Corpus Query Lingua Franca part II: Ontology

Stefan Evert[1], Oleg Harlamov[1], Philipp Heinrich[1], Piotr Bański[2]
[1]Lehrstuhl für Korpus- und Computerlinguistik
Friedrich-Alexander-Universität Erlangen-Nürnberg
[2]Leibniz-Institut für Deutsche Sprache, Mannheim
{stefan.evert,oleg.harlamov,philipp.heinrich}@fau.de
banski@ids-mannheim.de

Abstract

The present paper outlines the projected second part of the Corpus Query Lingua Franca (CQLF) family of standards: CQLF Ontology, which is currently in the process of standardization at the International Standards Organization (ISO), in its Technical Committee 37, Subcommittee 4 (TC37SC4) and its national mirrors. The first part of the family, ISO 24623-1 (henceforth CQLF Metamodel), was successfully adopted as an international standard at the beginning of 2018. The present paper reflects the state of the CQLF Ontology at the moment of submission for the Committee Draft ballot. We provide a brief overview of the CQLF Metamodel, present the assumptions and aims of the CQLF Ontology, its basic structure, and its potential extended applications. The full ontology is expected to emerge from a community process, starting from an initial version created by the authors of the present paper.

Keywords: CQLF, query languages, standardization, ISO, corpus

1. Introduction

In the past 30 years, the "corpus revolution" has yielded a wide variety of language corpora designed to fulfill a myriad of research purposes, some of which were not even foreseen at the time of creation. In order to satisfy specific information needs of the current and future users of these corpora, corpus query languages (CQLs) have been and will be created. While, at first sight, CQLs vary to a nearly unlimited extent, commonalities may easily be identified, both from the point of view of the corpus data models that they target and from the point of view of information needs that users may have. The CQLF standard family aims at capturing the commonalities while at the same time defining and circumscribing the matrix of potential variation among CQLs.

The existence of a large number of different corpus query languages poses an epistemic challenge for the research community, since (1) different CQLs have to be learned in order to be able to address specific information needs, and (2) some language resources are in practice only accessible through corpus management systems that feature only a single CQL (cf. Bański et al., 2016). We are thus in need of better interoperability across corpus query systems, realized – in one approach – by abstracting away from individual CQLs to see how far their queries can be compared or even transferred between the various corpus analysis platforms.

The present paper describes the second part of the Corpus Query *Lingua Franca* (CQLF) family of standards. CQLF aims, inter alia, at facilitating the comparison of the properties of different corpus query languages. It is meant to provide a common reference taxonomy based on "a consistent, stable and highly expressive set of category labels" (Smith, 2004: 34) following conventionally accepted definitions. CQLF is currently in the process of standardization at the International Standards Organization (ISO), in its Technical Committee 37, Subcommittee 4 (TC37SC4), and its national mirrors, notably at the DIN (Deutsches Institut für Normung – German Institute for Standardization), and more precisely the DIN NA 105-00-06 AA "Arbeitsausschuss Sprachressourcen".

The first part of the CQLF family, ISO 24623-1 (henceforth CQLF Metamodel), was successfully adopted as an international standard at the beginning of 2018. The present paper reflects the state of the second part, CQLF Ontology (ISO CD 24623-2), at the moment of its submission for the Committee Draft ballot, which is a means of achieving consensus among the technical experts of TC37SC4 and which – if successful – will result in a Draft International Standard, circulated among all ISO and liaison members for comments and discussion.[1]

In the following sections, we provide a brief overview of the CQLF Metamodel, and then present the assumptions and aims of the CQLF Ontology, its basic structure, and the way for the ontology to grow in a moderated community process.

2. CQLF part I: Metamodel

The CQLF Metamodel provides a coarse-grained classification of CQLs, describing their scope at a general level with conformance conditions meant to be satisfied by a wide range of CQLs. It provides the "skeleton" of a CQL taxonomy by setting up basic categories of corpus queries (CQLF levels and modules) and the dependencies among them, cf. Figure 1.[2]

[1] The ISO standardization process is described at https://www.iso.org/stages-and-resources-for-standards-development.html

[2] For the context of the CQLF endeavour, see Mueller (2010), Frick et al. (2012), and Bański et al. (2016). An implementation of early CQLF ideas in the KorAP project is discussed by Bingel and Diewald (2015).

