Het ontwerpen van bedrijfsregels met Blaze Advisor

Rule-engine scheidt applicaties van bedrijfslogica

Johan van der Graaf

Een voorbeeld hoe bedrijfsregels in een relationele database kunnen worden opgeslagen is te lezen in eerdere publicaties van Johan van der Graaf in DB/M 3-2003 'Het ontwerpen van een adaptief datamodel', en in DB/M 4-2003 'Het adaptief ontwerpen van bedrijfsregels'.
In het eerste artikel wordt de basis gelegd voor het ontwerpen van een adaptief datamodel, die in de tweede publicatie wordt uitgebouwd tot het adaptief ontwerpen van bedrijfsregels. In dit artikel gaat de auteur dieper in op de bedrijfsregels en de rule-engine die daarbij betrokken is.

Bedrijfsregels vormen een belangrijk onderdeel van productiesystemen. Zij kunnen op verschillende wijzen worden vastgelegd: b.v. in stored procedures of triggers op het niveau van de datalaag, of op het niveau van de toepassing in de applicatielaag. Beide zijn verschillende vormen van hardgecodeerde bedrijfsregels. Maar er zijn ook nog andere manieren om bedrijfsregels vast te leggen.

In 1990 besloot de Belastingdienst voor haar inkomstenbelastingsysteem, IBS, zelf een *rule-engine* te ontwerpen, omdat producten uit die tijd niet aan de gestelde eisen voldeden. Een rule-engine is een toepassing die bedrijfsregels evalueert; op basis van de uitkomst van de geëvalueerde bedrijfsregels neemt de toepassing die de rule-engine heeft aangeroepen een beslissing. Met andere woorden: een rule-engine kan ingezet worden op punten in een toepassing waar beslissingen genomen moeten worden. Bestaat de beslissing uit één simpele if-then-else structuur dan is het gebruik van een rule-engine een overkill. Echter, bestaat de beslissing uit tientallen, of zelfs honderden van deze *if-then-else* structuren, dan is het gebruik van een rule-engine zeker aan te bevelen. Dit is ook het geval wanneer de bedrijfsregels regelmatig wijzigen of pas laat ter beschikking komen. Door het gebruik van een rule-engine wordt de toepassingslogica – de applicatie – gescheiden van de bedrijfslogica, welke in de bedrijfsregels is vastgelegd. Op deze wijze bereikt men, dat de toepassing niet behoeft te worden herschreven zodra de bedrijfsregels wijzigen. In 2001 besloot de Belastingdienst voor haar aanslagbelastingsysteem, ABS, een rule-engine uit de markt te betrekken. Via een tenderprocedure is voor het product Blaze Advisor gekozen.

Ervaringen opgedaan door mij met beide rule-engines heeft geresulteerd in dit artikel. In het IBS-project ben ik verantwoordelijk geweest voor het opstellen van de software- architectuur en realisatie van de rule-engine. In het ABS-project was ik medeverantwoordelijk voor de software-architectuur, alsmede het integreren van Blaze Advisor in de nieuwe toepassing. Door middel van een voorbeeld wil ik duidelijk maken dat de syntax en de semantiek van de taal waarin de bedrijfsregels worden uitgedrukt een grote rol spelen.
Als voorbeeld wordt uitgegaan van een deel van het O-biljet voor het belastingjaar 2001, zoals weergegeven in Afbeelding 1.

Het getoonde deel van het aangiftebiljet laat zich eenvoudig uitdrukken in pseudocode zoals beschreven in Afbeelding 2.

Op basis van de getoonde sectie en pseudocode kunnen voor dit deel van het aangiftebiljet enkele bedrijfsregels worden afgeleid, voor de berekening van de velden 8a `totaalReisaftrek`, 8c `brutoReisEnFietsaftrek` en 8d `nettoReisEnFietsaftrek`, zie Afbeelding 1. Deze bedrijfsregels, in pseudocode, zijn beschreven in Afbeelding 3.

