KBSET – Knowledge-Based Support for Scholarly Editing and Text Processing with Declarative LaTeX Markup and a Core Written in *SWI-Prolog*

Jana Kittelmann[1] and Christoph Wernhard[2]

[1] Martin-Luther Universität Halle-Wittenberg, Germany
[2] Berlin, Germany

Abstract. *KBSET* is an environment that provides support for scholarly editing in two flavors: First, as a practical tool *KBSET/Letters* that accompanies the development of editions of correspondences (in particular from the 18th and 19th century), completely from source documents to PDF and HTML presentations. Second, as a prototypical tool *KBSET/NER* for experimentally investigating novel forms of working on editions that are centered around automated named entity recognition. *KBSET* can process declarative application-specific markup that is expressed in LaTeX notation and incorporate large external fact bases that are typically provided in RDF. *KBSET* includes specially developed LaTeX styles and a core system that is written in *SWI-Prolog*, which is used there in many roles, utilizing that it realizes the potential of Prolog as a unifying language.

1 Introduction

In the age of Digital Humanities, scholarly editing [11,12] involves the combination of natural language text with machine processable semantic knowledge, typically expressed as markup. The best developed machine support for scholarly editing is the XML-based TEI format [14], a comprehensive markup language for all sorts of text, mainly targeted at rendering for different media and extraction of metadata, which is achieved through semantics-oriented or declarative markup. Recent efforts stretch TEI by aspects that are orthogonal to its original *ordered hierarchy of content objects (OHCO)* text model, through support for entities like *names, dates, people, and places* as well as structuring with *linking, segmentation, and alignment* [14, Chap. 13 and 16]. Also ways to combine TEI with Semantic Web techniques, data modeling and ontologies are investigated [3]. In accord with these directions we observe a number of apparently open desiderata for the support of scholarly editing in today's practice and in future perspective, which we explicitly address with our environment *KBSET* (**K**nowledge-**B**ased **S**upport for Scholarly **E**diting and **T**ext Processing):

1. It should be possible for users from the application domain to *create, review, validate and maintain source documents* of the edition project. That

is, documents with annotated text, with metadata, and with data on relevant entities such as persons and locations. Text markup should be exposed to the users as far as it is relevant and interesting for the application field. Source documents must be stored and versioned. Since source texts with XML markup are hardly readable, in the TEI/XML approach typically an additional user-interface layer is added to the workflow, where apparently only a single – non-free – software system is suitable.[3] On the other hand, outside the Humanities, with LaTeX the direct use of text with markup is widespread, well supported by many free tools and supplemented by numerous free packages of high quality.[4]

2. It should be possible to generate *high-quality* print and hypertext presentations in a *reproducible* way, based on published source documents created in the edition project as well as additional documents and programs that are freely available and can be precisely identified.
3. Not just "final" presentations should be well-supported but also *internal tools* for developing the scholarly edition and *intermediate presentations* used there should be of high quality. This is in particular relevant as many edition projects take several years.
4. It should be possible to couple object text with associated information in ways that are *more flexible than in-place markup*: It may be convenient to maintain text annotations separately from the commented text sources. Markup can be by different authors, automatically generated, or for some specific purpose. Some queries and transformations should remain applicable also after changes of the markup.
5. It should be possible to incorporate advanced semantics related techniques that inherently deliver result that are *fuzzy*, *imprecise*, or *incomplete*. For example, named entity recognition or tools for statistics-based text analysis.
6. *Linking with external knowledge bases* should be supported. These include results of other edition projects as well as large fact bases such as authority files like *Gemeinsame Normdatei (GND)*,[5] metadata repositories like *Kalliope*,[6] domain specific bases like *GeoNames*, or aggregated bases like *YAGO* [5] and *DBpedia* [9].

[3] The *Oxygen XML Editor*. See also `https://en.wikipedia.org/wiki/Comparison_of_XML_editors`, accessed Nov 19 2019.

[4] In fact, [14, Sect. iv] notes that *"the TEI encoding scheme itself does not depend on this language* [XML]*; it was originally formulated in terms of SGML (the ISO Standard Generalized Markup Language), a predecessor of XML, and may in future years be re-expressed in other ways as the field of markup develops and matures."*

[5] `http://www.dnb.de/gnd`. The *GND* is maintained by the German-speaking library community and contains information about various entities, in particular about more than 11 million persons in more that 160 million fact triples. It is in the public domain (CC0) and can be downloaded as an RDF/XML document whose decompressed size is more than 18 GB.

[6] `http://kalliope-verbund.info`.

7. A digital edition project involves, more or less explicitly, the creation of data, in other words, the assertion of facts about relevant entities like persons, locations, dates, events and units of text such as, for example, letters as components of a correspondence, or distinguished positions in texts. Such data can be project-specific or obtained through combination with external fact bases. As a result of an edition project, such *data should be made explicit and accessible* in a way that facilitates to associate with them *machine processable semantics*, that is, meanings based on some logic that is supported by tools from automated reasoning and knowledge processing. Ontology reasoning in description logics is important here, but, by itself, not sufficient, as classification seems not a main operation of interest in the field. The *GND* fact base on persons, institutions and works, for example, gets by with a quite small ontology of 64 classes.

KBSET approaches these desiderata successfully through the involvement of two technologies: LaTeX and Prolog. More specifically, we defined a dedicated small set of descriptive markup elements that is tailored to the application domain, in our case the scholarly edition of correspondences of the 18th and 19th century, in the form of LaTeX commands and environments, and use *SWI-Prolog* [17] as a *single environment and language* to implement all tasks that involve parsing and composition of documents and fact bases in various formats, querying with respect to documents and fact bases, and evaluation of complex application constraints.

The current version of *KBSET* supports two flavors of application: The first, *KBSET/Letters*, is a practical environment for scholarly editions of correspondences. Implemented support covers in particular editions of correspondences from the 18th and 19th century in German language. The second, *KBSET/NER*, is a prototype system that allows to experiment with various advanced features centered around named entity recognition. *KBSET/Letters* is currently applied in a large project, the edition of the correspondence of philosopher and polymath Johann Georg Sulzer (1720–1779) with author, critic and poet Johann Jakob Bodmer (1698–1783), which will be published in print as [13, Vol. 10] in summer 2020. Including annotations and indexes, it spans about 2000 printed pages. The online HTML edition, also generated with *KBSET/Letters* from the same sources, will be published in parallel. In addition, *KBSET* is applied in a long-term project, *www.sulzer-digital.de*, a digital representation of Sulzer's complete correspondence, edited successively with *KBSET*. To illustrate the use of *KBSET/Letters*, the distribution of *KBSET* includes the edition of a small correspondence. For *KBSET/NER* it includes as an example a draft edition of a 19th century book. *KBSET* is available as free software from its home page

`http://cs.christophwernhard.com/kbset.`

The 2016 version of *KBSET/NER* was presented at DHd 2016 [7]. The Sulzer-Bodmer edition project and its use of *KBSET/Letters*, as well as related further interdisciplinary research topics, are described (in German) in [8]. Some com-

ponents of *KBSET/Letters* were derived from an earlier collaboration of the authors, *www.pueckler-digital.de* [6].

