Conversion of The Clark Hall Dictionary of Old English to TEI with RDF: An End-to-end Pipeline for Lexicographic Resource Retrodigitization

Sergei Stoliarov[1], **Maxim Ionov**[2], **Fahad Khan**[3], **Marina Buzzoni**[1], **Francesca Frontini**[3]
[1]Università Ca' Foscari Venezia, Italy , [2]University of Zaragoza, Spain,
[3]Istituto di Linguistica Computazionale "A. Zampolli" (CNR-ILC), Italy
[1]{900590@stud., mbuzzoni}.unive.it, [2]mionov@unizar.es, [3]{fahad.khan, francesca.frontini}@ilc.cnr.it

Abstract

In this submission we introduce a workflow/pipeline for creating TEI editions of legacy dictionaries using a parser based on a context-free grammar (CFG). This was developed as part of a project which we are currently carrying out and which aims to create a digital edition of an Old English dictionary, Clark-Hall's *A Concise Anglo-Saxon Dictionary* (Clark Hall, 1916). We begin the article by motivating our CFG-based approach, discussing its advantages and disadvantages, and comparing to it other approaches. We will argue that it is suitable to certain kinds of dictionaries as exemplified by the Clark-Hall. We then describe the microstructure of the dictionary itself with a view to justifying the kinds of rules which we subsequently describe. We then describe the CFG parser itself and give an account of our experiments in parsing the Clark-Hall dictionary. Finally, we outline the enrichment of the parsed dictionary with RDFa and the benefits it has for the published data.

Keywords: retrodigitization, context-free grammar, dictionary, tei, rdfa, old english, minimal computing

1. Introduction

The present submission introduces a workflow/pipeline for creating TEI-XML editions of legacy lexicographic works using Context-Free Grammars; these editions are subsequently semantically enriched with RDFa triples. This pipeline has been developed during a project which the authors are currently undertaking and which has the aim of producing a TEI-XML edition of a well-known legacy lexicographic work, namely the second edition of Clark-Hall's *A Concise Anglo-Saxon Dictionary* (Clark Hall, 1916). Our approach in this project takes inspiration from current efforts towards promoting minimal computing as a way of making the Digital Humanities accessible to a wider range of participants;[1] in particular we have sought, as far as possible, to use of standard off-the-shelf technologies without, e.g., the necessity of access to LLMs (although these can assist in several stages of the pipeline). We have also tried to give users as much control as possible in the process of parsing and extracting information from a lexicographic work. In addition, our approach is tailored to projects, or more generally, to settings, where there may be limited access to technical expertise and certain kinds of technologies/hardware, but where there is, in compensation, a greater availability of humanistic expertise and where therefore the construction/checking/adaption of human readable parsing rules therefore becomes a more viable and a more desirable approach to creating structured datasets.

As mentioned, our approach relies on the use of CFGs, for crafting human readable rules for parsing a text file. We are aware that a CFG-led approach to parsing lexicographic texts is potentially quite labour intensive and that it may also suffer from other kinds of (technical) limitations in terms of computational efficiency. We will discuss these limitations and the kinds of use cases where our approach should be most suitable[2] as well as comparing it to other tools and other approaches in what follows. The accessibility of our approach heavily depends on the availability, to subsequent users, of extensive documentation as well as pre-existing CFG-based parsers for a range of dictionaries which can be easily adapted to similar user cases. Consequently, we have made a point of documenting our work: this includes making available detailed notes on how we have gone about encoding different kinds of lexicographic/microstructural phenomena and that will be useful for those encoding dictionaries which include similar organisational features, *and* for understanding the rules which we have crafted. The CFG parser, documentation on the modelling of the dictionary in TEI-XML, and a selection of hand encoded entries and the TEI output of the parser (to be subsequently enriched in another stage of the project) have been made avail-

[1]See for instance https://sas-dhrh.github.io/dhcc-toolkit/toolkit/minimal-computing.html

[2]Spoiler: cases where the dictionary microstructure isn't too complicated and/or where there is a certain kind of expertise available.

Proceedings of the Fifteenth Language Resources and Evaluation Conference (LREC 2026), pages 7046–7055
11–16 May 2026.

able in our github repository as part of the current submission.[3]

The rest of the paper is structured as follows. In Section 2 we give some background on related work. In Section 3 we describe the Clark-Hall dictionary itself, give details on our approach to modelling it and the challenges it poses. Then in Section 4 we describe the parser itself. Finally, in Section 5 we outline our approach of enriching TEI with RDF and subsequently publishing the data.