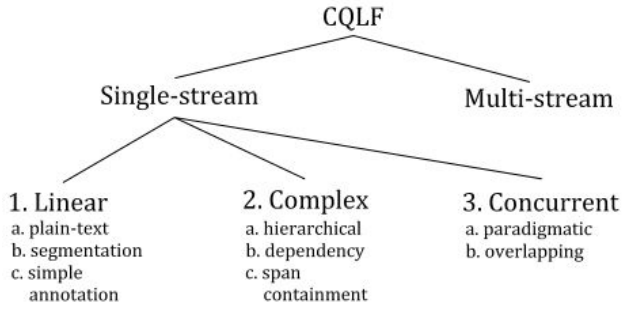


Figure 1. CQLF Metamodel and the taxonomy of levels (1-3) and Modules (a-c)

Conformance against the CQLF Metamodel is stated by providing the full path ending in a Module, e.g.

(1) `single stream / complex / dependency`

This is obviously very permissive and oriented mostly towards the data models assumed by the individual CQLs. Due to the coarse-grained nature of the CQLF Metamodel, the task of a more concrete characterization of CQLs falls to other parts of the CQLF standard family. CQLF part 2 (henceforth CQLF Ontology) described in the present paper is intended to provide a taxonomy of query functionalities as the basis for a more granular description and comparison of CQLs, including those not yet existing. It focuses on the expressive power of CQLs, i.e. the generalised information needs that can be satisfied by corpus queries. Only the top part of the ontology – describing individual aspects of query functionality at an abstract level – will be standardized. For the lower layers, which typically involve more or less complex combinations of multiple functionalities, a community process is envisioned in order to build and extend the ontology in step with the evolution of CQLs and user requirements.[3]

The third envisioned member of the CQLF standard family is a standard codifying corpus query architectures involving multiple data streams (e.g. aligned text corpora or multimodal corpora), while part 4 is projected as focusing on the format of query results.

3. CQLF Ontology: assumptions and aims

CQLs differ widely in their basic sets of capabilities. Whereas some are restricted to rather specific application scenarios, others are able to cover a wider variety of applications and search needs. It is therefore both the quality and the quantity of CQL capabilities – as well as the degree of their combination – that determine the expressive power of a CQL. The CQLF Ontology is not intended to articulate all the possible combinations of capabilities unless these are justified by genuine usage. Its aim is to provide representative categories for typical search needs within a taxonomy of CQL capabilities. This is achieved mainly through the Expressive Power taxonomy (described in Section 4), complemented by the CQLF Metamodel taxonomy (formalizing the first part of the standard) and a taxonomy of CQLs. Links between the CQL taxonomy and the Expressive Power taxonomy describe the search needs satisfied by a given CQL (and thus its degree of conformance to the CQLF Ontology), while at the same time providing end users with examples of the required query syntax (cf. Section 5).

The central goals of the CQLF Ontology are

- to allow CQL developers to describe the expressive power of their CQL as precisely as possible with respect to a well-defined taxonomy, as well as
- to enable end users to identify a CQL that supports their search needs.

In an active community of users and developers, the ontology will have further applications, e.g.

- to help end users learn a new CQL via a "cookbook" organized by common search needs rather than the formal design of CQL syntax;
- to support end users and front-end developers in translating between different CQLs via coordinated examples;
- to provide a common platform for end users and CQL developers, where the end users formulate requirements in terms of search needs, and thus guide CQL developers in the process of selecting and prioritizing new features.

4. CQLF Ontology: general structure

The key structural task of the CQLF Ontology is to provide a frame of reference for describing the expressive power of CQLs. To this end, it collects a taxonomy of typical search needs and organizes them in the concept hierarchy of an OWL DL ontology (Hitzler et al., 2012). Such an ontology consists of a **T-Box** (terminology), which defines concepts and their hierarchical relations, and an **A-Box** (assertions), which contains knowledge about individuals and the relations between them.[4]

The T-Box of the CQLF Ontology consists of three separate taxonomies. At its centre lies a taxonomy describing the **Expressive Power** of CQLs in terms of general capabilities and their combinations as well as more specific search needs at varying levels of concreteness. It is organised into three hierarchical layers, which contain from the top down:

- **Functionalities** that correspond to individual capabilities of CQLs at a general level. They serve as entry points for ontology navigation and are linked to the CQLF Metamodel.
- **Frames** that represent typical search needs of users at a relatively abstract level. Most Frames involve combinations of multiple Functionalities. They are

[3] In this respect, CQLF Ontology is intended to be similar to ISO 12620:2019, which also describes the structure of data category registry, while opening its content to a community process.

