt toegekend. Tevens wordt ook hier een pad bijgehouden. De variabele PRIORITEIT wordt verhoogd met de waarde van WAARDE nadat de procedure AFLOPEN is uitgevoerd. Deze procedure wordt later beschreven:

ELSE
FOR ALL a ∈ A [begin(a) ∈ K keten]
DO
IF EXISTS h ∈ H [a ∈ H h]
THEN
WAARDE:=1**;** PAD:={}**;**

```
                    AFLOPEN(a, WAARDE, PAD, keten);
                    PRIORITEIT(h):=PRIORITEIT(h) INCR WAARDE
                ENDIF
            ENDFOR;
```

Nu is voor elke HIS die een beginpunt bevat van de in behandeling zijnde keten een prioriteit bepaald. Het kan echter voorkomen dat een bepaalde HIS bij één keten een prioriteit heeft die hoger is dan de waarde één. Dit kan gebeuren als bijvoorbeeld verschillende beginpunten van een keten bij één HIS horen en zich in een AND-relatie bevinden. Een HIS kan echter hooguit in zijn geheel verantwoordelijk zijn voor één keten en de prioriteit bij die keten mag dan ook hoogstens één zijn. Vandaar dat voor elke HIS gekeken moet worden of de prioriteit bij die keten niet hoger is dan één. In dat geval wordt de waarde van die prioriteit gewoon op één gezet. Daarna vindt de feitelijke toekenning van prioriteiten plaats (de INTER_PRIORITEIT wordt dan pas verhoogd):

```
            FOR ALL h ∈ H
            DO
                IF PRIORITEIT(h)>1
                THEN
                    PRIORITEIT(h):=1
                ENDIF;
                INTER_PRIORITEIT(h):=INTER_PRIORITEIT(h)
                                        INCR PRIORITEIT(h);
                PRIORITEIT(h):=0
            ENDFOR
        ENDIF
    ENDFOR
```

De procedure AFLOPEN is belangrijk. Deze loopt de keten af vanaf een bepaald punt, waarbij gekeken wordt of dat punt zich in een OR-relatie bevindt. Is dat het geval, dan wordt het reeds bewandelde pad gecontroleerd. Valt het gehele pad binnen één HIS, dan wordt de meegegeven variabele WAARDE gedeeld door het aantal elementen van de desbetreffende OR-relatie, omdat dat bepaalde punt dan slechts verantwoordelijk is voor 1/n-de van het probleem, waarbij n staat voor het aantal elementen in de OR-relatie. Strekt het reeds bewandelde pad zich echter uit over meerdere HISsen, dan beschouw ik het als een op zich staande keten. Dat bepaalde punt is dan namelijk verantwoordelijk voor een reeks van knelpunten die zich over meerdere HISsen uitstrekt. De waarde van WAARDE blijft dan onveranderd. De in deze procedure gebruikte functie DIV is een zuivere deel-functie, dus zonder rest-gedeelte. Zo is 24 DIV 5 gewoon 4.8 en niet 4 rest 4:

PROCEDURE VOLGENDE(a, WAARDE, PAD, keten)**::**

PAD:=PAD∪{a}**;**

IF EXISTS ver ∈ V **SUCH THAT** [a $\in_{or}$ ver **AND** ver $\in_{K}$ keten]

THEN

IF EXISTS h ∈ H **SUCH THAT FOR ALL** a ∈ PAD [a $\in_{H}$ h]

THEN

WAARDE:=WAARDE **DIV** AANTAL_ELEMENTEN(ver)

ENDIF

ENDIF;

Nu moet de in behandeling zijnde keten verder bewandeld worden. De activiteit in kwestie kan veroorzaker zijn van meerdere paden in dezelfde keten:

FOR ALL v ∈ $V_{[a \in V v]}$, b ∈ A, k ∈ P **SUCH THAT** [v(v, b, k) $\in_{K}$ keten]

DO

AFLOPEN(b, WAARDE, PAD, keten)

ENDFOR

ENDPROC

Na het uitvoeren van dit laatste algoritme hebben alle HISsen een inter- en een intra-prioriteit gekregen. Met deze twee prioriteiten is een volgorde aan te geven waarin de HISsen ontwikkeld dienen te worden tot informatiesystemen. Zoals reeds eerder vermeld, is de inter-prioriteit de belangrijkste. Bij gelijke inter-prioriteiten kan gekeken worden naar de intra-prioriteit. Deze dient dan voor een volgorde van HISsen met gelijke inter-prioriteiten. De uiteindelijke volgorde waarin informatiesystemen ontwikkeld dienen te worden, is gemaakt op basis van knelpunten die aan activiteiten kleven, die op hun beurt weer onderdeel zijn van HISsen binnen de organisatie. Of de informatiesystemen uiteindelijk ook in deze volgorde ontwikkeld zullen worden, is maar zeer de vraag. Er spelen nog vele aspecten mee, vooral financiële en organisatorische, waardoor de uiteindelijke volgorde anders uit kan pakken. Ik heb hier echter een objectieve methode gepresenteerd, waardoor subjectiviteit bij het maken van beslissingen omtrent de ontwikkeling van informatiesystemen enigszins ingedamd wordt.

2.3 Een knelpunten-matrix bij het algoritme

Zoals uit de vorige paragraaf al bleek, is de interventie van de menselijke logica noodzakelijk voor het komen tot een juiste priorteitsstelling. Ik gaf echter aan dat de functie waarbij die interventie plaatsvindt, de IS_GEVOLG-functie, wellicht op een bepaalde manier geïmplementeerd zou kunnen worden. Dat zou de menselijke logica elimineren, omdat dan de beslissingen door het algoritme genomen zouden worden. In deze paragraaf richt ik mij op een mogelijke manier van implementatie.
Er is al gebleken dat er verschillende soorten knelpunten bestaan. Er bestaan actualiteit-knelpunten, beschikbaarheid-knelpunten, etc. Een idee voor de implementatie van de IS_GEVOLG-functie is om te kijken naar welke soorten knelpunten het gevolg kunnen zijn van welke andere soorten. Allereerst dient er een overzicht te zijn van de mogelijke soorten knelpunten. Ik presenteer hier een lijst van soorten, die echter nog aangevuld kan worden. Om het geheel werkbaar te houden, heb ik mij beperkt tot acht soorten knelpunten. Het werkelijke aantal is zeker het dubbele. De hoofdletters in de matrix van figuur 2.7 komen overeen met de hoofdletters achter de naam in de lijst.

