# Rekenfout Mesdag, *bug* in OPS-RIVM, of *clusterfuck*? Een technisch perspectief op *The Blame Game*.

*Evert Mouw, 2020-03-06*

Grote krantenkoppen. **Bug** in de software van het RIVM. Kunnen die amateurs wel programmeren? **Rekenfout** van het Mesdag – kunnen die lui wel met cijfers omgaan? Voer voor journalisten. Maar hoe zit het nu echt? Wie kun je vertrouwen?

### Wie ben ik? Waarom schrijf ik dit?

Het is arrogant om te zeggen dat je mij kunt vertrouwen, maar wat deze vraag betreft heb ik misschien wel de goede papieren. Ik word niet betaald. Ik ben geen onderdeel van het RIVM of van Mesdag. En: ik heb *dezelfde* berekeningen uitgevoerd als het Mesdag. Bij informatica had ik een negen voor software engineering, ik deed onafhankelijk onderzoek naar het stikstofbeleid waar ik een maand terug over publiceerde via een rapport, en ik heb wat ervaring met het gebruik van dit soort rekensoftware in grote clusteromgevingen (grid computing). Daarnaast had ik contact met zowel het RIVM als met Mesdag over deze kwestie. Hoewel ik zeer, *zeer* sympathiek sta richting de boeren en bouwers (ze hebben *echt* een punt) vind ik ook dat het RIVM goed werk doet, integere medewerkers heeft, en nog onafhankelijker moet worden. Het echte probleem zit m.i. bij hun "opdrachtgevers", bij onze bestuurders en politici, en dus ook bij de media die hiervan een verlengstuk zijn.

Ik heb dagen gewacht met dit opschrijven. Het is niet fijn voor Mesdag, het is niet fijn voor het RIVM, en tja… die zitten beide dus niet echt in mijn vizier. Maar na de poppenkast in de media moet het maar. Hou die *blame game* die de media ervan maken in gedachten terwijl je verder leest.

Nu ter zake. Wat chonologie, 't probleem simpel verwoord, en op het eind enkele conclusies.

### De ontdekking van de *fuck-up*

Mijn eigen onderzoek richtte zich op de uitstoot (emissiedata) en het beleid. Dus rekenen met het OPS hoefde ik niet te doen. Nadat ik klaar was, en de publieke reactie van het RIVM verwerkt had in een geactualiseerde versie van mijn rapport, werd ik nieuwsgierig naar de OPS berekeningen van Mesdag. Richard had na de presentatie bij Nieuwspoort de boel online gezet. Ik kon dus de bestanden downloaden en ze nadoen. Een pure *nerd*-interesse: kan ik het sneller doen dan hij? Op Twitter zag ik dat ie er uren over deed… Ik brak de invoer in stukjes op, *filterde* (!!!) de invoer, en gaan! Wat shellscript verder en ik had getalletjes. Alleen, ze klopten niet met die van Mesdag. Wel, ik liep maar

wat te kloten, dus ik vroeg 't de Mesdag analist, Richard, op Twitter, via een PM. Hele gesprekken volgden...

Ineens werd bij het project van Mesdag een "issue" gemeld, van iemand die ook opdrachten doet voor het RIVM. Probleem: de invoer voor OPS moet gesplitst worden. Dus regels invoer voor NOx mogen niet voorkomen als NH3 depositie berekend wordt. Hoe gaat dat? OPS is een command-line programma. Via een "stuurfile" (niet mijn term) geef je aan welke depositie je wilt berekenen (NOx, NH3, of wat anders), op welke punten je dat wilt doen (receptorpunten), welke meteo-bestanden je gebruikt, en ook welk bestand als invoer dient (de emissie brondata).

In de emissie brondata staat een kolom `component`. Daar staat in dat 't over NH3 emissie gaat, of juist NOx. Op die manier weet OPS om wat voor emissie het gaat... toch? Nee, toch niet. Het leest die hele kolom niet in. Je moet van tevoren al splitsen. Dat staat nergens in de documentatie, of we hebben er allemaal overheen gelezen. Wel had het RIVM de emissiedata gescheiden volgens deze twee stoffen, maar Mesdag heeft het dus gecombineerd.

Ook is er een Excel macro (*seriously?*) om de invoer te controleren en dus goede invoer voor het OPS te maken. Daar is `component` als "verplicht" aangeduid. Als je vrolijk NOx en NH3 door elkaar gooit, dan maakt het invoer die ook beide stoffen bevatten. Vervolgens gaat het OPS dus rekenen alsof het allemaal dezelfde stof betreft, volgens de specificatie van de "stuurfile".

