RENÉ SIEDERS

Waarom de modellen zijn zoals ze zijn

Misvattingen en misverstanden over ASL en BiSL (deel 2)

René Sieders, gespecialiseerd in het inrichten en verbeteren van applicatiebeheer en functioneel beheer, was nauw betrokken bij het ontstaan van ASL en BiSL. Werkend in de praktijk met beide modellen kwam hij echter verschillende misvattingen en misverstanden tegen. In dit nummer deel 2 van het tweeluik over de ware aard van ASL en BiSL.

Zoals in vorig nummer van IT Service Magazine al aangegeven, bestaan er rond ASL en BiSL een flink aantal misverstanden en misvattingen – zowel onder vriend als onder vijand van beide modellen. In dit tweeluik worden een aantal van die misvattingen en misverstanden nader belicht, met als doel meer inzicht te geven in de algemene opbouw van beide modellen.

Misvatting De grenzen van het táakgebied functioneel beheer dienen gelijk te zijn aan de grenzen van een áfdeling functioneel beheer.

Functioneel beheer is de benaming voor het taakgebied van een der drie domeinen zoals Looijen ze al meer dan tien jaar geleden heeft bepaald. In BiSL is het gehele taakgebied beschreven, van uitvoerend tot en met richtinggevend. Deze beschrijving staat los van een eventuele implementatie. In het BiSL-boek kun je niet lezen hoe je functioneel beheer moet inrichten voor een bank, noch voor een tapijtenfabriek, noch voor enige andere organisatie. Voor een toepassing van de theorie van BiSL zijn wel handvatten te geven, zoals Van der Pols doet in zijn boek dat één dezer dagen verschijnt. Daarnaast kun je gebruikmaken van best practices van binnen of buiten je eigen organisatie, maar dan houdt het wel op. En dat is logisch, want elke organisatie is anders, heeft andere eisen en stelt andere prioriteiten. In praktijk zie je vaak dat voor het richtinggevende niveau van de processen de verantwoordelijkheid ligt bij een afdeling informatiemanagement[1] en voor het uitvoerende niveau bij een afdeling functioneel beheer[2]. Dat wil echter niet zeggen dat daarmee precies die processen uit het BiSL-model ook daar belegd zijn of zouden moeten zijn. Vaak is het meer en vaak is het minder. Regelmatig zie je bijvoorbeeld dat een aantal van de activiteiten op het gebied van 'opstellen informatiestrategie' belegd zijn bij een aparte architectuurclub, en dat gebruikersondersteuning belegd is bij key-users in de gebruikersorganisatie of, deels, bij een helpdesk. Al die mensen voeren dan taken uit op het gebied van functioneel beheer. Is dat erg? Nee, ik neem aan dat daar goede redenen voor zijn en dat dit op dat moment de juiste belegging van taken binnen de organisatie is. Als het goed is worden er gebruikers ingezet bij een gebruikers-acceptatietest.
Vaak zie je ook dat de afdeling functioneel beheer het functioneel ontwerp onderhoudt. Op dat moment voert ze een taak uit op het gebied van applicatiebeheer. Is dat erg? Dat hoeft niet per se, hoewel ik dan wel van mening ben dat je op moet letten dat je de taken van opdrachtgever en opdrachtnemer goed blijft scheiden (doet de functioneel beheerder dan ook de functionele systeem- of integratietest, en accepteert hij het technisch ontwerp?).
Samengevat: BiSL schrijft niet voor hoe je je processen moet uitvoeren of waar je de taken moet beleggen; BiSL reikt slechts een overzicht aan van het gehele domein en geeft daarmee inzicht in welke dingen je zou moeten of kunnen regelen. Overigens geldt hetzelfde voor ASL en misschien nog wel in sterkere mate. In veel gevallen zijn bijvoorbeeld de beheertaken op een andere plaats belegd dan de onderhoudstaken. Tenslotte wijs ik nog op de taakverdeling bij softwarepakketten, of bij ASP (Application Service Providing) en SaaS (Software as a Service). Daar ligt een deel van de taken bij de leveranciers en een deel van de taken bij de afnemers. Zie verder over dit onderwerp de artikelen van Van der Pols en Van Outvorst in het literatuuroverzicht.

Misverstand In ASL en BiSL ontbreekt het proces probleembeheer (problem management).

