CLSINFRA COMPUTATIONAL LITERARY STUDIES INFRASTRUCTURE

D6.3
Standards beyond TEI
Extended Transformation Matrix
Alternative Formats

Authors: Matej Ďurčo, Vera Maria Charvat, Stefan Resch, Ingo Börner, Lukas Plank
Date: October 31, 2023

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101004984
Project Acronym: CLS INFRA
Project Full Title: Computational Literary Studies Infrastructure

Grant Agreement No.: 101004984

Deliverable/Document Information

Deliverable No.:	D6.3
Deliverable Title:	Standards beyond TEI / Extended Transformation Matrix / Alternative Formats
Authors:	Matej Ďurčo, Vera Maria Charvat, Stefan Resch, Ingo Börner, Lukas Plank
Review:	Ingo Börner, Maciej Eder
Dissemination Level:	PUBLIC

Document History

Version/Date	Changes/Approval	Author/Approved by
2023-08-15	initial draft of the outline	Vera Maria Charvat, Matej Ďurčo
2023-09-29	stable outline, raw text	Vera Maria Charvat, Matej Ďurčo
2023-10-23	version for review	Vera Maria Charvat, Matej Ďurčo, Stefan Resch, Lukas Plank
2023-10-30	reviewed version	Ingo Börner, Maciej Eder
2023-10-31	final version	Vera Maria Charvat, Matej Ďurčo

Contents

List of Abbreviations / Acronyms

Abbreviation/Acronym	Meaning
API	Application Programming Interface
CIDOC CRM	Comité international pour la documentation Conceptual Reference Model
CLARIN	Common Language Resources and Technology Infrastructure
CLS	Computational Literary Studies
CLS INFRA	Computational Literary Studies Infrastructure (project)
CLSCor	Core ontology developed in the CLS INFRA project for CLS corpora
D (e.g., D6.1, D5.1)	Deliverable
DraCor	Drama Corpora Project
FRBRoo	Functional Requirements for Bibliographic Records object oriented
ISO	International Organisation for Standardisation
LOD	Linked Open Data
M (e.g., M24)	Month
ML	Machine Learning
NLP	Natural Language Processing
OCR	Optical Character Recognition
ODD	One Document Does it All
OWL	Web Ontology Language
PoS	Part of Speech
RDF	Resource Description Framework
SKOS	Simple Knowledge Organisation System
TaDiRAH	Taxonomy of Digital Research Activities in the Humanities
TEI	Text Encoding Initiative
URI	Uniform Resource Identifier
WP	Work package
XML	Extensible Markup Language

List of Tables

1. Executive Summary

This deliverable builds on and further extends the findings of D6.1 "Inventory of existing data sources and formats" surveying the landscape of literary corpora, as well as D8.1 "Tools for NLP" cataloguing the set of tools in the context of CLS. Focusing on the wealth of formats used when encoding and processing text, it offers a comprehensive overview of common formats for encoding textual data, beyond the "lingua franca", TEI, both in the domain of computational literary studies and computational linguistics, highlighting potential discrepancies in the approach between these two areas of research. The overview reveals a very heterogeneous landscape with a plethora of formats, devised for differing tasks, from philological encoding of historical text material, to computational annotation and processing of text.

Considering interoperability an indispensable key to reusability, the deliverable explores the challenges and approaches converting between formats.

This information compilation is considered input for further developing the Transformation Matrix, introduced in D6.1, which shall serve as a conceptual framework to consolidate existing solutions for format conversion in the Transformation Toolbox to be delivered by the end of the project (D6.2). The Transformation Matrix shall allow to capture information about specific data structures (features) present in datasets as well as data structures required or produced by tools. This requires a sufficiently expressive formalised description, which is proposed in the CLSCor data model.

2. Introduction

While the TEI[1] is considered lingua franca in the context of the CLS INFRA project, a closer analysis of the different kinds of corpus data (whole corpora, corpus metadata, individual corpus documents) and tools (which expect input formats, perform format conversions and provide output formats) indicates a heterogeneity in the use of formats.

Based on surveys of the community practice and the existing data landscape performed in preceding project tasks, this deliverable explores the broad spectrum of formats used or usable beyond the default format TEI, examines their areas of application and looks at advantages and disadvantages as well as their compatibility with various tools and between each other. The goal is to extend the overview of the landscape with respect to formats.

This work is tightly intertwined with the development of the CLSCor ontology and the corpus metadata catalogue (referred to as "corpusTable") as part of the task T6.1, which need to support a rich representation of the existing formats and their application both on data and tools side.

[1] Strictly speaking, the Text Encoding Initiative as a consortium issues only guidelines. However these are used to encode text, following a certain structure of format. In this deliverable "TEI" refers to this induced format.

Furthermore, this overview represents the extension of the Transformation Matrix which serves as conceptual framework for the Transformation Toolbox to be developed by the end of the project (cf. D6.2).
Deliverable 6.3 is structured as follows: Chapter 3 provides an overview of formats for encoding text besides TEI. Chapter 4 discusses some general challenges in dealing with formats. Chapter 5 deals with issues and approaches when converting data between different formats and chapter 6 introduces the mechanisms in the CLSCor data model to represent format information. Finally, the report offers conclusions and an outlook on future tasks (chapter 7).

3. Formats

The topic of formats used in CLS has been discussed in some of the project's previous deliverables: While D3.1 "Baseline Methodological User Needs Analysis" provided an overview of the formats, methods and tools employed or worked with, within the CLS community based on an analysis of relevant scholarly articles, D6.1 "Inventory of existing data sources and formats" has evaluated the metadata information on formats in which corpora are made available. Moreover, D8.1 "Report of the tools for the basic NLP tasks in the CLS context", has assessed and compiled the input and output formats and compatibility with TEI of relevant tools. The following table provides a count of formats (using terms from the consolidated vocabulary, currently in development for the project) mentioned in the corpusTable. "While TEI (P4 and P5) is the most common format (over 40 corpora are using TEI), corpora/documents are also available in TXT (plain text format; 21 corpora are using the plain text format), XML (12 corpora), PDF (10 corpora), JSON, CSV as well as custom-made formats such as CorA-XML."[2]

Table 1: Count of formats mentioned in corpusTable

corpusFormat (consolidated vocabulary) mentioned in corpusTable	count of corpora in corpusTable mentioning these formats
TEI	41
TXT	21
XML	12
PDF	10
EPUB	8
HTML	7
CSV	3
JPEG	3
MOBI	3
TEI P5	3
ANNIS	2

[2] Table description cited from (D6.1, 2022, p. 26), adding updated numbers

JSON	2
LaTeX	2
ZIP	2
Alto XML	1
CorA XML	1
DOC	1
DOCX	1
ELAN	1
MARC21	1
RDF	1
SGML	1
TEI P4	1
XSL FO	1

An updated version of a comparison table between formats occurring in the corpusTable (= corpus metadata inventory) and D3.1[3] (formats mentioned in CLS literature), provided in D6.1 (D6.1, 2022, table 4, p. 26), now features an overview of tool input and output formats, derived from D8.1.

Table 2: Comparison of formats occurring in corpusTable and D3.1

(updated version, adding format information from D8.1)

corpusTable and D3.1 overlap (using the consolidated CLSCor vocabulary)	CSV, DOCX, EPUB, HTML, JPEG, JSON, PDF, SGML, TEI, TEI P4, TEI P5, TXT, XML, XML RDF
only in corpusTable (using the consolidated CLSCor vocabulary)	Alto XML, CorA XML, DOC, ELAN, JPEG, LaTeX, MARC21, MOBI, XSL FO, ZIP
only in D3.1	3D Max, 3ds, accdb, ADex, aiff, ArcheoML, ASCII, asf, AVI, bak, bmf, CAD, CARARE, CEI, cgm, CIDOC CRM, dae, db, DB, dBase, DBF, DCMI, DDI, dmp, dng, DTD, Dublin Core, dwf, dwg, dxf, EAD, EDM, EpiDoc, eps, ESRI, exif, fbx, flac, fmp12, fp5, FP7, FRBR, geoTIFF, GIF, indd, iptc-naa, JPEG2000, Keyhole Mark-up Language, LIDO, maff, Manuscriptum XML, MapInfo Interchange, matroska, mbwf, mdb, MEI, METS, mhtml, MIDAS, MIDI, mj2, mkv, MODS, mov, MP3, mp4, mpeg, mpeg-2, MPEG-7, MS Access, MuseData, MusicXML, mxf, MXML, NISO, nxs, oai_dc, odb, ods, odt, ogg, pdf/a, ply, PNG, prc, PREMIS, rf64, RTF, siard, siard2, SkP, SQL, sql Dumb *(sic!)*, stl, SVG, sxc, sxw, TeX, TIFF, TSV, u3d, UTF-

[3] The data used in this table originates from the csv files, provided by Fileva, et al., 2022 (results_formats_full.csv)

	8, VRML, W3C Prov, WARC, wav, webCGM, wma, wmv, X3D, XHTML, xls, xlsx, xmp, xsd
tool input formats derived from D8.1, overlaps with corpusTable and D3.1 are marked bold (using the consolidated CLSCor vocabulary)	CoNLL 2000, CoNLL2002, CoNLL 2003, CoNLL 2006, CoNLL 2009, CoNLL 2012, CoNLL CoreNLP, ConLL-U, **DOC**, **DOCX**, **HTML**, IMS CWB VRT, **JSON**, NIF, **ODT**, PERSEUS_2.1, **PDF**, **RTF**, TCF, **TEI**, TEI TXM, **TEI P5**, **TSV**, **TXT**, VERTICAL, **XML**
tool output formats derived from D8.1, overlaps with corpusTable and D3.1 are marked bold (using the consolidated CLSCor vocabulary)	**CSV**, CoNLL-U, GEXF, **JSON**, **RDF**, **TEI**, **XML**

In the remainder of this chapter, selected formats are briefly introduced with pointers to more information. The criteria for inclusion were the following: The formats are used to encode text (as opposed to image, audio, structured data etc.). They have been used within the community as observed in one of the source deliverables, importantly focussing not only on the computational literary studies but also computational linguistics and natural language processing domain. Some formats were also added for reference based on desktop research or prior knowledge of the authors (especially the case for FoLiA, LAF/GraF).
The overview is divided into following groups (roughly based on the underlying data structure):

- Tree – XML-based (e.g., TEI, FoLiA, TCF, ALTO, METS)
- Tabular – Verticals and other formats used in NLP (e.g., CoNLL)
- Graph – RDF-based (e.g., LAF, NIF)

A few general observations:

- The formats differ in terms of intended application domain. These can be seen on a spectrum from critical editions where rich structural and semantic annotation is a sine qua non, to NLP tasks, which require mostly plain text as input format, stripping of any "superfluous" markup being the first operation applied on texts. The ambition within the CLS INFRA project is a unification of these two approaches, aiming to retain the structural markup and enrich it with the NLP output.
- Most of the formats provide some generic extension mechanisms, introducing differing degrees of freedom, both with respect to types of annotation (annotation layers) and with respect to the allowed range of values (tagsets).
- Some formats are developed in the context of specific technologies and are correspondingly tightly tied to these.

