

Parameter-analyse module ten behoeve van het LIFE-instrumentarium Schelde estuarium

Detailontwerp

J. Ansink, P.J.G. Brummelhuis, M. Hogeweg en M.R.L. Ouboter



waterloopkundig laboratorium | WL

Inhoudsopgave

Samenvatting	ii
1 Inleiding	1
2 Opzet van de module	2
2.1 Uitgangspunten	2
2.2 Globale structuur van de module	2
2.3 Structuur Diagram	3
3 Data-structuren	4
3.1 Constanten ter specificatie van de parameter-analyse	4
3.2 De stoffen	4
3.3 De parameters	4
3.4 De 'vase'	5
3.5 De simulatie-resultaten	5
3.6 De hyper-kubussen voor een min/max representatie van de on- zekerheid	5
3.7 De metingen	5
4 Parameter-analyse module, main program	6
5 Level 1 subroutines	7
5.1 ReadDataBase	7
5.2 AnalyzeParameters	8
6 Level 2 subroutines	9
6.1 ReadModuleFile#1	9
6.2 ScanDataBase	10
6.3 Initialize	11
6.4 Update	12
6.5 DetermineHyperCube	13
6.6 Report	14
7 Level 3 subroutines	15
7.1 ReadModuleFile#2	15
7.2 ReadMeasurementsFile	17
7.3 InitializeVase	18
7.4 DetermineNewCombination	20
7.5 DetermineGoodnessOfFit	21
7.6 UpdateVase	23
7.7 Simulate	24
8 Level 4 subroutines	25
8.1 IntervalRandom	25
8.2 WriteParameterSection	26
8.3 ReadConcentrations	27
8.4 ReadProcessFluxes	27
9 Input-file voor parameter-analyse module	28
9.1 Inleiding	28
9.2 Format van de input-file	28
10 JSPOST invoer formaat	30
10.1 Inleiding	30
10.2 Het stuurbestand	30
10.3 Het data bestand	31
11 Referenties	32

Samenvatting

Dit rapport bevat een beschrijving van het technisch detailontwerp van een parameter-analyse module ter uitbreiding van het LIFE-instrumentarium. Het doel van deze module is de DELWAQ4-gebruiker in staat te stellen diverse proces-parameters en schalingsfactoren voor belastingen te bepalen zodat, bij gegeven waterbeweging, slibtransport en procesformuleringen, een bepaalde set veldmetingen zo goed mogelijk gereproduceerd wordt.

Dit technisch detailontwerp is er primair op gericht in de eerste versie de basis-functionaliteit van de parameter-analyse module, zoals vastgelegd in het functioneel ontwerp [1], te realiseren en waar mogelijk, tijdens de ontwerpfase reeds rekening te houden met eventuele functionele uitbreidingen.

De parameter-analyse module zal worden geïmplementeerd in FORTRAN77 en zal onder MSDOS verwerkt moeten worden. In voorkomende gevallen is gebruik gemaakt van bestaande DELWAQ programmatuur, zoals voor het lezen van de aan DELWAQ gerelateerde files en van de data-bases. Hierbij wordt zoveel mogelijk gebruik gemaakt van ODS-routines.

1 Inleiding

De ontwikkeling en implementatie van een parameter-analyse module als uitbreiding van het LIFE-instrumentarium heeft ten doel de DELWAQ4-gebruiker in staat te stellen diverse procesparameters en schalingsfactoren voor belastingen te bepalen zodat, bij gegeven waterbeweging, slibtransport en procesformuleringen, een bepaalde set veldmetingen zo goed mogelijk gereproduceerd wordt.

Met deze functionaliteit wordt beoogd bij te dragen aan het systeem-inzicht en de gebruiker een middel te bieden om de onzekerheid in de belastings- en proces-fluxen vast te stellen op grond van specifieke systeemkennis. Op basis hiervan kan vervolgens de onzekerheid worden bepaald in concentraties als resultaat van scenario-berekeningen.

In oktober 1994 is door RIKZ, in de persoon van drs. B.J. Kater (brief RIKZ/AB-94.60119) in het kader van het project SCHOON, WL verzocht een offerte uit te brengen voor het ontwikkelen van een prototype van een dergelijke module. Op basis van deze offerte heeft RIKZ WL de opdracht verleend voor het maken van een functioneel ontwerp (fase 1a, opdrachtbonnr. 42943099, dd. 27 oktober 1994) en een technisch detail ontwerp (fase 1b, opdrachtbonnr. 42943398, dd. 22 november 1994). Daarbij is tevens overeengekomen dat RIKZ en WL dit project gezamenlijk zullen financieren.

Dit rapport, eindproduct van fase 1b, bevat een beschrijving van het technisch detailontwerp van een prototype van een parameter-analyse module voor DELWAQ. Dit technisch ontwerp is afgeleid op basis van het functioneel ontwerp [1] en zal als uitgangspunt dienen voor de implementatie van de parameter-analyse module. Oplevering van het eindproduct is voorzien voor 1 mei 1995.