Afbeelding 1. Sectie 8 van het O-biljet, belastingjaar 2001.

```
type periodeVraag
(
      van                              datumveld,
      totEnMet                         datumveld
)
type array openbaarVervoer
(
      plaatsVanUwWerk                  tekstveld,
      enkeleReisafstandInKm            afstandveld,
      periode                          periodeVraag,
      reisaftrek                       bedragveld,
      dagenPerWeek                     dagenveld
)
type overigeReisaftrek                 bedragveld
type totaalReisaftrek                  bedragveld
type fietsaftrek                       bedragveld
type brutoReisEnFietsaftrek            bedragveld
type vergoedingenReisEnFietsaftrek     bedragveld
type nettoReisEnFietsaftrek
bedragveld
```

Afbeelding 2. Sectie 8 van het O-biljet in pseudocode.

```
totaalReisaftrek :
                = som(openbaarVervoer.reisaftrek)
                + overigeReisaftrek
brutoReisEnFietsaftrek :
                = min(maximaleReisEnFietsaftrek
                , totaalReisaftrek + fietsaftrek)
nettoReisEnFietsaftrek :
                = max(minimaleReisEnFietsaftrek
                , brutoReisEnFietsaftrek
                - vergoedingenReisEnFietsaftrek)
```

Afbeelding 3. Bedrijfsregels voor sectie 8 van het O-biljet.

De termen `maximaleReisEnFietsaftrek` en `minimaleReisEn-Fiets-aftrek` in de bedrijfsregels zijn constanten die volgens het aangiftebiljet resp. de waarden € 1659 en € 0 hebben. De termen `som`, '`min` en `max` zijn functies. Deze bepalen de som van alle `openbaar vervoer reisaftrek` voorkomens, de minimale resp. de maximale waarde van de functie argumenten.

Afhankelijk van de wijze waarop de bedrijfsregels toegepast moeten worden, ontstaan er nieuwe bedrijfsregels die afleidbaar zijn van de bedrijfsregels zoals getoond in Afbeelding 3.
Bijvoorbeeld voor het bepalen van alle telfouten in sectie 8 van het aangiftebiljet dienen de bedrijfsregels, in pseudocode, te worden herschreven zoals in Afbeelding 4a.

```
als totaalReisaftrek <>
    som(openbaarVervoer.reisaftrek) +
overigeReisaftrek
dan telfout('totaalReisaftrek')

als brutoReisEnFietsaftrek <>
    min(maximaleReisEnFietsaftrek,
totaalReisaftrek + fietsaftrek)
dan telfout('brutoReisEnFietsaftrek')

als nettoReisEnFietsaftrek <>
max(minimaleReisEnFietsaftrek,
    brutoReisEnFietsaftrek -
vergoedingenReisEnFietsaftrek)
dan telfout('nettoReisEnFietsaftrek')
```

Afbeelding 4a. Bedrijfsregels voor het opsporen van telfouten.

De term `telfout` is een functie die de gemaakte fout vastlegt. Wil men naast de constatering van een telfout ook de grootte vastleggen, dan kan dit door de bedrijfsregels te wijzigen zoals in Afbeelding 4b.

```
als totaalReisaftrek <>
    som(openbaarVervoer.reisaftrek) +
overigeReisaftrek
dan telfout('totaalReisaftrek', totaalReisaftrek -
    som(openbaarVervoer.reisaftrek) +
overigeReisaftrek)

als brutoReisEnFietsaftrek <>
    min(maximaleReisEnFietsaftrek,
totaalReisaftrek + fietsaftrek)
dan telfout('brutoReisEnFietsaftrek',
    brutoReisEnFietsaftrek -
    min(maximaleReisEnFietsaftrek,
totaalReisaftrek + fietsaftrek))

als nettoReisEnFietsaftrek <>
max(minimaleReisEnFietsaftrek,
    brutoReisEnFietsaftrek -
vergoedingenReisEnFietsaftrek)
dan telfout('nettoReisEnFietsaftrek',
nettoReisEnFietsaftrek -
    max(minimaleReisEnFietsaftrek,
    brutoReisEnFietsaftrek -
vergoedingenReisEnFietsaftrek))
```

Afbeelding 4b. Bedrijfsregels voor het opsporen van telfouten en grootte.