The rest of the paper is structured as follows: In Sect. 2 we describe *KBSET/Letters*, the environment for preparing scholarly editions of correspondences in practice, and in Sect. 3 the more experimentally oriented *KBSET/NER* flavor of *KBSET* centered around named entity recognition. We conclude the interdisciplinary paper in Sect. 4 with discussions of the *KBSET* environment from three different perspectives: Tools for scholarly editing, the role of *SWI-Prolog* as a unifying practical technology, and some encountered issues that might be of interest for future research on logic-based knowledge processing.

2 *KBSET/Letters*

The *KBSET/Letters* environment is at its current state of development adequate for scholarly editions of correspondences from the 18th and 19th century that are in German language and where the edited texts are represented in a character-preserving (*zeichengetreu*) but not position-preserving (*positionsgetreu*) way.

2.1 Descriptive Application-Specific Markup in LaTeX Notation

Figure 1 shows an overview on *KBSET*: Inputs, functionalities of the core system that is implemented in *SWI-Prolog*, and outputs. For creating a scholarly edition of a correspondence, the inputs are documents with domain specific markup expressed as LaTeX commands and environments, representing object texts of the edition project, that is, letters, and annotations by the editors that refer to the object texts, respectively (box I1 and I2 in Fig. 1). The parsimonious set of declarative markup elements *KBSET/Letters Markup*[7] is tailored to the requirements of such scholarly editions. Through the specialization, creating the markup is perceived by users as expressing statements of interest rather than a technical burden. Through the LaTeX notation, the marked-up text remains fairly readable and can be directly created by users with any text editor that supports LaTeX, such as, for example, *GNU Emacs*, which is free software and shown as representative tool in the figure.

Letters and annotations are represented by LaTeX environments. Here is an example of a letter environment:

```
\begin{letter}{bs:1745-02-14}{bodmer}{sulzer}{zuerich}{14. Februar 1745}
...
Der Hr.~\xperson{lange}{Pastor Lange von Laublingen}, hat mir, noch
 \xl{brief:lange}{ehe er den Brief von E~Hochedl. empfangen}, berichtet,
...
\end{letter}
```

Identifier `bs:1745-02-14` is declared to denote the represented letter. Arguments of the `\begin{letter}` statement provide essential meta data: Identifiers

[7] A specification draft is available from the KBSET home page.

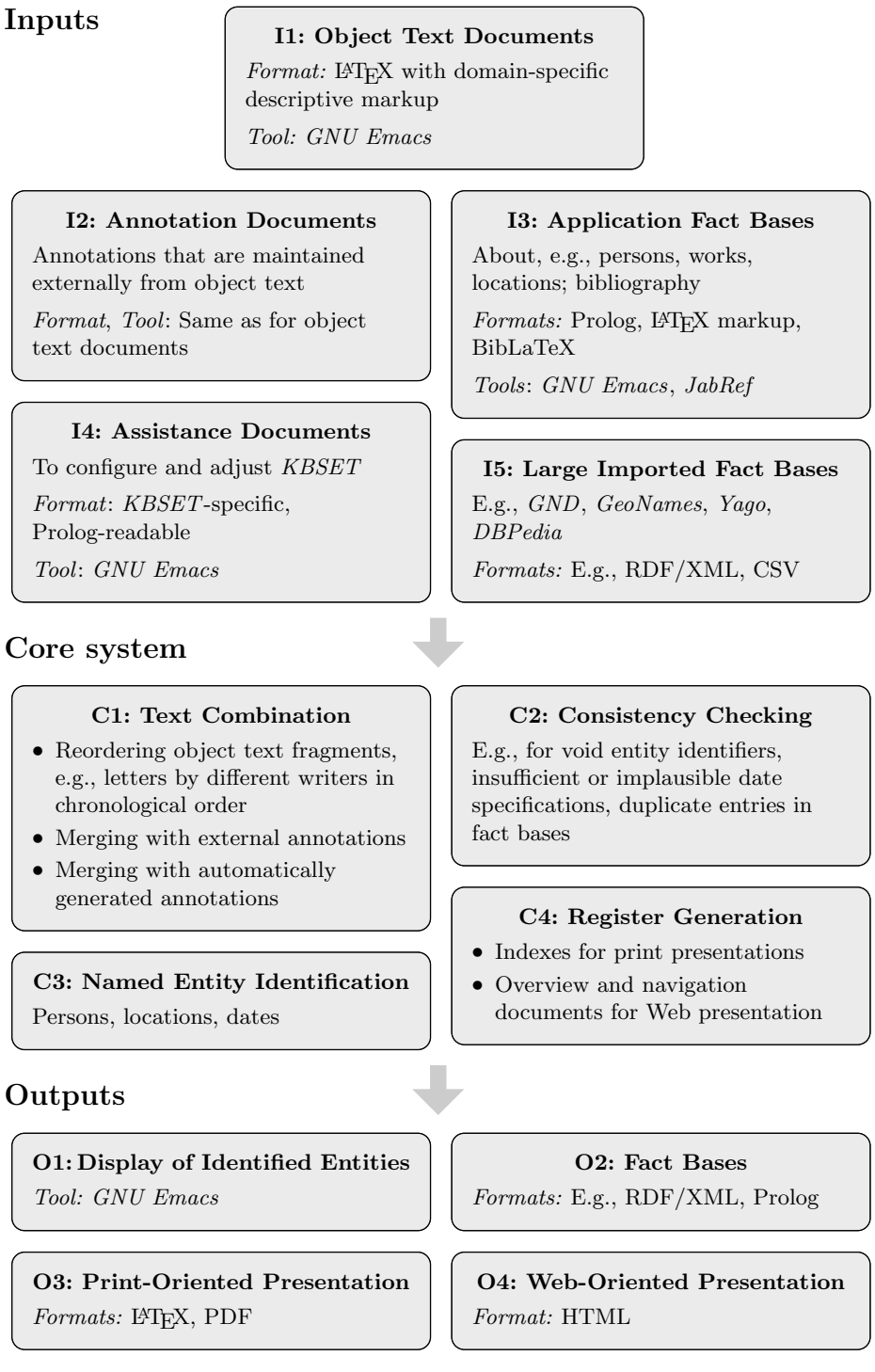


Fig. 1. *KBSET*: Overview on inputs, core system functionalities and outputs

of writer, addressee and location, as well as the date in a human readable but parsable form. The tilde for non-breaking space is transferred from LaTeX to the *KBSET/Letters* markup. The phrase *Pastor Lange von Laublingen* is marked-up as denoting the person with identifier `lange`. Identifiers used here can be mnemonic as they are local to the project. The identifier `brief:lange` is declared to denote the marked-up occurrence of the phrase *ehe er den Brief von E Hochedl. empfangen* in the letter. Its scope is the letter environment. The following example shows an annotation environment:

```
\begin{annotation}{bs:1745-02-14}
  ...
  \ksection{Stellenkommentar}
  \begin{klist}
  \kitem{brief:lange} Der Brief Bodmers an Samuel Gotthold Lange ...
  ...
  \end{klist}
\end{annotation}
```

The annotation block is about the example letter above, associated through the argument `bs:1745-02-14` of the `\begin{annotation}` statement. In the annotation environment the identifiers like `brief:lange` that were locally declared in the letter environment are re-activated for referencing. This permits a convenient way to express annotations that refer to specific places in the text of letters (*Stellenkommentare*).

