MARCELL

Agreement number: INEA/CEF/ICT/A2017/1565710
Action No: 2017-EU-IA-0136

CEF
programme

Deliverable 4.2
Document flow pipeline

Version No. 1.0
2020-04-31

Document Information

Activity:	Activity 4: Sustainability
Deliverable number:	D4.2
Deliverable title:	Document flow pipeline
Indicative submission date:	2020-04-31
Actual submission date of deliverable:	2020-05-01
Main Author(s):	Vasile Păiș
Participants:	Tamás Váradi, Tinko Tinchev, Nikola Obreshkov, Martin Yalamov, Marko Tadić, Bálint Sass, Bartłomiej Nitoń, Maciej Ogrodniczuk, Piotr Pęzik, Verginica Barbu Mititelu, Radu Ion, Elena Irimia, Maria Mitrofan, Dan Tufiș, Radovan Garabík, Simon Krek, Andraž Repar, Matjaž Rihtar
Version:	V1.0

History of Versions

Version	Date	Status	Name of the Author (Partner)	Contributions	Description/ Approval Level
V1.0	2020-04-31	Completed	Vasile Păiș	Report compilation	

EXECUTIVE SUMMARY

This report presents the technical details associated with the document flow pipeline developed to ensure the project's sustainability. The objective of this Activity is to build a single access point service that will integrate language specific processing pipelines. The proposed platform is based on the concept of containerization in order to ensure time-persistence of the language specific implementations against the OS updates and other changes between hosts and environments.

Contents

1. Purpose

This technical specification allows the implementation of the MARCELL sustainability objective as described in the project proposal: *"The activity aims to build a single access point service that will integrate the seven language specific processing pipelines. The final version of the processing flow pipeline will be a single access point to a web service active 24/7 on a dedicated virtual server. The incoming new documents will be language identified and directed to the language specific processing flow. The processing result for the new incoming documents will be automatically forwarded."*

2. General approach

There will be 8 Docker containers:

- 1 Docker for a web based GUI
- 7 Dockers corresponding to the 7 language specific processing flows: Croatian, Hungarian, Romanian, Bulgarian, Slovak, Slovenian, Polish

The Dockers will communicate via an internal local network and only the GUI will be exposed on the Internet. Therefore no additional precautions such as HTTPS are considered to be relevant between the Dockers.

The operating system host for the Dockers will allow mapping of a storage volume to the GUI Docker for storing the project’s data.

Since every language specific pipeline has its own requirements, the GUI is configurable allowing different communication mechanisms with each of the Dockers.

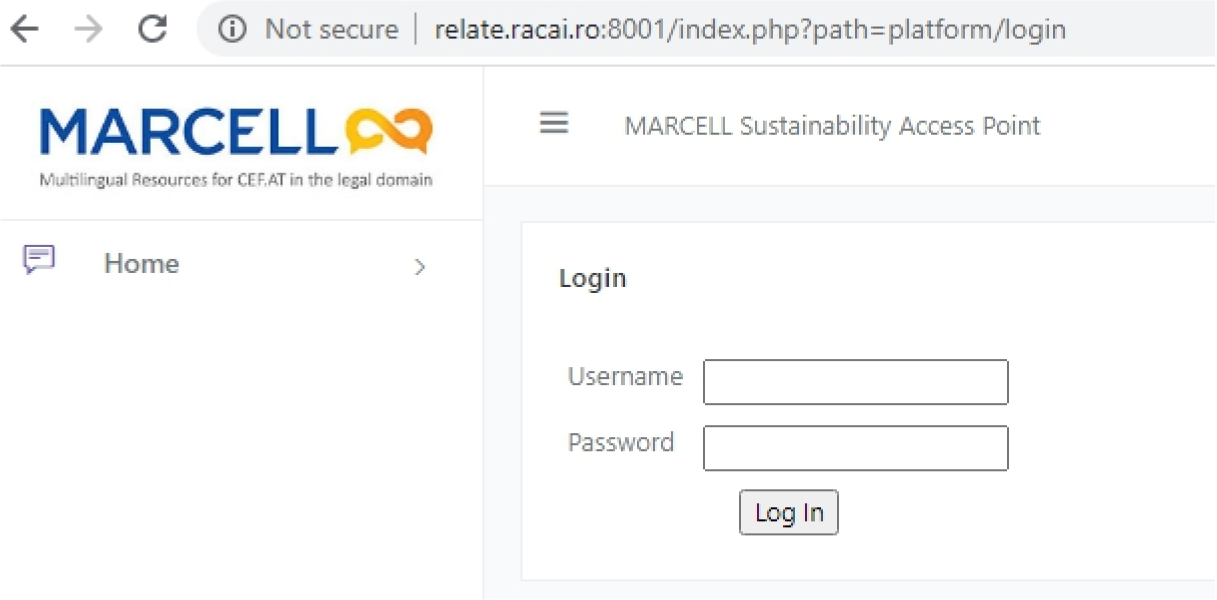
3. GUI

The GUI is the *single access point service* used to communicate with the project’s components. It allows for:

- user authentication and authorization
- management of language specific corpora
- upload of ZIP files with RAW text and standoff metadata
- identification of newly added files (in case of a ZIP file is uploaded with files which were already annotated)
- annotation via language specific pipelines
- creation of ZIP files with annotated files in Marcell’s CONLLUP format
- download of ZIP files