- Actualiteit (A) - informatie is niet up to date
- Tijdigheid (T) - informatie komt niet op tijd aan
- Lees- en interpreteerbaarheid (L) - informatie is niet duidelijk lees- of interpreteerbaar
- Juistheid (J) - informatie is onjuist
- Onvolledigheid (O) - te weinig gegevens ter verwerking
- Capaciteit (C) - activiteit kan het werk niet aan
- Beschikbaarheid (B) - geen autorisatie om over de juiste informatie te beschikken
- Overbodigheid (H) - te veel gegevens ter verwerking

Uit de lijst blijkt dat het knelpunten betreft met betrekking tot de informatie, op één enkele uitzondering na. Alleen het capaciteit-knelpunt heeft betrekking op de activiteit zelf. Deze scheiding kwam al naar voren in de inleiding van dit hoofdstuk. Bij zo'n knelpunt is er ook geen activiteit aan te wijzen die de veroorzaker is, tenzij er kunstgrepen toegepast worden, waardoor bijvoorbeeld de organisatie als activiteit beschouwd wordt. Knelpunten die betrekking hebben op de activiteit zelf zijn veelal het gevolg van organisatorische fouten.

Het probleem is nu om in te zien welke knelpunten het gevolg kunnen zijn van welke andere knelpunten. Het lijkt waarschijnlijk dat een lees- en interpreteerbaarheid-knelpunt niet het gevolg kan zijn van een tijdigheid-knelpunt, terwijl een tijdigheid-knelpunt wel het gevolg kan zijn van een lees- en interpreteerbaarheid-knelpunt. Zo kunnen de verschillende soorten knelpunten in een

	A	T	L	J	O	C	B	H
A	j	j	j	j	j	j	j	j
T	j	j	j	j	j	j	j	j
L	n	n	j	n	n	j	n	j
J	j	n	j	j	n	j	j	j
O	j	j	j	j	j	j	j	j
C	n	n	n	n	n	n	n	r
B	n	n	n	n	j	n	j	r
H	j	n	n	n	n	j	n	j

Figuur 2.7

knelpunten-matrix tegen elkaar afgezet worden. Figuur 2.7 toont zo'n matrix met de bijbehorende invulling. De soorten op de rij zijn het gevolg van de soorten op de kolom. Een *j* op de positie die aangegeven wordt door twee soorten geeft aan dat het soort knelpunt op de rij het gevolg kan zijn van het soort knelpunt op de kolom. Een *n* geeft aan dat het niet het gevolg kan zijn.

Bijvoorbeeld: Een lees- en interpreteerbaarheid-knelpunt kan niet het gevolg zijn van een actualiteit-knelpunt (*L* op de rij, *A* op de kolom, positie geeft *n* aan).

Het geven van een invulling aan deze matrix is zeer lastig. Het is gebaseerd op waarschijnlijkheid. Het is bijvoorbeeld waarschijnlijk dat een juistheid-knelpunt niet het gevolg kan zijn van een onvolledigheid-knelpunt, maar het is niet uitgesloten dat het mogelijk is. Men mag dan ook niet blind varen op zo'n matrix. Het dient als hulpmiddel voor de informatieplanner en als beslissingstabel bij een volledige implementatie van de algoritmen. Bij het raadplegen van zo'n matrix door de geïmplementeerde algoritmen kan bijvoorbeeld de gebruiker (de informatieplanner) gewaarschuwd worden. Daarbij wordt de te nemen beslissing en de waarschijnlijke oplossing (zoals uit de matrix blijkt) op het scherm getoond. Of dit dan ook in een specifiek geval de juiste oplossing is, kan de informatieplanner zelf beslissen.

Opvallend in de matrix is de rij *C*. Hierin komt alleen de invulling *n* voor, wat duidt op het feit dat een capaciteit-knelpunt (waarschijnlijk) niet het gevolg is van een ander soort knelpunt. Dit betekent eveneens dat dit knelpunt aan het begin van een keten moet staan, omdat er geen veroorzaker mogelijk is (anders dan de organisatie).

Een matrix als in figuur 2.7 is eenvoudig te implementeren. Daarmee is de IS_GEVOLG-functie

eveneens geïmplementeerd. Om een betrouwbare werking van de functie te krijgen, dienen van een groot aantal informatieplanningsprojecten de beslissingen omtrent de oorzaken en gevolgen van knelpunten onderzocht te worden. Daaruit moet blijken door welke soorten knelpunten andere knelpunten meestal veroorzaakt worden. Hieruit kan een redelijk betrouwbare invulling van een knelpunten-matrix afgeleid worden.

Als slotopmerking wil ik nogmaals duidelijk stellen dat ik met deze paragraaf niet een volledige knelpunten-matrix met een onbetwistbare invulling heb willen geven. Ik presenteer hier een mogelijkheid voor de volledige implementatie van de algoritmen.

2.4 Een voorbeeld uitgewerkt

Om de werking van de algoritmen enigszins te verduidelijken, werk ik in deze paragraaf het voorbeeld uit dat in §2.1 is gepresenteerd. Met negen knelpunten is dit natuurlijk maar een zeer beperkt voorbeeld, maar wel bruikbaar als verduidelijking. Vooral het algoritme voor het opsplitsen van de ketens zal hier aan de orde komen. Het algoritme voor het verwijderen van cykels zal niet behandeld worden, omdat de knelpuntsketen uit het voorbeeld geen cykels bevat. Het toepassen van het algoritme voor de prioriteitsstelling wordt, hoewel niet erg opwindend, wel behandeld. Ter illustratie van de knelpunten-matrix uit de vorige paragraaf zal de IS_GEVOLG-functie op basis van die matrix uitgevoerd worden.

Allereerst dient de knelpuntennota vertaald te worden naar een keten:

```
KETEN={v(VA01,VA02,K01), v(VA09,VA04,K02), v(VA09,VA05,K03),
v(VA04,VA08,K04), v(FA05,VA02,K05), v(VA08,FA05,K06),
v(FA05,VA07,K07), v(VA01,AA01,K08), v(VA05,VA01,K09), begin(VA09),
eind(VA07), eind(VA02), eind(AA01)}
```

Het beginpunt wordt gevonden doordat die activiteit geen veroorzaker kent. De eindpunten worden gevonden doordat die activiteiten geen knelpunt veroorzaken. Op elk eindpunt wordt het algoritme losgelaten. Wanneer blijkt dat de veroorzaker geen onderdeel is van een reeds beschouwde keten, dan wordt er niets vermeld.

```
eindpunt=VA07;
soort knelpunt(K07)=actualiteit;
veroorzaker(VA07,K07)=FA05;

FA05 ervaart knelpunt K06;
soort knelpunt(K06)=tijdigheid;
K07 IS_GEVOLG K06;
veroorzaker(FA05,K06)=VA08;

VA08 ervaart knelpunt K04;
soort knelpunt(K04)=lees- en interpreteerbaarheid;
K06 IS_GEVOLG K04;
veroorzaker(VA08,K04)=VA04;
```

```
VA04 ervaart knelpunt K02;
soort knelpunt(K02)=actualiteit;
NOT K04 IS_GEVOLG K02
```

