
# D3.2 Research Challenge Report v2

Author(s): Christian Fäth, Christian Chiarcos, Jorge Gracia, Julia Bosque-Gil, Bernardo Stearns, John McCrae, Fernando Bobillo, Philipp Cimiano, Thierry Declerck, Mohammad Fazleh Elahi, Basil Ell, Julian Grosse, Matthias Hartung, Marta Lanau-Coronas, Patricia Martín-Chozas, Elena Montiel-Ponsoda

Date: 30.06.2020

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

*D3.2 Research Challenge Report*

Deliverable Number: D3.2
Dissemination Level: Public
Delivery Date: 2020-06-30
Version: 1.0
Author(s): Christian Fäth, Christian Chiarcos, Jorge Gracia, Julia Bosque-Gil, Bernardo Stearns, John McCrae, Fernando Bobillo, Philipp Cimiano, Thierry Declerck, Mohammad Fazleh Elahi, Basil Ell, Julian Grosse, Matthias Hartung, Marta Lanau-Coronas, Patricia Martín-Chozas, Elena Montiel-Ponsoda

**Document History**

| Version Date | Changes | Authors |
|---|---|---|
| **2020-04-16** | **Initial document** | **Christian Fäth** |
| **2020-06-11** | **Added partner contributions** | **all partners** |
| **2020-06-12** | **Revisions for Reviewer's preliminary version** | **Christian Fäth Julia Bosque-Gil** |
| **2020-06-26** | **Minor additions** | **all partners** |
| **2020-06-30** | **Final revision** | **Christian Fäth** |

# Table of Contents

# D3.2 Research Challenge Report v2

## 1. Introduction

The Prêt-à-LLOD project aims at creating a data value chain (Figure 1) for Linguistic Linked Open Data (LLOD)[1] to be used across industrial sectors within the emerging Digital Single Market in Europe. Working with linguistic data can be a very time-consuming process. Discovering resources across existing repositories and endpoints is not the only challenge a user is faced with. After overcoming the hurdles of licensing and gaining access to the required datasets, heterogeneous data models, formats and annotation schemes with varying levels of detail may require complex transformation and linking processes in order to extract the pieces of information relevant to a specific research topic. Furthermore, existing Natural Language Processing (NLP) tools require very specific input data making intermediate transformation steps necessary within complex workflows.

Work Package 3 “Transforming, Linking and Workflows for Language Resources” (WP3) of Prêt-à-LLOD tackles these challenges by developing software components for each of them and placing them in the context of four industrial pilot projects to evaluate their usability. Our data and tools rely on Semantic Web standards such as RDF[2] and OWL[3].

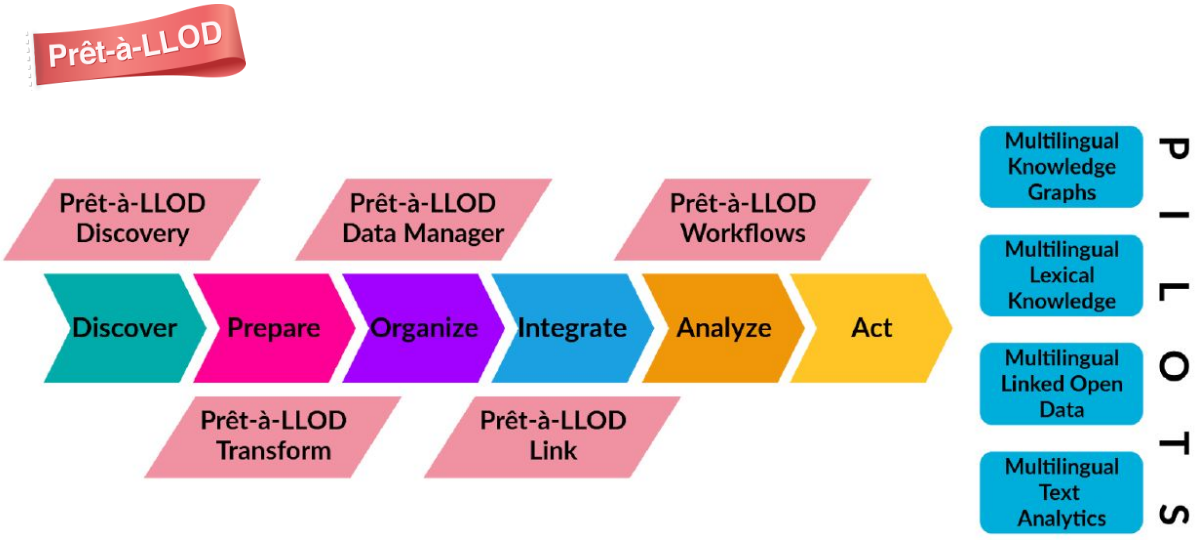


**Figure 1: Prêt-à-LLOD data value chain**

While the development of vocabularies and community standards as well as the discovery and license management of linguistic resources is pursued in other work packages (mainly Work Package 5 “Language Resource and Service Sustainability”, henceforth WP5), within

[1] For details about the Linguistic Linked Open Data cloud as a subgraph of the LOD cloud, see http://linguistic-lod.org/
[2] Resource Description Framework (RDF): https://www.w3.org/RDF/
[3] Web Ontology Language (OWL): https://www.w3.org/2001/sw/wiki/OWL

WP3 three research challenges are tackled by three respective tasks, each responsible for one software component:

- **Task 3.1**, **Prêt-à-LLOD Transform** addresses the challenge of "*Transforming language resources and language data*". Methodologies are developed for the transformation of language resources and language data into LLOD representations.
- **Task 3.2**, **Prêt-à-LLOD Link** addresses the challenge of "*Linking conceptual and lexical data for language services*". Novel (semi-)automatic methods are studied that aim at establishing links across multilingual LLOD datasets and models.
- **Task 3.3**, **Prêt-à-LLOD Workflows** addresses the challenge to create "*Workflows for Portable and Scalable Semantic Language Services*". A protocol, based on semantic markup, is developed to enable language services to be easily connected into multi-server workflows.

The primary software component for data transformation is **Fintan** (Section 2.2), the Flexible Integrated Transformation and Annotation eNgineering platform. It allows to integrate RDF converters for various input formats and combine them with stream-based graph transformation for building complex transformation pipelines. For establishing complex links between multilingual resources at the conceptual level, lexical level, or between the conceptual and lexical levels (lexicalisation), we provide a set of services which we aim to deploy through the **Naisc** platform currently developed in the context of Task 3.2, as well as through a lexicalisation component (Section 3.3). We furthermore introduce **Teanga** (Section 4) as our primary workflow management tool for NLP services.

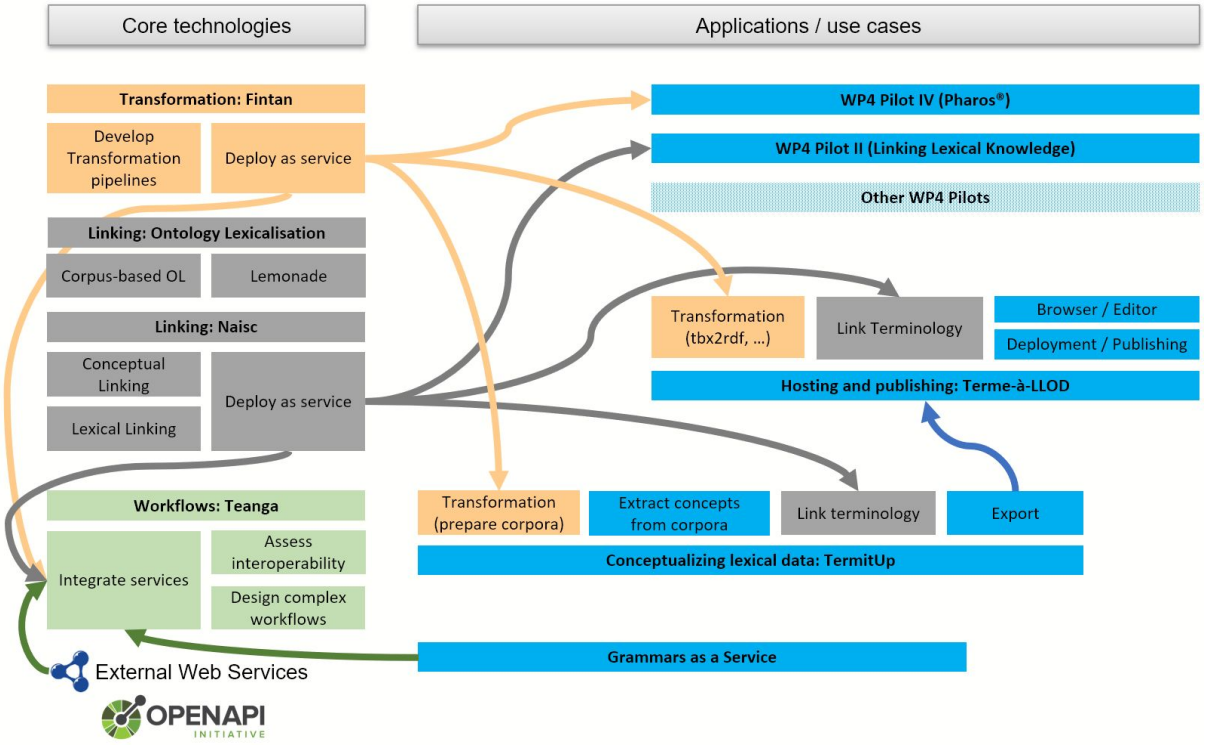


**Figure 2: Interaction of Prêt-à-LLOD software components**

In addition to these **core technologies**, in this report we will additionally focus on how these software components are planned to interact with those from our research and industry partners. For this purpose we also introduce a set of **applications** which build upon these core technologies and also contribute to their pool of services. Thus, not only do they aid directly the development but they also provide valuable insights as early-adopting **use cases**. Figure 2 provides a general interaction scenario.

While both TermitUp (Section 3.8) and Terme-à-LLOD (Sections 2.5 and 3.7) inherently provide linking services on conceptual level through Naisc, TermitUp further focuses on extracting terminologies from corpora and therefore also contributes to the ontology lexicalisation level (A). In order to add support for additional corpus formats, we are also assessing Fintan pipelines as additional preprocessing steps for TermitUp. Terme-à-LLOD, on the other hand, will provide its TBX2RDF converter as a Fintan module, and has an additional focus on aiding the publication process as Linked Data. Since both Fintan and Naisc will be able to export services to Teanga in the future, transformation and linking components will also become available for building NLP workflows.

The Business Pilots in Work Package 4 (WP4 “Pilots”) and the applications may use Fintan for adding support for multiple input formats or Naisc for implementing linking features. We already established direct collaboration specifically with some of the Pilots. Specifically, Semalytix has been evaluating the Apertium dataset, a family of bilingual dictionaries converted to RDF using Fintan (Section 2.4), for application in Pilot IV (“Multilingual Text Analytics for Extracting Real-World Evidence in the Pharma Sector”), and Oxford University Press (OUP) is collaborating on the development of Fuzzy *lemon* (Section 3.6), a framework and logic for chaining successively linked lexical semantic relations, for Pilot II (“Linking lexical knowledge to facilitate rapid integration and wider application of lexicographic resources for technology companies”).

The three tasks, Transform, Link and Workflows, alongside their inherent software components and use cases, are to be addressed in Sections 2, 3 and 4, respectively.

# 2. Transformation [T3.1]

## 2.1 Motivation

In order to prepare resources for use within the Prêt-à-LLOD project, and especially in order to fulfill the project goals of supporting 50 input formats and making available 1000 resources as LLOD, Task 3.1 aims at creating a generic framework for transforming resources to RDF. The main challenges herein stem from the vast amount of heterogeneous resources to be dealt with in the project and the industrial pilots. Since a second goal is the normalization of language resources to predefined target formats and existing community standards (WP5, Task 5.1 “Vocabularies and Interface Specifications”), the transformation goals are not only subject to quantitative assessment but also must be able to meet qualitative requirements.

One feasible approach would be the creation of a monolithic but mostly generic converter which is able to produce baseline RDF for use in further tasks. This would have the advantage of easily being able to meet quantitative requirements but also have the risk of lacking data quality. Apart from that, converters like this already exist, e.g. CSV2RDF (Tandy et al., 2015), R2RML (Das et al., 2012) but their output is not very easily adoptable for linguistic use cases. Furthermore, to some extent these formats tend to reflect the original data storage paradigm of their source material within RDF and therefore generate unnecessary overhead while not taking sufficient advantage of the native graph layout.

Instead, since data types relevant for Prêt-à-LLOD mainly comprise dictionaries and corpora our primary target models are OntoLex-Lemon (McCrae et al., 2012; Cimiano et al., 2016) as well as NIF (Hellmann et al., 2013), CoNLL-RDF (Chiarcos and Fäth, 2017) and POWLA[4] (Chiarcos, 2012) respectively. These formats are well established and widely used within the LLOD community. This is aiding in creating resources which are both linguistically rich and reusable across work packages and beyond the scope of the project.

However, the transformation steps needed to fully convert existing heterogeneous resources into these target models are far more complex than the simple RDF rendering approaches described above. By creating several monolithic but highly resource-specific converters, we can easily meet qualitative requirements but might never be able to catch up on the quantitative goals.

We therefore decided upon a more flexible approach: With the **Flexible and Integrative Transformation and Annotation eNgineering (Fintan)** platform (Fäth et al., 2020), we are developing a modular framework of interoperable transformation steps to combine the best of both worlds by creating simple baseline RDF converters (or integrating existing ones) and enriching their output using graph transformation to meet the qualitative requirements of desired target models, thus increasing reusability. Certain source material might only need

[4] POWLA has recently become one of the most relevant formats for representing corpora with more complex semantic or syntactic structures. (Cimiano et al., 2020)

some intermediate steps or minor adjustments to existing modules to be transformed into valid RDF resources. On the other hand, only some additional graph transformation steps might be necessary to support additional target models.

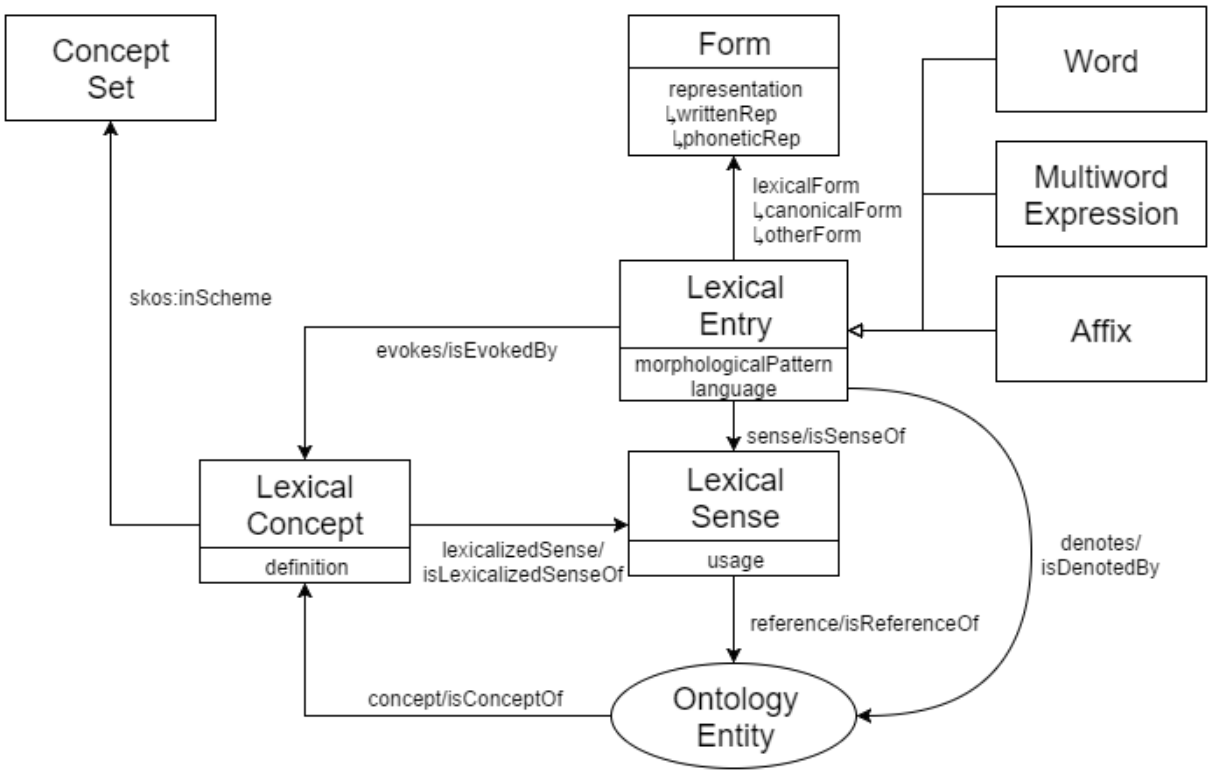


**Figure 3: The core module of OntoLex-Lemon: Ontology Lexicon Interface.**
**(Figure taken from https://www.w3.org/2016/05/ontolex/.)**

In the last report (D3.1), we presented a set of case studies on resource transformation and layed out the theoretical cornerstones for the Fintan platform. In this report, we introduce the first prototypical implementation of Fintan and also put it into the context of multiple resource transformation and publication efforts, some of which have evolved from the case studies in the last report. Since most of them are concerned with lexical data, they heavily rely on OntoLex-Lemon as a data model. The core model, linking a `LexicalEntry` to its respective `LexicalSense` and `Form`, is shown in Figure 3.

- The **PanLex**[5] dictionaries present a vast set of resources encompassing over 2,500 dictionaries, 5,700 languages, 25 Million words and 1.3 Billion translation pairs, now available as OntoLex-Lemon (cf. Figure 3), converted from CSV dumps of a relational data model (cf. Section 2.3).
- The **Apertium**[6] dictionaries encompass 44 languages and 53 translation sets. The original XML data has been converted to OntoLex-Lemon and the resulting RDF has already been used in a WP4 Pilot (cf. Section 2.4).
- The **TBX2RDF** converter used in **Terme-à-LLOD** evolved from the case study: *Transforming terminological data*. It has been implemented to create OntoLex-Lemon

[5] https://panlex.org/
[6] https://www.apertium.org/

representations of terminologies, and used for the conversion of the terminologies developed by the Centrum Voor Terminologie (CvT) in Gent - **GENTERM**[7] covering over 6000 terms in 2 languages, and the **I**nter**a**ctive **T**erminology for **E**urope - **IATE**[8] covering over 6 Million terms in over 25 languages (cf. Section 2.5).

- Three data sets included in the **Open Multilingual Wordnet**[9] (OMW), resulting in numerous instances of `ontolex:LexicalConcept` for French (59,091), Italian (15,553) and Spanish (38,512), and the **Exeter Latin Wordnet**[10] comprising 73,949 lexical entries and 1,219 morphological rules in the OntoLex-Lemon representation (cf. Section 2.6). The OMW data is also linked with morphological data of the Mmorph dataset (Petitpierre and Russell, 1995) covering over 2 Million morphological entries in 6 languages (cf. Section 2.7).

## 2.2 Fintan Prototype

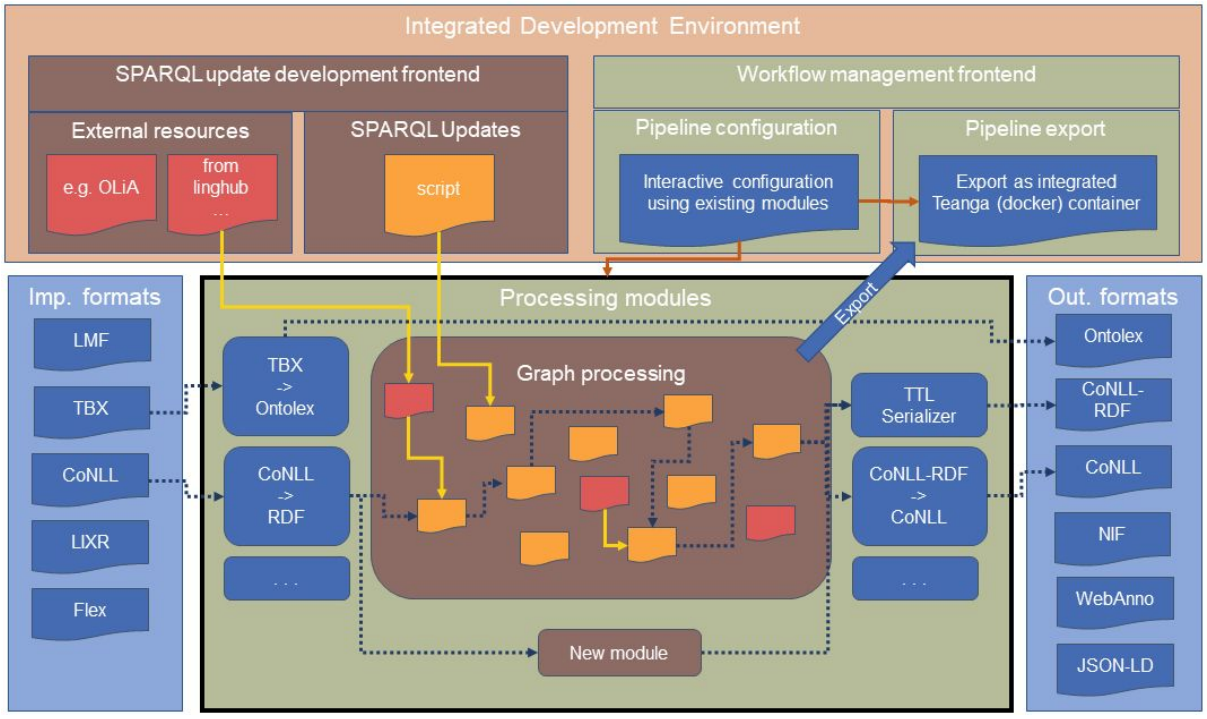


**Figure 4: Fintan platform**

The Fintan platform, as described in the last report, is an effort of combining existing converter frameworks with stream-based graph transformation (as originally developed in the context of CoNLL-RDF) and a workflow management engine in order to create integrated transformation pipelines for various input and output formats. By making data conversion modular, we increase the reusability of granular transformation steps. By choosing a stream-based approach, specifically for processing RDF data, we also address scalability

[7] https://cvt.ugent.be/
[8] https://iate.europa.eu
[9] http://compling.hss.ntu.edu.sg/omw/
[10] See https://latinwordnet.exeter.ac.uk/ and Fedriani et al. (2020)

issues typically arising with large scale datasets on triple stores. The general layout of Fintan, as shown in Figure 4, depicts our implementation plans:

- An interoperable pool of processing modules including:
  - External converter tools
  - Graph processing steps
  - Serializer tools and data writers
- A development environment for SPARQL updates and transformation workflows
- A means of deploying specific converter pipelines as integrated Docker[11] containers

Within the project, the Fintan platform will not provide data through a black box maintained by GUF[12] but will also allow other project partners, especially from WP4, to contribute their existing converters as modules or to make adjustments to pipelines on their own. In its final state, the system will be able to transform multiple kinds of resources into common data models applying task specific annotations. Additionally, converter pipelines can be deployed to Task 3.3's Teanga platform to make them publicly available, thus enabling RDF-based NLP modules to directly feed on generic resource types, further increasing scope and applicability for long-term use by a wider audience.

