ET 11

05348

1986

SERIE RESEARCH MEMORANDA

DE FUNDAMENTEN VAN LINC

E.R.K. Spoor

Researchmemorandum 1986-11 februari 1986

VRIJE UNIVERSITEIT
FACULTEIT DER ECONOMISCHE WETENSCHAPPEN
A M S T E R D A M

DE FUNDAMENTEN VAN LINC

door E.R.K. Spoor

Produktiviteitsverbetering in de ontwikkeling van informatiesystemen kan ten dele worden bewerkstelligd door het hanteren van een programmeertaal die nauw verwant is met de taal waarin het conceptueel model (het produkt van de systeemontwerper) wordt beschreven. LINC, een produkt van Burroughs Corporation, pretendeert zo'n taal te zijn. De vakgroep Bestuurlijke Informatiekunde van de Economische Faculteit der Vrije Universiteit heeft, in samenwerking met Burroughs B.V., LINC toegepast bij de ontwikkeling van een verzekerings-informatiesysteem. Daarbij zijn de fundamenten van LINC, te weten COMPONENTS en EVENTS, nader onderzocht. In dit artikel worden deze bouwstenen vergeleken met bekende conceptuele primitieven verenigd in een recent ontwikkelde visie op conceptuele modellen.

1. INLEIDING

Evenals zijn medemens wordt de informatiekundige geconfronteerd met fysische verschijnselen, die zich in eerste instantie aan hem voordoen als een chaos van objecten (ruimte) welke aan verandering onderhevig zijn (tijd). Hij tracht in deze perceptie orde te scheppen door zich te bedienen van zijn vermogen tot herkenning en classificatie van de objecten en het constateren van verbanden tussen deze klassen. Zo schept hij zich structuren. Het is als het nemen van een foto waarin patronen worden herkend (bijvoorbeeld personen en relaties).
De werkelijkheid openbaart zich echter meer als een reeks foto's: een film. Zij verandert in de tijd. Het beeld van de werkelijkheid is dientengevolge ook aan verandering onderhevig. Het vertoont een gedrag. Onze conceptie van de werkelijkheid is dus een combinatie van structuur en gedrag. Formuleren we deze conceptie dan ontstaat een conceptueel model.
Ook de informatiekundige creëert zich een conceptueel model als hij (een deel van) een organisatie beschrijft om een informatiesysteem te realiseren. Hij beschrijft dit model in een taal of met behulp van schematechnieken (schema's zijn te vertalen).
Zo'n conceptuele taal is echter nog geen programmeertaal. Nog niet. Ze naderen elkaar. Enerzijds richten onderzoekingen zich op modeldefinities waarin structuur en gedrag beiden zijn vertegenwoordigd, anderzijds groeien programmeertalen langzamerhand uit de wieg van het technische keurslijf. De aansluiting tussen de conceptuele taal waarin het model is beschreven en de programmeertaal waarin het moet worden weergegeven, bepaalt in hoge mate de produktiviteit van de ontwikkeling van een informatiesysteem. Goede aansluiting bevordert de produktiviteit. Een voorbeeld van produktiviteitsverbetering in deze richting toont LINC (Logic and Information Network Compiler), een produkt van Burroughs Corporation. LINC is een applicatiegenerator [9]. De specificaties van een model worden in een taal (Linc Definition Language) beschreven, waarna vervolgens een compleet en werkend systeem wordt gegenereerd, dat gebruik maakt van een netwerk-model-database en een data communicatie netwerk.
Dit artikel beschrijft waarmee, in LINC, de structuur- en gedragsaspecten van een conceptueel model kunnen worden vastgelegd.

In het hiernavolgende hoofdstuk introduceren we de centrale bouwstenen van LINC: de component en de event. In hoofdstuk 3 komt een methode aan de orde waarmee een gedetaileerd conceptueel schema kan worden ontworpen, waarin structuur- en gedragskenmerken zijn vertegenwoordigd. Hoofdstuk 4 beschrijft, aan de hand van een voorbeeld, wat components en events zijn in termen van de structuur- en gedragskenmerken van het conceptuele schema. Vervolgens geeft hoofdstuk 5 in kort bestek het gebruik van LINC weer, waarna tenslotte in hoofdstuk 6 een samenvatting en enige conclusies volgen.

2. BOUWSTENEN

Een LINC-systeem is primair een interactief systeem. Het openbaart zich middels een reeks beeldscherm gestuurde activiteiten zoals menuschermen, invoerschermen, helpschermen e.d. De schermen vormen het medium voor manipulaties met (database-)gegevens. Ook in de opbouw van een LINC-systeem staan de schermen centraal. Ze bepalen de subsystemen. Ieder subsysteem bestaat uit drie elementen: een schermbeschrijving, een beschrijving van (database-)gegevens die relevant zijn voor het scherm en een beschrijving van manipulaties op die gegevens. De subsystemen zijn in twee klassen onder te brengen: COMPONENTS en EVENTS. Alles in LINC draait om COMPONENT's en EVENT's. Letterlijk vertaald zijn het bestanddelen en gebeurtenissen.
Op het eerste gezicht lijken het duidelijke begrippen: components beschrijven de (bedrijfs-)gegevens en events geven aan wat er met die gegevens gebeurt. Maar er zitten addertjes onder het gras.
In [4] wordt de volgende omschrijving van deze begrippen gehanteerd:

COMPONENT - vastlegging en onderhoud van gegevens omtrent het statische deel van een bedrijf.

EVENT - vastlegging van de handelingen en gebeurtenissen binnen het bedrijf.

Aangezien beide bouwstenen, zoals eerder aangegeven, drie aspecten beschrijven, namelijk: een scherm, (database-)gegevens en een aantal instructies, betekent dit dat het verschil tussen component en event niet principieel van aard is (in de zin van bestanddeel versus handeling), maar eerder gezocht moet worden in hun intensie, het doel waarvoor ze gebruikt worden. In dat licht zijn er verschillen te noteren. Een component beschrijft stamgegevens en verzorgt het onderhoud daarop. Een aantal onderhoudsfuncties worden standaard toegevoegd (wijzigen, verwijderen en toevoegen). Bovendien is een stam-record direct opvraagbaar op grond van een identificerend gegeven. Eventgegevens zijn niet direct opvraagbaar (hooguit via een omweg: een daartoe in het leven geroepen index). Ze zijn daarvoor ook niet bedoeld. Eventgegevens leggen veelal relaties tussen stamgegevens, zoals een order een relatie legt tussen een debiteur en een aantal artikelen. Het procedurele gedeelte van een event beschrijft dan ook handelingen (anders dan onderhoud). De onscherpe scheiding tussen beide bouwstenen betekent dat er zich situaties zullen voordoen in het ontwerpproces waarin de keuze tussen component en event arbitrair is. Verderop in dit artikel wordt hierop teruggekomen (zie 4.3). Al met al betekent dit dat de namen COMPONENT (bestanddeel) en EVENT (gebeurtenis) niet geheel borg staan voor hun betekenis.