2. Background

2.1. Related Work

The Text Encoding Initiative (TEI) guidelines[4] and their associated XML schema, which we will refer to as TEI-XML in what follows, have become a standard for encoding lexicographic resources as richly annotated, highly machine actionable files in the standard XML format. Moreover, a number of tools and technologies have been proposed to assist in the creation, editing and visualisation of resources in TEI-XML, with a special focus on the use-case of retrodigitizing legacy dictionaries. Amongst these we can cite the machine learning based **GROBID dictionaries tool** (Khemakhem, 2020), (Khemakhem et al., 2017), which has been used in several projects, including those for retrodigitizing Classical Mixtec (Bowers, 2019) and Turkish (Özcan, 2018) dictionaries. The architecture of GROBID dictionaries is organised on the basis of cascading models and requires a small sample of annotated PDF or XML format files to produce a TEI encoding of a dictionary from an OCR'd text. Another, similar, tool which uses machine learning is the **ELEXIFIER** (McCrae et al., 2019) developed within the ambit of the ELEXIS EU horizon project; it too produces a TEI-XML output on the basis of an input OCR'd file and a small sample of annotated pages. As far as we are aware, at the time of writing, neither of these two tools is currently being updated and so they may no longer be very usable.

Adopting a machine-learning based approach for the parsing, extraction and structuring of lexicographic resources obviously has many advantages, and will indeed be the most suitable option in a wide range of situations. However, because of the reliance of such approaches on a certain kind of technical profile and access to computing power/technology – as well as their dependence on what are essentially 'black box' technologies – they may not always be desirable or viable in all situations. Our aim in general, and in the work described in this article in particular, is to test out the viability in specific, often specialised, cases of moving away from a 'black box' approach and towards one that is more 'human friendly' in the sense of being dependent on human readable rules (these can of course be generated by AI and checked by human annotators/modellers and in future work we propose to do just this) and freely available out of the box technologies. Clearly, this approach will work best in cases where the original source text isn't so complicated that the rules become overly complex and difficult to work with, or where it isn't possible to work with formalisms such as Context-Free Grammars due to their (relatively speaking) limited expressivity. For this reason, in the current instance, we have worked with a dictionary that has a fairly straightforward microstructure, even if it is one that is not entirely banal, and which depend upon the kind of skill-set which a lexicographer or linguist will possess (see Section 3.1 for an overview of the work and its structure; this will give a good idea of the kinds of resources for which we propose the application of our approach). Indeed, in our case, the work of developing the parser has been largely carried out by a Digital Humanist who started out without any specific training or expertise in working with CFGs or parsing texts using automated methods but who had an interest in working on encoding dictionaries.

3. Conversion

3.1. Clark-Hall's Concise Anglo-Saxon Dictionary

The particular legacy work with which we will be concerned here is **Clark-Hall's *A Concise Anglo-Saxon Dictionary* (CAS)** subtitled ***For the Use of Students*** (Clark Hall, 1916). It is, as the title suggests, a dictionary of Old English that was specifically intended to help students learn the language (there being a scarcity of accessible materials on the market at the time). Compiled by the Victorian barrister and Old English scholar John R. Clark Hall and published on 1894, it took substantial inspiration (and also reworked material) from an earlier, more scholarly lexicon of Old English compiled by Bosworth and Toller and published in 1838. The CAS is still in print and is currently in its fourth edition. The second edition which was published in 1916 has been made available as an ebook on the Gutenberg website, both in HTML and TXT versions.[5] The lexicon contains 36744 lemmas with

[3] https://github.com/Serg078/Conversion-of-The-Clark-Hall-Dictionary-of-Old-English-to-TEI

[4] In particular, Chapter 10 of the guidelines which are dedicated to dictionaries https://www.tei-c.org/release/doc/tei-p5-doc/it/html/DI.html

[5] https://www.gutenberg.org/ebooks/31543

a substantial number of these being orthographic variants of other lemmas.