Andere bedrijfsregels die in deze context belangrijk zijn kunnen het beste worden verduidelijkt met een voorbeeld van een 'echte' aangifte zoals weergegeven in Afbeelding 5a.

```
openbaarVervoer[1].reisaftrek       := 1950
openbaarVervoer[2].reisaftrek       :=  550
totaalReisaftrek                    := 2400
vergoedingenReisEnFietsaftrek       := 1250
nettoReisEnFietsaftrek              := 1050
```

Afbeelding 5a. Verstrekte aangiftegegevens.

Onder de verstrekte aangiftegegevens verstaat men, de waarden die via een papieren aangiftebiljet, dan wel een aangiftediskette aan de Belastingdienst zijn verstrekt.
Om te laten zien wat er niet juist is met deze aangifte wordt de aangifte eerst aangevuld, d.w.z., de ontbrekende aangiftegegevens worden berekend. Onder ontbrekende aangiftegegevens bedoelt men de waarden, die berekend kunnen worden zonder te letten op de tel- en overnamefouten, welke op een papieren aangifte mogelijk zijn. De dan ontstane aangiftegegevens worden getoond in Afbeelding 5b.

```
openbaarVervoer[1].reisaftrek       := 1950
openbaarVervoer[2].reisaftrek       :=  550
totaalReisaftrek                    := 2400
brutoReisEnFietsaftrek              := 1659
vergoedingenReisEnFietsaftrek       := 1250
nettoReisEnFietsaftrek              := 1050
```

Afbeelding 5b. Aangevulde aangiftegegevens.

De volgende stap is de aangiftegegevens te corrigeren volgens de bedrijfsregels uit Afbeelding 3. De aangiftegegevens volgens deze berekening zijn te zien in Afbeelding 5c.

```
openbaarVervoer[1].reisaftrek       := 1950
openbaarVervoer[2].reisaftrek       :=  550
totaalReisaftrek                    := 2500
brutoReisEnFietsaftrek              := 1659
vergoedingenReisEnFietsaftrek       := 1250
nettoReisEnFietsaftrek              :=  409
```

Afbeelding5c. Gecorrigeerde aangiftegegevens.

Onder gecorrigeerde aangiftegegevens verstaat men de waarden die paarsgewijs verschillen van de aangevulde aangiftegegevens.

Afbeelding 6 laat de verschillende aangiftegegevens nogmaals zien in tabelvorm.

naam	verstrekt	aangevuld	gecorrigeerd
openbaarVervoer[1].reisaftrek	1950	1950	1950
openbaarVervoer[2].reisaftrek	550	550	550
totaalReisaftrek	2400	2400	2500
brutoReisEnFietsaftrek		1659	1659
vergoedingenReisEnFietsaftrek	1250	1250	1250
nettoReisEnFietsaftrek	1050	1050	409

Afbeelding 6. Aangiftegegevens in verschillende fasen.

De bedrijfsregel voor het aanvullen van de aangiftegegevens laten zich beschrijven als in Afbeelding 7a.

```
als niet isIngevuld(totaalReisaftrek)
en  (isIngevuld(openbaarVervoer.reisaftrek)
of  isIngevuld(overigeReisaftrek))
dan totaalReisaftrek :=
    som(openbaarVervoer.reisaftrek) +
overigeReisaftrek

als niet isIngevuld(brutoReisEnFietsaftrek)
en  (isIngevuld(totaalReisaftrek)
of  isIngevuld(fietsaftrek))
dan brutoReisEnFietsaftrek :=
    min(maximaleReisEnFietsaftrek,
totaalReisaftrek + fietsaftrek)

als niet isIngevuld(nettoReisEnFietsaftrek)
en  (isIngevuld(brutoReisEnFietsaftrek)
of  isIngevuld(vergoedingenReisEnFietsaftrek))
dan nettoReisEnFietsaftrek :=
    max(minimaleReisEnFietsaftrek,
    brutoReisEnFietsaftrek -
vergoedingenReisEnFietsaftrek)
```

Afbeelding 7a. Bedrijfsregels voor het aanvullen.