### 2.2.1 Architecture and Implementation

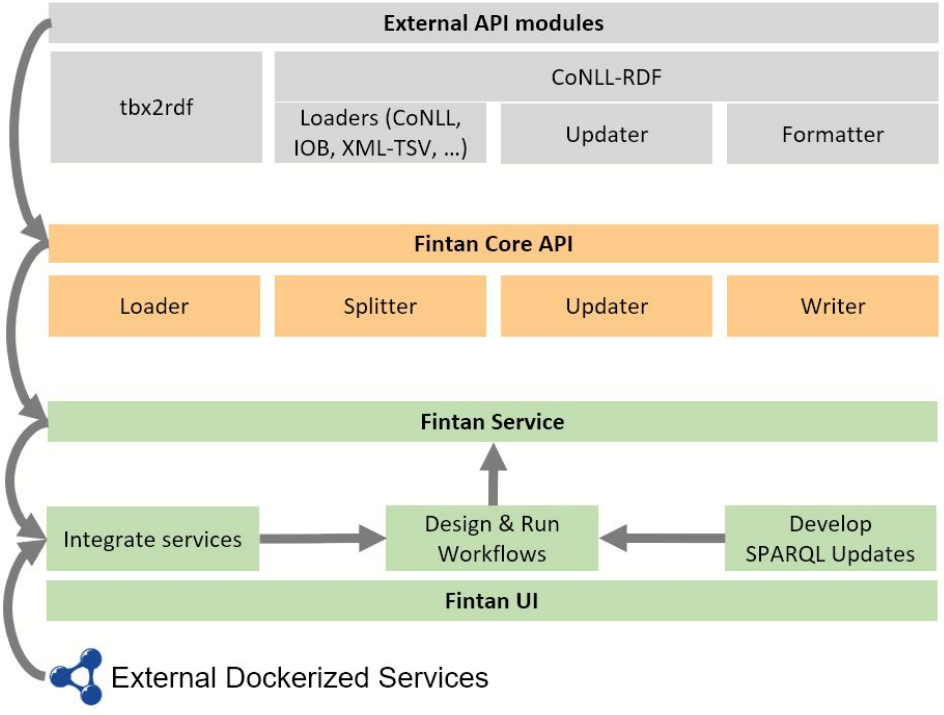


**Figure 5: Fintan architecture**

[11] http://www.docker.com/
[12] The Prêt-à-LLOD project partner Johann Wolfgang Goethe-Universität Frankfurt a. M.

In order to achieve these implementation goals, we designed a modular architecture which aids a decentralized development process (cf. Figure 5). With Fintan inheriting some of the core functionalities from CoNLL-RDF, we decided to keep the general design paradigms intact and stay within the Java-based environment including the Apache Jena API[13] for graph transformation. However, since CoNLL-RDF is designed as a self-contained tool which is focused on TSV-based input formats, Fintan establishes an additional abstraction layer.

The **Fintan Core API** defines four distinct interfaces which encapsulate methods for data streaming and ensure their interoperability. They are backed by an ontological design pattern (cf. Figure 6) for distinguishing central types of transformation steps:

- `Loader` modules read any kind of input data and write back RDF.
- `Splitter` modules read unstructured serializations of RDF data and write back segmented RDF data for stream-based graph transformation.
- `Update` modules (in Figure 6) are performed by `Updater` classes (in Figure 5), which read streams of RDF data, process multiple data segments in parallel and then write back the transformed RDF data. `Update` modules are defined by a set of SPARQL scripts and additional resources to be permanently side-loaded for specific processing steps, e.g. inferring annotation schemes by using OLiA[14] (Chiarcos and Sukhareva, 2015) or entity linking to DBpedia[15] (Auer et al., 2007) entries .
- `Writer` modules read RDF data and either create serializations (e.g. Turtle[16], RDF/XML) or export the data to other output formats.

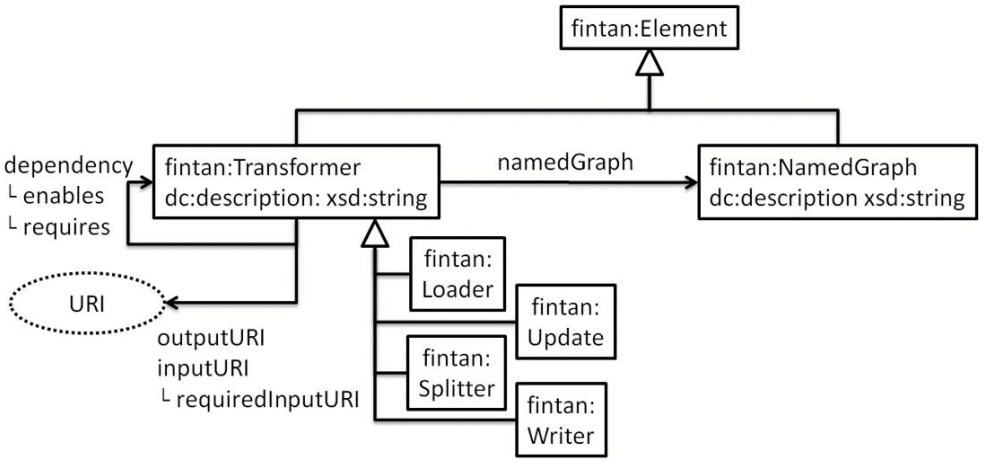


**Figure 6: Fintan ontology prototype**

While the `Updater` and `Splitter` classes are central functionalities provided by the Fintan Core, many of the `Loader` and `Writer` modules can be external programs published in other repositories. In order for them to become Fintan-compliant, they are encouraged to

[13] http://jena.apache.org/
[14] The Ontologies of Linguistic Annotation (OLiA) are a set of interoperable RDF/OWL models for various annotation schemes which are all linked to a central reference model, which enables a user to transform annotations between schemes.
[15] The RDF representation of Wikipedia: https://wiki.dbpedia.org/
[16] https://www.w3.org/TR/turtle/

implement the respective interfaces and make the source code available as a Maven[17] module. By using Maven dependencies, they can be directly imported into Fintan as **External API modules**. On the other hand, Fintan can be directly used within their respective toolset. The primary example for this is CoNLL-RDF, which is, at the time of writing this report, becoming refactored towards implementing the Fintan Core and will be the central module for processing TSV-based data streams. In this context, the `CoNLLStreamExtractor`, `CoNLLRDFUpdater` and `CoNLLRDFFormatter` will implement the `Loader`, `Updater` and `Writer` interfaces respectively. In addition, with TBX2RDF from Terme-à-LLOD (cf. Section 2.5) we will be able to add support for terminological input data. Also, for the Apertium dataset (cf. Section 2.4), an additional `Loader` module accepting XML files to be transformed to RDF/XML by XSLT[18] scripts has been developed. It is based on the Saxon[19] open-source edition.

All these External API modules can then be addressed and built into workflows using the **Fintan Service** and **Fintan UI,** which will be deployed as an integrated Docker container. Furthermore, we aim at the possibility to also address external OpenAPI[20]-compliant services. This will specifically be necessary for closed-source services or architectures which are incompatible with the Fintan Java API.

The user interface is a web application to be accessed from within a browser. The workflow manager is currently under development as a fork of the original Teanga UI (cf. Section 4). With Task 3.3 addressing similar challenges[21], we will be able to create a more streamlined user experience and we can benefit from a mutual exchange of features and design principles. In addition, the development of graph transformation steps will be aided by a set of tools for assessing SPARQL Updates and their dependencies. One of these tools has recently been published as **SparqViz**, a tool for creating visualizations of SPARQL queries and updates using GraphViz and the underlying dot format. A REST API accepts a SPARQL query and outputs a dot file and an SVG image. An exemplary lightweight editor website is included with the latest stand-alone version.[22]

### 2.2.2 Transforming the Universal Morphology using Fintan

In a case study, conducted for the last report, we had been assessing the capabilities of stream-based graph transformation by converting the Universal Morphology TSV datasets into OntoLex-Lemon, solely relying on CoNLL-RDF. In this section we will provide a quick overview of the case study and have a look at how it looks like in Fintan.

The Universal Morphology (UniMorph) project provides a universal way to annotate morphological data in a universal schema. This allows an inflected word from any language

[17] Apache Maven is a tool for software project management. Software modules can be deployed to a central repository by independent developers and imported into a project by a dependency structure. https://maven.apache.org/
[18] https://www.w3.org/TR/xslt/
[19] http://saxon.sourceforge.net/
[20] http://www.openapis.org/
[21] Specifically, the integration of external services is one of the main goals of the Teanga platform.
[22] https://github.com/acoli-repo/sparqviz/

to be defined by its lexical meaning, typically carried by the lemma, and by a rendering of its inflectional form in terms of a bundle of morphological features from the UniMorph annotation schema (Sylak-Glassman et al. 2015). In context of LLODifier[23], a larger toolset for transforming linguistic data into a shallow Linked Data representation, we already provided a transformation suite for mapping UniMorph data to OntoLex-Lemon (Chiarcos et al. 2018) by using CoNLL-RDF. Though CoNLL-RDF was originally built for transforming CoNLL[24] corpora into an isomorphic RDF representation, it was applicable to the dictionary-type UniMorph data out-of-the-box mainly because of their simple layout and TSV structure. This made it an ideal case study for testing CoNLL-RDF's streamed graph transformation capabilities on different types of data.

In CoNLL, each line represents a token and its annotations, separated by tabs. Empty lines mark the borders of a sentence. In UniMorph, each line represents a dictionary entry, also with the annotations separated by tabs. Therefore, CoNLL-RDF treats UniMorph entries as sentences, while sentence borders can easily be injected by adding empty lines. In order to use CoNLL-RDF as a converter, three processing steps were necessary:

- Transform the TSV data to CoNLL-RDF (reflecting a `Loader` in Fintan)
- Transform CoNLL-RDF to OntoLex-Lemon with SPARQL (reflecting an `Update` in Fintan)
- Convert the annotations to OLiA by loading the respective annotation model and performing another SPARQL update (reflecting another `Update` in Fintan)

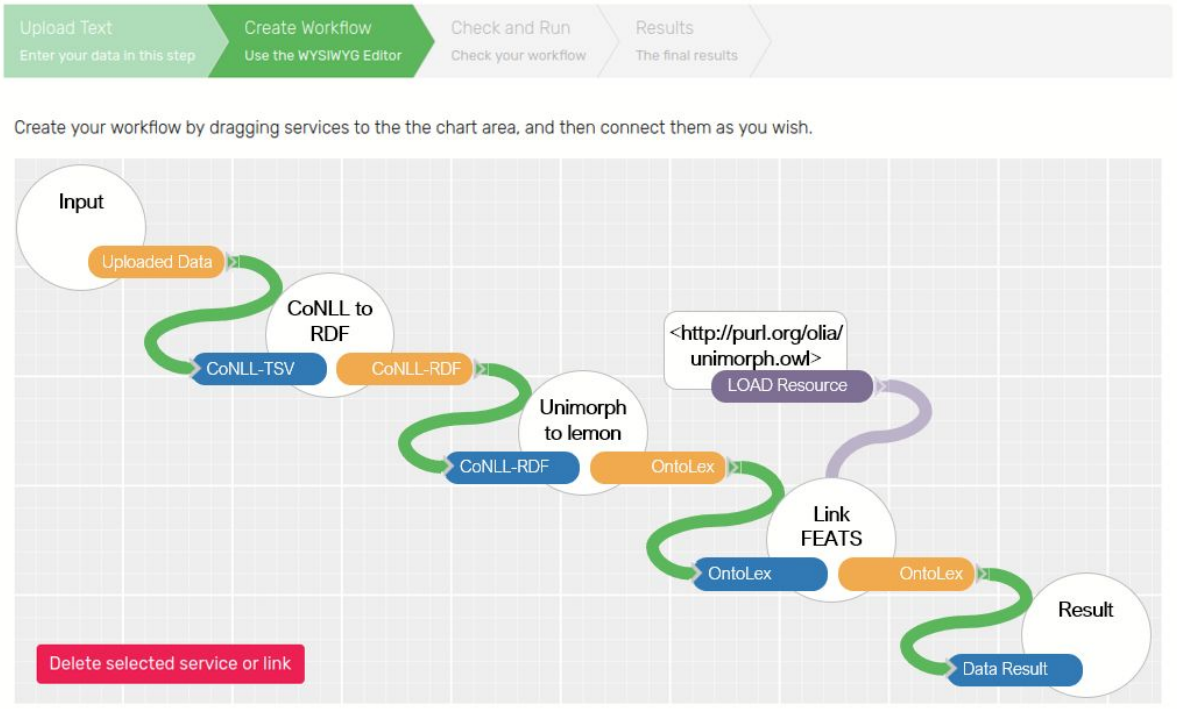


**Figure 7: The UniMorph pipeline as a Fintan workflow**

[23] The LLODifier tools are available at https://github.com/acoli-repo/LLODifier/

[24] CoNLL as a data format commonly refers to a family of TSV formats representing corpora and various annotations. Most of these formats have originally been designed in the context of specific shared tasks held by the Special Interest Group on Natural Language Learning: https://www.conll.org/

Since CoNLL-RDF is an External API module of Fintan, this pipeline can be rendered in the Fintan workflow manager (cf. Figure 7).

In addition, the SPARQL updates can be rendered in SparqViz (cf. Figure 8). The `INSERT` and `DELETE` statements are marked as green and red boxes respectively. Individual graphs addressed in SPARQL are rendered in labeled boxes. Triples are rendered with subjects and objects as nodes and properties as directed, labeled edges. Nodes occurring in multiple subgraphs are repeated for each subgraph and connected by light blue edges, in order to improve readability. Variable nodes are rendered as circles, while explicit nodes are rendered as boxes, thus attributing to the importance of their distinction in SPARQL. In the current version, `FILTER` statements are rendered in dotted boxes without further graphical post-processing. However, nodes addressed in the filters are connected by rounded edges.

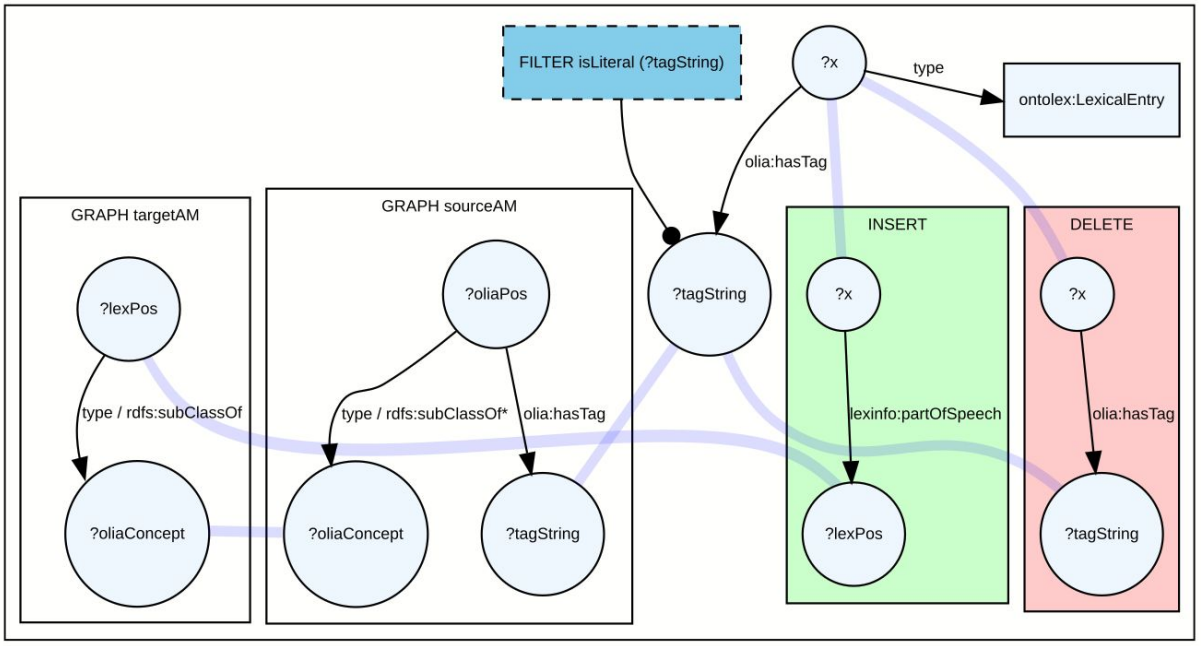


**Figure 8: Mapping UniMorph annotations to OLiA with SparqViz**

In order to assess the scalability of the pipeline, we performed the transformation in three different configurations:

- **en-bloc**: With this basic approach, we transformed the input file as a whole. CoNLL-RDF loads the whole file in an in-memory dataset and applies the SPARQL update, just like the ARQ command line tool which comes with Apache Jena. However, larger files will run into memory limits. Even when backed by a triple store, e.g. Fuseki, they become increasingly slower to process due to heavy I/O activities.
- **line-wise**: Using an old implementation of CoNLL-RDF, which did not yet support multithreading, we split the input data by line breaks, leading to each dictionary entry to be treated as an independent dataset to be processed. This eliminated the scalability issues regarding memory consumption, but led to decreased performance. In addition, the UniMorph linking model, linking the annotations to OLiA, had to be prefetched for each dataset.

- **line-wise multithread**: Using the most recent implementation of CoNLL-RDF, which serves as the basis for Fintan, we performed the transformation again. This time the annotation model was precached and the UniMorph data segments were distributed across multiple threads, to be executed in parallel. This allowed us to transform data of any size with highly increased processing speed.

| | **En-bloc** | **Line-wise single thread** | **Line-wise multithread** |
|---|---|---|---|
| **Transformation time** | 6m24s | 12m34s | 3m00s |
| **# of OLiA loads** | 1 | 33484 | 1 |

**Table 1: Time and size comparison of the three processing approaches.**
(Performed on a 3.79 GHz i5 quadcore with 16 GB of memory.)

Table 1 shows the results of our experiment. While the *en-bloc* approach is fairly fast, it is susceptible to size limitations. The *line-wise* processing without parallelization and precaching of external resources is much slower than the other two configurations due to the constant loading and unloading of the OLiA models. The *line-wise multithread* approach not only removes the loading penalty by precaching, it also displays that our stream-based graph transformation outperforms established database engines (here Apache Jena) in specific use cases. In the *en-bloc* approach, transformation is achieved by a single SPARQL update executed on the whole dictionary while the database engine is distributing memory and processing power. In the multithread approach we distribute processing power across all cores executing the same update multiple times but only on a single `LexicalEntry` resulting in much higher performance.

### 2.2.3 CoNLL Tree Extensions

In addition to the UniMorph case study, we also extended CoNLL-RDF with support for additional hybrid TSV dialects:

- The CoNLL bracketing format (e.g. for syntax trees)
- TSV formats augmented with XML markup

While the bracketing format is commonly used in treebanks such as the Penn Treebank (Marcus et al., 1993), typical examples for XML-augmented TSV formats are SketchEngine (Kilgarriff et al., 2014) and the Corpus Work-bench (Evert and Hardie, 2011).

Since the CoNLL-RDF vocabulary is limited to word level annotations, dependency syntax and semantic roles, the syntactic tree structure is rendered in POWLA (Chiarcos, 2012):

- Non-Terminal nodes are rendered as `powla:Node`.
- Hierarchic relations between nodes are rendered using the properties `powla:hasParent` and `powla:next`.

The Tree Extensions have recently been published by Chiarcos and Glaser (2020). An overview of the POWLA syntax is shown in Figure 9.

