D2.2

Interoperable Interface for Lemon and TEI resources

Author(s): John P. McCrae (NUIG)

Date: 31. 1. 2020

H2020-INFRAIA-2016-2017

Grant Agreement No. 731015

ELEXIS - European Lexicographic Infrastructure

D4.4 Crowdsourcing module

Deliverable Number:	D2.2
Dissemination Level:	Public
Delivery Date:	31. 1. 2020
Version:	1.0
Author(s):	John P McCrae

Project Acronym:	ELEXIS
Project Full Title:	European Lexicographic Infrastructure
Grant Agreement No.:	731015

Deliverable/Document Information

Project Acronym:	ELEXIS
Project Full Title:	European Lexicographic Infrastructure
Grant Agreement No.:	731015

Document History

Version Date	Changes/Approval	Author(s)/Approved by
31. 1. 2020	John P. McCrae	Simon Krek

Table of Contents

List of Figures

1 Executive summary

In order to enable users to simply use the REST API previously defined in D2.1, we implemented a simple application to create an instance of this service. This is implemented as a small cross-platform executable written in the Rust language using the Gotham web framework. This provides a small and very high performance implementation of the REST interface that can work as a standalone service. It is also planned that a more complete version of the interface will be implemented by the tools developed in LEX1[1], in particular the Lexonomy interface[2]. As this is a reference implementation that assumes that the dictionary publisher has their dictionary in a single form and at the moment we support the following formats:

1. **TEI Lex-0:** This XML format, based on the Text Encoding Initiative's (TEI) format for dictionaries but simplified, has been adopted by the project as one of the primary formats of the project.
2. **OntoLex-Lemon:** The OntoLex format is the de-facto standard for representing lexical information as RDF and as such is essential to the work of the project.
3. **JSON:** In addition, a custom JSON format is provided that is easy for the users to target as an input format for the tool.

The tool consists of a simple lifecycle where the dictionary is loaded into a SQLite database so that it may be easily accessed and then the REST server is started. Finally, there is an option to remove an existing database from a running REST service. The tool is open source and available at:

https://github.com/elexis-eu/dictionary-service

In addition, in this deliverable we describe changes to the REST API, previously described in D2.1 and a new interface which has been defined in order to describe how the linking tools, which will be documented in D2.3 can be accessed.

[1] Described in D8.1,2,3,4

[2] https://www.lexonomy.eu/

This project received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731015. The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Union.

1 Reference implementation

1.1 Installation

1.1.1 From Source

This tool can be built with Rust/Cargo[3] using the following command

```
cargo build --release
```

This will create a single binary at target/release/elexis-dictionary-service.

1.1.2 By Docker

The dictionary service is available from Docker Hub.

You can run the command with

```
docker run -it --rm -p 8000:8000 jmccrae/elexis-dictionary-service
```

1.2 Usage

The ELEXIS dictionary service supports a number of commands

1.2.1 Loading data

Data can be loaded with the load command[4]

```
USAGE:
    elexis-dictionary-service load [FLAGS] [OPTIONS] <data>
FLAGS:
    -h, --help       Prints help information
    -V, --version    Prints version information
OPTIONS:
    -c, --config <config>    Configuration to help with mapping
```

[3] https://www.rust-lang.org/

[4] Note if using the Docker replace the elexis-dictionary-service executable with the Docker command above, e.g., docker run -it --rm -p 8000:8000 jmccrae/elexis-dictionary-service load example/example.json

This project received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731015. The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Union.

```
    --db-path <db_path>  The path to use for the database (Default: eds.db)
-f, --format <json|ttl|tei>   The format of the input
    --genre <gen|lrn|ety|spe|his|ort|trm>  The genre(s) of the dataset
                                           (comma separated)
    --id <id>            The identifier of the dataset
    --release <PUBLIC|NONCOMMERCIAL|RESEARCH|PRIVATE>    The release level of
                                                         the resource
ARGS:
<data>    The data to host
```

For example, to load a file it is normally sufficient to give a command as follows:

```
# A JSON file
elexis-dictionary-service load example/example.json
# A TEI Lex-0 file
elexis-dictionary-service load example/example-tei.xml \
  --id tei_dict --release PUBLIC
# An OntoLex file
elexis-dictionary-service load example/example.rdf --release PUBLIC
```