[4] The CQLF Ontology only encodes information about the concept membership of individuals, not about their relations.

intended as the central layer for the characterisation and comparison of CQLs.

- **Use Cases** as concrete instantiations of Frames in a specific scenario, for which the conformance of a given CQL can be determined unambiguously and supported by an explicit query expression. Most Use Cases will be parameterised, i.e. involve one or more variable elements that are marked explicitly (see Section 5).

The second taxonomy is a formalisation of the CQLF Metamodel shown in Figure 1; its concepts represent CQLF levels and modules. The third taxonomy consists of a flat list of concepts representing individual CQLs.

The individuals in the A-Box of the CQLF Ontology are **positive conformance statements** between CQLs and Use Cases. They take the form of parameterised query expressions in the respective CQL, providing solid evidence for each conformance claim. Positive conformance statements to higher layers of the taxonomy are not allowed because they are rarely unambiguous. In many cases, a CQL will support some Use Cases of a Frame, but not others; or it will implement certain Functionalities, but not in all possible combinations.

A **negative conformance statement** documents a missing capability or design limitation of a CQL. It stipulates that the CQL does not provide a particular Functionality or cannot satisfy the search need of a particular Frame. As a consequence, there can be no positive conformance statements for any Use Cases that instantiate the Frame or involve the Functionality, respectively. Negative conformance statements are part of the T-Box because they address concepts rather than individuals.

The general structure of the CQLF Ontology is illustrated in Figure 2. A detailed example of a fragment of the ontology is given in Section 6.

5. CQLF Ontology: formalisation

The means for constructing the taxonomic framework of the CQLF Ontology are provided by the Web Ontology Language (Hitzler et al., 2012). OWL furnishes developers with a set of tools for (i) stating concept hierarchies and membership of individuals and (ii) defining highly expressive property restrictions. The underlying theoretical foundation is laid by the family of description logics (DL, see Krötzsch et al., 2012). More specifically, the CQLF Ontology relies on OWL DL – an OWL dialect optimising the trade-off between expressiveness and decidability that largely agrees with the $\mathcal{SROIN}$(D) description logic in its modeling capabilities.

The OWL is designed as a mixture of knowledge representation technologies, i.e. it blends the strengths of description logics (which provide the formal framework) and the Resource Description Framework (with RDF/XML as the normative exchange format). In particular, the CQLF Ontology makes use of the AnnotationProperty construct of OWL DL in order to associate additional information with concepts and individuals.