Also fact bases can be written with special markup commands in LaTeX notation. For example, the referenced person `lange` can be declared with the following statement:

```
\defperson{lange}{Lange, Samuel Gotthold (1711--1781)}
```

Person names in these declarations must be compatible with the regularities used by the *GND*.[8] They can be directly used in indexes and, with years of birth and death, allow to automatically determine the global *GND* identifiers of persons represented in the *GND*. These global identifiers make metadata maintained, for example, in the *GND* and *Wikipedia* available, relieving the edition project from the need to replicate them explicitly.

So far, the user perceives the project as a collection of documents with letters, annotations and fact bases in the specialized descriptive LaTeX markup. Indeed, *KBSET* provides an implementation of the specialized markup in form of a LaTeX package that is sufficient to generate a PDF representation of the letters and annotations with fairly high quality just by a pure LaTeX workflow. In the result, letters and associated annotations are connected through PDF hyperlinks. References like `\xperson{lange}{...}` to identifiers declared in a fact base are converted to index entries processed by *xindy*. The bibliography is handled by *BibLaTeX*. The involved LaTeX processors already ensure validity and consistency of the documents to some degree.

[8] We do not demand in full the principles of the *GND* for choosing *preferred* names, as *"Colombo, Cristoforo"* or *"Homerus"* is unusual in German texts.

2.2 From LaTeX to Prolog for Further Consistency Checking and Text Combination

The *KBSET* core system includes a LaTeX parser written in Prolog that yields a list of items, terms whose argument is a sequence of characters represented as atom, and whose functor indicates a type such as *word*, *punctuation*, *comment*, *command*, or *begin* and *end of an environment*. A special type *opaque* is used to represent text fragments that are not further parsed, such as LaTeX preambles. LaTeX commands and environments can be made known to the parser to effect proper handling of their arguments. The parser aims to be practically useful, without claiming completeness for LaTeX in full. It does not permit, for example, a single-letter command argument without enclosing braces. The parser is supplemented by conversions of parsing results to LaTeX and to plain text.

So far, additional syntactic checks at parsing and various semantics-oriented checks that are applied after the parsed documents are converted to Prolog fact bases are implemented (box C2 in Fig. 1). Further ways of consistency checking can be realized with respect to the generated HTML documents discussed below in Sect. 2.3.

Source documents with letters and with annotations are maintained in a large edition project not necessarily in the same ordering and fragmentation in which these should appear in presentations. Based on the parsed LaTeX, the *KBSET* core system can perform such rearrangements (box C1 in Fig. 1) and write out generated LaTeX documents. The conventional LaTeX workflow applied to these generated documents then results in high-quality PDF documents, which, depending on the configuration, are suitable for publication in print or on-screen reading.[9] Figure 2 shows example output pages.

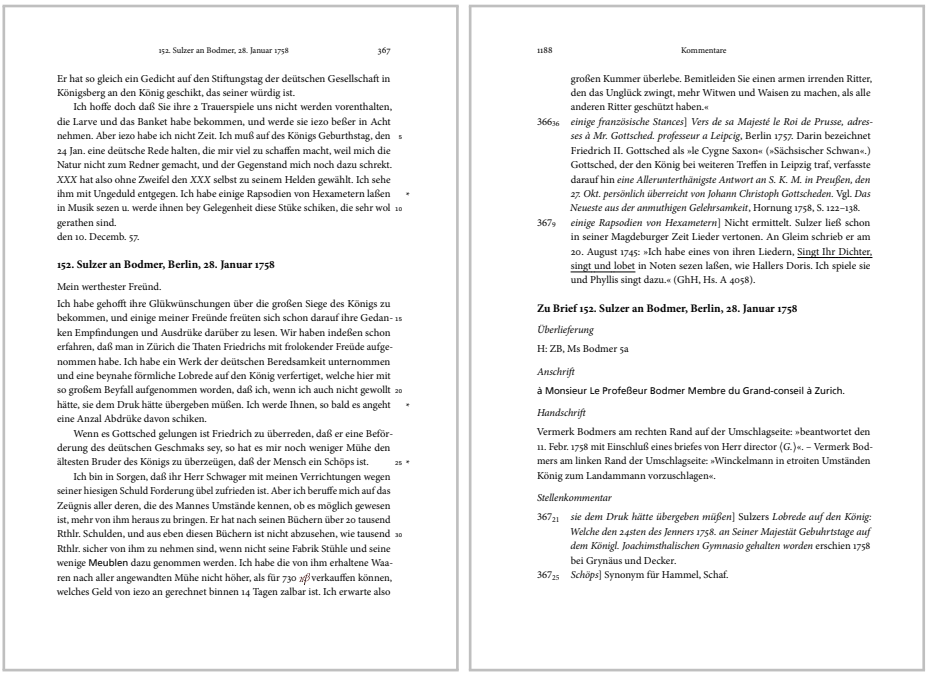
The functionalities for consistency checking and text combination are available as Prolog predicates in a user interface module, and, for users that do not want to interact with Prolog directly, with *Bash* shell scripts that invoke *SWI-Prolog*.[10]

2.3 HTML Presentation

The parsed source documents are converted to representations as Prolog predicates, which form the basis for generating an HTML representation of the scholarly edition. In general, our Web presentation is designed to open-up the edition, to make it easy to get an overview on the material and on the supported navigation possibilities.

On the basis of the identifiers in the LaTeX-syntax source documents, URIs for documents like letters and entities like persons and locations are generated.[11] These can be used as URLs of the respective generated pages, which then can persistently represent the respective document or entity with respect to the

[9] Before printing in high quality, LaTeX documents in general need manual adjustments in places that can not be handled satisfactorily by the automated layout processor.
[10] In Microsoft Windows, these scripts can be called from the *Cygwin* shell.
[11] This requires a syntactic conversion as "`:`" has a special meaning in URIs.



152. Sulzer an Bodmer, 28. Januar 1758 367

Er hat so gleich ein Gedicht auf den Stiftungstag der deütschen Gesellschaft in Königsberg an den König geschikt, das seiner würdig ist.

Ich hoffe doch daß Sie ihre 2 Trauerspiele uns nicht werden vorenthalten, die Larve und das Banket habe bekommen, und werde sie iezo beßer in Acht nehmen. Aber iezo habe ich nicht Zeit. Ich muß auf des Königs Geburthstag, den 24 Jan. eine deütsche Rede halten, die mir viel zu schaffen macht, weil mich die Natur nicht zum Redner gemacht, und der Gegenstand mich noch dazu schrekt. *XXX* hat also ohne Zweifel den *XXX* selbst zu seinem Helden gewählt. Ich sehe ihm mit Ungeduld entgegen. Ich habe einige Rapsodien von Hexametern laßen in Musik sezen u. werde ihnen bey Gelegenheit diese Stüke schiken, die sehr wol gerathen sind.

den 10. Decemb. 57.