1.2.2 Starting the server

The REST server may be started with the start command:

```
USAGE:
  elexis-dictionary-service start [FLAGS] [OPTIONS]
FLAGS:
  -h, --help       Prints help information
      --no-sql     Do not use SQLite (all data is temporary and session only)
  -V, --version    Prints version information
OPTIONS:
  -c, --config <config>     Configuration to help with mapping
  -d, --data <data>         Also load a single data file
      --db-path <db_path>   The path to use for the database (Default: eds.db)
```


This project received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731015. The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Union.

```
-f, --format <json|ttl|tei>  The format of the input
   --genre <gen|lrn|ety|spe|his|ort|trm>   The genre(s) of the dataset
                                           (comma separated)
   --id <id>            The identifier of the dataset
-p, --port <port>       The port to start the server on
   --release <PUBLIC|NONCOMMERCIAL|RESEARCH|PRIVATE>   The release level
                                                       of the resource
```

For example, to start a server

```
elexis-dictionary-service start
```

The server will be available at http://localhost:8000/

To start a temporary server for a single file (not using SQlite) the following command can be used

```
elexis-dictionary-service start -d example/example.json --no-sql
```

1.2.3 Deleting a dictionary

A dictionary may be removed from the server with the delete command

```
USAGE:
   elexis-dictionary-service delete [OPTIONS] [data]
FLAGS:
   -h, --help       Prints help information
   -V, --version    Prints version information
OPTIONS:
      --db-path <db_path>    The path to use for the database
                             (Default: eds.db)
ARGS:
   <data>    The data file to delete
```

For example

```
elexis-dictionary-service delete dict_id
```


This project received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731015. The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Union.

1.3 Formats

1.3.1 JSON

The JSON format consists of an object of the following form

```
{
   "dict_id": {
      "meta": {   },
      "entries: [   ]
   }
}
```

Where dict_id is the name of the dictionary, the meta value is exactly as would be returned by the /about REST call. The entries value is an array where each element is as would be returned by the entry as JSON REST call

1.3.2 TEI Lex-0

The TEI Lex-0 document should be a valid XML document with at least the following tags

```
<TEI xmlns="http://www.tei-c.org/ns/1.0">
   <teiHeader>
      <fileDesc>
         <titleStmt>
            <title>Name of the dictionary</author>
         </titleStmt>
         <publicationStmt>
            <publisher>Named of the publisher</publisher>
            <availability>
               <licence target="http://url.of.licence">...</licence>
            </availability>
         </publicationStmt>
         <sourceDesc>
```


This project received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731015. The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Union.

```
            <author>Name of the author</author>
          </sourceDesc>
        </fileDesc>
      </teiHeader>
      <body>
        <entry xml:lang="en" xml:id="test">
         <form type="lemma">
            <orth>girl</orth>
         </form>
         <form type="variant">
            <orth>girls</orth>
         </form>
         <gramGrp>
            <gram type="pos" norm="NOUN">noun</gram>
         </gramGrp>
         <sense>
            <def>young female</def>#
         </sense>
      </body>
   </TEI>
```

The following constraints are required

1. A licence must be given with a target
2. An entry must have a form[@type=lemma]
3. An entry must have a gram[@type=pos] and it should have a norm referring to a UD category unless mapping is used (see below)
4. An entry must have a language and an ID
5. An entry must not occur within another entry

1.3.3 OntoLex

An OntoLex document should be a valid Turtle document such as follows:

This project received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731015. The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Union.

```
@prefix lime: <http://www.w3.org/ns/lemon/lime#> .
@prefix ontolex: <http://www.w3.org/ns/lemon/ontolex#> .
@prefix dct: <http://purl.org/dc/terms/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix lexinfo: <http://www.lexinfo.net/ontology/2.0/lexinfo#> .

<#dictionary> a lime:Lexicon ;
   lime:language "en" ;
   dct:license <http://www.example.com/license> ;
   dct:description "A test resource" ;
   dct:creator [
      foaf:name "Joe Bloggs" ;
      foaf:mbox <mailto:test@example.com> ;
      foaf:homepage <http://www.example.com/>
   ] ;
   dct:publisher [
      foaf:name "Publisher"
   ] ;
   lime:entry <#entry1>, <#test> .

<#entry1> a ontolex:LexicalEntry ;
   lexinfo:partOfSpeech lexinfo:commonNoun ;
   ontolex:canonicalForm [
      ontolex:writtenRep "cat"@en
   ] ;
   ontolex:sense [
      skos:definition "This is a definition"@en
   ] .

<#test>  a ontolex:LexicalEntry ;
   ontolex:canonicalForm [
      ontolex:writtenRep "dog"@en
   ] ;
   ontolex:sense [
      ontolex:reference <http://www.example.com/ontology>
   ] .
```