3.1 Tree Structure - XML-based Formats

XML as a semi-structured text-based format, allowing to introduce structure into plain text through use of markup, lends itself ideally for encoding additional information about textual data. Accordingly, numerous XML-based formats exist for encoding text and any kind of enrichment on it, most prominently TEI.

3.1.1 TEI – Text Encoding Initiative

TEI[4] is a standard for the representation of texts in digital form. It was introduced in 1994 and is defined through the TEI Guidelines devised by the TEI consortium. In addition to the Guidelines themselves, the Consortium provides a variety of resources and software developed for or adapted to the TEI. TEI is most widely used in disciplinary communities dealing with text as an object of research and is considered lingua franca in the context of the CLS INFRA project.
TEI offers a rich set of XML elements to encode various aspects of a text, be it linguistic, structural or semantic. TEI also introduces a powerful customisation mechanism[5], allowing to reuse only a selected subset of elements and features to define a bespoke schema custom-tailored to specific (encoding) needs of a project.

3.1.2 FoLiA – Format for Linguistic Annotation

FoLiA[6], short for "Format for Linguistic Annotation" is a XML-based open-source[7] annotation format, developed mainly by Maarten van Gompel at Tilburg University (now at Radboud University Nijmegen), with input from Antal van den Bosch, Ko van der Sloot, Martin Reynaert amongst others. As stated on their website, the "intended use is as a format for storing and/or exchanging language resources, including corpora. Our aim is to introduce a single rich format that can accommodate a wide variety of linguistic annotation types through a single generalised paradigm. We do not commit to any label set, language or linguistic theory. This is always left to the developer of the language resource and provides maximum flexibility."[8] The format has established itself primarily in the Dutch and Flemish Natural Language Processing community and is supported by various software projects (e.g., ucto[9], frog[10])
FoLiA comes with an extensive and well documented set of tools, a python library[11], command-line tools[12], as well as FLAT[13], short for "FoLiA Linguistic Annotation Tool", is a web-based multi-user

[4] https://tei-c.org/
[5] https://www.tei-c.org/release/doc/tei-p5-doc/en/html/USE.html#MD
[6] Website: https://proycon.github.io/folia/, Documentation: https://folia.readthedocs.io/, GitHub source repository: https://github.com/proycon/folia, RelaxNG schema: https://github.com/proycon/folia/blob/master/schemas/folia.rng
[7] FoLiA and its accompanying resources are licensed under the GNU Public License v3
[8] https://proycon.github.io/folia/
[9] https://languagemachines.github.io/ucto/
[10] https://languagemachines.github.io/frog/
[11] FoLiA Python library: https://pypi.org/project/FoLiA/ and documentation: https://foliapy.readthedocs.io/
[12] https://github.com/proycon/foliatools
[13] GitHub repository: https://github.com/proycon/flat and documentation: https://flat.readthedocs.io/

environment for working with FoLiA documents. Additionally, there is "FoliAutils"[14] a series of programs for format conversions (e.g., converting documents from PAGE, plain text, ABBYY finereader, hocr, ALTO DIDL to FoLiA and FoLiA to plain text)

3.1.3 GrAF – Graph Annotation Format

GrAF, short for "Graph Annotation Format", was presented in 2007 by Nancy Ide and Keith Suderman (Ide and Suderman, 2007) as "the final XML serialisation of the LAF[15] interchange format" with subsequent slight modifications" (Ide and Suderman, 2014), enshrined together with LAF as ISO standard ISO 24612:2012.
The main focus of GrAF lies "on the representation of texts and associated linguistic annotations. Texts are annotated using stand-off markup, and the annotations themselves can be part of a tree, for example to annotate dependencies. Annotations consist of a label and a so-called feature structure, which contains the individual elements - features - of an annotation." (Hinkelmanns, 2021; translated into English) "GrAF is intended to serve as a pivot format into and out of which representations of annotations in other formats can be mapped to facilitate interoperability, and not as a stand-alone format on its own." (Ide, et al. 2017)

3.1.4 TCF – Text Corpus Format

TCF[16] (currently v.0.4), short for "Text Corpus Format" is an XML-based exchange format, which has been developed in the context of the WebLicht[17] (Web-Based Linguistic Chaining Tool) to ensure efficient interoperability between the tools. It is "fully compatible with the Linguistic Annotation Format (LAF) and Graph-based Format for Linguistic Annotations (GrAF)" and "supports incremental enrichment of linguistic annotations at various levels of analysis in a stand--off XML-based format. Each tool may add one or more annotation layers to the data document, but tools are not permitted to remove or alter any existing layers within the document. "[18]

3.1.5 ALTO – Analysed Layout and Text Object

ALTO[19], short for "Analysed Layout and Text Object" is an open XML-Schema, which was initially developed to describe and store technical metadata on textual (OCR) content and layout information of digitised materials. It was designed to be used in combination with METS[20] but can exist independently as a standalone document. "METS provides metadata and structural information while

[14] https://github.com/LanguageMachines/foliautils
[15] see chapter "3.3.3 LAF - Linguistic Annotation Framework" of this report for more information
[16] https://weblicht.sfs.uni-tuebingen.de/weblichtwiki/index.php/The_TCF_Format
[17] https://weblicht.sfs.uni-tuebingen.de/weblichtwiki/index.php/Main_Page
[18] https://weblicht.sfs.uni-tuebingen.de/weblichtwiki/index.php/The_TCF_Format#Background_on_WebLicht_and_Motivation_for_TCF
[19] Official Website for ALTO: https://loc.gov/standards/alto/, GitHub page: https://altoxml.github.io/ and repository https://github.com/altoxml
[20] https://www.loc.gov/standards/mets

ALTO contains content and physical information."[21] Schema version 4.4 was released on 2023-04-07[22] and can be accessed under https://www.loc.gov/standards/alto/v4/alto-4-4.xsd.

3.1.6 METS – Metadata Encoding and Transmission Standard

The Metadata Encoding and Transmission Standard (METS) was created in 2001 and is "a standard test for encoding descriptive, administrative, and structural metadata regarding objects within a digital library [ed. or other kind of repository], expressed using the XML Schema of the World Wide Web Consortium[23]. The standard is maintained by the METS Board in collaboration with the Network Development and MARC Standards Office[24] of the Library of Congress, and started as an initiative of the Digital Library Federation."[25] The current METS Schema version is 1.12.1[26] Thus METS is not intended to encode textual content. However, it is oftentimes used in tandem with TEI, or ALTO formats to encode the inner structure of a document collection.
"The open flexibility of METS means that there is not a prescribed vocabulary which allows many different types of institutions, with many different document types, to utilise METS. The customization of METS makes it highly functional internally, but creates limitations for interoperability. Interoperability becomes difficult when the exporting and importing institutions have used vocabularies."[27]

3.2 Tabular, Vertical and other Formats for NLP

A typical starting point for NLP tasks is plain text without any markup. However, a traditional format in NLP is the so-called vertical format, where each token is on a separate line with annotation layers being represented as columns, basically forming a tabular structure which can be represented very efficiently, typically as CSV / TSV (comma / tab-separated values) formats, and accordingly also easily processed by any tooling. At the same time the structure is very flexible, allowing addition of any custom annotation layers by simply introducing new columns. This generic extensible structure however also carries no predefined semantics, and each variant must be accompanied by a definition of the annotation layers and their respective value range.

3.2.1 CoNLL

A prominent example of a vertical format is the CoNLL-family of formats devised by the shared task annually put forward by the CoNLL[28] (Conference on Natural Language Learning). Special aspect of the CoNLL formats is that they represent a syntactic parse tree, i.e. there are extra columns describing the syntactic structure of words within the sentence (`id`, `head`, `deprel`). These variants of CoNLL differ in

[21] https://www.loc.gov/standards/alto/description.html
[22] https://www.loc.gov/standards/alto/news.html#4-4-released
[23] https://www.w3.org/
[24] https://www.loc.gov/marc/ndmso.html
[25] https://www.loc.gov/standards/mets/
[26] https://www.loc.gov/standards/mets/version1121/mets.xsd
[27] https://en.wikipedia.org/wiki/Metadata_Encoding_and_Transmission_Standard
[28] https://www.signll.org/conll/

the additional layers introduced as dictated by the specific task for which they were designed, but they share the general tabular or vertical structure. CoNLL-X was devised for the task of multilingual Dependency Parsing and was the result of converting and unifying treebanks for 13 languages (Buchholz and Marsi, 2006). CoNLL-2003 was the format of the dataset for named entity recognition task[29].

A popular variant is CoNLL-U[30] which combines the vertical format with the Universal Dependencies (UD) framework[31] (aiming for consistent linguistic annotation across different human languages). It defines following fields:

1. ID: Word index, integer starting at 1 for each new sentence; may be a range for multi-word tokens; may be a decimal number for empty nodes (decimal numbers can be lower than 1 but must be greater than 0).
2. FORM: Word form or punctuation symbol.
3. LEMMA: Lemma or stem of word form.
4. UPOS: Universal part-of-speech tag.
5. XPOS: Optional language-specific (or treebank-specific) part-of-speech / morphological tag; underscore if not available.
6. FEATS: List of morphological features from the universal feature inventory or from a defined language-specific extension; underscore if not available.
7. HEAD: Head of the current word, which is either a value of ID or zero (0).
8. DEPREL: Universal dependency relation to the HEAD or a defined language-specific subtype of one.
9. DEPS: Enhanced dependency graph in the form of a list of head-deprel pairs.
10. MISC: Any other annotation.

Example of the CoNLL-U format:

```
1    They     they     PRON    PRP    Case=Nom|Number=Plur               2    nsubj 2:nsubj|4:nsubj
2    buy      buy      VERB    VBP    Number=Plur|Person=3|Tense=Pres    0    root     0:root
3    and      and      CCONJ   CC     _                                  4    cc       4:cc
4    sell     sell     VERB    VBP    Number=Plur|Person=3|Tense=Pres    2    conj     0:root|2:conj
5    books    book     NOUN    NNS    Number=Plur                        2    obj      2:obj|4:obj
6    .        .        PUNCT   .      _                                  2    punct    2:punct
```

3.2.2 UD – Universal Dependencies

"Universal Dependencies (UD) is a framework for consistent annotation of grammar (parts of speech, morphological features, and syntactic dependencies) across different human languages. The annotation scheme is based on an evolution of (universal) Stanford dependencies (de Marneffe et al., 2006, 2008, 2014), Google universal part-of-speech tags (Petrov et al., 2012), and the Interset interlingua for morphosyntactic tagsets (Zeman, 2008)."[32]

[29] https://paperswithcode.com/dataset/conll-2003
[30] https://universaldependencies.org/format.html
[31] https://universaldependencies.org/
[32] https://universaldependencies.org/introduction.html

UD consists of a set of encoding and annotation guidelines, together with defined sets of tags, features and relations.[33] The data is encoded in the so-called ConLL-U format based on ConLL-X format. Next to the guidelines and datasets, UD also offers a collection of scripts in Perl and Python.[34]
UD is an open community effort with over 500 contributors, offering over 245 treebanks and 141 languages, collected as a set of GitHub repositories[35], but also offered through other channels, e.g. available as one dataset in the Czech LINDAT repository[36], and also for querying online via TEITOK[37] maintained by the Charles University in Prague.

