

Functioneel Specificeren met Hatley Pirbhai methodiek

INLEIDING

Deze handout geeft een beknopte beschrijving van de Hatley/Pirbhai methodiek, zoals deze door Eekels wordt gebruikt voor het opstellen van een Functionele Specificatie.

STRUCTUUR

Met de Hatley/Pirbhai methodiek kunnen specificaties van (real time) systemen worden gemodelleerd in een zogenaamd *Requirements Model* (RM). Dit model gebruikt een aantal diagrammen waarin de specificatie wordt vastgelegd. De diagrammen zijn middels een bepaalde hiërarchie met elkaar gekoppeld.

Boven in de hiërarchie staat het zogenaamde *Context Diagram*. In dit diagram wordt aangegeven in welke context de te specificeren functionaliteit moet worden gezien. Dit wordt aangegeven met terminators (zie paragraaf Symboliek). In het context diagram is één centraal proces gedefinieerd. De functionaliteit van dit proces wordt in onderliggende diagrammen gespecificeerd.

De detail specificatie wordt vastgelegd in *Data Flow Diagrams* (DFD) en *Control Flow Diagrams* (CFD). Deze diagrammen tonen (deel)processen met de data- en controlflow tussen de diagrammen.

De activering en de-activering van processen op een C/DFD vindt plaats vanuit zogenaamde State Diagrams (STD). Deze diagrammen kunnen het best worden vergeleken met stap/transitie diagrammen. States worden met rechthoeken aangeduid, de transities met een pijl.

In een transitie staat met *I*: aangegeven wanneer de transitie wordt gemaakt. Alle processen die bij deze overgang moeten worden geactiveerd staan met *A*: vermeld. PER DEFINITIE WORDEN ALLE ANDERE PROCESSEN BEHOREND BIJ DEZE C/DFD INACTIEF. Aan uitgangsvariabelen kan met *O*: een waarde toegekend worden.

Indien een proces dermate elementair wordt en de relatie tussen in- en uitgangen eenvoudig kan worden omschreven, wordt een *Process Specification* (PSPEC) aangemaakt. Aan de formuleringen binnen een PSPEC zijn geen eisen gesteld. In de meeste gevallen zal een goed leesbare pseudo programmeertaal worden gehanteerd.

Alle control- en dataflow signalen die in het Requirements Model worden gebruikt, worden vastgelegd in de Requirements Dictionary. Deze lijst geeft een overzicht en beschrijving van de signalen en ook de hiërarchie ervan.

SYMBOLIEK

Er worden in het context diagram, CFD en DFD de volgende tekensymbolen gehanteerd:




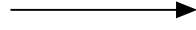
Cirkel met
enkelvoudige rand

Symboliseert een (complex) proces. Dit proces is opgebouwd uit meerdere sub processen en primitieven.



Cirkel met dubbele
rand

Symboliseert een primitief proces. De functie van dit proces is dermate elementair dat het kan worden beschreven met een zogenaamde PSPEC (Process specification).

	Rechthoek	Symboliseert een eindproces (terminator). Hiermee wordt de grens van het te specificeren gedeelte afgebakend.
	Verticale streep	Control bar. Geeft de koppeling weer van een C/DFD en een state-diagram of decision table. In het state-diagram wordt bepaald wanneer welke processen uit de C/DFD actief zijn.
	Dubbele horizontale lijn	Symboliseert een opslag van gegevens.
	Ononderbroken lijn met pijl	Symboliseert een dataflow waarmee informatie tussen processen wordt uitgewisseld. Dit kan zowel een enkel- als meervoudig signaal zijn.
	Onderbroken lijn met pijl	Symboliseert een controlflow waarmee stuursignalen tussen processen worden uitgewisseld. Dit kan zowel een enkel als meervoudig signaal zijn.

NAAMGEVING FLOWSIGNALLEN

Zoals aangegeven kan een getekend flowsignaal zowel enkel- als meervoudig zijn. Vergelijk dit met signaaladers in een elektrische kabel. Een kabel bestaat uit één of meerdere sub-kabels die zelf weer uit signaal aders bestaan. Om dit in het ontwerp duidelijk zichtbaar te maken, is daarom de volgende afspraak gemaakt:

Meervoudige signalen (kabel) zijn in hoofdletters, enkelvoudige signalen (aders) zijn in kleine letters.

Om aan te geven tot welke kabel een ader behoort, is afgesproken dat de kabelnaam terugkomt in de naam van de ader. Hierdoor ontstaat een duidelijke hiërarchie. Afspraak is daarom dat de naam van een enkelvoudig signaal begint met de naam van de kabel(s) waartoe deze behoort.

