D3.4 Language Resource and Service Linking

Author(s): Jorge Gracia, Julia Bosque-Gil, Bernardo Stearns, Christian Chiarcos, Mohammad Fazleh Elahi, Patricia Martín-Chozas, John P. McCrae, Mariano Rico

Date: 31.07.2021

This project received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825182. 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.

H2020-ICT-29b

Grant Agreement No. 825182

Prêt-à-LLOD - Ready-to-use Multilingual Linked Language Data for Knowledge Services across Sectors

D3.4 Language Resource and Service Linking

Deliverable Number: D3.4
Dissemination Level: Public
Delivery Date: 2021-07-31
Version: 1.0
Author(s): Jorge Gracia, Julia Bosque-Gil, Bernardo Stearns, Christian Chiarcos, Mohammad Fazleh Elahi, Patricia Martín-Chozas, John P. McCrae, Mariano Rico

Document History

Version Date	Changes	Authors
2021-06-16	**initial document**	**Jorge Gracia**
2021-07-09	**added partner contributions**	**all partners**
2021-07-15	**pre-final version, ready to review**	**Jorge Gracia**
2021-07-26	**reviewed version**	**Mohammad Fazleh Elahi**
2021-07-30	**final version**	**Jorge Gracia**

Table of Contents

D3.4 Language Resource and Service Linking

This document is a short report that accompanies the software deliverable D3.4 “Language Resource and Service Linking". The idea is to serve as a guideline of the different linking algorithms and systems developed in Prêt-à-LLOD as part of the “Prêt-à-LLOD-Link" component: what are their basic functionalities, how the different sub-components are interrelated, and how they relate with other Prêt-à-LLOD components, particularly “Prêt-à-LLOD-Transform" and “Prêt-à-LLOD-Workflows”. The document serves also as a pointer to the repositories where the code, along with a more complete technical documentation, is available.

While the initial Prêt-à-LLOD linking software is released with this deliverable, it will be continuously improved and updated until later stages of the project. In fact, further testing and development is expected as part of the integration of the linking component with the “Language Resource Transformation Software” (deliverable D3.3, due in month M33), with the pilots (deliverables D4.3, D4.5, D4.7, D4.9, due in month M36), and as part of the progress of the overall Prêt-à-LLOD research challenges (deliverable D3.6, due in month M39). Such deliverable D3.6 will provide the final snapshot of the software component initially delivered in D3.4, and will update its status at the end of the project.

1. Introduction

Task 3.2, **Prêt-à-LLOD Link,** addresses the challenge of “*Linking conceptual and lexical data for language services*" in Prêt-à-LLOD. Novel (semi-)automatic methods have been studied that aim at establishing complex links between monolingual and multilingual resources at the conceptual level, lexical level, or between the conceptual and lexical levels (lexicalisation).

The different linking services have been developed to work both as stand-alone components or interacting with other Prêt-à-LLOD components in an NLP workflow. To that end, they have been developed as Docker components compatible with Teanga (see deliverable D3.5). This will allow seamless interaction between the Prêt-à-LLOD Transformation, Prêt-à-LLOD Link, and Prêt-à-LLOD Workflow components.

In the remainder of this document, we will present an overview of the Prêt-à-LLOD linking component and its Teanga-based architecture (Section 2). Then, we will review the particular linking services developed at three different levels (Section 3 to 5): lexicalisation, conceptual linking, and lexical linking. Since D3.4 is a software deliverable, we are not entering into the scientific and technical details of every sub-component, referring to the “research challenges reports" (D3.1, D3.2, D3,6) to that end, as well as to the technical documentation that comes with each component in their respective repositories. Then, in Section 6, we will review three

other linking components (namely, the Lexicon-LLOD linking service, the Naisc framework, and Lemonade++) that, due to different reasons, are not part of this deliverable but will be used in conjunction with the services developed for the Prêt-à-LLOD Link component. In Section 7, we show some examples of potential linking workflows that can be developed in Teanga on the basis of the introduced linking services. Finally, Section 8 contains some concluding remarks.

2. The Prêt-à-LLOD linking component

This section gives an overview of the Prêt-à-LLOD linking component, its integration into the Teanga workflow system, and an enumeration of the different linking services.