Een nadere beschouwing leert dat beide begrippen verwantschap vertonen met entiteiten en relaties uit het Entiteit/Relatie (ER-)model [3] en de toestandsovergangen in Petri-netten [8]. Het ER-model representeert met name het statische deel van de bedrijfsproblematiek, een Petri-net brengt de handelingen en gebeurtenissen in kaart.
Recent onderzoek op het gebied van conceptuele modellen concentreert zich vooral op de vereniging van de statische en dynamische karakters van modellen in een uniforme benadering [1] [7] [10] [11] [12].
Solvberg en Kung [12] beschrijven een methode om zowel structuur als gedragsaspecten te modelleren. De methode kenmerkt zich door een samengaan van aspecten uit de theorie van de Petri-netten en het ER-model en vormt daarmee een basis ter verdere bestudering van de LINC-begrippen component en event.

In het hiernavolgende worden de twee genoemde theorieën geïntroduceerd, vervolgens komt het conceptuele model van Solvberg en Kung aan de orde, waarna een aangepaste versie van een voorbeeld uit het laatst genoemde artikel wordt gepresenteerd, vervolgens gemodelleerd en met LINC gerealiseerd. Eerst daarna kan worden vastgesteld wat components en events zijn in termen van entiteit en relatie (ER-model) en toestandsovergangen (Petri-netten).

3. MODELLEREN

3.1 Entiteit en Relatie

Een ER-model onstaat door een beschouwd deel van de werkelijkheid af te beelden op een geheel van entiteiten die aan elkaar gerelateerd zijn. Een entiteit kan een object zijn (b.v. een zekere auto), maar ook een subject (b.v. een eigenaar) of een gebeurtenis (b.v. een rit). Iets wat, gegeven een zekere context bestaansrecht heeft. Met andere woorden het bepalen van entiteiten is, evenals de hierna te definieren relaties, een subjectief abstractieproces.
Individuele entiteiten worden in klassen ondergebracht die we entiteit-types noemen. Zo vormen alle beschouwende auto's een klasse: het entiteit-type AUTO. Evenzo duidt het entiteit-type EIGENAAR de klasse van alle beschouwde eigenaren aan.
Brengen we individuele entiteiten met elkaar in verband dan ontstaan relaties die we kunnen classificeren tot relatie-types. Formeel gedefinieerd: een relatie-type R tussen de entiteit-types E1,E2,..,En wordt bepaald door $R \subset ((e1, e2,..en) \mid e1 \in E1, e2 \in E2,..,en \in En)$.
Toegepast betekent dit dat een bezit ($r=(e1,e2) \in R$) een samengaan is van een eigenaar ($e1 \in E1$) en een auto ($e2 \in E2$) en dus een relatie. Het is duidelijk dat de zo gedefinieerde entiteit- en relatie-types nog maar weinig betekenen. Ze krijgen meer betekenis door er kenmerken aan toe te voegen die we attributen noemen. We onderscheiden hierbij identificerende en overige attributen. De combinatie van de identificerende attributen van de gerelateerde entiteit-types vormt een identificatie voor het relatie-type (b.v. E1(<u>A1</u>,A2), E2(<u>A3</u>,A4) en R(<u>A1, A3</u>,A5), waarbij de onderstrepingen de identificerende kenmerken aangeven). Een attribuut is ook een klasse, namelijk een klasse van attribuutwaarden (b.v. e1(a1,a2) en r(a1,a3,a5)).
M.a.w. een zekere eigenaar (jansen, BE) uit de klasse EIGENAAR (<u>naam</u>, rijbevoegdheid) is in het bezit (jansen, mazda 323, 20/04/85, utrecht) uit de klasse BEZIT (<u>naam, type</u>, aankoopdatum, plaats) van een zekere auto (mazda 323, 1980) uit de klasse AUTO (<u>type</u>, bouwjaar).

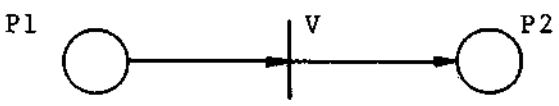
Schematisch wordt dit als volgt weergegeven (zie figuur 1):

figuur 1

In de loop der jaren zijn er veel publicaties verschenen waarin uitbreiding van het oorspronkelijke ER-model (gepresenteerd in [3]) worden voorgesteld. Ze verhelen niet dat het ER-model een statische weergave van de beschouwde werkelijkheid is. Met het ER-model is het niet goed mogelijk om tijdsafhankelijke fenomenen te modelleren. Ook een gebeurtenis als de hierboven genoemde rit wordt als een statisch object gezien. Als we gebeurtenissen als gedragspatronen in beeld willen brengen, dan moeten we andere methoden hanteren. Bijvoorbeeld het Petri-net principe.

3.2 Gebeurtenissen

Een Petri-net in z'n eenvoudigste vorm ziet er als volgt uit (zie figuur 2).



figuur 2

P1 en P2 in figuur 2 zijn, niet noodzakelijk disjuncte, plaatsen (places), waarbij een plaats een fenomeen representeert, bijvoorbeeld een gegeven, in een bepaalde toestand.