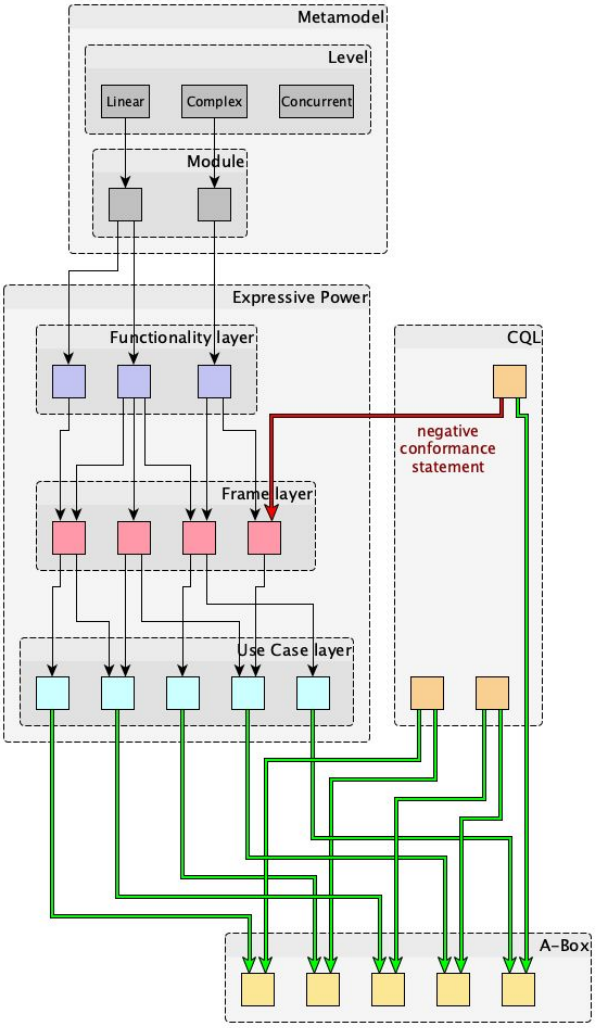


Figure 2. General structure of the CQLF Ontology. Positive conformance statements are indicated by green arrows, negative conformance statements by red arrows.

In the T-Box of the terminology, concepts are organised into a hierarchical taxonomy by subsumption relations of the form $A \sqsubseteq B$ ("A is subsumed by B"), which represent "is-a" relationships. The most important consequence is that if an individual x is a member of A, it must also be a member of B ($x \in A \rightarrow x \in B$). Other important operators are $A \sqcap B$ (the intersection of the concepts), $A \sqcup B$ (their union or disjunction) and $A \equiv B$ (concept equivalence). The "top" concept $\top$ contains all individuals, while the "bottom" concept $\bot$ represents an empty set. The A-Box of the terminology makes concept assertions of the form $x \in A$, indicating that x is a member of the concept A.

The top layer of the Expressive Power taxonomy consists of **Functionality** concepts. Each Functionality F is

subsumed by the abstract root concept of the layer (named Functionality) and by one or more concepts M_i representing suitable CQLF modules (in the Metamodel taxonomy). In mathematical notation:

$$F \sqsubseteq \text{Functionality} \sqcap M_1 \sqcap M_2 \sqcap \ldots$$

The abstract root concept enables us to stipulate that the Functionality layer must cover the entire universe of individuals, i.e. every query expression has to involve some Functionality F:

$$\text{Functionality} \equiv \top$$

(and similarly for the other layers). The current draft of the CQLF Ontology lists 17 Functionalities, including the examples listed below (with links to CQLF modules).

- Annotation $\sqsubseteq$ SimpleAnnotation $\sqcup$ Paradigmatic[5]
- PartialMatch $\sqsubseteq$ PlainText $\sqcup$ SimpleAnnotation
- TreeRelation $\sqsubseteq$ Hierarchical

The middle layer of the taxonomy consists of **Frame** concepts, which represent typical search needs at a relatively high level of abstraction. Each Frame A is subsumed by one or (typically) more Functionalities that it combines into a complex search need:

$$A \sqsubseteq \text{Frame} \sqcap F_1 \sqcap F_2 \sqcap \ldots$$

Frames can also be specialisations of other frames A_i if they combine and/or extend their respective search needs:

$$A \sqsubseteq \text{Frame} \sqcap A_1 \sqcap A_2 \sqcap \ldots$$

In addition to a short descriptive label (*rdfs:label*), every Frame A must be annotated with a clear human-readable description of the search need (*cqlf:searchNeed*). As an example, consider the search need of matching some annotation value against a regular expression:

Frame A_1 $\sqsubseteq$ Annotation $\sqcap$ PartialMatch
label: RegEx(Annotation(Object))
searchNeed: find object whose annotation matches a given regular expression

The bottom layer consists of **Use Case** concepts as concrete instantiations of Frames. In most cases, a Use Case U instantiates a single Frame A, i.e.

$$U \sqsubseteq \text{UseCase} \sqcap A$$

The search need represented by a Use Case should be so concrete that an explicit query expression satisfying it can be formulated – the basis for a positive conformance statement. Any variable element of the search need must be represented by a parameter, using the Unicode symbols ① (U+2460) … ⑳ (U+2473). As an example, consider Use Case U_1 as an instantiation of A_1 above:

Use Case U_1 $\sqsubseteq A_1$
label: find lemma matching regexp ①
searchNeed: find a token whose lemma annotation matches the regular expression ①

The individuals of the CQLF domain represent **positive conformance statements** in the form of parameterised query expressions for specific Use Cases. Every individual x is a member of the CQL Q in which it is formulated and of the Use Case U that it satisfies:

$$x \in Q \sqcap U$$
$$x ::= \text{(some corpus query)}$$

Concepts in the CQLF Ontology thus have an extensional interpretation: the extension of a CQL Q consists of all the query expressions formulated in Q; the extension of a Use Case U consists of all query expressions that satisfy U; and the extension of a Frame A or Functionality F consists of all query expressions that satisfy a Use Case U subsumed by A or F.