In ASL en BiSL is er bewust voor gekozen om probleembeheer niet als apart proces te definiëren, maar als deelproces van kwaliteitsbeheer, respectievelijk behoeftemanagement. Ik noem de belangrijkste reden eerst. Stel er worden veel fouten gevonden in de gebruikers-acceptatietest. Wat is er dan aan de hand? Was de functionele test wel grondig genoeg? Hebben de applicatiebeheerders in de unittest en in de technische test hun werk niet goed gedaan? Of heeft de gebruiker toch andere verwachtingen en waren de specificaties en het daarop gebaseerde functioneel ontwerp toch niet in orde? Structure-

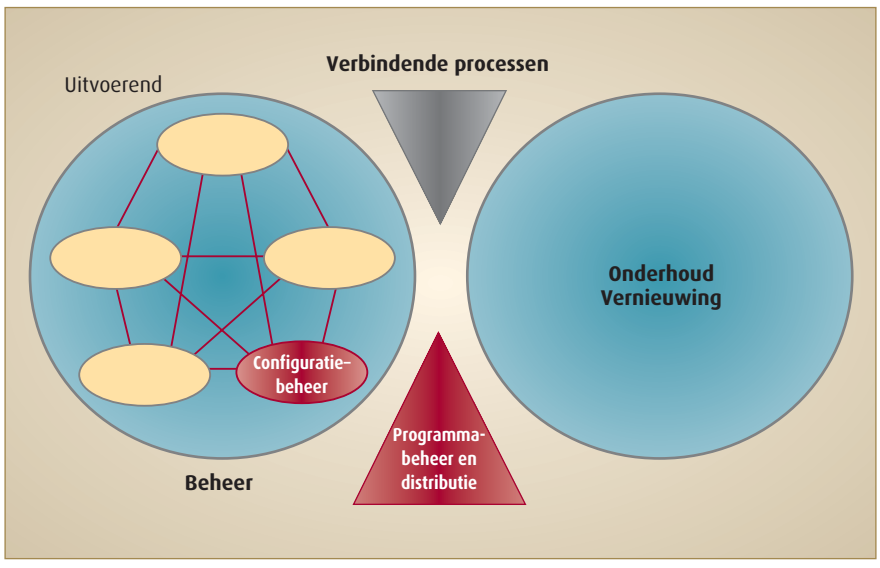
[1] Vandaar dat de auteurs deze term hebben toegevoegd aan de titel van het boek.
[2] Enkele andere namen die ik gehoord heb zijn: gebruikersdesk, functioneel applicatiebeheer, applicatiebeheer, gebruikersondersteuning, informatiebeheer, accountbeheer.

le maatregelen binnen applicatiebeheer kunnen zijn: testopleiding, ontwerpopleiding, cursus 'klantgericht werken', enzovoort. Binnen functioneel beheer kan men denken aan het aanwijzen van betere gebruikersvertegenwoordigers (voor specificeren of testen). Ik geef een ander voorbeeld. Stel dat we bij onderhoudstrajecten steeds uit de planning lopen. Wat is er aan de hand? Probeert de opdrachtgever de eisen en specificaties steeds te veranderen? Kan de applicatiebeheerder niet plannen? Zijn we niet goed in impactanalyse? Zijn de bouwers niet ervaren genoeg? Veelal wordt een probleem gedefinieerd als een incident met een structurele oorzaak. Met voorgaande voorbeelden toon ik aan dat structurele verbeteractiviteiten die dienen om de dienstverlening te verbeteren en verstoringen in de dienstverlening te voorkomen niet alleen voortkomen uit incident management (incidenten 'met een staartje') maar uit alle processen. Verbeterloops, verbeteractiviteiten zijn een primair onderwerp van de processen kwaliteitsmanagement en behoeftemanagement. Vandaar dat probleembeheer daar is ondergebracht als subproces.
Een andere, minder belangrijke reden is dat in de domeinen applicatiebeheer en functioneel beheer nu eenmaal minder incidenten voorkomen dan in het domein technisch beheer. Binnen technisch beheer kan bijvoorbeeld iedereen de helpdesk bellen met zaken als wachtwoordproblemen, etc. Vanuit de redenering 'minder incidenten dus minder problems', kom je dan vanzelf tot de conclusie dat het *over done* is om er een apart proces voor te definiëren.
En wat doen we dan binnen ASL en BiSL met de incidenten die zo lastig zijn dat we ze niet direct kunnen oplossen? Dat laten we gewoon incidenten zijn zolang de gebruiker niet (goed) kan werken. Als het brandje is geblust en de gebruiker weer kan werken, en er moet (ooit) nog een nette structurele oplossing aangebracht worden (zonder urgent karakter), dan definiëren we een 'probleem' dat we via het proces 'problem management' afhandelen.
Tenslotte ben ik nog wel van mening dat probleembeheer binnen ITIL in de praktijk vaak nogal gericht is op het oplossen van opgetreden problemen (reactief), terwijl het voor mijn gevoel bij ASL en BiSL meer gaat om het voorkomen van problemen en incidenten (proactief). Vandaar ook dat het in die twee modellen binnen kwaliteitsmanagement c.q. behoeftemanagement is opgenomen.