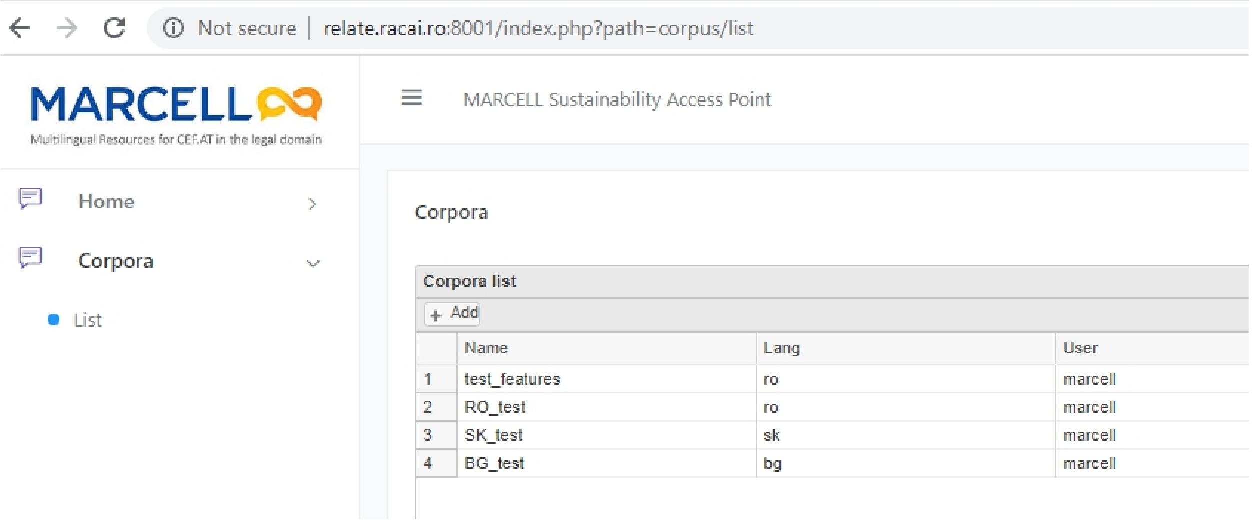
Example usage scenario:

a) Login into the GUI



Not secure
relate.racai.ro:8001/index.php?path=platform/login
MARCELL
Multilingual Resources for CEF.AT in the legal domain
MARCELL Sustainability Access Point
Home
Login
Username
Password
Log In

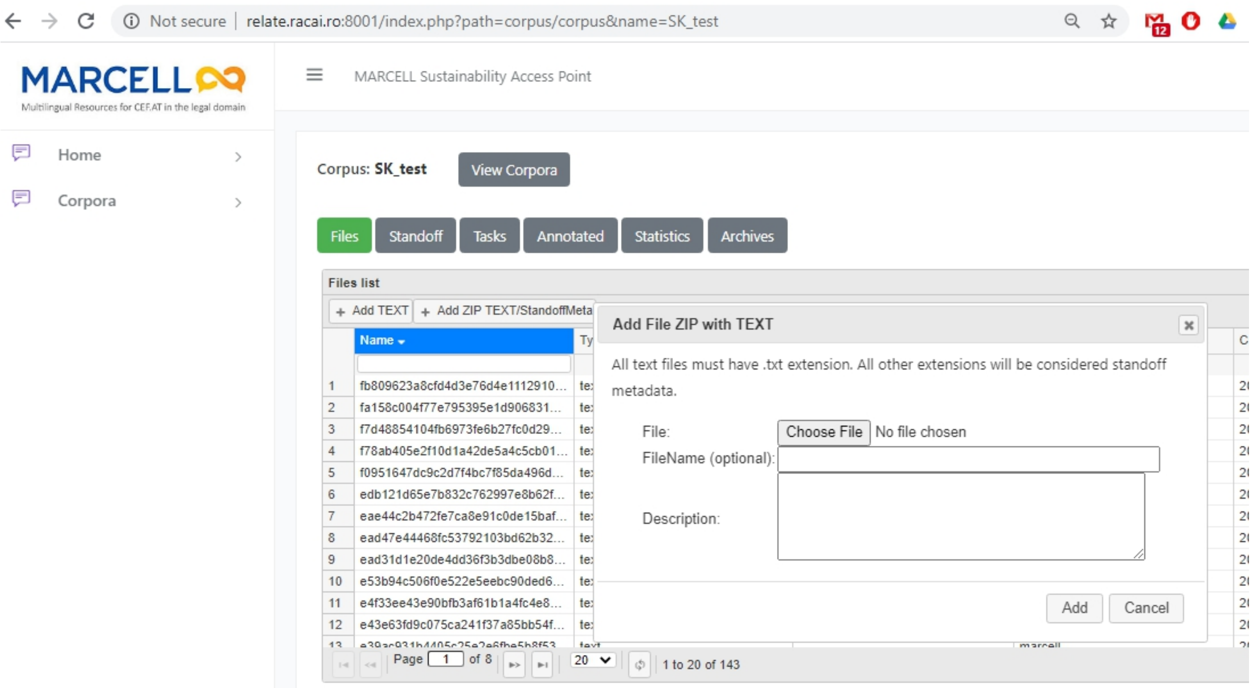
b) Access the desired corpus (or create one if needed, using the "Add" button); selecting an existing corpus is realized using "double click" with the mouse, or double tap on a touch screen.