De veranderingen die een fenomeen ondergaat in de tijd kunnen we weergeven als toestandsovergangen. In het figuur vertegenwoordigt P1 het fenomeen vóór de toestandsovergang V en P2 erna. Aan de overgang V kunnen ook meerdere ingangsplaatsen P11, P12, ..,P1n en meerdere uitgangsplaatsen P21, P22,..,P2m verbonden zijn.
De toestandsovergangen worden bestuurd met behulp van zogenaamde besturingstekens (tokens). De overgang V kan alleen dan plaatsvinden als alle P1i (i=1,2,..,n) over een besturingsteken beschikken. De aanwezigheid van besturingstekens is dus bepalend voor een toestandsverandering. Na de overgang bevindt zich in iedere uitgangsplaats een besturingsteken, in de ingangsplaatsen echter niet meer: de verandering heeft plaatsgevonden.
Het principe van Petri-netten kan worden toegepast om systemen te modelleren, in het bijzonder twee aspecten van die systemen: gebeurtenissen en condities met hun onderlinge relaties. Gebeurtenissen vinden altijd plaats onder zekere voorwaarden en hebben nieuwe voorwaarden en transport van (besturings) gegevens tot gevolg.
Een Petri-net is niet geschikt om statische gegevens en relaties tussen die gegevens in kaart te brengen, het beschrijft voornamelijk het dynamisch karakter van een systeem.

3.3 De synthese

De waarheid ligt in het midden. Een combinatie van het ER-model en het Petri-net principe kan leiden tot een conceptueel model waarin zowel de structuur- als de gedragsaspecten zijn vertegenwoordigd. Het recept van Solvberg en Kung om tot een dergelijk conceptueel model te komen laat zich als volgt formuleren:

1. Tussen een plaats en een entiteit-type of relatie-type kan een lijn worden getekend met een pijl in de richting van het type. De plaats krijgt een symbool toegevoegd, onderscheiden in een kleine letter (x,y,z,..) of een hoofdletter (X,Y,Z,..). Een kleine letter geeft een individuele entiteit of relatie aan, een hoofdletter representeert een klasse van entiteiten of relaties. Als de plaats een kleine letter bevat, zeg x, dan wordt de voornoemde lijn met een $\in$-teken gemerkt. Dit geeft aan dat de lijn een individuele verwijzing is. Bevat de plaats daarentegen een grote letter, zeg X, dan krijgt de lijn een =-teken, daarmee klasseverwijzing aangevende.

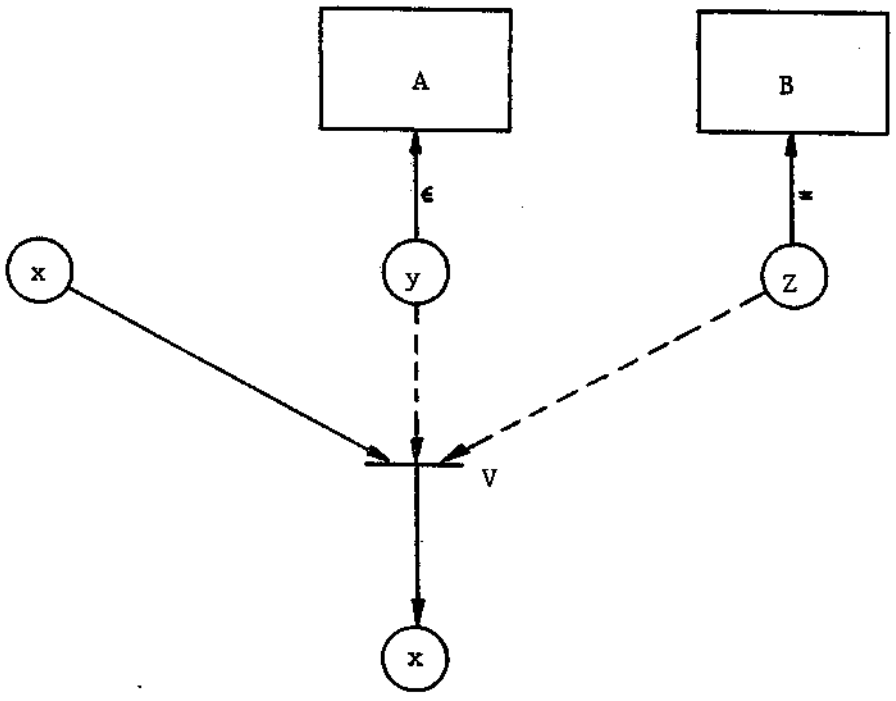
2. Naar een plaats die een hoofdletter bevat, kan een overgang alleen refereren. Dit wordt weergegeven door een onderbroken lijn van de plaats naar de overgang.

Als aan de plaats een kleine letter is toegevoegd, betekent dit:
a. dat door de overgang naar de plaats wordt gerefereerd (onderbroken lijn) of
b. dat de aangegeven entiteit door de overgang wordt geabsorbeerd en weer kan verschijnen (al dan niet gewijzigd) in een uitgangsplaats (getrokken lijn van de ingangsplaats naar de overgang).

3. Aan iedere toestandsovergang wordt precies één pre-conditie en precies één post-conditie toegevoegd. De pre-conditie beschrijft de voorwaarden waaraan de ingangsplaatsvariabelen moeten voldoen om de overgang mogelijk te maken en de post-conditie beschrijft de situatie erna.

4. Een overgang kan alleen plaatsvinden als iedere ingangsplaats een besturingsteken heeft en als tevens aan de pre-conditie is voldaan. Alleen de besturingstekens van de plaatsen die door middel van een getrokken lijn met de overgang zijn verbonden, worden verwijderd. De andere ingangsplaatsen behouden het besturingsteken.
Na de overgang staat in iedere uitgangsplaats een besturingsteken en voldoet de post-conditie.

Een eenvoudig voorbeeld verduidelijkt deze aannames (zie figuur 3):



figuur 3

Voor de overgang V in figuur 3 geldt b.v. de volgende pre-conditie:

$$(x.p = y.p) \wedge \neg(\exists z)\ (z \in Z \wedge z.p = x.p).$$

Met andere woorden als er voor de x-entiteit een bijbehorende y-entiteit is en geen bijbehorende entiteit in Z. Is deze pre-conditie waar en zijn bovendien P1, P2, en P3 voorzien van een besturingsteken, dan kan V plaatsvinden. De overgang heeft tot gevolg dat x en het besturingsteken in P1 worden overgezet naar P4, waarna de post-conditie voldoet. P2 en P3 behouden hun besturingstekens.
Een voorbeeld van een post-conditie na de overgang V is de expressie (x.q = y.r), die wil zeggen dat de waarde q in x nu gelijk is aan de waarde r in y.
Een uitgebreider voorbeeld komt in het volgende hoofdstuk aan de orde.