Fase 1b is uitgevoerd door ing. J. Ansink, dr. ir. P.G.J ten Brummelhuis, drs. M. Hogeweg en drs. M.R.L. Ouboter, met medewerking van drs. B.J. Kater (RIKZ); de kwaliteitscontrole is verzorgd door ir. L. Postma.

2 Opzet van de module

2.1 Uitgangspunten

Bij het opstellen van dit technisch detailontwerp zijn de volgende uitgangspunten gehanteerd:

- De parameter-analyse module zal worden geïmplementeerd in FORTRAN77 en is bedoeld om onder MSDOS verwerkt te worden.
- De parameter-analyse module moet uitgaan van de specificaties van DELWAQ-versie 4 met betrekking tot de aansturing en de output.
- De resultaten worden in een zodanig format weggeschreven dat deze met behulp van JSPOST verwerkt kunnen worden.
- Het moet mogelijk zijn om de parameter-analyse module op eenvoudige wijze uit te breiden met andere opties voor
 - ▷ Het updaten van de 'vase' en
 - ▷ Het representeren van de onzekerheid in de parameters, de concentraties en de procesfluxen.
- De eerste versie van de parameter-analyse module wordt niet voorzien van een user interface, de aansturing wordt via een ASCII-file geregeld.
- Voor het lezen en schrijven van de aan DELWAQ gerelateerde files en het lezen uit data-bases wordt zoveel mogelijk gebruik gemaakt van ODS-routines¹.

2.2 Globale structuur van de module

Binnen de parameter-analyse module zijn 3 delen te onderscheiden:

- Initialisatie van de parameter-analyse:
 - ▷ Selecteren van alle relevante metingen uit de opgegeven data-base,
 - ▷ Samenstellen van een initiële 'vase'.
- Het updaten van de 'vase' door een (van tevoren opgegeven) aantal DELWAQ-simulaties uit te voeren met telkens andere parameter-combinaties, de bijbehorende Goodness-of-Fit-waarde te bepalen en die parameter-combinaties te behouden waarmee de veldmetingen zo goed mogelijk gereproduceerd kunnen worden. Hierdoor wordt de kwaliteit van de 'vase' stap voor stap verhoogd.
- Nadat de 'vase' zijn uiteindelijke inhoud heeft gekregen wordt de variatie in zowel de parameter-ruimte als de concentratie- en de procesflux-ruimte bepaald en geëxporteerd naar programmatuur om deze resultaten grafisch weer te geven.

¹ODS staat voor Open Data Structuur

2.3 Structuur Diagram

De bovenstaande beschrijving van de parameter-analyse heeft geresulteerd in het volgende structuur diagram:

```
Main
  ReadDataBase
    ReadModuleFile#1
    ScanDataBase
  AnalyzeParameters
    Initialize
      ReadModuleFile#2
      ReadMeasurementsFile
      InitializeVase
        IntervalRandom
        Simulate
          WriteParameterSection
          ReadConcentrations
          ReadProcessFluxes
          DetermineGoodnessOfFit
    Update
      DetermineNewCombination
      IntervalRandom
      Simulate
        WriteParameterSection
        ReadConcentrations
        ReadProcessFluxes
        DetermineGoodnessOfFit
      UpdateVase
    DetermineHyperCube
      Simulate
        WriteParameterSection
        ReadConcentrations
        ReadProcessFluxes
  Report
```

3 Data-structuren

De data-structuren die in de parameter-analyse module voorkomen zijn:

3.1 Constanten ter specificatie van de parameter-analyse

INTEGER	NOVAS	omvang van de 'vase'
INTEGER	NOPAR	aantal parameters
INTEGER	NOSUB	aantal stoffen
INTEGER	NOWIN(NOPAR)	lengte time-windows, uitgedrukt in tDelta
INTEGER	NORAN	omvang random trekking uit 'vase'
INTEGER	NOSIM	aantal DELWAQ-simulaties
INTEGER	NOPRO	aantal processen
INTEGER	NOSEG	aantal segmenten
INTEGER	tUpdte	id. voor procedure ter bepaling van nieuwe parameter combinatie 1 = UPDATE-PRICE 2 = nog niet geïmplementeerde procedure
INTEGER	tReprs	id. voor wijze van onzekerheidsrepresentatie 1 = MIN/MAX-REPRESENTATION 2 = nog niet geïmplementeerde onzekerheidsrepresentatie
INTEGER	tStore	id. voor wijze van opslag van simulatie-resultaten 1 = STORAGE-OFFLINE 2 = STORAGE-ONLINE
INTEGER	tStart	starttijd van de DELWAQ-simulatie
INTEGER	tEnd	eindtijd van de DELWAQ-simulatie
INTEGER	tDelta	uitvoer-tijdstap van DELWAQ

