Inventarisatie mogelijke WP3 VRE Workflows

Maarten van Gompel, 27 Oktober 2019

Er zijn een aantal workflows voorgesteld voor de VRE. Ik weet niet precies wat de motivatie achter de samenstelling geweest is, maar ik vermoed om een eerste begin te maken en wat interoperabiliteit te testen. De workflows waren echter erg kort en ongespecificeerd geformuleerd, wat naar mijn indruk tot wat onduidelijk leidde voor de VRE ontwikkelaars. Ook traden er al gelijk wat andere problemen op. Het lijkt mij zinvol om een wat uitgebreidere inventarisatie van mogelijke concrete workflows en tools daarin uit te voeren en een wat breder beeld van mogelijke workflows die relevant voor WP3 en de VRE zouden kunnen zijn te vormen.

Ter overzicht, de volgende workflows waren voorgesteld:

- Word document => OpenConvert => FoLiA => Frog => FoLiA => AutoSearch upload => Queries in Autosearch
  - De OpenConvert naar FoLiA converter is redelijk out of date en men liep tegen een bug, ik heb met Piereling een alternatieve webservice gelanceerd om dit op te lossen, aangezien we inmiddels nieuwere convertors hebben.
- HTML document => OpenConvert => FoLiA => Nijmegen Alpino => uploaden in PaQu / SPOD
  - Deze kan überhaupt niet want er is geen FoLiA input laag voor onze Alpino webservice (wel FoLiA output).
  - Kan ik desgewenst best implementeren uiteraard
- Word Doc => OpenConvert =>plain text => upload in PaQu => queries in PaQu
- TIFF doc (scan) => PICCL met pos-tagging etc => FoLiA => AutoSearch upload => Queries in Autosearch
- CHA corpus => upload in GrETEL 4 => Queries in GrETEL 4
- TEI corpus => upload in AutoSearch => Queries in Autosearch
- TEI corpus => upload in GrETEL 4 => Queries in GRETEL 4
- CHA corpus => upload in GrETEL 4 => Generate %MOR, %GRA => new version of CHA corpus with %MOR and % GRA tiers (not possibe yet)

Ik zie een aantal tools meermaals voorkomen en een aantal grote WP3 tools zoals FLAT en ucto überhaupt niet. Ook zie ik geen tools van de VU terwijl die ook hele pipelines hebben ontwikkeld (al dan niet buiten WP3, maar toch relevant lijkt me).

In dit overzicht kijk ik voornamelijk vanuit het perspectief van de beschikbare tools: welke tools zijn er beschikbaar, werkt naar behoren, en is daadwerkelijk in gebruik. Uiteraard is mijn blik maar beperkt, en reikt het in eerste instantie over de software waar ik zelf bij betrokken ben, dus aanvullingen en commentaar is zeer welkom. Het doel van dit document is dan ook om een discussie op gang te brengen.

Het lijkt me ook zeer relevant om echte concrete gebruikers (en mogelijke testers) van de te implementeren workflow bij elk scenario te hebben, maar daar heb ik zelf niet echt zicht op.

Webservice workflows

Webservice Workflow 1: Automatische linguïstische verrijking van een Nederlandse tekst en doorzoeken in Autosearch

- **Use cases**:
  - Een gebruiker wil automatisch linguïstische verrijkingen aanbrengen op een Nederlandse tekst en hier vervolgens in kunnen zoeken.
- **Data- & Transformatie pipeline**: (n.b: tools/services zijn vetgedrukt, data niet)
  - Word (docx) document met Nederlandse tekst
  - **Piereling** service voor conversies van en naar FoLiA)
    - Webservice (CLAM): https://webservices-lst.science.ru.nl/piereling
    - REST API specificatie: https://webservices-lst.science.ru.nl/piereling/info
    - HTTP Basic Authenticatie, geen CLARIN federatie, eigen user database
  - FoLiA document, ongetokeniseerd met structuurinformatie (paragraaf, lijsten) etc.
  - **Frog** service voor linguistische verrijking
    - Webservice (CLAM): https://webservices-lst.science.ru.nl/frog
    - REST API specificatie: https://webservices-lst.science.ru.nl/frog/info
    - HTTP Basic Authenticatie, geen CLARIN federatie, eigen user database
    - Bron: https://github.com/LanguageMachines/frog

