Meetdoelenproject

Workflow Python FASE 1

Technologiecentrum (AM)

water voor nu en later

Vitens

Fase 1: Toelichting

Inventarisatie van meetdoelen o.b.v. criteria

Doel van Fase 1 is om geautomatiseerd (via scripting in Python) de meetdoelen van alle Vitens peilbuizen te inventariseren.

Waarom geautomatiseerd?
→ Doordat de meetdoelen in het verleden niet juist zijn gedocumenteerd is het momenteel onbegonnen werk om deze van alle 10.000 peilbuizen handmatig te achterhalen.

Geautomatiseerd, hoe doe je dat?
→ O.b.v. vooraf opgelegde criteria (bijv. hoe diep staat een filter? Hoe ver staat een buis van een winning? Welk landgebruik past bij de buis?) worden middels een script in python meetdoelen toegewezen aan een peilbuis.
→ Per peilbuis wordt in feite de vraag gesteld: Welk meetdoel zou je verwachten bij een peilbuis met X filters, op afstand Y van een winning en landgebruik Z? Of: Welk meetdoel zou deze peilbuis kunnen dienen?

Weten we dan van geen enkele peilbuis waarom die oorspronkelijk is geplaatst?
→ Uiteraard zijn de meetdoelen van een groot aantal peilbuizen gelukkig wel gedocumenteerd. Denk aan vergunningsplichtig, maar ook commerciële meetnetten.
→ In fase 1 zal daarom onderscheid worden gemaakt tussen een gedocumenteerd en geïnventariseerd meetdoel.

Wat levert fase 1 uiteindelijk op?
→ Aan iedere peilbuis van Vitens wordt een (gedocumenteerd of geïnventariseerd) meetdoel toegewezen.
→ Gedocumenteerd meetdoel: Van de peilbuizen waarvan we nog weten waarom ze bestaan
→ Geïnventariseerd meetdoel: Oorspronkelijk meetdoel is onbekend, maar peilbuis kan een meetdoel dienen

Fase 1: workflow pythonscript

Stap 1. Importeren van Objectenbeheer: ‘Meetpunt’

Tabblad ‘Meetpunt’ van Objectenbeheer wordt geïmporteerd in python.

→ Op 17 maart 2020 worden 10.188 rijen en 30 kolommen ingeladen.