Not secure
relate.racai.ro:8001/index.php?path=corpus/list
MARCELL
Multilingual Resources for CEF.AT in the legal domain
MARCELL Sustainability Access Point
Home
Corpora
List
Corpora
Corpora list
+ Add

	Name	Lang	User
1	test_features	ro	marcell
2	RO_test	ro	marcell
3	SK_test	sk	marcell
4	BG_test	bg	marcell

c) Upload a ZIP file: From the corpus page, in the "Files" tab (this is the default tab), press the "Add ZIP TEXT/StandoffMeta" button. The only required field is the ZIP archive. "Filename" and "Description" are optional parameters which can be displayed in the GUI if completed.



relate.racai.ro:8001/index.php?path=corpus/corpus&name=SK_test

MARCELL Sustainability Access Point

Corpus: **SK_test**

Files list

	Name	Ty
1	fb809623a8cfd4d3e76d4e1112910…	te
2	fa158c004f77e795395e1d906831…	te
3	f7d48854104fb6973fe6b27fc0d29…	te
4	f78ab405e2f10d1a42de5a4c5cb01…	te
5	f0951647dc9c2d7f4bc7f85da496d…	te
6	edb121d65e7b832c762997e8b62f…	te
7	eae44c2b472fe7ca8e91c0de15baf…	te
8	ead47e44468fc53792103bd62b32…	te
9	ead31d1e20de4dd36f3b3dbe08b8…	te
10	e53b94c506f0e522e5eebc90ded6…	te
11	e4f33ee43e90bfb3af61b1a4fc4e8…	te
12	e43e63fd9c075ca241f37a85bb54f…	te

Page 1 of 8 20 1 to 20 of 143

Add File ZIP with TEXT

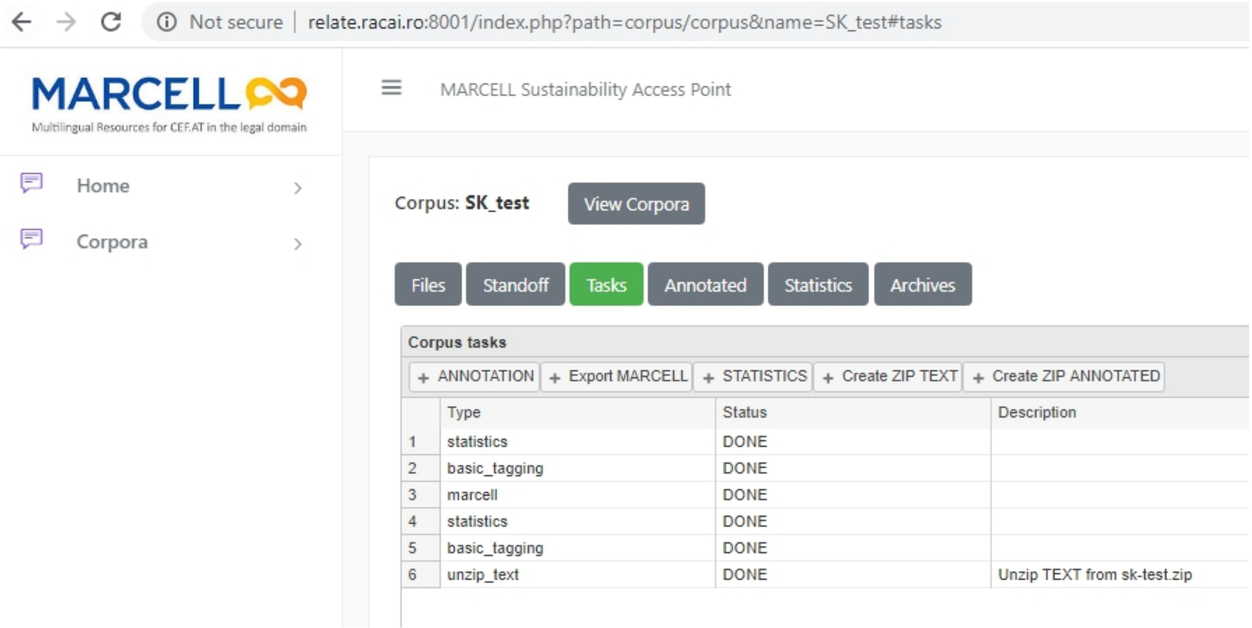
All text files must have .txt extension. All other extensions will be considered standoff metadata.

File: Choose File No file chosen

FileName (optional):

Description:

d) Tasks: Once uploaded an “unzip_text” task is automatically created. This will extract all TEXT + METADATA files in the platform’s folders. The status of this Task can be checked in the “Tasks” tab. Additionally, in this tab new tasks can be created: “Annotation” will call the appropriate Docker for annotating new files; “Export Marcell” will create ZIP archives with Marcell files.