De term `isIngevuld` is een functie welke de waarde 'waar' retourneert indien de variabele of variabelen van een waarde zijn voorzien en anders de waarde 'niet waar'. De berekening dient alleen te worden uitgevoerd indien het te berekenen veld niet is ingevuld maar wel één of meer van zijn bestanddelen aanwezig is. De bedrijfsregels voor het corrigeren zijn bijna geheel identiek aan die van het aanvullen, behalve dat nu de berekening altijd dient te worden uitgevoerd als één of meer van de bestanddelen aanwezig zijn.

Deze bedrijfsregels worden weergegeven in Afbeelding 7b.

```
als isIngevuld(openbaarVervoer.reisaftrek)
of  isIngevuld(overigeReisaftrek)
dan totaalReisaftrek :=
    som(openbaarVervoer.reisaftrek) +
overigeReisaftrek

als isIngevuld(totaalReisaftrek)
of  isIngevuld(fietsaftrek)
dan brutoReisEnFietsaftrek :=
    min(maximaleReisEnFietsaftrek,
totaalReisaftrek + fietsaftrek)

als isIngevuld(brutoReisEnFietsaftrek)
of  isIngevuld(vergoedingenReisEnFietsaftrek)
dan nettoReisEnFietsaftrek :=
    max(minimaleReisEnFietsaftrek,
    brutoReisEnFietsaftrek -
vergoedingenReisEnFietsaftrek)
```

Afbeelding 7b. Bedrijfsregels voor het corrigeren.

Alle tot nu toe besproken bedrijfsregels zijn beschreven in een pseudocode die voor de meeste lezers als leesbaar en begrijpbaar zal worden ervaren.
Totaal anders is dit voor een rule-engine. Hier is een exacte beschrijving van de syntaxis nodig die vaststelt of een bedrijfsregel is toegestaan in de gehanteerde beschrijvingstaal.
Ook dient de gehanteerde semantiek volledig te zijn beschreven, zodat de opsteller van de bedrijfsregels kan voorspellen wat de uitkomst zal zijn bij evaluatie van een gegeven aangifte.
Om dit te illustreren wordt de `brutoReisEnFietsaftrek` berekend met de aangiftegegevens uit Afbeelding 5a. Hierbij wordt de bedrijfsregel:

```
brutoReisEnFietsaftrek :
               = min(maximaleReisEnFietsaftrek
               , totaalReisaftrek + fietsaftrek)
```

uit Afbeelding 3 gehanteerd. Substitueren we de aangiftegegevens in deze bedrijfsregel dan krijgen we:

```
brutoReisEnFietsaftrek :
               = min(1659, 2400 + onbekend)
```

De keuze van de gehanteerde semantiek bepaalt nu of de term `min(1659, 2400 + onbekend)` evalueerbaar is of niet. Met

andere woorden, kan de term) `2400 + onbekend)` worden berekend en wat is het resultaat? In de rule-engine van IBS is voor dit soort termen gekozen voor `automatische nul substitutie`, met andere woorden de term wordt `2400 + 0`. De waarde voor de variabele `brutoReisEnFietsaftrek` wordt hierdoor gelijk aan 1659.
In de rule-engine van ABS – Blaze Advisor – is het resultaat van `2400 + onbekend` gelijk aan `onbekend`. De variabele `brutoReisEnFietsaftrek` krijgt toch de waarde 1659 door de eigen gedefinieerde functie `min`, zie Afbeelding 10. Beide keuzes zijn valide en hebben ieder hun speciale voor- en nadelen, die in het vervolg van het artikel worden verduidelijkt.
Een derde mogelijke uitkomst is een runtime foutmelding omdat de optelling `2400 + onbekend` niet is gedefinieerd.