4. EEN TOEPASSING

4.1 De bedrijfssituatie

Een handelsbedrijf in huishoudelijke artikelen heeft een aantal geregistreerde klanten. Deze klanten worden geïdentificeerd door een klantnummer en hebben verder slechts een adreskenmerk. Ieder artikel heeft de kenmerken: nummer, prijs en voorraad.
Onder de fictieve restrictie dat een klant slechts één order per dag kan plaatsen en een order nooit meer dan één artikel betreft, bevat een order de volgende kenmerken: klantnummer, plaatsingsdatum, artikelnummer, hoeveelheid, bedrag, leverdatum en orderstatus.
De restrictie in aanmerking nemende, zijn klantnummer en plaatsingsdatum tesamen nodig en voldoende om een order te identificeren. De orderstatus geeft aan of de order kan worden uitgevoerd of dat de levering van het bestelde is opgeschort.

De activiteiten van het bedrijf zijn in twee clusters onder te brengen:

1. Plaatsing van een order. Op een bepaalde datum wil een klant levering van een zekere hoeveelheid van een artikel.
   Een direct gevolg van deze plaatsing is een klassificatie van de order:
   a. de order wordt uitgevoerd (orderstatus = levering) als de order wordt geplaatst door een geregistreerde klant, het een leverbaar artikel betreft en als de bestelde hoeveelheid de voorraad niet overschrijdt. Uitvoering betekent dat de leverdatum en het bedrag worden vastgesteld en de voorraad wordt verminderd met het geleverde.
   b. levering van de order wordt opgeschort (orderstatus = uitstel) als, bij een geregistreerde klant en een leverbaar artikel, de gevraagde hoeveelheid groter is dan de voorraad.
   c. de order wordt geweigerd (en geretourneerd) als degene die de order plaatst geen geregistreerde klant is of als het betreffende artikel niet door het bedrijf gevoerd wordt.

Plaatsing van een order heeft altijd één van de drie hierbovengenoemde situaties tot gevolg.

2. Verhoging van de voorraad van een artikel.
   Een direct gevolg van deze activiteit is de levering van uitgestelde orders waarvan de gevraagde hoeveelheid na de verhoging gelijk of lager is dan de voorraad.

4.2 Het conceptuele model

Analyserend zijn een drietal klassen van entiteiten en/of relaties te onderkennen: klanten, artikelen en orders.
Klanten en artikelen kunnen zonder problemen tot entiteiten worden bestempeld (KLANT respectievelijk ARTIKEL). Bij een order moeten we voorzichtiger zijn. ORDER is geen relatie-type tussen KLANT en ARTIKEL, omdat een order uniek wordt bepaald door een klantnummer en een datum. ORDER is een KLANT-afhankelijk entiteit-type ('existence dependency', zie [3]). Het complete model van de toepassing naar het in paragraaf 3.3 beschreven recept is afgebeeld in figuur 4, waarbij, ten behoeve van de eenvoud van het schema, de formeel aanwezige, maar geen attributen bevattende relaties tussen ORDER en KLANT en tussen ORDER en ARTIKEL zijn weggelaten.
De toestandsovergangen V1a, V1b en V1c in figuur 4 corresponderen met respectievelijk de activiteiten 1a, 1b en 1c in de beschrijving van de toepassing. Deze overgangen zijn een gevolg van de plaatsing (externe invoer in plaats P1).
Het verhogen van de voorraad van een artikel (overgang V2, als het artikel bekend is) heeft als direct gevolg de levering van uitgestelde orders (V2a). Als de ingevoerde voorraadverhoging een artikel betreft dat niet door het bedrijf wordt gevoerd, dan wordt de verhoging geweigerd (overgang V2b).
Twee overgangen beschouwen we in detail: V1a en V2a.
De overgang V1a dient, conform activiteit 1a, aan de volgende preconditie te voldoen:

(v.nummer=x.klantnummer) ∧ (u.nummer=x.artikelnummer) ∧
(u. voorraad ≥ x.hoeveelheid)

terwijl de post-conditie is te formuleren als:

(x.leverdatum=datum) ∧ (x.bedrag=x.hoeveelheid*u.prijs) ∧
(u.voorraad=u.voorraad-x.hoeveelheid) ∧ (x.orderstatus='geleverd').

Merk op dat de overgang V1a drie ingangs- en twee uitgangsplaatsen heeft. Naar de ingangsplaatsvariabele v wordt alleen gerefereerd. De v-ingang behoudt dus het besturingsteken. De variabele x (een order) wordt geabsorbeerd door V1a en inclusief het besturingsteken overgezet naar een uitgangsplaats (gemerkt x), waarbij de datum aan de order wordt toegevoegd, het bedrag uitgerekend, de voorraad verminderd en de order de status 'geleverd' krijgt.
De ingangsplaats aangegeven met u is tevens uitgangsplaats, immers de bijgewerkte voorraad moet in het artikel verwerkt worden. Het bestu-

ringsteken komt dientengevolge ook weer op de betreffende ingang.
Omdat de u-ingang het besturingsteken steeds weer terugkrijgt en de v-ingang z'n besturingsteken behoudt, kan de plaatsing van een nieuwe order, als aan de pre-conditie is voldaan, opnieuw V1a tot gevolg hebben.
Aan de overgang V2a is de volgende pre-conditie toegevoegd:

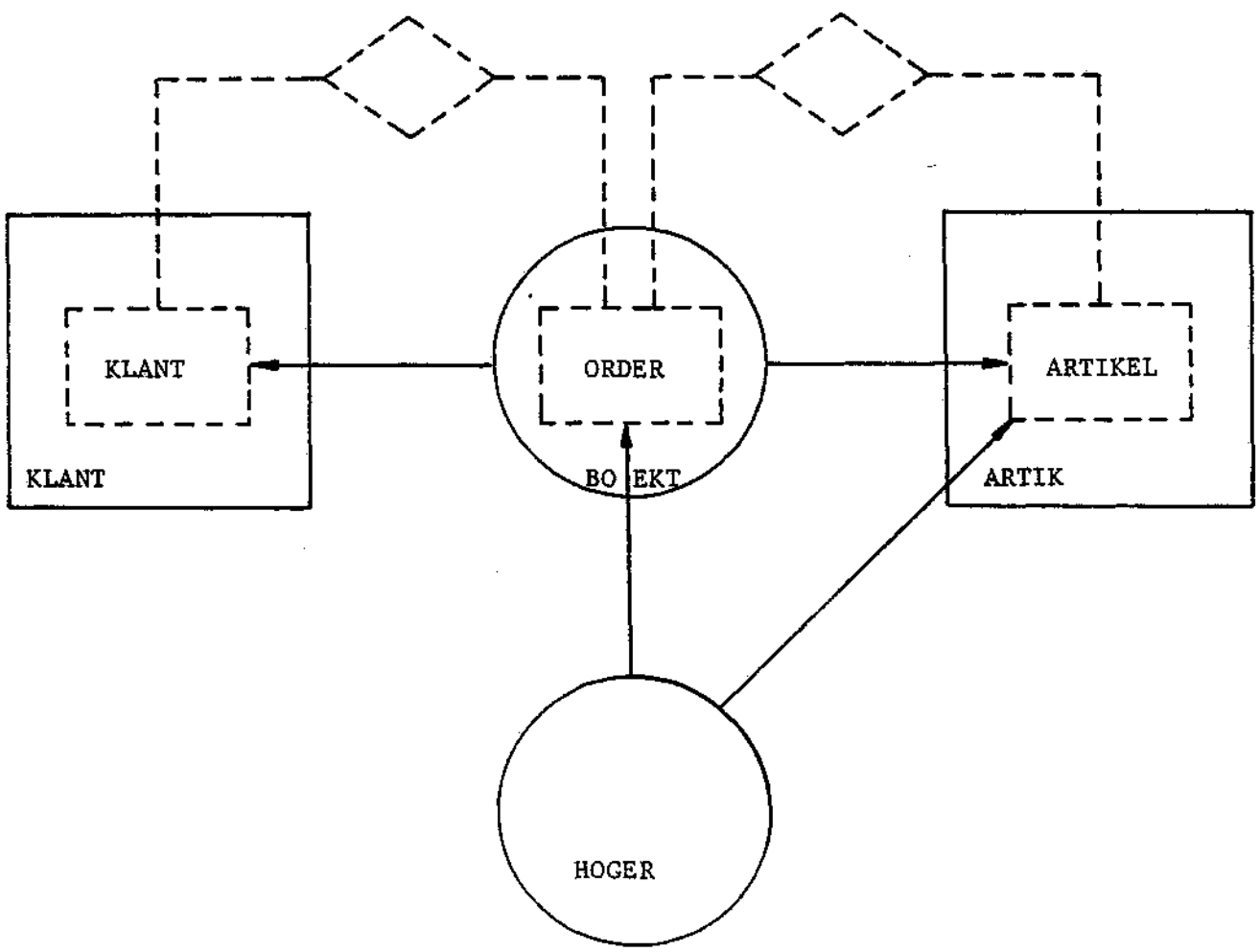
(u.nummer=x.artikelnummer) $\wedge$ (u.voorraad $\geq$ x.hoeveelheid) $\wedge$
(x.orderstatus='uitstel')

De post-conditie van V2a is dezelfde als die van V1a.
Op te merken valt dat de beide ingangsplaatsen van V2a tevens uitgangsplaatsen zijn. Dit betekent dat de besturingstekens steeds weer op de respectievelijke ingangen terugkeren, zodat voor een zeker artikel alle uitgestelde orders kunnen worden geleverd als de voorraad dat toestaat. De verandering van status van 'uitstel' naar 'geleverd' staat daarbij garant voor de eenmalige behandeling van een order.