relate.racai.ro:8001/index.php?path=corpus/corpus&name=SK_test#tasks

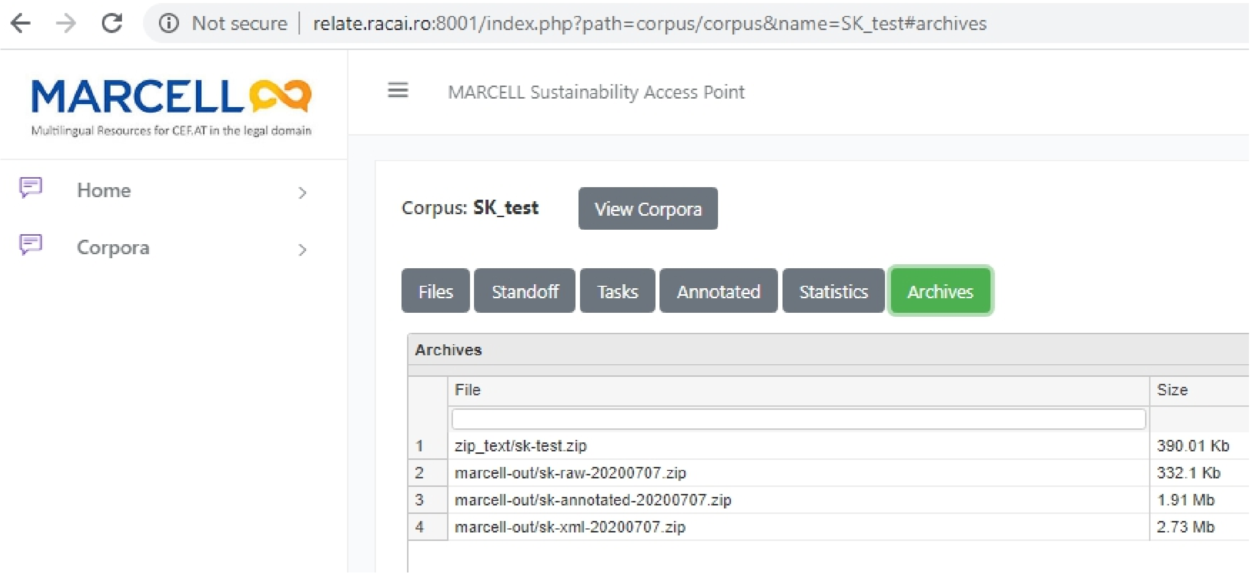
MARCELL Sustainability Access Point

Corpus: **SK_test**

Corpus tasks

	Type	Status	Description
1	statistics	DONE	
2	basic_tagging	DONE	
3	marcell	DONE	
4	statistics	DONE	
5	basic_tagging	DONE	
6	unzip_text	DONE	Unzip TEXT from sk-test.zip

e) Download Marcell annotated corpus: After the “Annotation” and “Export Marcell” tasks are finished, the resulting archives are present in the “Archives” tab. These can be downloaded using “double click” with a mouse or “double tap” on a touch screen.


Not secure | relate.racai.ro:8001/index.php?path=corpus/corpus&name=SK_test#archives
MARCELL
Multilingual Resources for CEF.AT in the legal domain
MARCELL Sustainability Access Point
Home
Corpora
View Corpora
Files
Standoff
Tasks
Annotated
Statistics
Archives

Corpus: SK_test

Archives

	File	Size
1	zip_text/sk-test.zip	390.01 Kb
2	marcell-out/sk-raw-20200707.zip	332.1 Kb
3	marcell-out/sk-annotated-20200707.zip	1.91 Mb
4	marcell-out/sk-xml-20200707.zip	2.73 Mb

4. Platform API

Additionally to the user interface functionality, the Access Point offers an API for uploading new files. The usage of this API is not required, since archives can be manually uploaded in the GUI itself via the “Add ZIP Text/StandoffMeta” button.

Example CURL call:

```
curl \
    -F 'file=@test.zip' \
    -F 'name=' -F 'type=zip_text' -F 'desc=' -F 'corpus=<CORPUS_NAME>' \
    -H 'Username: <USERNAME>' -H 'Password: <PASSWORD>' \
    http://<PLATFORM_URL>/index.php?path=corpus/data/add
```

API parameters:

file = the ZIP archive to be uploaded (containing text+metadata); must be in ZIP format

type = zip_text : this value must be hardcoded to zip_text to allow proper interpretation for the archive

name = this can be any name associated with the zip file to be displayed in the GUI. Usually should be left empty (then the actual filename will be displayed)

desc = description associated with the archive. Usually empty.

corpus = corpus to which the archive will be uploaded

The API requires the following headers for authentication:

username = user used to access the GUI

password = the associated password

Return:

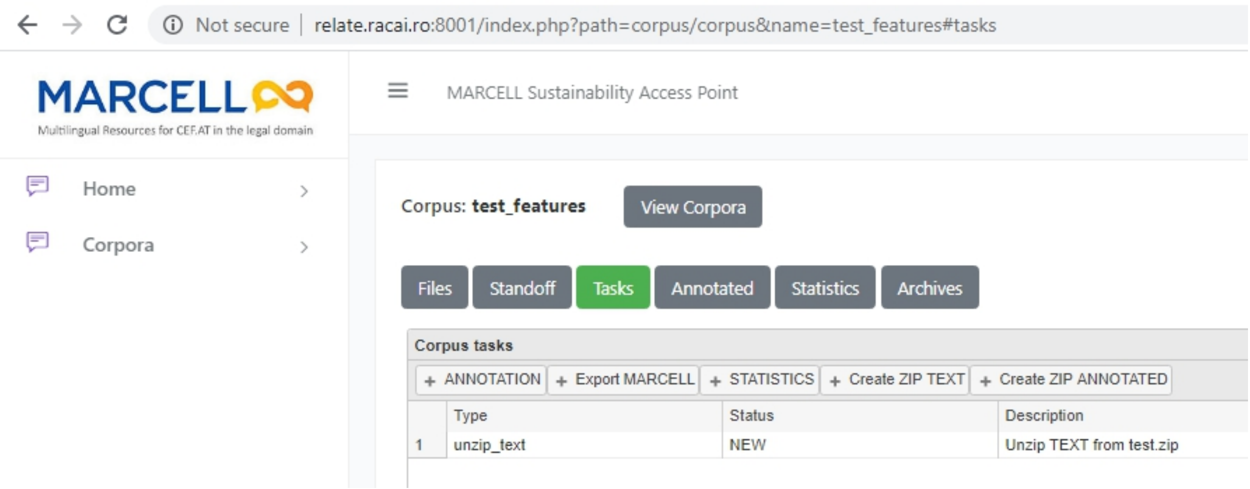
Status OK:
`{"status":true,"message":"File added"}`