Hieronder een screenshot van hoe dat eruit ziet.

Is dit een *fuck-up*? Volgens mij wel. Invoervalidatie is toch wel iets dat bij de *basics* hoort. Ik liet de screenshot aan m'n vriendin zien, arts, maar heeft ook ooit een intro programmeren in C++ gedaan op het LIACS, en die zag 't meteen. Ze vond de code ook heel leesbaar. Logisch; Fortran is best wel gemakkelijk vergeleken met C++.

Tussendoor: Ik heb al heel wat mensen horen klagen dat het Fortran code is. *Bullshit.* Fortran is de beste programmeertaal voor dit soort rekenwerk, sneller dan C, terwijl het toch goed leesbaar is voor andere wetenschappers. Ga maar wat kantoorsoftware in Java knutselen met objectjes, maar blijf s.v.p. met je handen af van serieuze rekensoftware als je denkt dat Fortran ongeschikt is. Fortran-bibliotheken zijn nog steeds de basis voor veel functies en modules in R, Python en andere talen. Wel heeft de huidige OPS-code echt behoefte aan modernisatie, refactoring en andere verbeteringen.

### Hoe komt dit zo?

Maar hoe is dat ontstaan? Ik heb het sterke vermoeden dat het ongeveer zo gegaan is.

1. We maken een mooi OPS om een simulatie mee te doen. We zitten op UNIX of zoiets (de Windows versie klaagt over een ontbrekend `clear` commando,

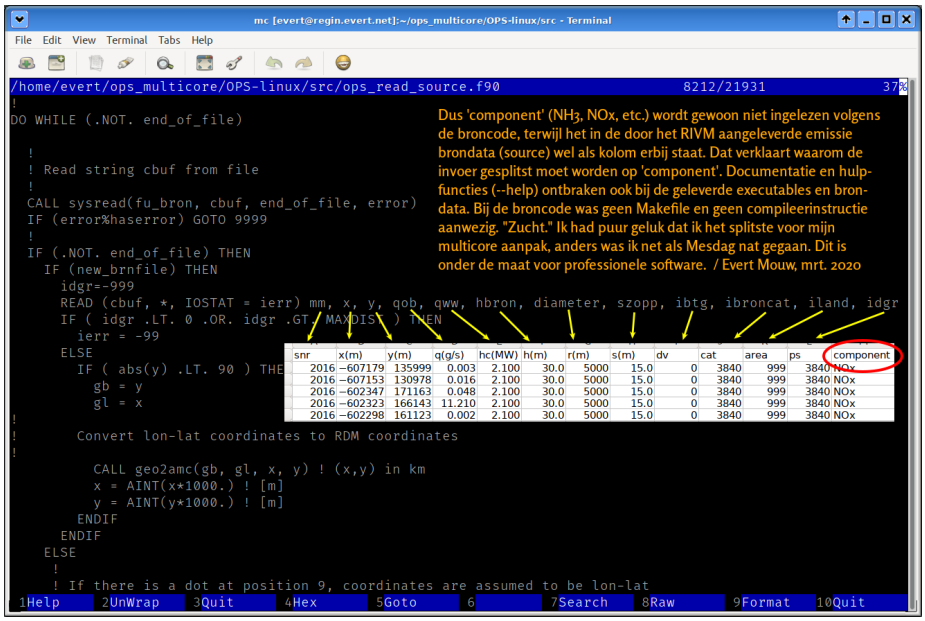


Figure 1: Een korte samenvatting van het probleem, zoals ik dat eerder op Twitter plaatste bij technische discussies.

iets wat standaard in Unix / Linux zit), dus op serieuze computers of zelfs rekenclusters. Ik gok eind jaren '70, jaren '80 ofzo.