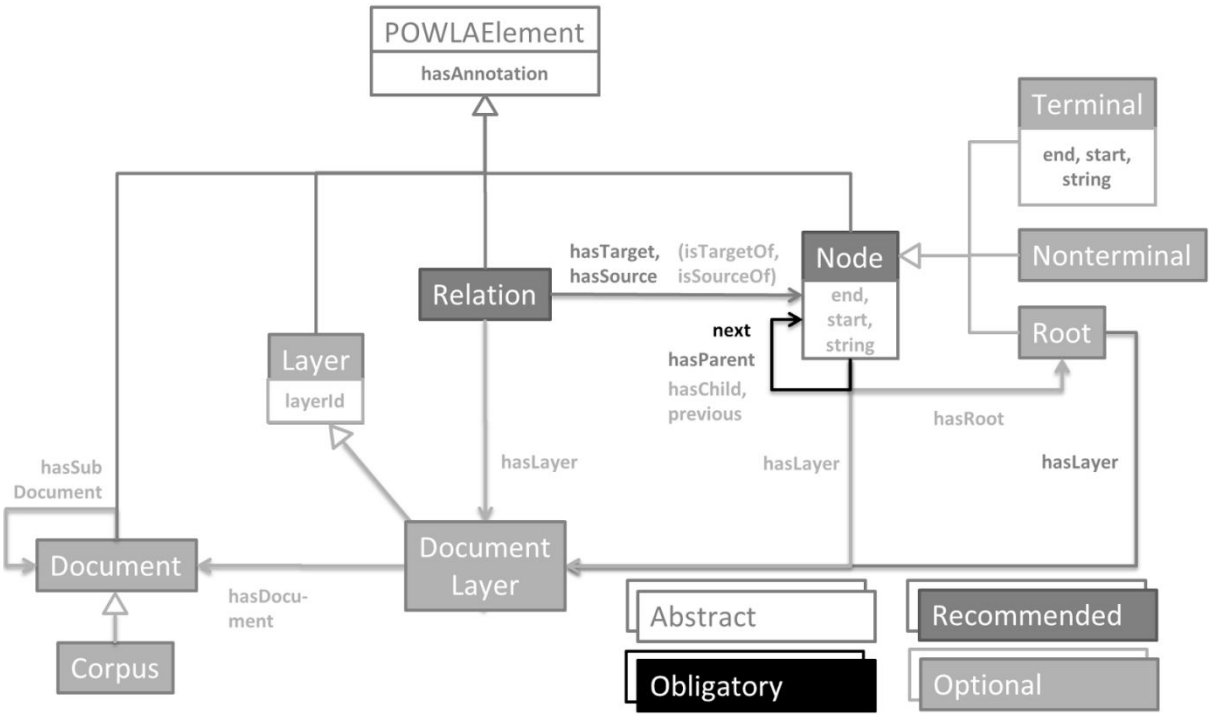


**Figure 9: The POWLA vocabulary**

### 2.2.4 State of Integration in the Context of Prêt-à-LLOD

The development of Fintan is still an ongoing effort. While all aforementioned features do exist, they are still not integrated into a full tool. The SparqViz and workflow manager UIs are still unconnected stand-alone applications and the API structure is still unpublished. Nevertheless, Fintan has already been used in transformation pipelines listed in subsequent sections. For Terme-à-LLOD, the TBX2RDF component will be integrated as an external API module in the near future. For the PanLex (Section 2.3) and Apertium pipelines (Section 2.4), a Fintan workflow using the XSLT transformation module and the `Updater` is in the process of publication. Furthermore, Fintan as a transformation tool is being assessed for usage within Pharos®, among other WP4 pilots and also for preprocessing corpora in TermitUp.

## 2.3 PanLex and the ACoLi Dictionary Graph

The ACoLi Dictionary Graph[25] (Chiarcos et al., 2020a) presents an effort to create a large collection of multilingual dictionaries represented in canonical OntoLex-Lemon and an additional simple TSV format to be used in TIAD (cf. Section 3.5) or other NLP-related tasks. While some of the dictionaries have been compiled in the context of other projects, in Prêt-à-LLOD we made two very large additions to the graph: an updated version of the Apertium family of dictionaries, which are described in Section 2.4, and the PanLex[26] database.

[25] https://github.com/acoli-repo/acoli-dicts/
[26] https://panlex.org/

At the time of writing, Panlex consists of 2,500 dictionaries, 5,700 languages, 25,000,000 words and 1,300,000,000 translations, as listed on their website. Apart from a web UI, the data can also be downloaded as CSV or JSON dumps under a CC0[27] license. In addition to that, an RDF edition of PanLex had already been designed by Westphal et al. (2015). However, it did not employ the OntoLex-Lemon W3C specification as published by Cimiano et al. (2016) and the data is no longer publicly available.

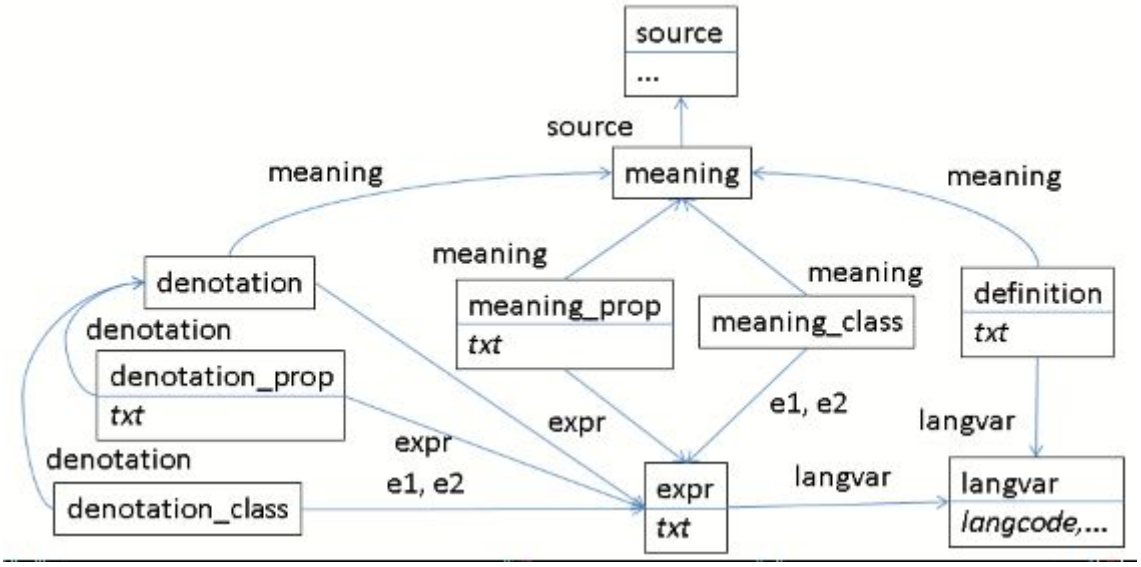


**Figure 10: PanLex data model**

The original data model of PanLex (cf. Figure 10) follows a tabular scheme:

- Entries of the *source* table correspond to a respective source document and contain its respective meta-information. They can be reflected as `lime:Lexicon`.
- The *meaning* table aggregates different translations of the same meaning. In OntoLex-Lemon, this information can be rendered as `ontolex:LexicalConcept`.
- The *definition* table optionally covers descriptive information about individual meanings.
- The *meaning_prop* table optionally provides pointers to external identifiers or definitions of a corresponding meaning.
- The *meaning_class* table optionally provides concept-level annotations from the controlled PanLex vocabulary.
- The *denotation* table contains information about actual dictionary entries and provides pointers to their written representations encoded in the *expr* table. It can therefore be mapped as `ontolex:LexicalEntry` along its `ontolex:canonicalForm`.
- The *denotation_prop* table optionally provides free-text entry-level annotations to be filled into property slots from the controlled PanLex vocabulary. In our OntoLex-Lemon representation, the properties are mapped to datatype properties in the `panlex:` namespace.

[27] https://creativecommons.org/share-your-work/public-domain/cc0/

- The *denotation_class* table optionally provides entry-level annotations. Both the properties and their objects are from the controlled PanLex vocabulary. In our OntoLex-Lemon representation, they are therefore mapped as object properties pointing to individuals within the `panlex:` namespace.
- The *expr* table covers multiple types of expressions. They can either correspond to annotations or property names referenced by the **_prop/class* tables or actual textual representations referenced by the denotation table (i.e. `ontolex:LexicalForm ontolex:writtenRep "expr"`).
- The *langvar* table contains the respective ISO 639-3 language code of an *expr*ession or *definition*. Since in the *expr* table all controlled expressions are marked by the artificial language code "art", actual written representations can be easily distinguished.

The resulting OntoLex-Lemon representation of the PanLex data is depicted in Figure 11.

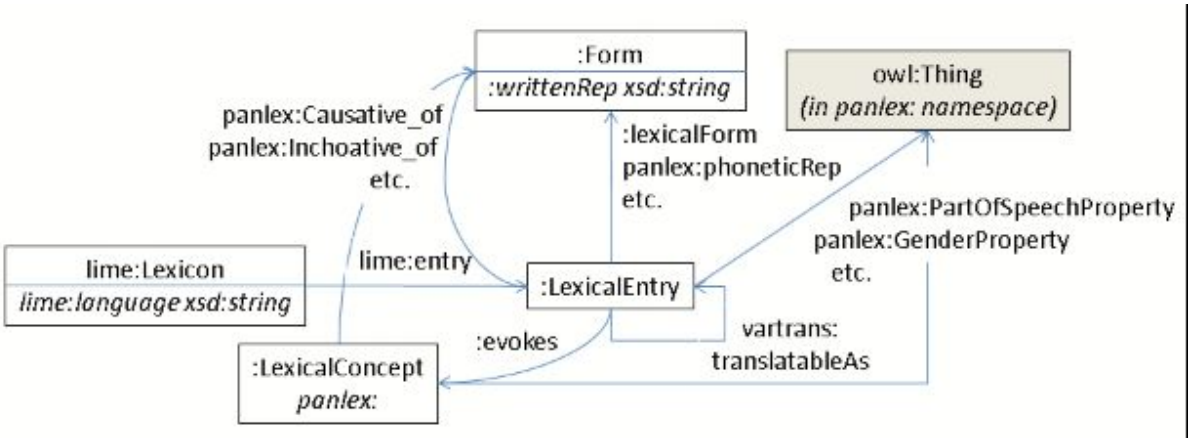


**Figure 11: OntoLex-Lemon representation of PanLex dictionaries**

The transformation workflow consists of the following steps:

- The PanLex database dumps represent the original tabular data model as a (zipped) folder structure with multiple CSV files per individual resource. Employing a custom built Java class, these are transformed into raw XML files.
- Using XSLT transformation, these XML files are subsequently transformed into valid RDF/XML serialization of the OntoLex-Lemon model.
- In addition, using a SPARQL query, a TSV representation of translation pairs is created and split into bilingual dictionaries (primarily intended for use in the TIAD shared task).

A more detailed description of the pipeline is provided by Chiarcos et al. (2020a). The initial publication predated the Fintan release and directly employed Saxon and ARQ for XSLT and TSV generation. Therefore, the transformation process required very powerful servers, specifically to process some of the larger datasets, which can be over 80GB in size. The Fintan XSLT and TSV modules introduced above will address these limitations in future releases. Furthermore, we aim at employing Fintan's graph transformation for harmonizing the vocabularies used in the ACoLi dictionaries. For Apertium, we already implemented a

mapping to LexInfo (Cimiano et al., 2011) parts of speech (POS), which will be described in the following section.

## 2.4 Apertium dictionaries in Pharos®

Apertium[28] is a free/open-source machine translation platform (Forcada et al. 2011) originally designed for translation between closely related language varieties but expanded to deal with more divergent language pairs. The Apertium platform mostly relies on the use of symbolic methods and currently includes around 50 language pairs.[29] It provides NLP components for many languages, as well as transfer rules and bilingual dictionaries for their respective translation.

A subset of such a family of bilingual dictionaries developed in Apertium was converted to the ISO standard LMF (Francopoulo et al. 2006) as part of the METANET4U Project.[30] From that subset of Apertium dictionaries, only the entries in Apertium which were annotated as nouns, proper nouns, verbs, adjectives and adverbs were considered (from a long list of heterogeneous POS tags present across datasets). This LMF subset constituted the basis for the first RDF representation of the Apertium dictionaries (Gracia et al. 2018) developed by Universidad Politécnica de Madrid (UPM) and Universitat Pompeu Fabra (UPF), which was released as LLOD.[31] We will refer to it as **Apertium RDF v1.0.** This RDF version of the Apertium dictionary data was based on the original *lemon*[32] model, the predecessor of the OntoLex-Lemon, and its translation module.

Given that Apertium RDF v1.0 only covered the language pairs for which an LMF version was available, we decided to expand Apertium RDF by accessing the Apertium source data directly and converting them into OntoLex-Lemon. An initial converter was developed to generate RDF from *all language pairs* in the Apertium family (Chiarcos et al. 2020a). Following the approach adopted in Apertium RDF v1.0, for each language pair in a source Apertium dictionary three files are generated, one for each dictionary (source and target), and the third one for the translation relations between the senses of lexical entries. In addition, to represent the POS tags of Apertium as RDF, a URI in the Apertium namespace is associated with each tag, using the string value of every tag as its local name, e.g. `apertium:n` for the tag `n` (noun). The resulting data model is shown in Figure 12.

[28] https://www.apertium.org/
[29] See http://wiki.apertium.org/wiki/Main_Page
[30] http://www.meta-net.eu/projects/METANET4U/
[31] http://linguistic.linkeddata.es/resource/id/apertium
[32] https://lemon-model.net/

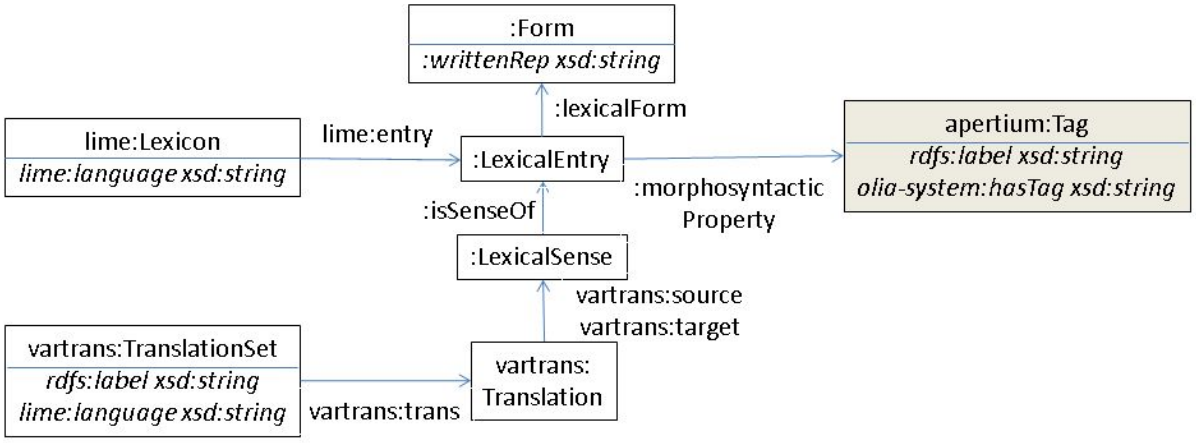


**Figure 12: OntoLex-Lemon representation of Apertium dictionaries**

However, the tags in Apertium to indicate POS and other morphosyntactic properties are not normalised. In order to allow for the integration of the Apertium dictionaries among themselves and with external resources, a normalisation process was necessary. To that end we have mapped the Apertium POS tags to LexInfo, resulting in an homogeneous tagging across all the Apertium dataset family and facilitating its querying and reuse. The mapping[33], in the form of a CSV file and performed manually, provides *predicate* - *object* pairs for each of those Apertium tags acting as object of the `lexinfo:morphosyntacticProperty` statement. (e.g. **`apertium:vblex`**, `lexinfo:morphosyntacticProperty`, **`lexinfo:verb`**). The initial number of Apertium categories identified as POS was 104, which were mapped into 28 different LexInfo categories. In this way, the RDF can be updated via SPARQL updates in Fintan, which results in the new version **Apertium RDF v. 2.0**, covering 44 languages and 53 translation sets, and linked to LexInfo POS tags.

Similar to PanLex, the full transformation workflow for Apertium consists of the following steps:

- The Apertium data dumps consist of multiple XML files which are transformed into OntoLex-Lemon using XSLT transformation
- A SPARQL update employs an RDF representation of the LexInfo-mapping table to infer the LexInfo annotations.
- In addition, using a SPARQL query, a TSV representation of translation pairs is created.

At the time of writing, this pipeline has already been partly implemented in the yet-to-release Fintan prototype. Figure 13 shows a representation of the pipeline in the workflow manager.

[33] Available at https://github.com/sid-unizar/apertium-lexinfo-mapping

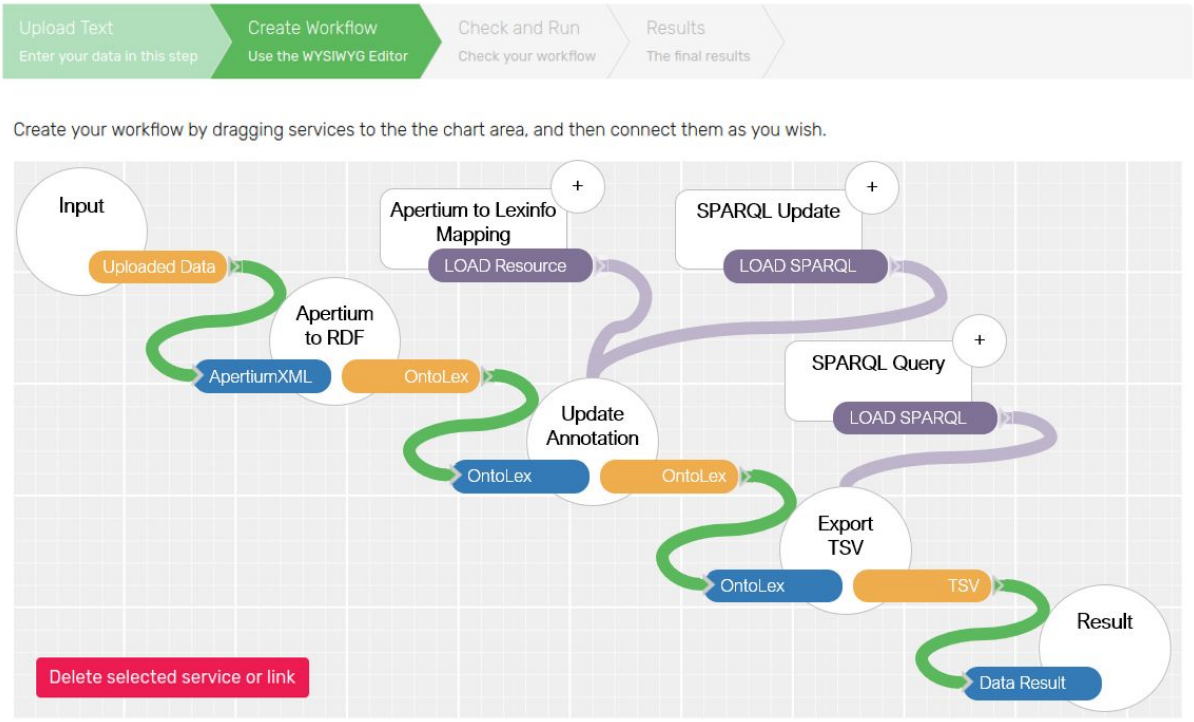


**Figure 13: Apertium conversion workflow in Fintan**

In the context of WP4, the resulting Apertium data has also been applied to extend the scope of Semalytix' Pharma Analytics Platform Pharos®. The platform is designed to provide international customers from the global pharmaceutical industry with actionable real-world evidence (RWE). The process of extracting RWE heavily relies on the analysis of multilingual unstructured text such as subjective medical assessments from medical experts and patients and thus represents a complex NLP task.

Given the complex requirements arising from multiple languages of interest and domain-specific challenges, training individual NLP models per language from scratch is infeasible. Additionally, depending on the source languages, existing training data and pretrained models may be insufficient. In order to address these challenges, the Apertium data has been successfully used to transfer models for sentiment analysis from English to Spanish in a cross-lingual transfer approach based on deep learning. Capitalizing on bilingual lexical information from Apertium as seed data, this transfer learning approach was shown to be more accurate for sentiment analysis in the target language than a pipeline approach based on machine translation (Hartung et al., 2020).

In future work, this transfer learning approach will be applied to different NLP problems such as concept extraction or entity recognition in order to yield parameterizable transfer workflows for various NLP models across multiple languages in the Semalytix stack. The final goal is to be able to run a fully automated pipeline comprising the steps of (i) periodically fetching updated Apertium data, (ii) processing it in Fintan, (iii) using it as bilingual seed knowledge for transfer learning, and (iv) supplying the resulting model to Pharos®.

## 2.5 Transforming Terminologies with Fintan and Terme-à-LLOD

### 2.5.1 Terme-à-LLOD

Terme-à-LLOD (TAL) (Buono et al., 2020) is a virtualization paradigm for easing the process of transforming terminological resources into RDF and hosting them as Linked Data. The virtualization paradigm relies on three main components: a converter (A), a Virtuoso Server[34] (B) (Erling, 2012), and a Docker container (C). The benefit of such a TAL virtualization approach is that the owner of a terminology can easily publish the terminology as Linked Data without the need to understand the underlying vocabularies in detail nor of the RDF data model or how to set up a Linked Data server. Yet, the data remains under full control and can be published under a namespace to represent ownership and provenance.

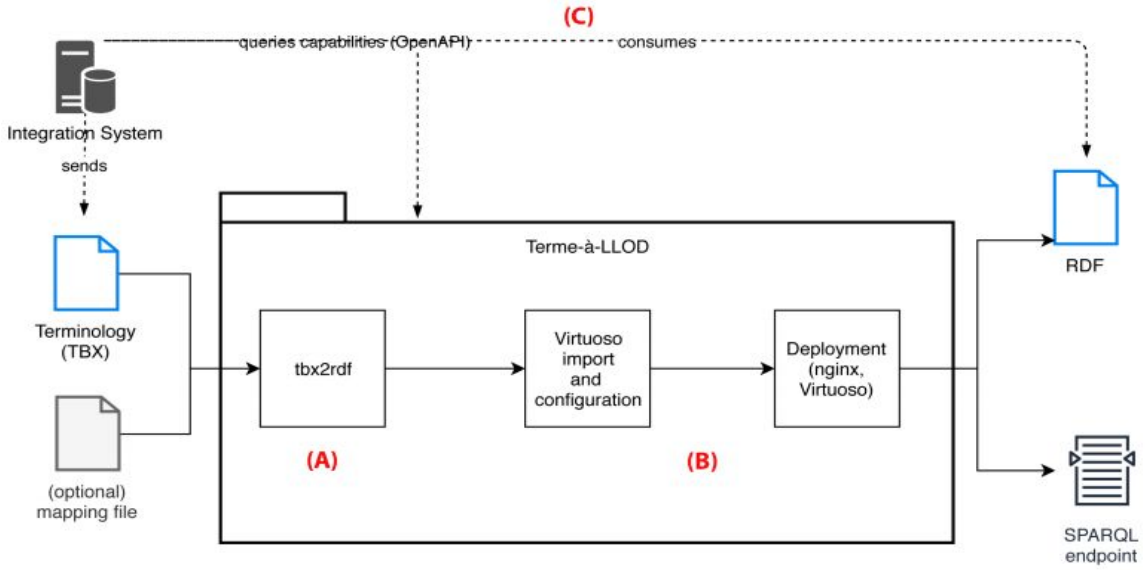


**Figure 14: Terme-à-LLOD virtualization paradigm**

The Terme-à-LLOD virtualization paradigm (Figure 14) works as follow:

- The converter element (A) of Terme-à-LLOD automatically converts the terminology of TBX format into RDF by the TBX2RDF service[35] (Cimiano et al., 2015).
- The converter produces RDF output which serves as input to a Virtuoso server (B), the second component of the TAL virtualization technology. Once the RDF output has been uploaded, the pre-installed server, which hosts the service, exposes the converted data through an endpoint which allows access to them. The server also provides a SPARQL endpoint to other services.
- The third element of the virtualization technology is a Docker container that allows to bundle components, libraries and configuration files of the TAL service and to run the service on different computing environments. Once the container is installed and instantiated, the terminological resource can be pushed via HTTP/Advanced Message Queuing Protocol (AMQP) request to the TBX2RDF converter.