Obj_Meetpunt - DataFrame

Index	atum_aangemaak	datum_gewijzigd	id	business_id	asset_id	olga_code	nitg_code	bro_code	oude_codering	meetpuntsoort	coordinaat_x	coordinaat_y	meetpuntstatus	datum_plaatsing
0	2014-12-05 08:31:53.4680…	2018-10-31 13:02:53.1960…	12204	40C3540	nan	nan	B40C3540	nan	40C-E1st GRK	waarnemingsput	186807.8	436942	actief	2014-04-07 00:00:00
1	2014-03-27 12:00:00	2020-01-17 09:47:46.8780…	10148	33DP0056	nan	33DP0056	B33D0236	nan	nan	waarnemingsput	198332.8	458252.7	actief	NaT
2	2015-09-29 08:47:29.0760…	2018-01-25 10:18:03.1470…	12279	41E0876	nan	nan	B41E0876	nan	Kbveen B	waarnemingsput	242591	445956	actief	2014-08-20 00:00:00
3	2014-03-27 12:00:00	2019-06-18 09:04:01.6180…	5348	MEWP006	nan	31GP0196	B31G0196	nan	nan	waarnemingsput	129955.1	455725.4	actief	NaT
4	2014-03-27 12:00:00	2020-01-09 13:43:59.0910…	4384	26F0095	nan	26FP0095	B26F0095	nan	nan	waarnemingsput	178065.9	492940.2	actief	NaT
5	2014-03-27 12:00:00	2019-08-22 11:42:48.2170…	9116	B26G0306	nan	26GL0005	B26G0306	nan	nan	waarnemingsput	168660	481270	actief	2002-07-18 00:00:00
6	2018-02-26 10:51:02.1460…	2019-08-22 11:42:50.7350…	12495	32BM001	nan	nan	B32B2747	nan	N10-001	waarnemingsput	157469	473242	actief	2016-03-08 00:00:00
7	2018-02-26 09:47:38.8090…	2019-08-22 11:42:50.8550…	12494	32BM002	nan	nan	B32B0539	nan	nan	waarnemingsput	158338	473647	actief	1995-02-01 00:00:00
8	2014-03-27 12:00:00	2019-10-11 08:36:57.9420…	10637	11BL0061	nan	11BL0061	B11B0264	nan	nan	waarnemingsput	199950	564095	actief	1961-12-01 00:00:00
9	2016-09-21 08:37:11.1410…	2019-11-25 10:34:59.3790…	12373	21HM001	nan	nan	B21H0349	nan	nan	waarnemingsput	210813.6	504227	actief	2014-11-14 00:00:00
10	2014-03-27 12:00:00	2019-06-04 07:37:29.5220…	10967	34DP0259	nan	34DP0259	B34D0259	nan	nan	waarnemingsput	236858	458457.3	actief	2004-11-19 00:00:00
11	2014-03-27 12:00:00	2019-06-26 11:13:04.5020…	10130	33DP0017	nan	33DP0017	B33D0209	nan	nan	waarnemingsput	198003.5	459639	actief	NaT
12	2016-05-04 11:33:04.1020…	2019-12-11 12:38:14.0400…	12344	21BM008	nan	nan	nan	nan	nan	waarnemingsput	197659.9	520623.1	actief	2015-02-13 00:00:00
13	2014-03-27 12:00:00	2019-04-12 13:16:12.7990…	10170	33DP0165	nan	33DP0165	B33D0165	nan	nan	waarnemingsput	195400	458400	actief	1973-08-14 00:00:00
14	2014-03-27 12:00:00	2019-12-17 14:23:33.3550…	5347	MEWP005	nan	31GP0195	B31G0195	nan	nan	waarnemingsput	129882.6	455547.2	actief	1982-04-15 00:00:00
15	2014-03-27 12:00:00	2020-01-09 13:44:00.4540…	4381	26F0092	nan	26FP0092	B26F0092	nan	nan	waarnemingsput	178425.2	492643	actief	1980-02-28 00:00:00
16	2014-03-27 12:00:00	2019-06-04 07:37:29.7160…	10924	34DL0047	nan	34DL0047	B34D0356	nan	nan	waarnemingsput	236810	458140	actief	1958-12-15 00:00:00
17	2014-03-27 12:00:00	2018-12-28 08:56:25.8510…	10456	B33D0265	nan	33DL0037	B33D0265	nan	nan	waarnemingsput	198544.5	459659.3	actief	NaT
18	2014-03-27 12:00:00	2018-09-17 11:00:13.0240…	10433	B27D0338	nan	27DL0068	B27D0338	nan	nan	waarnemingsput	195297.9	476911.4	actief	NaT

Fase 1: workflow pythonscript

Stap 1. Importeren van Objectenbeheer: ‘Meetpunt’

Van de geïmporteerde 10.188 rijen, worden alle waarnemingsputten gefilterd.

→ Via kolom ‘meetpuntsoort’ kan worden gefilterd op ‘waarnemingsput’, ‘monsterpunt’, ‘oppervlakte_meetpunt’’, ‘pompput’, ‘zakbaken’ en ‘zoutwacht’.

→ Na filtering op ‘waarnemingsput’, vervallen 710 rijen.

→ Na filtering op ‘waarnemingsput’, blijven 9478 rijen met meetpunten over.

Fase 1: workflow pythonscript

Stap 1. Importeren van Objectenbeheer: ‘Maaiveldaanpassing’

Vanuit Objectenbeheer wordt het tabblad ‘Maaiveldaanpassing’ geïmporteerd.