152. Sulzer an Bodmer, Berlin, 28. Januar 1758

Mein werthester Freünd.

Ich habe gehofft ihre Glükwünschungen über die großen Siege des Königs zu bekommen, und einige meiner Freünde freüten sich schon darauf ihre Gedanken Empfindungen und Ausdrüke darüber zu lesen. Wir haben indeßen schon erfahren, daß man in Zürich die Thaten Friedrichs mit frolokender Freüde aufgenommen habe. Ich habe ein Werk der deütschen Beredsamkeit unternommen und eine beynahe förmliche Lobrede auf den König verfertiget, welche hier mit so großem Beyfall aufgenommen worden, daß ich, wenn ich auch nicht gewollt hätte, sie dem Druk hätte übergeben müßen. Ich werde Ihnen, so bald es angeht eine Anzal Abdrüke davon schiken.

Wenn es Gottsched gelungen ist Friedrich zu überreden, daß er eine Beförderung des deütschen Geschmaks sey, so hat es mir noch weniger Mühe den ältesten Bruder des Königs zu überzeügen, daß der Mensch ein Schöps ist.

Ich bin in Sorgen, daß ihr Herr Schwager mit meinen Verrichtungen wegen seiner hiesigen Schuld Forderung übel zufrieden ist. Aber ich beruffe mich auf das Zeügnis aller deren, die des Mannes Umstände kennen, ob es möglich gewesen ist, mehr von ihm heraus zu bringen. Er hat nach seinen Büchern über 20 tausend Rthlr. Schulden, und aus eben diesen Büchern ist nicht abzusehen, wie tausend Rthlr. sicher von ihm zu nehmen sind, wenn nicht seine Fabrik Stühle und seine wenige Meubles dazu genommen werden. Ich habe die von ihm erhaltene Waaren nach aller angewandten Mühe nicht höher, als für 730 *xβ* verkauffen können, welches Geld von iezo an gerechnet binnen 14 Tagen zalbar ist. Ich erwarte also

1188 Kommentare

großen Kummer überlebe. Bemitleiden Sie einen armen irrenden Ritter, den das Unglück zwingt, mehr Witwen und Waisen zu machen, als alle anderen Ritter geschützt haben.«

366$_{36}$ *einige französische Stances*] *Vers de sa Majesté le Roi de Prusse, adresses à Mr. Gottsched. professeur a Leipcig*, Berlin 1757. Darin bezeichnet Friedrich II. Gottsched als »le Cygne Saxon« (»Sächsischer Schwan«.) Gottsched, der den König bei weiteren Treffen in Leipzig traf, verfasste darauf hin *eine Allerunterthänigste Antwort an S. K. M. in Preußen, den 27. Okt. persönlich überreicht von Johann Christoph Gottscheden*. Vgl. *Das Neueste aus der anmuthigen Gelehrsamkeit*, Hornung 1758, S. 122–138.

367$_{9}$ *einige Rapsodien von Hexametern*] Nicht ermittelt. Sulzer ließ schon in seiner Magdeburger Zeit Lieder vertonen. An Gleim schrieb er am 20. August 1745: »Ich habe eines von ihren Liedern, Singt Ihr Dichter, singt und lobet in Noten sezen laßen, wie Hallers Doris. Ich spiele sie und Phyllis singt dazu.« (GhH, Hs. A 4058).

Zu Brief 152. Sulzer an Bodmer, Berlin, 28. Januar 1758

Überlieferung

H: ZB, Ms Bodmer 5a

Anschrift

à Monsieur Le Profeßeur Bodmer Membre du Grand-conseil à Zurich.

Handschrift

Vermerk Bodmers am rechten Rand auf der Umschlagseite: »beantwortet den 11. Febr. 1758 mit Einschluß eines briefes von Herr director (*G.*)«. – Vermerk Bodmers am linken Rand der Umschlagseite: »Winckelmann in etroiten Umständen König zum Landammann vorzuschlagen«.

Stellenkommentar

367$_{21}$ *sie dem Druk hätte übergeben müßen*] Sulzers *Lobrede auf den König: Welche den 24sten des Jenners 1758. an Seiner Majestät Gebuhrtstage auf dem Königl. Joachimsthalischen Gymnasio gehalten worden* erschien 1758 bei Grynäus und Decker.

367$_{25}$ *Schöps*] Synonym für Hammel, Schaf.

Fig. 2. *KBSET/Letters*: PDF presentation, a letter and an annotation page.

edition. An HTML presentation of the project bibliography is generated from the *BibLaTeX* sources via an invocation of the *Biber* processor with options such that it produces an XML representation of the processed bibliography that is then read into *SWI-Prolog*.

The Web presentation just uses static pages, in HTML5, with CSS3 and – very little – JavaScript. This makes the loading of pages fast, requires no maintenance efforts, and facilitates the interaction with search engines, general Web search engines as well as dedicated engines for the online publication.

Some simple but useful means for navigation were realized: Letter pages have links to a chronologically next and previous letter, with respect to the writer and also with respect to the correspondence with the addressee. These four links are always displayed at the same position in the page and thus allow to quickly move within the letters by an author or in a correspondence.

Another realized useful navigation means is what we call *chains* (*Ketten*), or, more explicitly, *result value chains*: The value of a query is often a "chain", that is, an ordered set of entities, represented as a series of links. Navigating through such a chain is facilitated by a special type of Web pages, *chain pages*, which just display the chain of links but are invoked parameterized by an index into the chain. They scroll their content automatically such that the indexed link appears at the top. By clicking at some link or a *next* button (for the indexed link) in the chain window the respective linked document is opened in a different

window, and the index of the chain window is incremented (a *previous* button has analogous effect). We actually use chains for a finite number of precomputed queries of general interest such as the set of all letters in which a given person is referenced, and whose results are also displayed on the respective entity pages – but are there less convenient to browse through. Chain pages are by default shown in a small pop-up window positioned top left on the screen. If possible, an existing chain window and an existing window for displaying a page linked from a chain window are re-used. Our implementation utilizes the CSS3 `target` attribute. Figure 3 shows an example of a generated Web page representing a letter, accompanied by a chain page.