In ASL en BiSL is er bewust voor gekozen om probleembeheer niet als apart proces te definiëren

Misverstand Functioneel beheerders dienen het functioneel ontwerp te maken en te onderhouden[3].

In BiSL treft men het proces 'specificeren' aan en in ASL het proces 'ontwerp'. In praktijk blijkt dat over het onderscheid tussen deze twee veel verwarring bestaat. Vaak zie je dat het maken van een functioneel ontwerp (FO) als een taak belegd is bij functioneel beheer.
Hoewel het ASL-boek vrij helder is over het feit dat, volgens ASL, het proces 'ontwerp' bestaat uit het opstellen en onderhouden van het FO, terwijl het opstellen van het technisch ontwerp (TO) binnen 'realisatie' valt, hanteert de schrijver van ASL ook de term 'specificaties' als over FO wordt gesproken. Technisch gezien klopt dat. De specificaties uit functioneel beheer gaan over de eisen van de business over de complete informatievoorziening, zowel geautomatiseerd als niet-geautomatiseerd. Deze eisen worden gedefinieerd na analyse van de bedrijfsactiviteiten en de hierbij benodigde informatievoorziening. Deze analyse wordt ook wel requirementsanalyse of informatieanalyse genoemd. De specificaties van de functioneel ontwerper gaan over de eisen ten aanzien van de logische invulling van deze eisen, dus het specificeren van de werking van het systeem om invulling te geven aan deze eisen. Maar verhelderend is het niet! Ik heb van de auteur van ASL begrepen dat dit wordt rechtgezet in de volgende versie van ASL.
Functioneel beheer gaat over 'het wat' en applicatiebeheer gaat over 'het hoe' (FO) en 'het waarmee' (TO). Dit is voor de Nederlandse markt soms verwarrend omdat over het FO altijd gesproken wordt als 'het wat'. Methodisch en qua architectuur is dat echter onjuist. Methodisch (Vandenbulcke, Bemelmans e.a.) hebben we het doorgaans over het onderscheid tussen informatieanalyse, functioneel ontwerp, technisch ontwerp en bouwen. In de IT-architectuur (Dya, Zachman e.a.) praten men over contextueel niveau (bedrijf), conceptueel niveau (bedrijfsactiviteiten en informatie – het wat – te definiëren door de eigenaar of analist), logisch niveau (het ontwerp van de functies van het systeem – het hoe –, op



Een uitsnede uit het ASL-model. De processen Configuratiebeheer en 'Programmabeheer en distributie'.

[3] In onderstaande tekst heb ik dankbaar gebruikgemaakt van een notitie over dit onderwerp van mijn vriend en collega Dolf Hoogland

te stellen door de ontwerper) en technisch niveau (het ontwerp van 'het waarmee'). Waar applicatiebeheer verantwoordelijk is voor het maken en onderhouden van het functioneel ontwerp en het functioneel of logisch datamodel, is functioneel beheer verantwoordelijk voor zaken als het functiemodel (welke functies zijn nodig voor de bedrijfsactiviteiten), het semantische gegevensmodel (welke informatie speelt hierbij een rol), en de gebeurtenissenlijst (wat zijn de triggers in de werkelijkheid).