  - getokeniseerde FoLiA met (naar parameterisering): PoS (CGN tagset), lemma, named entities, shallow parsing, dependency parsing (Alpino tags)
  - **Autosearch** via de upload webservice
    * Federated CLARIN authenticatie
    * Eindresultaat:
      · Gebruiker kan nu queries doen op het document en de lingüistische annotaties (gelimiteerd tot alleen PoS, lemma)

Webservice Workflow 2: Automatische linguïstische verrijking van een Nederlandse tekst en visualiseren/bewerken/doorzoeken in FLAT

- **Use cases**:
  - Een gebruiker wil zijn verrijkte document simpelweg visualiseren (het document met alle linguïstische annotaties zien)
  - Een gebruiker wil het geannoteerde documenten bewerken/corrigeren/aanvullen.
  - Een gebruiker wil automatisch linguïstische verrijkingen aanbrengen op een Nederlandse tekst en hier vervolgens in kunnen zoeken. (hetzelfde als pipeline 1, maar met wat vergaandere zoekmogelijkheden maar minder grafische query interface)
- **Data- & Transformatie pipeline**: (n.b: tools/services zijn vetgedrukt, data niet)
  - OpenOffice (odt) document met Nederlandse tekst (een ander formaat als kleine variatie op pipeline 1)
  - **Piereling** (webservice voor conversies van en naar FoLiA)
  - FoLiA document, ongetokeniseerd, met structuurinformatie (paragraaf, lijsten) etc.
  - **Frog** (Linguistische verrijkingstool voor het Nederlands)
  - FoLiA document, getokeniseerde, met (naar parameterisering) PoS (CGN tagset), lemma, named entities, shallow parsing, dependency parsing (Alpino tags)
  - **FLAT** via de public upload webservice
    * Webapp: https://flat.science.ru.nl
    * Upload documentatie: https://flat.readthedocs.io/en/latest/administration_guide.html#public-anonymous-upload-for-third-party-applications
    * Eindresultaat:
      · Gebruiker krijgt een interactieve visualisatie van het document en al haar annotaties
      · Gebruiker kan de annotaties bewerken of nieuwe annotaties toevoegen
      · Gebruiker kan queries doen op het document en alle lingüistische annotaties (middels CQL of FQL)

Webservice Workflow 3: Automatische linguïstische verrijking van een Nederlandse corpus en doorzoeken in Autosearch

- **Use cases**:
  - Een gebruiker wil automatisch linguïstische verrijkingen aanbrengen op een Nederlands corpus (meerdere teksten) en hier vervolgens in kunnen zoeken.
- **Motivatie:**
  - Dit is een variant van pipeline 1, maar expliciet gebruik makend van meerdere teksten, waar AutoSearch juist geschikt voor is (FLAT daarentegen is in dit opzicht gelimiteerd tot één document)
- **Data- & Transformatie pipeline**: (n.b: tools/services zijn vetgedrukt, data niet)
  - Word (docx) document met Nederlandse tekst
  - **Piereling** service voor conversies van en naar FoLiA)
  - Zip archief van FoLiA documenten, ongetokeniseerd met structuurinformatie (paragraaf, lijsten)
  - **Frog** service voor linguistische verrijking
  - Zip archief van getokeniseerde FoLiA met (naar parameterisering): PoS (CGN tagset), lemma, named entities, shallow parsing, dependency parsing (Alpino tags)
  - **Autosearch** via de upload webservice
    * Federated CLARIN authenticatie
    * Eindresultaat:
      · Gebruiker kan nu queries doen op het document en de lingüistische annotaties (gelimiteerd tot alleen PoS, lemma)

Webservice Workflow 4: Alpino: Automatische syntactische verrijking van een Nederlandse tekst en visualisatie in PaQU