2.1. Overview

We have divided the different linking subcomponents into three categories (see deliverable D3.2), namely:

A. **Lexicalisation**, which comprises services that, given an ontology and/or a lexicon, produces links between ontology and lexicon. Frequently, the lexicon is created or expanded in the process.
B. **Conceptual linking**, whose services establish links among ontology entities.
C. **Lexical linking**, where the links are established among the lexical information contained in the lexicon.

Figure 2.1 shows an overview of the developed subcomponents, organised in the three different categories. In addition to the possibility of operating in isolation, all of these sub-components have been implemented as Docker containers to enable its inclusion as part of Teanga NLP workflows.

Given that the Naisc linking framework has been also dockerized in a Teanga compatible manner, interoperation between Naisc and the Prêt-à-LLOD linking services will be possible, as shown in the figure. Also, the lexicalisation sub-components will be able to interoperate with Lemonade++, an external tool not included in this deliverable, but developed in the context of WP3 for user-guided lexicalisation (see Section 6).

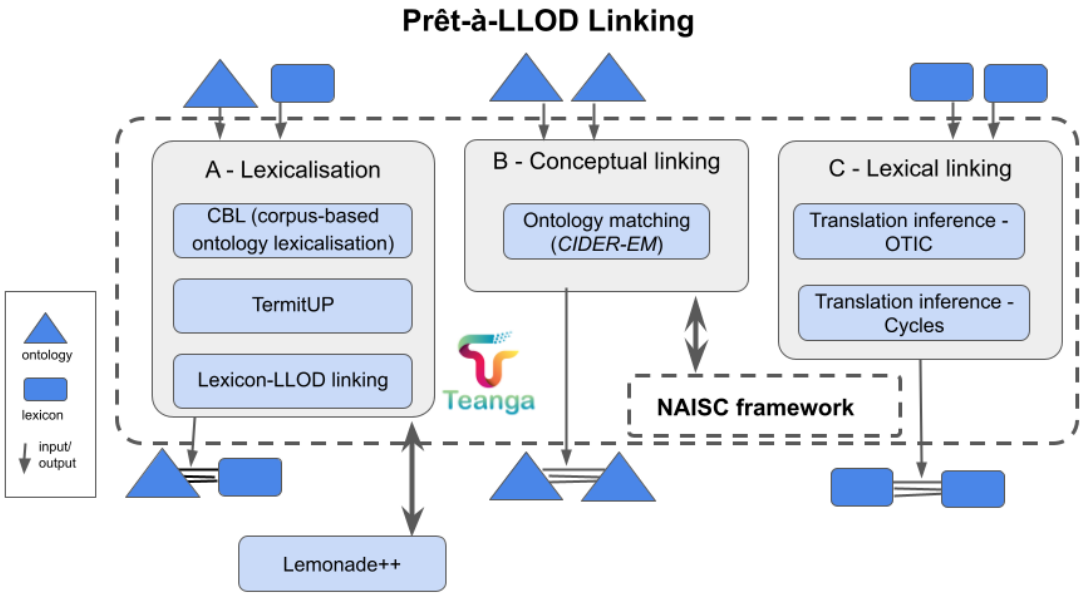

Figure 2.1: An overview of the Prêt-à-LLOD Linking component

Notice that the containerisation of the linking services enables also the integration of such services with other components such as the Prêt-à-LLOD Transform component (Fintan) in the context of Prêt-à-LLOD Workflow (Teanga).

2.2. Teanga-based architecture

Teanga is a workflow manager that helps users create, explore and execute complex workflows, such as linked data, sentiment analysis and other natural language processing applications. Teanga is distributed as a command-line Tool allowing users to access a user interface locally where they can create new workflows or execute existing ones. Summed up with having user-friendly interfaces and accessibility Teanga manages the infrastructure required for executing those workflows enabling users to scale it organically by testing it locally and scaling to multiple servers. Apart from end users which use the command-line tool for managing workflows, Teanga also enables service providers to reliably integrate and test their service in Teanga using Docker and REST APIs.

The current code base for Teanga resides in GitHub and documentation is available as a website. Those repositories are being continuously updated and the latest release of the software can be accessed at any time and examples can be found for better comprehension.

- GitHub (codebase)
- Teanga documentation

Accessibility, reliable standards, and organic scalability fit well with linking components workflows as there are a variety of available services which can be combined in diverse

ways and once those combinations are validated they require scalability of large experiments.