Voorbeeld

Stel er is een vereniging met leden die contributie moeten betalen. Het administratieve proces van de vereniging bevat dan twee hoofdactiviteiten: het registreren van de leden en het innen van de contributie. De eisen vanuit functioneel beheer zijn dan:

- 1. Registratie van de leden: het moet mogelijk zijn leden te administreren.
- 2. Het afhandelen van de contributie: aan de hand van de leeftijd wordt de jaarlijkse contributie bepaald. De leden kunnen maandelijks de contributie betalen middels een acceptgiro. Maandelijks moet er inzicht bestaan in de openstaande contributie.

Het FO zal dan bestaan uit een reeks functies. Ten eerste een functie voor het registreren van nieuwe leden, het muteren van registraties en het verwijderen van de leden. In het FO staat dan iets als 'bepaal aan de hand van de naam of deze al voorkomt in de administratie'. Zo ja, dan moeten de gegevens van het lid opgehaald worden en op het mutatiescherm getoond worden. Indien deze niet voorkomt moet het invoerscherm 'nieuwe leden' getoond worden, etc. Verder: een functie voor het opnemen van de hoogte van de contributie afhankelijk van bijvoorbeeld de leeftijd; een functie voor het maandelijks opsturen van de acceptgiro; een functie voor het inboeken van de betaalde contributie en het bepalen van de openstaande contributie; etc., etc..
Het TO bevat dan de volgende zaken:

- toon het initiële invulscherm;
- indien het lidnr. in de tabel Leden niet voorkomt toon dan het invoerscherm;
- indien het lidnr. in de tabel Leden wel voorkomt, haal dan uit tabel Adres de adresgegevens op en plaats deze op het adresgedeelte van het mutatiescherm; haal de persoonsgegevens op uit tabel Leden en toon dit in het blok Persoonsgegevens;
- indien het betaalde bedrag uit tabel Contributie kleiner is dan het te betalen bedrag, geef dan de boodschap 'lid heeft betalingsachterstand'.

Misverstand In ASL is het proces configuratiebeheer dubbel opgenomen en in BiSL is men het vergeten.

Voor het eerste deel van dit misverstand moeten we eerst inzoomen op het ASL-model. Als we naar de operationele laag kijken, dan zien we dat het model is opgebouwd uit drie clusters van processen: links de beheerprocessen, rechts de onderhoudsprocessen en daartussenin de verbindende processen. En inderdaad, je komt twee processen tegen die zich bezighouden met het onderwerp configuratiebeheer: 'configuratiebeheer' binnen de beheerprocessen en 'programmabeheer en distributie' binnen de verbindende processen (zie afbeelding).

Configuratiebeheer

Men moet zich realiseren dat de beheerprocessen slechts de productiesituatie betreffen en niet de onderhoudssituatie. Configuratiebeheer gaat dus alleen over die configuratie-items die in productie zijn. De software-items en dergelijke horen hier dus niet bij[4]. Of, in technische termen: de (verschillende versies van de) executables spelen hier een rol en niet de sources die ertoe hebben geleid. Nog anders gezegd: configuratiebeheer gaat over welke versie van de programmatuur in welke productieomgeving draait.
Je zou je binnen configuratiebeheer zelfs kunnen beperken tot releasenummers. Wat er precies in die release zit, is op dit punt niet interessant. Dat is slechts belangrijk als je het over onderhoud hebt.
Onderdeel van configuratiebeheer is trouwens wel het beheren van servicelevel-items (contracten en SLA's) betreffende de productiesituatie. Die heb je voor je beheer namelijk ook weer nodig. Als je werkzaamheden uitvoert in het kader van beschikbaarheid, continuïteit, capaciteit of incidenten, is het belangrijk dat je de vigerende serviceafspraken weet te vinden.

Programmabeheer en distributie

Het voorgaande betekent dat je elders wel moet registreren welke software-items in welke release zijn opgenomen. En dat gebeurt in het proces 'programmabeheer en distributie' (P&D). Zoals gezegd: de beheerprocessen betreffen slechts de productiesituatie. Daarnaast betreffen de onderhoudsprocessen slechts de onderhoudssituatie, en de verbindende processen de schakels tussen die twee. Waar gaat P&D dan precies over? P&D is gericht op het beheren en distribueren van software-items. Dit houdt de volgende vier zaken in:

- 1. Het opslaan van software-items.
- 2. Het registreren van informatie over software-items: welke versies bevinden zich waar.
- 3. Het overzetten (vrijgeven) van software-items van de ene naar de andere omgeving. Dat wil dus zeggen: binnen de gehele OTAP-straat, vanaf het vrijgeven voor onderhoud via de verschillende ontwikkelomgevingen (O), testomgevingen (T), naar de acceptatietestomgevingen (A) en tenslotte naar de productieomgevingen (P).
- 4. Het verstrekken van informatie over voorgaande twee punten, bijvoorbeeld aan het proces 'impactanalyse'.