3.2 De stoffen

CHARACTER*(*)	sName(NOSUB)	naam van de stof
INTEGER	sNwin(NOSUB)	lengte time-window
REAL	sExp(NOSUB)	exponent
REAL	sWght(NOSUB)	weegfactor

3.3 De parameters

INTEGER	pType(NOPAR)	type van de parameter
CHARACTER*(*)	pName(NOPAR)	parameter-naam of stofnaam
CHARACTER*(*)	pLoc(NOPAR)	'DOMAIN' of naam van de lozingslocatie
REAL	pMin(NOPAR)	ondergrens parameter-waarde
REAL	pMax(NOPAR)	bovengrens parameter-waarde

3.4 De 'vase'

REAL	vPset(NOVAS,NOPAR)	de verzameling van parameter-combinaties in de 'vase'
REAL	vGoF(NOVAS)	de verzameling van GoF-waarden behorende bij parameter-combinaties in de 'vase'
INTEGER	vIGoF	index van parameter-combinatie in 'vase' met max GoF
REAL	vAvg(NOPAR)	gemiddelde over de 'vase' per parameter
REAL	vMax(NOPAR)	maximum over de 'vase' per parameter
REAL	vMin(NOPAR)	minimum over de 'vase' per parameter

3.5 De simulatie-resultaten

REAL	rConc(NOSUB,Nt,NOSEG)	concentratie per stof per uitvoer-tijdstap per segment
REAL	rFlux(NOPRO,Nt,NOSEG)	proces-flux per stof per uitvoer-tijdstap per segment
		$Nt = \text{int}[(tEnd - tStart)/tDelta]$

3.6 De hyper-kubussen voor een min/max representatie van de onzekerheid

REAL	hMinC(NOSUB,Nt,NOSEG)	minimum concentratie, per stof, plaats en tijd
REAL	hMaxC(NOSUB,Nt,NOSEG)	maximum concentratie, per stof, plaats en tijd
REAL	hAvgC(NOSUB,Nt,NOSEG)	gemiddelde concentratie, per stof, plaats en tijd
REAL	hMinF(NOPRO,Nt,NOSEG)	minimum flux, per proces, plaats en tijd
REAL	hMaxF(NOPRO,Nt,NOSEG)	maximum flux, per proces, plaats en tijd
REAL	hAvgF(NOPRO,Nt,NOSEG)	gemiddelde flux, per proces, plaats en tijd

3.7 De metingen

CHARACTER*(*)	mName(NOSUB)	stofnaam
INTEGER	mNval(NOSUB)	aantal metingen
INTEGER	mTime(MAXMET,NOSUB)	tijd in dddhhmmss
REAL	mPlace(MAXMET,NOSUB)	plaats (segmentnummer)
REAL	mVal(MAXMET,NOSUB)	concentratie

4 Parameter-analyse module, main program

In dit en de volgende hoofdstukken zijn de submodulen van de parameter-analyse module weergegeven in pseudo-code. Van elke functie is de interface met de omgeving beschreven en, waar mogelijk, een referentie naar het functioneel ontwerp.

PROGRAM: main

Purpose: Dit is hoofd-programma

Context: -

Pseudo code:

 ReadDataBase (parinp, datbas, meapst)

 AnalyzeParameters (tStore, tReprs, tUpdata, dlwinp, parinp, meapst)

 end

5 Level 1 subroutines

5.1 ReadDataBase

SUBROUTINE: ReadDataBase

Purpose: Doorzoekt de opgegeven metingen data-base op de aanwezigheid van relevante veldmetingen en schrijft deze in '*.pst'-format naar meapst.

Context: -

Description: Uit 'parinp' wordt de simulatie-periode gelezen waarna de in de data-base 'datbas' aanwezige relevante veldmetingen worden geselecteerd en naar een *.pst-file weggeschreven. In ReadMeasurementsFile worden uit deze *.pst-file die stoffen geselecteerd die in 'parinp' zijn aangegeven als de stoffen waar het Goodness-of-Fit criterium op wordt gebaseerd.

SUBROUTINE ReadDataBase (parinp, datbas)

Parameters:

CHARACTER*(*) parinp
CHARACTER*(*) datbas

Pseudo code:

ReadModuleFile#1 (parinp, tStart, tEnd)

ScanDataBase (datbas, tStart, tEnd)

5.2 AnalyzeParameters

SUBROUTINE: AnalyzeParameters

Purpose: Dit is de hoofd-functie van de parameter-analyse module

Context: De parameter-analyse module wordt aangeroepen via een batch *commando* (een *user interface* is nog geen onderdeel van de huidige functionaliteit).

Description: Na initialisatie volgt het iteratief aanpassen van de 'vase'. Indien nodig wordt met de parameter-combinaties in de Uiteindelijke 'vase' nog simulaties uitgevoerd ter bepaling van onzekerheden in concentraties en proces-fluxen. Tenslotte worden de resultaten gepresenteerd in tabelvorm als in *.pst format zodat deze grafisch gepresenteerd kunnen worden m.b.v JSPOST.