Every positive conformance statement must be annotated with the parameterised query expression (*rdfs:label*) using the same parameters as the satisfied Use Case U; a description of admissible parameter values and required transformations (*cqlf:parameters*); and a fully realized example of the query (*cqlf:example*) that can be executed directly in an implementation of Q. As an example, consider the positive conformance statement for Use Case U_1 above in the CQP query language (Evert & Hardie, 2011):

Positive conformance statement x_{42} $\in$ CQP $\sqcap U_1$
label: `[lemma = "①"]`
parameters: ① is a PCRE[6] regular expression, which is automatically anchored at the start and end of the lemma; double quotation marks in ① must be escaped by reduplication (`"` → `""`)
example: `[lemma = "(over|under).*ise"]`

A **negative conformance statement** specifies that the extensions of a CQL Q and a Use Case U, Frame A, or Functionality F are disjoint, e.g.

$$Q \sqcap A \equiv \bot$$

As a consequence, there can be no positive conformance statement x that belongs to Q and A, nor to any Use Case $U \sqsubseteq A$ instantiating A. As an example, CQP does not have any capabilities for searching parse trees, which can be expressed by a negative conformance statement at the level of Functionalities:

$$\text{CQP} \sqcap \text{TreeRelation} \equiv \bot$$

6. Example: an ontology fragment

This section illustrates the connection between Functionalities, Frames, Use Cases and conformance statements with three concrete examples. A larger fragment of the CQLF Ontology is visualized in Figure 3.

Frame: A_6 $\sqsubseteq$ Frame $\sqcap A_3 \sqcap A_4 \sqcap A_5$
label: RegEx(Annotation(Domination))(Annotation(Object), Annotation(Object))

[5] Note the use of a disjunction ($\sqcup$) operator rather than intersection ($\sqcap$). This indicates that Annotation functionality is relevant both in the SimpleAnnotation module and in the Paradigmatic module, *not* just if both modules are combined.

[6] Perl Compatible Regular Expressions, see www.pcre.org

searchNeed: domination relation with functional annotation matched by regular expression between two tree nodes with specific annotation values (single attribute=value constraints)

Use Case: $U_7 \sqsubseteq$ UseCase $\sqcap A_6$
label: immediate dominance matching regex ① between phrase of category ② and token with POS tag ③
searchNeed: find a phrase node A of category ② in a syntactic parse tree and a token B with part-of-speech tag ③ such that A is the immediate parent of B and the dominance relation is annotated with a function matching regular expression ①

Positive conformance: $x_{10} \in$ ANNIS $\sqcap U_7$
label: `cat = "②" & pos="③" & #1 >[func=/①/] #2`
parameters:
- ① is a basic regular expression, which is automatically anchored at the beginning and end of the annotation value
- forward slashes in ① must be escaped by backslashes
- double quotes in ② and ③ must be escaped by backslashes

example: `cat = "NP" & pos="ADJA" & #1 >[func=/NK.*/] #2`
comment: The precise flavour of regular expression syntax is not specified in the ANNIS documentation and may be implementation-specific. It is assumed that all elements of POSIX.1 Basic Regular Expression syntax are supported.

Figure 3. A fragment of the extended CQLF Ontology, showing examples of Frames, Uses Cases and conformance statements.