Below we will highlight the main aspects of the technologies and architecture:

Airflow is an easy to use open source platform created by the Python community to programmatically author, schedule and monitor workflows. Implemented in the Python programming language and it gained popularity (21.9k stars on GitHub) due to being scalable, dynamic, extensible and elegant.

Docker provides a standard and reusable way to create infrastructure and uses the created infrastructure abstracting from the current Operating System in which it is being used. It provides the ability to package and run an application in a loosely isolated environment called a container. The isolation and security allow you to run many containers simultaneously on a given host.

Rest API Application Programming Interface (API) defines the allowed interactions between two pieces of software, just like a user interface defines the ways in which a user can interact with a program. An API is composed of the list of possible methods to call (requests to make), their parameters, return values and any data format they require (among other things).

OpenAPI Specification, defines a standard, programming language-agnostic interface description for HTTP APIs, which allows both humans and computers to discover and understand the capabilities of a service without requiring access to source code, additional documentation, or inspection of network traffic.

Docker Hub is a registry service on the cloud that allows users to download Docker images that are built by other communities. Developers can also upload their own Docker built images to Docker Hub. Teanga is using Docker Hub for storing registered services.

The naming strategy decided for the linking services uploaded in DockerHub is as follows:

Level A (lexicalisation):
https://hub.docker.com/r/pretallod/lex-<component name>

Level B and C (linking at the conceptual and lexical levels):
https://hub.docker.com/r/pretallod/link-<component name>

2.3. Teanga requirements

Teanga enables service providers to integrate their services with it by following some defined standards. In this section, we describe the requirements for integrating a **linking service** in Teanga.

1. Creating a **Rest API application**. Currently, Teanga can only handle services with JSON outputs, which means services that output a single unnamed value, an array of unnamed

values, a schema (a dictionary of named values) or an array of schemas. It's necessary to create an application that receives HTTP requests and outputs JSON.

2. Describing the Rest API with **OpenAPI** specification. Creating an OpenAPI specification is very flexible, it only requires to follow the basic structure suggested in their documentation as a guideline and create it as a file named openapi.yaml file.

3. "Dockerizing" the API. In order to "dockerize" your application, you need to create a Dockerfile which will be used to create your image with the command "docker build". Again, creating a Dockerfile is very flexible, the main objective is installing all dependencies of your application and having a command to start the application when docker starts the container.

4. Publishing the Docker Image on **Docker Hub**. Before publishing your docker image in the docker hub, you have to create an account there, and login in the terminal with the command "docker login".

2.4. Data interchange mechanisms

As referred in the previous section, JSON will be the required format for exchanging information across the linking components and with other Prêt-à-LLOD components. There are, however, different possible representation mechanisms for such information exchange, as well as the possibility of using external resources to support data exchange. In the following, we summarize the design decisions that we adopted for the interchange of information in the **Teanga linking workflows**:

- **File exchange**. Files can be ingested as input in the first component of a Teanga linking workflow, or served as output by a final component, but they will not be used to exchange information between intermediate components.

- **SPARQL Endpoints**. In case that large amounts of semantic data are required or produced by any docker component in a workflow, external triple stores and SPARQL Endpoints might be used to act as mediators of such a data exchange. If possible, this solution should be avoided for small/medium dta exchanges, but it is still acceptable for large data.

- **Semantic representation format**. Depending on the linking level, different representation mechanisms should be used for the data that is produced by the different components:
    - Levels A (lexicalisation) and C (lexical linking): the output data will be represented using **Ontolex lemon**[1] as a prefered way.
    - Level B (conceptual linking): the output data will follow well-established Semantic Web formats to describe an ontology alignment in **RDF** (e.g., the Alignment Format[2]).

[1] https://www.w3.org/2016/05/ontolex/
[2] https://moex.gitlabpages.inria.fr/alignapi/format.html

- **Other formats**. Other output formats, in addition to the previous ones, are also possible, to adapt the system to the requirements of particular use cases or linking service consumers. For instance a set of translations that can be inferred by lexical linking components can be represented both in Ontolex lemon and in the TIAD format[3] (conceived for TSV files). In such cases in which several formats can be used, a **format selection parameter** is needed in the linking component, to specify the format in which the data will be served.