We decided that we would adopt the CAS in this edition as a test case for our approach for a number of (largely pragmatic) reasons. The first is fairly obvious: the fact that the CAS was already available as a proof-checked OCR'd file from Gutenberg meant that we didn't have to take care of this part of the process ourselves. Secondly, it remains a useful work for studying Old English, one that over a hundred and thirty years after its publication still hasn't been completely superseded as a resource for students of Old English – which is a strong motivation for making it available as a machine actionable resource using a schema and in a format that is quickly becoming standard for such works, namely TEI-XML. Thirdly, as we discuss below, in Section 3.2, our approach is best adapted to dictionaries that have a fairly simple microstructure associated with most entries; the CFG approach is probably not suitable to dictionaries with extensive nested senses and a more richly organised microstructure – such as for instance the Bosworth-Toller dictionary which was Clark-Hall's inspiration. However, it is far from being the case that parsing a dictionary such as the CAS is therefore without its challenges, as we discuss in the next section.

3.2. TEI modelling of the Clark-Hall dictionary

The main idea of our approach is to produce a parse tree from our source text that can then be transformed into TEI-XML by a simple script. We decided to make the TEI encoding conformant with the TEI Lex-0 customisation of the original TEI (Romary and Tasovac, 2018), intended specifically for encoding dictionaries and lexicographic resources. This was not always straightforward as the TEI Lex-0 documentation does not seem to be complete at the time of writing but we stuck to it as closely as possible. In what follows we describe some of the main challenges of encoding the CAS. Some of these are specific to Old English; many will apply more generally. We have extensively documented the modelling challenges which we have encountered as well as our proposed solutions.[6]

As mentioned above, the microstructure of the CAS is comparatively simple on the surface level (at least in relation to many other dictionaries of Old English, or other 'historical languages' like Ancient Greek or Latin). Each entry generally follows a standard microstructural organisational schema which consists of the following main elements: **headword**, **variant forms**, **basic grammatical information** (part of speech, gender, strong verb class), **definition**, **etymological information** and on occasion: **homonymic entries** and/or **related entries**. Homonymic entries are listed under the same headword and separated by Roman numerals. Additional grammatical information/commentary is also sometimes present, even if as the dictionary's name suggests it tends to be very expressed in a very concise fashion. However there is very little in the way of sense nesting; citations and etymological information aren't very detailed or extensive either (in the case of the latter these are usually just cognates). Nevertheless we encountered many cases where the compactness and conciseness of the dictionary created significant ambiguity for the parser. In these cases, we felt that it would be useful to document our modelling decisions in a way that was complementary to the guidance offered by the TEI Lex-0 or the TEI Guidelines. We will briefly describe some of the challenges we encountered via the analysis of some typical entries. This will help motivate our account of the CFG parser which we describe in Section 4.

3.2.1. Some Typical Entries

The following entry for ***wealh-moru*** is typical of the laconic style of the CAS. In spite of its apparent simplicity – it is simpler than a lot other entries in the same dictionary since it only contains a headword, variant form informartion, some explicit grammatical information pertaining to the gender of the word and simple definition – the entry is fairly rich in lexical information and is indicative of the challenges in writing a parser to process such text and extract lexicographic information from it.

> wealh-moru, -more (a[1]) f., -mora m. carrot, parsnip.

Take for instance the question of variants of the lemma (something which is important in a dictionary for a pre-standardised language such as Old English without a single orthography). The exact number and form of all variants of *wealh-moru*, as well as which forms carry which gender may not be obvious to readers unused to the conventions adopted by Clark-Hall (and these are not always explicitly stated by Clark-Hall himself, or not fully), and needs to be rendered explicit when it comes to writing a parse for the text. In this case the complete list of variants for the lemma form is *wealh-moru*, *walh-more*, *wealh-mora* and we encode this in TEI-XML as:

```
<form type="variant">
  <orth extent="part" expand="walh-
more">
    <seg>-more</seg>
```

[6] https://github.com/Serg078/Conversion-of-The-Clark-Hall-Dictionary-of-Old-English-to-TEI/blob/main/TEI%20choices%20documented.pdf.

```
        (a<lbl rend="sup">1</lbl>)
  </orth>
</form>
```

In order to be able to determine the number of variants in cases such as these, we analysed other entries in which the parenthesised cue happens to be separated by a comma from other variants, as well as by cross checking entries with the information in other Old English dictionaries and resources.[7]