Nu wordt de oude keten (knelpuntsketen *KETEN*) afgebroken, omdat blijkt dat het knelpunt dat *VA04* ervaart niet oorzakelijk kan zijn voor het knelpunt dat hij veroorzaakt. Het tot dan toe bewandelde pad wordt een nieuwe keten en wordt afgesplitst van de oude keten.

```
NIEUWE_KETEN1={v(FA05,VA07,K07), v(VA08,FA05,K06),
v(VA04,VA08,K04), begin(VA04)};
KETEN={v(VA01,VA02,K01), v(VA09,VA04,K02), v(VA09,VA05,K03),
v(FA05,VA02,K05), v(VA01,AA01,K08), v(VA05,VA01,K09), begin(VA09),
eind(VA02), eind(AA01), eind(VA04)}
```

Nu wordt er een nieuw eindpunt in behandeling genomen.

```
eindpunt=VA02
```

Dit eindpunt kent twee knelpunten. Elk knelpunt wordt apart behandeld.

```
soort knelpunt(K01)=actualiteit;
veroorzaker(VA02,K01)=VA01;

VA01 ervaart knelpunt K09;
soort knelpunt(K09)=tijdigheid;
K01 IS_GEVOLG K09;
veroorzaker(VA01,K09)=VA05;

VA05 ervaart knelpunt K03;
soort knelpunt(K03)=actualiteit;
K09 IS_GEVOLG K03;
veroorzaker(VA05,K03)=VA09;

VA09 ervaart geen knelpunt
```

Omdat *VA09* geen knelpunt ervaart (en daarom geen veroorzaker kent), wordt het tot dan toe bewandelde pad uit de oude keten verwijderd en als nieuwe keten gedefinieerd.

```
NIEUWE_KETEN2={v(VA01,VA02,K01), v(VA05,VA01,K09),
v(VA09,VA05,K03), begin(VA09)};
KETEN={v(VA09,VA04,K02), v(FA05,VA02,K05), v(VA01,AA01,K08),
begin(VA09), eind(VA02), eind(AA01), eind(VA04)}
```

Het tweede knelpunt dat *VA02* ervaart wordt nu in behandeling genomen.

```
soort knelpunt(K05)=actualiteit;
veroorzaker(VA02,K05)=FA05;

FA05 is onderdeel van NIEUWE_KETEN1;
FA05 ervaart binnen NIEUWE_KETEN1 knelpunt K06;
soort knelpunt(K06)=tijdigheid;
K05 IS_GEVOLG K06
```

Het tweede knelpunt dat *VA02* ervaart blijkt het gevolg te zijn van het knelpunt dat *FA05* binnen NIEUWE_KETEN1 ervaart, dus *VA02* wordt opgenomen in NIEUWE_KETEN1.

```
NIEUWE_KETEN1={v(FA05,VA07,K07), v(VA08,FA05,K06),
v(VA04,VA08,K04), v(FA05,VA02,K05), begin(VA04)};
KETEN={v(VA09,VA04,K02), v(VA01,AA01,K08), begin(VA09),
eind(AA01), eind(VA04)}
```

Nu wordt er weer een nieuw eindpunt in behandeling genomen.

```
eindpunt=AA01;
soort knelpunt(K08)=actualiteit;
veroorzaker(AA01,K08)=VA01;

VA01 is onderdeel van NIEUWE_KETEN2;
VA01 ervaart binnen NIEUWE_KETEN2 knelpunt K09;
soort knelpunt(K09)=tijdigheid;
K08 IS_GEVOLG K09
```

Het knelpunt dat *AA01* ervaart blijkt het gevolg te zijn van het knelpunt dat *VA01* binnen NIEUWE_KETEN2 ervaart, dus *AA01* wordt opgenomen in NIEUWE_KETEN2.

```
NIEUWE_KETEN2={v(VA01,VA02,K01), v(VA05,VA01,K09),
v(VA09,VA05,K03), v(VA01,AA01,K08), begin(VA09)};
KETEN={v(VA09,VA04,K02), begin(VA09), eind(VA04)}
```

Nu wordt het laatste eindpunt in behandeling genomen.

```
eindpunt=VA04;
soort knelpunt(K02)=actualiteit;
veroorzaker(VA04,K02)=VA09;

VA09 is onderdeel van NIEUWE_KETEN2;
VA09 veroorzaakt binnen NIEUWE_KETEN2 knelpunt K03;
soort knelpunt(K03)=actualiteit;
K02 IS_GELIJK K03
```

Het knelpunt dat *VA04* ervaart door *VA09* blijkt gelijk te zijn aan het knelpunt dat *VA09* binnen NIEUWE_KETEN2 veroorzaakt, dus *VA04* wordt opgenomen in NIEUWE_KETEN2.

```
NIEUWE_KETEN2={v(VA01,VA02,K01), v(VA05,VA01,K09),
v(VA09,VA05,K03), v(VA01,AA01,K08), v(VA09,VA04,K02), begin(VA09)}
```

De oude keten bevat geen eindpunten meer. Er blijken twee nieuwe ketens te zijn ontstaan door het opsplitsen. Hierop kan het algoritme voor de prioriteitsstelling losgelaten worden. Zoals reeds vermeld, is deze weinig opwindend, omdat er maar twee knelpuntsketens te behandelen zijn.

De twee ketens kennen allebei slechts één beginpunt. Aangezien de beide ketens zich over meerdere HISsen uitstrekken en er geen OR-relaties zijn, wordt van de HIS waarvan een beginpunt onderdeel is de INTER_PRIORITEIT met waarde één verhoogd. In dit geval blijkt zowel beginpunt *VA04* als beginpunt *VA09* tot de HIS Orderbehandeling te behoren. Deze krijgt dan ook INTER_PRIORITEIT 2, terwijl de overige HISsen INTER_PRIORITEIT 0 behouden. In dit geval is het dus duidelijk dat de HIS orderbehandeling als eerste ontwikkeld dient te worden tot een informatiesysteem.

Conclusie

Het is mogelijk een objectieve uitspraak te doen over de volgorde van ontwikkeling van informatiesystemen. Door de knelpunten en de HISsen binnen een organisatie in ogenschouw te nemen, heb ik een methode gegeven waarmee prioriteiten toegekend kunnen worden. Hierbij wordt een grote mate van subjectiviteit ingedamd.

Ik heb getracht de methode voldoende abstract te houden, zodat het op veel algemene zaken van toepassing kan zijn. Het niet laten meetellen van de zwaarte en geaardheid van een knelpunt bij het toekennen van prioriteiten voorkomt het verzanden in details. Aanpassingen van de prioriteiten zoals die uit mijn methode zullen voortkomen, kunnen natuurlijk achteraf gedaan worden, doordat bijvoorbeeld het ene knelpunt dat een keten veroorzaakt veel moeilijker is op te lossen dan een ander knelpunt, dat ook een keten veroorzaakt. Door de mogelijkheid in te passen dat een bepaald knelpunt door meerdere activiteiten in een OR- en/of AND-relatie wordt veroorzaakt, blijft de methode breed toepasbaar.