3. Ontology lexicalisation services (Level A)

The goal of ontology lexicalisation is to enrich and link existing ontologies with lexical entries that verbalise the ontology elements, ideally across languages.

3.1. CBL - Corpus Based Lexicalisation

Short description

The corpus-based lexicalization (CBL) is a tool for bridging the lexical gap between natural language (NL) expressions (i.e. linguistic patterns) and the content stored in an RDF knowledge base (i.e. ontology). The tool uses class-specific association rules (Ell et al., 2021) together with null-invariant measures of interestingness to induce correspondences between lexical elements and KB elements. As an example of such an association rule, consider the rule that predicts that if the text about a person contains the token 'Greek', then this person has the relation *nationality* to the entity *Greece*. Another rule predicts that if the text about a settlement contains the token 'Greek', then this *settlement* has the relation *country* to the entity *Greece*.

Formats, inputs, outputs

Input: Corpus (i.e. DBpedia abstracts) and DBpedia Knowledge Graph

Output: ontolex Lemon in JSON-LD format

Code repository

https://github.com/Pret-a-LLOD/ontology-lexicalization

Teanga-compliant version
https://hub.docker.com/r/pretallod/lex-cbl

[3] https://tiad2021.unizar.es/

3.2. TermitUp

Short description

TermitUp (Martin-Chozas et al., 2021) is a tool for terminology enrichment: given a domain specific corpus, TermitUp performs statistical terminology extraction and post-process the resulting term list with a series of linguistic processes and external tools to remove noise and non-terminological structures. Then, TermitUp queries several language resources (some part of the Linguistic Linked Open Data cloud) for candidate terms matching those in the term list. TermitUp builds sense indicators for both the source and the candidate terms, and performs a Word Sense Disambiguation process (with Semantic Web Company's service), linking the extracted terms with those with the closest domain from external resources. Once the linking is done, TermitUp retrieves every piece of information available (translations, synonyms, definitions, usage notes and terminological relations), already disambiguated, and enriches the source term lists, creating links amongst the resources in the LLOD. Afterwards, TermitUp offers the possibility of creating hierarchical relations amongst the terms in the source list and also of validating those relations retrieved from the external resources, by applying linguistic patterns and additional language resources. Finally, the results are published in separate JSON-LD files, modeled in SKOS and Ontolex (users' choice). TermitUp API publishes the enriched terminologies generated in a Virtuoso Endpoint, where they can be freely queried.

Formats, inputs, outputs

Input: Corpus (raw text), KGs (user can select the KG to select info from, e.g., EuroVoc, Wikidata, ...)

Output: Ontolex or SKOS, in JSON-LD files that can be stored in TermitUp's SPARQL Endpoint.

Code repository

https://github.com/Pret-a-LLOD/termitup

Teanga-compliant version in Dockerhub

https://hub.docker.com/r/pretallod/lex-termitup

4. Ontology matching services (Level B)

This comprises services that discover links between ontology entities by performing ontology alignment tasks.

4.1. CIDER-EM

Short description

CIDER-EM (Context and Inference baseD ontology alignER - EMbeddings version) is a tool for monolingual and cross-lingual ontology matching. This system is a new version of CIDER-CL (Gracia et al., 2013). A series of adaptations have been done in order to improve the cross-lingual capabilities of the tool through the use of monolingual and multilingual word embeddings. These word vectors are utilised for measuring the similarity between the ontology entities.

Formats, inputs, outputs

Inputs: two ontologies given in OWL or RDFS

Output: the alignment between them (in the RDF Alignment Format)

Code repository

https://github.com/Pret-a-LLOD/CIDER-EM-api

Teanga-compliant version in Dockerhub

https://hub.docker.com/r/pretallod/link-cider-em

5. Lexical Linking (Level C)

These services discover links at the lexical level; for instance, translation relations between lexical entries from lexicons in different languages.

5.1. Translation Inference: OTIC

Short description

This is a re-implementation of the "One Time Inverse Consultation" algorithm (Tanaka and Umemura, 1994). It infers translation pairs between two languages indirectly connected in a graph through a pivot language (i.e. an intermediary language with translations with the other two languages). It has been extensively used as a baseline in the TIAD campaign organised by Prêt-à-LLOD. It produces translations for a single source term or infers a whole bilingual dictionary between two given languages and a pivot language.