Error Status:
`{"status":false,"reason":"Invalid corpus"}`

Wrong address, wrong authentication parameters, user with no access
In this case an HTML page will be returned with the default platform landing page

After a successful upload, a task will be automatically created to extract the archive. This can be seen in the GUI in the “Tasks” tab of the specified corpus. The URL of this page is: http://<PLATFORM_URL>/index.php?path=corpus/corpus&name=<CORPUS_NAME>#tasks

Example:



Not secure | relate.racai.ro:8001/index.php?path=corpus/corpus&name=test_features#tasks

Corpus: **test_features**

Corpus tasks

	Type	Status	Description
1	unzip_text	NEW	Unzip TEXT from test.zip

Once the archive is extracted the task will have Status = DONE (may require page refresh).

5. Language specific Dockers

Each language specific Docker will expose an API which can be used from the GUI to annotate new documents. In order to account for the various technologies used in the different pipelines, each language specific Docker may have a different API. Nevertheless, it is foreseen to keep to a minimum the number of different APIs.

5.1. Romanian Docker

The Docker image specific to processing Romanian language exposes an annotation API allowing individual files to be annotated by requiring the RAW text file and standoff metadata XML file.

API endpoint:
http://<container_url>:<exposed_port>/annotate.php

API parameters (via GET or POST):

text : RAW text file
meta : standoff metadata in XML format
docid : document id to be inserted into final CONLLUP file

Example standoff metadata:

```
<root>
 <Metadata>
<DocumentTitle>RAPORT 4393 11/03/2019</DocumentTitle>
<ArticleTitle>RAPORT 4393 11/03/2019</ArticleTitle>
<AuthorName>-</AuthorName>
<PublicationDate>2019</PublicationDate>
<Source>Website</Source>
<SourceName>http://legislatie.just.ro/Public/FormaPrintabila/00000G0004U3GHG6INE11CXTOEQ9RL5L</SourceName>
<TranslatorName>-</TranslatorName>
 <Medium>Written</Medium>
<DocumentType>RAPORT</DocumentType>
<NewDocId>ro-00000G0004U3GHG6INE11CXTOEQ9RL5L</NewDocId>
<DocumentTextStyle>Law</DocumentTextStyle>
 <DocumentTextDomain>-</DocumentTextDomain>
 <DocumentTextSubdomain>-</DocumentTextSubdomain>
 <CollectionDate>2018</CollectionDate>
 <SubjectLanguage>ro</SubjectLanguage>
 <IssnIsbn>-</IssnIsbn>
 </Metadata>
 </root>
```

5.2. Bulgarian Docker

The Docker image specific to processing Bulgarian language exposes an annotation API allowing individual files to be annotated by requiring the RAW text file and standoff metadata JSON file.

API endpoint:
http://<container_url>:<exposed_port>/annotate.php

API parameters (via GET or POST):

text : RAW text file
meta : standoff metadata in JSON format
docid : document id to be inserted into final CONLLUP file

Example standoff metadata:

```
{
  "identifier": "bg-100257",
  "date_effect": "2003-01-03",
  "date_approved": "2002-12-29",
  "title": "Наредба № 5 от 29 декември 2002 г. за съдържанието и реда за изпращане на уведомлението по чл. 62, ал. 4 от Кодекса на труда",
  "typeBg": "Наредби",
  "typeEn": "Decrees",
  "issuerEn": "Ministries and other institutions",
  "url": "http://dv.parliament.bg/DVWeb/showMaterialDV.jsp?idMat=100257",
  "parent_doc_id": "bg-100257"
}
```

Example API call:

curl -s --data-urlencode "text@5c390e02b7477c5332c29011.txt" --data-urlencode "meta@/tmp/meta" --data-urlencode docid=bg-100257 -X POST http://<container_url>:<exposed_port>/annotate.php

The Bulgarian pipeline docker export can be downloaded here https://ibl.bas.bg/marcell/docker/marcell-bg.tar.bz2

5.3. Polish Docker

The Docker image specific to processing Polish language exposes an annotation API allowing individual files to be annotated by requiring the RAW, HTML or PDF text file and standoff metadata JSON file. HTML format should be compatible with structure returned by the Polish government API for legislative documents (http://isap.sejm.gov.pl/api/isap/).

Requirements:
Recommended: 16 GB RAM, 4 CPU cores

API endpoint (POST):
http://<container_url>:<exposed_port>/annotate

API parameters (via FILES):

text: RAW (*"content-type": "text/plain"*), HTML (*"content-type": "text/html"*) or PDF (*"content-type": "application/pdf")* text file
meta: metadata file in JSON format

API response:

200: success, body contains conllup data
400: missing or wrong metadata values, short error message provided
other: other errors

Example metadata file (here for HTML source file):

```
{
   “pipeline”: “marcell”,
   "content-type": "text/html",
   “language”: “pl”,
   “publisher”: “Dziennik Ustaw”,
   “year”: 2001,
```