[34] https://virtuoso.openlinksw.com/
[35] http://tbx2rdf.lider-project.eu/converter/

In this section, we discuss briefly the approach of the converter (A) that converts the terminology from TBX format into RDF. The linking and publishing of terminologies as Linked Data will be presented in Section 3.7.

### 2.5.2 Converting TBX Resources to RDF

```
<termEntry id="IATE-84">
  <descripGrp>
     <descrip type="subjectField">1011</descrip>
  </descripGrp>
  <langSet xml:lang="en">
     <tig>
        <term>competence of the Member States</term>
        <termNote type="termType">fullForm</termNote>
        <descrip type="reliabilityCode">3</descrip>
     </tig>
  </langSet>
  <langSet xml:lang="da">
     <tig>
        <term>medlemsstatskompetence</term>
        <termNote type="termType">fullForm</termNote>
        <descrip type="reliabilityCode">3</descrip>
     </tig>
  </langSet>
  <langSet xml:lang="nl">
     .....
  </langSet>
  <langSet xml:lang="es">
     .....
  </langSet>
  .....
</termEntry>
```

```
<http://webtentaclel.techfak.uni-bielefeld.de/tbx2rdf_iate/
competence+of+the+Member+States-en>
      a ontolex:LexicalEntry ;
      dct:language <http://www.lexvo.org/id/iso639-3/eng> ;
      tbx:reliabilityCode 3 ;
      <http://www.lexinfo.net/ontology/2.0/lexinfo#termType>
            <http://www.lexinfo.net/ontology/2.0/lexinfo#fullForm> ;
      <http://www.w3.org/ns/lemon/lime#language>
            "en" ;
      ontolex:canonicalForm <http://webtentaclel.techfak.uni-bielefeld.de/
      tbx2rdf_iate/data/iate/competence+of+the+Member+States-en#CanonicalForm> ;
      ontolex:sense <http://webtentaclel.techfak.uni-bielefeld.de/
      tbx2rdf_iate/data/iate/competence+of+the+Member+States-en#Sense> .

<http://webtentaclel.techfak.uni-bielefeld.de/tbx2rdf_iate/
data/iate/dal%C4%ABbvalstu+kompetence-lv#Sense>
      a ontolex:LexicalSense ;
      ontolex:isLexicalizedSenseOf :IATE-84 .

<http://webtentaclel.techfak.uni-bielefeld.de/tbx2rdf_iate/
data/iate/competence+of+the+Member+States-en#CanonicalForm>
      ontolex:writtenRep "competence of the Member States"@en .

<http://webtentaclel.techfak.uni-bielefeld.de/tbx2rdf_iate/
data/iate/medlemsstatskompetence-da#CanonicalForm>
      ontolex:writtenRep "medlemsstatskompetence"@da .
.......
```

**Figure 15: TBX (top) to RDF (bottom) conversion in Term-a-LLOD**

TermBase eXchange[36] (i.e. TBX) is a popular international standard for representing and exchanging information about terminology as Linked Data. The guidelines have been released describing how to publish terminologies in TBX format as Linked Data using the OntoLex-Lemon model (McCrae et al., 2011; McCrae et al., 2015). The converter element (A) of Terme-à-LLOD automatically converts the terminology of TBX format into RDF by the TBX2RDF service that maps TBX inputs, including TBX public dialects, i.e., TBX-Core, TBX-Min and TBX-Basic, into RDF format, reusing a set of classes and properties from existing Linked Open Data vocabularies (e.g., OntoLex-Lemon). An example of a conversion from TBX into RDF is shown in Figure 15.

In order to provide a proof-of-concept of this approach to simplify the process of transforming terminological resources into RDF and hosting the RDF as Linked Data, TAL used data from two sources represented in TBX format:

- IATE[37], a central terminology database for all the institutions, agencies and other bodies of the European Union, is considered to be the largest multilingual terminology database in the world. The terminology is provided in TBX format and made available without copyright protection.
- The second sample of data has been extracted from the termbases developed by the Centrum Voor Terminologie (CvT) in Gent - GENTERM.[38] The center, active within the Department of Translation, Interpreting and Communication of Ghent University, coordinates the Department's activities on terminology and terminography and makes available a small set of bilingual termbases, which are the result of several students' projects. GENTERM termbases belong to different domains (e.g., pharmaceutica, waste management, solar energy, diseases, printmaking).

| TBX input | Runtime | # Terms | # Triples | # Lang |
|---|---|---|---|---|
| IATE | 25.2m | 5851035 | 52603182 | 25 |
| Pharmaceutical* | 5.2s | 4629 | 71347 | 2 |
| Diseases* | 3.0s | 799 | 12650 | 2 |
| Waste management* | 2.5s | 396 | 6109 | 2 |
| Solar energy* | 2.9s | 205 | 3758 | 2 |
| Printmaking* | 2.5s | 223 | 3426 | 2 |

**Table 2: Information about IATE and GENTERM conversion process (Entries marked with * are terminologies of GENTERM).**

The IATE and GENTERM terminologies are converted using the Terme-à-LLOD converter (A) and exposed instances on a central demonstration server[39] that can be used in combination with other workflows. As can be seen from Table 2, for each converted termbase, it presents the runtime needed, the number of terms stored in the termbase, the number of triples resulting in the output files, and the number of languages converted.

[36] https://www.w3.org/2015/09/bpmlod-reports/multilingual-terminologies/
[37] https://iate.europa.eu
[38] https://cvt.ugent.be/
[39] http://scdemo.techfak.uni-bielefeld.de/termeallod/

### 2.5.3 Fintan Integration

Since TBX2RDF is a native Java program, it is fairly easy to adapt to Fintan by adding an additional main class implementing the `Loader` interface and adjusting the input parameters for stream processing, which in itself is already supported as an optional mode in TBX2RDF for larger files.

In order to also be able to use other Fintan modules within Terme-à-LLOD, we are currently evaluating a best practice for a native integration of the full Fintan framework into Java applications. Instead of relying on a Docker-based service, we are also considering to directly import Fintan as a native Java library through Maven. Using this approach, in the future, Terme-à-LLOD will be able to expand its scope of support input formats beyond TBX. In addition, Fintan's graph transformation module could be used to adjust or extend other RDF-based terminologies in order to add support for TAL's ontology browsing and linking features described in Section 3.7.

## 2.6 Multilingual Wordnets in CSV and TSV Formats

In Deliverable D3.1 we reported on the transformation onto OntoLex-Lemon of lexical-semantic data included in the Open Multilingual Wordnet (OMW)[40], work which is also reported in (Racioppa and Declerck, 2019). While this work addressed the TSV encoding of Wordnet data for French, Italian and Spanish, we extended the approach to Latin with another Wordnet resource available in a CSV format. This resource is made available by the Latin WordNet initiative at the University of Exeter[41], and is developed in the context of a cooperation with the LiLa project[42] and the University of Genoa.

The data of Latin WordNet is organized in different categories, from which we considered the lemmas, synsets and literal senses. Figure 16 shows the lexical and morphological information associated with the lemmas of Latin WordNet. In Figure 17 we display the information associated with the synsets (sets of cognitive synonyms), whose glosses come from the Princeton WordNet (Fellbaum, ed., 1998). Figure 18 depicts how the synsets are related to the lemmas by the use of their respective ids.

```
id,uri,lemma,pos,morpho,principal_parts,irregular_forms,alternative_forms,pronunciation,prosody,validated
19117,a0031,abdicatio,n,n-s---fn3-,abdication,,,[ab.dɪˈka:.ti.o:],abdicatio,1
46056,a1807,amygdaleus,a,aps---mn1-,amygdale amygdale amygdale,,,[a.'myg.ɗałews],amygdaleus,0
19118,a0035,abdico,v,v1spia--3-,abdic abdix abdict,,,/abˈdi:.ko:/,abdico,1
19119,a0038,abdo,v,v1spia--3-,abd abdid abdit,,,/ˈab.do:/,abdo,1
19141,a0116,aboleo,v,v1spia--2-,abol aboleu abolit,,,[aˈbɔ.ɬe.o:],aboleo,1
19155,19155,abrupte,r,rp--------,abrupt,,,['a.brʊ.pʈɛ],abrupte,1
19124,a0053,abicio,v,v1spia--3i,abic abiec abiect,,,[aˈbɪ.ki.o:],abicio,1
19161,19161,abscise,r,rp--------,abscis,,,['aps.kɪ.sɛ],abscise,1
```

**Figure 16: Lemmas as encoded in the Latin WordNet CSV file format**

[40] http://compling.hss.ntu.edu.sg/omw/

[41] See https://latinwordnet.exeter.ac.uk/ and (Fedriani et al., 2020)

[42] https://lila-erc.eu/

```
id,offset,pos,language,gloss,semfield
132985,04841846,n,10,"a numeral or string of numerals that is used for identification:
                        ""she refused to give them her Social Security number""",
103300,00010123,n,10,an object occurring naturally; not made by man,
103304,00012704,n,10,"one of a class of artifacts: ""an article of clothing""",
103309,00014045,n,10,"the psychological feature of experiencing affective and emotional states:
                        ""he had a feeling of euphoria""",1502
103311,00015185,n,10,"the spatial arrangement of something as distinct from its substance:
                        ""geometry is the mathematical science of shape""",
103325,00020709,n,10,"an action: ""how could you do such a thing?""",
```

**Figure 17: Synsets as encoded in the Latin WordNet CSV file format**

```
id,lemma,synset,period,genre,notes
2,19117,136508,,,
3,19117,136706,,,
4,19117,104057,,,
6,19118,179477,,,
7,19118,179134,,,
8,19118,172209,,,
9,19118,179920,,,
10,19118,175746,,,
11,19119,178747,,,
13,19119,179083,,,
14,19119,178755,,,
15,19119,178753,,,
16,19120,129805,,,
```

**Figure 18: Relations between synsets and lemmas encoded in the Latin WordNet CSV format**

Our work resulted in the generation of 73,949 entries (19,998 adjectives, 38135 nouns, 60 prepositions, 4902 adverbs, 10,854 verbs) and 1,219 morphological rules (192 for nouns, 192 for adjectives and 835 for verbs) in the OntoLex-Lemon representation framework.

The sample below displays, in RDF Turtle serialization (TTL), the OntoLex-Lemon representation for the entry *abactio*" including the "canonical" form (lemma) and additional forms:

```
:lex_abactio a ontolex:LexicalEntry ;
     lexinfo:gender lexinfo:feminine ;
    lexinfo:partOfSpeech lexinfo:noun ;
    ontolex:canonicalForm :form_abactio ;
    ontolex:morphologicalPattern
                    ontolex:la-noun_3 ;
    ontolex:otherForm :form_abactio_root .


:form_abactio a ontolex:Form ;
     lexinfo:case lexinfo:nominative ;
    lexinfo:number lexinfo:singular ;
    ontolex:phoneticRep
        "a.'bak.tˌɪ.jɔ"@la-fonipa ;
```

```
    ontolex:writtenRep "abactio"@la .

:form_abactio_root a ontolex:Form ;
     ontolex:writtenRep "abaction"@la .
```

The corresponding morphological pattern and some associated sub-paradigms and rules are displayed further down:

```
:la-noun_3 a morph:paradigm ;
  rdfs:comment "Latin 3rd noun declension" .

:la-noun_3i a morph:subParadigm ;
     morph:paradigm :la-noun_3 .

:la-noun_3i_abl_m-f_sg a morph:rule ;
     morph:inflectsFor [ lexinfo:case
               lexinfo:ablative ;
          lexinfo:gender lexinfo:feminine,
               lexinfo:masculine ;
          lexinfo:number lexinfo:singular ] ;
     morph:replacement [ morph:source "*$" ;
          morph:target "e" ] ;
     morph:subParadigm :la-noun_3i .


:la-noun_3i_abl_n_pl a morph:rule ;
     morph:inflectsFor [ lexinfo:case
               lexinfo:ablative ;
     lexinfo:gender lexinfo:neutrum ;
     lexinfo:number lexinfo:plural ] ;
     morph:replacement [ morph:source "*$" ;
     morph:target "ia" ] ;
     morph:subParadigm :la-noun_3i .
```

This way, we are making the morphological information included in Latin WordNet available in a declarative way. Section 3.9 briefly describes the relevance of this approach to be able to cross-link WordNet and morphological data.

## 2.7 Transformation of Morphology Datasets

In addition to this transformation, we aim at enriching the OMW data sets already encoded in OntoLex-Lemon with further morphological and semantic information. In doing so, we are in the position of bridging/linking the two types of data sources within the same encoding space offered by OntoLex-Lemon.

As already described in D3.1 and briefly summarized in this deliverable in Section 2.6, data sets from the Open Multilingual Wordnet infrastructure have been transformed onto the

OntoLex-Lemon model (Declerck and Racioppa, 2019). This work has been complemented with the porting of rich morphological resources for the same languages, French, Italian and Spanish. The morphological resources are taken from an updated version of the Mmorph resources (Petitpierre and Russell, 1995). The transformation of such morphological data in OntoLex-Lemon was done also in the context of ongoing discussions towards a new morphology module for the OntoLex-Lemon model and is described by Declerck and Racioppa (2019). Figure 19 shows the quantitative results of the conversion. In total, over 2 Million morphological entries covering 6 languages were converted.

**Source files**

| | | ADJ | ADV | CONJ | DET | INTJ | NOUN | NUM | PART | PREP | PRON | VERB | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Dutch** | *base* | 35.079 | 5.927 | 82 | 9 | - | 94.571 | 282 | - | 64 | 132 | 11.693 | **147.839** |
| | *full* | 89.204 | 5.946 | 84 | 12 | - | 166.361 | 292 | - | 63 | 156 | 109.689 | **371.807** |
| **English** | *base* | 13.410 | 5.744 | 38 | 41 | 27 | 30.386 | 63 | 43 | 93 | 62 | 6.702 | **56.609** |
| | *full* | 14.678 | 6.702 | 39 | 46 | 26 | 60.259 | 62 | 43 | 92 | 67 | 33.268 | **115.282** |
| **French** | *base* | 8.376 | 521 | 39 | 17 | - | 13.575 | 38 | - | 69 | 44 | 4.606 | **27.285** |
| | *full* | 51.469 | 524 | 46 | 53 | - | 29.182 | 37 | - | 83 | 129 | 192.215 | **273.738** |
| **German** | *base* | 14.345 | 1.580 | 74 | 31 | 59 | 68.476 | 44 | 320 | 96 | 64 | 13.479 | **98.568** |
| | *full* | 837.293 | 1.606 | 77 | 334 | 58 | 346.681 | 43 | 321 | 135 | 462 | 332.465 | **1.519.475** |
| **Italian** | *base* | 4.750 | 86 | 53 | 6 | - | 22.026 | - | 15 | 66 | 11 | 3.318 | **30.331** |
| | *full* | 37.996 | 85 | 57 | 11 | - | 41.545 | - | 41 | 116 | 43 | 285.020 | **364.914** |
| **Spanish** | *base* | 8.050 | 298 | 23 | 23 | 50 | 17.939 | 80 | - | 85 | 52 | 2.798 | **29.398** |
| | *full* | 27.166 | 314 | 25 | 73 | 49 | 36.788 | 191 | - | 83 | 119 | 547.232 | **612.040** |

**TTL files**

| | | ADJ | ADV | CONJ | DET | INTJ | NOUN | NUM | PART | PREP | PRON | VERB | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Dutch** | *base* | 35.068 | 5.924 | 82 | 9 | - | 94.434 | 282 | - | 64 | 132 | 11.690 | **147.685** |
| | *full* | 88.818 | 5.927 | 82 | 11 | - | 160.058 | 282 | - | 64 | 161 | 88.620 | **344.023** |
| **English** | *base* | 13.391 | 2.925 | 38 | 41 | 27 | 29.763 | 63 | 43 | 93 | 62 | 6.680 | **53.126** |
| | *full* | 14.655 | 6.700 | 39 | 42 | 27 | 59.091 | 63 | 43 | 93 | 67 | 33.232 | **114.052** |
| **French** | *base* | 3.768 | 519 | 38 | 15 | - | 13.522 | 38 | - | 68 | 44 | 4.559 | **22.571** |
| | *full* | 29.685 | 521 | 47 | 50 | - | 28.550 | 38 | - | 84 | 112 | 214.063 | **273.150** |
| **German** | *base* | 14.119 | 1.505 | 70 | 31 | 59 | 67.362 | 44 | 320 | 94 | 61 | 13.274 | **96.939** |
| | *full* | 464.073 | 1.505 | 70 | 217 | 59 | 217.541 | 44 | 320 | 114 | 309 | 287.023 | **971.275** |
| **Italian** | *base* | 4.750 | 86 | 53 | 6 | - | 22.068 | - | 15 | 66 | 11 | 3.318 | **30.373** |
| | *full* | 37.044 | 86 | 53 | 12 | - | 41.478 | - | 39 | 99 | 31 | 165.899 | **244.741** |
| **Spanish** | *base* | 8.025 | 195 | 23 | 23 | 50 | 18.405 | 79 | - | 83 | 45 | 2.783 | **29.711** |
| | *full* | 27.123 | 300 | 23 | 74 | 50 | 36.645 | 185 | - | 85 | 108 | 522.901 | **587.494** |

**Figure 19: Morphological entries in the Mmorph source files and OntoLex-Lemon (TTL files).** Showing the figures for both the base forms (lemmas) and the full forms (morphological variants).

# 3. Linking [T3.2]

## 3.1 Motivation

### 3.1.1 Levels of Linking

Challenge 4 of Prêt-à-LLOD ("Linking conceptual and lexical data for language services") is addressed by the *Prêt-à-LLOD Linking* technical component. The development of such a component is carried out in the context of task T3.2, whose progress is described in this section. In this task, novel (semi-)automatic methods are studied, aiming to establish monolingual and cross-lingual links across LLOD datasets and models.

To that end, state-of-the art similarity and relatedness measures will be adapted to Linked Data, in particular to exploit the variety and richness of linguistic features found in the LLOD cloud. This task considers emerging techniques based on word embeddings and deep learning, jointly with knowledge-based and distributional semantics-based ones. In this task, research is being done on LLOD-based models, methods, and techniques for accessing and exploiting data across different languages. Further, methods for lexicalising existing ontologies in multiple languages are also studied, by linking them to lexical resources and implementing APIs that provide access to this knowledge. This task will provide the linking technology that underpins many of the pilots and the discovery mechanisms in WP5.

In this task, links at three different levels are being analysed:

*Level A: links between the conceptual and lexical level (lexicalisation).* In this case, links are established between concepts and their lexical realisations, e.g, through the `ontolex:reference` property. See the example in Figure 20, where two ontologies in EN and FR respectively are lexicalised through their corresponding OntoLex-Lemon lexicons.

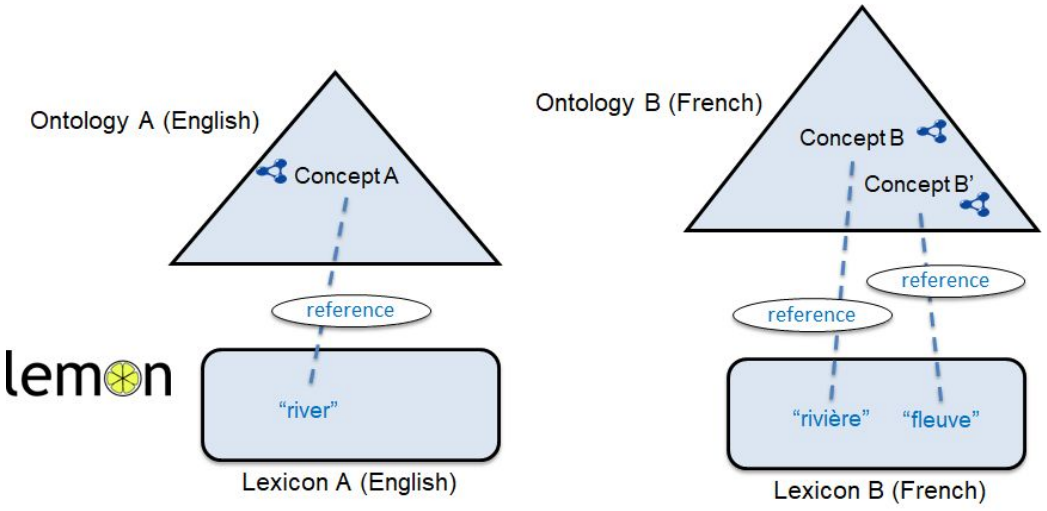


**Figure 20: Linking between the conceptual and the lexical layers**

*Level B: linking at the conceptual level.* Consider for instance the cross-lingual example in Figure 21, where two ontologies in EN and FR, with their corresponding lexical layers modelled as OntoLex-Lemon lexicons, are put in relation. In the example, the concept that describes “river” in ontology A is linked to the concept that describes “rivière” in Ontology B, by a subsumption relation (`rdfs:subclassOf`).