Met het geven van de algoritmen uit §2.2 is de methode formeel opgeschreven. De moeilijkheid van een volledige implementatie van de methode zit in de IS_GEVOLG-functie. Een goede uitvoer van deze functie heeft de interventie van de menselijke logica nodig. In §2.3 heb ik echter een mogelijke manier van implementatie aangegeven.

Hoofdstuk 3
Workflow Management

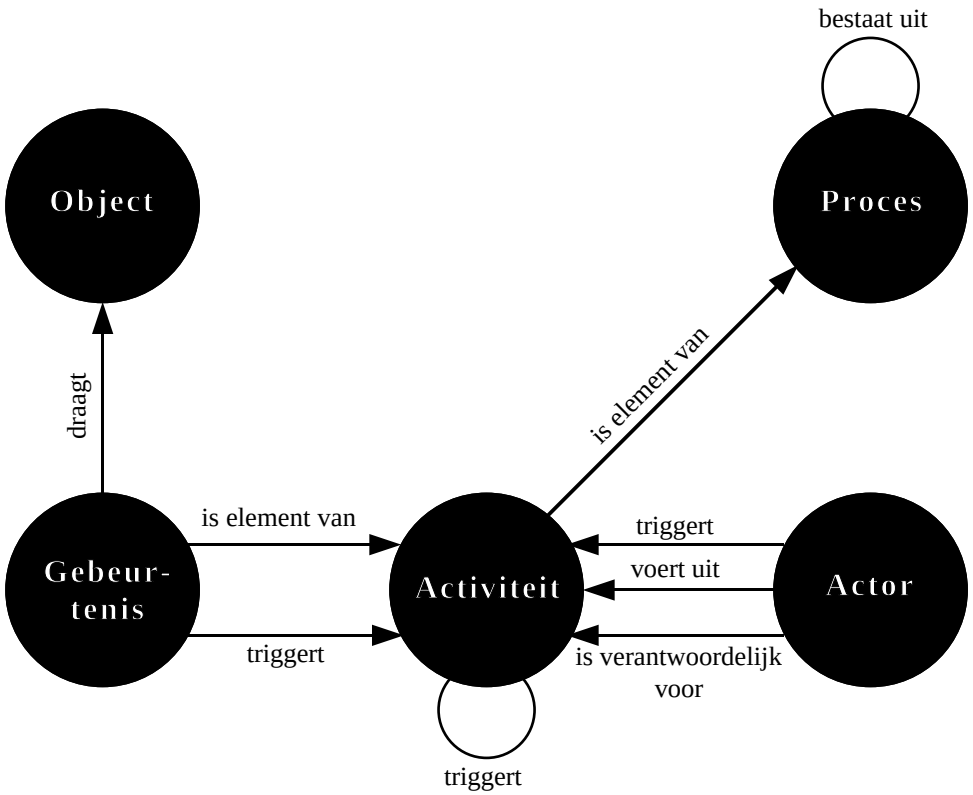
Inleiding

In dit hoofdstuk maak ik een vergelijking tussen mijn methode voor het toekennen van prioriteiten en het momenteel zeer in zwang zijnde workflow management. Zo op het eerste gezicht lijken beide methoden, laten we hier workflow management ook maar een methode noemen, op z'n minst raakvlakken te hebben. Workflow management (voortaan WfM) rept over workflows en ik heb het over hoofdinformatiestromen. Zijn deze hetzelfde? En waar past WfM in mijn verhaal? Of, evengoed, waar past mijn verhaal in WfM? Vragen waar ik in dit korte hoofdstuk antwoord op zal trachten te geven. Daartoe beschrijf ik eerst wat WfM is. Daarna zal ik beschrijven wat de raakvlakken zijn tussen mijn methode en WfM.

3.1 Workflow Management nader bekeken

Het is niet eenvoudig een eenduidige definitie te geven voor WfM. Net als bij object-oriëntatie zijn er voor WfM vele definities in omloop. Om enigszins eenduidigheid in de definities omtrent WfM te krijgen, heeft een internationale organisatie, The Workflow Management Coalition genaamd, een rapport uitgebracht waarin een groot aantal WfM-definities is gespecificeerd.[4] Helaas zijn de definities nogal vaag beschreven en wordt er zelfs geen definitie gegeven van workflow management op zich. Een goede definite van WfM heb ik gevonden in [BAU1994]. Dit artikel is geschreven met medewerking van de Universiteit van Twente, waar men veel onderzoek verricht naar WfM. De definitie die [BAU1994] hanteert voor WfM is: *WfM is het beheersen van de werkstroom van een administratief proces. Een systeem voor WfM is een combinatie van hard- en software die ondersteuning biedt bij het uitvoeren, het beheersen en het beheren van zo'n proces* ([BAU1994], p.29). Voor de uitvoerder van een taak die onderdeel is van een door WfM ondersteund proces is het WfM-systeem een hulpmiddel voor de uitvoering. Hij of zij weet precies wat er gedaan moet worden en kan snel bij de benodigde informatie. Managers hebben een beter inzicht in de voortgang van de lopende processen en kunnen makkelijker ingrijpen ([BAU1994],

4 The Workflow Management Coalition, *Glossary - A Workflow Management Coalition Specification*, Brussel, 1994.



Figuur 3.1 - Samenhang tussen workflow-concepten ([BAU1994], p.29)

p.29). Het belang van WfM wordt ook duidelijk door de uitspraak: ‘de kwaliteit van de dienstverlening van een organisatie is mede afhankelijk van de beheersing van de werkstroom’ ([BAU1994], p.29). Of, zoals blijkt uit een rapport over een onderzoek naar bevindingen in de praktijk met WfM[5] : ‘*Workflow Management is rapidly becoming important. Businesses all over the world are using automated support of workflow to turn their islands of automated administration into an efficient and effective instrument with important commercial consequences. Success stories have caused a general belief that workflow management has at least the potential of being a major step forward in the use of information technology’*. WfM blijkt dus belangrijk en succesvol te zijn. Maar feitelijk weten we nu nog niets. Vragen als: *Wat is een werkstroom?* en *Hoe beheerst een WfM-systeem zo’n werkstroom?* zijn nog steeds niet beantwoord. Laat ik daarom eerst enkele concepten van WfM beschrijven. Ik ben tot deze beschrijving gekomen dankzij [WMC1994], [BAU1994] en gesprekken met dr ir P.W.P.J. Grefen van de Universiteit van Twente.

5 Joosten, S., G. Aussems, M. Duitshof en R. Huffmeijer, *WA-12: An empirical study about the practice of Workflow Management*, Enschede, 1994.