Moving on to look at the representation of grammatical information in the *wealh-moru* entry, we note that the part of speech isn't listed explicitly. In fact this information is largely (but not always) implicit in the CAS. There are abbreviations which help us determine the part of speech in a specific case, e.g., subscript numbers attached to a lemma which give the (strong) verb class, but often we will have to determine this from the ending of the lemma and/or the definition of the entry. The entry for *wealh-moru* is also typical in terms of the complicated way it presents other grammatical information; something that inevitably results in more complicated parsing rules. For instance, we were able to determine that if a headword and its variants are followed by one gender mark in CAS, then that gender applies to the entire block and can be encoded in the entry-level `<gramGrp>`; if multiple genders are presented, then the first gender that is present has scope over the lemma and all variants up to that point; any subsequent gender applies only to the immediately preceding form and is placed inside that variant's `<form>`. If the lemma's gender is explicitly given and at least one more gender is present, that value is encoded inside the lemma's `<form>`, while later gender marks are attached only to the specific forms they follow. Applied to "wealh-moru, -more (aˆ1) f., -mora m.," this means that "f." covers the lemma and the first variant unit, and "m." attaches solely to -mora.

We can see some of the other challenges which the CAS poses to the construction of hand-crafted rule based parsers[8] by looking at another short entry. The entry in this case is for the verb ±*edcwician*:

> ±edcwician (cwyc-, cuc-) to re-quicken, revive, Æ.

Here we see the use of the prefix notation "+" and "±"which is very common in the CAS. Clark-Hall uses the convention that + stands for the prefix ge- and ± in cases when the lemma is found in both prefixed and unprefixed forms. Since neither TEI Guidelines nor TEI Lex-0 prescribe a specific representation for this typographic convention, we encode the expanded form(s) in `@expand` in `<orth>`, keep the symbol in `<lbl>`, and put the base string in `<seg>`:

```
<form type="lemma">
  <orth extent="full"
        expand="edcwician, ge-
edcwician">
    <lbl expand="ge-
_optional">±</lbl>
    <seg>edcwician</seg>
  </orth>
</form>
```

The same pattern is used for + (single ge-... in `@expand`).

In cases when the lemma with *ge-* prefix is followed by variant segments as with ±*edcwician*, the prefix should apply to both variants as well. Both expanded forms are listed, with the optional *ge-* prefix preserved, `<seg>` is used since both variants are segments of the full form:

```
<form type="variant">
  <orth extent="part" expand=
            "edcwycian, ge-
edcwycian"><seg>cwyc-</seg>
  </orth>,
</form>
<form type="variant">
  <orth extent="part" expand=
            "edcucian, ge-edcucian">
    <seg>cuc-</seg>)
  </orth>
</form>
```

This approach retains information about the distribution of ± while making the underlying forms explicit for validation and querying.

Let us now turn to one of the bulkier entries in the CAS, in this case for the anomalous verb *habban*:

> habban anv. ptc. pres. hæbbende; pres. 1 sg. hæbbe, 2. hæfst, 3. hæfð, pl. hæbbað, habbað; pret. 3 sg. hæfde; subj. hæbbe; imperat. hafa; pp. (±) hæf(e)d to 'have,' Bl, G: possess, own, hold, B, Bl, G: keep, retain, Bl, VPs: entertain, cherish: esteem, consider, BH: be subject to, experience, Mk: obtain, Chr, Mt: assert, Lcd: used as auxiliary to indicate past tense, have, Æ, Chr, Ex. h. for consider: (+) hold, keep from, preserve, retain, CP, Æ. yfle +h. afflict, torment.

[7] If "-more (a^1)" is considered a single "unit", then: either both *wealh-more* and *walh-more* count as variants (because "(aˆ1)" indicates the replacement of the first vowel cluster in the base), or only *walh-more* is intended in case "(aˆ1)" applies solely to the first segment when the second segment is *-more*; a third possibility would be that "(aˆ1)" targets the second segment *-more* itself, yielding *-mare*. And if *-more* and "(aˆ1)" apply separately, then *wealh-moru* and *walh-more* would both be valid alternatives.

[8] Such information is also useful for creating a gold-standard for using machine learning based approaches.

Although considerably longer than most entries, its structure remains simple: **headword**; **inflected forms**; **definitions separated by ":"**; **one extended grammatical note**; and **two related entries**. This example highlights three conventions relevant for our parser and the TEI modeling. When more than one definition is present, we encode the separators as

```
<metamark function=
    "senseSeparator">:</metamark>.
```

The dot that closes the main block and precedes nested content (e.g., related or homonymic entries) is distinguished from other dots (e.g., in abbreviations) and encoded as `<metamark>.</metamark>`. Related entries are modeled as separate nested entries: `<entry type="relatedEntry">`. Consider the following part of the entry:

yfle +h. afflict, torment.