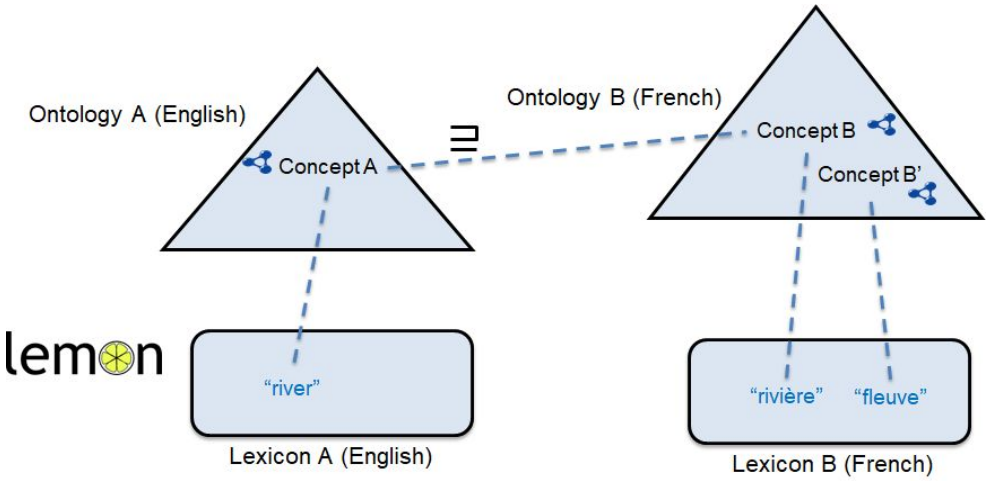


**Figure 21: Linking at the conceptual level**

*Level C: Linking at the lexical level.* In this case, links are established among the information contained in the lexicons, as in the example pictured in Figure 22, where a translation relation is established between lexical senses.

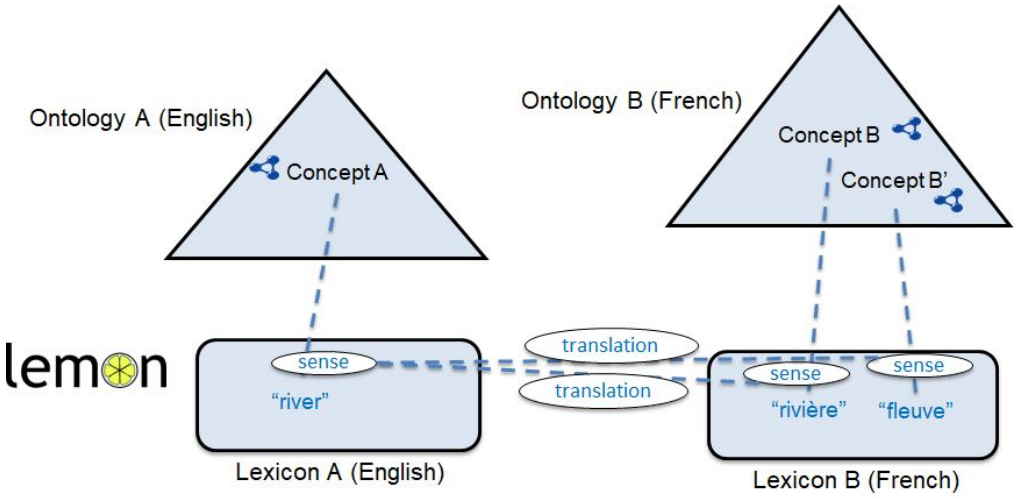


**Figure 22: Linking at the lexical level**

The discovered links can be **monolingual/cross-lingual** and techniques supporting link discovery can be either **automatic** or **semi-automatic**.

In the following subsections we describe the main techniques that we have started exploring in this early stage of the project.

### 3.1.2 Prospective Software Deliverable(s) and Datasets

The software component associated with that task, i.e., the *Prêt-à-LLOD Linking* technical component, will be delivered on M24. Figure 23 gives an overview of the components that will take part in *Prêt-à-LLOD Linking* and how they will interact.

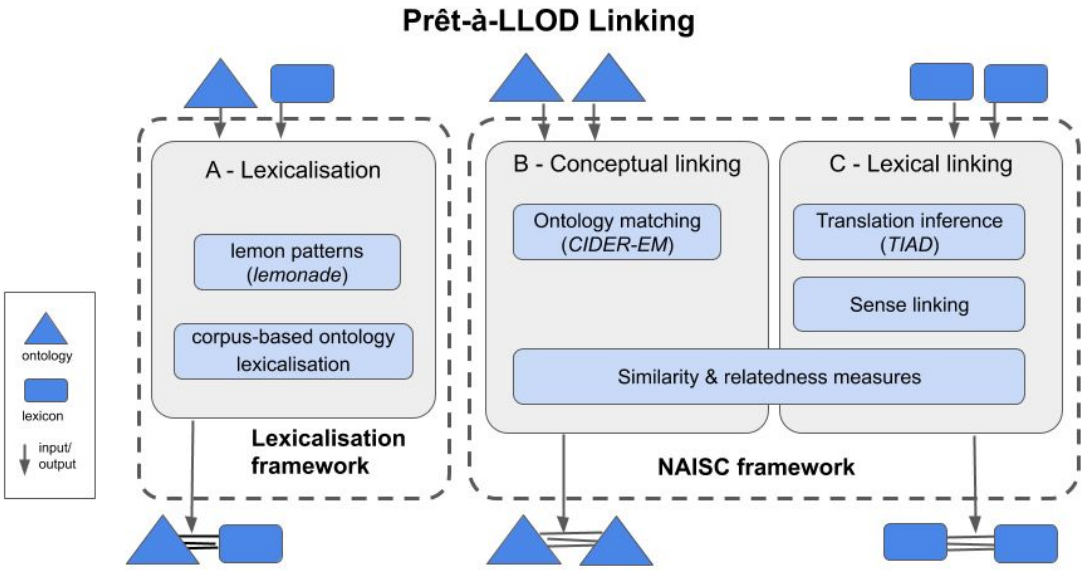


**Figure 23: An overview of the Prêt-à-LLOD Linking component**

We devise at least the following services that the *Prêt-à-LLOD Linking* technical component will support:

*Service*: New version of **Lemonade**
*Responsible*: UPM (Universidad Politécnica de Madrid)
*Input*: Ontology and lexicalisation data (manually provided)
*Output*: *lemon* pattern instances
*Description*: Lemonade (Rico and Unger, 2015) was intended to provide one *lemon* pattern for a given ontology element, but did not support the lexicalisation of the complete ontology. This new version will allow to manage the complete ontology in a collaborative environment (see Section 3.2).

*Service*: Corpus-based **ontology lexicalisation**
*Responsible*: UNIBI (Universität Bielefeld)
*Input*: Linguistic resources (including corpora)
*Output*: OntoLex-Lemon lexica, RDF-graph based patterns/SPARQL queries
*Description*: Algorithm for inducing OntoLex-lexicalisations for a given ontology on the basis of a given corpus

*Service*: Semantic **similarity** between ontology entities
*Responsible*: UNIZAR (Universidad de Zaragoza), NUIG (National University of Ireland Galway)
*Input*: Two ontology entities
*Output*: Semantic similarity value in [0,1]
*Description*: Computation of the degree of similarity between two ontology entities documented in the same or different languages

*Service*: Semantic **relatedness** between ontology entities
*Responsible*: UNIZAR, NUIG
*Input*: Two ontology entities
*Output*: Semantic relatedness value in [0,1]
*Description*: Computation of the degree of relatedness between two ontology entities documented in the same or different languages

*Service*: Generic **ontology matching**
*Responsible*: UNIZAR, NUIG
*Input*: Two monolingual or multilingual ontologies
*Output*: An alignment in the Alignment Format
*Description*: Generic ontology matching service for the discovery of cross-lingual and monolingual semantic equivalences between classes and properties of the two ontologies.

*Service*: **Lexicon matching**
*Responsible*: UNIZAR
*Input*: Two lexicons in the same or different languages
*Output*: A set of ontolex-based correspondences
*Description*: Service for the discovery of links across OntoLex-Lemon lexicons

*Service*: **Translation inference**
*Responsible*: UNIZAR
*Input*: Two dictionaries
*Output*: A set of translations with a confidence score
*Description*: Service for the discovery of indirect translations across two initially disconnected dictionaries that belong to the same RDF graph of dictionary data.

*Service*: **Imprecise/vague translation inference**
*Responsible*: UNIZAR
*Input*: Two dictionaries annotated with degrees of truth
*Output*: A set of translations annotated with degrees of truth
*Description*: Service for the discovery of indirect translations across two initially disconnected dictionaries that belong to the same RDF graph of dictionary data. Input dictionaries must be annotated with degrees of truth denoting imprecision/vagueness, i.e., each translation can be annotated with a degree in [0, 1] estimating to which extent a translation holds.

*Service*: **Uncertain translation inference**
*Responsible*: UNIZAR
*Input*: Two dictionaries annotated with degrees of certainty
*Output*: A set of translations annotated with degrees of certainty
*Description*: Service for the discovery of indirect translations across two initially disconnected dictionaries that belong to the same RDF graph of dictionary data. Input dictionaries must be annotated with degrees of certainty denoting uncertainty, i.e., each translation can be annotated with a degree in [0, 1] measuring our confidence in the correctness of the translation.

*Service*: **Terminology enrichment**
*Responsible*: UPM
*Input*: A monolingual term list with contexts and existing resources from the LLOD cloud
*Output*: An enriched multilingual linked terminology
*Description*: Given a corpus, the tool extracts terms and contexts and retrieves disambiguated multilingual data from the LLOD cloud (translations, definitions, synonyms and other terminological data). The result is an enriched multilingual terminology linked with resources in the LLOD cloud.

## 3.2 Ontology Lexicalisation (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.

**Enhance Lemonade tool**

As introduced in D3.1, the members of the consortium have been working on the problem of ontology lexicalisation previously and have developed Lemonade (Rico and Unger, 2015), a web application aimed at assisting[13] users to create *lexicalisations for an existing ontology*. It is a prototype to show the utility of natural language sentences in the process of creating lexicalisations. Any error in the lexicalisation produces a wrong sentence, which users can easily identify, while correct lexicalisations produce correct sentences. The three most used *lemon* patterns (`Class Noun`, `State Verb` and `Relational Noun`) can be created for any ontology in three languages (English, Spanish and German).

Regarding the ontology lexicalisation research challenge, in Pret-à-LLOD we have been working on extending the functionality of the tool, although the interface is not available yet. Specifically, some progress has been achieved on:

- Supporting multiple users for a more effective interaction, considering the agreement level among them by means of Fleiss' Kappa statistics.
- Software refactoring to provide a general infrastructure that allows an easy integration of *lemon* patterns and Grammatical Framework (GF)[43], a programming

[43] http://www.grammaticalframework.org/

language for multilingual grammar applications, on top of which new web apps can be created. This general infrastructure will be released as an R package.

### Corpus-based Ontology Lexicalisation

Furthermore, we are developing an approach that goes beyond ontology lexicalisation seen as the task of identifying linguistic patterns that verbalise individual ontology elements (i.e., classes, instances, and properties). Instead, our approach strives at identifying linguistic patterns that verbalise graph patterns occurring in a knowledge base and at populating a *lemon* lexicon. This is an ongoing effort. So far, we have implemented the part of the approach that mines a large RDF graph for RDF graph patterns that correspond to an *ngram*. The next step consists in representing these *ngrams* using OntoLex-Lemon.

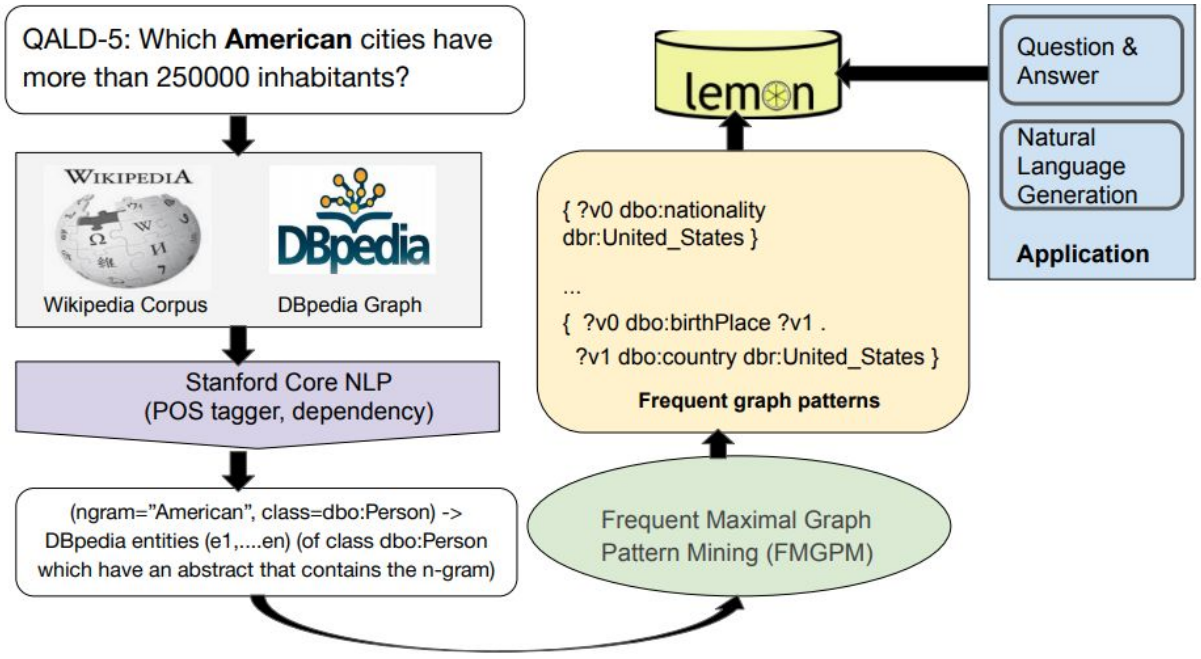


**Figure 24: Corpus based ontology lexicalization**

The approach to extract RDF graph patterns corresponding to an *ngram* makes use of the DBpedia abstract corpus[44] and the DBpedia knowledge graph[45]. The DBpedia abstract corpus consists of entities and the corresponding texts from the Wikipedia articles about these entities (e.g., the text from the Wikipedia page about Germany for the DBpedia entity `dbr:Germany`.) Given that an *ngram* is found in the articles of multiple entities that belong to the same class (e.g., `Actor` or `City`), we perform pattern mining to investigate what these entities have in common, i.e., whether they have a similar neighbourhood in the graph.The approach is shown in Figure 24. The similarity might be that each entity appears in the subject position of an RDF triple with `dbo:nationality` in predicate position and `dbr:Germany` in object position expressing that all entities are of German nationality. The similarity might also be more complex, e.g., that each entity can be bound to the variable *?v0* in the graph pattern `{ ?v0 rdf:type dbo:Settlement . ?v0 dbo:country`

[44] http://downloads.dbpedia.org/2015-04/ext/nlp/abstracts/
[45] https://wiki.dbpedia.org/

`dbr:Greece }`-expressing that each entity is a Greek settlement. As a final example, each entity can be bound to the variable *?v0* in the graph pattern `{ ?v0 dbo:spouse ?v1 . ?v0 dbo:nationality dbr:Greece . ?v0 rdf:type dbo:Person }`-expressing that all entities are married people of Greek nationality. Note that a graph pattern can contain any number of variables.

Similarities in terms of graph patterns among members of a given set of entities are computed via Frequent Maximal Graph Pattern Mining (FMGPM). We define FMGPM from a single large RDF graph, given an RDF graph $g = \{t_1, ..., t_n\}$ and a minimum support parameter $1 \leq \tau \leq n$, as the task of obtaining the set of graph patterns $Q$ where each $q$ in $Q$ is frequent regarding the graph $g$ and the minimum support parameter $\tau$, and each $q$ in $Q$ is *maximal*, i. e. the patterns that are isomorphic to subgraphs of other patterns in $Q$ are not frequent. A graph pattern $p$ is *frequent* with respect to an RDF graph $g$ and a support threshold $\tau$ if $sup(p,g) \geq \tau$. A support measure *sup* is a function that maps a set of mapping functions to an integer value. A mapping for a pattern $p$ and a graph $g$ is a function that maps each variable in $p$ to a term in $g$. This process provides us with a set of frequent graph patterns that are common among the entities that correspond to an *ngram*.

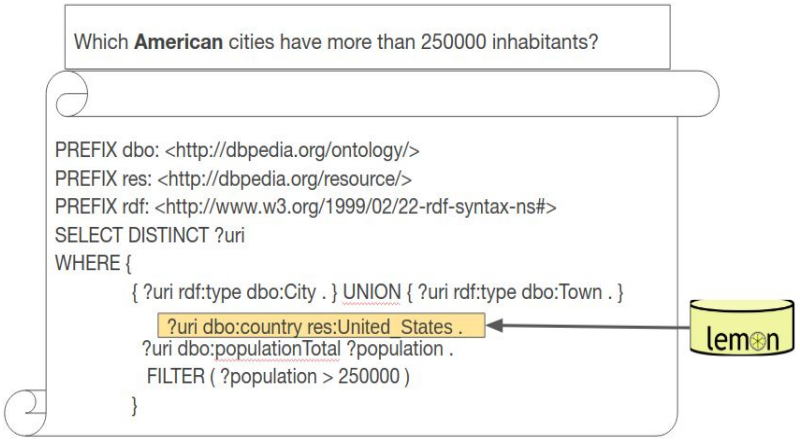


**Figure 25: an example of using lemon in question answering system**

We believe that the problem that we are addressing with this approach is relevant, as natural language can be very concise. When generating natural language text from RDF data, a simple approach would be to verbalise each triple by a sentence, to order all generated sentences and to output this sequence as the resulting verbalisation of the RDF graph. However, this leads to unnatural texts such as “Adam is a man. Adam has nationality Greek. Adam is married to Berta.” Given the *lemon* lexicon populated with the patterns we are considering in our approach, we can generate the sequence “Adam is a married Greek man” given an entity that can be bound to the variable *?v0* in the RDF graph pattern `{ ?v0 dbo:nationality dbr:Greece . ?v0 dbo:spouse ?v1 . ?v0 foaf:gender "male"@en }` - thus a very concise verbalisation that is much shorter than the one obtained by verbalising triples one by one.

Besides the applicability of such a dictionary in the content of generating text from an RDF graph, we are evaluating our approach in the context of question answering over an RDF dataset where the task consists of the following: Given a natural language question and an

RDF graph, answer the question using the RDF graph. Here it is common to analyse the question, identify relevant terms in the graph and to generate formal queries to the graph (i.e., SPARQL queries). In this case, we can look up *ngrams* of the question in our dictionary to obtain graph patterns (Figure 25) that we can then use to assemble SPARQL queries. We are currently carrying out an evaluation with the QALD-5 (question answering over Linked Data) dataset.[46]

## 3.3 Naisc Framework (Level B and C)

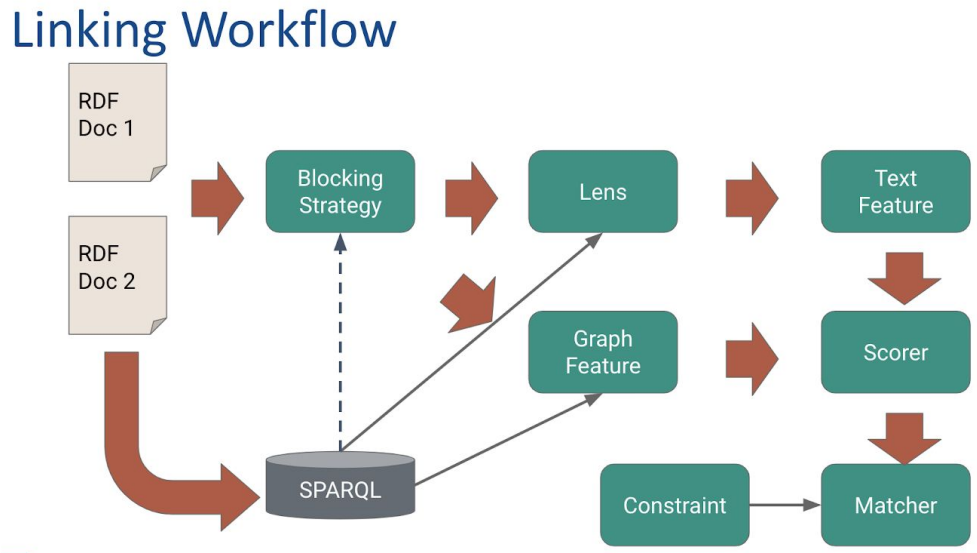


**Figure 26: The architecture that Naisc uses for linking datasets**

Naisc[47] is a toolkit for linking RDF datasets that implements a flexible architecture as depicted in Figure 26. It consists of the following steps:

1. **Blocking Strategies:** These perform a first pass over the two datasets to decide what kind of elements may be linked in the datasets. The simplest strategy, *exhaustive*, links all URIs in both datasets, but more complex strategies may use *approximate string matching* to provide a long list of candidate pairs or take advantage of specific models such as OntoLex-Lemon.
2. **Lenses:** Lenses extract text from elements of the datasets, this could be the *labels,* but also may include the URI or neighbouring elements.
3. **Text Features:** Here the main NLP analysis is done and features are extracted ranging for *basic string* features such as Jaccard similarity to more sophisticated methods using *word embeddings* or *BERT* (Devlin et al. 2019).
4. **Graph Features:** Here any non-NLP features are analyzed from the dataset including any matching *properties* such as part of speech or graph analysis using methods such as *personalized PageRank*.
5. **Scorer:** This is a typical machine learning component that predicts a probability for each possible relation based on the features. This can be implemented by simple

[46] http://qald.aksw.org/index.php?x=challenge&q=5
[47] 'Naisc' means 'links' in Irish and is pronounced 'nashk'.

methods such as *averaging*, but is more typically implemented by *support vector machines* or *logistic regression.*

6. **Matcher:** The goal of the matcher is to find from all candidate links an overall best solution. Here, simple *thresholds* may be applied or more complex **constraints** evaluated using the *greedy algorithm* or *beam search*. We are actively exploring the use of *deep reinforcement learning* to improve the solutions.