Bij WfM worden de bestaande processen van een organisatie in een model vastgelegd. Sommige (delen van) processen kunnen geautomatiseerd worden m.b.v. een WfM-systeem en sommige niet. De eerste worden Workflow Processes genoemd en de laatste Manual Processes. Workflow Processes kunnen worden beheerst en uitgevoerd door een WfM-systeem. Om tot een beter inzicht in workflows te komen, worden er enige begrippen gehanteerd. Zie ter verduidelijking figuur 3.1: Een *gebeurtenis* is iets dat plaatsvindt of kan gebeuren; een *object* is iets dat bestaat (kan gezien of gevoeld worden); een *actor* is iets of iemand die handelt; een *activiteit* is een verzameling gebeurtenissen die plaatsvinden onder de verantwoordelijkheid van een actor; een *trigger*: een bepaalde gebeurtenis triggert een activiteit als het plaatsvinden van die gebeurtenis de uitvoer van die activiteit tot gevolg heeft; een *proces* is een verzameling van activiteiten met een gemeenschappelijke bedoeling; een *workflow* (ofwel werkstroom) is een systeem waarvan de elementen activiteiten zijn, die door een trigger-relatie met elkaar in verband staan en van buiten het systeem door een externe trigger in gang worden gezet. Deze definities komen uit [BAU1994], evenals figuur 3.1.

Een WfM-systeem weet precies welke processen er binnen de organisatie zijn en kan de geautomatiseerde processen beheersen en uitvoeren. Allerlei beslissingen die er genomen kunnen worden omtrent de processen zijn ook bekend bij het systeem. Zo'n systeem weet welke taken er zijn en welke werkactiviteiten (iets dat men kan doen) er uitgevoerd moeten of kunnen worden. Het systeem omvat de verschillende applicaties die deze taken kunnen uitvoeren. Als een gebruiker bijvoorbeeld een formulier in wil vullen, dan kan hij of zij werkactiviteit 'invullen formulier' selecteren van het systeem en de bijbehorende applicatie opstarten. WfM weet wanneer welke informatie waarnaartoe moet.

Bij WfM wordt er een belangrijk onderscheid gemaakt tussen informatie- en controlestromen. Een informatiestroom is bijvoorbeeld de reeks activiteiten die een order afloopt vanaf binnenkomst tot de complete verwerking ervan. Over zo'n stroom is allerlei informatie vast te leggen, zoals bijvoorbeeld de positie van de order op dit moment. Het WfM-systeem houdt dit allemaal bij. Een controlestroom loopt meestal analoog aan de informatiestroom en houdt onder andere bij welke handelingen er reeds gedaan zijn. Echter, het is mogelijk dat een controlestroom zich van de informatiestroom afsplitst. Stel bijvoorbeeld dat de verwerking van een order verloopt via activiteiten *a* en *b* en in de beslissingscriteria is vastgelegd dat wanneer de order activiteit *a* verlaat de manager in kwestie gewaarschuwd moet worden. In dit geval loopt de informatiestroom van activiteit *a* naar *b*, maar de controlestroom loopt van activiteit *a* naar de manager die gewaarschuwd moet worden en van daaruit naar activiteit *b*. Het feit dat activiteit *a* klaar is, is een trigger voor een controlestroom naar de manager.

WfM maakt de papierstroom overbodig, omdat alle informatie via het systeem loopt en daarmee volledig geautomatiseerd is. Dit bevordert natuurlijk de snelheid. Een WfM-systeem heeft altijd betrekking op de informatiestromen binnen een organisatie. Fysieke processen zijn niet van belang bij WfM, hoewel de eventuele beheersprocessen met informatie over de fysieke processen wel weer onderdeel kunnen zijn van een WfM-systeem. Er zijn tegenwoordig al vele pakketten voor de ondersteuning en beheersing van workflows beschikbaar. Met zo’n pakket kunnen de bedrijfsprocessen gedefinieerd en beheerst worden.

3.2 Een vergelijking

Na de korte beschrijving van WfM uit de vorige paragraaf, zal ik in deze paragraaf aangeven waar mijn methode in het verhaal over WfM past of andersom. Met de kennis over HISsen valt bij het lezen van de vorige paragraaf gelijk op dat een workflow eigenlijk hetzelfde is als een HIS. Beide zijn informatiestromen die door een extern element getriggerd worden, wat een reeks activiteiten tot gevolg heeft. Laten we dus zeggen dat een HIS hetzelfde is als een workflow, mits de gehele HIS automatiseerbaar is. Een workflow is immers onderdeel van een volledig geautomatiseerd WfM-systeem. Dat een HIS automatiseerbaar moet zijn, blijkt natuurlijk uit het feit dat een HIS een afbakening is van een mogelijk te ontwikkelen informatiesysteem. De onttrekking van workflows gebeurt op een zelfde manier als die van HISsen.

Aan HISsen kleven knelpunten. Daarom zullen workflows eveneens knelpunten (kunnen) bevatten. Ik heb de knelpunten gebruikt om een volgorde aan te geven voor de ontwikkeling van informatiesystemen, zoals die afgebakend zijn door HISsen. Een volgorde geven aan de ontwikkeling van workflows in een WfM-systeem heeft ook zin, omdat meestal niet alle workflows geïmplementeerd worden. Vaak worden de meest gestandaardiseerde processen of de processen die de meeste problemen veroorzaken (knelpunten!) geïmplementeerd. Mijn methode kan dus prioriteiten toekennen aan de workflows binnen een organisatie, zodat de meest problematische workflows voorrang krijgen.

Een nuttig aspect van WfM met betrekking tot de prioriteitsstelling is de grote controle die WfM heeft op de voortgang van een proces. Daarmee kunnen knelpunten opgelost worden. Stel bijvoorbeeld dat een activiteit *b* een bepaald document *d* (een electronisch document) voor een bepaald tijdstip dient te hebben, opdat er geen knelpunt ontstaat. Dit document loopt via activiteit *a* voordat het naar *b* gaat. Het WfM-systeem merkt dan bijvoorbeeld op dat *a* de belangrijke informatie voor *b*, namelijk document *d*, nog steeds niet verwerkt heeft. Activiteit *a* kan bijvoorbeeld te druk zijn met het verwerken van minder belangrijke informatie. Dan kan het systeem een waarschuwing geven aan *a* dat hij of zij document *d* moet gaan behandelen. Daarmee voorkomt het systeem een knelpunt. Het WfM-systeem kent de relaties tussen de verschillende werkplekken en kan prioriteiten toekennen aan verschillende informatie-dragers binnen bepaalde processen. Daarmee kunnen bepaalde soorten knelpunten in een vroeg stadium opgelost worden, alleen al door het gebruik van WfM. Sommige soorten kunnen misschien gedeeltelijk opgelost worden en andere helemaal niet. Voor de prioriteitsstelling heeft dit alleen gevolgen wanneer men van te voren weet dat er een WfM-systeem geïmplementeerd zal worden. Op grond van die voorkennis kunnen dan bepaalde knelpuntsketens buiten beschouwing gelaten worden, omdat zij sowieso verdwijnen bij het gebruik van een WfM-systeem. Welke soorten knelpunten invloed

kunnen ondervinden van WfM laat ik hier buiten beschouwing. Dat zou een diepgaande studie vereisen.