Punt 1 en 3 betreffen dus inderdaad activiteiten op het gebied van configuratiebeheer, maar dan gericht op de onderhoudssituatie, terwijl het proces configuratiebeheer gaat over de productiesituatie.
Overigens wordt soms ook aan mij gevraagd: maar waarom houdt applicatiebeheer informatie bij over executables in de productieomgeving? Dat is toch een taak van technisch beheer?". Ja, dat is inderdaad een taak van technisch beheer, maar ook van applicatiebeheer. Applicatiebeheer heeft die informatie namelijk nodig om de andere beheerprocessen goed uit te kunnen voeren. Als je bijvoorbeeld een incidentmelding krijgt van-

De procesindeling in BiSL is ontstaan door het bundelen van activiteiten rond de datastores

[4] Het boek van Van der Pols is daar niet echt helder in. Enerzijds is de opbouw helder beschreven, anderzijds lees je in de beschrijving van het proces configuratiebeheer dat het wel over software-items gaat, waaronder ook testbestanden. De auteur heeft aangegeven, dat dit in de volgende versie van het ASL-boek gecorrigeerd zal worden.

uit een gebruikersorganisatie, dan is het wel belangrijk om te weten welke versie van de programmatuur (welke release) daar op dat moment draait. Daaruit kun je trouwens ook concluderen dat het proces minder relevant is voor applicatiebeheerorganisaties die slechts één productieomgeving hebben.

Configuratiebeheer in BiSL

In BiSL komt men het proces configuratiebeheer niet tegen. Daar is een simpele reden voor: configuratiebeheer behoort niet tot de primaire bezigheid van functioneel beheer. Bij applicatiebeheer (en ook technisch beheer) betreft het beheren van je configuratie wel een primair proces. Applicatiebeheer gaat over het beheren en onderhouden van configuratie-items: applicaties, documentatie, et cetera. Functioneel beheer gaat over het beheren en onderhouden van de informatievoorziening en dat komt in grote lijnen neer op twee taken: het ondersteunen van gebruikers en, namens de gebruikersorganisatie, het opdrachtgeverschap richting ICT. Dat je daarbij ook objecten beheert, waarbij het belangrijk is om voor die objecten configuratiebeheer te doen, is evident. Ik noem bijvoorbeeld contracten, gebruikershandleidingen en werkinstructies. Maar het beheren van die objecten behoort niet tot de primaire activiteiten van functioneel beheer. Op het punt van het beheren van dat soort objecten zijn er twee opties: of je doet het binnen de betreffende processen waar die objecten beheerd worden, zoals binnen het proces 'onderhouden niet-geautomatiseerde IV', of je laat de objecten beheren door de applicatiebeheer- of technisch-beheerorganisatie. Zij zijn daar heel goed in!

Misverstand De auteurs van BiSL zijn autorisatiebeheer als proces vergeten.

In het BiSL-boek moet je met een lichtje zoeken naar het proces 'autorisatiebeheer'. In het hoofdstuk 'gebruikersondersteuning' staat wel een korte tekst: autorisatieaanvragen worden via gebruikersondersteuning in behandeling genomen en in het hoofdstuk over transitie staat iets over het verstrekken en/of intrekken van autorisaties in verband met de nieuwe of vernieuwde functionaliteit, maar verder komt men weinig tegen. Is dat terecht? Hoe komt het? Is men autorisatiebeheer soms vergeten?
Ik denk dat het niet terecht is dat autorisatiebeheer amper genoemd wordt. Voor veel functioneel beheerders is het verstrekken en intrekken een van de dagelijkse taken. Grofweg zou je kunnen zeggen dat de volgende zeven taken op het gebied van autorisatie behoren tot het taakgebied van een functioneel beheerder:

- 1. Verstrekken, aanpassen en intrekken van autorisaties naar aanleiding van verzoeken/opdrachten vanuit de gebruikersorganisatie.
- 2. Verstrekken, aanpassen en intrekken van autorisaties naar aanleiding van aanpassingen in de IV.
- 3. Vastleggen van en rapporteren over autorisaties.
- 4. Vertalen businessrollen naar autorisatieprofielen.
- 5. Specificeren autorisatie-eisen met betrekking tot geautomatiseerde en niet-geautomatiseerde IV.
- 6. Verstrekken opdrachten op het gebied van autorisaties aan ICT.
- 7. Vastleggen autorisatieniveaus ten aanzien van bedrijfsgegevens.