Recently in the context of T3.3, we have adopted a REST-based architecture for Naisc, which will be integrated with the Teanga system. To this end, we have defined a set of REST interfaces[48] that can be used by the project to integrate components into Naisc. We expect this to be ready for the 1.0 release of the software in Summer 2020.

As it can be seen in Figure 23 in Section 3.1.2, Naisc will be the framework in which the main linking components (Levels B and C) will be integrated to support the *Prêt-à-LLOD Linking* module. For the moment, such elementary components are under separate development and will be integrated in a later stage of T3.2 task.

## 3.4 CIDER-EM (Level B)

CIDER-EM is a tool for monolingual and cross-lingual ontology matching. Since it operates as an aligner between ontologies, the tool needs two input ontologies given in standard formats (OWL, RDFS…) and as result it produces the alignment between them (in the Alignment Format[49] and other suitable formats).

This system is a new version of CIDER-CL (Gracia et al., 2013), developed by members of the Prêt-à- LLOD consortium in the past. 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 (Ruder et al., 2019). These word vectors are utilised for measuring the similarity between the ontology entities.

The aim of the tool is to operate in two different modes:.

- In the **monolingual** case, when both ontologies are in the same language, the embedding from the ontologies' language is used to compute the word embedding-based value of the relatedness between two entities of the ontologies.
- In the **cross-lingual** case, when the ontologies are in different languages, the embeddings from both ontologies' languages are processed. These word vectors must have the same nature in terms of dimensions and a similar distribution in the space for similar words in order to make comparisons between them. However, each word embedding has been trained with monolingual corpora. Because of that, some transformations (like rotation) in the vectors must be made to allow for the computation of the cosine distance between entities from different languages. These cross-lingual mappings are done in a supervised mode, without using parallel

[48] See https://app.swaggerhub.com/apis/jmccrae/Naisc/1.0#/
[49] http://alignapi.gforge.inria.fr/format.html

corpora, with the tool Vecmap (Artetxe et al., 2018). Then, CIDER-EM computes the word embedding-based value of the relatedness between two entities of the ontologies.

In **both** configurations, the system performs elementary computations of *cosine similarity* to compare several features of the ontological description of the entities under comparison. Such features are combined by means of multilayer perceptrons (one for classes and another one for properties) to produce a final value of relatedness.

## 3.5 Translation Inference (Level C)

### 3.5.1 TIAD Shared Task

We have organised the Translation Inference Across Dictionaries (TIAD 2020) shared task (Kernerman et al., 2020). As in the 2019 edition (Gracia et al., 2019), the aim of this task is to evaluate methods and techniques for automatically generating new bilingual dictionaries from existing ones. Particularly, we focused on a scenario in which a number of pre-existing translations is available among different dictionary data in different languages (e.g., as in the Apertium RDF graph (Gracia et al., 2018)), and we want to find translations between data in two languages for which there are no available translations. For a complete description of the datasets and results, see https://tiad2020.unizar.es/ .

Several Prêt-à-LLOD partners (NUIG, GUF, UNIZAR) developed their own translation inference systems to participate in the shared task:

NUIG (McCrae and Arcan, 2020). The proposed system combines unsupervised NLP and Graph Metrics for Translation Inference. This system includes graph-based metrics calculated using novel algorithms, with an unsupervised document embedding tool called ONETA and an unsupervised multi-way neural machine translation method. The results improve the system that the authors presented in the last TIAD edition and produces the highest precision among all systems in the task while preserving a reasonable recall.

GUF (Chiarcos et al., 2020b) contributed with a method based on symbolic methods and the propagation of concepts over a graph of interconnected dictionaries, which evolves the system presented by the authors in the previous TIAD edition (Donandt and Chiarcos, 2019). Given a mapping from source language words to lexical concepts (e.g., synsets) as a seed, they use bilingual dictionaries to extrapolate a mapping of pivot and target language words to these lexical concepts. Translation inference is then performed by looking up the lexical concept(s) of a source language word and returning the target language word(s) for which these lexical concepts have the respective highest score. They participated with two instantiations of such a system: one using WordNet synsets as concepts, and one using lexical entries (translations) as concepts.

UNIZAR (Lanau-Coronas and Gracia, 2020) contributed two different systems to the shared task. On the one hand Cycles-OTIC, a hybrid technique based on graph exploration that combines a method that explores the density of cycles in the translations graph with the

translations obtained by the One Time Inverse Consultation (OTIC) (Tanaka and Umemura, 1994), which obtained better coverage than OTIC alone but slightly reduced precision. On the other hand, Cross-lingual embeddings, based on the distribution of embeddings across languages (Artetxe et al., 2018), were used to build cross-lingual word embeddings trained with monolingual corpora and mapped afterwards through an intermediate language.

The experiments of all the participants in the TIAD 2020 were assessed against a blind dataset used as golden standard and the results of the evaluation are reported in https://tiad2020.unizar.es/results.html.

### 3.5.2 Materialised Translations in Apertium RDF

In the previous section, we have mentioned some ongoing research efforts in inferring new translations among languages, based on existing ones, and their application to discover new translation links in the Apertium RDF graph. As a step further, we plan to materialise such relations (or at least a part of them with enough quality) as new triples in the RDF graph, to contribute to the constant enrichment and improvement of the LLOD cloud.

As the Apertium RDF Graph has recently been enriched by Prêt-à-LLOD partners with new bilingual dictionaries, resulting in Apertium RDF v.2.0, the idea is to leverage the new connections and the larger number of translations for each pair of language to generate new translations between languages not directly connected, and subsequently enrich the Apertium graph with them and their confidence score. A series of experiments will be performed to test this hypothesis for the inference of new translation pairs between densely connected languages present in both versions of Apertium RDF, given our hypothesis that the method proposed by Villegas et al. (2016) for inferring new translations based on graph exploration and cycle density will obtain better results the more connected the languages are in the given graph.

The confidence score provided by our system encodes information about the certainty of that translation, i.e., how sure the system is that the new translation holds true. It is calculated by means of cycle density. The confidence score of a potential target is assigned by the density value of the most dense cycle where the source and target words occur. This value can achieve values from 0 to 1 (from completely disconnected to fully connected graph). We will explore the use of the new extension to OntoLex-Lemon to represent uncertainty currently being developed at UNIZAR and OUP, Fuzzy *lemon*. The use of the Prov-O ontology[50] (Lebo et al. 2013) will also be considered in order to capture different confident scores obtained from diverse systems, associated to the same inferred translation, as well as the details of such an inference activity.

In addition, Apertium RDF v.2.0 includes languages that were previously absent in Apertium RDF v.1.0, for example, Russian or Icelandic. Some of these languages are only connected to a densely-connected node through a single path. For instance, Russian is connected to

[50] https://www.w3.org/TR/prov-o/

English only through Kazakh, acting as a pivot language. Since applying a circle-density algorithm would not yield satisfactory results due to the lack of cycles, in order to infer new translations through these pivot languages we plan to explore the use of the OTIC algorithm as a future step.

## 3.6 Fuzzy *lemon* (Level C)

Fuzzy *lemon* aims to provide the framework and logic to represent uncertainty and chaining successively linked lexical semantic relations, e.g. senses, as represented in OntoLex-Lemon. In the simplest form it consists of a chain of three senses and aims to answer the following question: Given the (imperfect) links between senses *S1* and *S2* and senses *S2* and *S3*, can we infer/qualify/quantify the sense link between senses *S1* and *S3*?

There are different ways to express the uncertainty (ambiguity, vagueness, etc.) present in linked senses. Currently, OUP is considering three categories defining the linked senses' qualitative relationship: perfect, narrower/wider, and partial. Editors can, following a brief training period, identify this categorical relationship between two senses based on information taken from the respective dictionaries. Agreement among annotators is fairly high (kappa>0.6), suggesting that the categories are sufficiently distinct. UNIZAR on the other hand has a different approach: Each sense link is given a truth degree between 0 and 1, thereby quantifying the extent to which the two senses agree with one another. This method lends itself well to automated chaining of sense links (e.g., exploiting transitivity of the relationships).

We aim at bringing the two approaches together, benefitting from the strengths and addressing the weaknesses of each. Thus far, we have identified the ways that OUP's sense granularity categories interact when chained to form a new sense link. Figure 27 shows the levels of ambiguity that result.

Sense granularity of L1-L3 links based on sense granularity of L1-L2 links and L2-L3 links

| Legend | |
|---|---|
| = | perfect |
| > | wider |
| < | narrower |
| ≈ | partial |
| X | not a link |
| | definite link and granularity |
| | definite link but ambiguous granularity |
| | fully ambiguous |

| | | L1 - L2 | | | |
|---|---|---|---|---|---|
| | | = | > | < | ≈ |
| L2 - L3 | = | = | > | < | ≈ |
| | > | > | > | >, <, =, ≈, X | >, ≈, X |
| | < | < | >, <, =, ≈ | < | <, ≈ |
| | ≈ | ≈ | >, ≈ | <, ≈, X | >, <, =, ≈, X |

**Figure 27: Sense granularity of sense links resulting from sense chains**

Further, we have begun thinking about how the sense granularity, or semantic overlap, is affected by chaining numerically. Rather than simply saying that one sense is partial or wider/narrower than the other, we aim to quantify what percentage of each sense is covered by the other sense. Figure 28 visualizes this idea of semantic overlap:

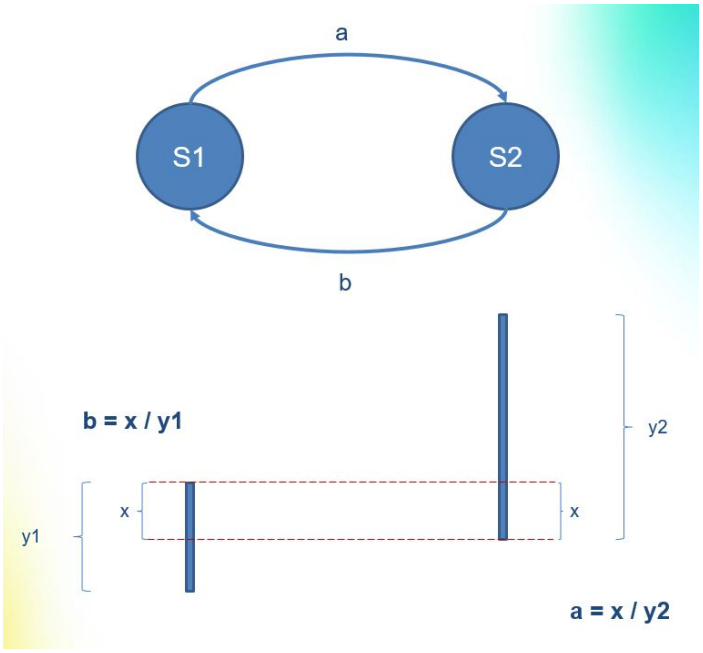


**Figure 28: Semantic overlap visualized; *a* represents the amount *S2* is covered by *S1* and *b* represents the amount *S1* is covered by *S2*.**

With this approach, we can formulate the problem of chaining sense links as follows: Knowing the bidirectional semantic coverage of *S1* and *S2* and of *S2* and *S3*, can we calculate the (expected) semantic coverage of *S1* and *S3*? Looking at Figure 29, the question becomes: Given *a*, *b*, *c*, and *d*, can we calculate the (expected) value of *e* and *f*?

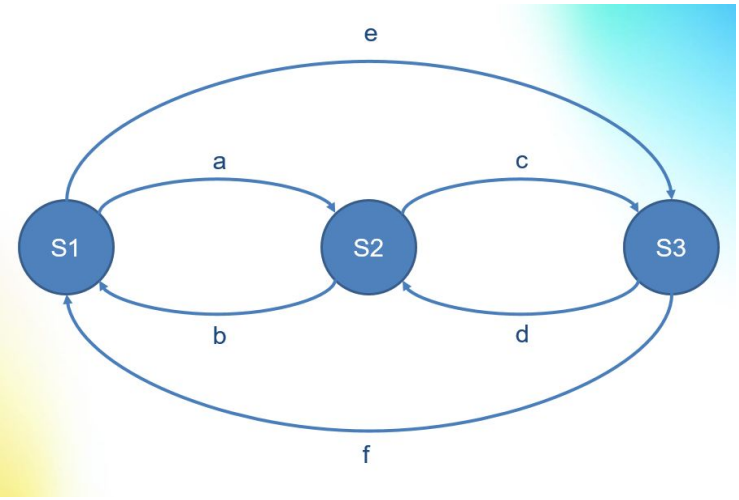


**Figure 29: Sense chaining as calculating semantic overlap**

The values for *e* and *f* can be ranges rather than numbers. We can therefore think of several values of interest: the expected value, the minimum value, the maximum value, or even a probability distribution over all values. Under this assumption that two senses *S1* and *S3* only overlap within the semantic scope of a middle sense (or pivot sense) *S2* we have:

- *e* expected = *a* * *d* * (*c*/*d*) = *a* * *c*
- *e* minimum = max(0, *a*+*d*-1) * (*c*/*d*)
- *e* maximum = min(*a*,*d*) * (*c*/*d*)

## 3.7 Terme-à-LLOD - Navigate, Link and Publish Terminologies (Level B)

In Section 2.5, we present Terme-à-LLOD (TAL) paradigm (Figure 14) for transforming and publishing terminologies as Linked Data which relies on a virtualization approach. We described briefly the approach and the transformation component that automatically converts the terminology of TBX format into RDF (Cimiano et al., 2015) and as a proof of concept how to apply it to the conversion of the well-known IATE terminology, as well as to various smaller terminologies. In this section, we discuss briefly the approach of TAL for linking and publishing terminologies as Linked Data.

**Publishing terminologies**:
The Terme-à-LLOD (TAL) converter (as discussed in Section 2.5) produces RDF output which serves as input to a Virtuoso server. Once the RDF output has been uploaded, the pre-installed server, which hosts the service, exposes the converted data through an endpoint which allows access to them. The server also provides a SPARQL endpoint to other services. The RDF is also exposed to a user-friendly browser where a user can browse terms in alphabet order in 24 different languages and view the details of each term. Figure 30 shows an example of TAL final output, namely the exposure of an RDF terminological resource which can be browsed to access more specific information about each term. The architecture of hosting and publication of terminologies works as follows:

- The TAL approach rests on a preconfigured virtual image of a server that can be downloaded and installed. The virtualization technology is contained into a preconfigured virtual image that can be hosted in a corresponding environment consisting of virtual machines communicating with each other over standard protocols.
- The TAL service automatically adds a Node.js[51] application behind a Nginx reverse proxy[52] for HTTP communication with the service. This application is used to orchestrate the different internals of the container and monitor the status or health of the container.

[51] http://nodejs.org/en/
[52] http://www.nginx.com/

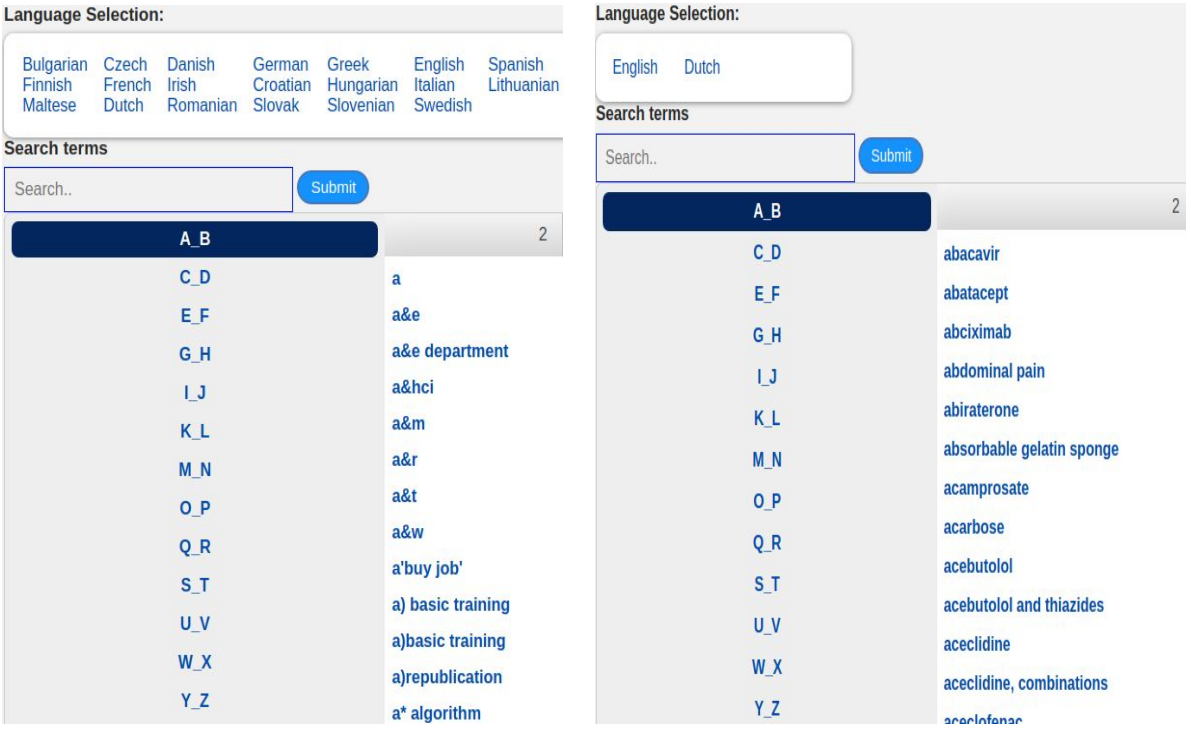


**Figure 30: Example of converted terminological resources exposed in browser**

**Linking terminologies**:
In Section 2.5, we discussed briefly how we converted two terminologies (The IATE and GENTERM) to RDF. In this section, we will explain how the links between them are established.

Terme-à-LLOD established links among the different termbases tested in the use case by means of Simple Knowledge Organization System (SKOS)[53] concepts. The linking across GENTERM and IATE terminologies has been accomplished matching the corresponding lexical entries in different languages by means of a string comparison based on `skos:exactMatch`. This match allows linking, for instance, the term *nefopam*, from the Pharmaceutical termbase in GENTERM, to the corresponding term in IATE, which has an alpha-numeric identifier, i.e., IATE-3545983. Once links among the terminologies have been established, users can explore a term across all the converted and exposed termbases. Table 3 shows the number of links between GENTERM and IATE. Even though the GENTERM terminology covers different domains in two languages (English and Dutch), the termbases available are very small in comparison to IATE. This explains the low number of links for some of the proposed domains, e.g., GENTERM Solar energy-IATE in Table 3.

[53] SKOS is a vocabulary for representing knowledge organization systems (KOS), such as thesauri, classification schemes, subject heading and taxonomies in RDF: https://www.w3.org/TR/skos-reference/

| Termbases | Lang | Number of Links |
|---|---|---|
| GENTERM Pharmaceutical-IATE | English | 1380 |
| | Dutch | 1084 |
| GENTERM Diseases-IATE | English | 22 |
| | Dutch | 27 |
| GENTERM Waste management-IATE | English | 114 |
| | Dutch | 109 |
| GENTERM Solar energy-IATE | English | 12 |
| | Dutch | 20 |
| GENTERM Printmaking-IATE | English | 35 |
| | Dutch | 21 |

**Table 3: Results from the linking process**

## 3.8 TermitUp (Level B + A)

With TermitUp we propose a tool to automatically create terminologies from domain specific corpora and enrich them with external knowledge retrieved from terminologies in the LLOD cloud by establishing links with them. We therefore perform a conceptualization process by retrieving terms that represent a given sense, and interlink, at conceptual level, senses in different resources. As illustrated in Figure 31, we follow a 5-step methodology composed of diverse NLP tasks that transform the input corpus into an RDF terminology linked with resources in the LLOD cloud. Currently, the corpus needs to be in .txt format; however, since the tool is still under development, we are exploring the integration of TermitUp and Fintan so we can process different file formats.

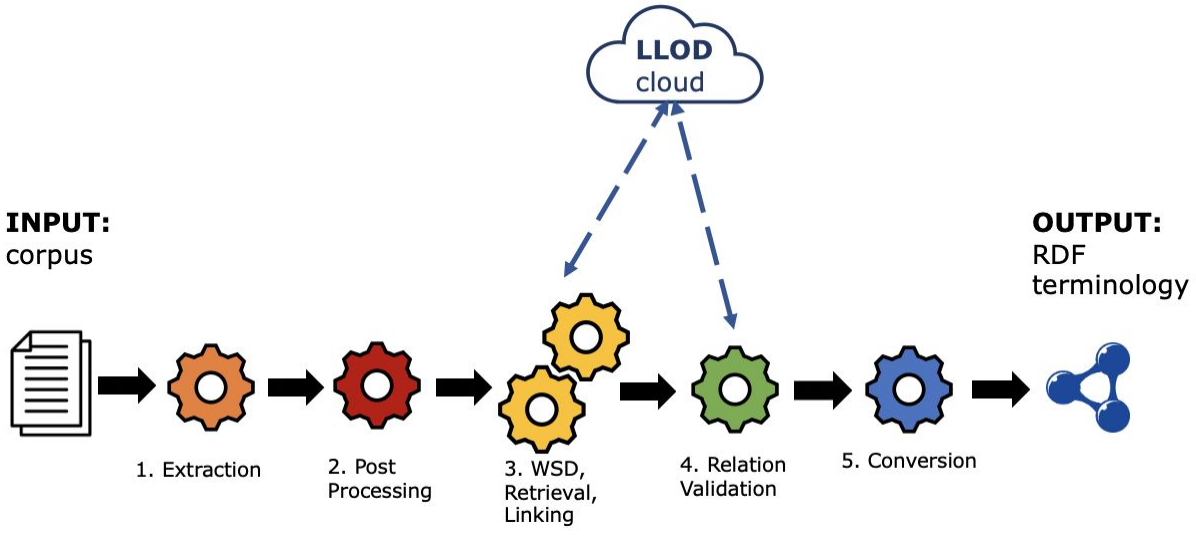


**Figure 31: TermitUp Workflow**

1. **Terminology and Context Extraction**

Terminology extraction is the first step in the process. We are currently implementing an existing open source terminology extraction method named as TBXTools (Oliver and Vàzquez, 2015), which performs term extraction based on statistical methods and returns a list of terms. On top of this extraction process, the tool performs three normalisation processes to refine the initial term list, namely, case normalisation, morphological normalisation and nesting. Once we have a refined term list, we proceed with context

extraction, that is, identifying a window of text in which the term appears in the corpus. This will be needed in a later step for disambiguation purposes.