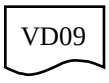
Eigenlijk is het laatste geval, de invloed van WfM op knelpunten, niet als dusdanig te onderkennen. Bij het toekennen van prioriteiten heb ik afstand genomen van de zwaarte en geaardheid van de knelpunten. Zou ik rekening gaan houden met het feit dat een WfM-pakket gebruikt gaat worden, dan ben ik van conceptueel niveau naar implementatie niveau gegaan. In de informatieplanningsfase, de fase waarin mijn methode toegepast wordt, mag nog geen rekening gehouden worden met de manier van implementatie.

Conclusie

Er zijn zeker raakvlakken tussen mijn methode en Workflow Management. Zo mogen HISsen en workflows als hetzelfde beschouwd worden. Het vergelijken tussen de twee aspecten wordt bemoeilijkt door het feit dat mijn methode zich op conceptueel niveau bevindt en WfM op implementatie niveau. Bovendien wordt de prioriteitsstelling toegepast in de informatieplanningsfase en WfM pas in een later stadium. Wel kan gezegd worden dat WfM bepaalde soorten knelpunten geheel of gedeeltelijk kan oplossen. Dit zou dan invloed hebben op de prioriteitsstelling. Eveneens kan mijn methode gebruikt worden om workflows te selecteren voor implementatie in een WfM-systeem.

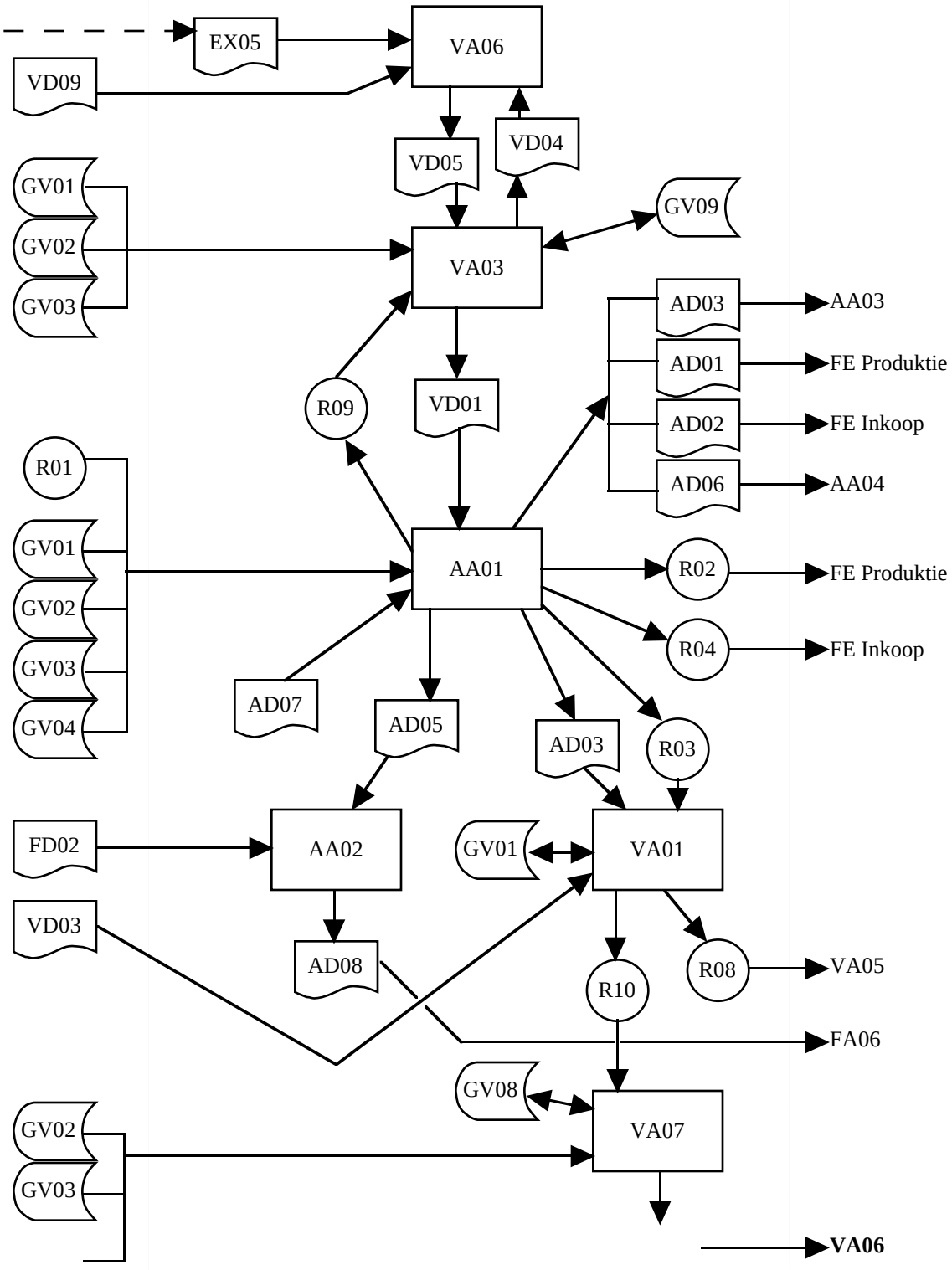
Bijlage II

In deze bijlage worden drie HISsen gepresenteerd. Zij zijn afgeleid van de functionele eenheden uit bijlage I. Zo heb ik bijvoorbeeld besloten om drie activiteiten met een tijd-stimulus uit de functionele eenheid Financiën & Administratie in één HIS te stoppen. Ik zal niet verder ingaan op de manier waarop ik tot deze HISsen ben gekomen. Zij dienen slechts ter illustratie. Voor een uitgebreide beschrijving over het bepalen van HISsen verwijs ik naar §1.4 en [HUI1989].