Aan de hand van hetgeen tot nu toe besproken is, wordt uitgewerkt hoe de bedrijfsregels in Blaze Advisor kunnen worden gerealiseerd.
De beschrijvingstaal die hiervoor wordt gebruikt noemt men SRL en staat voor Structured Rule Language. Bedrijfsregels in Blaze Advisor worden gedefinieerd op een zogenaamd 'object model'. Dit object model is een 'in-memory' structuur die middels data ontsluiting, bijvoorbeeld uit een relationele database, van data wordt voorzien. In de Afbeeldingen 8a tot en met 8d wordt het object model voor het behandelde voorbeeld getoond.

```
a constantenObject is a object with
{
   a minimaleReisEnFietsaftrek: a real,
   a maximaleReisEnFietsaftrek: a real
}
```

Afbeelding 8a. Object model voor de gebruikte constanten.

```
a periodeVraagObject is a object with
{
   a van:            a date,
   a totEnMet:       a date
}
```

Afbeelding 8b. Object model voor het type periodeVraag.

```
a openbaarVervoerObject is a object with
{
  a plaatsVanUwWerk:      a string,
  a enkeleReisAfstandInKm:a integer,
  a periode:              some periodeVraagObject,
  a reisaftrek:           a real,
  a dagenPerWeek:         a integer
}
```

Afbeelding 8c. Object model voor het type openbaarVervoer.

```
a aangifteObject is a object with
{
 //aangifte variabelen
 a openbaarVervoer: some association from integer
 to openbaarVervoerObject,
   a overigeReisaftrek:                a real,
   a totaalReisaftrek:                 a real,
   a fietsaftrek:                      a real,
   a brutoReisEnFietsaftrek:           a real,
   a vergoedingenReisEnFietsaftrek:    a real,
   a nettoReisEnFietsaftrek:           a real,
   //hulp variabelen voor de nul
   substitutie
   a valOverigeReisaftrek:             a real,
   a valTotaalReisaftrek:              a real,
   a valFietsaftrek:                   a real,
   a valBrutoReisEnFietsaftrek:        a real,
   a valVergoedingenReisEnFietsaftrek: a real,
 //hulp variabele voor het berekenen
 van een som
 a somReisaftrek:                      a real
}
```

Afbeelding 8d. Object model voor sectie 8 van het O-biljet.

```
function calcNettoReisEnFietsaftrek for
{
   aangifte: a aangifteObject, constanten: a
   constantenObject
}
is
{
   aangifte.nettoReisEnFietsaftrek =
    max(constanten.minimaleReisEnFietsaftrek,
     aangifte.valBrutoReisEnFietsaftrek –
     aangifte.valVergoedingenReisEnFietsaftrek).
}

function calcBrutoReisEnFietsaftrek for
{
   aangifte: a aangifteObject, constanten: a
   constantenObject
}
is
{
   aangifte.brutoReisEnFietsaftrek =
    min(constanten.maximaleReisEnFietsaftrek,
     aangifte.valTotaalReisaftrek +
aangifte.valFietsaftrek).
}
```

```
function calcTotaalReisAftrek for
{
   aangifte: a aangifteObject
}
is
{
   aangifte.totaalReisaftrek =
      aangifte.somReisaftrek +
aangifte.valOverigeReisaftrek.
}
```

Afbeelding 9. Bedrijfsregels voor sectie 8 van het O-biljet.

```
function max for
{
   argument1: a real, argument2: a real
}
returning a real is
{
   if       (argument1 is unknown)
            then return max(0, argument2)
   else if  (argument2 is unknown)
            then return max(argument1, 0)
   else if  (argument1 > argument2)
            then return argument1
   else     return argument2.
}

function min for
{
   argument1: a real, argument2: a real
}
returning a real is
{
   if       (argument1 is unknown)
            then return min(0, argument2)

  else if   (argument2 is unknown)
            then return min(argument1, 0)
   else if  (argument1 < argument2)
            then return argument1
   else     return argument2.
}
```

Afbeelding 10. De functies `max` en `min`.

De bedrijfsregels uit Afbeelding 3 laten zich in Blaze Advisor beschrijven als functies zoals weergegeven in Afbeelding 9. De functies `max` en `min` die in de bedrijfsregels worden gebruikt zijn gedefiniëerd zoals weergegeven in Afbeelding 10.