Frame: $A_{11} \sqsubseteq$ Frame $\sqcap$ Annotation $\sqcap$ Containment
label: Containment(Annotation(Object), Context)
searchNeed: find object with specific annotation value (single attribute=value constraint) contained in particular context span

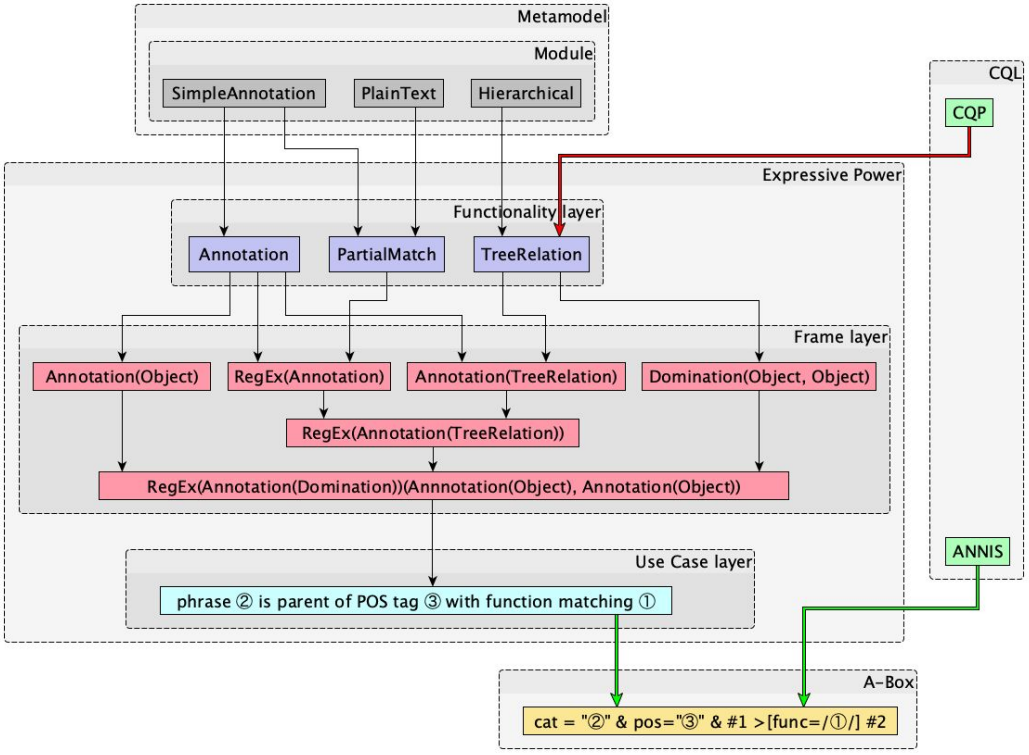
Use Case: $U_{13} \sqsubseteq$ UseCase $\sqcap A_{11}$
label: token with POS tag ① in span of category ②
searchNeed: find a token with part-of-speech tag ① that is contained in a span annotation of category ②

Positive conformance: $x_{32} \in$ CQP $\sqcap U_{13}$
label: `[pos="①"] within ②`
parameters:
- ① is an arbitrary string value; all PCRE metacharacters in ① must be escaped with a backslash (\); double quotation marks must be escaped by reduplication (" → "")
- ② is a simple identifier

example: `[pos="\$\."] within np`

Negative conformance: CQP $\sqcap$ TreeRelation $= \bot$

This statement indicates that CQP does not have the capability of searching for relations in tree structures. As a consequence, it cannot satisfy any Frame or Use Case subsumed by TreeRelation, such as A_6 or U_7.



7. The community process

Recall that only the structure of the CQLF Ontology and the contents of the top layer of the Expressive Power taxonomy are in the normative scope of standardisation. The lower layers – Frames and Use Cases – are expected to be supplied by the community of end users and query tool developers in a moderated process. The GitHub organisation `cqlf-ontology` has already been set up as a possible home for this community process. An informative annex to the standards document points to an initial version of the extended ontology, documenting the CQP (Evert & Hardie, 2011) and AnnisQL (Krause & Zeldes, 2016) query languages.[7]

Key features of the envisioned community process are (assuming that GitHub will be used as a platform):

- Authentication:
  - via GitHub, the ontology is only editable directly by users whose accounts are members of the `cqlf-ontology` organisation;
  - other users can submit pull requests, e.g. with conformance statements for a new (or newly added) CQL, or with entries for new Frames and Use Cases.
- Version control: all submissions are automatically recorded together with a date stamp and the user name of the submitter.
- Moderation: members of the GitHub organisation `cqlf-ontology` review pull requests and ensure that they meet all requirements; existing Frames and Use Cases will only be modified or deleted in exceptional circumstances by the moderators.
- It is expected that submitters ensure well-formedness of the ontology with their modifications before they initiate a pull request.
- Moderation, curation, error reporting and the verification of the submitted conformance statements will be driven by the ticketing system automatically coordinated with pull requests.