SUBROUTINE AnalyzeParameters(tStore, tReprs, tUpdate, parinp, dlwinp, meapst)

Parameters:

INTEGER	tStore
INTEGER	tReprs
INTEGER	tUpdte
CHARACTER(*)	parinp
CHARACTER(*)	dlwinp
CHARACTER(*)	meapst

Pseudo code:

```

Initialize( dlwinp, parinp, meapst)

do i=1,NOSIM
    Update( i, tStore, tUpdte, dlwinp)
enddo

if tReprs = MIN_MAX
then
    DetermineHyperCube( dlwinp, tStore)
else
    'Andere representatie van onzekerheid niet geïmplementeerd'
endif

Report( tReprs)

end

```

6 Level 2 subroutines

6.1 ReadModuleFile#1

SUBROUTINE: ReadModuleFile#1

Purpose: Het lezen van de input-file van de parameter-analyse module

Context: Aangeropen vanuit ReadDataBase

Description: In de input file van de parameter-analyse module staat o.a. de simulatie-periode vermeld.

SUBROUTINE ReadModuleFile#1 (parinp, tStart, tEnd)

Parameters:

CHARACTER* (*)	parinp
INTEGER	tStart
INTEGER	tEnd

Pseudo code:

Open de file 'parinp'

Lees achtereenvolgens:

tStart,
tEnd

6.2 ScanDataBase

SUBROUTINE: ScanDataBase

Purpose: Selecteren van relevante metingen uit de data-base 'datbas'

Context: Aangeropen vanuit ReadDataBase

Description: De relevante metingen uit de data-base worden in 'meapst' weggeschreven.

SUBROUTINE ScanDataBase (datbas, meapst)

Parameters:

CHARACTER*(*) datbas
CHARACTER*(*) meapst

Pseudo code:

open datbas
open meapst

de bestaande DELWAQ-routine METOJS wordt gebruikt om de relevante metingen uit 'datbas,m betreffende de periode <tStart, tEnd>, in *.pst format naar 'meapst' weg te schrijven.

end

6.3 Initialize

SUBROUTINE: Initialize

Purpose: Het initialiseren van de parameter-analyse module

Context: Aangeropen vanuit AnalyzeParameters

Description: Achtereenvolgens worden de input files voor de parameter-analyse module ('parinp') en de metingen-file ('meapst') gelezen. Deze 'meapst'-file wordt in WriteMeasurementsFile in hetzelfde format (3D-array) weggeschreven als de door DELWAQ berekende concentraties na aanroep van REAHIS. Vervolgens wordt de te analyseren parameter-ruimte gespecificeerd en de 'vase' geïntialiseerd.

SUBROUTINE Initialize (parinp, dlwinp, meainp)

Parameters:

CHARACTER*(*) parinp
CHARACTER*(*) dlwinp
CHARACTER*(*) meainp

Pseudo code:

```
ReadModuleFile#2( parinp )  
  
ReadMeasurementsFile( meapst)  
  
InitializeVase()  
  
end
```

6.4 Update

SUBROUTINE: Update

Purpose: Het updaten van de 'vase'

Context: -

Description: Functioneel ontwerp, paragraaf 3.6.

SUBROUTINE Update (inIter, tStore, tUpdte, dlwinp)

Parameters:

INTEGER	inIter	index van de iteratie
INTEGER	tStore	id. voor de manier waarop de simulatie resultaten worden opgeslagen
INTEGER	tUpdte	id. voor de wijze waarop een nieuwe parameter combinatie wordt bepaald
CHARACTER*(*)	dlwinp	DELWAQ input-file

Pseudo code:

```
REAL    newpar(NOPAR)

DetermineNewCombination( tUpdte, newpar)

Simulate( dlwinp, newpar, inIter)

GoF = DetermineGoodnessOfFit()

UpdateVase( GoF, newpar)

end
```

6.5 DetermineHyperCube

SUBROUTINE: DetermineHyperCube

Purpose: Het bepalen van de onzekerheid a.d.h. van de resulterende 'vase'

Context: Deze functie wordt aangeroepen na de laatste update slag

Description: Zie functioneel ontwerp [1], hoofdstuk 4. In de huidige versie is alleen de min/max representatie voor de onzekerheid geïmplementeerd.