Configuratiebeheer gaat over welke versie van de programmatuur in welke productieomgeving draait

Als je naar het bovenstaande rijtje kijkt, kom je al snel tot de conclusie dat deze activiteiten onderdeel zijn van processen die wel in het BiSL-model staan: gebruikersondersteuning (1, 3, 4), beheer bedrijfsgegevens (3, 7), operationele ICT-aansturing (6), specificeren (5) en transitie (2). Welnu, dat is precies de reden waarom autorisatiebeheer niet als apart proces wordt onderkend in BiSL: autorisatiebeheer is verspreid over verschillende BiSL-processen, waarbij overigens de kern van de verantwoordelijkheid in het proces gebruikersondersteuning ligt.
De procesindeling in BiSL is ontstaan door het bundelen van activiteiten rond de datastores. Je ziet dat terug in de plaatjes in het boek. Veel mensen bekijken deze plaatjes niet, omdat ze er voor een tekstboek als BiSL niet toegankelijk uitzien. Dat is jammer, want deze plaatjes vormen eigenlijk het hart van het boek. Deze datastores komen allen slechts één maal voor en zo is de procesindeling ontstaan. Door autorisatiebeheer niet als een zelfstandig proces te beschrijven is deze structuur in tact gebleven. Dat neemt echter niet weg dat ik wel van mening blijf dat de auteurs iets meer aandacht hadden moeten besteden aan het onderwerp autorisatiebeheer. Voor een geïnteresseerde lezer valt het niet mee om uit te pluizen waar autorisatiebeheer precies zit en blijft het gokken hoe BiSL het ziet. Overigens geldt dat ook voor onderwerpen als licentiebeheer, securitymanagement en wellicht nog andere.

Literatuurlijst:

- **Backer, Y, J.J. Sybenga en R. van der Pols**, 2004, *Professioneel applicatiebeheer bij pakket- en ASP-providers*, bijdrage aan IT Servicemanagement Best Practices deel 1, ITSMF/Van Haren Publishing
- **Bakker, P.P. en M.E.E. Meijer -Veldman**, 2000, *R2C, LCE, ASL versus ITIL*, Roccade Public
- **Deurloo, C.D., R. van der Pols, M.E.E. Meijer-Veldman**, 1998, *Model voor functioneel beheer*, bijdrage aan IT-beheer Jaarboek 1998, Ten Hagen en Stam
- **Donatz, R en F. van Outvorst**, 2003, *Functioneel Beheer van pakketten*, bijdrage aan IT Beheer Jaarboek 2003, Ten Hagen en Stam
- **Hoogland D.**, 1999, *Specificaties en Ontwerp*, interne notitie PinkRoccade Atribit
- **Looijen M.**, 1997, *Het beheer van informatiesystemen*, Kluwer, Deventer
- **Meijer, Machteld en Jolanda Meijers**, 2002, *Effectief IT-beheer: samenwerken waar nodig en zelfstandig opereren waar mogelijk*, bijdrage aan IT Service-management Best Practices 2002, Ten Hagen & Stam
- **Outvorst, Frank en Ralph Donatz**, 2004, *Functioneel beheer bij pakketten (deel 2)*, Bijdrage aan IT Service-management best practices deel 1, ITSMF/Van Haren Publishing
- **Pols, R. van der**, 2001, *ASL een framework voor applicatiebeheer*, Ten Hagen Stam, ISBN 90 440 0266X
- **Pols, R. van der, F. van Outvorst en R. Donatz**, 2005, *BiSL: een framework voor functioneel beheer en informatiemanagement*, Van Haren Publishing, ISBN 90-77212-40-X
- **Roccade**, 1997, *Beheer en vernieuwen met R2C*, ISBN-90-803102-4-7

RENÉ SIEDERS
is Principal Consultant bij Getronics PinkRoccade en nauw betrokken bij het ontstaan van ASL en BiSL.