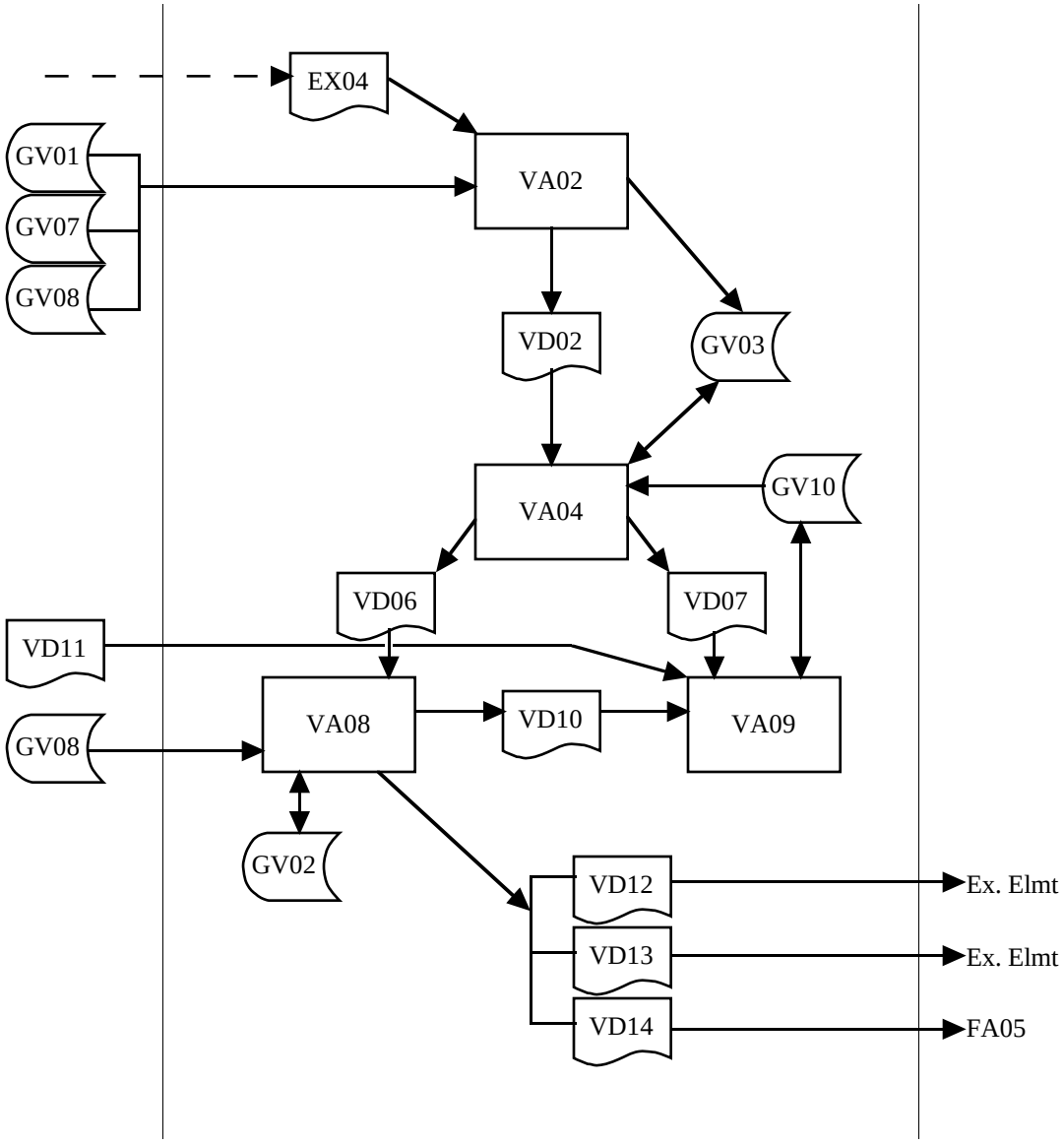
HIS Bedrijfsplanning

stimulus: EX05 - marktgegevens



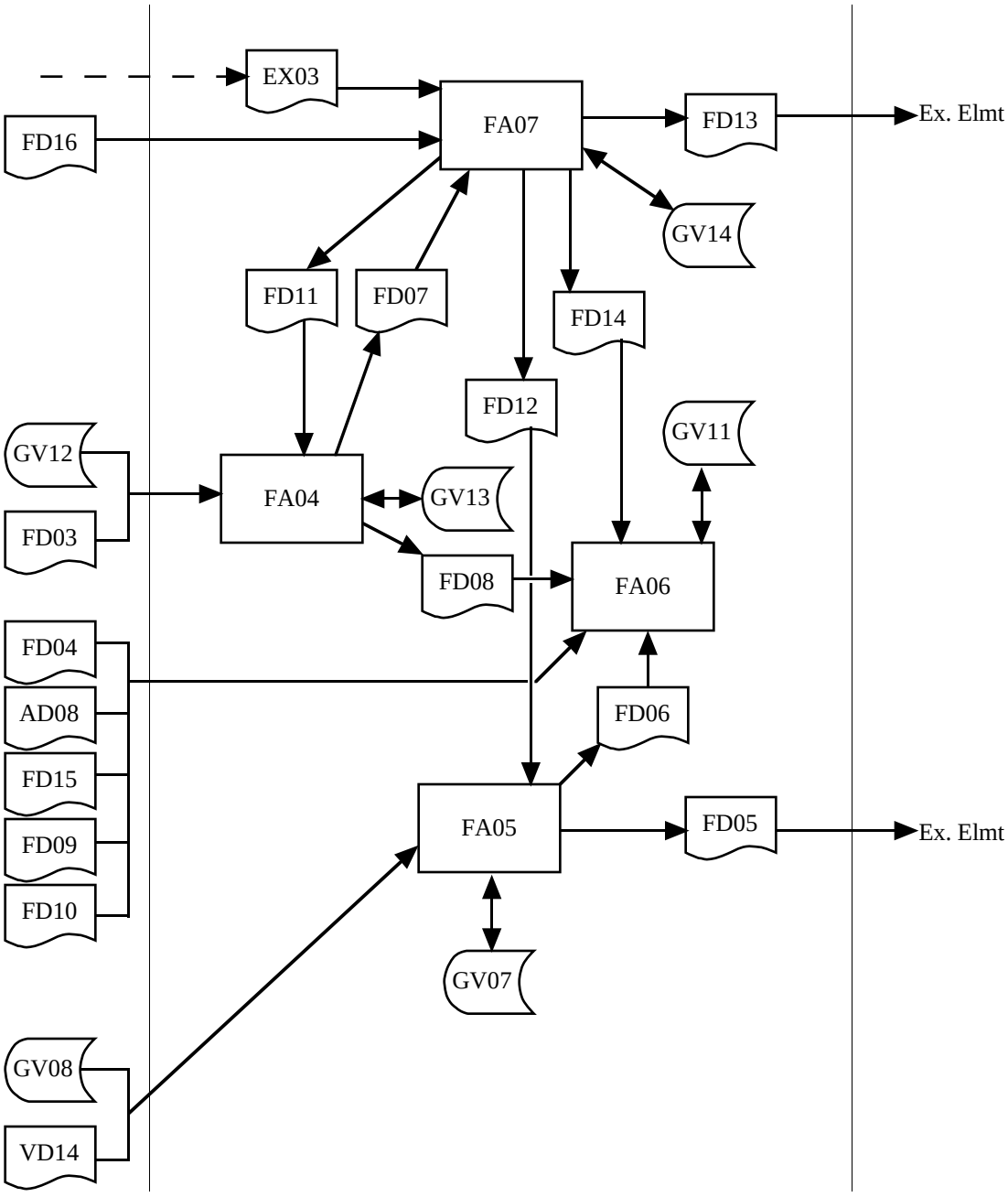
HIS Orderbehandeling

stimulus: EX04 - order



HIS Betalingen & Boekhouden

stimulus: EX03 - ontvangsten



Verwijzingen

[AAR1989] Aarts, A.F.H., P.M.C. Janssen, *Handboek informatieplanning: een instrument voor het beheersbaar maken van de informatievoorziening*, Den Haag, 1989.