Formats, inputs, outputs

Inputs: source, pivot, and target languages, and source entry (optional)

Output: array of translatable pairs in JSON

Code repository

https://github.com/Pret-a-LLOD/OTIC-api

Teanga-compliant version in Dockerhub

https://hub.docker.com/r/pretallod/link-otic

5.2. Translation Inference: Cycles

Short description

It uses a “cycle density based” algorithm to infer translation pairs between languages not directly connected in a graph. It explores cycles to identify potential translations and measure their density to compute the confidence degree. It produces translations for a single source term or infers a whole bilingual dictionary between two given languages. See (Villegas et al., 2016) and (Lanau-Coronas and Gracia, 2020).

Formats, inputs, outputs

Inputs: source and target languages, and source term (optional)

Output: array of translatable pairs in JSON

Code repository

https://github.com/Pret-a-LLOD/Cycles-api

Teanga-compliant version in Dockerhub

https://hub.docker.com/r/pretallod/link-cycles

6. Integration with other linking components

In previous sections we have described the different services developed as part of this software deliverable (D3.4). There are, however, other linking components that, due to different reasons, are not part of this deliverable but will be used in conjunction with the services developed for the Prêt-à-LLOD Link component, illustrating also the potential extensibility of the Prêt-à-LLOD linking component.

6.1. Lexicon-LLOD linking

This service, not initially planned for D3.4 (but that will be reported in D3.6) discovers links between bilingual dictionaries, represented in Ontolex lemon as translations sets between two monolingual lexicons, and a multilingual ontology in the LLOD cloud. The used algorithm, simple but effective, looks for ontology entities containing both translated terms, and creates intermediate lexical senses between such terms in the original lexicons and the discovered ontology entity. A first implementation of this algorithm is used to discover links between bilingual dictionaries and the RDF version of BabelNet[4].

We consider this linking service a sort of “inverse lexicalisation” service, where two lexicons are taken as starting point, and then links to ontology references are found (and not the other way around, which is typical in lexicalisation). An example of its possible interaction with other Prêt-à-LLOD components is illustrated later in section 7.2.

6.2. Lemonade++

This tool, evolution of (Rico 2015), is a web application aimed at providing lexicalization of ontologies in a collaborative way. The lexicalization is generated in lemmon-patterns format. Additionally, this tool can read lexicalizations (initially, manually created) and provide a refinement environment in which users can polish the lexicalizations in an intuitive way. Specifically, users do not need technical knowledge, and their unique requirement is a proficient level in the language(s) of the lexicalization(s).

The software of Lemonade++ is not formally part of deliverable D3.4, but it serves as a prototypical implementation of a downstream application that combines Pret-a-LLOD linking components with real-world applications[5]. This tool will be reported with more detail in D3.6.

[4] http://babelnet.org/rdf
[5] The application will be maintained by the Ontology Engineering Group (OEG) at UPM

Figure 6.1 shows how the two lexicalisation components (CBL and TermitUp, introduced in Section 3) can interact with Lemonade++. In principle, Lemonade++ can mediate the interaction with the user and drive the manual lexicalisation of an ontology (A in the figure), which is later enriched with the lexicalisation produced by the other components automatically (B), for its later integration and validation.