Voor iedere waarnemingsput wordt vervolgens op dit tabblad de meest recente maaiveldaanpassing uitgelezen.

→ Deze maaiveldhoogtes staan bekend als de bovenkant van de buis van iedere waarnemingsput.

Fase 1: workflow pythonscript

Stap 1. Importeren van Objectenbeheer: ‘Instantie’

Belangrijk onderdeel van het meetdoel is de bijbehorende instantie die op het tabblad ‘Meetpunt’ worden weergegeven. Onderscheid wordt gemaakt in:

- Eigenaar meetpunt
- Opdrachtgevende instantie
- Beherende instantie
- Waarnemende instantie

Bijvoorbeeld: Een buis van de provincie, die door Vitens wordt beheerd.

Tabblad ‘Instantie’ wordt ingeladen om de nummering om te zetten naar namen

Fase 1: workflow pythonscript

Stap 1. Importeren van Objectenbeheer: ‘Filter’

Importeer tenslotte ook het tabblad ‘Filter’ vanuit Objectenbeheer.

Bevat informatie over onder andere:

- Filterid
- Diepte bovenkant (BK) filter
- Diepte onderkant (OK) filter

Fase 1: workflow pythonscript

Stap 1. Importeren van Objectenbeheer: Samenvoegen tabbladen

De informatie van de tabbladen Meetpunt en Filter wordt samengevoegd op basis van de meetpunt id’s.

Dit resulteert in een tabel van 18208 rijen, waarbij één meetpunt meerdere filters kan hebben.

De resulterende tabel wordt vervolgens gesorteerd op Meetpunt ID en Filternr.

Tenslotte worden de kolomnamen verandert naar herkenbare namen.

Output is een tabel genaamd Obj_Filters

Fase 1: workflow pythonscript

Stap 2. Data-cleaning van Objectenbeheer import

Tot nu toe zijn van alle meetpunten in Objectenbeheer (10.188 meetpunten), 710 meetpunten gefilterd. Hieruit volgden 9478 waarnemingsputten.

Na import van alle filters, volgden uit deze 9478 waarnemingsputten in totaal 18208 bijbehorende filters.

Uit een eerste check op de data van deze 18208 filters, volgt dat een groot gedeelte van de data incompleet is. Bijvoorbeeld door een missende maaiveldhoogte, onjuiste coördinaten en onrealistische filterstellingen (te diep, te ondiep, etc).

Tenslotte, kunnen ook alle meetpunten met een ‘vervallen’ status worden gefilterd.

Fase 1: workflow pythonscript

Stap 2. Data-cleaning van Objectenbeheer import

Allereerst worden alle meetpunten met een meetpuntstatus van ‘vervallen’ gefilterd, waarna ook de filters met foutieve metadata volgen.

Van de 18208 filters van waarnemingsputten uit Objectenbeheer worden 5859 filters gefilterd met foutieve metadata.

Dit resulteert in 12349 filters waarover de data-analyse kan worden uitgevoerd.

Toelichting per onderdeel in de volgende slides.

Fase 1: workflow pythonscript

Stap 2. Toelichting Data-cleaning van Objectenbeheer import

18208 waarnemingsputten

→ Totaallijst opgeslagen als *0_VitensFilters_Totaal.xlsx*

1. Selectie op vervallen Filters. 3187 vervallen filters.

→ Lijst opgeslagen als *1_VitensFilters_vervallen.xlsx*

→ 18208 – 3187 = **15021** overgebleven filters in Obj_Filters_clean

2. Selectie op verkeerde coördinaten (X<=100.000 & Y<=400.000). 265 filters met foutieve coördinaten uit de totale 18208.