SUBROUTINE DetermineHyperCube (dlwinp, tReprs, tStore)

Parameters:

CHARACTER(*)	dlwinp	DELWAQ input file
INTEGER	tReprs	de wijze van onzekerheidsrepresentatie
INTEGER	tStore	de wijze waarop om simulatie-resultaten worden gestored

Pseudo code:

```

do i=1,NOVAS

  if tStore = STORAGE_OFFLINE
    Simulate( dlwinp, vPset(i,:), -1)
  else
    lees de par-combinatie uit 'par<i>.sim'
    lees de concentraties uit 'conc<i>.sim'
    lees de proces-fluxen uit 'flux<i>.sim'
  endif

  if tReprs = REPRESENTATION_MINMAX

    do j=1,NOPAR
      bepaal vAvg(j)
      bepaal vMin(j)
      bepaal vMax(j)
    enddo

    do j=1,NOSUB
      gebruik rConc(j,...) bij bepaling hAvgC, hMinC en hMaxC
    enddo

    do j=1,NOPRO
      gebruik rFlux(j,...) bij hAvgF, hMinF en hMaxF
    enddo
  else
    'Onbekend type voor representatie onzekerheid'
  endif
enddo
end

```

6.6 Report

Function: Report

Purpose: Het rapporteren van de resultaten

Context: Aangeropen na bepaling van de onzekerheid

Description: Het exporteren van de concentraties en proces-fluxen naar JSPOST voor grafische verwerking.

SUBROUTINE Report (tReprs)

Parameters:

INTEGER tReprs id. voor representatie onzekerheid

Pseudo code:

```
if TReprs = REPRESENTATION_MINMAX
```

```
    schrijf de concentratie en de proces-flux hyper-cubes  
    als uitvoer in JSPOST formaat, volgens de beschrijving  
    van hoofdstuk 5 van dit technisch detailontwerp.
```

```
else
```

```
    'Onbekend representatietype'
```

```
endif
```

7 Level 3 subroutines

7.1 ReadModuleFile#2

SUBROUTINE: ReadModuleFile#2

Purpose: Het lezen van de input-file van de parameter-analyse module

Context: Aangeropen vanuit de subroutine Initialize

Description: In de input file van de parameter-analyse module staan vermeld:
de simulatie-periode, de DELWAQ uitvoer-tijdstap, file-namen, de parameters, de stoffen, de procedure voor het updaten van de 'vase', de wijze van opslag van de simulatie-resultaten en de wijze van representatie van de onzekerheid.

SUBROUTINE ReadModuleFile#2 (parinp)

Parameters:

CHARACTER*(*) parinp

Pseudo code:

Open de file parinp

Lees achtereenvolgens:

tStart,
tEnd,
tDelta,

dlwinp,
meapst,

tStore,
tReprs,
tUpdate,

NOVAS,
NOPAR,
NORAN,
NOSIM,
NOSUB,
NOPRO,
NOSEG

```
do i=1,NOPAR
  lees pType(i)
  lees pName(i)
  lees pLoc(i)
  lees pMin(i)
  lees pMax(i)
enddo
```

```
do i=1,NOSUB
  lees sName(i)
  lees sNwin(i)
  lees sExp(i)
  lees sWght(i)
enddo
end
```

7.2 ReadMeasurementsFile

SUBROUTINE: ReadMeasurementsFile

Purpose: Het vullen van een 3D-array met gemeten concentraties

Context: Deze functie is onderdeel van de initialisatie

Description: De metingen bevatten bestaan uit een serie concentraties, geïndexeerd op (stof-id., tijd, plaats).

SUBROUTINE ReadMeasurementsFile (meapst)

Parameters:

CHARACTER*(*) meapst

Pseudo code:

open 'meapst'

lees 'meapst'-file

initialiseer mConc(NOSUB, NOTOT, NOSEG)

met:

1e index = index van de stof in sName

2e index = tijd-index (geheel aantal keren de uitvoer-tijdstap van DELWAQ)

3e index = segmentnummer van de meetlocatie

end

7.3 InitializeVase

Function: InitializeVase

Purpose: Het initialiseren van de 'vase'

Context: Laatste stap in initialisatie van parameter-analyse

Description: Zie functioneel ontwerp [1], paragraaf 3.4.

SUBROUTINE InitializeVase ()

Parameters: -

Pseudo code:

```

REAL          p(NOVAS,NOPAR)

do j=1,NOPAR

    min = pMin(j)
    max = pMax(j)
    intervalStep = (max - min)/NOVAS

    do i=1,NOVAS
        intervalMin = min + (i-1)*intervalStep
        intervalMax = min + i*intervalStep
        p(i,j) = IntervalRandom( intervalMin, intervalMax)
    enddo
enddo

do i=1,NOVAS

    do j=1,NOPAR
        k = INT (IntervalRandom( 1, NOVAS - i+1))
        vPset(i,j) = p(i,k)

        do index=k,NOVAS - i
            p(index,j) = p(index+1,j)
        enddo
    enddo
enddo

Bereken vMax, vMin en vAvg:

GoFMax = 0.0
vIGoF = 0

do i=1,NOVAS
    Simulate( dlwinp, vPset(i,:), -1)
    vGoF(i) = DetermineGoodnessOfFit()

    if (vGoF(i) > GoFMax)
        vIGoF = i
        GoFMax = vGoF(i)
    endif
enddo

```

```
Store de parameter-combinatie in 'par<i>.sim'

if (tReprs = MIN_MAX)
    Store de concentraties in 'conc<i>.sim'
    Store de proces-fluxen in 'flux<i>.sim'
endif
enddo
end
```

7.4 DetermineNewCombination

SUBROUTINE: DetermineNewCombination

Purpose: Het bepalen van een nieuwe parameter-combinatie

Context: Aanroep tijdens update-slag van de 'vase'

Description: Momenteel is alleen de methode van Price geïmplementeerd
Andere methoden kunnen worden toegevoegd, zie stap 1
en 2 van de procedure op pag. 10 van het FO [1].