8. Outlook: envisaged usage scenarios

With the CQLF Ontology infrastructure in place, we envision the following potential usage scenarios:

An **end user** wants to find a CQL supporting their search needs, and therefore

- starts by specifying the relevant functionalities to narrow down the list of relevant Frames;
- navigates the ontology to locate the appropriate Frame meeting their search needs;
- finds CQLs that conform to Use Cases instantiating the Frame; and
- obtains parameterised query expressions that show directly how to satisfy the search need in the CQL.

A **developer** wants to document the expressive power of a new CQL, and therefore

- makes a positive conformance statement for each supported Use Case, giving an explicit query expression as evidence;
- makes negative conformance statements against Use Cases, Frames and Functionalities outside the scope of the CQL; and thus
- enables detailed qualitative and quantitative comparison with other CQLs.

The **community** of corpus researchers wants to guide CQL software development. The key factors relevant to this scenario are the following:

- the ontology resulting from the community process records the search needs deemed most important by end users; so that
- developers are able to discover and target Frames not supported by existing CQLs.

[7] The ontology is located at https://github.com/cqlf-ontology/.

Acknowledgements

We would like to thank Elena Frick for her work on a pilot project that yielded a very early version of the CQLF Ontology, and Andreas Witt, who conceived of the ISO initiative in 2011 and has supported it for years.

9. Bibliographical References

Bański, P., Frick, E., and Witt, A. (2016). Corpus Query Lingua Franca (CQLF). In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 2804–2809, Portorož, Slovenia. European Language Resources Association (ELRA).

Bingel, J. and Diewald, N. (2015). KoralQuery – a General Corpus Query Protocol. In *Proceedings of the Workshop on Innovative Corpus Query and Visualization Tools at NODALIDA 2015*, Vilnius, Lithuania.

Evert, Stefan and Hardie, Andrew (2011). Twenty-first century corpus workbench: Updating a query architecture for the new millennium. In *Proceedings of the Corpus Linguistics 2011 Conference*, Birmingham, UK.

Frick, E.; Schnober, C. and Bański, P. (2012). Evaluating Query Languages for a Corpus Processing System. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*. Istanbul, Turkey, May 2012. pp. 2905-2911 - European Language Resources Association (ELRA).

Hitzler, P.; Krötzsch, M.; Parsia, B.; Patel-Schneider, P., and Rudolph, S. (2012). OWL 2 Web Ontology Language Primer (Second Edition). W3C Recommendation, 11 December 2012. (Latest version available at http://www.w3.org/TR/owl2-primer/.)

ISO 24611:2012. Language resource management — Morpho-syntactic annotation framework (MAF). International Organization for Standardization, Geneva, CH.

ISO 24612:2012. Language resource management — Linguistic annotation framework (LAF). International Organization for Standardization, Geneva, CH.

ISO 24613:2008. Language resource management — Lexical markup framework (LMF) (currently under significant revision). International Organization for Standardization, Geneva, CH.

ISO 24615-1:2014. Language resource management — Syntactic annotation framework (SynAF) — Part 1: Syntactic model. International Organization for Standardization, Geneva, CH.

ISO 12620:2019. Management of terminology resources — Data category specifications. International Organization for Standardization, Geneva, CH., May 2019.

ISO 24623-1. Language resource management — corpus query lingua franca (CQLF) — part 1: Metamodel. Technical Report ISO 24623-1, International Organization for Standardization, Geneva, CH, April 2018.

Krause, T. and Zeldes, A. (2016). ANNIS3: A new architecture for generic corpus query and visualization. *Digital Scholarship in the Humanities* 31(1): 118–139.

Krötzsch, M., Simancik, F., and Horrocks, I. (2012). A description logic primer. arXiv:1201.4089

Mueller, M. (2010). Towards a digital carrel: A report about corpus query tools. Technical report. Unpublished report submitted to the Mellon Foundation. http://panini.northwestern.edu/mmueller/corpusquerytools.pdf

Smith, B. (2004). Ontology and information systems. http://ontology.buffalo.edu/ontology(PIC).pdf