Opvallend bij de bedrijfsregels getoond in Afbeelding 9, is het feit dat het expressie deel geheel wordt uitgedrukt middels de hulpvariabelen. Dit is nodig om te voorkomen dat de waarde `unknown` ontstaat bij het evalueren van de bedrijfsregels (zie de eerder gemaakte opmerking over semantiek). Deze hulpvariabelen worden van een waarde voorzien met een speciaal soort regel, die men in Blaze Advisor 'event rules' noemt. Deze 'event rules' voor de hulp variabelen wordt getoond in Afbeelding 11, zij worden éénmalig uitgevoerd en wel alleen als men de hulpvariabele gebruikt tijdens de evaluatie van een bedrijfsregel.

```
event rule valVergoedingenReisEnFietsaftrek is
whenever a
aangifteObject.valVergoedingenReisEnFietsaftrek is
needed
do val VergoedingenReisEnFietsaftrek =
       val(vergoedingenReisEnFietsaftrek).

event rule valBrutoReisEnFietsaftrek is
whenever a
aangifteObject.valBrutoReisEnFietsaftrek is needed
do valBrutoReisEnFietsaftrek =
      val(brutoReisEnFietsaftrek).

event rule valFietsaftrek is
whenever a aangifteObject.valFietsaftrek is needed
do valFietsaftrek = val(fietsaftrek).

event rule valTotaalReisaftrek is
whenever a aangifteObject.valTotaalReisaftrek is
needed
do valTotaalReisaftrek = val(totaalReisaftrek).

event rule valOverigeReisaftrek is
whenever a aangifteObject.valOverigeReisaftrek is
needed
do valOverigeReisaftrek = val(overigeReisaftrek).

event rule somReisAftrek is
whenever a aangifteObject.somReisaftrek is needed
do somReisaftrek = som(openbaarVervoer).
```

Afbeelding 11. Event rules.

De functies `val` en `som` die in de 'event rules' worden gebruikt zijn gedefiniëerd zoals weergegeven in Afbeelding 12.

In de functie `som` doorloopt de variabele `it`, die gebruikt wordt in de for-each lus, iedere waarde uit de associatie `openbaarVervoer`.

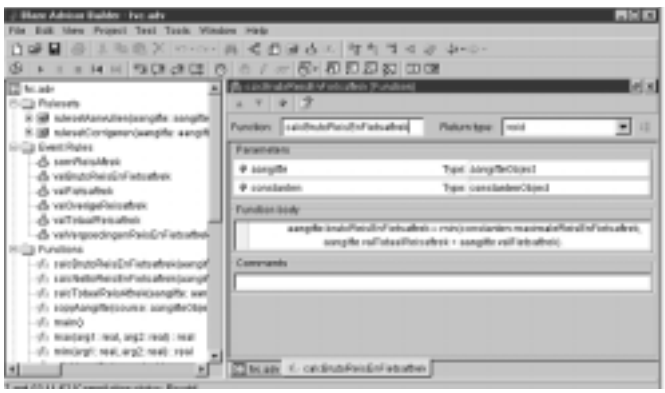
```
function val for
{
   argument: a real
}
returning a real is
{
   if (argument is unknown) then return 0 else
return argument.
}

function som for
{
   openbaarVervoer : a association from integer
                       to openbaarVervoerObject
}
returning a real is
{
   som is a real initially 0.

   if openbaarVervoer is known then
   {
      for each openbaarVervoerObject in
openbaarVervoer do
      {
         som += it.reisaftrek.
      }
   }
   return som.
}
```

Afbeelding 12. De functies `val` en `som`.

Afbeelding 13 toont de user interface van Blaze Advisor.
De regels voor het aanvullen van de aangiftegevens zoals weergegeven in Afbeelding 7a, worden in SRL beschreven zoals weergegeven in Afbeelding 14a. Bedrijfsregels die een verband hebben met elkaar worden gegroepeerd in een structuur – in Blaze Advisor – een 'ruleset' genoemd.