SUBROUTINE DetermineNewCombination (TUpdte, outCom)

Parameters:

INTEGER	tUpdte	id. voor de optie m.b.t. het bepalen van een nieuwe parameter-combinatie
REAL	outCom	de nieuwe parameter-combinatie

Pseudo code:

```

REAL      pRan(NORAN,NOPAR)
REAL      pNew(NOPAR)
LOGICAL    drawn(NOVAS)

if tUpdte = UPDATE_PRICE

  do i=1,NORAN
    drawn(i) = FALSE
  enddo

  do i=1,NORAN
    j = INT (IntervalRandom( 1, NOVAS))

    if ( drawn(j) = FALSE)
      then
        pRan(i,.) = vPset(j,.)
        drawn(j) = TRUE
      else
        i = i - 1;
      endif
    enddo

    pNew = (berekening volgens formule 10 FO-paragraaf 3.6)

  else
    'een onbekende update methode is aangegeven'
  endif
end

```

7.5 DetermineGoodnessOfFit

FUNCTION: DetermineGoodnessOfFit

Purpose: Bepaalt Goodness-of-Fit voor gegeven parameter-combinatie

Context: Aangeropen bij initialisatie, bij update en bij
onzekerheidsbepaling

Description: Bepaling GoF volgens formules (1)-(3) van het functioneel
ontwerp [1].

FUNCTION DetermineGoodnessOfFit ()

Parameters:

-

Pseudo code: z.o.z.

Pseudo code:

```

REAL          concentration

do i=1,NOSUB

  NWMAX(i) = INT((tEnd - tStart)/(sNwin(i)*tDelta))
  substanceTerm = 0.0

  het gewicht van stof i = sWght(i)

  do m=1,NOSEG
    nFits = 0

    do w = 1, NWMAX(i)
      tMin = tStart + (w-1) * sNwin(i) * tDelta
      tMax = tStart + w * sNwin(i) * tDelta
      nMeting = 0

      do n=1,sNwin(i)
        if mConc(i, (w-1)*sNwin(i)+n, m) /= -999.
          then
            meting = meting + mConc(i, (w-1)*sNwin(i)+n, m)
            nMeting = nMeting + 1
          endif

          do n = 1, sNwin(i)
            concentration = concentration +
              rConc( i, (w-1)*sNwin(i)+n, m)
          enddo

          concentration = concentration / sNwin(i)

          if nMeting > 0
            then
              meting = meting/nMeting
              term = term + ABS(meting-concentration)**sExp(i)
              nFits = nFits + 1
            endif
          enddo
        enddo

        substanceTerm = sWght(i) * term / nFits
      enddo

      DetermineGoodnessOfFit =
        DetermineGoodnessOfFit + substanceTerm

    end
  end

```

7.6 UpdateVase

SUBROUTINE: UpdateVase

Purpose: Het updaten van de 'vase' als een parameter-combinatie gevonden is met $GoF < GoF_{Max}$. Als `STORAGE_ONLINE` is gekozen worden de resultaten behorende bij de parameter-combinatie met $GoF = GoF_{Max}$ overschreven door de resultaten die behoren bij de nieuwe parameter-combinatie.

Context: Deze functie sluit een update-iteratie af.

Description: Zie functioneel ontwerp [1], par. 3.6, stap 5 van de update procedure.

SUBROUTINE UpdateVase (inFit, inComb, tStore)

Parameters:

REAL	inFit	GoF-waarde
REAL	inComb(*)	parameter-combinatie
LOGICAL	tStore	id. voor wijze van opslag simulatie resultaten

Pseudo code:

```

if (inFit < vGoF(vIGoF))
  vervang vPset(vIGoF,.) door inComb

  Store de parameter-combinatie in 'par,vIGoF>.sim'

  if tStore = STORAGE_ONLINE
    Store de concentraties in 'conc<vIGoF>.sim'
    Store de proces-fluxen in 'flux<vIGoF>.sim'
  endif

  GOFMax = 0.0
  vIGoF = 0

  do i=1,NOVAS

    if (vGoF(i) > GOFMax)
      vIgoF = i
      GOFMax = vGoF(i)
    endif

  enddo
endif
end