- **Use Cases**:
  - Gebruiker wil de syntactische (dependency/constituency) structuur visualiseren en erin zoeken
- **Data- & Transformatie pipeline:**
  - HTML document met Nederlandse tekst
    - Opmerking: Ik vind HTML een wat ongelukkige keuze, de meeste web-html zal dermate vervuild zijn dat het niet goed als input bruikbaar is, dus of het een realistisch scenario is vraag ik me af, maar ik hanteer het even omdat het in het oorspronkelijke lijstje stond.
  - **OpenConvert** service (webservice voor conversie)
  - plain-text, UTF-8
    - Opmerking: Het oorspronkelijke had FoLiA invoer maar dat kan in dit stadium als Alpino (service) invoer dus niet (niet geïmplementeerd)
  - **Alpino** service
    - Webservice (CLAM): https://webservices-stl.science.ru.nl/alpino
    - REST API specificatie: https://webservices-lst.science.ru.nl/alpino/info
    - HTTP Basic Authenticatie, geen CLARIN federatie, eigen user database
    - Bron: https://github.com/rug-compling/alpino
  - Zip archief met Alpino XML (één bestand per zin)
  - **PaQu**
    - https://paqu.let.rug.nl:8068/
      - Groningen single-sign on (oauth?)
      - Google (oauth)
      - eigen authenticatie (auto request pw via mail)
      - (ik kan zo snel geen documenttie over API endpoints voor uploads vinden)
    - Eindresultaat:
      - De gebruiker kan nu de boomstructuren visualiseren en erdoor zoeken

Webservice Workflow 5: PICCL: OCR, Tekstnormalisatie en Linguistische verrijking

- **Use cases:**
  - Gebruiker wil een gescande tekst digitaliseren
- **Data- & Transformatie pipeline:**
  - TIFF document (gescande tekst)
  - **PICCL service**
    - Met OCR
    - Met TICCL (tekstnormalisatie)
    - Met Frog (linguïstische verrijking)
    - Webservice (CLAM): https://webservices-lst.science.ru.nl/piccl
    - REST API specificatie: https://webservices-lst.science.ru.nl/piccl/info
    - HTTP Basic Authenticatie, geen CLARIN federatie, eigen user database
  - FoLiA document, met eventueel naar parameterisering dubbele tekstlagen en string-annotaties (ticcl), tokenisatie (ucto), verdere verrijking (frog), of Zip archief van meerdere zulke FoLiA documenten
- **Eventuele vervolgstappen:**
  - (5a) Autosearch zoals in pipeline 1 (enkel document) of 4 (meerdere documenten)
  - (5b) FLAT zoals in pipeline 2 (enkel document), FLAT kan tot op zekere hoogte (beetje buggy nog) de verschillende tekstlagen (genormaliseerd vs niet genormaliseerd) visualiseren.
- **Eventuele varianten:**
  - PDF (met scan) input ipv TIFF

Minimale workflows

Een aantal van de gesuggereerde workflows waren minimaal, in de zin dat er maar één tool/schakel bij betrokken is. Op zich zijn dat juist goede bouwblokken voor subworkflows en kunnen er veel workflows van die soort opgesteld worden. Ik begin even met degenen die oorspronkelijk gegeven waren:

Minimale Webservice Workflow 6: GreTeL: TEI corpus upload en zoeken in treebanks

- **Data- & Transformatie pipeline**:
  - TEI documenten
    - Opmerking: Dit zou wat nader gespecificeerd moeten worden want TEI is enorm breed en kent vele vormen. Ik weet niet wat voor vorm van TEI GreTeL kan hanteren
  - **GreTeL** service
    - http://gretel.hum.uu.nl/gretel4/ng/home
    - Eindresultaat:
      - Gebruiker kan in de treebank zoeken en de bomen visualiseren
- **Varianten:**
  - CHILDES input ipv TEI input.

Minimimale Webservice Workflow 7: Autosearch

- **Data- & Transformatie pipeline**:
  - TEI documenten
    - Opmerking: Zie workflow 6
  - **AutoSearch** service
    - Eindresultaat:
      - Gebruiker kan in het corpus zoeken
- **Varianten:**
  - FoLiA input ipv TEI input

Directe Workflows

Alle bovenstaande workflows zijn webservice georiënteerd, dit geeft de nodige overhead (veel netwerk overhead bij elke schakel). Bij het verwerken van grote hoeveelheden data (denk: grote corpora) kan dit al snel een bottleneck worden. Het lijkt me daarom relevant om ook directere workflows te ontwikkelen, die lokaal of gedistribueerd over een rekencluster gedraaid kunnen worden. Hier komen dan deployment oplossingen bij kijken zoals in het originele VRE plan (waar LaMachine deels in kan voorzien).