3.2.3 Penn Treebank Format

The Penn Treebank, developed already in the 1990s, is one of the classical, notorious datasets for the syntactic parsing tasks. The format this dataset is encoded in, defines a tagset for part of speech as well as for syntactic annotation and uses brackets to encode the syntactic tree.
Example of a Penn Treebank-style annotation[38]:

```
(IP-MAT (NP-SBJ (PRO I))
        (VBD saw)
        (NP-OB1 (D the)
                (N man)))
```

3.2.4 Brat Standoff Format

The Brat Standoff format is connected to the open-source "Brat Rapid Annotation Tool"[39], which was "initially created as an extension of *stav*, an annotation visualisation tool"[40] in 2011.
Annotations are stored in a file separate from the annotated document text, which is never modified by the tool.
Example of a BRAT-annotation[41]:

```
T1              Organization 0 4                    Sony
T2              MERGE-ORG 14 27                     joint venture
T3              Organization 33 41                  Ericsson
E1              MERGE-ORG:T2 Org1:T1 Org2:T3
T4              Country 75 81                       Sweden
R1              Origin Arg1:T3 Arg2:T4
```

3.2.5 AI-models and Training Data

With ML-based approaches the trained models, as well as the training data used to train the model represent a resource on their own. To allow for reuse and also for reproducibility of research results, it is essential to make training data and models available and well documented (including information

[33] https://universaldependencies.org/guidelines.html
[34] https://github.com/UniversalDependencies/tools
[35] https://github.com/UniversalDependencies
[36] http://hdl.handle.net/11234/1-5150
[37] http://lindat.mff.cuni.cz/services/teitok/ud212/
[38] https://www.ling.upenn.edu/~janabeck/tutorial.html
[39] Manual for the Brat Annotation Tool: https://brat.nlplab.org/manual.html, Overview of Brat features: https://brat.nlplab.org/features.html
[40] https://brat.nlplab.org/about.html
[41] https://brat.nlplab.org/standoff.html

about the data sources, preprocessing steps, model architectures, and hyperparameters.) However, even though lot of energy and computing resource go into generating AI models, their encoding (i.e. the neuronal weights and the shapes of their matrices) including their configuration parameters are intrinsically strongly linked/tied with the systems they have been generated with, and they are in general not freely reusable across different libraries/systems, since these are direct implementation of specific ML architectures or types of neural networks employed, which are highly complex and can differ greatly. Where architectures are similar, transformations of models between libraries might be possible, by transferring the neuronal weights from one structure into another, but due to the high data complexity, this is an error-prone process, and usually it is more economical to just use the same training data to retrain the model. Oftentimes, models are not even compatible across different versions of the same library, especially if the underlying ML architecture has changed substantially, which is often the case given the rapid development of this field. There are efforts however to mitigate this issue, foremost ONNX[42], an open format built to represent machine learning models, and to allow the use of models with a variety of frameworks, tools, runtimes, and compilers.

There is also no standard format for the training data serving as input for training the models, individual libraries using bespoke data structures, dictated by the needs of the processing components. These data structures are usually conveyed in a pragmatic fashion by means of documentation and example files, relying on wide-spread conventions in the field rather than by defining a strict schema. To mention but one example, the documentation and tutorial material of the NLP library spaCy where the training data is all simple json files.[43] The formats are correspondingly considered subject to change, with no guarantee for long-term support. Rather it is the practice to tweak and transform the data with simple ad hoc scripts, to match the expected input structure of a given library's API. Typically, the data is encoded in JSON (JavaScript Object Notation), "an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute-value pairs and arrays."[44] [45]

An example JSON structure commonly used in spaCy's NER processing could look like this, where cooking ingredients are recognized (superfluous elements are replaced with '...')[46]:

```
{
    "text": "Cream cheese is really good in mashed potatoes.",
    …
    "spans": [
        {
            "text": "Cream cheese",
            "start": 0,
            "end": 12,
            "label": "INGRED",
            "token_start": 0,
            "token_end": 1
        }
    ],
    "tokens": [
        {"text": "Cream","start": 0, "end": 5,"id": 0},
```

[42] https://onnx.ai/

[43] https://github.com/explosion/projects/tree/v3

[44] https://en.wikipedia.org/wiki/JSON

[45] Note: JSON's underlying data structure is a hierarchical tree, so formally it would belong into section 3.1.

[46] https://github.com/explosion/projects/blob/v3/tutorials/ner_food_ingredients/assets/food_data.jsonl

```
            {"text": "cheese","start": 6,"end": 12,"id": 1},
            {"text": "is","start": 13,"end": 15,"id": 2},
            {"text": "really","start": 16,"end": 22,"id": 3},
            ...
        ],
        ...
    }
```

spaCy framework delivers also an extreme counterexample for interoperability. Changing from version 2 to 3, the framework deprecated the JSON input format in favour of the binary serialisation format spaCy DocBin[47], encoding a collection of spaCy's internal Doc objects, optimised for space usage and ease of consumption by the spaCy library. Consequently, it is exclusively tied to SpaCy and the data encoded in the format can only be read and modified by it in a python runtime, ruling out inspection through generic text editors, as well as processing with third-party code. As such it is a prime example of a format unsuitable for exchange or long-term archiving.

Even before having the model or the training data in the right format, the first challenge is to be able to find it and access it. Regarding findability, the platform HuggingFace[48] evolved as the go-to repository and marketplace for ML datasets and models.

As for the accessibility, besides traditional datasets of serialised data in the shape of classical files available for download, more and more, data can be loaded from external public repositories directly into a python runtime with the help of loader APIs, as is the case with many high-quality python NLP libraries / platforms. As an example, HuggingFace makes data retrieval trivial, by simply passing dataset identifiers to a python function, which downloads the data from the Hugging Face repositories and makes it available directly in the python runtime as python dictionary objects. The downloaded data is cached in the background, to avoid redundant downloads, and its files are not made directly visible to the developer. The resulting data (in the shape of enhanced python dictionaries) hence lives only in the python runtime and its structure is defined by the conventions of Hugging Face. Similar direct data loading functions are also provided by other popular ML libraries, such as TensorFlow[49] or PyTorch[50].

The underlying technical serialisation format for the datasets at Hugging Face parquet[51], a column-oriented format developed by the Apache foundation. It is a column-oriented format, designed for computational efficiency. As a custom non-plaintext format, it also requires dedicated tooling to read from and write to. Hence, it is more suitable for data exchange between components within a strongly coupled infrastructure, rather than for a general exchange between users and project contexts.

[47] https://spacy.io/api/docbin
[48] https://huggingface.co/
[49] https://www.tensorflow.org/
[50] https://pytorch.org/
[51] https://parquet.apache.org/

3.3 Graph-based Formats

There are a number of formats with graphs as the underlying model, most of them based on RDF. A big advantage of such approaches is the ability to express "overlapping" phenomena, where usual hierarchical tree structures (expressed in XML) fall short or require workarounds.

3.3.1 RDF – Resource Description Framework

RDF[52] is a W3C standard for representing and linking structured data on the web. RDF underlying data model being a graph, it does not lend itself well for representation of sequential data such as text, but can be used to annotate and describe textual content in a semantically rich way, making it more machine-readable and facilitating data integration and linking.
RDF is especially suitable for expressing relationships both within and between texts, and enriching textual data with semantics by linking expressions/text units to semantic reference resources, forming the large LOD cloud which serves as a natural global reference multi-resource (e.g. linking mentions of a person to their representation in Wikidata). These linked resources can be considered auxiliary structured data and are typically available in RDF.
RDF itself is a generic framework, which requires definition of a vocabulary or ontology to model a specific domain. NLP Interchange Format and Web Annotation Model described below are two examples of RDF-based models or formats.
RDF is also often used – and indeed was originally designed – to encode metadata about resources. Also, the CLSCor data model proposed in D6.1 for describing corpora is formalised as a RDF ontology. The initiative Linguistic Linked Open Data[53] gathers a large array of language-related resources following the principles of LOD.

3.3.2 POSTDATA – Ontology for European Poetry

POSTDATA[54] was a European project aiming to make poetry available as machine-readable data, to standardise description and encoding of poetry works and to publish the data on poetic works and their analysis as LOD. As part of this effort the project developed an abstract model for the representation of poetry and formalised it as a set of interconnected ontologies[55], dealing with different aspects of a poetic work: the work as creative product of an actor (author) and potentially many contributors together with potential complementary entities, such as illustrations; its structural and prosodic elements, the literary analysis, etc.
For more information see also D7.2 section 6.2, (González-Blanco Garciá et al. 2022).

[52] https://www.w3.org/TR/rdf11-concepts/
[53] https://linguistic-lod.org/
[54] https://postdata.linhd.uned.es/
[55] https://postdata.linhd.uned.es/results/network-of-ontologies/

3.3.3 LAF – Linguistic Annotation Framework

The Linguistic Annotation Framework (LAF) is a standard, developed since 2001 and specified by ISO 24612:2012[56] with the purpose of representing multi-modal linguistic annotations of language data (e.g., language corpora, audio files, etc.) and "providing full interoperability among annotation formats" (Lapponi, et al., 2014). This interoperability is achieved through an abstract graph-based data model provided by the LAF and its serialisation in XML: GrAF (cf. section 3.1.3).
"The complete LAF data model ultimately includes (1) a structure for describing media, consisting of anchors that reference locations in primary data, and regions defined in terms of these anchors; (2) a graph structure, consisting of nodes, edges, and links to regions; and (3) an annotation structure for representing linguistic information with feature structures." (Ide and Suderman, 2014, p. 3) "(4) provision for URI-based references to linguistic categories defined in existing repositories as a means to achieve semantic interoperability." (Ide, et al., 2017, HLA, p. 127) These mechanisms allow to inter-link linguistically-annotated resources, making LAF a pre-cursor of the RDF/OWL "Linguistic Linked Data''. Thanks to the underlying graph-based model, LAF is expressive enough to accommodate most other formats and is trivially mappable to RDF/OWL. As such, it represents a major step toward syntactic interoperability.