[BAU1994] Baumann, M., 'Workflow-tools aan elkaar gewaagd', *Computable*, 21 oktober, 1994.

[BUS1987] Bushoff, R., J.A. Oosterhaven, 'Information Strategy Planning', *Informatie*, jrg. 29, n° 3, 1987.

[GRE1985] Greveling, N.J.W., *Een referentiemodel als hulpmiddel bij informatieplanning*, Arnhem, 1985.

[HUI1989] Huis in't Veld, H.A., *Inhoud en fasering van informatieplanning*, Arnhem, 1989.

[JMS1982] Martin, J., *Strategic data-planning methodologies*, New Jersey, 1982.

[MAC1986] Macdonald, I.G., *Information Engineering: An improved automatable methodology for the design of data sharing systems*, Noordwijkerhout, 1986.

[SPR1986] Sprague jr., R.H., B.C. McNurlin, *Information systems management in practice*, New Jersey, 1986.

[TUR1990] Turner, W.S., R.P. Langerhorst, G.F. Hice, H.B. Eilers, A.A. Uijttenbroek, *SDM - System Development Methodology*, Rijswijk, 1990.

[WMC1994] The Workflow Management Coalition, *Glossary - A Workflow Management Coalition Specification*, Brussel, 1994.

Bibliografie

- Aarts, A.F.H., P.M.C. Janssen, *Handboek informatieplanning: een instrument voor het beheersbaar maken van de informatievoorziening*, Den Haag, 1989.
- Baumann, M., ‘Workflow-tools aan elkaar gewaagd’, *Computable*, 21 oktober, 1994.
- Bemelmans, T.M.A., *Bestuurlijke informatiesystemen en automatisering*, Deventer/Leiden/Antwerpen, 1991.
- Boer, J.G. de, N.J.W. Greveling, ‘Informatieplanning met behulp van referentie-informatiemodellen 2 - Een methode voor informatieplanning’, *Informatie*, jrg. 28, n° 3, 1986.
- Boer, J.G. de, R. Vonk, ‘Help...de informatie-analist verzuipt’, *Informatie*, jrg. 26, n° 12, 1984.
- Bosch, J., ‘Op weg naar strategische workflowsystemen’, *Computable*, 17 juni, 1994.
- Boynton, A.C., G.C. Jacobs, R.W. Zmud, ‘Onder wiens verantwoordelijkheid valt het management van IT?’, *Management en Organisatie van Automatiseringsmiddelen*, n° 2, 1994.
- Bushoff, R., J.A. Oosterhaven, ‘Information Strategy Planning’, *Informatie*, jrg. 29, n° 3, 1987.
- Dissel, H.G. van, D. Park, ‘Informatiebeleids- en informatieplanningsmethoden - Nog een lange weg te gaan of op het verkeerde pad?’, *Informatie*, jrg. 31, n° 10, 1989.
- Doorewaard, H, H. Regtering, *Integraal automatiseren*, Deventer, 1990.
- Galliers, R.D., ‘IT-strategieën: meer dan alléén concurrentievoordeel’, *Management en Organisatie van Automatiseringsmiddelen*, n° 5, 1994.
- Geurts, M., I. van den Kamp, *Methoden voor informatieplanning - Een vergelijking, synthese en de ondersteunende gereedschappen*, Moret Advies, Utrecht, 1989.
- Greveling, N.J.W., *Een referentiemodel als hulpmiddel bij informatieplanning*, Arnhem, 1985.
- Hein, K.P., ‘Information System Model and Architecture Generator’, *IBM Systems Journal*, vol. 24, n° 3/4, 1985.
- Henderson, J.C., N. Venkatraman, ‘Strategic Alignment: het gebruik van informatietechnologie voor veranderingsprocessen in organisaties’, *Management en Organisatie van Automatiseringsmiddelen*, n° 4, 1994.
- Huis in’t Veld, H.A., *Inhoud en fasering van informatieplanning*, Arnhem, 1989.
- Huis in’t Veld, H.A., *Informatieverzorging in organisaties*, KUN, Nijmegen, 1991.
- In’t Veld, J., *Analyse van organisatieproblemen - Een toepassing van denken in systemen en processen*, Leiden/Antwerpen, 1988.
- Joosten, S., G. Aussems, ‘Met de stroom mee?’, *Computable*, 16 december, 1994.
- Joosten, S., G. Aussems, M. Duitshof, R. Huffmeijer, *WA-12: An empirical study about the*

practice of Workflow Management, Enschede, 1994.
- Kim, Y., G.C. Everest, 'De bouw van een IS-architectuur; verzamelde ervaringen uit de praktijk', *Management en Organisatie van Automatiseringsmiddelen*, n° 4, 1994.
- Lederer, A.L., A.L. Mendelow, 'Planning van informatiesystemen en de uitdaging van veranderende prioriteiten', *Management en Organisatie van Automatiseringsmiddelen*, n° 3, 1994.
- Macdonald, I.G., *Information Engineering: An improved automatable methodology for the design of data sharing systems*, Noordwijkerhout, 1986.
- Martin, J., *Strategic data-planning methodologies*, New Jersey, 1982.
- Meel, J.M. van, M.C. de Heer, 'Workflow-management eist meer dan pakket', *Computable*, 2 december, 1994.
- Pruijm, R., 'Ook IT-organisatie is toe aan herontwerp', *Computable*, 14 januari, 1994.
- Riksen, D., P. Blokdijk, 'TSS: Toepassing Systeem Studie - Het specificeren van systeem-eisen door de gebruiker in zes weken', *Informatie*, jrg. 28, n° 3, 1986.
- Sprague jr., R.H., B.C. McNurlin, *Information systems management in practice*, New Jersey, 1986.
- Starreveld, R.W., H.B. de Mare, E.J. Joël, *Bestuurlijke informatieverzorging - Algemene grondslagen deel 1*, Alphen aan den Rijn/Zaventem, 1994.
- Stol, H.R., *Informatieplanning in de praktijk - Grondbeginselen voor de opzet van bestuurlijke informatiesystemen*, Alphen aan den Rijn/Deurne, 1990.
- The Workflow Management Coalition, *Glossary - A Workflow Management Coalition Specification*, Brussel, 1994.
- Theeuwes, J.A.M., *Informatieplanning*, Deventer, 1987.
- Turner, W.S., R.P. Langerhorst, G.F. Hice, H.B. Eilers, A.A. Uijttenbroek, *SDM - System Development Methodology*, Rijswijk, 1990.
- Uden, M. van, 'Werkstroombeheer is haalbare optie', *Computable*, 15 juli, 1994.
- Vanhomwegen, G., 'Drempelvrees voor imaging en workflow overwonnen', *Computable*, 15 juli, 1994.
- Wiseman, D., 'Information Economics: een praktische methode voor de waardering van informatiesystemen', *Management en Organisatie van Automatiseringsmiddelen*, n° 2, 1994.