Zelf heb ik al een aantal minimale directe workflows geimplementeerd rondom bv ucto (tokeniser) en Frog. Dit heb ik gedaan met Nextflow (3rd party) en deze workflows maken deel uit van aNtiLoPe. Voordeel is dat deze gelijk over een heel rekencluster (eventueel nog met tussenkomst van iets als SGE of SLURM) geparalelliseerd gedraaid kunnen worden. Ik geloof dat anderen zoals bv bij eScience met vergelijkbare dingen bezig zijn geweest voor andere pipelines (met de VU Newsreader geloof ik).

Directe Workflow 1: Automatische lingüistische verrijking van een Nederlands corpus (geen verdere nabewerking)

Dit is een variant op pipeline 1 of 3 die geheel lokaal gedraaid wordt zonder webservices en waarbij ook geen verdere nabewerking zit.

- **Use Cases**:
  - Gebruiker wil corpus verrijken op een eigen rekencluster met minime overhead
- **Data- & Transformatie pipeline**: (n.b: tools/services zijn vetgedrukt, data niet)
  - Word (docx) document met Nederlandse tekst
  - **pandoc** voor documentconversie
  - ReStructuredText
  - **rst2folia** uit **FoLiA-Tools** voor conversie
    - https://github.com/proycon/foliatools
  - FoLiA document, ongetokeniseerd, met structuurinformatie
  - **Frog**
    - https://github.com/LanguageMachines/frog
  - FoLiA document, getokeniseerd en verrijkt

Suggesties voor andere relevante (minimale) workflows

Ik speculeer even kort in willekeurige volgorde op andere mogelijkheden die relevant zijn voor WP3, al zijn ze niet allemaal officieel binnen CLARIAH WP3 ontwikkeld (dat vind ik niet zo relevant), maar het zijn wel componenten die

binnen het de WP3 scope passen, compatible zijn, en zo ingeplugd kunnen worden. Dit zijn minimale pipelines met maar één schakel:

Ucto

- **Use Cases**:
  - Een gebruiker heeft ongetokeniseerde tekst en wil dit tokeniseren
- **Data- & Transformatie pipeline**:
  - Plaintext of FoLiA
  - **Ucto** service: spellingscorrectie voor Nederlands
    - Multilingual, specifieke regelsets voor een aantal Europese talen. En een generieke set.
    - Webservice (CLAM): https://webservices-lst.science.ru.nl/ucto
    - REST API specificatie: https://webservices-lst.science.ru.nl/ucto/info
    - HTTP Basic Authenticatie, geen CLARIN federatie, eigen user database
    - Ucto is ook geïntegreerd in Frog, dus als je Nederlandse lingüistische verrijking wil doen kan je gelijk Frog pakken.
  - plaintext of FoLiA, getokeniseerd
- **Eventuele vervolgstappen**:
  - Hier zijn eigenlijk heel veel mogelijkheden

Valkuil

- **Use Cases**:
  - Een gebruiker heeft een Nederlandse tekst en dit checken of spellingsfouten/grammaticafouten .
- **Data- & Transformatie pipeline**:
  - Plaintext of FoLiA
  - **Valkuil** service: spellingscorrectie voor Nederlands
    - Webservice (CLAM): https://webservices-lst.science.ru.nl/valkuil
    - REST API specificatie: https://webservices-lst.science.ru.nl/valkuil/info
    - HTTP Basic Authenticatie, geen CLARIN federatie, eigen user database
    - Backend aangedreven door gecco (https://webservices-lst.science.ru.nl/gecco)
  - FoLiA, getokeniseerd met suggesties voor correctie
- **Eventuele vervolgstappen**:
  - Visualisatie van de fouten en correcties in FLAT (dit is ook wat onze valkuil.net website doet)

WikiEnte

- **Use Cases**:
  - Een gebruiker heeft een tekst en wil hier namen in herkennen en deze gelinked hebben naar Wikipedia/DBPedia
- **Data- & Transformatie pipeline**:
  - Plaintext of FoLiA
  - **WikiEnte** named entity recognition & entity linking
    - Multilingual, backend is DBPedia Spotlight (3rd party)
    - Nog geen webservice, kan op verzoek zeer snel gerealiseerd worden.
    - https://github.com/proycon/wikiente
  - FoLiA met named entities en links naar DBPedia/WikiPedia
- **Eventuele vervolgstappen**:
  - Visualisatie van de named entities en links in FLAT