→ Lijst opgeslagen als *2_Peilfilters_FoutieveCoördinaten.xlsx*

→ 15021 – 265 = **14756** overgebleven filters in Obj_Filters_clean

Fase 1: workflow pythonscript

Stap 2. Toelichting Data-cleaning van Objectenbeheer import

3. Filtering van verkeerde maaiveldhoogtes (<-5m+NAP, >100m+NAP en nan) uit de totale lijst van 18208. hieruit volgen 1273 filters.

→ Lijst opgeslagen als *3_Peilfilters_FoutiefMaaiveld.xlsx*

→ In de 1273 filters zitten duplicaten van vervallen buizen en coördinaten.

→ Uiteindelijk zijn er in Obj_Filters_clean 13973 overgebleven filters.

Fase 1: workflow pythonscript

Stap 2. Toelichting Data-cleaning van Objectenbeheer import

4. Filtering van foutieve filterdieptes. Uit de lijst van 18208 volgen 3054 filters met foutieve dieptes.

→ Lijst opgeslagen als 4_Peilfilters_FoutieveFilterDiepte.xlsx

→ Ontbrekende diepte = 92

→ Negatieve diepte = 509

→ Extreme diepte = 316

→ Foutieve filterlengte = 2219

→ Zeer ondiep = 349

→ Na verwijdering van de duplicaten van bovenstaande optelsom, komen we ook op 3054 filters met foutieve dieptes.

Fase 1: workflow pythonscript

Stap 2. Toelichting Data-cleaning van Objectenbeheer import

We zijn begonnen met 18208 filters.
Na cleaning zijn er 12349 filters overgebleven.

De optelsom van foutief gefilterd is 7779
→ Foutieve coördinaten: 265
→ Foutief maaiveld: 1273
→ Foutieve filterdiepte: 3054
→ Vervallen filters: 3187

Verwijderen van duplicaten, obv MP_id en Filt_id:
Obj_Filters_foutiefgefilterd.drop_duplicates(subset=[‘MP_id’, ‘Filt_id’]) → 5859 filters

De defintieve optelsom: 12349 + 5859 = 18208 filters. Check, de filtering klopt!!

Fase 1. workflow pythonscript

Stap 3. clippen van filters naar interesse-gebied.

Selecteert alle waarnemingsputten in een gegeven gebied.

Fase 1. workflow pythonscript

Stap 4. Toewijzen filter aan watervoerend pakket

Op basis van het de filterdiepte en het lagenmodel van het grondwatermodel, kunnen de filters worden toegewezen aan een watervoerend pakket.

Meetpunten met filters in meerdere watervoerende pakketten zullen meer unieke informatie bevatten ten opzichte van meetpunten met filters in het zelfde watervoerende pakket.

Werkwijze:

A. Inladen van lagenmodel
B. Pixellocatie bepalen in raster obv XY
C. Watervoerend pakket voor filter bepalen op pixellocatie
D. Correctie WVP voor freatisch pakket.

Fase 1. workflow pythonscript

Stap 4. Toewijzen filter aan watervoerend pakket

Stap A. Inladen van lagenmodel

Een functie is opgezet om voor een gegeven lagenmodel (AZURE voor UtrechtNoordHolland) met N modellagen (9 voor AZURE), de tops, bottoms, KD’s en C’s uit te lezen.

Rekening houdend voor de weerstand C, er N-1 lagen beschikbaar zijn, gezien het feit dat de weerstand van laag N, gelijk is aan de geohydrologische basis, waaraan geen waarden worden gehangen.

Tenslotte worden ook de X en Y coördinaten opgeslagen.

Fase 1. workflow pythonscript

Stap 4. Toewijzen filter aan watervoerend pakket

Stap B. Pixellocatie bepalen in raster obv XY

Voor het lagenmodel zijn ook de raster met X en Y coördinaten opgeslagen.

Voor een gegeven filter, kunnen op basis van zijn X en Y coördinaten, de locatie van dat filter binnen het raster van het lagenmodel worden bepaald. Dit is de pixellocatie van het filter in het lagenmodel.