. The "+" symbol marks the *ge-* prefix, and "h." here is an abbreviation for the headword *habban*. In the TEI encoding we expand the abbreviation with `<ref type="oRef">` and encode the collocation with `<form type="collocation">`; the related entry itself is nested under main entry:

```
<entry xml:id="yfle_ge-habban"
        type="relatedEntry" xml:lang="ang">
    <form type="collocation">
        <orth>yfle</orth>
        <orth extent="prefix"
            expand="ge-habban">
        <lbl expand="ge-">+</lbl>
        <seg><ref type="oRef">h.
                    </ref></seg>
        </orth>
    </form>
    ...
</entry>
```

Similarly in our encoding, homonymic entries are encoded as separate entries under the main entry. They are introduced by Roman numerals and separated from the preceding material by a terminal metamark, for example:

ǣr- = I. early, former. II. intensitive prefix.

Each homonym receives its own `@xml:id` (e.g., `ǣr-_1`, `ǣr-_2`), with sense identifiers inheriting that prefix (e.g., `<sense xml:id="ǣr-_1.1">`). Roman numerals are captured as `<lbl type="homNum">I.</lbl>`.

Old English also distinguishes strong/weak categories across verbs, adjectives, and nouns. For strong verbs, the superscript indicators (e.g., "±helpan^{3}" or "sv^{3}") are encoded as `<gram type="pos" value="strong verb"/>` `<gram type="inflectionType">` `<hi rend="superscript">3</hi></gram>`; While weak verbs are usually marked "wv.". For adjectives and nouns, explicit weak marking (e.g., "wk gearwa", "pða wm.") is encoded as:

```
<gram type="declension"
        value="weak">wk</gram>
```

As mentioned above a fuller account of these different modelling challenges are available in the associated documentation.[9] The information which we encode using different TEI elements and attributes here should be extracted by the parser. We could have chosen a simpler modelling approach (e.g., without expanding variant forms) – and this would have resulted in a simpler parser – but we wanted to push the envelope somewhat and see the extent to which we could build a parser that would still be (reasonably) human readable and, most importantly, be practical in terms of efficiency.

4. Reliable Conversion with Context-Free Grammars: a Case Study with the CAS

The following section provides is a brief description of the development of our parser. The parser itself is available in our repository, together with the parse trees and TEI output.

4.1. Creating the Parser

This section describes the practical construction of the parser that produces TEI-ready trees from CAS entries. We adopt a rule-based, deterministic pipeline rather than a model-driven one. Every decision is encoded in explicit productions and normalization steps, making behavior predictable, auditable, and reproducible across runs. This design also provides clear failure points that can be traced and fixed, which is crucial for scholarly encoding. As mentioned above, the development work itself was largely carried out by a Digital Humanist without any prior experience in CFGs or automated parsing, therefore, with this approach the emphasis is on transparent rules, reproducible heuristics, and an iterative workflow.

4.1.1. Scope, design goals, parsing

We parse the dictionary entry-by-entry, with each entry handled independently. Entry boundaries in CAS are stable (i.e., in the OCR'd text file), cross-entry dependencies are negligible (each entry is self-contained, so the parser can safely treat every entry in isolation), and isolating failures to a

[9] https://github.com/Serg078/Conversion-of-The-Clark-Hall-Dictionary-of-Old-English-to-TEI/blob/main/TEI%20choices%20documented.pdf.

single entry simplifies debugging, parallelization, and incremental reruns. The main design goals were to ensure that the parser behaved predictably, provided clear, structured error messages indicating where and why a failure occured, and that its output already corresponded to our predetermined TEI conventions, so that the transformer could, where possible, map each element directly without too much additional guesswork or post-processing. The grammar mirrors the microstructure described in Section 3.2. At the top level an `entry` consists of a headword, an optional block of preceding material, a main block (senses, homonymic entries, or a simple cross-reference), a block dedicated to possible complex cross-references (these had to be kept out of `main_content` since they needed their own set of preceding rules) and optional subsequent material. For compactness we show only the spine rules here:

Listing 1: Spine of the Lark grammar

```
?start: entry
entry: headword
       ((preceding_content? main_content)
     | (etym? (poet_symb | ge_pref)*
       form* xr_section+))
       subsequent_content?
main_content: hom_entry+
            | sense_section+
            | simple_xr
```

The grammar (specifically terminals) largely relies on CAS typographic and punctuation conventions which we determined during modeling. *Definitions (senses)* appear in italicized spans (enclosed in underscores) and are separated by ":" (and occasionally ";"); *Homonymic subentries* always begin with Roman numerals. *Attestation sources* begin with a capital letter and may be enclosed in underscores or hash marks. Content we treat as *glosses* consistently appears in parentheses. *References to the OED* are always given in single quotes. *Cross-references* are introduced by fixed labels (e.g., "=", "==", "v.", or patterns such as "headword/variant-grammatical information-of-referenced word"). Most grammar-related content is captured directly via a limited, dictionary-internal set of abbreviations (e.g., "adv.", "n.", etc.), which are documented in CAS and used consistently.