“position”: 677,
“date”: “2001-05-23”,
“title”: “Ustawa z dnia 23 maja 2001 r. o ustanowieniu programu wieloletniego \"Budowa Kampusu 600-lecia Odnowienia Uniwersytetu Jagiellońskiego\".”,
“status”: “akt posiada tekst jednolity”,
“in_effect”: true,
“type”: “Ustawa”,
“keywords”: [“budownictwo”, “budżet”, “studenci”, “szkolnictwo wyższe”, “Uniwersytet Jagielloński”],
“source_url”: “http://isap.sejm.gov.pl/api/isap/deeds/WDU/2001/677/text.html”,
“meta_url”: “http://isap.sejm.gov.pl/isap.nsf/DocDetails.xsp?id=WDU20010670677“,
“file_url”: “http://isap.sejm.gov.pl/isap.nsf/download.xsp/WDU20010670677/O/D20010677.pdf”,
“date_approved”: “2001-05-23”,
“date_effect”: “2001-07-14”,
“issuer”: [“SEJM”]
}

Exemplary usage in Python:

```
import requests


url = 'http://<container_url>:<exposed_port>/annotate'
with open('/text/file/path.[txt/html/pdf]', 'rb') as text_file,  open('/meta/file/path.json',
'rb') as meta_file, open('/output/file/path.conllup', 'wb') as conllup_file:
    response = requests.post(url=url, files={'text': text_file, 'meta': meta_file},
timeout=3600)
    conllup_file.write(response.content)
```

Exemplary usage with curl:

```
curl -X POST -F 'text=@/text/file/path.[txt/html/pdf]' -F 'meta=@/meta/file/path.json'
http://<container_url>:<exposed_port>/annotate
```

5.4. Slovak Docker

The Docker image specific to processing the Slovak language exposes an annotation API allowing individual files to be annotated by requiring the RAW text file and standoff metadata file.

API endpoint:
http://<container_url>:<exposed_port>/cgi-bin/annotate
or
http://<container_url>/cgi-bin/annotate
(the port of the docker image is by default 80)

API parameters (via POST):
text : RAW text file
meta : standoff metadata in key-value colon-separated format

API response
Successful annotation is marked by a 200 http status response code.
On success, API returns annotated file in MARCELL CoNLL-U+ format, with a MIME type *text/prs.conllup* and in UTF-8 encoding.

Errors are indicated by returning a 518 http error status, with a short error message describing the nature of the error (e.g. empty raw text or metadata) and an optional debugging message in the response body.

Example standoff metadata:

number:80/2019

entype:decree

type:VYHLÁŠKA

predpistyp:VYHLÁŠKA

predpisdatum:z 11. marca 2019,

nadpis: ktorou sa mení a dopĺňa vyhláška Ministerstva hospodárstva Slovenskej republiky č. 416/2012 Z. z., ktorou sa ustanovujú podrobnosti o postupe pri uplatňovaní obmedzujúcich opatrení pri stave núdze a o opatreniach zameraných na odstránenie stavu núdze v elektroenergetike a podrobnosti o postupe pri vyhlasovaní krízovej situácie a jej úrovne, o vyhlasovaní obmedzujúcich opatrení v plynárenstve pre jednotlivé kategórie odberateľov plynu, o opatreniach zameraných na odstránenie krízovej situácie a o spôsobe určenia obmedzujúcich opatrení v plynárenstve a opatrení zameraných na odstránenie krízovej situácie

podnadpis:Ministerstva hospodárstva Slovenskej republiky

title:VYHLÁŠKA z 11. marca 2019, Ministerstva hospodárstva Slovenskej republiky ktorou sa mení a dopĺňa vyhláška Ministerstva hospodárstva Slovenskej republiky č. 416/2012 Z. z., ktorou sa ustanovujú podrobnosti o postupe pri uplatňovaní obmedzujúcich opatrení pri stave núdze a o opatreniach zameraných na odstránenie stavu núdze v elektroenergetike a podrobnosti o postupe pri vyhlasovaní krízovej situácie a jej úrovne, o vyhlasovaní obmedzujúcich opatrení v plynárenstve pre jednotlivé kategórie odberateľov plynu, o opatreniach zameraných na odstránenie krízovej situácie a o spôsobe určenia obmedzujúcich opatrení v plynárenstve a opatrení zameraných na odstránenie krízovej situácie

file:SK/ZZ/2019/80/vyhlasene_znenie.html

date:2019

docid:sk-legal-2019-80

5.5. Slovenian Docker

The Docker image specific to processing Slovenian language exposes an annotation API allowing individual files to be annotated by requiring RAW text and standoff metadata in JSON format.

API endpoint:
http://<container_ip>/annotate

API parameters (form data via POST):
"text": raw text data
"meta": standoff metadata in JSON format

API response:

200: success, body contains conllu data
500: invalid parameters

Standoff metadata format example:

{
"doc_id":"sl-test123",
"language":"sl",
"date":"2020-06-30",
"title":"Poskusni dokument",
"type":"poskus",
"entype":"test"
}

5.6. Hungarian Docker

The Docker image specific to processing Hungarian language exposes an annotation API allowing individual files containing Hungarian legislative text to be annotated by requiring only the text file. In our case, input metadata as a parameter is not needed because all metadata is determined by the docker itself.

API endpoint (POST):
http://<container_url>:<exposed_port>/annotate

API parameters (via FILES):

file : text file path, file should contain valid Hungarian legislative text with usual title, last line of the file must be an empty line

API response:

200: success, body contains CoNLL-U data
500: invalid parameters or invalid input file or internal error

Usage example with curl:

```
curl -F "file=@INPUTFILEPATH" http://<container_url>:<exposed_port>/annotate
```

5.7. Croatian Docker

The repository can be found here: https://github.com/zzl-ffzg/hr-marcell-pipeline . Production version is in the master branch.