3.3.4 LIF – LAPPS Interchange Format

The LAPPS interchange format, short LIF, was created in context of the Linguistic Applications (LAPPS) Grid.[57] The LAPPS Grid can be described as the "transatlantic counterpart of WebLicht [...], an open, web-based infrastructure that offers a very good range of language-related tools." (Zinn, et al., 2022, p. 86)
LIF is a JSON-LD based schema and serves for the exchange and transport of data between LAPPS services. The current schema is available under http://vocab.lappsgrid.org/schema/lif-schema.json and uses elements of the LAPPS Web Services Exchange Vocabulary (WS-EV)[58] , which "provides a terminology for a core of linguistic objects and features exchanged among NLP tools that consume and produce linguistically annotated data. The goal is to bring the field closer to achieving semantic interoperability among NLP data, tools, and services." (Verhagen, et al., 2016)

3.3.5 NIF – NLP Interchange Format

The NLP Interchange Format (NIF)[59] is an RDF/OWL-based format that aims to achieve interoperability between Natural Language Processing (NLP) tools, language resources and annotations. NIF consists of specifications, ontologies and software. "NIF addresses the annotation interoperability problem on three layers: the structural, conceptual and access layer. NIF uses a Linked Data enabled URI scheme for identifying elements in (hyper-)texts that are described by the NIF Core Ontology (structural layer)

[56] https://www.iso.org/standard/37326.html
[57] https://www.lappsgrid.org/, Wiki: https://wiki.lappsgrid.org/
[58] http://vocab.lappsgrid.org/
[59] https://nif.readthedocs.io/en/latest/

and a selection of ontologies for describing common NLP terms and concepts (conceptual layer)." (Ide, et al., 2017a, p. 104)
NIF uses character offset to address string sequences and further describe them with a set of RDF properties.
Originally developed in the context of the NLP2RDF project at the University of Leipzig over 10 years ago, its further development and adoption seems to be stalled.

3.3.6 Web Annotation Model (W3C)

"The Web Annotation Data Model[60] provides an extensible, interoperable framework for expressing annotations such that they can easily be shared between platforms, with sufficient richness of expression to satisfy complex requirements while remaining simple enough to also allow for the most common use cases, such as attaching a piece of text to a single web resource. Annotations are typically used to convey information about a resource or associations between resources. Simple examples include a comment or tag on a single web page or image, or a blog post about a news article."
Thus, the Web Annotation Model is not meant to encode text, however allowing to establish relations between any resources, it can be used to encode certain kinds of annotation, notably semantic entity linking.
Web Annotation Model defines an annotation as a *Web Resource*. Typically, an *Annotation* has a single *Body*, which is a comment or other descriptive resource, and a single *Target* that the *Body* is somehow "about". The *Annotation* likely also has additional descriptive properties.
The Web Annotation Vocabulary[61] specifies the set of RDF classes, predicates and named entities that are used by the Web Annotation Data Model. It also lists recommended terms from other ontologies that are used in the model and provides the JSON-LD Context and profile definitions needed to use the Web Annotation JSON serialisation in a Linked Data context.

4. Challenges

This chapter reflects on some general challenges when dealing with formats.

4.1 Formalising the Description of the Data Structure

As elaborated in D6.1 (sections: 5.5.5 Format / Schema and 6. Transformation Matrix) "format" in the sense of indication of the inner structure of the data or documents is a complex phenomenon. Though in everyday use it is approximated by the file extension, this is in many cases just a very rough (and oftentimes unreliable) indication of the potential inner structures, most notably for text and XML-based formats. Schemas are a way to formally describe possible data/document structures and to validate documents.
Additionally, the framework, model, format, and serialisation need to be differentiated. While framework and model refer to the abstract conceptualisation of some domain of discourse, format

[60] https://www.w3.org/TR/annotation-model/
[61] https://www.w3.org/TR/annotation-vocab/

and serialisation refer to the actual encoding or representation in data structures, which can be processed and stored in digital form.
As stated in D6.1 (section 5.5.6), more fine-grained information about specific features a document exhibits beyond the indication of format or schema is needed to understand its inner structure, and to allow assessing the suitability for particular re-use. Typical examples of such features are lemmatisation or PoS-tags, but it could be any semantic or structural aspects, like paragraphs, utterances, verses, person names, references etc. These features are encoded in the various formats as some form of annotations or markup.
Individual processing tasks/tools require specific features to be present in the data to operate on. (For example, network analysis of speakers in a drama requires the speakers to be explicitly annotated in the text.) Therefore, a more fine-grained distinction recognizing specific atomic features of a digitally encoded text is needed, be they structural (e.g., acts or speakers in a dramatic work), linguistic (e.g., lemmas, PoS-tags, morphological information), semantic (e.g., named entities, including persons and places), text-genetical / text-critical or intertextual features.
A useful generic distinction was formulated by Ide et al. 2017: "Generally, standardization for representing language resources deals with two phenomena: the *representation format* (syntax) and the *data categories* used to identify linguistic phenomena in the resource. Each poses its own problems for standardization, although standardization of linguistic categories is substantially more problematic because of (sometimes subjective) differences in definitions, granularity, theoretical orientation, etc. […] Recent times have seen considerable convergence of practice for representation format as well as more general agreement on the means and mechanisms by which to provide "semantic interoperability" via standardized data categories among resources. Nevertheless, to date, no single, universally accepted set of best practice guidelines for either of these concerns for the creation of linguistically-annotated resources exists." (Ide, et al., 2017, HLA, p. 114-115)
Given that there is "no universally accepted set of best practice guidelines" in the NLP, let alone CLS community, the proposed CLSCor data model introduces a generic mechanism to express information about the presence of individual features in corpora or their constituent documents, see chapter *6.*

4.2 Format Variability / Expressivity

There is a notorious trade-off between expressivity, complexity and applicability of any given format. The more restricted, the easier to handle, but at the same time the more limited its scope of application. On the other hand, flexible formats which can be customised to fit a broad range of scenarios are more difficult to process. TEI is an extreme example of such a flexible customisable format. Its customisability allows it to be applied for many different tasks, however the corresponding customisations are so far flung that it is difficult for processing tools to cover the whole range. However, most of the formats introduce some extension mechanisms, allowing to introduce custom annotation layers, or custom sets of data categories (like a specialised tagset).
This high degree of flexibility, needed and valued in scientific projects, makes the automatic processing and harmonisation of data and metadata more challenging, potentially requiring project- or dataset-specific adaptations to any processing, especially transformation paths. This task is further complicated by the fact that not all projects sufficiently document their considerations - encoding guidelines and ODDs/schemas are not always made publicly available.

In general, this challenge needs to be addressed by a robust processing pipeline able to deal with unknown structures and missing information, at least by exiting gracefully with useful diagnostic messages about the encountered problems. Additionally, generic inspection tools can be included into the processing pipelines to identify the actual structures present in the data (e.g. which and how many elements and attributes are actually present in the documents, as opposed to those allowed by the schema).

4.3 Metadata

As already elaborated on in D6.1, one aspect to consider when dealing with corpora is the distinction between the actual content and the metadata describing it, and how these are represented in different formats. Some formats include metadata as part of the content files, most notably the infamous *<teiHeader>* in the TEI files. Alternatively, there are a number of dedicated metadata formats (like dublincore[62], CMDI[63], METS[64]/MODS[65], MARC[66], CIDOC-CRM[67]/FRBRoo[68], BIBFRAME[69] etc.).

There are also different kinds of information to be captured by the metadata:

- Information on the corpus as a whole, who is responsible for its creation, what are its characteristics and underlying motivation for its creation.
- Information on the individual documents the corpus is a collection of. In a typical corpus of literary works this would be chiefly author, title, publication date, etc. but depending of the items constituting the corpus, this is potentially a rich set of descriptive attributes
- Technical information about the format of the documents and encoding of specific features. This can be ideally expressed by reference to a schema, accompanied by human-readable documentation of the editorial/annotation guidelines.

The CLSCor data model devised in WP6, conceptually grounded in the CIDOC CRM family of conceptual models, aims to allow expressing all this information in a systematic manner. Existing heterogeneous metadata about corpora, as well as information on document level potentially available inside the corpus data shall be collected and consolidated as part of the work on populating the CLSCor catalogue. This will require format or even source-specific custom processing pipelines to gather and convert the available information. By doing so, the CLSCor catalogue will allow a harmonised structured view on the currently scattered, distributed corpora landscape.

[62] https://www.dublincore.org/specifications/dublin-core/dcmi-terms/
[63] https://www.clarin.eu/content/component-metadata
[64] https://www.loc.gov/standards/mets/
[65] https://www.loc.gov/standards/mods/
[66] https://www.loc.gov/marc/
[67] https://www.cidoc-crm.org/
[68] https://www.cidoc-crm.org/frbroo/
[69] https://www.loc.gov/bibframe/

4.4 Tokenisation

Tokenisation – the task of breaking down a text into individual words or tokens – is typically a critical initial task in NLP pipelines, with any downstream processing relying on it. However, different rules/approaches lead to inconsistent/different tokenisations, influencing further processing and especially making it impossible/troublesome to unify/compare results from different pipelines. Some of the phenomena where tokenisation conventions can differ are:

- word contractions ("can't" for "cannot" or "I'm" for "I am").
- hyphenated words: ("mother-in-law")
- compound words
- punctuation (esp. end of sentence vs. abbreviation)
- special characters
- whitespace and line breaks (in languages with complex scripts like Thai or Japanese, word boundaries are not always clear from whitespace)
- tokenization in non-segmented languages
  (some languages, like Chinese and Japanese, do not use spaces to separate words)
- out-of-vocabulary words
- named entities

5. Converting between Formats

One of the core objectives of the CLS INFRA project is to ease the application of existing tools for NLP and other tasks on existing and new datasets by bridging the format gap, i.e. the situation, where data is available in a different format than expected by a given tool. For example: While an individual corpus document may be encoded with JSON, its metadata might be expressed in a separate file using the Dublin Core standard. On the project homepage, the whole corpus including corpus documents as well as their metadata information might be available for download as a zip-file. A tool used for analysis on this very corpus might expect as input format a plain text file to perform a task (e.g., annotation) and then provide a TEI file for output/export.

There exist multiple approaches to tackle format conversion:

- **Stand-alone conversion tools**
  There already exist numerous conversion tools, most notably the generic Pandoc[70] – „a universal document converter", and TEIGarage[71] converting documents between TEI and many other formats. Numerous further conversion tools are collected in the Conversion Hub[72] developed in the SSHOC project.
- **Research platforms integrating processing**

[70] https://pandoc.org/
[71] https://teigarage.tei-c.org/
[72] https://conversion-hub.sshopencloud.eu/

Integrated frameworks or research environments/platforms, which integrate the processing as part of their functionality, handling the interfacing with external tools internally. Most prominent example in the context of CLS INFRA being TEITOK[73] or TXM[74] platforms

- **Data available in different formats**
  Data providers may expose data already in multiple formats. This is at the core of the concept of Programmable Corpora (cf. D7.1), introducing a set of APIs allowing harmonised, yet multifaceted access to data.
- **Tools supporting multiple formats**
  Individual tools may provide support for multiple input and output formats, basically incorporating the conversion to internal processing. (e.g. FLAT[75])

A special case is the framework Weblicht[76] developed in the context of CLARIN. It relies on a distributed set of task-specific services with harmonised interface, i.e. all speaking the same language (TCF format) and exposes to the user a user interface to flexibly build tool chains out of these building blocks, allowing data in a number of formats as initial input to these chains, performing the conversion internally.