Met de pixellocatie kunnen ter plekke van het gegeven filter, de tops, bottoms, KD’s en C’s van het lagenmodel worden uitgelezen.

Fase 1. workflow pythonscript

Stap 4. Toewijzen filter aan watervoerend pakket

Stap C. Watervoerend pakket voor filter bepalen op pixellocatie

Tenslotte, met de filterdieptes van het gegeven filter en de waarden voor tops, bottoms, KD’s en C’s van de lagen in het lagenmodel op de locatie van het filter, kan per filter een watervoerend pakket (WVP) worden toegewezen.

Op basis van de BK en OK van het filter, wordt een gemiddelde filterdiepte berekend, welke wordt gebruikt voor de bepaling van het WVP.

Fase 1. workflow pythonscript

Stap 4. Toewijzen filter aan watervoerend pakket

Er wordt door iedere laag van het lagenmodel geloopt.

Per laag wordt bekeken of filterdiepte tussen de top en bot van de betreffende laag zit. Zo ja, → bijbehorend WVP is gevonden!

Indien het filter boven het maaiveld steekt, (door inaccuraat lagenmodel), wordt deze aan het eerste WVP toegekend.

Indien het filter onder de diepste laag steekt, (door inaccuraat lagenmodel), wordt deze aan het diepste WVP toegekend.

Fase 1. workflow pythonscript

Stap 4. Toewijzen filter aan watervoerend pakket

Door het grove lagenmodel dat wordt gehanteerd, kan het ook voorkomen dat het filter (gedeeltelijk) overlapt met de slecht doorlatende lagen (SDL).

Vier typen:

- Filter in SDL overlapt met bovenliggende WVP.
- Filter in SDL overlapt met onderliggende WVP.
- Filter in SDL overlapt met zowel bovenliggende als onderliggende WVP.
- Filter zit volledig in de SDL.

Fase 1. workflow pythonscript

Stap 4. Toewijzen filter aan watervoerend pakket

Stel dat het filter in de SDL zit en alleen overlapt met het bovenliggende WVP, dan wordt deze in dat bovenliggende WVP toegekend.

Stel dat het filter in de SDL zit en alleen overlapt met het onderliggende WVP, dan wordt deze in dat onderliggende WVP toegekend.

Indien het filter met de SDL overlapt en zowel in het bovenliggende als onderliggende WVP zit, dan wordt het filter toegekend aan het WVP waarmee de grootste overlap zit. Dus het WVP waar het grootste stuk filter zit.

Indien het filter volledig in de slecht doorlatende laag zit, dan wordt het filter toegekend in het dichtst bijgelegen watervoerende pakket. Oftewel, het WVP dat de kleinste afstand heeft tot de BK en OK van het betreffende filter.

Fase 1. workflow pythonscript

Stap 4. Toewijzen filter aan watervoerend pakket

Tenslotte zijn alle filters toegewezen aan een watervoerend pakket. Aangezien er binnen de lagenmodellen wordt gewerkt met dummy-lagen, kan het voorkomen dat het filter is toegewezen aan het derde watervoerende pakket, maar dat er feitelijk geen weerstand boven het filter zit. (De kleilagen hebben allen de waarde 0, en zijn ter plekke van het filter dummy-lagen met dikte 0.01m bijvoorbeeld.)

In dat geval wordt het watervoerend pakket gecorrigeerd naar WVP=1. Dit is relevant om bijvoorbeeld later te bepalen of het om een droogteschadebuis gaat, die veelal freatisch zijn.

Als criterium voor de weerstand is een waarde van C=10dagen genomen.

Fase 1. workflow pythonscript

Stap 5. Uitlezen van landgebruik

Op basis van de landgebruikskaart LGN 7, wordt op de locatie van iedere waarnemingsput het bijbehorende landgebruik uitgelezen. Op basis hiervan kan een meetdoel worden bepaald. Bijvoorbeeld voor onderscheid landbouw of natuur droogteschade.