2. Steeds opnieuw inloggen op een command line shell... Nee, regel even een fijne grafische interface! Voor Windows wordt er een grafisch schilletje gemaakt. En om de invoerbestanden wat gemakkelijker aan te maken komt er een Excel macro. De eindgebruikers zitten niet meer op het rekencluster, de bediening gaat niet meer door informatici, maar de eindgebruikers zijn nu mensen met een Windows desktop.
3. We maken een extra kolom (invoerveld) met de naam `component`, zodat we weten of we het over NOx, NH3 of wat anders hebben. We updaten onze Excel macro's. We vergeten de OPS code te actualiseren, want we spreken intern af dat we de invoerbestanden gescheiden naar component aanleveren. Dus of enkel NOx, of enkel NH3. De macro controleert dat verder niet.
4. Niemand maakt zich heel druk over de OPS code omdat het allemaal werkt. Het kost geld en tijd om de boel actueel te houden.
5. Ineens komt er grote druk om onze spullen openbaar te maken. We moeten de boel online knallen. Mesdag gaat ermee rekenen.
6. Onvoorzien: Mesdag denkt dat het de invoer kan combineren, omdat er een kolom `component` in staat.
7. Mesdag krijgt foute resultaten. Oeps. Snel informeren... maar de cijfers zijn al wereldkundig gemaakt.

8. *Blame game.*

**Echt een bug? Wie kunnen we de schuld geven?**

De "blame" betreft m.i. beide partijen. Als Mesdag in een eerder stadium de berekeningen en werkwijze openbaar had gemaakt, dan waren fouten en/of misverstanden in een eerder stadium aan het licht gekomen. Dan waren zowel het RIVM als Mesdag niet hiermee in slecht vaarwater gekomen. Ik kon ook pas erbij na de presentatie bij Nieuwspoort. Maar het RIVM had ook al jaren terug de boel *open source* kunnen maken, had de eigen OPS codebase en documentatie echt veel beter kunnen onderhouden (hallo! dit is de basis van een fucking stikstofcrisis!) enzovoorts. Ik gok dat veel technische jongens dat ook wel wilden, maar ja, managers, belangen, iedereen weet wel hoe dat gaat.

Open en transparant werken geeft in het eerste stadium wat pijn. Want alle gebreken en fouten worden zichtbaar. Op lange termijn maakt het je spul juist solide en sterk, en vermijd je een grote clusterfucks. Zo'n analist van Mesdag, die dagen voor niks heeft lopen rekenen... Wetenschappers van het Mesdag, die te horen krijgen dat hun cijfers niet kloppen... Media en politiek, die het allemaal nog extra opblazen... Al snel ontstaat een dynamiek waarbij de inhoud op de achtergrond raakt. Dus, heren van beide kampen, geen nood, de emotie is begrijpelijk. Maar als schrale troost: na mijn stikstofrapport en journalistieke aandacht kreeg ik vanuit extreemlinkse hoek (enkele GroenLinks dwaallichten en antifa) kwalificaties als neonazi e.d., dus het kan altijd nog een graadje erger. Ondertussen worden massaal bomen gekapt in het kader van "natuurherstel" en bepalen mensen die communicatie gestudeerd hebben een groot deel van het publieke debat. Inhoud is hun vijand: dan moeten ze rekenen en computercode begrijpen. Emotie, framing en labels zijn hun vrienden.

Het woord "bug" is wat mij betreft wat ongelukkig, maar niet geheel onterecht. Een klassieke "bug" of insect kon in de prehistorie nog wel 's voor computerstoringen zorgen. Hier een voorbeeld van een bug, in pseudocode:

```
functie Optellen (a, b) {
    Resultaat = a - b
    retourneer Resultaat }
```

Kijk, da's een *typo*. Waar een `-` staat, had een `+` moeten staan. Het OPS had niet zo'n bug. Maar het OPS had ook geen goede invoervalidatie, en dat is een "in gebreke blijven" of gebrek. Ook leest het OPS de latere, nieuwere kolom `component` niet eens in, waardoor het doet alsof alle invoer de component betreft die in de "stuurfile" aangegeven is. Dat is een gebrek. En het is niet goed gedocumenteerd ook nog. Dat is zeker een gebrek. Kortom, geen klassieke bug, maar goed in orde is het ook niet. Er zijn softwareprojecten die elke fout of elk gebrek in de documentatie zijn als *bug*. Volgens die wat erg strenge definitie is er inderdaad een bug. Dat Mesdag dan foute rekenresultaten kreeg is niemand helemaal verwijtbaar, maar wel begrijpelijk.

Het is wel een soort ongeschreven regel binnen dit soort informaticakringen om zulke rekenprogramma's alleen de invoer aan te bieden die nodig is voor de berekening. Vandaar ook dat ik zelf al ging splitsen in mijn scriptjes (`grep NOx emissies.brn`), maar ongeschreven regels zijn weinig transparant, en vaak ook niet bekend bij andere disciplines. Computer- en programmeerkennis was bij de Mesdag groep meer dan voldoende aanwezig, maar was meer gericht op geografische informatieverwerking e.d.