In practice, these approaches are combined in various ways, so that for example the platform DraCor delivers not only the raw full text of the corpus in multiple formats, but offers also auxiliary resources, derived from the original input documents, like the character network in a drama, or various precalculated metrics, basically incorporating certain processing steps and transformation and exposing only the resulting output.

Also, on the side of the tooling, the tendency is towards integrated pipelines, performing a number of NLP tasks, rather than multiple separate tools each performing one task, an example being UDPipe[77], performing tokenization, tagging, lemmatization and dependency parsing taking files in CoNLL-U format as input. This reduces the need for format conversion between individual tools.

And finally, there is a myriad of project-specific solutions implemented as ad hoc conversion scripts. Where feasible and useful, some of these solutions existing in the context of consortium partners can be consolidated and integrated as part of the Transformation Toolbox.

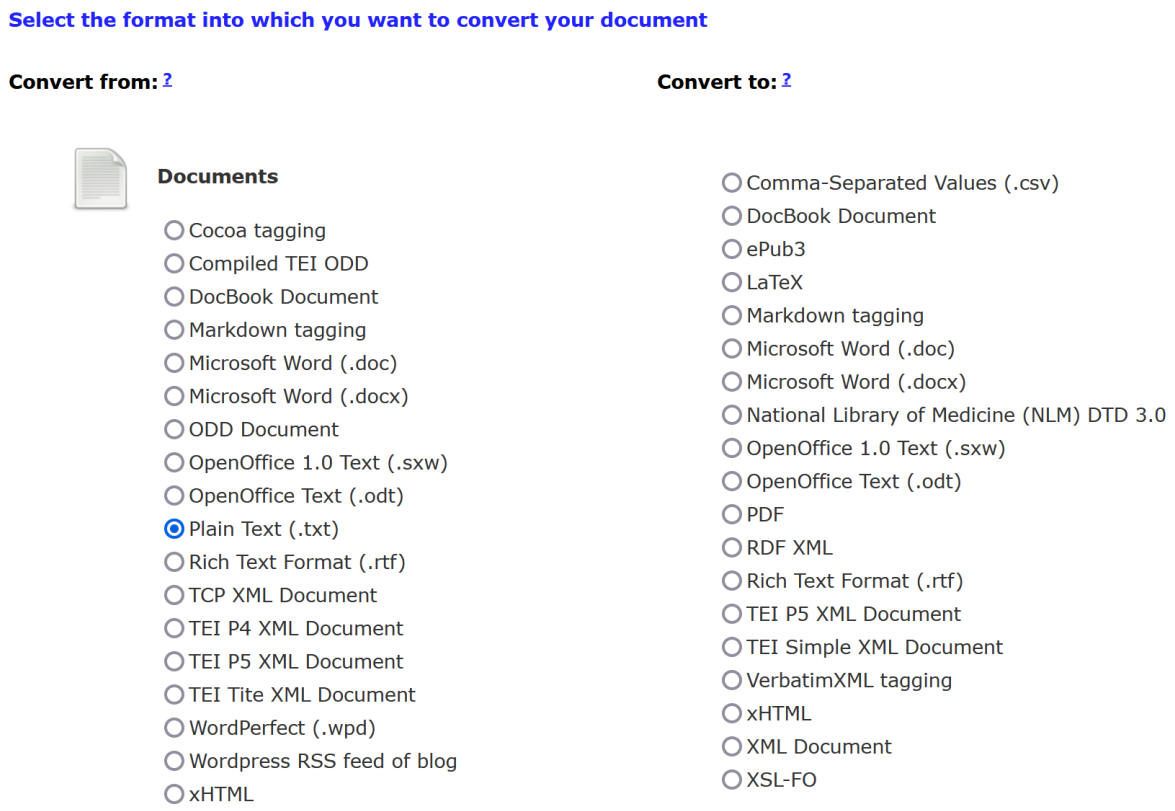
One more related aspect is the availability of and the mode of access to the data. As the analysis in D6.1 found, the raw data of a corpus is made available in a very heterogeneous manner, if at all. It may be published via a git repository, as a link to a zip file on the corpus website, or through an API. This aspect also needs to be considered when building processing pipelines.

Next to conversion between formats needed for further processing, one very frequent conversion is that of documents (typically in TEI) to HTML for an online representation within the corpus website or maybe as part of a digital edition. While oftentimes this representation is generated dynamically on the fly (upon user request) as part of some server-side processing, an alternative increasingly popular

[73] http://teitok.corpuswiki.org/
[74] https://txm.gitpages.huma-num.fr/textometrie/en/Presentation/
[75] https://flat.readthedocs.io/
[76] https://weblicht.sfs.uni-tuebingen.de/weblicht/
[77] https://lindat.mff.cuni.cz/services/udpipe/info.php

approach is to generate static (HTML-)pages for all the content only when there are changes to data and serve this pre-generated static pages. This approach greatly reduces the resources needed to serve the content and makes the resulting web application much more stable and maintainable. CETEIcean[78], another recent approach to present TEI data as styled content in web browser, is a JavaScript library, which does not convert TEI at all, but relies on the emerging Web Components[79] standards to display TEI data directly.

Figure 1: Snapshot of the conversion form of TEIGarage offering a list of source and target formats to convert between



When converting data between formats, the crucial question is how much of the information can be retained in the target format, depending on its expressivity. Comprehensive lossless (isomorphic) conversion between feature-rich formats is seldom possible and very tedious and time-consuming to implement. In practice, usually the transformation is lossy, i.e. only selected features depending on the task at hand are extracted and encoded in the new format. In the typical case of applying NLP on TEI texts this oftentimes means extracting a flat sequence of tokens or simple plain text, i.e. stripping all markup, as most NLP tools expect plain text as input.
This seemingly rather trivial default transformation path from TEI to plain text gets more complicated with the additional requirement that the features generated by the NLP tools should be integrated

[78] https://github.com/TEIC/CETEIcean
[79] http://webcomponents.org

with the original TEI document. This requires an alignment of the processing pipeline with respect to the tokenisation, pertaining to the token sequence and token identity between the tasks.
The task becomes even more challenging with complex annotation layers identifying structures spanning multiple tokens, especially phrase structures or dependencies.

5.1 Transformation Matrix – Update

Transformation Matrix proposed in D6.1 serves as conceptual grounding for the format conversion task. When instantiated and populated it shall become a systematic overview of formats the corpus data is available in, laid out against the formats the tools to become part of the infrastructure accept as input, or produce as output, indicating potential/needed transformation paths. This overview, which will be an integral part of the CLS catalogue, will also serve as input for task 6.2, which is to deliver a Transformation Toolbox (D6.2, month 48), a set of scripts or tools allowing to convert data between formats. Information provided in this deliverable D6.3 about formats and their usage will feed into the Transformation Matrix.
In practice, format, schema, individual features, as well as methods are intricately intertwined. As an example: if one wishes to apply the *method* part of speech tagging with *tool* A on a set of documents from *corpus* C, the tool must support the *format* of the corpus documents. Furthermore, depending on the tool, the document must feature explicitly annotated tokens, a result of the method tokenisation. On the other hand, in *tool B*, the tokenisation may be already included as part of the internal NLP pipeline, however it may expect a different format on the side of the input.
This understanding implies the main dimensions of the Transformation Matrix: *format, feature* and *method*. For details of the formalisation of these dimensions see chapter 6.

6. Update on the CLSCor Data Model

D6.1 introduced a first draft of the CLSCor data model for describing corpora. In the following, we present further evolvement of the data model with focus on mechanisms for describing the structure of a set of documents.

6.1 Format, Schema, Feature, Method

As elaborated in D6.1, formally describing structural aspects is challenging due to the diversity and variability of formats used to encode the data and the broad range of annotation dimensions to be covered. A set of classes and properties introduced to capture this information is listed in Table 3.

Table 3: Classes introduced in CLSCor for capturing information about data structure

class	subClassOf	property
X3_Feature	crm:E73_Information_Object	crmcls:Y1_exhibits_feature
X6_Method	crm:E29_Design_or_Procedure	
X7_Format	(owl:equivalentClass) pem:PE43_Encoding_Type	crmcls:Y2_has_format
X8_Schema	(owl:equivalentClass) pem:PE38_Schema	crmcls:Y3_adheres_to_schema

- **Format** – roughly corresponds to file format.

This is globally formalised via IANA media-type[80], however this global authority is not ideal, as it provides a large number of media-types irrelevant in the scope of CLS, and on the other hand offers only a very coarse-grained classification for text- or XML-based formats.

- **Schema** – defines the structure of a document. In XML this means the definition of elements and attributes and the encoding pattern in which combinations these may appear. Feature-rich customizable formats like TEI, allow a lot of flexibility regarding the allowed structures. These are formally expressed in custom schema documents.
- **Feature** – While schema defines what is allowed and applies to the document as whole, the entity Feature allows to capture individual (atomic) aspects a document or a set of documents exhibits.
- **Method** – is a defined procedure, specifically procedure for processing/enriching/annotating text. Typical examples are lemmatisation or named entity recognition. There is almost a one to one relation between methods and the features they generate.

6.2 Corpus, Corpus Document, Prototypical Document

To capture information about the corpus, or about the documents in the corpus, following three classes are introduced:

- **crmcls:X1_Corpus**
  (rdfs:subClassOf lrm:F3_Manifestation, crmdig:D1_Digital_Object)
  represents the collection of documents
  Example properties: number of documents in corpus, number of words in corpus, corpus type
- **crmcls:X2_Corpus_Document**
  (rdfs:subClassOf lrm:F3_Manifestation, crmdig:D1_Digital_Object)
  represents a specific individual document
  Example properties: language, format, feature, licence
- **crmcls:X11_Prototypical_Document**
  (rdfs:subClassOf crm:E55_Type)
  a surrogate representing documents in a corpus, assumed to have certain properties. This entity allows dealing with the problem of uncertainties when providing information on document level, e.g. from a general project description, or through anecdotal inspection, as opposed to actual inspection or processing of every single document.