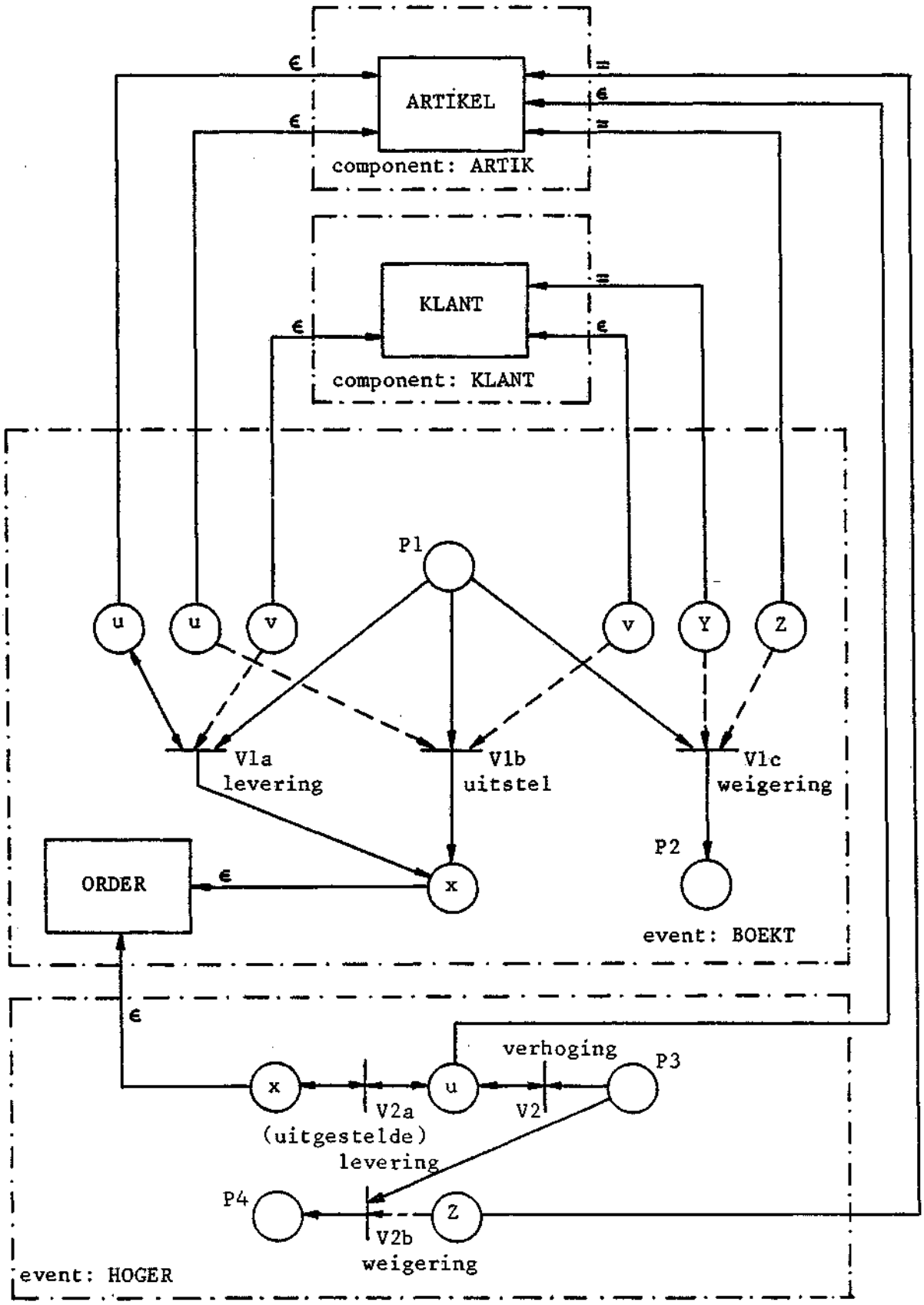
4.3 Een component/event model

Er zijn verschillende mogelijkheden om, uitgaande van het model in figuur 4, events en components te bepalen. Voor enkele components ligt de keuze voor de hand, namelijk artikelen en klanten (respectievelijk de components ARTIK en KLANT in figuur 4). Dit zijn statische gegevens waarop onderhoudsfuncties van toepassing zijn. Het betekent overigens niet dat components alleen entiteit-types kunnen beschrijven. Stel bijvoorbeeld dat we de toepassing uitbreiden met een relatie-type KORTING, waarmee voor iedere klant en ieder artikel een kortingspercentage wordt aangegeven. De enige bewerking die de percentages hier ondergaan is onderhoud. Bovendien moeten ze hiertoe direct opvraagbaar zijn. KORTING kan in dit geval goed in een component worden opgenomen.
De plaatsing van een order betekent dat P1 uit externe invoer een waarde krijgt, op grond waarvan tot levering, uitstel of weigering wordt overgegaan (V1a, V1b en V1c respectievelijk). De laatstgenoemde overgangen betrekken hun invoer van de components KLANT en ARTIK en van de hierboven aangeduide externe invoer aangegeven door P1. Dit is een argument om V1a, V1b em V1c op te nemen in een event, de event BOEKT.
Voor de cluster (V2, V2a, V2b) is eenzelfde redenering te houden. Deze cluster vormt de event HOGER.
Blijft over het entiteit-type ORDER: component of event?
Voor de toepassing verkiezen we de opname van ORDER in de event BOEKT, omdat er een sterke binding is tussen de activiteiten rondom een order (plaatsing e.d.) en de eigenlijke opslag van de ordergegevens. Er zijn

overigens meerdere oplossingen mogelijk.
Het model is in figuur 5 eenvoudiger weergegeven. Hierin is de pijl een referentie-aanduiding; de cirkel geeft een event aan; een rechthoek bepaalt een component en een met onderbroken lijnen getekende rechthoek of ruit geeft het entiteit- respectievelijk het relatie-type aan.
De event HOGER beschrijft dus geen in de database voorkomende gegevens. In LINC krijgt zo'n event het predicaat 'usage=input'.



figuur 5


ARTIKEL
component: ARTIK
KLANT
component: KLANT
P1
u
u
v
v
Y
Z
V1a
levering
V1b
uitstel
V1c
weigering
ORDER
x
P2
event: BOEKT
verhoging
P3
x
u
V2a
V2
(uitgestelde)
levering
P4
Z
V2b
weigering
event: HOGER

figuur 4

4.4 Het LINC-programma

Een LINC-programma wordt globaal gekenmerkt door de volgende indeling:

- OPTION — aanwijzingen voor de LINC-compiler;
- USER — naamgevingen als user, database enz.;
- NETWORK — aanduiding van het data communicatie netwerk;
- PROFILE — indexen om events en components te benaderen;
- GLOBAL — algemene data-definities

Per component of event:

- INTERFACE — gespecificeerde (standaard-)functies, b.v. REFRESH verzorgt automatische schoning van schermvelden;
- DATA — schermindeling en gegevensbeschrijving;
- LOGIC — beschrijving van de bewerking

In het hiernavolgende zullen de belangrijkste aspecten van het programma (zie appendix) aan de orde komen waarbij steeds verwezen wordt naar het model in figuur 4.
De components ARTIK en KLANT beschrijven een paar standaard functies in het interface-gedeelte, een aantal display-instructies ten behoeve van de schermindeling en een aantal gegevens met de 'ordinate' als sleutel (data-gedeelte). Ze bevatten geen logic, maar LINC staat de onderhoudsfuncties toevoegen, wijzigen en verwijderen toe. Ten behoeve van de eenvoud van het schema zijn deze LINC-functies niet getekend in figuur 4.
De event HOGER heeft geen directe koppeling met database-gegevens (usage=input). Het intoetsen van een artikelnummer en een voorraadtoename heeft wel een automatische zoekactie in de database tot gevolg.
In LINC heet dit: AUTOMATIC.LOOKUP. Het wordt bewerkstelligd door het gelijk zijn van de benamingen ARTNR in de event HOGER en in de component ARTIK. Als de zoekactie geen succes heeft zal de verhoging niet plaatsvinden: ze wordt geweigerd (overgang V2b). Heeft de zoekactie wel succes, dan wordt de verhoging gerealiseerd (overgang V2) door de ADD-(verhoog) en de FLAG-(opslag in de database) instructies in het logic-gedeelte van de event.
De tweede overgang (V2a) wordt beschreven in het gedeelte DETERMINE tot en met END;. Hierin bepaalt de DETERMINE.EVERY alle orders met het juiste artikelnummer, via de eerder gedefinieerde BOEKINDEX. De twee DO.WHEN's beschrijven de overige voorwaarden voor de overgang.
Het plaatsen van een order in de event BOEKT heeft eveneens een automatische zoekactie tot gevolg. Het resultaat hiervan kan de weigering

van de order betekenen (overgang V1c). De beide overgebleven toestandsovergangen in de event BOEKT (V1a en V1b) worden beschreven onder respectievelijk de eerste en de tweede DO.WHEN.
De besturing van het programma neemt hier, in opdracht van expliciete instructies als DETERMINE.EVERY of impliciete instructies zoals de invoeracceptatie, de rol van het besturingsteken-mechanisme waar.