### 2. Term post processing

In this step our objective is to remove named entities and dates from the term list. The former are removed with simple stopword lists, while for the latter we are considering available tools to detect events (as, for instance, Añotador[54] (Navas-Loro et al., 2020)). Furthermore, we are working on exclusion linguistic patterns to remove non-terminological structures that are, in some occasions, extracted by the tool. This post-processing script is still under development.

### 3. Terms Enrichment: Word Sense Disambiguation, Retrieval and Linking

In this step, three different processes occur simultaneously. Once terms and contexts have been extracted, we start the so-called enrichment phase. In this phase our main objective is to link the extracted terms to resources available in the LLOD cloud. At this moment, we are querying three knowledge bases: EuroVoc[55], IATE[56] and Wikidata[57]. Since these resources contain cross-domain data, we need to go through a disambiguation process to determine the sense in which a given term has been used. For this, we use a Word Sense Disambiguation service provided by Semantic Web Company, as partners of the project.[58] Therefore, given an automatically extracted term, we take N candidate concepts with different senses from the LLOD (in the current implementation, from the resources mentioned above), and compare them with our context, choosing from that list the concept with the closest sense to our term as computed by the WSD service. Through that selected concept we retrieve translations, synonyms, definitions and related terms, enriching thus our initial term list. Consequently, we establish links between our initial concepts, denoted by the terms extracted automatically from the corpus, and those that are retrieved from the queried resources in the LLOD cloud.

Currently we are exploring the integration of this linking algorithm in the Naisc architecture.

### 4. Relation Validation

This task was originally thought to validate synonyms retrieved from Wikidata (Martín-Chozas et al., 2020). We noticed that many of the “alternative labels” contained in Wikidata for a certain term were not actually synonyms but broader, narrower or related terms. Thus, we designed an algorithm that compares the tokens of such terms and determines the type of relation by the use of linguistic rules (see Figure 32).

[54] http://annotador.oeg-upm.net/
[55] https://op.europa.eu/en/web/eu-vocabularies/th-dataset/-/resource/dataset/eurovoc
[56] https://iate.europa.eu/
[57] https://www.wikidata.org/
[58] The implementations of the methods, not the webservice, are available at http://github.com/semantic-web-company/ptlm_wsid.

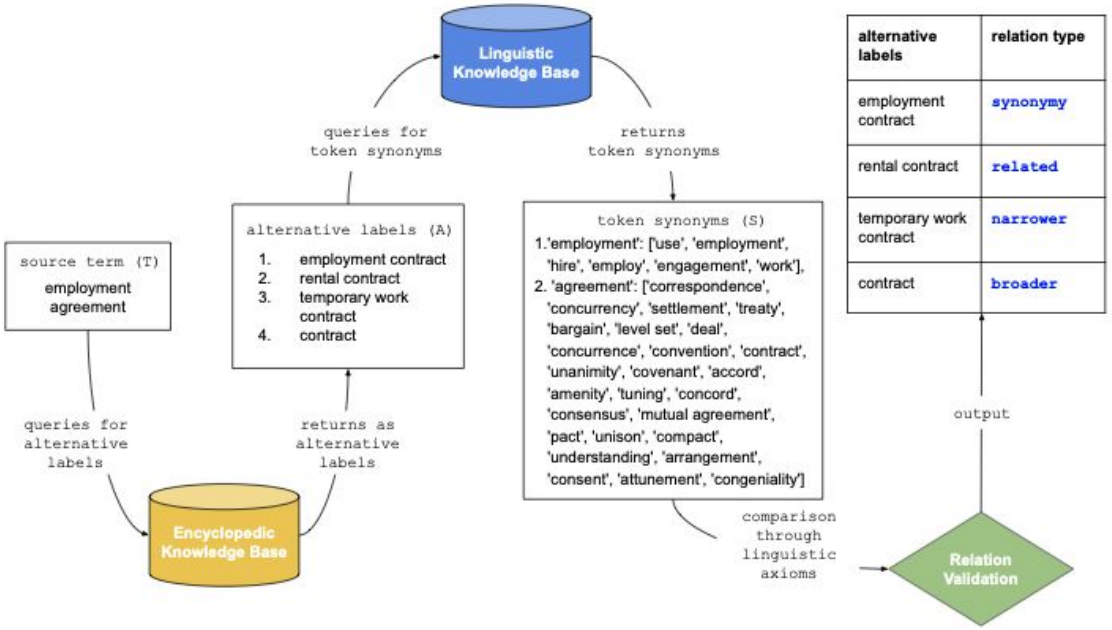


**Figure 32: Example of the Relation Validation task with Wikidata alternative labels**

We are currently trying to apply the relation validation process to the rest of extracted terms within the original term list to discover terminological relations amongst them.

### 5. RDF Conversion

In the first version of TermitUp, we have modeled the terminologies following the SKOS vocabulary, generating one JSON-LD[59] file per concept. We chose this vocabulary due to its simplicity and adequacy to our initial needs. However, after analysing end-user requirements (translators and terminologists), we are planning to move to a more fully-fledged model such as Ontolex-Lemon. Although we do not have the need of modelling different senses that Ontolex-Lemon offers, we do need to capture the provenance or source of every item of information (definitions, scope notes, labels), as well as the relations between preferred and alternative labels, known as *term variants*, which cannot be represented with SKOS.

The output RDF terminologies are planned to be published using Terme-à-LLOD publication components.

## 3.9 Linking WordNet entries to morphological data (Level C)

The transformation work applied to both WordNet data sets and to morphological data sets (both described in Section 2.6 and 2.7) are leading to a bridging or linking of two types of lexical information: lexical semantics on the one hand and morphological variants on the other hand. It is now possible, via the use of the OntoLex-Lemon model, to express that certain Wordnet senses are associated with a specific morphological form, depending on the gender, as described in Declerck and Racioppa (2019) for Romance languages. But this has been shown as well for plural forms in the case of the English entries in the Princeton WordNet, as developed in Gromann and Declerck (2019).

[59] https://www.w3.org/TR/json-ld11/

Recent experiments are pursued on extracting from Wiktionary[60] pronunciation data and to link those to Wordnet entries, allowing thus to mark that the diverse pronunciations of a noun are an indication of its corresponding senses, like for example for the German substantive “Boot” (in IPA notation [bu:t]: *boot*) versus “Boot” ([bo:t]: *boat*). This linking is currently being implemented between the pronunciation data extracted from Wiktionary and an emerging lexical semantics resource for German, which has been already ported to OntoLex-Lemon (Declerck and Siegel, 2019), linking (or merging) thus pronunciation data with lexical semantics information.

[60] https://www.wiktionary.org/

# 4. Workflows for Portable and Scalable Semantic Language Services [T3.3]

## 4.1 Motivation

With a big trend in the software engineering ecosystem moving from monolithic systems to interconnected microservices, the availability of diverse services such as NLP Services and *Semantic Language Services* is growing immensely. As a consequence of this abundance of services, there is a big demand for frameworks that can manage workflows aiming at balancing the ease of integrating new services (portability) with consistency and stability (scalability).

Teanga aims to address the challenge to create "*Workflows for Portable and Scalable Semantic Language Services*". Building on the original implementation by Ziad et al. (2018), a protocol, based on semantic markup, is developed to enable language services to be easily connected into multi-server workflows.

Beyond developing Teanga, in T3.3 we have focused on model-driven approaches to generate domain-specific grammars as a service starting from an ontology lexicon. This provides the basis for containerization of grammar-based services such as question answering (QA) systems and contributes to portability and scalability in the creation and deployment of such services (cf. Section 4.4).

## 4.2 Teanga Architecture, Services Integration and Stakeholders

Teanga aims to address the creation of complex multi-services workflows by minimising the complexity of an user creating this workflow description while also making the integration of services by service partners easy and clear. With this view, Teanga has three main stakeholders:

1. Teanga end users, people who want to use available services to create a workflow description that then can be executed.
2. Teanga service partners, people who want to integrate their services into Teanga and make sure their service will work.
3. Teanga maintainers, people who want to make sure users and partners can accomplish their goals, while guaranteeing infrastructure performance.

### 4.2.1 Overview

The Teanga architecture is essentially divided into Frontend (UI), Backend and Services, where each has its own GitHub repository. The Frontend component's main challenge is

assisting users creating a standard workflow description that then can be processed by the Backend component. Currently both UI and Backend are splitted in two different components that are yet to be connected as a single Docker image. The short term goal was to provide a minimal viable implementation of the Backend[61] such that users can easily setup and execute the Teanga platform locally and execute a workflow if they have a standard workflow description created.

### 4.2.2 Teanga's Current Architecture

**The Teanga UI** uses most common web technologies such as

- **angular.js[62],** a JavaScript-based open-source frontend web framework
- **express.js[63],** an open-source web application framework for Node.js. It is designed for building web applications and APIs.
- **node.js**,an open-source, cross-platform, JavaScript runtime environment that executes JavaScript code outside of a web browser.

The main goal is usability, supporting users to create workflows and search for new services. Currently, the UI allows creating sequential service workflows step-by-step by dragging and dropping existing service bubbles. As shown in Figure 33, in a few steps the user declares the dependencies between services and some of their inputs.

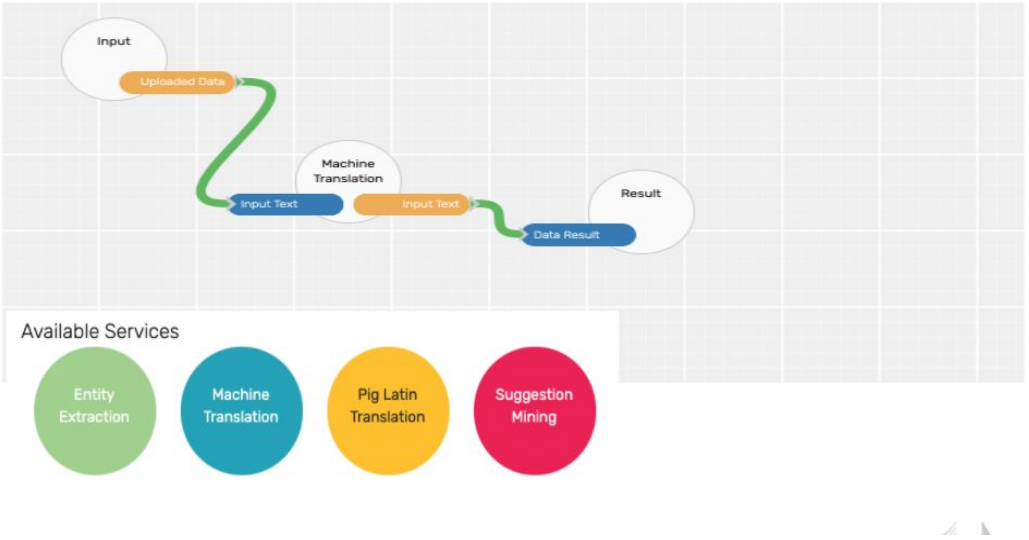


**Figure 33: Teanga UI, interactive view for creating workflows descriptions.**

**The Teanga Backend** was recently implemented and uses Apache Airflow, Docker, Python and OpenAPI-specification as core technologies.

[61] The Teanga Backend is publicly available: https://github.com/Pret-a-LLOD/teanga-executor-service
[62] http://angularjs.org/
[63] http://expressjs.com/

**Airflow**[64] is a platform to programmatically author, schedule and monitor code workflows following the principles of being Dynamic, Extensible, Elegant and Scalable.

**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. Docker is installed on each server and provides simple commands to build, start, or stop containers.

**Python** is a scripting programming language, it is useful for configuration and infrastructure as code and it fits well with Apache Airflow, which is also implemented in Python.

**OpenAPI Specification**, originally known as the Swagger Specification, is a specification for machine-readable interface files for describing, producing, consuming, and visualizing RESTful web services.

**Docker Hub**[65] is a registry service on the cloud that allows 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.

As shown in Figure 34, the Teanga Backend converts a high level workflow description created by the user using Teanga UI (1, 2) to a lower level workflow of operations generated by Airflow (3, 4). Airflow UI allows the user to initialise and keep track of the execution of the workflow operations and running status (5). The user can get the output of the workflow after completion in their local folder (6).

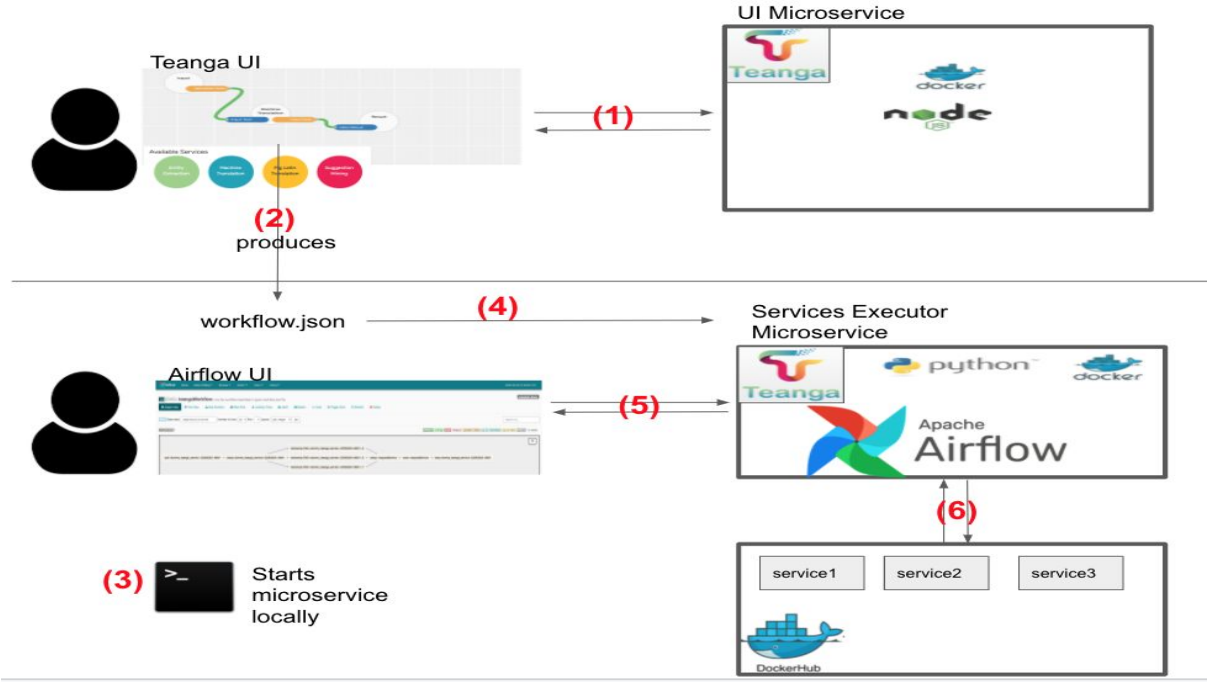


**Figure 34: Teanga flow of execution.**

[64] http://airflow.apache.org/
[65] https://hub.docker.com/

## 4.3 Updates on Teanga's Architecture

Although the flow for generation of a workflow is functioning, there are two main challenges to be tackled that are interrelated, namely:

1. How to define and use service descriptions to infer matchings and avoid conflicts
2. How to guarantee the workflow created will work on the Backend

As initial efforts were devoted mostly to the Teanga UI component, the specific standard definition of the workflow description and the definitions of services were not yet decided. With this in mind, recent updates concentrated in creating a robust Backend architecture that could direct how the UI should define workflow definitions, how service creators should define their services and respective descriptions, and that could also integrate some services as a case study.

### Recent implementations

- Definition of the core structure infrastructure of the Backend choosing Apache Airflow, Docker, OpenAPI-specification and Python. Apache Airflow was selected over Kubernetes[66] and its container-native workflow engine Argo[67] and argoflow mostly due to its flexibility in customizing workflows dynamically. More evaluation should be done in terms of scaling for huge workloads study cases. Both Docker and OpenAPI-specification were selected mostly for seemingly being the most widely adopted technologies in their respective fields. Docker for containerization technologies and OpenAPI-specification for defining RESTful APIs.

- Automation and Containerization of the generation of Airflow workflows (DAGs) based on a workflow description file (.json).

Airflow defines a code-level execution workflow called a DAG, which is composed of different types of operators such as bash operators, Python operators, Docker operators, Kubernetes and others. This task converts a high level workflow description (Figure 35) into code operators (Figure 36) that can then be executed by the Airflow scheduler.

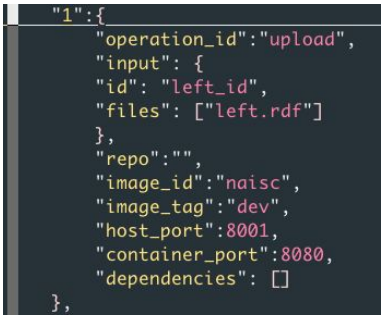


**Figure 35: High level workflow description**

[66] http://kubernetes.io/
[67] http://argoproj.github.io/

- Creation of a Teanga service template image. Examples for the creation of services are available in the GitHub repository.[68]

- Creation of a core Docker image that handles workflow execution and matching between inputs and outputs of the workflow services. Apart from the automation of the creation of the infrastructure with airflow, it is necessary to have a main component that manages the execution flow and requests to the created services. This main component is called *requests manager* and it maps each step in the workflow description to a list of requests.

- Initial implementation of services matching strategy. The requests manager requires matching inputs from one service with outputs from its dependencies. We implemented an initial matching strategy based on the variables *name*, *schema name* and *schema type*.

- First integration with Naisc. We created a workflow description for Naisc and tested its execution.

## 4.3 Case study (Naisc Linking Workflow)

- After all the recent implementations we created a workflow description file for Naisc Linking Workflow as described in Section 3.3. The workflow steps consist of **Blocking Strategies**, **Lenses, Text Features**, **Graph Features, Scorer, Matcher.**
- A demo[69] is available with instructions on how to run the client and an input example available on the *workflows/* folder named `dev_naisc_workflow.json`.

The execution of the workflow pipeline consists of four main steps: setting up the infrastructure, matching dependencies between steps, executing requests to servers and receiving outputs, and finishing infrastructure after final output.
As first operations, Teanga sets three Airflow operations for setting up the Naisc Server: (1) Pulling Naisc Docker image from Docker Hub if it is not available locally, (2) running Naisc image as a Docker container and (3) copying Naisc service's OpenAPI-specification to the request manager container.

After the operations for setting up the infrastructure, Teanga executes the request manager container, going through each step of the workflow and dynamically creating the requests that will be made to Naisc server at that workflow step. If one step of the workflow does not have all the needed inputs for the request, we verify in the dependencies for named variables with same names or for outputs in which the schema are the same.

After all steps on the workflow are concluded, Teanga stops all used server containers and keeps only the Teanga container running. The view of Naisc in Apache Airflow is shown in

[68] https://github.com/Pret-a-LLOD/teanga-executor-service
[69] https://github.com/berstearns/teanga-client

Figure 36. After the execution of this workflow, the user can search in their local teanga/IO folder the output file that describes the inputs and outputs of each step of the workflow.

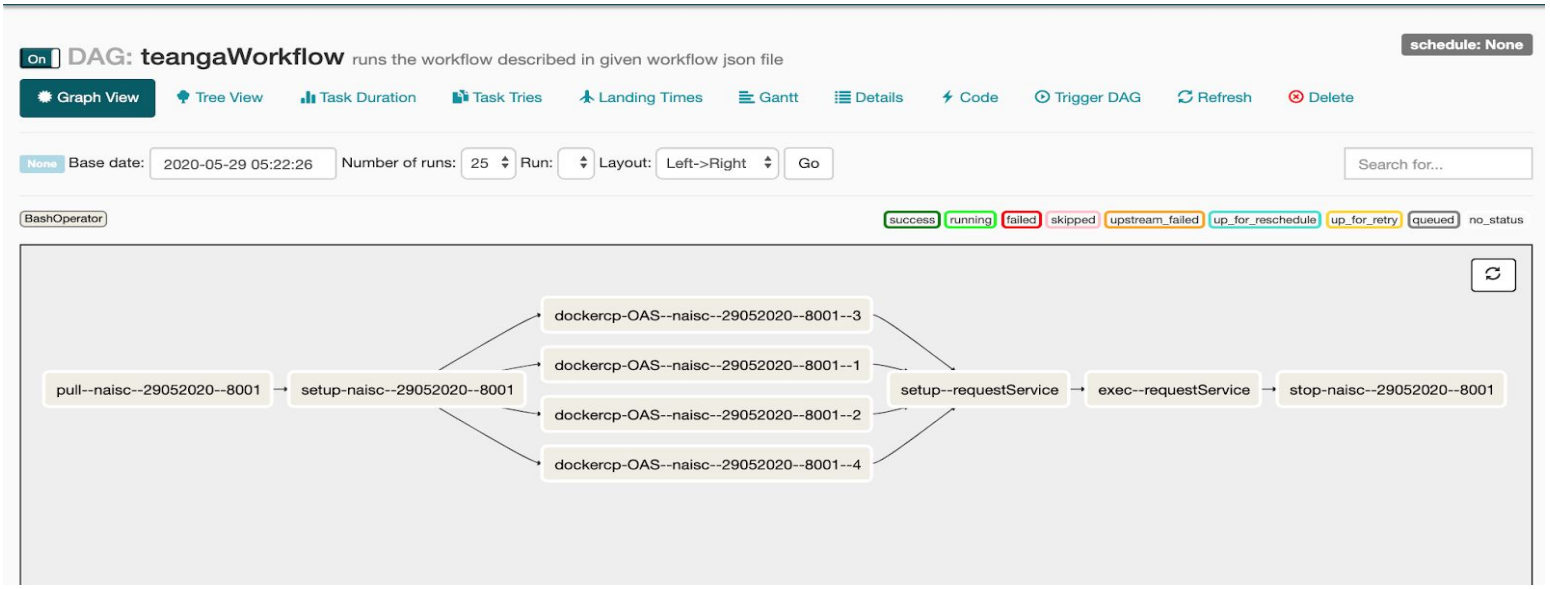


**Figure 36: Teanga-Naisc workflow in Apache Airflow**

## 4.4 Grammars as a Service (GaaS)

In addition to Teanga, we are developing a new paradigm that follows a containerization approach to developing and deploying advanced NLP services including QA services. While originally listed under T3.2 as an Ontology Lexicalisation service (Section 3.2), a prospective integration with Teanga, given its focus on NLP services, in the scope of T3.3 proved more promising. Our vision is to develop containerized QA systems that can be instantiated and configured to a particular domain at a moderate effort only by (declarative) configuration.