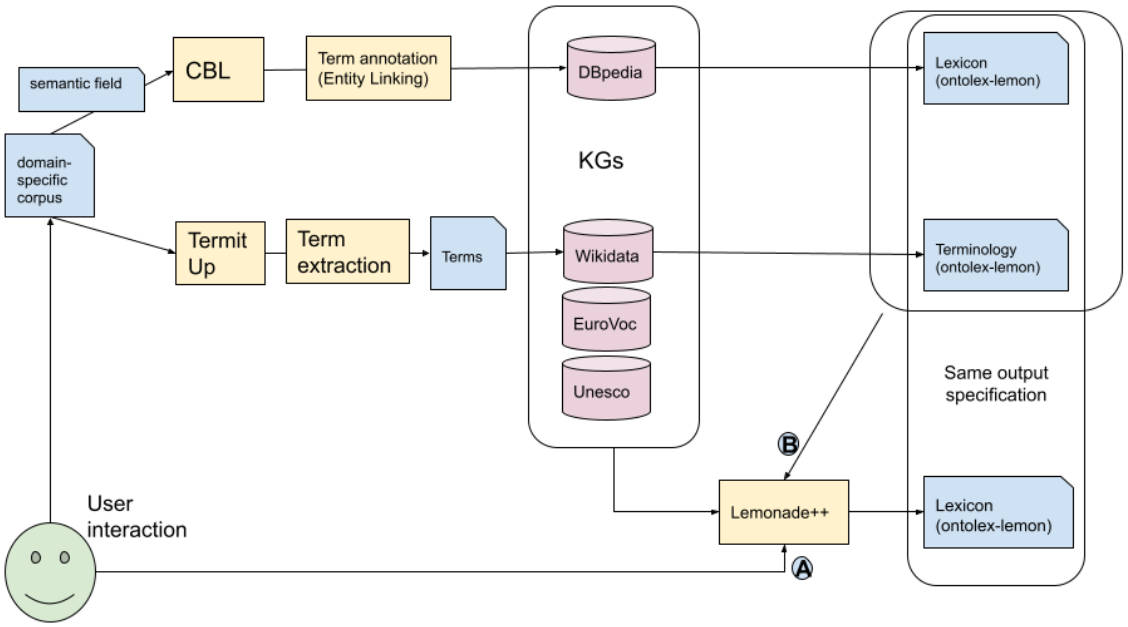

Figure 6.1: Interaction of the lexicalisation components with Lemonade++

6.3. Naisc Framework

Naisc[6] is a framework developed by NUIG for dataset linking as part of the Insight Centre for Data Analytics and the ELEXIS H2020 project. This framework is perfect for the integration of linking components with other systems. The Naisc framework has already been developed into components and integrated with Teanga as described in deliverable D3.5. This enables the possibility of a later interaction with the Prêt-à-LLOD Linking component (see Figure 2.1), given the fact their respective services have been developed in a Teanga-compatible way. Naisc organizes the system around the following components (McCrae et al. 2021):

- **Blocking**: The blocking step finds the set of pairs that are required to be linking. For most tasks a balance between finding too many candidates, which leads to long computation times and reduced performance, and discarding relevant candidates must be found. For specific tasks such as dictionaries, linking matching headwords

[6] https://github.com/insight-centre/naisc

can simply find matching headwords and output every sense pair between these two entries making the task more feasible.

- **Lens**: The lens examines the data around the sense pair to be linked and extracts text that can be compared for similarity. These can include definitions of concepts or other textual information such as examples.
- **Text features**: The text features extract a set of similarity judgements about the texts extracted with the lenses and are described in more detail in the following section.
- **Graph features**: Graph (or non-textual) features do not rely on the text in the dataset but instead look at other features.
- **Scorer**: From a set of features extracted either from the text or from other graph elements, a score must be estimated for each of the sense pairs. This can be done in either a supervised or unsupervised manner and we implement standard methods for supervised classification such as SVMs and unsupervised classification using voting.
- **Matcher and Constraint**: There are normally some constraints that we wish to enforce on the matching and these are applied by the matcher.

Each of these components are offered individually by Naisc as Teanga components and can be replaced by external REST services. The definitions of the REST services are available at https://app.swaggerhub.com/apis/jmccrae/Naisc/1.0 and descriptions of the existing implemented components are available at https://github.com/insight-centre/naisc/blob/master/CONFIGURATION.md.

7. Workflow examples

In this section we show, for illustration purposes, several Teanga-based workflows, not yet realised, but that will be possible to build thanks to the fact that the linking services are containerised, and can interact with other Prêt-à-LLOD components, such as Fintan, for RDF transformation. **Note**: For the sake of presentation, some elements have been simplified, and some parameters are visualised in violet components.

7.1. Lexicon and terminology generation

This simple workflow illustrates how, taking advantage of the common output of both CBL and TermitUp lexicalisation components, their results are aggregated and served as a common output.