5. NOG EEN BEETJE LINC

Uit het voorgaande blijkt dat LINC zelf een aantal functies verzorgt (b.v. AUTOMATIC.LOOKUP). Deze functies kunnen we beïnvloeden door het specificeren van logic in de betreffende event of component.
Logic is in een aantal categorieën te verdelen. Ten eerste de 'pre-screen' logic, bedoeld om bepaalde schermgegevens voorafgaande aan de weergave op het scherm te vullen met waarden.
Vervolgens de 'pre-linc' logic, waarmee LINC-functies kunnen worden uitgeschakeld en gespecificeerde functies kunnen worden uitgevoerd.
Ten slotte de 'post-linc' logic, waarin gespecificeerde functies worden uitgevoerd na de automatische activiteiten van LINC.
De LINC-taal is uiteraard veel uitgebreider dan in een artikel als het onderhavige naar voren kan komen. Ze omvat bijvoorbeeld een uitgebreide netwerk-definitietaal om een datacommunicatie-netwerk [5] vast te leggen en een rapporttaaal om de uitvoer te beschrijven [6].
De huidige interactieve middelen om een LINC-programma te creëeren zijn echter nog primitief. Behoudens eenvoudige schermformatteringsfaciliteiten, dienen programma's met behulp van een editor-programma te worden gemaakt. Dit heeft bovendien als bezwaar dat syntactische fouten pas in de daarna uit te voeren vertaalslagen worden geopenbaard. Syntactische fouten moeten eigenlijk zoveel mogelijk tijdens het interactief formuleren worden gesignaleerd, om te kunnen worden gecorrigeerd.
Met de inmiddels ontwikkelde LINC-2 versie is juist aan deze aspecten veel aandacht geschonken, zodat LINC meer en meer evolueert naar een volwaardig applicatie ontwikkel systeem [2].

6. SAMENVATTING EN CONCLUSIES

De fundamentele bouwstenen van LINC zijn de component en de event. Ofschoon de betekenis van deze begrippen op het eerste gezicht geen problemen biedt (een component beschrijft gegevens en een event gebeurtenissen), staan de namen component en event zeker niet borg voor hun betekenis. De beide bouwstenen blijken ieder zowel een gegevens beschrijvend als een procedure beschrijvend aspect te beschikken.
Een nadere beschouwing levert op dat een component vergelijkbaar is met een entiteit- of een relatie-type uit het Entiteit/Relatie (ER-) model [3], waaraan (onderhouds-)functies zijn toegevoegd. Voor een deel zijn dit standaardfuncties van LINC. De overige functies moeten in een procedure gedeelte van de component worden beschreven.
Waar het accent bij een component ligt op de gegevensbeschrijving, bevindt het accent bij een event zich juist op de verwerking van gegevens. Ook hier is veelal sprake van een entiteit- of een relatie-type waaraan functies zijn toegevoegd. Dit zijn echter geen onderhoudsfuncties, maar verwerkingsfuncties. Bovendien kan een event bestaan zonder entiteit- of relatie-typebeschrijvend gedeelte.
De onderhoudsfuncties van een component en vooral de verwerkingsfuncties van een event kunnen we in detail visualiseren door middel van een Petri-net [8]. Een combinatie van het ER-model en het Petri-net principe levert een basis om alle aspecten van components en events in beeld te brengen. In dit artikel is hiertoe uitvoerig gebruik gemaakt van een recente publicatie van Solvberg en Kung [12] aangaande zo'n combinatie. De in deze publicatie beschreven methode kan met succes worden toegepast om een systeem gedetailleerd te conceptualiseren met betrekking tot de structuur- en gedragsaspecten, waarna het resultaat (een conceptueel model) eenvoudig kan worden omgezet in eerst een LINC-model en vervolgens een LINC-programma.

7. LITERATUUR

[1] Antonellis de, V., G. Degli Antoni, G. Mauri, and B. Zonta, "Extending The Entity-Relationship Approach to take into account Historical Aspects of Systems", in: "Entity-Relationship Approach to Systems Analysis and Design", P.P. Chen (ed.), First Int. Conf. on ERA, Dec. 10-12 1979, North-Holland Publ. Company, 1980.

[2] Cardenas, A.F. and W.P. Grafton, "Challenges and Requirements for New Application Generators", AFIPS, Proceedings of the 1982 National Computer Conference, June 7-10, 1982, Houston, Texas.

[3] Chen, P.P., "The Entity-Relationship Model - Toward a Unified View of Data", ACM Transactions on Database Systems. Vol. 1, No.1, March 1976.

[4] LINC-Basis, deel I en II, Doc. No. EP4404-1 resp. EP4404-2, Burroughs Corporation, 1985.

[5] Logic and Information Network Compiler (LINC), reference Manual Relative to the 10.2 Software Release, Doc. No. 1149903, Burroughs Corporation, January 1984.

[6] Logic and Information Report Compiler (LIRC), Reference Manual Relative to the 10.2 Software Release, Doc. No. 1154341, Burroughs Corporation, October 1983.

[7] Olle, T.W., H.G. Sol and A.A. Verrijn-Stuart, "Information Systems Design Methodologies: A Comparative Review", Proc. of the IFIP TC8 WC on Comparative Review of Information Systems Design Methodologies, Noordwijkerhout, the Netherlands, 10-14 May, 1982, North Holland Publishing Company, 1982.

[8] Peterson, J.L., "Petri Nets", Computing Surveys, Vol. 9, No. 3. September 1977.

[9] Rudolphy, E., "Productivity in Computer Application Development", Working Paper No. 9, Dept. of Management Studies, Univ. of Auckland March 1983.

[10] Sakai, H., "A Method for Entity-Relationship Behavior Modelling", in "Entity-Relationship Approach to Software Engineering", C.G. Davis, S. Jajoda, P.A. Ng and R.T. Yeh (eds.), Third Int. Conf. on ERA, October 5-7, 1983, North-Holland Publ. Company, 1983.

[11] Sernadas, A., J. Rubenko and A. Olive, "Information Systems: Theoretical and Formal Aspects", Proc. of the IFIP WG 8.1 WC on Theoretical and Formal Aspects of Information Systems, April 16-18, Barcelona, Spain, 1985, North-Holland Publ. Company, 1985.