In order to process the file well, certain information should be grouped together, in particular all information about the lexicon should follow after the triple

This project received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731015. The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Union.

```
<#dictionary> a lime:Lexicon
```

A dictionary must have a lime:language and a dct:license.

The entry starts with a triple of the form

```
<#entry1> a ontolex:LexicalEntry
```

All triples after this until another similar triple occurs in the file are considered the description of this entry

All entries must have an ontolex:canonicalForm with an ontolex:writtenRep.

All entries must be given by URIs and referred to by a lime:entry triple from a lexicon

1.4 Configuration

Configuration may be performed using a configuration file. This is particularly useful for providing mappings. An example configuration is as below

```
{
    "posProperty": "http://www.lexinfo.net/ontology/2.0/lexinfo#partOfSpeech",
    "posMapping": {
        "substantive": "NOUN",
        "http://www.lexinfo.net/ontology/2.0/lexinfo#pronoun": "PRON"
    },
    "defaultId": "dict_id",
    "defaultRelease": "PUBLIC"
}
```

The configuration has the following values

- posProperty: The URI of the RDF property used to indicate part-of-speech

This project received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731015. The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Union.

- posMapping: A mapping of values, either RDF URI or the content of TEI tags that is mapped to a given UD value (ADJ, ADP, ADV, AUX, CCONJ, DET, INTJ, NOUN, NUM, PART, PRON, PROPN, PUNCT, SCONJ, SYM, VERB, X)
- defaultId: The default ID for a dictionary (instead of a --id flag)
- defaultRelease: The default release level of the dictionary (PUBLIC, NONCOMMERCIAL, RESEARCH, PRIVATE)

This project received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731015. The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Union.

2 Updates to REST API Specification

The following changes were made to the API as documented in D2.2

- Usage is now a string in the custom JSON format.
- A reference URL is no longer required in the sense of an entry in the custom JSON format.
- The part of speech values in the custom JSON format no longer require that they are prefixed with lexinfo:
- It is now possible to specify an API key using the X-API-KEY HTTP header. If this API key is required and missing or is incorrect then the REST server may now return a 403 (unauthorized) error.
- Some cosmetic updates to the specification document were also made.

This project received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731015. The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Union.

3 Linking REST API

The REST API for linking is designed to enable the core infrastructure of LEX1 to interact with the linking services developed in WP2. The architecture and proposed workflow is depicted below in Figure 1.