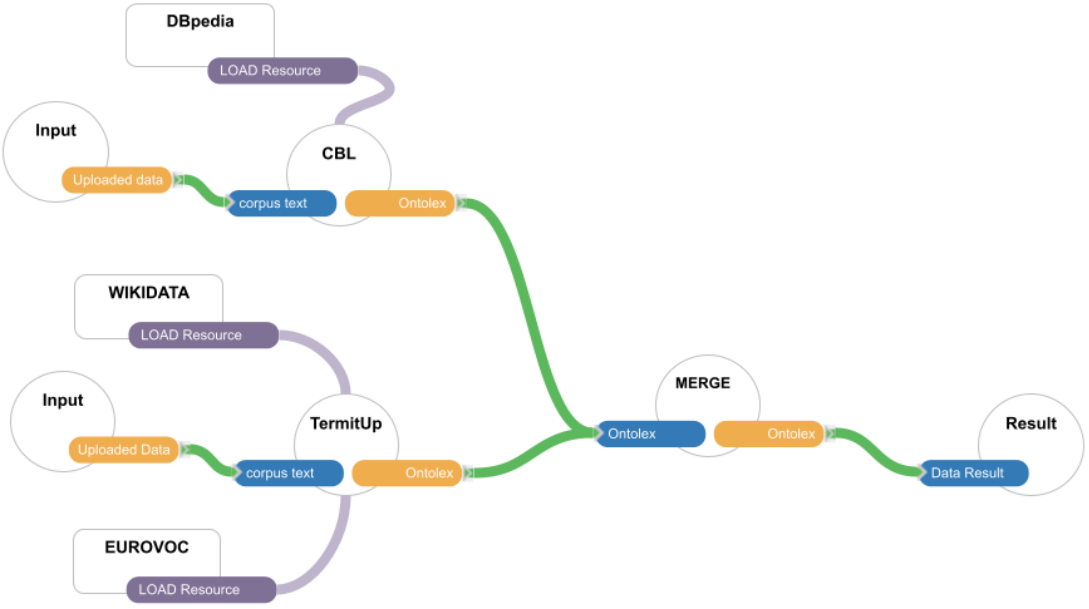

Figure 7.1: Lexicon and terminology generation workflow

7.2. Dictionary data transformation + linking into the LLOD cloud

This workflow exemplifies how Fintan is used to transform the Apertium dictionary data into RDF, and the result uploaded into a triplestore accessible through a SPARQL. Such data is taken by a linking component to discover links between Apertium RDF and the LLOD cloud (BabelNet in particular).

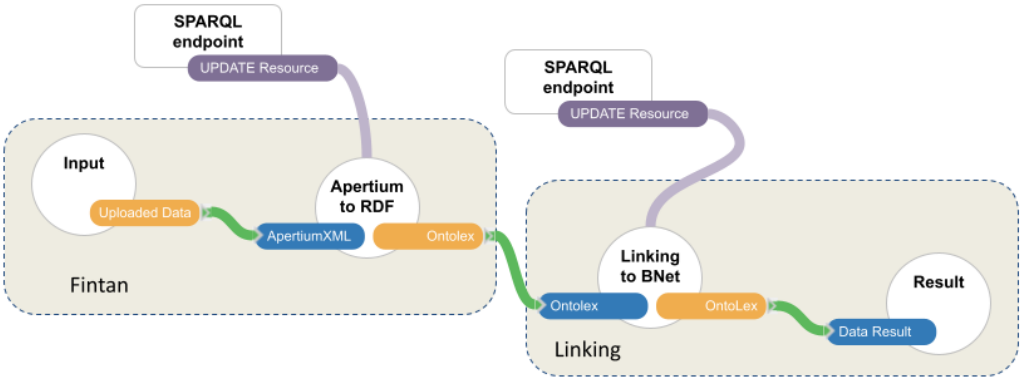

Figure 7.2: Apertium RDF transformation + linking into BabelNet

7.3. Dictionary data transformation + translation inference + exploitation

This workflow illustrates the data transformation in Fintan of Apertium RDF. As a result of this initial transformation, a number of bilingual dictionaries are built, interconnected in a unified RDF graph. However, not every language pair is linked with direct translations in the Apertium grap. To that end, the output of Fintan is connected in Teanga to the input of the OTIC linking component (see Section 5.1), to discover translations between language pairs not initially connected in the Apertium graph (e.g., French-English). Such enriched translations are finally used by one of the Prêt-à-LLOD pilots (Semalytix' Pharos) to perform cross-lingual model transfer in the Pharma domain (see Gracia et al. 2019).