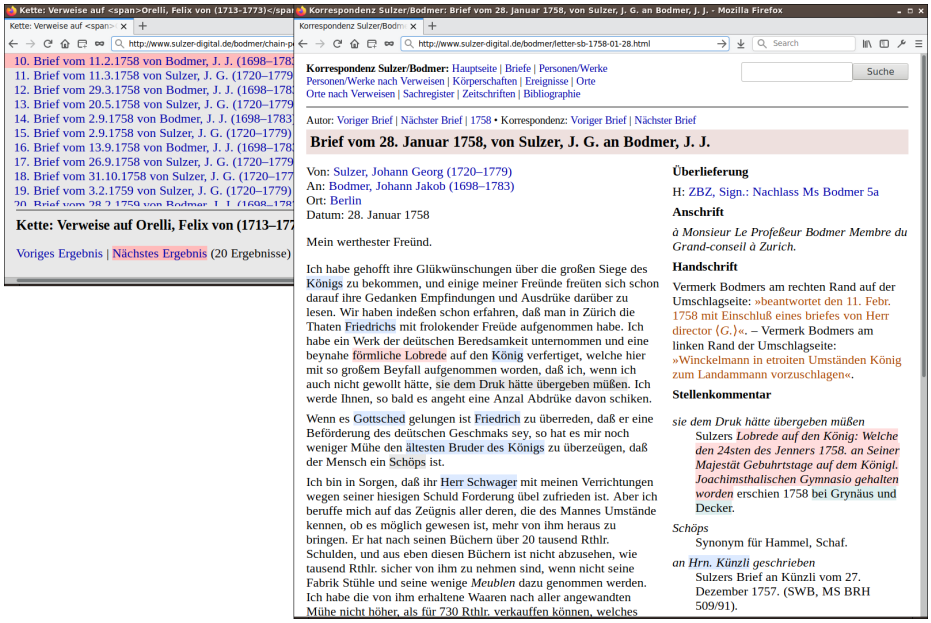


Fig. 3. *KBSET/Letters*: HTML presentation, a letter and a chain page.

2.4 Access from Prolog and Export of Fact Bases

The advanced consistency checking and text combination, as well as the HTML generation can be invoked via *Bash* shell scripts or directly from Prolog. The representation of the parsed source documents as Prolog predicates underlying the HTML conversion can in principle also be applied for other applications, such as conversion to further formats like RDF/XML and TEI/XML, or to export fact bases as indicated with box O2 in Fig. 1. The plan is to specify a suitable set of Prolog predicates such that editions can offer exported data, and also the parsed text, for download. Currently ways to produce RDF and XML on the basis of the internal Prolog predicates are indicated with small examples in the source code.

3 *KBSET/NER*

While *KBSET/Letters* addresses the practical aspects of comprehensive scholarly correspondence editions, the focus of *KBSET/NER* is to explore experimentally potential future directions of scholarly editing. Specifically the integration of techniques that return non-symbolic, fuzzy or incomplete results, the utilization of large external fact bases from the library community such as the *GND* and from Semantic Web activities such as *YAGO*, *DBPedia* and *GeoNames*, and ways to handle the association of annotations with places in the object text that are not explicitly marked as reference target. The functionality of *KBSET/NER* can be accessed from the Prolog interpreter or with menus and keyboard shortcuts from *GNU Emacs*. A draft edition of *Geschichte der Reaction*, vol. 1, 1852, by philosopher Max Stirner that has been created with these novel techniques is included with the *KBSET* distribution.

3.1 Caching External Knowledge Bases for Access Patterns

The inputs of *KBSET/NER* include, aside of object texts and annotation documents (boxes I1 and I2 in Fig. 1), also large imported fact bases (box I5). Before use, the configured fact bases, which are typically available in Semantic Web formats like RDF/XML or as CSV tables, have to be downloaded, parsed and preprocessed. This can be done with a utility predicate, but, as it may take several hours, for the example application also a TAR archive with the results of the preprocessing can be downloaded from the *KBSET* home page.[12] The preprocessed fact bases are then loaded into the Prolog system. At the first loading they are compiled into *SWI-Prolog*'s *quick-load* format. In that format our fact base with 12 million ternary facts on persons born before 1850 extracted from the *GND* takes 7 seconds to load on a modern notebook computer.

KBSET then accesses these data as Prolog predicates stored in main memory. The indexing mechanisms of *SWI-Prolog* are utilized by maintaining predicates that are adapted to the represented entities, such as persons or locations (in contrast to generic triple predicates as might be suggested by the RDF format), and to access patterns. For example, a predicate for accessing data about a set of persons via a given last name and another predicate for accessing data about a person via a given *GND* identifier. We call these predicates, which may be in part redundant from a semantic point of view, *caches*. In the current implementation, the caches are in part computed when preprocessing the fact bases and in part when loading them. With this approach, the system can evaluate the several 10.000s of queries against the fact bases required for named entity recognition on the example document in a few seconds. Another useful feature is the semantics-based restriction of the large fact bases at preprocessing them. Since our example edition is a book from 1852, we keep of the *GND* only the facts about persons born before 1850.

[12] Also the original fact bases used for the example application are archived on the *KBSET* home page, as none of them has a persistent URL.

3.2 Named Entity Identification

Working with *KBSET/NER* is centered around a subsystem for named entity recognition, which detects dates by parsing as well as persons and locations based on the *GND* and *GeoNames* as gazetteers, using additional knowledge from *YAGO* and *DBpedia*. Persons can be detected in two modes, characterized by names as well as by functional roles like *King of*, *Duke of* and *Bishop of*. Differently from systems like the *Stanford Named Entity Recognizer* [4], *KBSET/NER* does not just associate entity types such as *person* or *location* with phrases but attempts to actually *identify* the entities, hence we also speak of *named entity identification*.

The identification of persons and locations is based on single word occurrences with access to a context representation that includes the text before and after the respective occurrence. Hence an association of *word occurrences* to entities is computed, which is adequate for indexes of printed documents and for hypertext presentations, but not fully compatible with TEI, where the idea is to enclose a *phrase that denotes an entity* in markup.

Figure 4 shows the presentation of named entity identification results in *GNU Emacs*. In the upper buffer, which contains the object text, the system highlights words or phrases about which it assumes that they denote a person, location or date. In the lower buffer additional information on the selected occurrence of *Gleim* is displayed: Links to *Wikipedia* and *GND*, an explanation *why* the system believes the entity to be a plausible candidate for being referenced by the word occurrence, and an ordered listing of lower-ranked alternate candidate entities. Menus and keyboard shortcuts allow to jump quickly between the highlighted text positions with associated entities.

Aside of the presentation in *GNU Emacs*, the results of named entity identification can be output in different formats, in particular merged into a LaTeX source document as annotations. In this merging process also external annotation documents can be considered, where the positions to insert particular annotations are abstractly specified, for example by some form of text pattern. Further supported output formats of the named entity identification results include the presentation as TEI/XML elements merged into a source document, as a Prolog fact base, or, for identified locations, as a CSV table that can be loaded into the *DARIAH-DE* geo browser.