Example of relations between instances of these three classes:

```
corpus:corpus1 a crmcls:X1_Corpus.
corpus:protodoc/1 a crmcls:X11_Prototypical_Document.
corpus:document/1 a crmcls:X2_Corpus_Document;
     crm:P137_exemplifies corpus:protodoc/1; # just if protodoc/1 is based on examining document/1
corpus:corpus1 crm:P148_has_component corpus:document/1, corpus:protodoc/1.
```

[80] https://www.iana.org/assignments/media-types/media-types.xhtml

to indicate corpus documents have/inherit features of the prototypical document:

```
corpus:document/1 crm:P2_has_type corpus:protodoc/1
corpus:document/2 crm:P2_has_type corpus:protodoc/1
```

6.3 Generic Attribute Assignment – Dealing with Dualities and Uncertainties

The representation of metadata on formats, schemas, features (annotations) and methods in the CLSCor data model has undergone some changes and extensions/additions since the data models original introduction in D6.1. The most important change is the replacement of the so called "should-have"-properties: The properties "Y2_documents_should_have_language", "Y3_documents _should_cover", "Y4_documents_should_have_type" and "Y5_documents_should_have_feature" were initially "introduced to allow to express uncertainty when capturing certain characteristics of a corpus based on some second-hand evidence, e.g., description on a corpus website." (D6.1, 2022, p. 16) However the semantics of these properties was not well grounded conceptually, relying solely on the verbose definition. Thus, in the meantime, these were superseded in favour of the well-established reification mechanism of attribute assignment (`crm:E13_Attribute_Assignment`) introduced by CIDOC CRM. This construct allows to enrich every piece of information about a corpus or a document with its provenance, or any other contextual information, ensuring transparency and comprehensibility. It also allows to formally express conflicting pieces of information, e.g. information about number of tokens from the corpus description on the website, vs. number of tokens as reported by some analytical script, which actually processed the corpus content.

The general pattern for individual attribute assignments looks as follows:

```
corpus:attrassign/1 a E13_Attribute assignment ;
     crm:P140_assigned_attribute_to corpus:protodoc/1 or corpus:corpus;
     crm:P177_assigned_property_of_type {property}
     crm:P141_assigned {literal-value or entity reference}
```

6.4 Corpus Description Event

Individual attribute assignments are grouped under a `crmcls:X9_Corpus_Description` event, which allows to efficiently capture, who, when and based on which sources the information about the corpus was produced:

```
corpus:descevent/1 a crmcls:X9_Corpus_Description
     crm:P16_used_specific_object {reference to the information source – typically corpus
             website}.
     crm:P14_carried_out_by {Actor}
     crm:P4_has_time-span {Time-Span}

# connecting an attribute assignment to the corpus description event:
corpus:attrassign/1 crm:P9_consists corpus:descevent/1 ;
```

6.5 Assigning Quantities, Measurements and Features

When expressing some quantities or measurements (e.g. number of tokens in a document), the attribute assignment pattern is slightly adjusted making use of the class `crm:E54_Dimension`:

```
corpus:attrassign2 a crm:E13_Attribute_Assignment;
     crm:P9_consists corpus:descevent/1;
     crm:P140_assigned_attribute_to {instance of one of crmcls:X1_Corpus,
             crmcls:X2_Corpus_Document, crmcls:X11_Prototypical_Document};
     crm:P141_assigned corpus:dimension/1.

corpus:dimension/1 a crm:E54_Dimension ;
     crm:P90_has_value {xsd:Integer};
     crm:P91_has_unit {crm:E58_Measurement_Unit}.
```

The class crmcls:X3_Feature being a subclass of `crm:E58_Measurement_Unit`, all instances of crmcls:X3_Feature can be used as units for describing some dimension of a corpus or a document. The property `crm:P140_assigned_attribute_to` takes as object an instance of either crmcls:X1_Corpus, crmcls:X11_Prototypical_Document or crmcls:X2_Corpus_Document.
To merely express presence of a feature in a document (assumed based on indirect information) one can use a prototypical document as proxy:

```
clscore:paraX a crmcls:X3_Feature.
corpus:protodoc/1 crmcls:Y1_exhibits_feature clscore:paraX.
clscore:paraXencode a crmcls:X10_Encoding_Pattern;
crmcls:Y5_encodes clscore:paraX.
corpus:document/1 crm:P2_has_type corpus:protodoc/1.
```

Alternatively, to quantify a feature in a document, it needs to be assigned directly:

```
corpus:attrassign/3 a crm:E13_Attribute_Assignment;
     crm:P134_continued corpus:descevent/1;
     crm:P140_assigned_attribute_to corpus:document/1
     crm:P141_assigned corpus:dimension/2.
corpus:dimension/2 a crm:E54_Dimension
crm:P90_has_value {xsd:Integer};
crm:P91_has_unit clscore:paraX.
```

6.6 Vocabularies

To allow formalised descriptions, to process information about formats and features automatically, and thus to enable matchmaking between data and tools, a **harmonised, consolidated vocabulary or nomenclature** is required. As shown in the overview of formats, there exists no one agreed upon vocabulary, but rather a multitude of attempts to codify both the features or annotation layers, as well as the allowed value ranges, i.e., tagsets, for these layers. Consequently, the WP6 team introduced bespoke opportunistic vocabularies adjacent to the CLSCor data model, based on the empirical evidence collected in preceding project activities (surveys of user practice, corpus data landscape and existing tooling), as well as input from fellow scholars. These vocabularies are open, subject to further additions and consolidations, based on new information about the data or tools to be integrated.
Apart from the vocabularies for corpus types, languages, licences, literary genres, appellation types and link types, vocabularies for features, formats and methods are currently under development.

The modus operandi is to collect the vocabulary terms in spreadsheets and then serialise them from CSV into RDF/TTL using a Jinja[81] template. This automated approach allows easy and fast updating of the vocabularies. The resulting RDF representation, currently stored in the project gitlab repository next to the CSV source table, will be published as a SKOS vocabulary in the DARIAH Vocabs repository[82] and also used directly in the CLSCor catalogue.
Example of definition of a Feature and a Method concept as vocabulary terms serialised in RDF/TTL:

```
<https://core.clscor.io/entity/type/method/named_entity_recognition> a skos:Concept,
        crmcls:X6_Method ;
    skos:exactMatch <https://vocabs.dariah.eu/tadirah/namedEntityRecognition> ;
    skos:inScheme <https://core.clscor.io/entity/type/method> ;
    skos:prefLabel "named entity recognition" .

<https://core.clscor.io/entity/type/feature/token> a skos:Concept,
        crmcls:X3_Feature ;
    skos:inScheme <https://core.clscor.io/entity/type/feature/> ;
    skos:prefLabel "named entity" .

<https://core.clscor.io/entity/type/method> a skos:ConceptScheme ;
     skos:hasTopConcept <https://core.clscor.io/entity/type/method/named_entity_recognition> ,
        <https://core.clscor.io/entity/type/method/relation_extraction> .

<https://core.clscor.io/entity/type/feature> a skos:ConceptScheme ;
     skos:hasTopConcept <https://core.clscor.io/entity/type/feature/token>.
```

Each term in the vocabulary is both a skos:Concept as well as an instance of the respective CLSCor class – crmcls:X3_Feature or crmcls:X6_Method.

6.6.1 Vocabularies for Features and Methods

The initial assumption when creating the vocabulary for features and methods was, to view a feature as an output produced by a method - i.e., the method "named entity recognition" produces the feature "named entity". The inventory of methods and features was correspondingly structured as a list of pairs. However, since the relation between an applied method and its resulting feature is not always 1:1, the initial inventory served only as a starting point to seed the respective vocabularies for features and methods, which can be continuously extended based on new information.
As a convention, the singular form of a feature is used for the terms in the vocabulary, e.g., part-of-speech-tag, syllable, word, etc.

Note that the model embraces a very broad, general interpretation of the Feature concept, covering all aspects of a text, both linguistic annotations (e.g., tokens, tags, lemmas), as well as structural information (e.g., paragraphs, verse lines, acts, sentences). It is even extended to describe certain aspects of the corpus (e.g., number of documents).

[81] https://jinja.palletsprojects.com/en/3.1.x/
[82] https://vocabs.dariah.eu/

The methods are mapped to the TaDiRaH[83] taxonomy, wherever possible, using the semantic relations skos:exactMatch or skos:broaderMatch following the LOD paradigm.

6.6.2 Vocabulary for Formats

Even though with IANA Media Types[84] there is a global authority file defining file formats, it does not fit very well the specific situation and needs of the CLS INFRA project. It is on the one hand too general – it lists a great number of formats not relevant in the context of CLS INFRA -, and on the other hand too coarse grained – it does not provide sufficient coverage for the formats of interest.
Thus to better tailor the vocabulary of formats to the needs of CLS INFRA, a bespoke vocabulary was created compiling formats mentioned in the corpusTable and in the deliverables D3.1 and D8.1. The formats in the vocabulary were mapped to IANA Media Types where possible, using SKOS semantic relations (skos:exactMatch/skos:broaderMatch).
As is the case with methods and features, each term is modelled both as a skos:Concept as well as an instance of crmcls:X7_Format.

Example for a term from the format vocabulary:

```
<https://core.clscor.io/entity/type/format/csv> a skos:Concept,
          crmcls:X7_Format ;
      rdfs:label "CSV" ;
      skos:exactMatch <media-type:textslashcsv> ;
      skos:inScheme <https://core.clscor.io/entity/type/format> ;
      skos:prefLabel "CSV" .
<https://core.clscor.io/entity/type/format> a skos:ConceptScheme ;
      skos:hasTopConcept <https://core.clscor.io/entity/type/format/csv> .
```

7. Conclusion and Outlook

This deliverable offers an overview of common formats as observed in the realms of Computational Literary Studies and Computational Linguistics or Natural Language Processing, both regarding data encoding and with respect to the input and output formats of the relevant tooling. It also describes the challenges when dealing with and especially converting between formats.
Further work is needed on populating and enriching the vocabularies for the dimensions Format, Method, Feature, and collecting, harmonising and formalising information about corpora and tools in the scope of CLS INFRA.
Information collected in this deliverable shall serve as input for population of the Transformation Matrix as part of the CLSCor catalogue and also serve as basis for the Transformation Toolbox, a consolidated set of scripts to convert data between formats to be delivered as D6.2 by the end of the project.