```

7.7 Simulate

SUBROUTINE: Simulate

Purpose: Het uitvoeren van een simulatie met DELWAQ met een nieuwe parameter-combinatie

Context: Deze functie wordt gebruikt tijdens het updaten van de vase

Description: Eerst wordt de bestaande invoer file aangepast, zodat een DELWAQ-simulatie kan worden uitgevoerd met de nieuwe parameter-combinatie. Tenslotte worden de concentraties en proces-fluxen gelezen en evt. opgeslagen

SUBROUTINE Simulate (dlwinp, inComb)

Parameters:

CHARACTER*(*)	dlwinp	DELWAQ input file
REAL	inComb	de te simuleren parameters

Pseudo code:

WriteParameterSection(dlwinp, inComb, dlwsim)

voer een DELWAQ-run uit met dlwsim als invoerfile

ReadConcentrations(dlwsim)

ReadProcessFluxes(dlwsim)

end

8 Level 4 subroutines

8.1 IntervalRandom

Function: IntervalRandom

Purpose: Het trekken van een random getal uit een interval

Context: Deze functie wordt op diverse plaatsen aangeroepen.

Description: Zie functioneel ontwerp [1], paragraaf 3.5. RANX genereert een random getal uit een uniform verdeeld interval $<0,1]$

FUNCTION IntervalRandom(lower, upper)

Parameters:

REAL lower	ondergrens interval
REAL upper	bovengrens interval

Pseudo code:

seed = system time

IntervalRandom = (lower + (upper - lower)*RANX(seed))

end

8.2 WriteParameterSection

SUBROUTINE: WriteParameterSection

Purpose: Het aanpassen van de invoer file voor DELWAQ2 zodat een DELWAQ-simulatie met de gegeven parameter-combinatie kan worden uitgevoerd.

Context: Deze functie wordt bij elke simulatie met DELWAQ gebruikt, zowel tijdens het updaten van de 'vase' als tijdens het bepalen van de onzekerheden in concentraties en fluxen.

Description: De parameters kunnen zijn proces-constanten of schaal factoren voor de waste-loads.

SUBROUTINE WriteParameterSection (dlwinp, parcom)

Parameters:

CHARACTER*(*)	dlwinp	DELWAQ input file
CHARACTER*(*)	parcom	file met de nieuwe parameter-combinatie

Pseudo code:

CHARACTER*(*) dlwsim = invoer-file van de nieuwe simulatie

Het format van parcom is:

'de reactie-coëfficiënten'

.		
.		
.		
Parameter-naam	'Domain'	parameter-waarde
.		
.		
.		

'de schalingsfactoren voor de belastingen'

.		
.		
.		
Stofnaam	Naam locatie	parameter-waarde
.		
.		
.		

8.3 ReadConcentrations

SUBROUTINE: ReadConcentrations

Purpose: Het lezen van de concentraties van de stoffen uit de uitvoer file van de simulatie behorende bij een nieuw bepaalde parameter-combinatie.

Context: Aangeropen na elke simulatie.

Description:

SUBROUTINE ReadConcentrations (dlwsim)

Parameters:

CHARACTER*(*) dlwsim DELWAQ resultaat file

Pseudo code:

leest de berekende concentraties in uit de DELWAQ *.HIS file

slaat de berekende concentraties op in
rConc(conc. id., fysieke tijd, segmentnummer) in dezelfde volgorde
als in sName (d.w.z. de 1e index in rConc komt overeen met de
index van de stof in sName). Hierbij wordt gebruik gemaakt van de
bestaande routine REAHIS.

end

8.4 ReadProcessFluxes

SUBROUTINE: ReadProcessFluxes

Purpose: zie ReadConcentrations, nu voor proces-fluxen

Context: Aangeropen na elke simulatie.

Description:

SUBROUTINE ReadProcessFluxes (dlwsim)

Parameters:

CHARACTER*(*) dlwsim DELWAQ resultaat file

Pseudo code:

leest de berekende procesfluxen in uit DELWAQ *.HIS file

slaat de fluxen op in rFlux(flux. id., fysieke tijd, segmentnummer)

end

9 Input-file voor parameter-analyse module

9.1 Inleiding

De parameter-analyse module maakt zoveel mogelijk gebruik van informatie die al in de DELWAQ input-files en metingen-data bases is opgeslagen. Specificatie van een aantal grootheden die voor de parameter-analyse van nodig zijn is in deze aparte input-file ontworpen die met een ASCII-editor kan worden samengesteld. Als de parameter-analyse module in een later stadium onder een user interface wordt gebracht, kan dit user interface de aanmaak van de input-file 'parinp' overnemen. De parameter-analyse module zelf hoeft dan niet aangepast te worden.