Hoe gingen de partijen ermee om? Heel netjes. Een RIVM medewerker gaf al vroegtijdig een noodsignaal richting Mesdag. Ik deed wat berekeningen en communiceerde dat naar Mesdag. Ook lichtte ik het bestuur van de FDF in, omdat ik vreesde dat betrokken partijen benadeeld zouden worden. Hun voorman, Mark, reageerde dat de cijfers wel correct moesten zijn. Het Mesdag bracht zelf het slechte nieuws naar buiten. Het RIVM nodigde ze daarna uit voor een gesprek, om e.e.a. scherp te krijgen. Journalisten en politici maakten er vervolgens, je verwacht het verd* niet, een spektakel van.

De Mesdag analist gaf via een PM als commentaar dat de code niet voldoet aan diverse eisen die de overheid zelf vastgesteld heeft voor software die voor/door de overheid geschreven wordt; en dat ook andere ISO normen, SCRUM best practices e.d. niet gebruikt worden. Hoewel ik dat zeker wil geloven, ben ik dat verder niet nagelopen, omdat ik niet veel geef om protocollen en checklists. Ze hebben zeker hun nut. Maar de *elegantie* van code en/of systemen is lastig meetbaar of te vangen in checkboxes, terwijl het OPS ook eigenlijk geen "gewone" software is maar erg specifiek is, voor niet-kantoor omgevingen ontworpen. Dat er later een Windows GUI en Excel macro's tegenaan gegooid zijn bewijst dat de "human interface" of interdisciplinaire scheiding in benaderingen ook kan hebben bijgedragen aan het niet aansluiten van invoervelden, documentatie etc.

Genoeg over de problemen. Vooruit nu. Wat moeten we doen?

### Verbeterpuntjes (*actionables* voor managers)

Gelukkig hebben we nu het rapport van de Adviescollege Meten en Berekenen Stikstof (commissie Hordijk). Die constateert enkele zaken die overeenkomen met conclusies uit mijn rapport. Ze pleiten voor meer transparantie (check), meer onafhankelijkheid van het RIVM (check), etc. Maar wat hebben ze nu precies gedaan? Met de software gespeeld? Zelf gerekend met emissiedata, of de broncode bekeken, of een berekening met het OPS zelf uitgevoerd? Niets van dat alles. Als ik zelf in zo'n commissie zou zitten, zou ik wel even willen zien waar de cijfertjes nu precies mee gemaakt worden. Ik zou misschien ook contact zoeken met de auteur van een rapport dat op de RIVM site van een inhoudelijke reactie voorzien wordt (het mijne). Dat soort dingen. Het is een beetje alsof je een APK keuring doet door de eigenaar van een auto te interviewen, zonder onder de motorkap te kijken. Daarom hier nog wat puntjes voor verbeteringen onder de motorkap.

Mijn technische aanbevelingen:

- Neem de OPS code in actief onderhoud. Als hele en halve sectoren en overheidsbeleid hierop hangen, dan moet het goed zijn.
- Fortran is prima, de beste keuze zelfs, maar doe refactoring naar modern Fortran, die ook alle rekenkernen van een CPU tegelijk kan benutten.
- Verbeter de invoervalidatie. Zorg dat de Excel macro aansluit op de OPS code. Of, misschien nog beter, stop met dat soort maco's.
- Doe serieus open source. Geen `Makefile` bijleveren, kom op... Controle is een illusie.
- Verbeter de *logging*. Niet alleen een `.err` bestand, maar ook naar `stderr` e.d., eventueel via command line options.
- Voeg voorbeelden en/of validatiesets toe.
- Haal compiler afhankelijkheden weg. Zorg dat het compileerbaar is met GNU Fortran (gfortan), zoals al gedaan door Jeroen Laros.
- Open source hoeft niet te betekenen dat *iedereen* jouw berekeningen kan overdoen. Een beetje expertise mag verwacht worden. Plus linkjes naar waar je die expertise zelf kunt opdoen. (Mening!)

Succes ermee. Nog vragen? Je weet me te vast wel te vinden. Maar ik moet ook weer 's wat aan mezelf gaan denken; dit soort vrijwilligerswerk voor de samenleving vreet tijd en ik moet weer 's wat aan de pecunia gaan denken. Dus s.v.p. enkel vragen sturen nadat je zelf wat moeite gedaan hebt om e.e.a. te snappen. Waar ik fouten maak, ontvang in uiteraard graag correcties.