The image is moderately memory-heavy, and requires at least 4GB of RAM on the host machine.

Sample call:

```
curl -X POST -F 'text=Ovaj tekst treba anotirati.' -F 'metadata={"id":"identifikator", "year":
"2020", "title": "Testni dokument", "type": "odluka", "entype": "decision", "descriptors":
[{"descriptor": "testni deskriptor", "tld": "36"}, {"descriptor": "drugi testni deskriptor",
"tld": "36"}], "url": "https://marcell-project.eu", "in_effect_since": "2020"}'
http://localhost:8080/annotate
```

6. Clusterization Docker

Docker container for automated clusterization of multilingual corpora in CoNLL-U Plus format.

Required:

Three shared directories between the docker and the host:

1. Shared input/output directory on mounting point /marcell/mnt. It should contain a directory named "corpora" where the input multilingual corpora in CoNLL-U Plus format is stored with the following directory structure: <SHARED-IO-DIR>/corpora/<lang>/conllup/<lang>-<Id>.conllup. The clusterization results will be saved in the shared folder under a directory named Clusterization_Results. The result of each processing run will create a separate directory with a timestamp in its name.

Sample: result directory structure.

```
<SHARED-IO-DIR>/Clusterization_Results/
<SHARED-IO-DIR>/Clusterization_Results/1617157924_all
<SHARED-IO-DIR>/Clusterization_Results/1617157924_all/clusters_domains.json
<SHARED-IO-DIR>/Clusterization_Results/1617157924_all/centroids.txt
<SHARED-IO-DIR>/Clusterization_Results/1617157924_all/metadata.txt
<SHARED-IO-DIR>/Clusterization_Results/1617157924_all/language_distribution.txt
<SHARED-IO-DIR>/Clusterization_Results/1617157924_all/tfidf.txt
<SHARED-IO-DIR>/Clusterization_Results/1617157924_all/clusters.txt
```

2. Shared Initial feeds directory on mounting point /marcell/Initial_Feeds, containing a configuration file where the initial resources locations are described and the processing steps are defined.

3. Four types of resources are supported: EuroVoc, IATE, multiword term dictionary, and lemma dictionary. Each of them should be stored in a separate directory in the shared initial feeds directory.

Sample directory structure:

```
<SHARED-INITIAL-FEEDS-DIR>/config.txt
<SHARED-INITIAL-FEEDS-DIR>/Terms/Terms.tsv
<SHARED-INITIAL-FEEDS-DIR>/IATE/export_SL_2021-01-08_All_Langs.tbx
<SHARED-INITIAL-FEEDS-DIR>/IATE/export_PL_2021-01-08_All_Langs.tbx
<SHARED-INITIAL-FEEDS-DIR>/IATE/export_SK_2021-01-08_All_Langs.tbx
<SHARED-INITIAL-FEEDS-DIR>/IATE/export_BG_2021-01-08_All_Langs.tbx
<SHARED-INITIAL-FEEDS-DIR>/IATE/export_RO_2021-01-08_All_Langs.tbx
<SHARED-INITIAL-FEEDS-DIR>/IATE/export_HR_2021-01-08_All_Langs.tbx
<SHARED-INITIAL-FEEDS-DIR>/IATE/export_HU_2021-01-08_All_Langs.tbx
<SHARED-INITIAL-FEEDS-DIR>/Eurovoc/uf_hr.xml
<SHARED-INITIAL-FEEDS-DIR>/Eurovoc/uf_pl.xml
<SHARED-INITIAL-FEEDS-DIR>/Eurovoc/uf_ro.xml
<SHARED-INITIAL-FEEDS-DIR>/Eurovoc/desc_sk.xml
<SHARED-INITIAL-FEEDS-DIR>/Eurovoc/desc_sl.xml
<SHARED-INITIAL-FEEDS-DIR>/Eurovoc/uf_sl.xml
<SHARED-INITIAL-FEEDS-DIR>/Eurovoc/desc_hu.xml
```

```
<SHARED-INITIAL-FEEDS-DIR>/Eurovoc/desc_hr.xml
<SHARED-INITIAL-FEEDS-DIR>/Eurovoc/desc_bg.xml
<SHARED-INITIAL-FEEDS-DIR>/Eurovoc/desc_ro.xml
<SHARED-INITIAL-FEEDS-DIR>/Eurovoc/uf_bg.xml
<SHARED-INITIAL-FEEDS-DIR>/Eurovoc/desc_pl.xml
<SHARED-INITIAL-FEEDS-DIR>/Eurovoc/uf_hu.xml
<SHARED-INITIAL-FEEDS-DIR>/Eurovoc/uf_sk.xml
<SHARED-INITIAL-FEEDS-DIR>/Lemmas/Lemmas.tsv
```

Config file <SHARED-INITIAL-FEEDS-DIR>/config.txt supported params:

LANGUAGES - * list of language codes, separated by comma
ANNOTATION_LANGUAGE - selected languages for the document transformation
USE_EUROVOC - whether to use Eurovoc or not
USE_IATE - whether to use IATE or not
USE_TERM - whether to use the multiword term dictionary or not
USE_LEMMA - whether to use the lemma dictionary or not
USE_INTERSECT - whether to use only intersection of lexical unitspresent in all languages
EUROVOC_FOLDER - the folder where to save and read Eurovoc data
IATE_FOLDER - the folder where to save and read IATE data
IATE_XML - list of IATE TBX files ordered as in LANGUAGES
TERM_FOLDER - the folder where to save and read Term dictionary data
TERMS_FILE - path to the input TSV file with terms
LEMMA_FOLDER - the folder where to save and read Lemma dictionary data
LEMMAS_FILE - path to the input TSV file with lemmas
MULTI_CORPUS_FOLDER - folder where to save and read the data extracted from the corpora
CLUSTERS_COUNT - number of clusters
DATA_EUROVOC_2 - document representation with Eurovoc MTs
DATA_FOLDER - path to the document transformation
DATA_ANNOTATED_FOLDER - path to the annotated transformations of documents
RESULTS_FOLDER - folder to save the clusterization results. For every experiment, a new subfolder is created.
CORPORA_FOLDER - path to the corpora. For every language from LANGUAGES, a subfolder with the language code has to be present containing a conllup folder for the documents.

Sample configuration file

```
LANGUAGES = bg, hr, hu, pl, ro, sk, sl
ANNOTATION_LANGUAGE = bg
CLUSTERS_COUNT = 5000
USE_EUROVOC = 1
USE_IATE = 1
USE_TERM = 1
USE_LEMMA = 1
USE_INTERSECT = 0
IATE_FILTER_DUPLICATES = 1
FIRST_MATCH = 0
IATE_XML = export_BG_2021-01-08_All_Langs.tbx, export_HR_2021-01-08_All_Langs.tbx, export_HU_2021-01-08_All_Langs.tbx, export_PL_2021-01-08_All_Langs.tbx, export_RO_2021-01-08_All_Langs.tbx, export_SK_2021-01-08_All_Langs.tbx, export_SL_2021-01-08_All_Langs.tbx
DATA_EUROVOC_2 = 1
CORPORA_FOLDER = /mnt/corpora
EUROVOC_FOLDER = <SHARED-INITIAL-FEEDS-DIR>/Eurovoc
IATE_FOLDER = <SHARED-INITIAL-FEEDS-DIR>/IATE
TERM_FOLDER = <SHARED-INITIAL-FEEDS-DIR>/Terms
LEMMA_FOLDER = <SHARED-INITIAL-FEEDS-DIR>/Lemmas
LEMMAS_FILE = <SHARED-INITIAL-FEEDS-DIR>/Lemmas/Lemmas.tsv
```

TERMS_FILE = <SHARED-INITIAL-FEEDS-DIR>/Terms/Terms.tsv

NB: The Shared Initial Feeds directory is required in two cases:
1. During the first (initial) run to create the configuration structure and the processing of the resources. The results of the initial run are stored under <SHARED-IO-DIR>/configs/config_v<VERSION_NUMBER> and <SHARED-IO-DIR>/resources/Clusterization_Resources_v<VERSION_NUMBER>
2. In case of any change in the Initial Feeds or the config file.

4. Shared Logs directory on mounting point /marcell/logs. All pipeline orchestration logs are stored in <SHARED-LOGS-DIR>/orchestrator.log

Starting Docker container example:

```
docker run -v <SHARED-IO-DIR>:/marcell/mnt -v <SHARED-LOGS-DIR>:/marcell/logs -v <SHARED-INITIAL-FEEDS-DIR>:/marcell/Initial_Feeds  -td  marcell_clusterisation:v1
```

7. Clusterization UI Docker

Dedicated docker container for visualizing clusterization results in a web interface.

Required:

1. Shared input/output directory on mounting point /marcell/mnt where the clusterization results are stored. The same input/output directory used for running the Clusterization Docker.
2. Port mapping: A docker host port mapped to the container port 80.

Starting Docker Clusterization UI container example:

```
docker run -p 8080:80 -v <SHARED-IO-DIR>/:/marcell/mnt  -td  marcell_clusterization_ui:v1
```

8. Semantic Alignment Docker

Container Description

The Semantic Alignment Docker Container contains the following services:

· Corpora Enrichment & Indexing Service: the service takes the **annotated CoNLL-U legal** documents as input, segments them using per-language rules into sections and indexes the data in a full text search index using the Lucene library

· Alignment Service with Web Interface allowing on-the fly alignment and fine tuning of alignment parameters using a Web Interface and a REST API service for alignment

· Automatic Tuning of Alignment Parameters Service allowing unsupervised tuning of parameters for better alignment between individual language pairs.

Container Requirements

The container requires the following:

File system:

- a shared source folder
- a shared index location

Ports:

- 80 for the web interface

Based on the image role, docker should be configured to run either:

- /var/marcell/indexer/run.sh for indexing
- /var/marcell/optimizer/run.sh for parameter optimization
- /var/marcell/

Recommended Setup

For example, a docker composer configuration could look like this:

```
services:
  aligner-service:
image: semantic-micro-alignment
ports:
- 80:80
volumes:
- /var/marcell/shared/source:/var/marcell/source
- /var/marcell/shared/index:/var/marcell/index
- /var/marcell/shared/iate-tbx:/var/marcell/iate-tbx
```

Main Repository Location

https://github.com/clarinsi/semantic-micro-alignment

Indexing Service Instructions

The indexing service is a command line tool located in the "/var/marcell/indexing" folder. Running the tool without any parameters prints the usage instructions:

"Usage: CorpusIndexer <indexMode> <sourcePath> <indexPath> <topicTbxFileUri> <LogPath>

Source path must contain one folder per language (sl, hr, ...) with the Xml CoNLL-UP files.

Index mode can be: 0=single index file per type, 1=index file per language"

If required, the tool can be configured as a CRON job to regularly update the index.