9.2 Format van de input-file

tStart	tEnd	tDelta		
dlwinp	meapst			
tStore	tReprs	tUpdte		
NOVAS	NOPAR	NORAN	NOSIM	
NOSUB	NOPRO	NOSEG		
pType(1)	pName(1)	pLoc(1)	pMin(1)	pMax(1)
..	..	..	..	..
..	..	..	..	..
pType(NOPAR)	pName(NOPAR)	pLoc(NOPAR)	pMin(NOPAR)	pMax(NOPAR)
sName(1)	sNwin(1)	sExp(1)	sWght(1)	
..	..	..	..	
..	..	..	..	..
sName(NOSUB)	Nwin(NOSUB)	sExp(NOSUB)	sWght(NOSUB)	

Waarin:

Naam	Type	Beschrijving
tStart	INTEGER	starttijd van de DELWAQ-simulatie
tEnd	INTEGER	eindtijd van de DELWAQ-simulatie
tDelta	INTEGER	uitvoertijdstep van DELWAQ
dlwinp	CHARACTER*(*)	DELWAQ invoer-file
meapst	CHARACTER*(*)	metingen-file in *.pst format
tStore	INTEGER	id. voor wijze van opslag simulatie-resultaten
tReprs	INTEGER	id. voor wijze waarop onzekerheid gerepresenteerd wordt
tUpdte	INTEGER	id. voor wijze waarop nieuwe parameter-combinatie wordt bepaald
NOVAS	INTEGER	Omvang van de 'vase'
NOPAR	INTEGER	Aantal parameters
NORAN	INTEGER	Omvang random trekking uit 'vase'
NOSIM	INTEGER	Aantal DELWAQ simulaties
NOSUB	INTEGER	Aantal stoffen
NOPRO	INTEGER	Aantal processen
NOSEG	INTEGER	Aantal segmenten
pType	INTEGER	Parametertype
pName	CHARACTER*(*)	Parameter-naam of stofnaam
pLoc	CHARACTER*(*)	Locatie-naam of DOMAIN (default)
pMin	REAL	Ondergrens waarde van de parameter
pMax	REAL	Bovengrens waarde van de parameter
sName	INTEGER	Stofnaam
sNwin	INTEGER	Lengte time-window
sExp	REAL	Exponent
sWght	REAL	Weegfactor

"

10 JSPOST invoer formaat

10.1 Inleiding

Invoer files voor JSPOST komen altijd als paar voor. Het gaat dan om een stuurbestand met informatie over welke data op file staan en een data bestand met alleen de daadwerkelijke data.

10.2 Het stuurbestand

Het stuurbestand heeft extensie 'stu'. Het is een ASCII file die fixed format ingelezen wordt:

```

2I5      NOSEG, NOVAL
3F10.0    TSTART, TSTOP, DELTT
A        HEAD1
A        HEAD2
A        LABELX
2A,2F12.0  SYSNAM(1), LABELY(1), SCLMIN(1), SCLMAX(1)
.
2A,2F12.0  SYSNAM(NOVAL), LABELY(NOVAL), SCLMIN(NOVAL), SCLMAX(NOVAL)
A        SEGNAM(1)
.
A        SEGNAM(NOSEG)

```

Het laatste record met SEGNAM is optioneel.

Naam	Type	Beschrijving
NOSEG	INTEGER	aantal segmenten
NOVAL	INTEGER	aantal uitvoer grootheden
TSTART	REAL	starttijd
TSTOP	REAL	stoptijd
DELTT	REAL	stapgrootte
HEAD1	CHARACTER*80	titel
HEAD2	CHARACTER*80	subtitel
LABELX	CHARACTER*12	label x-as
LABELY	CHARACTER*12	label y-as
SYSNAM	CHARACTER*24	systeem naam
SEGNAM	CHARACTER*24	segment naam
SCLMIN	REAL	minimum waarde y-as
SCLMAX	REAL	maximum waarde y-as

Het aantal tijdstippen in de file wordt berekend volgens:

$$NOTIME = NINTEGER \left((TSTOP - TSTART) / DELTT + 1 \right)$$

Met de reals is het oppassen dat hier geen afrondings problemen zijn want anders wordt het data bestand met een andere NTIME ingelezen dan dat het geschreven is wat tot leesfouten of

verschuivingen van data kan leiden.

10.3 Het data bestand

Het data bestand heeft de extensie 'pst'. Het is een binaire file zonder records (Microsoft FORM='BINARY', Lahey, Salford ACCESS='TRANSPARENT').

Dit bestand bevat de gehele matrix aan getallen in de volgende volgorde (FORTRAN notatie):

REAL VALUE(NOVAL,NOTIME,NOSEG)

Waarin:

Naam	Type	Beschrijving
NOSEG	INTEGER	aantal segmenten
NOTIME	INTEGER	aantal tijdstippen
NOVAL	REAL	aantal uitvoer grootheden

Missing values worden weergegeven door het getal -999.

11 Referenties

[1]

P.G.J. ten Brummelhuis, M.R.L. Ouboter; Functioneel ontwerp parameter-analyse module; WL,T1496, November 1994