[83] https://vocabs.dariah.eu/tadirah/en/
[84] https://vocabs.sshopencloud.eu/browse/media-type/en/

8. References

8.1 Publications

Börner, I., Trilcke, P. (2023). CLS INFRA D7.1 On Programmable Corpora (v1.0.0). Zenodo. https://doi.org/10.5281/zenodo.7664964

Buchholz, S., Marsi, E. (2006). CoNLL-X Shared Task on Multilingual Dependency Parsing. In Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X). pp. 149-164, New York City. Association for Computational Linguistics. https://aclanthology.org/W06-2920

Cinková, S., Birkholz, J. M., Börner, I., Dejaeghere, T., Heiden, S., Janssen, M., Křen, M., Pozo, A. P. (2023). CLS INFRA D8.1 Report of the tools for the basic Natural Language Processing (NLP) tasks in the CLS context. Zenodo. https://doi.org/10.5281/zenodo.7951060

de Marneffe, M. - C., MacCartney, B., Manning, C. D. (2006). Generating Typed Dependency Parses from Phrase Structure Parses. In: Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06), Genoa, Italy. European Language Resources Association (ELRA). Online version: http://www.lrec-conf.org/proceedings/lrec2006/pdf/440_pdf.pdf

de Marneffe, M. - C., Manning, C. D. (2008). The Stanford Typed Dependencies Representation. In: Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation, pages 1–8, Manchester, UK. Coling 2008 Organizing Committee. Online version: https://aclanthology.org/W08-1301.pdf

de Marneffe, M. - C., Dozat, T., Silveira, N., Haverinen, K., Ginter, F., Nivre, J., Manning, C. D. (2014). Universal Stanford dependencies: A cross-linguistic typology. In Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), pages 4585–4592, Reykjavik, Iceland. European Language Resources Association (ELRA). Online version: http://www.lrec-conf.org/proceedings/lrec2014/pdf/1062_Paper.pdf

Ďurčo, M., Charvat, V. M., Börner, I., Mrugalski, M., Odebrecht, C. (2022). CLS INFRA D6.1 Inventory of existing data sources and formats. Zenodo. https://doi.org/10.5281/zenodo.7520287

González-Blanco Garciá, E., Ros, S., Khalil, O., Pozo, A.P., de la Rosa, J., de Sisto, M., Hernández, L., del Mar Jiménez, E., Corchoc, O., Rodríguez, J.L. (2022 – rejected) Applying Ontology Engineering to build a Poetry Domain Ontology. In: Semantic Web Journal https://www.semantic-web-journal.net/content/applying-ontology-engineering-build-poetry-domain-ontology

Hinkelmanns, P. (2021). Textformate: GrAF. In: KONDE Weißbuch. Hrsg. v. Helmut W. Klug unter Mitarbeit von Selina Galka und Elisabeth Steiner im HRSM Projekt "Kompetenznetzwerk Digitale Edition". Handle: hdl.handle.net/11471/562.50.185. PID: o:konde.185. Online version: https://www.digitale-edition.at/o:konde.185

Hinkelmanns, P. (2021). Alternativen zur Textkodierung mit TEI. In: KONDE Weißbuch. Hrsg. v. Helmut W. Klug unter Mitarbeit von Selina Galka und Elisabeth Steiner im HRSM Projekt "Kompetenznetzwerk Digitale Edition". Handle: hdl.handle.net/11471/562.50.15. PID: o:konde.15 Online version: https://www.digitale-edition.at/o:konde.15

Hinrichs, E., Ide, N., Pustejovsky, J., Hajič, J., Hinrichs, M., Elahi, M. F., Suderman, K., Verhagen, M., Rim, K., Straňák, P., Mišutka, J. (2018). Bridging the LAPPS Grid and CLARIN. In: Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), Miyazaki, Japan. European Language Resources Association (ELRA). https://aclanthology.org/L18-1206

Ide, N., Romary, L., de la Clergerie, E. (2003). International Standard for a Linguistic Annotation Framework. In: Proceedings of the HLT-NAACL 2003 Workshop on Software Engineering and Architecture of Language Technology Systems (SEALTS), pp. 25–30. https://aclanthology.org/W03-0804

Ide, N., Suderman, K. (2007) GrAF: A Graph-based Format for Linguistic Annotations. In: Proceedings of the Linguistic Annotation Workshop, Association for Computational Linguistics, pp 1–8. Online version: https://www.cs.vassar.edu/~ide/papers/LAW.pdf

Ide, N., Baker, C., Fellbaum, C., Passonneau, R. (2010a). The manually annotated sub-corpus: A community resource for and by the people. In: Proceedings of the ACL 2010 conference short papers (pp. 68–73). Uppsala: Association for Computational Linguistics.

Ide, N., Suderman, K. (2014). The Linguistic Annotation Framework: A Standard for Annotation Interchange and Merging. In: Language Resources and Evaluation. Vol. 48. Issue 3. pp. 395-418. Online version: https://doi.org/10.1007/s10579-014-9268-1

Ide, N., et al. (2017). Community Standards for Linguistically-Annotated Resources. In: Ide, N., Pustejovsky, J. (eds) Handbook of Linguistic Annotation. Springer, Dordrecht. pp.113-165. https://doi.org/10.1007/978-94-024-0881-2_4

Ide, N., Chiarcos, C., Stede, M. and Cassidy, S. (2017a). Designing annotation schemes: From model to representation. In: Ide, N., Pustejovsky, J. (eds) Handbook of Linguistic Annotation. Springer, Dordrecht. pp.73-111. https://doi.org/10.1007/978-94-024-0881-2_3

Lapponi, E., Velldal, E., Oepen, S., Knudsen, R. L. (2014). Off-Road LAF: Encoding and Processing Annotations in NLP Workflows. In: Proceedings of the 9th edition of the Language Resources and Evaluation Conference (LREC) Reykjavik, Iceland, 2014. Online version: http://www.lrec-conf.org/proceedings/lrec2014/pdf/978_Paper.pdf

Petrov, S., Das, D., McDonald, R. (2012). A universal part-of-speech tagset. In: Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12), pp. 2089–2096, Istanbul, Turkey. European Language Resources Association (ELRA). Online version: http://www.lrec-conf.org/proceedings/lrec2012/pdf/274_Paper.pdf

Rim, K., Lynch, K., Verhagen, M., Ide, N., Pustejovsky, J. (2020). Interchange Formats for Visualization: LIF and MMIF. In: Proceedings of the Twelfth Language Resources and Evaluation Conference, pp. 7230-7237, Marseille, France. European Language Resources Association. https://aclanthology.org/2020.lrec-1.893

Sanderson, R., Ciccarese, P., Young, B.. W3C. W3C Recommendation. URL: http://www.w3.org/TR/annotation-model/

Schöch, Ch., Fileva, E., Dudar, J. (2022). CLS INFRA D3.1 Baseline Methodological User Needs Analysis. Zenodo. https://doi.org/10.5281/zenodo.6389333

Stenetorp, P., Pyysalo, S., Topić, G., Ohta, T., Ananiadou S., Tsujii, J. (2012). brat: a Web-based Tool for NLP-Assisted Text Annotation. In: Proceedings of the Demonstrations Session at EACL 2012

van Gompel, M., Reynaert, M. (2014). FoLiA: A practical XML format for linguistic annotation - a descriptive and comparative study. In: Computational Linguistics in the Netherlands Journal. 3:63-81. 2013. Online version: https://clinjournal.org/clinj/article/view/26/22

van Gompel, M., van der Sloot, K., Reynaert, M., van den Bosch, A. (2017). FoLiA in Practice: The Infrastructure of a Linguistic Annotation Format. In: CLARIN in the Low Countries. Eds: Jan Odijk and Arjan van Hessen. pp. 71-81. Online version: https://www.jstor.org/stable/j.ctv3t5qjk.13?seq=1#metadata_info_tab_contents

van Gompel, M. (2019). FoLiA: Format for Linguistic Annotation – Documentation. Release v2.0 (rev 9.0). In: Documentation Language and Speech Technology Technical Report Series. Radboud University Nijmegen. Online version: https://github.com/proycon/folia/raw/master/docs/folia.pdf

Verhagen, M., Suderman, K., Wang, D., Ide, N., Shi, C., Wright, J., Pustejovsky, J. (2016). The LAPPS Interchange Format. pp. 33-47. DOI: 10.1007/978-3-319-31468-6_3

Zeman, D., et al. (2023). Universal Dependencies 2.12, LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University, http://hdl.handle.net/11234/1-5150

Zinn, C., Dima, E. (2022). The CLARIN Language Resource Switchboard. In: D. Fišer & A. Witt (Ed.), CLARIN: The Infrastructure for Language Resources (pp. 83-106). Berlin, Boston: De Gruyter. https://doi.org/10.1515/9783110767377-004

8.2 Collection of Links and Websites used for D6.3

(last accessed in October 2023)

- ALTO
  - https://loc.gov/standards/alto/
  - https://altoxml.github.io/
  - https://github.com/altoxml

    - https://www.loc.gov/standards/alto/description.html
    - https://www.loc.gov/standards/alto/news.html#4-4-released
    - https://www.loc.gov/standards/alto/v4/alto-4-4.xsd
- BRAT
    - https://brat.nlplab.org/manual.html
    - https://brat.nlplab.org/features.html
    - https://brat.nlplab.org/about.html
    - https://github.com/nlplab/brat
    - https://brat.nlplab.org/standoff.html
    - https://huggingface.co/DFKI-SLT
    - https://huggingface.co/datasets/DFKI-SLT/brat
    - https://aphp.github.io/edsnlp/latest/utilities/connectors/brat/
    - https://github.com/TsujiiLaboratory/stav
- CoNLL
    - https://www.signll.org/conll/
    - https://paperswithcode.com/dataset/conll-2003
- FoLiA / FLAT
    - https://proycon.github.io/folia/
    - https://folia.readthedocs.io/
    - https://github.com/proycon/folia
    - https://github.com/proycon/folia/blob/master/schemas/folia.rng
    - https://pypi.org/project/FoLiA/
    - https://foliapy.readthedocs.io/en/latest/
    - https://github.com/proycon/foliatools
    - https://github.com/LanguageMachines/foliautils
    - https://github.com/LanguageMachines/libfolia
    - https://crates.io/crates/folia
    - https://folia.readthedocs.io/en/latest/fql.html
    - https://github.com/proycon/flat
    - https://flat.readthedocs.io/
    - https://github.com/proycon/flat#flat---folia-linguistic-annotation-tool
    - https://github.com/proycon/flat#features
    - https://languagemachines.github.io/ucto/
    - https://languagemachines.github.io/frog/
- Hugging Face (datasets and models repository)
    - https://huggingface.co/
- JINJA
    - https://jinja.palletsprojects.com/en/3.1.x/
- JSON
    - https://en.wikipedia.org/wiki/JSON
- LAF
    - https://www.iso.org/standard/37326.html