Om duidelijke scheiding te maken in de (deel)namen worden scheidingstekens gebruikt.

Als scheidingsteken voor een dataflow signaal ader wordt een slash (/) gebruikt.

Als scheidingsteken voor een controlflow signaal ader wordt een punt (.) gebruikt.

Kabelsignalen eindigen altijd met een scheidingsteken (punt of slash)

In de naam van de signaalader zelf komen in principe geen spaties voor. Als scheiding wordt een underscore (_) gehanteerd. In de diagrammen (DFD/CFD/STD) worden de _ tekens soms vervangen door een spatie. Dit om de leesbaarheid te vergroten.

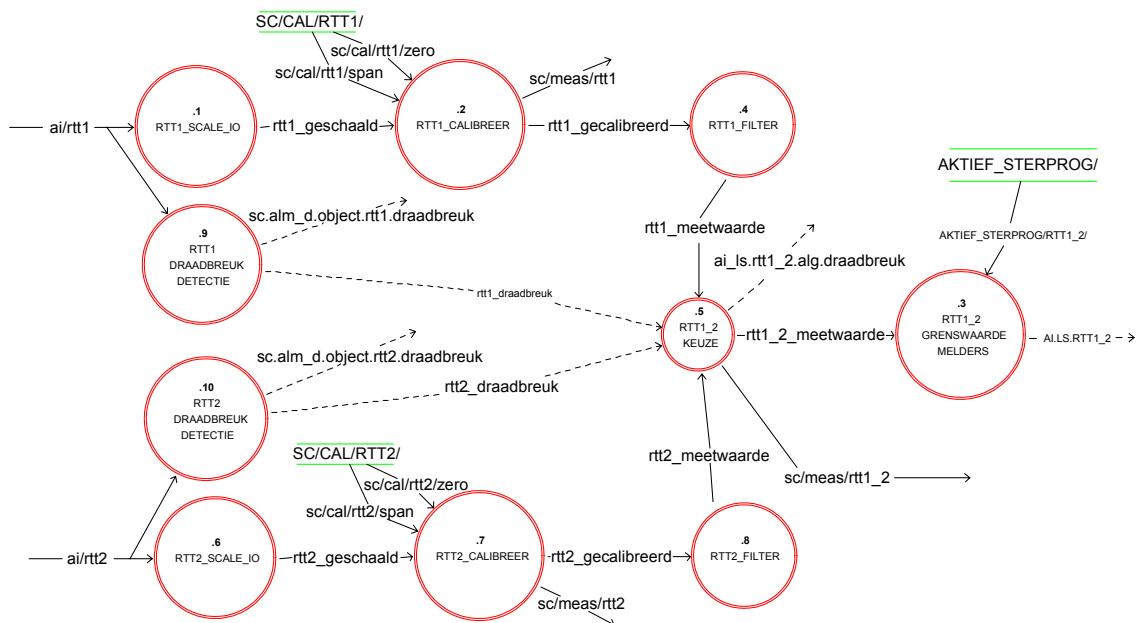
Opmerking: er is op bovenstaand afspraken een uitzondering: bepaalde interne enkelvoudige signalen worden omwille van leesbaarheid in hoofdletters weergegeven. Omdat deze signalen niet eindigen met punt of slash, kan worden afgeleid dat het geen kabelsignaal betreft.

Voorbeelden:

SC.ALM_D.	Signaalkabel met stuursignalen. In dit geval met alarmen die naar SCADA worden gestuurd.
SC/MEAS/ sc/meas/r1	Signaalkabel met meetdata. In dit geval met meetsignalen naar SCADA. Signaalader met de enkelvoudige meetwaarde.
STER1_VULNIVO_BEREIKT	Intern enkelvoudig stuursignaal. Omwille van leesbaarheid in hoofdletters weergegeven.

Voorbeeld 1. DFD - KAMER_TEMP_METING

Het meten van een kamertemperatuur met twee gescheiden meet opnemers. Slechts één meting wordt uiteindelijk in het proces gebruikt.



Voorbeeld 2. RTT1_2 KEUZE Process Specification

```
RECOGNIZE: rtt2_draadbreuk
RECOGNIZE: rtt1_draadbreuk
INPUT: rtt1_meetwaarde
INPUT: rtt2_meetwaarde
```

```
GENERATE: ai_ls.rtt1_2.alg.draadbreuk
OUTPUT: rtt1_2_meetwaarde
OUTPUT: sc/meas/rtt1_2
```

```
* De maximum temperatuur meetwaarde is bepalend *
rtt1_2_meetwaarde = MAX( rtt1_meetwaarde, rtt2_meetwaarde )
```

```
* Afhandeling draadbreuk *
ALS rtt1_draadbreuk
DAN rtt1_2_meetwaarde := rtt2_meetwaarde
```

```
ALS rtt2_draadbreuk
DAN rtt1_2_meetwaarde := rtt1_meetwaarde
```