The named entity identification is controlled by rules which can be specified and configured and determine the evaluation of syntactic features matched against the considered word, for example, *is-no-stopword* or *is-no-common-substantive*, and of semantic features matched against candidate entities, for example, *is-in-wikipedia*, *is-linked-to-others-identified-in-context*, *has-an-occupation-mentioned-in-context*, or *date-of-birth-matches-context*. Evaluation of these features is done with respect to the mentioned context representation, which includes general information like the date of text creation and inferred information such as a set of entities already identified near the evaluated text position. Features that are cheap to compute and have great effect on restricting the set of candidate entities are evaluated first. This allows, for example, to apply named

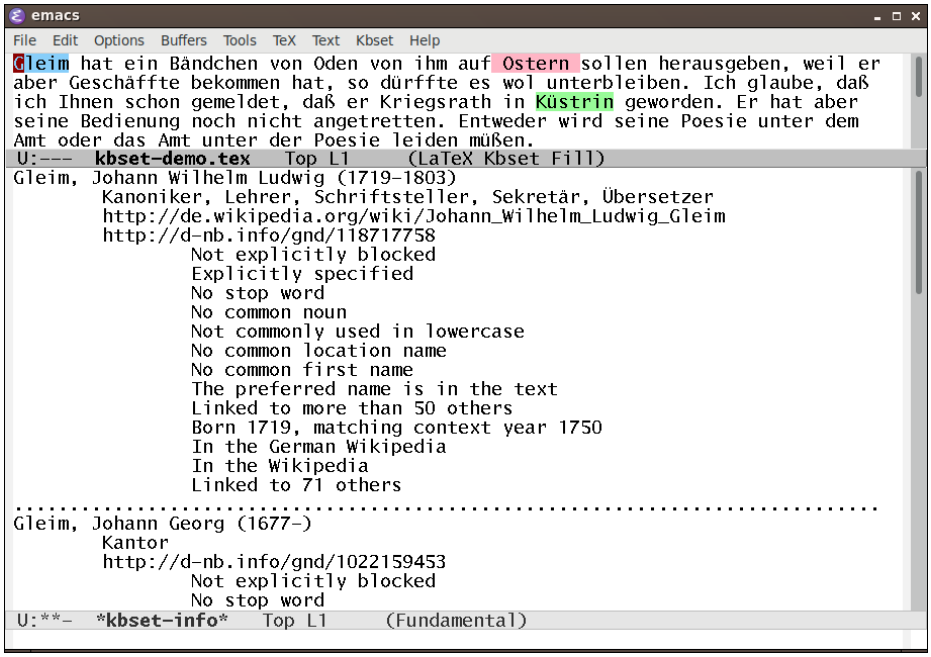


Fig. 4. *KBSET/NER*: Named entity identification from *GNU Emacs*.

entity identification of persons on the 300-pages example book provided with the system in about 7 seconds on a modern notebook computer. Feature evaluation results are then mapped to Prolog terms whose standard order represent their plausibility ranking. Information about the features that contributed to selection of a candidate entity is preserved and used to generate the displayed explanations shown in Fig. 4.

3.3 Assistance Documents

The automated named entity identification produces incomplete, partially incorrect and, by presenting a ranked list of plausible entities, fuzzy results. Such results may be helpful for developing a scholarly edition but should not remain in a released version. Hence, there must be a possibility to adapt them. This can be done in *KBSET/NER* with configuration files, so-called *assistance documents* (box I4 in Fig. 1). These specify the complete configuration of *KBSET/NER*, including the URLs of the external fact bases, the preprocessing required to use them, and how to bias or override automated inferencing in named entity identification. The idea for the latter is that the user, instead of annotating identified entities manually, lets the system do it automatically and mainly gives hints in *exceptional* cases, where the automatic method would otherwise not recognize an entity correctly. That method was used in the example document supplied with *KBSET*.

In the assistance document the explicit appearance of technical identifiers such as the identifiers from the *GND* should be avoided. This is achieved by permitting to specify a person just by some attributes like name, year of birth, and/or a profession. These specifiers are evaluated with the current fact base of the system, that is, essentially the *GND*. In contexts where a unique person must be designated an error is signaled if no or several persons match the specifier. (Of course, this method is not stable against importing an extended version of the *GND*.) In addition, context properties can be specified that characterize when the biasing should be applied. Other options are to register persons that can not be found in the *GND* and to supplement attributes of persons in the *GND*. Also simple syntactic exclusions, for example that a certain word should not denote a person or location can be specified. Here is an excerpt from the assistance document for the included example. It "assists" the automated named entity identifier in distinguishing two persons named *Tacitus*, classifying *Starcke* (the printer of the book) as a person and identifying him, as well as in identifying the person referenced in the text as *Herzog von Luxemburg*:

```
entity(person,
       [name='Tacitus',
        professionOrOccupation='Historiker'],
       [near_word_in=['Römern']]),
entity(person,
       [name='Tacitus',
        variantNameForThePerson='Tacitus, Römisches Reich, Kaiser'],
       [near_word_in=['Adel']]),
entity(person,
       [name='Starcke, Johann Friedrich',
        professionOrOccupation='Drucker'],
       [near_word_in=['Druck']]),
supplement(person,
           [name='Joseph II., Heiliges Römisches Reich, Kaiser'],
           [biographicalOrHistoricalInformation
            =lang(de,'Herzog von Luxemburg (1765-1790)')]),
```

Like Prolog program files, assistance documents can be re-loaded, which effects updating of the specified settings. Thus, the named entity identification of *KBSET/Letters* can be improved in an iteration of adjustments of the assistance documents and reviewing the effects in the *GNU Emacs* presentation.

This mode of interaction has, however, in the Sulzer-Bodmer edition project only been used in occasional cases. Each letter has there been transcribed and extensively commented by scientists, where the manual entity tagging emerged as a by-product. The automated named entity identification has been applied in special situations such as initializing the tagging of locations, examining and completing the manual tagging of persons, and generating auxiliary fact bases that map the project-local entity identifiers to global ones from the *GND* and *GeoNames*.

4 Discussion

We conclude this inter-disciplinary paper with discussions of the *KBSET* environment from three different perspectives.

4.1 *KBSET* in the World of Tools for Digital Scholarly Editing

KBSET has been designed and written within the paradigm of programming and creating mechanizable formalizations in Artificial Intelligence,[13] which is considered there as an integral component of the research activity. The creation of text with markup (LaTeX) is daily routine for researchers in computer science in general, as well as in numerous further fields.

In contrast, in the Humanities the use of formally defined languages entered in the last decade largely from the outside, with the requirement to make publicly funded results openly available in the HTML-based Web. Customizations of TEI/XML seemed the format of choice.[14] Hence the creation of TEI/XML documents became a component of the scholarly editing workflow.[15] However, this should not be misunderstood as equating the creation of digital editions to working with TEI/XML. The text represented in TEI/XML documents is hardly readable and the computational treatment of TEI/XML usually requires familiarity with several dedicated transformation and query languages, such that edition projects are typically large undertakings that are accompanied by support from a specialized IT department, which mediates between the formal languages and the researcher. Observe that this is quite different from research in Artificial Intelligence, where the researcher herself creates mechanizable formalizations and programs. Where should the Digital Humanities go?

As TEI/XML is a general scheme for encoding all sorts of text, *"it is almost impossible to use the TEI schema without customizing it in some way"* [14, Sect. 23.3]. Applications such as scholarly editions of letters typically use project- or organization-specific customizations. Such customizations should be formalized in a schema language and explained in an informal document, both of which should be made accessible with the digital edition ([14, Sect. 23.4]

[13] In the sense of the discipline *Artificial Intelligence*, not as synecdoche for its subfield *Machine Learning*.