4.1.2. Batch architecture and outputs

The parser is implemented as a batch process that automates the entire workflow from input normalization to TEI conversion. The batch runner accepts a list of entry strings and, for each, performs: light normalization (quotes, NBSP, line breaks), parsing with the Lark[10] grammar (Earley), and – on success – hands-off to the transformer. Each batch produces exactly three files:

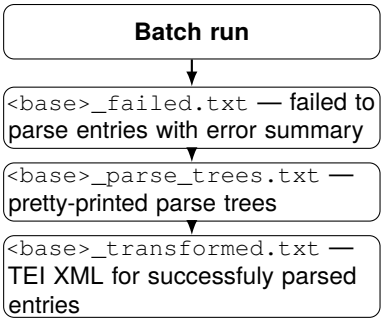

Figure 1: Artefacts produced by each batch run.

This split lets us focus debugging on the failed file, review structure independently of output via the trees, and validate TEI directly. The runner also prints batch statistics (totals, success rate) and timestamps to support regression tracking. The statistics are also printed within each output file. The transformer maps the parsed tree to TEI in a single pass. For each entry it builds a main <*entry*> with a lemma form; expands ge-prefix markers + / ±" (with ±" yielding both forms); reconstructs vowel-slot/partial variants as variant forms with computed expansions; and currently attaches grammatical information at the entry level (inserting *pos="noun"* when gender appears alone). Homonymic entries are outputted as nested entries with stable sense IDs; senses split on ":"/";" and become translation citations; attestations are rendered as <*bibl*> with numeric strings classified into pages, lines, volumes, or folios. Cross-references are output as <*xr*> with a single target, and related one-word or collocational items are produced as nested entries with normalized *xml:id* and resolved lemma abbreviations. Design-wise, printed cues are preserved diplomatically while semantics are normalized in attributes. Known gaps are listed in Section 4.1.4 and will be targeted in the future.

4.1.3. Development workflow

The development began with an extended period of experimentation inside the Lark online IDE.[11] At first we worked exclusively with single entries, writing and adjusting grammar rules one at a time as new structural elements were encountered. Each rule was tested directly against the text of a specific entry, then refined to accommodate the next case. This incremental approach allowed us to build a working grammar gradually, while developing an intuition for the variety of structures in the dictionary. Once the grammar reached a somewhat stable level, we implemented it in a batch

[10] https://github.com/lark-parser/lark

[11] https://www.lark-parser.org/ide/

parser that could process thousands of entries automatically. The standard workflow was to parse all 36000+ entries one thousand entries per batch (though in total only the first 6000 were fully tested during development). After each run, all failed entries were reviewed manually: every parsing error was opened again in the Lark IDE, diagnosed, and corrected through grammar adjustments or new token definitions. Each failure has been logged, categorised, and annotated with a suggested solution for future runs. This error log, combined with examples of both correct and problematic parses, forms part of the documentation available in our github repository. Larger annotation projects can benefit from creating unit-tests that are triggered after each run to make sure that changes in rules did not break parses of previously correctly parsed entries.

Table 1: Batch metrics across development

Milestone	Success (%)	Entries
Initial prototype	70.0	1,000
Post-normalization	89.0	1,000
1st 1,000 batch	90.3	1,000
2nd 1,000 batch	91.4	1,000
3rd 1,000 batch	93.8	1,000
4th 1,000 batch	94.1	1,000
5th 1,000 batch	94.4	1,000
6th 1,000 batch	94.7	1,000
All entries	95.0	36,768