[12] Solvberg, A. and C.H. Kung, "On Structural and Behavioral Modelling of Reality", Technical Report No. 25/84, Dept. of Comp. Science, The Norwegian Institute of Technology, Trondheim.

```
FRIDAY    02 AUG 85 **   B U R R O U G H S   L I M I T E D   **        PAGE    1


                                                                SECTION 00:00:00
      SECTION 00:00:00


   OPTION; ZERO.SUPPRESS
   OPTION; NEW.DATABASE
   OPTION; NO.INPUT.LOG
   OPTION; LIST

   USER;         (VU)
   DATABASE;     EDUDB
   SYSTEM.NAME; SYEDU
   DF;           VUSER
   NETWORK;      LINCNDL

   PROFILE = BOEKINDEX
           EVENT.NAME = BOEKT
           ORDINATE   = ARTNR


   GLOBAL.SD; LEVERING  (L) ED=A LE=1
   GLOBAL.SD; UITSTEL   (U) ED=A LE=1
```

SECTION 00:00:00

SECTION 00:00:00

```
COMPONENT  = ARTIK    COMPONENT  = ARTIK
-----------------     -----------------
REFRESH;
RECALL;
: DISPLAY CONST NO 001  IS THE FIRST IN THIS SCREEN
  DISPLAY  =(*** ARTIKELGEGEVENS ***)
                       LI =06  POS=43  RV; RA;
  DISPLAY  =( )        LI =06  POS=46  SE; RA;
  DISPLAY  =(ARTIKELNUMMER)
                       LI =10  POS=29  RA;
  DISPLAY  =(PRIJS)    LI =11  POS=21  RA;
  DISPLAY  =(VOORRAAD) LI =12  POS=24  RA;
  ORDINATE = ARTNR     LE =05  ED=N    LI =10  POS=47  RA; -
  DATA     = PRIJS     LE =07  ED=N    LI =11  POS=49  RA; DE=2
  DATA     = VOORR     LE =05  ED=N    LI =12  POS=47  RA;
```

SECTION 00:00:00

SECTION 00:00:00

```
COMPONENT  = KLANT    COMPONENT  = KLANT
-----------------     -----------------
REFRESH;
RECALL;
: DISPLAY CONST NO 006  IS THE FIRST IN THIS SCREEN
  DISPLAY  =(*** KLANTGEGEVENS ***)
                       LI =07  POS=46  RV; RA;
  DISPLAY  =( )        LI =07  POS=49  SE; RA;
  DISPLAY  =(KLANTNUMMER)
                       LI =09  POS=19  BR; RA;
  DISPLAY  =(ADRES)    LI =10  POS=13  RA;
  ORDINATE = KLANTNR   LE =05  ED=N    LI =09  POS=31  RA;
  DATA     = ADRES     LE =40  ED=A    LI =10  POS=66  RA;
```

SECTION 00:00:00

SECTION 00:00:00

```
EVENT      = HOGER    EVENT      = HOGER
-----------------     -----------------
USAGE = INPUT
REFRESH;
RECALL;
: DISPLAY CONST NO 010  IS THE FIRST IN THIS SCREEN
  DISPLAY  =( *** VOORRAADVERHOGING ***)
                       LI =07  POS=43  RV; RA;
  DISPLAY  =( )        LI =07  POS=46  SE; RA;
  DISPLAY  =(ARTIKELNUMMER)
                       LI =10  POS=32  RA;
  DISPLAY  =(VERHOGING) LI =11  POS=28  RA;
  DATA     = ARTNR     LE =05  ED=N    LI =10  POS=41  RA;
  DATA     = TOENAME   LE =05  ED=N    LI =11  POS=41  RA;

SETUP.DATA; BEDR      ED=N LE=9 DE=2
SETUP.DATA; NIEUWVOOR ED=N LE=5

ADD; TOENAME ARTIK.VOORR GIVING; NIEUWVOOR
FLAG; NIEUWVOOR ARTIK.VOORR


DETERMINE; EVERY BOEKINDEX (ARTNR)
        DO.WHEN; EVENT.TOESTAND = UITSTEL AND
        DO.WHEN; EVENT.AANTAL NOT > ARTIK.VOORR
                 SUBTRACT; EVENT.AANTAL ARTIK.VOORR GIVING; NIEUWVOOR
                 MULTIPLY; EVENT.AANTAL ARTIK.PRIJS GIVING; BEDR
                 FLAG; NIEUWVOOR ARTIK.VOORR
                 FLAG; LEVERING EVENT.TOESTAND
                 FLAG; INPUT-DATE EVENT.LEVERDAT
                 FLAG; BEDR EVENT.BEDRAG
        END;
END;
```

```
FRIDAY    02 AUG 85 **   B U R R O U G H S   L I M I T E D   **        PAGE    5


                                                                SECTION 00:00:00
   SECTION 00:00:00


EVENT      = BOEKT    EVENT      = BOEKT
-----------------     -----------------
REFRESH;
RECALL;
: DISPLAY CONST NO 014  IS THE FIRST IN THIS SCREEN
  DISPLAY  =(*** ORDERBOEKING ***)
                       LI =07  POS=39  RV; RA;
  DISPLAY  =( )        LI =07  POS=42  SE; RA;
  DISPLAY  =(KLANTNUMMER)
                       LI =09  POS=27  RA;
  DISPLAY  =(HOEVEELHEID)
                       LI =10  POS=27  RA;
  DISPLAY  =(ARTIKELNUMMER)
                       LI =11  POS=29  RA;
  DISPLAY  =(PLAATSING DATUM)
                       LI =12  POS=31  RA;
  DATA     = KLANTNR   LE =05  ED=N    LI =09  POS=39  RA;
  DATA     = AANTAL    LE =04  ED=N    LI =10  POS=38  RA;
  DATA     = ARTNR     LE =05  ED=N    LI =11  POS=39  RA;
  DATA     = PLAATSDAT LE =07  ED=A    LI =12  POS=41  RA;
  DATA     = LEVERDAT  LE =07  ED=A        US= OUTPUT
  DATA     = BEDRAG    LE =09  ED=N        DE=2    US= OUTPUT
  DATA     = TOESTAND  LE =01  ED=A        US= OUTPUT

SETUP.DATA; BEDR ED=N LE=5

DO.WHEN; AANTAL NOT > ARTIK.VOORR
         MOVE; PLAATSDAT LEVERDAT
         SUBTRACT; AANTAL ARTIK.VOORR GIVING; BEDR
         FLAG; BEDR ARTIK.VOORR
         MOVE; LEVERING TOESTAND
         MULTIPLY; AANTAL ARTIK.PRIJS GIVING; BEDRAG
END;
DO.WHEN; AANTAL > ARTIK.VOORR
         MOVE; UITSTEL TOESTAND
END;
```