Het LGN7 heeft 19 verschillende codes, die als volgt te verdelen zijn:

- Landbouw = [1, 2, 3, 4, 5, 6, 7, 9, 10]
- Bebouwing = [8, 18]
- Natuur = [11, 12, 13, 14, 19]
- Overig = [0, 15, 16, 17]

Fase 1. workflow pythonscript

Stap 5. Uitlezen van landgebruik

Van ieder filter worden de XY-coördinaten gebruikt om de pixellocatie in het LGN7-raster te vinden.

Vervolgens wordt uit het LGN7 raster de landgebruik waarde op de pixellocatie voor het bijbehorende filter uitgelezen en wordt de gevonden code omgezet naar ‘landbouw’, ‘natuur’ of ‘bebouwing’.

Indien een code is gevonden die toebehoort aan ‘overig’, dan betekent dit dat het filter bijv in een cell staat met oppervlakte water. Dit is ongewenst.

Vanuit de pixellocatie wordt vervolgens iets uitgezoomd. In plaats van te kijken naar alleen de pixel op de locatie van het filter, worden de omliggende cellen gepakt, voor X-2 tot X+2 en Y-2 tot Y+2. Binnen dit gebiedje, wordt de landgebruik waarde gepakt die het meest voorkomend is. Deze waarde wordt toegewezen aan het filter.

Fase 1. workflow pythonscript

Stap 6. Uitlezen van effectcontouren winningen

In het HyKaWi zijn effectcontouren berekend van de grondwaterwinningen van Vitens, op basis van het verschil tussen ‘Winning aan’ en ‘Winning uit’.

Voor de contouren uit het bepompte pakket wordt bekeken welke waarnemingsputten binnen dit effectcontour vallen. Deze waarnemingsputten worden aan de betreffende winning toegewezen. Het is mogelijk dat één waarnemingsput binnen meerdere effectcontouren valt en dus aan meerdere winningen kan worden toegewezen. Dit kan dus een zeer nuttige peilbuis zijn!

Fase 1. workflow pythonscript

Stap 7. Inventarisatie van meetdoelen

Uiteindelijke inventarisatie van de meetdoelen vindt plaats op basis van vooraf opgelegde criteria, voor het landgebruik, filterstelling en dichtstbijzijnde winning.

De criteria zijn als volgt:

Meer dan één filter en binnen effect contour: Meetdoel is ‘Effect winning’

Minimaal één filter in 1e WVP, landgebruik is landbouw of natuur: Meetdoel is ‘Droogteschade’.

Minimaal één filter in 1e WVP, landgebruik is bebouwd: Meetdoel is ‘Zettingschade’

Fase 1. workflow pythonscript

Stap 8. Vaststellen huidige meetdoelen

De huidige meetdoelen kunnen worden vastgesteld op basis van informatie van de peilbuizen die momenteel bekend is, uit eerdere documentatie.

Denk aan commerciële meetnetten (bijvoorbeeld in Gelderland via Ton Ebbing).

Maar denk ook aan vergunningsplichtige peilbuizen. (Lijsten komen beschikbaar in 2020 per provincie, via John Bourgondiën. UtrechtNH is reeds beschikbaar)

Denk aan kwaliteitsmeetnet (via lijsten van Martin de Jonge)

De betreffende lijsten worden uitgelezen en eventuele huidige meetdoelen van de peilbuizen toegevoegd aan de kolom Meetdoel_Huidig.

Fase 1. workflow pythonscript

Stap 9. Peilbuizen + filterinformatie opslaan

De informatie van de peilfilters wordt samengevoegd per waarnemingsput.

Dit betekent dat alle filterinformatie, zoals het aantal filters, de filternummers, maar ook de watervoerende pakketten wordt samengevoegd tot één rij per peilbuis.

De uiteindelijke peilbuizen kunnen worden opgeslagen als een shapefile die kan worden ingeladen in Fase 2.

water
voor nu
en later

www.vitens.nl

Vitens