The next stage involved examining the supposedly successful parses – since between one and three percent of them typically contained subtle errors that had passed unnoticed. These were identified and fixed in the following iterations, progressively improving both the coverage and the precision of the parser. Many adjustments were, however, short-lived: rules that solved one ambiguity often introduced another elsewhere and had to be removed or rewritten, thus many previously solved cases would return with the next batch, leading to a potentially never ending cycle of testing and fine-tuning. Despite this, the overall trajectory was steadily upward – from an initial success rate of roughly 70% in the earliest trials to the low-to-mid 90s in the final runs, with the latest full parse reaching approximately 95% success – defined here strictly as entries processed without a fatal parser error, and not accounting for the small percentage of successfully parsed entries that may still contain subtle structural errors. Transformation to TEI would warrant its own success metrics (the current state of that stage is discussed in Section 4.1.4.). The main drawback is diminishing returns beyond the high-accuracy plateau: every extra percentage point demands much finer rules and trickier disambiguation. The remaining, most irregular $\sim 5\%$ is better handled by a guided manual pass – using the error ledger and checklists – rather than adding more grammar complexity.

4.1.4. Transformer Coverage and Known Errors

Even after the grammar reached a high level of stability, a considerable amount of problematic entries remained. These typically represent cases where the printed source itself is inconsistent or where the structure of the entry departs from the otherwise regular patterns of CAS. In several instances, punctuation errors or other assorted inconsistencies in the source text that make automated parsing impossible without prior manual correction. A number of failures also arise from entries that combine otherwise typically separate components – for example, cross-references embedded within definitions, or from extended grammatical commentary (such as "used as auxiliary to indicate past tense" as seen in the *habban* entry for example). At the structural level, some issues persist in how grammatical information attaches to variants and senses. The most frequent among these is the incorrect scoping of gender, where a gender mark that should apply only to a variant is instead attached to the lemma or the entry as a whole. Known transformer-related errors include the misattribution of parenthesized variant segments to the headword rather than the variant, as in *sæt-ere, -nere (ē)*, where the correct result should be a single variant *sēt-nere*, but the transformer produces two separate forms, *sēt-ere* and *-nere*. A similar problem affects the scope of gender marks: when multiple genders are present, as in *biblioðēce f., biblioðeoco (AO), bibliðeca m.*, both lemma and variant genders appear at the entry level instead of under their respective forms. While this could be corrected in the grammar, doing so would make the *gramGrp* rule overly greedy. A transformer-side fix is still being developed. In some cross-references, the ge-prefix marker attached to the referenced word is lost. Related entries following homonymic entries fail to appear, and variants occurring inside a homonymic block are likewise lost. Some of these problems are purely technical and can be fixed through better scoping in the transformer; others require a more complex redesign of how certain grammatical bundles are interpreted. As such, a meaningful evaluation of the transformation stage is not yet feasible in its current state, and will be addressed as part of future development alongside a gold-standard validation effort.

4.1.5. Future curation and outlook

As part of our planned workflow TEI output which results from the process discussed above undergoes manual curation and checking. All automatically produced entries are compared against a set of manually encoded "golden standard" examples that represent the intended structure and interpretation of complex cases. These examples are already included in the encoding file accompanying the github repository and serve both as reference material and as a regression test for future development. Although the current parser achieves a high success rate, further work will concentrate on refining both the grammar and the transformer to eliminate remaining errors and to extend the coverage to still unresolved major components such as OED references, etymologies, usage; various finer-grained features (such as verb inflection type, sources within variant forms or cross-references, and many more), and countless small scale variations that are already successfully parsed but are not yet transformed. Manual encoding of the most irregular entries, which constitute roughly five percent of the total, will also be required and will serve as an important source of feedback for further refinement. The long-term goal is to reach a fully validated, machine-readable TEI version of the entire CAS, in which both automatically and manually encoded entries conform to the same consistent model.

5. Extending the TEI with RDF Annotations

5.1. RDFa Modelling

In order to further enrich the TEI version of CAS, we decided to add machine-readable semantic information as RDF. Following the recommendations in (Khan et al., 2024), we opted to use RDFa, *Resource Description Framework in Attributes*, a W3C Recommendation for extending HTML/XML markup with RDF triples as attributes of existing elements (Adida et al., 2015).