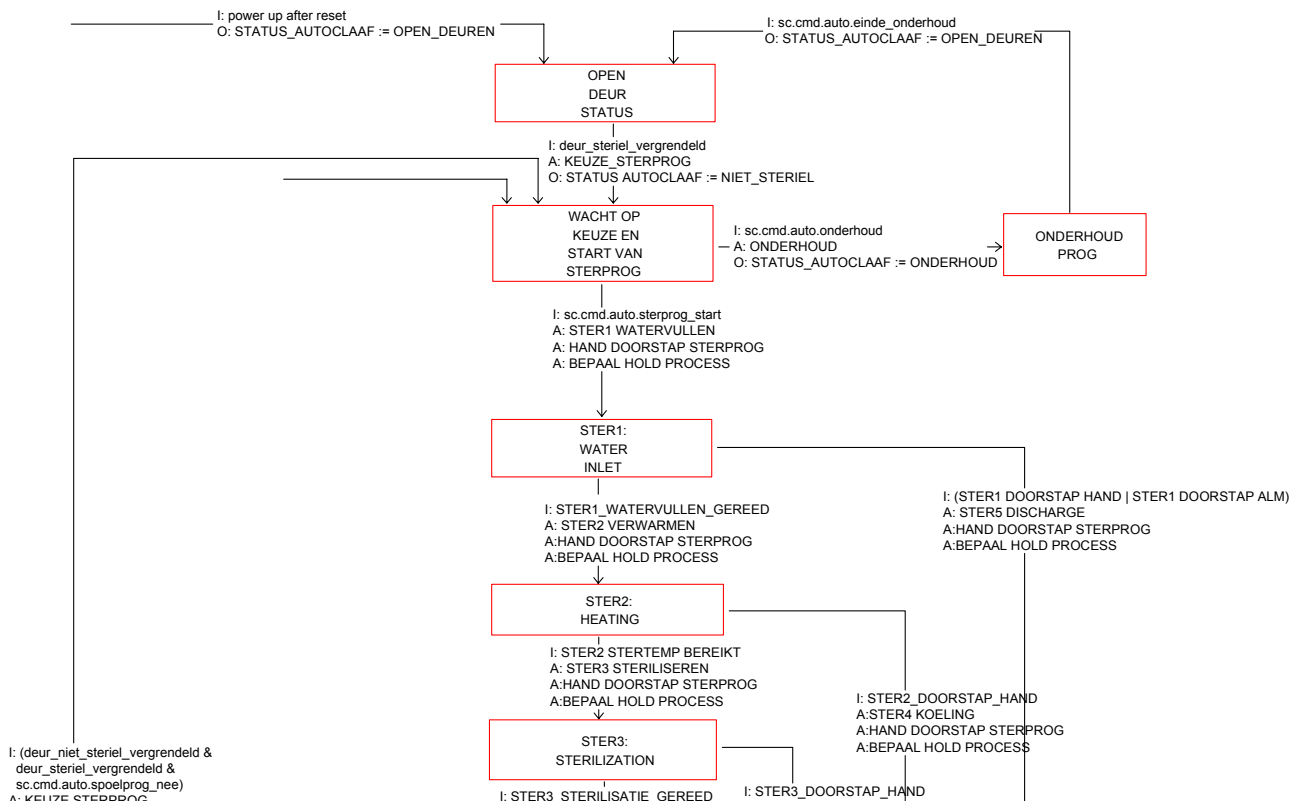
```
ALS rtt1_draadbreuk EN rtt2_draadbreuk
DAN ai_ls.rtt1_2.alg.draadbreuk := TRUE
    rtt1_2_meetwaarde := 0.0
```

```
* actuele meetwaarde naar SCADA *
sc/meas/rtt1_2 := rtt1_2_meetwaarde
```

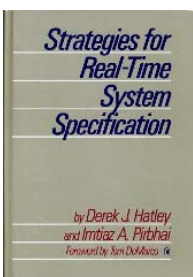


Voorbeeld 3. STD - STER_AUTO.c

Gedeeltelijk State Diagram voor een Automaatbesturing sterilisatiecyclus:



Voor een uitgebreide uiteenzetting van de methodiek wordt verwezen naar:



Strategies for Real-Time System Specification

by Derek J. Hatley, Imtiaz A. Pirbhaj, Tom DeMarco

Hardcover: 386 pages ; Dimensions (in inches): 1.00 x 10.50 x 7.25

Publisher: Dorset House; (January 1988)

ISBN: 0932633110