Oersetter

- **Use Cases**:
  - Een gebruiker heeft Nederlandse tekst die hij/zij naar het Fries wil vertalen, of vice versa.
- **Data- & Transformatie pipeline**:
  - Plaintext in het Nederlands of in het Fries
  - **Oersetter** Nederlands-Friese MT
    - Multilingual, backend is DBPedia Spotlight (3rd party)
    - Webservice (CLAM): https://webservices-lst.science.ru.nl/oersetter

        * REST API specificatie: https://webservices-lst.science.ru.nl/oersetter/info
        * HTTP Basic Authenticatie, geen CLARIN federatie, eigen user database
        * Web front-end: https://taalweb.frl/oersetter
        * In samenwerking met Fryske Akademy
    - Plaintext in het Nederlands of in het Fries (tegensteld aan de input)
        * met hier en daar een tag voor unknown words
- **Eventuele vervolgstappen**:
    - Voor Nederlandse uitvoer; verdere werking met bv. Frog

SpaCy

- **Use Cases**:
    - Een gebruiker heeft een tekst en wil hier lingüistische verrijkingen op aanbrengen
- **Data- & Transformatie pipeline**:
    - Plaintext of FoLiA
    - **spacy2folia**: Linguistische verrijking (tokenisatie, named entity recognition, PoS, dependency parsing, shallow parsing).
        * Multilingual, backend is spaCy (3rd party) met slechts een kleine wrapper voor FoLiA invoer/uitvoer
        * Nog geen webservice, kan op verzoek zeer snel gerealiseerd worden.
    - FoLiA, getokeniseerd, met naar parameterisering: named entities, PoS, dependency parsing, shallow parsing. Tagset verschilt per taal en is bij Nederlands anders dan bij Frog of Alpino!
- **Eventuele vervolgstappen**:
    - Visualisatie en eventuele verdere manuele bewerking in FLAT
    - Zoeken in Autosearch
    - Ik weet niet of PaQu overweg kan met niet-Alpino tagsets, maar anders is dat ook een mogelijkheid voor het visualiseren en doorzoeken van de syntactische structuur.

De VU heeft een vergelijkbare wrapper voor SpaCy naar NAF (https://github.com/cltl/SpaCy-to-NAF)

Colibri Core

- Plaintext of FoLiA (maar alleen tekst, geen annotaties)
- **colibri-core**: N-gram en skip-gram extractie met frequentieinformatie en meer
    - Multilingual, backend is spaCy met slechts een kleiner wrappre voor FoLiA invoer/uitvoer
    - Webservice (CLAM): https://webservices-lst.science.ru.nl/colibricore
    - REST API specificatie: https://webservices-lst.science.ru.nl/colibricore/info
    - HTTP Basic Authenticatie, geen CLARIN federatie, eigen user database
    - Bron: https://github.com/proycon/colibri-core
- FoLiA met allerlei lingüistische verrijkingen

T-Scan

- Plaintext of FoLiA (maar alleen tekst, geen annotaties)
- **T-scan**: Berekend allerlei tekstmetrieken (voor leesbaarheidspredictie)
    - Multilingual, backend is spaCy met slechts een kleiner wrappre voor FoLiA invoer/uitvoer
    - Webservice (CLAM): https://webservices-lst.science.ru.nl/tscan
    - REST API specificatie: https://webservices-lst.science.ru.nl/tscan/info
    - HTTP Basic Authenticatie, geen CLARIN federatie, eigen user database
    - Bron: https://github.com/proycon/tscan
    - Sinds geruime tijd ontwikkeld aan en onder beheer van Universiteit Utrecht (Nijmegen is nog slechts een hoster/distributeur)
- FoLiA met tekstmetrieken

Colibri Lang / FoLiA-textcat / folialangid

We hebben drie verschillende tools die taalidentificatie kunnen doen

- Plaintext of FoLiA (maar alleen tekst, geen annotaties)
- **colibri-lang / FoLiA-textcat / folialangid**: Taalidentificatie
    - ondersteuning voor oud-Nederlands