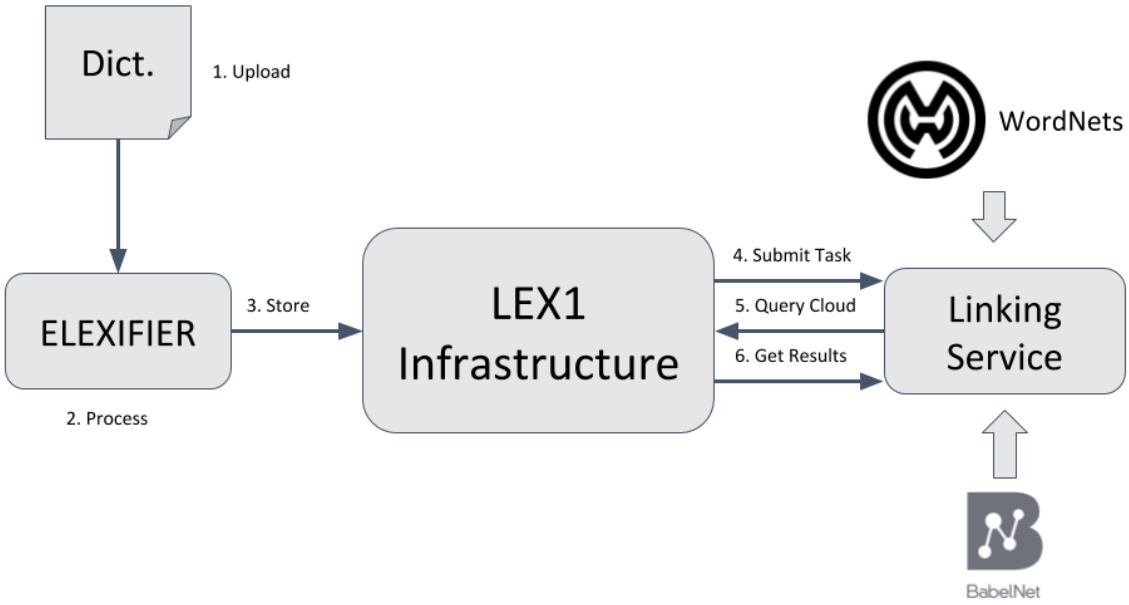

Figure 1: Proposed interaction of components in LEX1 with Linking Service

1. The user **uploads** the dictionary to the ELEXIS infrastructure, this may be achieved either by directly uploading a file or by using the REST interface described in this document.
2. The dictionary is **processed** by using the automatic segmentation and identification tools (D1.3) and/or the conversion tools (D1.4).
3. The resulting dictionary in TEI Lex-0 or OntoLex is **stored** in the cloud.
4. The LEX1 infrastructure **submits a linking task** to the linking service
5. The linking service **queries the ELEXIS cloud** to obtain the dictionaries in a standard form.
6. On completion of the linking the LEX1 infrastructure **gets the resulting linking** and adds it to the dictionary matrix.

This project received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731015. The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Union.

3.1 Implementation of Linking API

This section briefly describes the linking API, more details of which can be found at:

https://elexis-eu.github.io/elexis-rest/linking.html

3.1.1 Submitting a linking task

To submit a linking task a POST request is made to /submit. It has the following parameters

- A **source** dictionary, which must be identified either by a unique ID in the ELEXIS cloud or by an endpoint URL (and optionally an API key) for a dictionary not in the ELEXIS cloud. In addition, it is possible to specify a list of entries in the dictionary to link.
- A **target** dictionary, specified like the source dictionary.
- A **configuration** object, whose parameters depend on the implementation of the linking service

If successful, the service returns a unique **tracking ID**.

3.1.2 Getting the status of the linking task

As the linking service may take a long time to complete linking of large dictionaries, it is necessary for the clients of the linking service (LEX1 infrastructure) to poll the service to check the status. This is done by making a POST request to the /status endpoint. The content of this request must be exactly the tracking ID returned by submitting the task. The response provides a message and a status which is one of PROCESSING, COMPLETED or FAILED.

3.1.3 Getting the results of a linking

Once the status of the linking has reached COMPLETED, then it is possible to obtain the results using the /result call. As before the tracking ID must be posted to the endpoint. The result is a JSON document describing the linking, this consists of an array of objects containing the following

- The identifier of the source entry linked
- The identifier of the target entry linked
- A list of links specified as follows
    - The identifier of the source sense
    - The identifier of the target sense
    - The type of linking, either exact, broader, narrower or related
    - A confidence score between 0 and 1

This project received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731015. The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Union.

An example document is given below


```
[{
   "source_entry": "cat-n",
   "target_entry": "cat-EN",
   "linking": [
     {
       "source_sense": "sense1",
       "target_sense": "00016606n",
       "type": "exact",
       "score": 0.8
     },
     {
       "source_sense": "sense2",
       "target_sense": "00001844n",
       "type": "broader",
       "score": 0.2
     }
   ]
 },{
   "source_entry": "dog-n",
   "target_entry": "dog-EN",
   "linking": [
     {
       "source_sense": "sense1",
       "target_sense": "00015267n",
       "type": "exact"
}]}]
```


This project received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731015. The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Union.

4 Summary

In this deliverable we have presented the REST API implementation that acts as a basis for the linking work in this work package. This will contribute to the next deliverable D2.3 which will provide an implementation of the linking service. We have also in addition to the description of work provided the intended structure of this service by means of the REST interface and how it will use to communicate with the rest of the LEX1 infrastructure.

This project received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731015. The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Union.