Towards this goal, we have so far designed two prototypes that are able to automatically induce a QA grammar from an ontology lexicon or configuration file, respectively. The first prototype (Figure 37) is a system that can induce a QA grammar from a knowledge graph and an OntoLex-Lemon lexicon[70]. We are currently providing the proof-of-concept for this approach using DBpedia and the QALD dataset for evaluation. We are developing a corresponding ontology lexicon for developing and testing the service.
The second prototype (Figure 38) is a system that can automatically generate a grammar for a domain-specific QA system that retrieves text passages from a larger corpus. This prototype has been developed together with the Semantic Web Company in the scope of Prêt-à-LLOD for use across industrial sectors. In collaboration with the Lynx Project[71], it has

[70] https://github.com/vbenz/question-grammar-generator
[71] The Lynx project is developing a knowledge graph and services centered around compliance and the legal domain: http://www.lynx-project.eu/

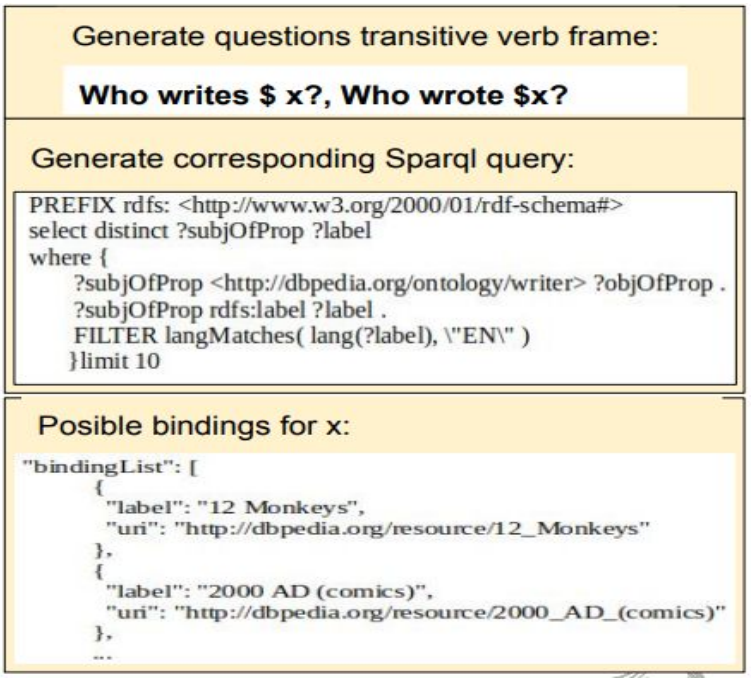


**Figure 37: first prototype: question generation from Lemon**

already been tested on a use case in legal QA[72] (Figure 39). First feedback by lawyers has revealed that the approach is very promising. Currently we are in discussion with the Lynx project to understand whether a larger evaluation can be organized. The code for this prototype is publicly available.[73]

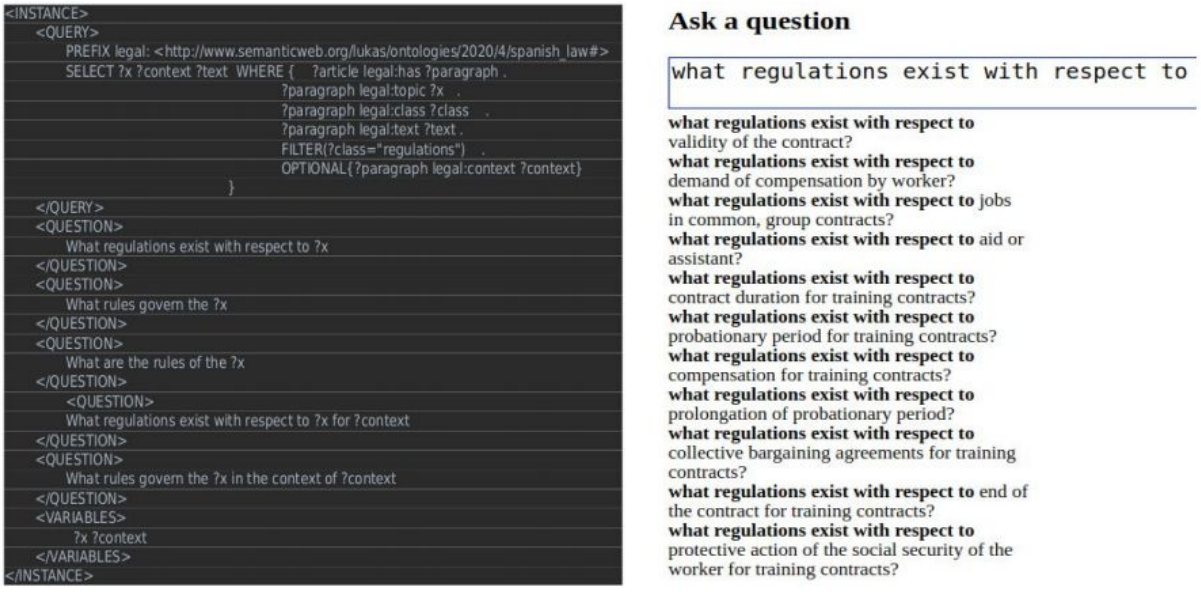


**Figure 38: Prototype 2: generating questions from configuration file**

[72] http://scdemo.techfak.uni-bielefeld.de/TreeLinker/
[73] https://github.com/fazleh2010/TreeLinker

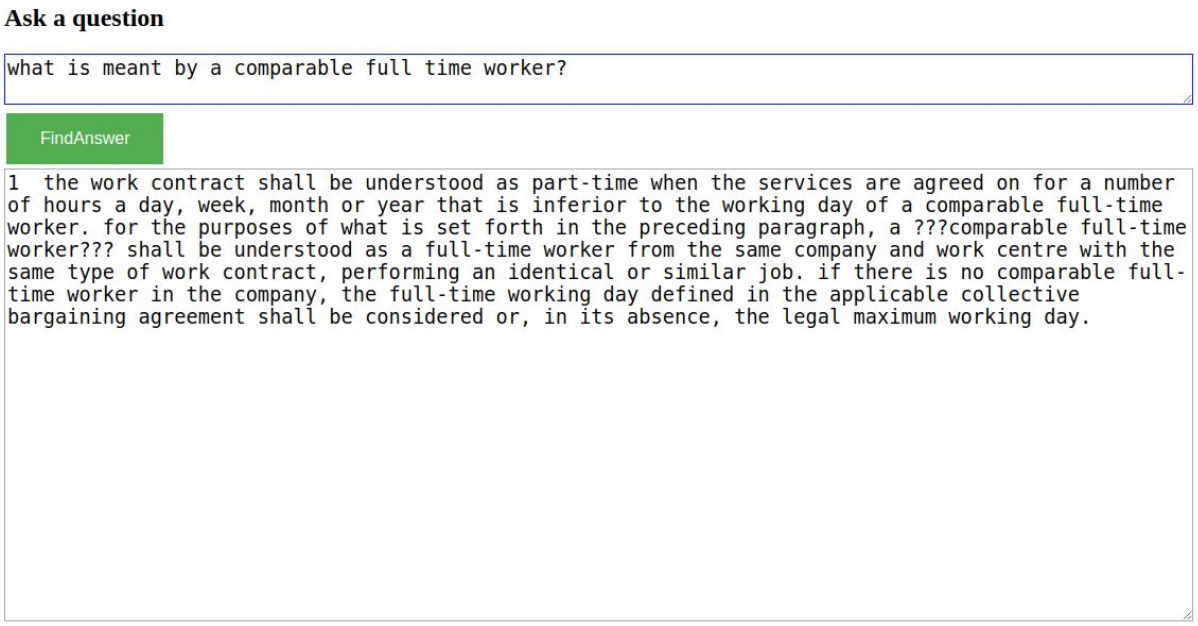


**Figure 39: Prototype 2: The interface of the QA system applied to the legal domain**

# References


Artetxe, M., Labaka, G., and Agirre, E. (2018). A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 789–798.

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. (2007) DBpedia: A Nucleus for a Web of Open Data. In Aberer K. et al. (eds) *The Semantic Web. ISWC 2007, ASWC 2007*. Lecture Notes in Computer Science, vol 4825, pp. 722-735. Springer, Berlin, Heidelberg

Buono, P. M., Cimiano P., Elahi, F. M., and Grimm, F. (2020). Terme-à-LLOD: Simplifying the Conversion and Hosting of Terminological Resources as Linked Data. In *Proceedings of the 7th Workshop on Linked Data in Linguistics, LDL 2020* at *LREC 2020*, pp. 2503‑2510 Marseille, France.

Chiarcos, C. (2012). POWLA: Modeling linguistic corpora in OWL/DL. In *Proceedings of the Extended Semantic Web Conference, ESWC 2012*, pp. 225–239.

Chiarcos, C., Donandt, K., Ionov, M., Rind-Pawlowski, M., Sargsian, H., Wichers-Schreur, J., Fäth, C. (2018). Universal Morphologies for the Caucasus region. In *Proceedings of the 11th International Conference on Language Resources and Evaluation, LREC 2018*, pp. 2631-2640 Miyazaki, Japan.

Chiarcos, C., and Fäth, C. (2017). CoNLL-RDF: Linked Corpora Done in an NLP-Friendly Way. In *Language, Data, and Knowledge, LDK 2017,* pp. 74–88. Galway, Ireland.

Chiarcos, C., Fäth, C., and Ionov, M. (2020a): The ACoLi dictionary graph. In *Proceedings of the 12th International Conference on Language Resources and Evaluation, LREC 2020*, pp. 3281‑3290. Marseille, France.

Chiarcos, C., and Glaser, L. (2020): A Tree Extension for CoNLL-RDF. In *Proceedings of the 12th International Conference on Language Resources and Evaluation, LREC 2020*, pp. 7161–7169. Marseille, France.

Chiarcos, C., Schenk, N., and Fäth, C. (2020b) Translation Inference by Concept Propagation, in *Proc. of Globalex'20 Workshop on Linked Lexicography at LREC 2020*, 2020, pp. 98–105.

Chiarcos, C. and Sukhareva, M. (2015). OLiA–ontologies of linguistic annotation. *Semantic Web*, 6(4), pp. 379–386.

Cimiano, P., Buitelaar, P., McCrae, J., and Sintek, M. (2011). LexInfo: A declarative model for the lexicon-ontology interface. In *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, 9(1), pp. 29–51.

Cimiano, P., Chiarcos, C., McCrae, J.P., Gracia, J. (2020). Linguistic Linked Data - Representation, Generation and Applications. Springer.

Cimiano, P., McCrae, J., Rodriguez-Doncel, V., Gornostay, T., Gomez-Perez, A., Siemoneit, B., and Lagzdins, A. (2015). Linked Terminology: Applying Linked Data Principles to Terminological Resources. In *Proceedings of the eLex 2015 conference,* pp. 504-517. Herstmonceux Castle, United Kingdom.

Cimiano, P., McCrae, J.P., and Buitelaar, P. (2016). Lexicon Model for Ontologies: Community Report. URL https://www.w3.org/2016/05/ontolex/

Erling, O. (2012). Virtuoso, a hybrid rdbms/graph column store. IEEE Data Eng. Bull., 35(1):3–8.

Das, S., Sundara, S., Cyganiak, R. (2012). R2RML: RDB to RDF mapping language. W3C recommendation, World Wide Web Consortium. URL https://www.w3.org/TR/csv2rdf/

Declerck, T., and Racioppa, S. (2019). Porting Multilingual Morphological Resources to OntoLex-Lemon. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2019,* pp. 233--238. Varna, Bulgaria.

Declerck, T. and Siegel, M. (2019). Porting a Crowd-Sourced German Lexical Semantics Resource to Ontolex-Lemon. In: *Proceedings of the eLex 2019 conference, Sintra, Portugal, Lexical Computing CZ s.r.o., CELGA-ILTEC 2019,* pp. 970-984. University of Coimbra, Brno.

Devlin, J., Chang, M. W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Vol. 1, June, Minneapolis, Minnesota, pp. 4171–4186. Association for Computational Linguistics.

Donandt, K., Chiarcos, C.: Translation inference through multi-lingual word embedding similarity. In: Proc. of TIAD-2019 Shared Task Translation Inference Across Dictionaries, at 2nd Language Data and Knowledge (LDK) conference. CEUR-WS (May 2019)

Evert, S. and Hardie, A. (2011). Twenty-first century Cor-pus Workbench: Updating a query architecture for thenew millennium. In *Proceedings of the Corpus Linguistics 2011 conference*, Birmingham, UK.

Fäth, C., Chiarcos, C., Ebbrecht, B., and Ionov, M. (2020): Fintan - Flexible, Integrated Transformation and Annotation eNgineering. In *Proceedings of the 12th International Conference on Language Resources and Evaluation, LREC 2020*, pp. 7212-7221 Marseille, France.

Fedriani, C., Felice, I. D., and Shorth, W. M. (2020). The digital lexicon translaticium latinum: Theoretical and methodological issues. In *Cristina Marras, et al., eds., Atti del IX Convegno Annuale AIUCD. La svolta inevitabile: sfide e prospettive per l'Informatica Umanistica*, pp. 106–113. Associazione per l'Informatica Umanistica e la Cultura Digitale

Fellbaum, C. (ed.) (1998). WordNet: An Electronic Lexical Database. Cambridge, MA: MIT Press.

Forcada, M. L., Ginestí-Rosell, M., Nordfalk, J., O'Regan, J., Ortiz-Rojas, S., Pérez-Ortiz, J. A., Sánchez-Martínez, F., Ramírez-Sánchez, G. and Tyers, F. M. (2011). Apertium: a free/open-source platform for rule-based machine translation. Machine translation, 25(2), 127-144.

Francopoulo, G., Bel, N., George, M., Calzolari, N., Monachini, M., Pet, M., and Soria, C. (2006). Lexical Markup Framework (LMF) for NLP multilingual resources. In *Proceedings of the Workshop on Multilingual Language Resources and Interoperability*, *MLRI '06*, pp. 1-8 Association for Computational Linguistics, USA, 1-8.

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., Kasabi, B., and Kernerman, I. (eds.) (2019): Proceedings of TIAD-2019 Shared Task – Translation Inference Across Dictionaries, co-located with the 2nd Conference on Language, Data and Knowledge, LDK 2019. Leipzig, Germany.

Gracia, J., Villegas, M., Gómez-Pérez, A., and Bel, N. (2018) The Apertium bilingual dictionaries on the web of data. In *Semantic Web*, 9(2), pp. 231–240.

Gromann, D, and Declerck, T. (2019). Towards the Detection and Formal Representation of Semantic Shifts in Inflectional Morphology. In: Maria Eskevich, Gerard de Melo, Christian Fäth, John P. McCrae, Paul Buitelaar, Christian Chiarcos, Bettina Klimek, Milan Dojchinovski (eds.): *2nd Conference on Language, Data and Knowledge (LDK) volume 70, OpenAccess Series in Informatics (OASIcs), Pages 21:1-21:15,* Leipzig, Germany, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 5/2019

Hartung, M., Orlikowski, M. and Veríssimo, S. (2020): Evaluating the Impact of Bilingual Lexical Resources on Cross-lingual Sentiment Projection in the Pharmaceutical Domain. Technical Report. URL http://doi.org/10.5281/zenodo.3707940

Kilgarriff, A., Baisa, V., Busta, J., Jakubıcek, M., Kovár, V.,Michelfeit, J., Rychlý, P., and Suchomel, V. (2014). The Sketch Engine: ten years on. In *Lexicography*, 1(1), pp. 7–36.

Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B.(1993). Building a Large Annotated Corpus of English: The Penn Treebank. In *Comput. Linguist.*, 19(2), pp. 313–330.

McCrae, J. P., Aguado-de Cea, G., Buitelaar, P., Cimiano, P., Declerck, T., Gomez-Perez, A., Garcia, J., Hollink, L., Montiel-Ponsoda, E., Spohr, D., and Wunner, T. (2012). Interchanging Lexical Resources on the Semantic Web. In *Language Resources and Evaluation*, 46(4), pp. 701–719.

McCrae, J. P., and Arcan, M. (2020). NUIG at TIAD: Combining Unsupervised NLP and Graph Metrics for Translation Inference. In *Proc. of Globalex'20 Workshop on Linked Lexicography at LREC 2020*, pp. 92–97.

McCrae, J. P., Cimiano, P., and Doncel, V. R. (2015). Guidelines for linguistic linked data generation: Multilingual terminologies (tbx).
URL https://www.w3.org/2015/09/bpmlod-reports/multilingual-terminologies/.

McCrae, J. P., Spohr, D., and Cimiano, P. (2011). Linking Lexical Resources and Ontologies on the Semantic Web with lemon. In *Extended Semantic Web Conference*, pp. 245–259.

McGuinness, D. L., Van Harmelen, F., et al. (2004). OWL Web Ontology Language Overview. W3C recommendation, World Wide Web Consortium.

Hellmann, S., Lehmann, J., Auer, S., and Brümmer, M. (2013). Integrating NLP using linked data. In *Proceedings of the 12th International Semantic Web Conference, ISWC 2013*. Sydney, Australia, pp. 98–113.

Kernerman, I., Krek, S., McCrae, J.P., Gracia, J., Ahmadi, S. and Kabashi, B. (2020). Introduction to the Globalex 2020 Workshop on Linked Lexicography. In *Proceedings of Globalex 2020 Workshop on Linked Lexicography at LREC 2020*. Marseille, France, pp. 11–16.

Lanau-Coronas, M. and Gracia, J., Graph Exploration and Cross-lingual Word Embeddings for Translation Inference Across Dictionaries, in *Proc. of Globalex'20 Workshop on Linked Lexicography at LREC 2020*, 2020, pp. 106–110.

Lebo, T., Sahoo, S., McGuinness, D., Belhajjame, K., Cheney, J., Corsar, D., Garijo, D., Soiland-Reyes, S., Zednik, S., & Zhao, J. (2013). PROV-O: The PROV Ontology. (W3C Recommendation). World Wide Web Consortium.
URL http://www.w3.org/TR/2013/REC-prov-o-20130430/

Martín-Chozas, P., Ahmadi, S., & Montiel-Ponsoda, E. (2020). Defying Wikidata: Validation of terminological relations in the web of data. In *Proceedings of the 12th International Conference on Language Resources and Evaluation, LREC 2020*, pp. 5654‑5659. Marseille, France.

Navas-Loro, M. , Rodríguez-Doncel, V. (to appear, 2020) Annotador: a Temporal Tagger for Spanish. Journal of Intelligent and Fuzzy Systems

Oliver, A., and Vàzquez, M. (2015). TBXTools: a free, fast and flexible tool for automatic terminology extraction. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2015.* Hissar, Bulgaria, pp. 473-479.

Petitpierre, D., and Russell, G. (1995). MMORPH: The Multext Morphology Program. Multext deliverable 2.3.1, ISSCO, University of Geneva.
URL http://www.issco.unige.ch/downloads/multext/mmorph.doc.ps.tar.gz.

Racioppa, S., Declerck, T. (2019). Enriching Open Multilingual Wordnets with Morphological Features. In: Raffaella Bernardi, Roberto Navigli, Giovanni Semeraro (eds.): *Proceedings of the Sixth Italian Conference on Computational Linguistics,* Bari, Italy, CEUR, 10/2019

Rico, M., and Unger, C. (2015). Lemonade: A web assistant for creating and debugging ontology lexica. In Proceedings of the *International Conference on Applications of Natural Language to Information Systems, NLDB 2015*. Passau, Germany. Lecture Notes in Computer Science, 9103. pp. 448-452. Springer, Cham.

Ruder, S., Vulić, I., and Søgaard, A. (2019) A Survey Of Cross-lingual Word Embedding Models. In *Journal of Artificial Intelligence Research*, vol. 65, pp. 569-631.
URL https://doi.org/10.1613/jair.1.11640

Sylak-Glassman, J., Kirov, C., Yarowsky, D., and Que, R. (2015). A Language-Independent Feature Schema for Inflectional Morphology. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Beijing, China, pp. 674–680.
URL https://doi.org/10.3115/v1/P15-2111

Tanaka, K., and Umemura, K. (1994). Construction of a Bilingual Dictionary Intermediated by a Third Language. In *15th International Conference on Computational Linguistics, COLING 1994*. Kyoto, Japan, pp. 297–303.

Tandy, J., Herman, I., Kellogg, G., et al. (2015). Generating RDF from Tabular Data on the Web. W3C recommendation, World Wide Web Consortium.
URL https://www.w3.org/TR/csv2rdf/

Villegas, M., Melero, M., Bel, N., and Gracia, J. (2016). Leveraging RDF Graphs for Crossing Multiple Bilingual Dictionaries. In *Proceedings of the 10th Language Resources and Evaluation Conference*, *LREC 2016*. Portorož, Slovenia, pp. 868–876.

Westphal, P., Stadler, C., and Pool, J. (2015). Countering Language Attrition with PanLex and the Web of Data. *Semantic Web*, 6(4):347–353.

Ziad, H., McCrae, J., and Buitelaar, P., (2018). Teanga: A Linked Data based platform for Natural Language Processing. In *Proceedings of the 11th International Conference on Language Resources and Evaluation, LREC 2018*. Miyazaki, Japan.