  - Nog geen webservice (kan op verzoek)
- FoLiA met taalinformatie

Nederlab Enrichment Pipeline

- FoLiA of TEI P5/Lite XML zoals opgeleverd voor de DBNL collectie door de KB, TEI XML voor andere collecties zoals opgeleverd door INT
- **Nederlab Pipeline**: Verrijking van historisch Nederlands
  - Dit is an sich dus al een grote pipeline, het combineert tools die al elders langsgekomen zijn als tei2folia, Ucto, Frog, WikiEnte, Colibri-Lang
  - Nog geen webservice (kan op verzoek)
- FoLiA

Engelse Spraakherkenning

- Audio in WAV, MP3 of OGG vorm
- **eng_ASR** webservice
  - Webservice (CLAM): https://webservices-lst.science.ru.nl/eng_ASR/info/
  - RESTUL specificatie: https://webservices-lst.science.ru.nl/eng_ASR/info/
  - HTTP Basic Authenticatie, geen CLARIN federatie, eigen user database
  - Bron: https://github.com/schemreier/eng_ASR
- Plain-text (utf-8) transcriptie van de tekst

Automatic Transcript of Oral History Interviews (Nederlandse Spraakherkenning)

- Audio in WAV, MP3 of OGG vorm
- **Automatic Transcript of Oral History Interviews (Nederlandse Spraakherkenning)** webservice
  - Webservice (CLAM): https://webservices-lst.science.ru.nl/oral_history/info/
  - RESTUL specificatie: https://webservices-lst.science.ru.nl/oral_history/info/
  - HTTP Basic Authenticatie, geen CLARIN federatie, eigen user database
  - Bron: https://github.com/schemreier/oral_history
- Plain-text (utf-8) transcriptie van de tekst of AudioDoc XML transcriptie van de tekst

Fries-Nederlandse Spraakherkenning

- Audio in WAV, MP3 of OGG vorm
- **fy_NL_ASR** webservice: Detecteert zowel Nederlands als Fries (code switching), ontwikkeld in het FAME project.
  - Webservice (CLAM): https://webservices-lst.science.ru.nl/fy_nl_ASR/info/
  - RESTUL specificatie: https://webservices-lst.science.ru.nl/fy_nl_ASR/info/
  - HTTP Basic Authenticatie, geen CLARIN federatie, eigen user database
  - Bron: https://github.com/schemreier/fy_nl_ASR
- Plain-text (utf-8) transcriptie van de tekst

Mogelijkheden met software van andere CLARIAH partners

De bovenstaande lijst van suggesties is uiteraard niet volledig en beperkt zich tot de dingen waar ik zicht op heb en/of bij betrokken ben geweest. Wat betreft andere partners liggen er denk ik ook nog interessante mogelijkheden:

Vrije Universiteit Amsterdam, CLTL?

Wat betreft interoperabiliteit met de VU pipelines (o.a. NewsReader) en onze Nijmeegse pipelines zijn er gesprekken en initiatieven geweest, zo zijn we ook aan een naf2folia/folia2naf conversie begonnen die daarin een sleutelrol zou moeten vervullen door onze respectievelijke dataformaten te converteren. Maar helaas is dit door tijdsgebrek en gebrek aan mankracht nooit genoeg van de grond gekomen om een echt werkbare interoperabiliteit op te leveren.

Naast de Newsreader pipeline zijn andere met name interessante tools voor de WP3 VRE (allen nog niet als webservice beschikbaar voor zover ik weet):

- Entity Detection for Historical Dutch: https://github.com/cltl/entity-detection-for-historical-dutch (CLARIAH-PLUS project)

- Word Sense Disambiguation: https://github.com/cltl/BERT-WSD
- SpaCy-to-NAF: https://github.com/cltl/SpaCy-to-NAF (analoog aan spacy2folia)
- En volgens mij is er nog veel meer!

INT

Er zijn vast interesssante mogelijkheden met Blacklab en de nieuwe frontend daarvoor. Ik weet ook niet hoe zich dat precies tot AutoSearch verhoudt.

Meertens

Hier denk ik aan eerste instantie aan koppelingen met MTAS, om corpora doorzoekbaar te maken.

Mogelijkheden met andere CLARIN partners?

Als we buiten Nederland kijken zijn er ook interessante mogelijkheden, bijvoorbeeld tot interoperabiliteit met het Duitse Weblicht (waar dan als sleutelcomponent een TCF-FoLiA/FoLiA-TCF converter ontwikkeld zou moeten worden).

Verdere is er een zekere overlap tussen een deel van de VRE en de activiteiten van het Switchboard.