[14] See for example the DFG (German Research Foundation) document *Förderkriterien für wissenschaftliche Editionen in der Literaturwissenschaft, Ausgabe 11/2015*, `https://www.dfg.de/download/pdf/foerderung/grundlagen_dfg_foerderung/informationen_fachwissenschaften/geisteswissenschaften/foerderkriterien_editionen_literaturwissenschaft.pdf`.

[15] Scholarly editions of correspondences that offer an openly available TEI/XML presentation include *Alfred-Escher Briefedition* (`https://www.briefedition.alfred-escher.ch`), *Briefe und Texte aus dem intellektuellen Berlin um 1800* (`https://www.berliner-intellektuelle.eu`), *Digitale Edition der Korrespondenz August Wilhelm Schlegels* (`https://august-wilhelm-schlegel.de`), *hallerNet* (`http://hallernet.org`), and *edition humboldt digital* (`https://edition-humboldt.de`).

gives very specific notions of this, even claimed to be presuppositions for calling a document *TEI-conformant*). Unfortunately, in current practice such schema specifications and documentations for digital editions are only rarely made easily accessible.[16] The suggested way to associate a TEI/XML document instance with a schema by the `xml-model` processing instruction [14, Sect. v.7.2] seems not used at all.[17]

Writing a conversion from *KBSET/Letters* source documents to some customization of TEI/XML is an easy task based on the extraction process implemented for the HTML transformation. The markup in LaTeX-syntax is there available in parsed form, metadata appear as Prolog predicates, and routines for converting identifiers are already implemented. A module in *KBSET* illustrates the concrete proceeding for XML and RDF conversions of metadata. In fact, a conversion to a TEI/XML customization is much simpler than the HTML translation included in *KBSET*. It is not yet implemented for the reason that, so far, it seems difficult to identify a particular formally defined TEI/XML customization for correspondences for which interesting tools or services are openly available, for example, to generate further presentations or for integration with other editions.

In the light of the standardization efforts via TEI/XML, *KBSET* can be taken as a user-friendly and economic environment for developing scholarly editions that approaches compliance with the desiderata described in the introduction. The generation of a representation in some customized TEI/XML format for interchange and archival is a marginal feature that is easy to add. In the long run, variations of the *KBSET* markup language should perhaps be adapted to reflect some suitable TEI/XML customizations more explicitly, or even be considered as realizations of TEI customizations in LaTeX-syntax.

Vice versa, *KBSET/Letters* can also be taken as a tool for generating presentations. It is not difficult to translate a representation of a correspondence in a TEI/XML customization to the *KBSET/Letters* markup (this can be implemented on the basis of the term representations of documents obtained from the XML parser of *SWI-Prolog*) such that the PDF and HTML presentations offered by *KBSET/Letters* become available. Since, as already mentioned, projects use different and hardly documented TEI/XML customizations it is expected that the translations need to be project-specific and some trial-and-error is involved in the development.

KBSET is free software. It depends only on a TeX distribution (it has been tested with *TeX Live*) and on *SWI-Prolog*, both of which are also free software,

[16] Actually, the authors were (in November 2019) not able to find any correspondence edition where a formal specification of the used customized schema is referenced from the TEI/XML documents or specified on the Web site. Informal edition guidelines can be found, for example, on the Web sites of *Alfred-Escher Briefedition*, *Briefe und Texte aus dem intellektuellen Berlin um 1800* and *hallerNet*.

[17] The well-intentioned postulation *"Um die Austauschbarkeit und Nachnutzung zu ermöglichen, werden die projektspezifisch verwendeten XML-Elemente und Attribut-Wert-Paare im TEI-Header dokumentiert"* in the DFG document mentioned in footnote 14 can technically not refer to the `teiHeader` element.

platform independent, and, moreover, mature, stable and widely used such that the current implementation of *KBSET* can be expected to operate also with future releases of these environments.[18]

The sources of an edition project like the Sulzer-Bodmer correspondence can be published and archived together with the used version of *KBSET/Letters*. The following functionalities are then freely available, through the stability and platform independence of LaTeX and *SWI-Prolog* also in the foreseeable future: Generation of various high-quality PDF and HTML representations, generation of fact bases in Prolog representation,[19] and the representation in some TEI/XML customization (which still needs to be implemented). Moreover, if users want to improve or extend these functionalities, *KBSET/Letters* is available as a concrete and working free software environment to begin with.

The use of *KBSET* with other languages than German is supported to some degree: All input documents created for *KBSET* are encoded in UTF-8. The *GNU Emacs* user interface of *KBSET/NER* can be configured to English or German. Some of the word lists included in the implementation are, however, so far provided only for German. Also the presentation templates of *KBSET/Letters* are currently only in German. The *BibLaTeX* configuration included currently with *KBSET/Letters* is based on practices of the Humanities in Germany, but it is no problem to replace it with a different configuration.

4.2 *SWI-Prolog* as a Unifying Practical Technology

The core system of *KBSET/Letters* is written in *SWI-Prolog*, which realizes the potential of Prolog as a unifying language. As noted on the *SWI-Prolog* home page,[20] it considers Prolog *"primarily as glue between various components. The main reason for this is that data is at the core of many modern applications while there is a large variety in which data is structured and stored. Classical query languages such as SQL, SPARQL, XPATH, etc. can each deal with one such format only, while Prolog can provide a concise and natural query language for each of these formats that can either be executed directly or be compiled into dedicated query language expressions. Prolog's relational paradigm fits well with tabular data (RDBMS), while optimized support for recursive code fits well with tree and graph shaped data (RDF)."* The particular roles of Prolog, and in particular *SWI-Prolog*, for *KBSET* can be compiled as follows:

1. *Declarative representation mechanism for relational fact bases.* As outlined in Sect. 2.4, we convert the document sources created in scholarly edition projects and large external fact bases to an intermediate representation as

[18] Some of the functions of *KBSET* can be invoked in addition from *Bash* shell scripts. A *Bash* shell can be presupposed on Unix-like platforms and can be added, for example with *Cygwin*, to Microsoft Windows platforms.

[19] Considering that there is an ISO standard for Prolog, such fact bases are actually in a *standardized* format. However, the ISO standard for Prolog is only with respect to ASCII encoding. Modern implementations like *SWI-Prolog* support UTF-8.

[20] `https://www.swi-prolog.org/features.html`, accessed Nov 21 2019.

Prolog predicates, which are then used, for example, to generate HTML pages, but are also available for other purposes, including export as fact bases or interactive querying on the Prolog shell. The declarative view brings *semantics* into the focus and offers a bridge to the wealth of semantics-based techniques for knowledge representation and knowledge-based reasoning, in particular deductive databases, model- and answer-set computation, first-order theorem proving, and ontology reasoning.