Afbeelding 13. User interface van Blaze Advisor.

```
ruleset rulesetAanvullen for
{
   aangifte: a aangifteObject, constanten: a
constantenObject
}
is
{
   rule nettoReisEnFietsaftrek is
   if   aangifte.nettoReisEnFietsaftrek is
unknown
   and  (aangifte.brutoReisEnFietsaftrek is known
   or   aangifte.vergoedingenReisEnFietsaftrek is
known)
   then calcNettoReisEnFietsaftrek(aangifte, con-
stanten).

   rule brutoReisEnFietsaftrek is
   if   aangifte.brutoReisEnFietsaftrek is
unknown
   and  (aangifte.totaalReisaftrek is known
   or   aangifte.fietsaftrek is known)
   then calcBrutoReisEnFietsaftrek(aangifte, con-
stanten).

   rule totaalReisaftrek is
   if   aangifte.totaalReisaftrek is unknown
   and  (aangifte.openbaarVervoer is known
   or   aangifte.overigeReisaftrek is known)
   then calcTotaalReisAftrek(aangifte).
}
```

Afbeelding 14a. Bedrijfsregels voor het aanvullen in SRL.

```
ruleset rulesetCorrigeren for
{
   aangifte: a aangifteObject, constanten: a
constantenObject
}
is
{
   rule nettoReisEnFietsaftrek is
   if   aangifte.brutoReisEnFietsaftrek is known
   or   aangifte.vergoedingenReisEnFietsaftrek is
known
   then calcNettoReisEnFietsaftrek(aangifte, con-
stanten).

   rule brutoReisEnFietsaftrek is
   if   aangifte.totaalReisaftrek is known
   or   aangifte.fietsaftrek is known
   then calcBrutoReisEnFietsaftrek(aangifte, con-
stanten).

   rule totaalReisaftrek is
   if   aangifte.openbaarVervoer is known
   or   aangifte.overigeReisaftrek is known
   then calcTotaalReisAftrek(aangifte).
}
```

Afbeelding 14b. Bedrijfsregels voor het corrigeren in SRL.

De regels voor het corrigeren van de aangiftegevens, weergegeven in Afbeelding 7b, worden getoond in Afbeelding 14b.

De volgorde waarin de bedrijfsregels uitgevoerd in een 'ruleset', wordt bepaald door het RETE algoritme[1] en niet door de volgorde van de bedrijfsregels zoals gespecificeerd in de 'ruleset'.
Het RETE algoritme zorgt ervoor dat eerst de waarde voor het veld 8a `totaalReisaftrek`, vervolgens het veld 8c `brutoReisEnFietsaftrek` en tot slot het veld 8d `nettoReisEnFietsaftrek` wordt berekend ('forward-chaining').

Samenvatting en conclusie

De Afbeeldingen 3, 4a, 4b, 7a en 7b tonen vijf verschillende typeringen van bedrijfsregels, waarvan de laatste vier afleidbaar zijn uit de bedrijfsregels in Afbeelding 3.
Wanneer men ervoor kiest zelf een rule-engine te realiseren kan men de evaluatie-functie voorzien van een indicator. Die geeft aan hoe de bedrijfsregels dienen uitgevoerd te worden, met als voordeel dat alleen de regels zoals in Afbeelding 3 moeten worden onderhouden.
Is de evaluatie-functie een gegeven en kan er niet op de werking ervan worden gestuurd middels een indicator, dan moeten de verschillende typeringen van bedrijfsregels worden onderhouden.
Een alternatieve aanpak is het investeren in een vertaalprogramma die als invoer de pseudocode uit de Afbeeldingen 2 en 3 heeft en deze vertaalt naar SRL zoals getoond in de Afbeeldingen 8a, 8b, 8c, 8d, 9, 11, 14a en 14b.

Johan van der Graaf is als consultant werkzaam bij Cap Gemini Ernst & Young, johan.vander.graaf@cgey.nl.

Literatuur

1 Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem, Charles Forgy, Artificial Intelligence, 19, p. 17-37, 1982