    - https://www.digitale-edition.at/o:konde.187
- LAPPS and LIF
    - https://www.lappsgrid.org/
    - https://wiki.lappsgrid.org/
    - http://vocab.lappsgrid.org/
    - http://vocab.lappsgrid.org/schema/lif-schema.json
    - https://wiki.lappsgrid.org/interchange/
    - https://github.com/lapps
- Licenses
    - GNU Public License v3
- Linguistic LOD
    - https://linguistic-lod.org/
- MARC
    - https://www.loc.gov/marc/ndmso.html
- METS
    - https://www.loc.gov/standards/mets
    - https://www.loc.gov/standards/mets/version1121/mets.xsd
    - https://en.wikipedia.org/wiki/Metadata_Encoding_and_Transmission_Standard
- NIF
    - https://persistence.uni-leipzig.org/nlp2rdf/
    - https://nif.readthedocs.io/en/latest/
    - https://github.com/NLP2RDF
    - https://github.com/NLP2RDF/ontologies/tree/master
- ONNX
    - https://onnx.ai/
- PENN Treebank
    - https://www.ling.upenn.edu/~janabeck/tutorial.html
- PANDOC
    - https://pandoc.org/
- POSTDATA and Averell
    - https://postdata.linhd.uned.es/results/ontologies/domain-model/
    - https://postdata.linhd.uned.es/averell-new-poetrylab-tool-by-postdata/
    - https://pypi.org/project/averell/
    - https://zenodo.org/record/5702404
- SSHOC Conversion Hub
    - https://conversion-hub.sshopencloud.eu/
- SpaCy / DocBin
    - https://spacy.io/api/docbin
    - https://github.com/explosion/projects/tree/v3
    - https://github.com/explosion/projects/blob/v3/tutorials/ner_food_ingredients/assets/food_data.jsonl
- TEI / TEIGarage

    - https://tei-c.org
    - https://teigarage.tei-c.org/
- TEITOK
    - http://www.teitok.org/
    - http://teitok.corpuswiki.org/
    - https://lindat.mff.cuni.cz/services/udpipe/info.php
    - https://lindat.mff.cuni.cz/services/teitok/ud212/
    - https://lindat.mff.cuni.cz/services/teitok/
- TCF / Weblicht
    - https://weblicht.sfs.uni-tuebingen.de/weblichtwiki/index.php/The_TCF_Format
    - https://weblicht.sfs.uni-tuebingen.de/weblichtwiki/index.php/Main_Page
    - https://weblicht.sfs.uni-tuebingen.de/weblichtwiki/index.php/The_TCF_Format#Background_on_WebLicht_and_Motivation_for_TCF
    - https://weblicht.sfs.uni-tuebingen.de/weblicht/
    - https://www.digitale-edition.at/o:konde.212
- TSV
    - https://www.loc.gov/preservation/digital/formats/fdd/fdd000533.shtml
- TXM
    - https://txm.gitpages.huma-num.fr/textometrie/en/Presentation/
- UD
    - https://universaldependencies.org/format.html
    - https://universaldependencies.org/
    - https://universaldependencies.org/introduction.html
    - https://universaldependencies.org/guidelines.html
    - https://github.com/UniversalDependencies
    - https://lindat.mff.cuni.cz/services/udpipe/info.php
    - https://lindat.mff.cuni.cz/services/teitok/ud212/
    - https://github.com/UniversalDependencies/tools
    - http://hdl.handle.net/11234/1-5150
- UIMA
    - https://uima.apache.org/d/uimaj-current/tutorials_and_users_guides.html
    - https://uima.apache.org/
- Vocabularies
    - https://vocabs.dariah.eu/
    - https://vocabs.dariah.eu/tadirah/en/
    - https://vocabs.sshopencloud.eu/browse/media-type/en/
- Web Annotation Model (W3C)
    - https://www.w3.org/
    - https://www.w3.org/TR/annotation-model/
    - https://www.w3.org/TR/annotation-vocab/
    - https://www.w3.org/TR/annotation-vocab/#bib-annotation-model

- https://www.w3.org/TR/annotation-protocol/

9. Appendix

9.1 List of Prefixes / Namespaces

prefix	namespace / URI-schema	description
clscore	<https://core.clscor.io/entity/{entity-type}/{hash-id}>	Dedicated CLSCor namespace for general, i.e., not corpus-specific instances
clst	<https://core.clscor.io/entity/type/{vocab-key}/{term-key}>	Dedicated CLSCor namespace for defining types
corpus	<https://{corpus-key}.clscor.io/corpus> Example: <https://textgrid.clscor.io/entity/corpus>	Dedicated CLSCor namespaces for instances from individual corpora. These will be grouped in a corresponding named graph.
crm	<http://www.cidoc-crm.org/cidoc-crm/>	Basic CIDOC CRM ontology
crmcls	<https://clscor.io/ontologies/CRMcls/>	Main CLSCor ontology, extension of CIDOC CRM
crmdig	<https://cidoc-crm.org/crmdig/>	CIDOC CRM compatible ontology to encode methods of production of digital objects
lrm	<http://www.cidoc-crm.org/lrmoo/>	LRMoo (formerly FRBRoo); CIDOC CRM compatible ontology
media-type	<https://vocabs.sshopencloud.eu/vocabularies/media-type/>	IANA Media Types vocabulary
owl	<http://www.w3.org/2002/07/owl#	Web Ontology Language
pem	<http://parthenos.d4science.org/CRMext/CRMpe.rdfs>	PARTHENOS Entities Model
rdf	<http://www.w3.org/1999/02/22-rdf-syntax-ns#>	Resource Description Framework
rdfs	<http://www.w3.org/2000/01/rdf-schema#>	Resource Description Framework Schema
skos	<http://www.w3.org/2004/02/skos/core#>	Simple Knowledge Organisation System

9.2 Vocabularies

9.2.1 Vocabulary for Features

term-key	skos:prefLabel
pos_tag	part of speech tag

named_entity	named entity
token	token
lemma	lemma
stemma	stemma
morphological_category	morphological category
relation	relation
linked_entity	linked entity
sentence	sentence
topic	topic
syntactic_relation	syntactic relation between tokens within one sentence
coreference	coreference
sentiment_value	sentiment value
text_snippet	text snippet matching a query
syntactic_tree	syntactic tree matching a query
collocation_profile	collocation profile
manually_annotated_feature	manually annotated feature
network_graph	network graph with characters as nodes and interactions as edges
event	event
stanza_type	stanza type
enjambment	enjambment
morphological_segment	morphological segment
number_of_words	number of words
number_of_sentences	number of sentences
excluded_stop_words	excluded stop words
label_for_semantic_role	label for semantic role
segmented_topic	segmented topic
syllable	syllable
word	word
document	document
rhyme_scheme	rhyme scheme

metrical_scheme	metrical scheme
document_classification	document classification
document_cluster	document cluster; distance/similarity matrix
linguistically_annotated_feature	linguistically annotated feature
phrase	phrase
word_meaning	word meaning
word_ending	word ending
spelling	spelling
conjugation_form	conjugation form
name_of_character	name of character
paragraph	paragraph
segment	segment
act	act
scene	scene
co_occurrence	co-occurrence
variant	variant
word_form	word form

9.2.2 Vocabulary for Formats

Prefix: media-type = <https://vocabs.sshopencloud.eu/vocabularies/media-type/>

term-key	skos:broaderMatch	skos:exactMatch
alto_xml	media-type:textslashxml	
annis	media-type:textslashxml	
conll	media-type:textslashtab-separated-values	
conll-u	media-type:textslashtab-separated-values	
conll2000	media-type:textslashtab-separated-values	
conll2002	media-type:textslashtab-separated-values	
conll2003	media-type:textslashtab-separated-values	
conll2006	media-type:textslashtab-separated-values	

conll2009	media-type:textslashtab-separated-values	
conll2012	media-type:textslashtab-separated-values	
conll_corenlp	media-type:textslashtab-separated-values	
ims_cwb_vrt	media-type:textslashtab-separated-values	
cora_xml	media-type:textslashxml	
csv		media-type:textslashcsv
doc	media-type:applicationslashmsword	
docx	media-type:applicationslashmsword	
elan		
epub	media-type:applicationslashepubpluszip	
html		media-type:textslashhtml
horizontal	media-type:plain	
jpeg		media-type:jpeg
json		media-type:applicationslashjson
latex	media-type:textslashvnd.latex-z	
marc21	media-type:applicationslashmarc	
mobi		media-type:applicationslashvnd.amazon.mobi8-ebook
odt		
pdf		media-type:applicationslashpdf
rtf		media-type:textslashrtf
sgml		media-type:textslashsgml
tei_p4	media-type:applicationslashteiplusxml	
tei_p5	media-type:applicationslashteiplusxml	
tei		media-type:applicationslashteiplusxml
tsv		media-type:textslashtab-separated-values
ttl		media-type:textslashturtle
txt		media-type:plain
vertical	media-type:plain	

xml		media-type:textslashxml
xml_rdf		media-type:applicationslashrdfplusxml
xsl_fo	media-type:textslashxml	
zip		media-type:applicationslashzip
nif		
perseus_2.1	media-type:textslashxml	
tcf		
gexf	media-type:textslashxml	
gdf	media-type:textslashcsv	
gml		
graph_ml	media-type:textslashxml	
laf		
graf		
tei_txm	media-type:applicationslashteiplusxml	

9.2.3 Vocabulary for Methods

term-key	skos:prefLabel	skos:exactMatch	skos:broaderMatch
annotation_of_stanza_type	annotation of stanza type		
classification	classification		
clustering	clustering		https://vocabs.dariah.eu/tadirah/clusterAnalysis
collocations_analysis	collocations analysis	https://vocabs.dariah.eu/tadirah/collocationAnalysis	
coreference_resolution	coreference resolution		
entity_linking	entity linking		
event_extraction	event extraction		https://vocabs.dariah.eu/tadirah/extracting
filtering_stop_words	filtering stop words		
lemmatization	lemmatization	https://vocabs.dariah.eu/tadirah/lemmatizing	
linear_text_search	linear text search		

manual_annotation	manual annotation		https://vocabs.dariah.eu/tadirah/annotating
morphological_segmentation	morphological segmentation		https://vocabs.dariah.eu/tadirah/segmenting
morphological_tagging	morphological tagging		https://vocabs.dariah.eu/tadirah/tagging
named_entity_recognition	named entity recognition	https://vocabs.dariah.eu/tadirah/namedEntityRecognition	
network_analysis	network analysis	https://vocabs.dariah.eu/tadirah/networkAnalysis	
part_of_speech_tagging	part of speech tagging	https://vocabs.dariah.eu/tadirah/posTagging	
relation_extraction	relation extraction		https://vocabs.dariah.eu/tadirah/extracting
rule_based_enjambment_detection	rule-based enjambment detection		
semantic_parsing	semantic parsing		
semantic_role_labelling	semantic role labelling		
sentence_counting	sentence counting		
sentence_splitting	sentence splitting		
sentiment_analysis	sentiment analysis	https://vocabs.dariah.eu/tadirah/sentimentAnalysis	
stemming	stemming		
syllabification	syllabification		
syntactic_parsing	syntactic parsing		https://vocabs.dariah.eu/tadirah/parsing
syntactic_tree_search	syntactic tree search		
tokenization	tokenization		
topic_modelling	topic modelling	https://vocabs.dariah.eu/tadirah/topicModeling	
topic_segmentation	topic segmentation		https://vocabs.dariah.eu/tadirah/segmenting
word_counting	word counting		