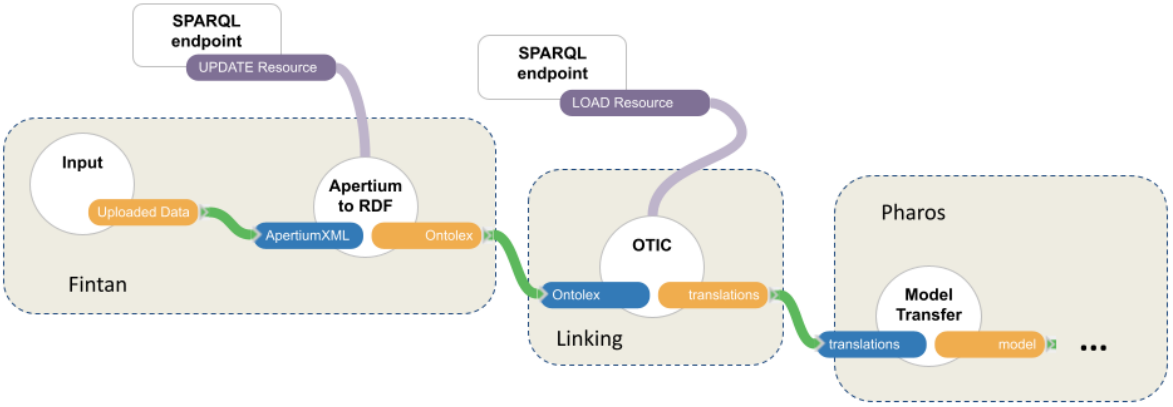

Figure 7.3: Apertium RDF transformation + OTIC translation inference + exploitation in Pharos

8. Conclusions

This document is the report that accompanies the software deliverable D3.4 "Language Resource and Service Linking". It gives the basic information of the different linking services developed in Prêt-à-LLOD as part of the "Prêt-à-LLOD-Link" component and serves also as a pointer to the repositories where the code and the Docker components are available. Some exemplary Teanga workflows are pictured as well, to illustrate the possible interaction of the linking component with other Prêt-à-LLOD components.

References

Ell B, Elahi MF, Cimiano P. Bridging the gap between Ontology and Lexicon via Class-specific Association Rules Mined from a Loosely-Parallel Text-Data Corpus. accepted at the LDK 2021 – 3rd Conference on Language, Data and Knowledge, Zaragoza, Spain.

Gracia, J., and Asooja, K. (2013). Monolingual and cross-lingual ontology matching with CIDER-CL: Evaluation report for OAEI 2013. In Proceedings of the 8th Ontology Matching Workshop (OM 2013), at 12th International Semantic Web Conference, ISWC 2013, pp. 109-116. Sydney, Australia.

Gracia, J., Fäth, C., Hartung, M., Ionov, M., Bosque-Gil, J., Veríssimo, S., Chiarcos, C., & Orlikowski, M. (2020). Leveraging Linguistic Linked Data for Cross-Lingual Model Transfer in the Pharmaceutical Domain. In B. Fu & A. Polleres (Eds.), *Proc. of 19th International Semantic Web Conference (ISWC 2020)* (pp. 499–514). Springer.

Lanau-Coronas, M., & Gracia, J. (2020, May). Graph Exploration and Cross-lingual Word Embeddings for Translation Inference Across Dictionaries. In Proc. of the 2020 Globalex Workshop (pp. 106-110).

Martin-Chozas, P., Vázquez-Flores, K., Calleja, P., Montiel-Ponsoda, E., and Rodríguez-Doncel V. TermitUp: Generation and Enrichment of Linked Terminologies [UNDER REVIEW]. Semantic Web Journal. IOS Press. 2021

McCrae, J. P., Ahmadi, S., Yim, S., and Bajčetić, L. The ELEXIS system for monolingual sense linking in dictionaries. Proceedings of The Seventh Biennial Conference on Electronic Lexicography, eLex 2021.

Rico, M., and Unger, C., “Lemonade: A web assistant for creating and debugging ontology lexica”. In 20th International Conference on Applications of Natural Language to Information Systems (NLDB), Springer LNCS 9103, pp 448-453, 2015.

Tanaka, K., and Umemura, K. "Construction of a bilingual dictionary intermediated by a third language". In COLING, pages 297–303, 1994.

Villegas, M., Melero, M., Bel, N., & Gracia, J. (2016, May). Leveraging RDF graphs for crossing multiple bilingual dictionaries. In Proceedings of LREC'16 (pp. 868-876).