2. *Efficient representation mechanism for relational fact bases.* We utilize the predicate indexing facilities of *SWI-Prolog*'s with predicate caches that are specialized to access patterns as outlined in Sect. 3.1.
3. *Query language.* The standard predicates *findall* and *setof* provide expressive means to specify queries in a declarative manner. Complex tests and constructions can be smoothly incorporated, as query and programming language are identical, without much impedance mismatch. Of course, queries written in Prolog can not rely on an optimizer, and have to be designed "manually" such that their evaluation is done efficiently. A further important feature of Prolog is fast sorting based on a standard order of terms, which we quite often use to canonicalize representations of sets and is also the basis of our implementation of ranked answers in named entity identification.
4. *Representation mechanism for structured documents.* As in Lisp, data structures are in Prolog by default terms that are print- and readable, a feature which is supplemented to "non-AI" languages often by XML serialization. In our application context this is particularly useful as it allows to represent XML and HTML documents directly as Prolog data structures, that is, terms.
5. *Parser for XML and Semantic Web formats. SWI-Prolog* comes with powerful interfaces to Semantic Web formats, of which we use in particular the XML parser and the RDF parser, which provides a call-back interface that allows to process in succession the triples represented in a large RDF document such as the *GND* (see footnote 5 in Sect. 1).
6. *Parser for natural language text fragments and for formal languages.* Prolog has been developed originally in the context of applications in linguistics and traditionally supports syntax for grammar rules that are translated into an advanced parsing system. In *KBSET* this feature is used to parse date specifications in various contexts, to parse person specifications by functional roles in named entity identification, and to implement the LaTeX parser.
7. *Practical workflow model.* Workflow aspects of experimental AI programming seem also useful in the Digital Humanities: loading and re-loading documents with formal specifications as well as invocation of functionality and running of experiments through an interpreter. All of this manageable by the researcher herself instead of further parties.
8. *Programming language.* Not to forget: Prolog is a programming language that is "different, but not *that* different" [10, Introduction].

4.3 Some Issues for Logic-Based Knowledge Processing

KBSET is an implemented system that has been proved workable in an application project and allows to experimentally study further possibilities. Some of the issues encountered in the course of implementing that were solved in specific ways seem to deserve further investigation. One of these issues is the interplay of knowledge that is inferred by automated and statistic-based techniques such as named entity recognition with manually supplied knowledge, which is addressed in *KBSET* so far with the *assistance documents*. Non-monotonic reasoning should in principle be a logic-based technique that is applicable here. Related to this issue is the handling of *ranked* query results used in *KBSET* for named entity identification. This is known in the field of databases as *top-k querying*. Is it possible to add some systematic and logic-based support for this to Prolog and perhaps also bottom-up reasoners like deductive database systems and model generators?

The approach to access fact bases with several millions of facts via preprocessed caches as realized by *KBSET* might be of general interest and could be investigated and implemented more systematically. If queries are written in a suitable fragment of Prolog, they can be automatically optimized, abstracting from caring about indexes (i.e., which cache is used), the order of subgoals, and the ways in which answer components are combined. Recent approaches to interpolation-based query reformulation [15,1] investigate a declarative approach for this. The optimized version of a query is there extracted as a Craig interpolant [2,16] from a proof obtained from a first-order prover. It seems also possible to apply this approach to determine from a given set of queries the caches that need to be constructed for efficient evaluation of the queries.

Digital Scholarly editing involves the interplay of natural language text with formal code and with formalized knowledge bases. From a general point of view, the contribution of the computer in digital scholarly editing may be viewed as a variant of the classical Artificial Intelligence scenario, where an agent in an environment makes decisions on actions to perform: General background knowledge in the AI scenario corresponds to knowledge bases like *GND* and *GeoNames*; the position of the agent in the environment may correspond to a position in the text; temporal order of events to the order of word occurrences; the environment which is only incompletely sensed or understood by the agent corresponds to incompletely understood natural language text; coming to decisions about actions to take corresponds to decisions about denotations of text phrases and about annotations to associate with text components. This suggests that digital scholarly editing is an interesting field for applying, improving and inventing AI techniques.

References

1. Benedikt, M., Leblay, J., ten Cate, B., Tsamoura, E.: Generating Plans from Proofs: The Interpolation-based Approach to Query Reformulation. Morgan & Claypool (2016)
2. Craig, W.: Three uses of the Herbrand-Gentzen theorem in relating model theory and proof theory. The J. of Symbolic Logic **22**(3), 269–285 (1957)
3. Eide, O.: Ontologies, data modeling, and TEI. Journal of the Text Encoding Initiative **8** (2015)
4. Finkel, J.R., Grenager, T., Manning, C.: Incorporating non-local information into information extraction systems by Gibbs sampling. In: ACL 2005. pp. 363–370. ACL (2005)
5. Hoffart, J., Suchanek, F.M., Berberich, K., Weikum, G.: YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. Artificial Intelligence **194**, 28–61 (2013)
6. Kittelmann, J., Wernhard, C.: Semantik, Web, Metadaten und digitale Edition: Grundlagen und Ziele der Erschließung neuer Quellen des Branitzer Pückler-Archivs. In: Krebs, I., et al. (eds.) Resonanzen. Pücklerforschung im Spannungsfeld zwischen Wissenschaft und Kunst, pp. 179–202. trafo Verlag (2013)
7. Kittelmann, J., Wernhard, C.: Knowledge-based support for scholarly editing and text processing. In: DHd 2016. pp. 178–181. nisaba verlag (2016)
8. Kittelmann, J., Wernhard, C.: Von der Transkription zur Wissensbasis. Zum Zusammenspiel von digitalen Editionstechniken und Formen der Wissensrepräsentation am Beispiel von Korrespondenzen Johann Georg Sulzers. In: Kittelmann, J., Purschwitz, A. (eds.) Aufklärungsforschung digital. Konzepte, Methoden, Perspektiven, IZEA – Kleine Schriften, vol. 10/2019, pp. 84–114. Mitteldeutscher Verlag (2019)
9. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., Bizer, C.: DBpedia – A large-scale, multilingual knowledge base extracted from Wikipedia. Semantic Web **6**(2), 167–195 (2015)
10. O'Keefe, R.A.: The Craft of Prolog. The MIT Press (1990)
11. Plachta, B.: Editionswissenschaft: Eine Einführung in Methode und Praxis der Edition neuerer Texte. Reclam (1997)
12. Sahle, P.: Digitale Editionsformen, Zum Umgang mit der Überlieferung unter den Bedingungen des Medienwandels, 3 volumes, Schriften des Instituts für Dokumentologie und Editorik, vol. 7-9. Books on Demand (2013)
13. Sulzer, J.G.: Gesammelte Schriften. Kommentierte Ausgabe. Adler, H., Décultot, E. (eds.). Schwabe (2014–2021)
14. The TEI Consortium: TEI P5: Guidelines for Electronic Text Encoding and Interchange, Version 3.6.0. Text Encoding Initiative Consortium (2019), `http://www.tei-c.org/Guidelines/P5/`
15. Toman, D., Weddell, G.: Fundamentals of Physical Design and Query Compilation. Morgan and Claypool (2011)
16. Wernhard, C.: Facets of the PIE environment for proving, interpolating and eliminating on the basis of first-order logic. In: DECLARE 2019, Revised Selected Papers. LNCS (LNAI), vol. [this volume] (2020)
17. Wielemaker, J., Schrijvers, T., Triska, M., Lager, T.: SWI-Prolog. Theory and Practice of Logic Programming **12**(1-2), 67–96 (2012)