RDFa provides several attributes that can be included in XML elements to construct RDF triples. Most notably, `resource` and `about`, which set the context (i.e. current subject), `property` and `rel`, which correspond to an RDF predicate that should be used in a generated triple, and `typeof`, which creates a statement about the type of the subject. The object is often defined by the tag content, e.g.:

Listing 2: RDFa encoding of an entry *wealh-moru*

```
<entry xml:id="wealh−moru"
    resource="wealh−moru_entry"
    typeof="ontolex:LexicalEntry">
    <form type="lemma"
        rel="ontolex:canonicalForm"
        resource="wealh−moru_form"
        typeof="ontolex:Form">
        <orth
          property="ontolex:writtenRep">wealh−moru</orth>
    </form>
    ...
    <gramGrp about="wealh−moru_entry">
      <gram type="pos" value="noun"
        rel="lexinfo:partOfSpeech"
        resource="lexinfo:noun" />
      <gram
        type="gender" value="masculine"
        rel="lexinfo:gender"
        href="lexinfo:masculine">m.</gram>
    </gramGrp>
    <sense xml:id="wealh−moru.1"
        resource="wealh−moru_sense−1"
        rel="ontolex:sense"
        typeof="ontolex:LexicalSense">
      <cit type="translation" xml:lang="en">
        <quote property="rdfs:label">carrot, parsnip</quote>
      </cit>
</entry>
```

To model the dictionary in RDF the obvious choice was to use OntoLex-Lemon, a widely used standard for representing lexicographic information as Linked Data (McCrae et al., 2017). For the version presented with this paper, we only modeled lexical entries, forms, senses and basic grammatical information, leaving the rest for future work. The correspondence between TEI elements and OntoLex classes is given in Table 2.[12]

Table 2: Mapping between TEI and OntoLex

Element	XML	OntoLex
Lexical entry	`<entry>`	LexicalEntry
Form	`<form>`	Form
String representation	`<orth>`	writtenRep
Sense	`<sense>`	Sense
Sense label	`<quote>`	rdfs:label
Grammar	`<gram>`	lexinfo:*

This partial representation does not add any information that was not present in the TEI, but it links it to external ontologies (i.e. OntoLex and Lexinfo) which contextualises it and allows users to use off-the-shelf RDF technologies, such as SPARQL to query the data in a federated way, combining it with other resources for Old English in RDF.

5.2. Accessing the Data

There exists a variety of technologies and open source tools that can extract RDF triples from TEI XML with RDFa annotations,[13] which, in turn, can be queried with SPARQL. However, by itself, this does not provide a user-friendly way for digital humanists to search and navigate the data. To do so,

[12] Full mapping with paths and attributes is available in the repository.

[13] See `https://rdfa.info/tools` for an overview.

while keeping the solution within the framework of minimal computing which we have adopted, we created a static web site[14] using the VitePress-SPARQL plugin,[15] that allows adding SPARQL queries to Markdown files that can be executed against several endpoints including remote files with RDF triples. Together with CETEIcean,[16] a JavaScript library that can render TEI files to HTML, this provides a lightweight solution for viewing, navigating and querying TEI+RDF data.

6. Conclusions

In this paper we have presented a workflow for creating TEI editions of legacy dictionaries using hand-crafted context-free grammars. This deterministic approach guarantees predictable results and simplifies conversion workflows, allowing parallelisation and quality control with unit-testing. In addition, we outlined how to enrich the TEI source with RDFa annotations to provide interoperability with Linked Data resources, and presented a way to publish the dictionary in a user-friendly and user-facing way.

Ethics Statement

This work digitizes and structurally encodes a historical lexicographic resource. The data do not contain personal or sensitive information. Ethical considerations primarily concern provenance and responsible reuse of historical content. Our work draws on the 1916 edition of Clark Hall's *Concise Anglo-Saxon Dictionary*, made available via Project Gutenberg, which states that the eBook may be copied and re-used under the terms of the Project Gutenberg License. We further note that the dictionary remains in print in later editions, and we therefore distinguish the public-domain source used here from subsequent published editions. Finally, historical dictionaries may reflect outdated terminology or perspectives, so we present the resulting encodings as research artefacts and report known sources of errors and limitations so that outputs are not treated as definitive without validation and, where appropriate, human review.

Limitations

While the pipeline presented in this paper demonstrates a viable approach to the retrodigitization of legacy lexicographic resources, several limitations should be acknowledged. The grammar and parser have been developed specifically for the Clark Hall Dictionary of Old English, and adapting the approach to other dictionaries would require substantial rework, as entry formats and conventions vary widely across lexicographic traditions. The use of a context-free grammar also imposes inherent expressiveness limits. Highly ambiguous or context-dependent structures may not be fully resolvable within a CFG framework. Despite the approximately 95% parse success rate, the pipeline is not fully automatic in practice. Failures and silent errors within successfully parsed entries still require manual curation at this stage. Finally, as discussed in Section 4.1.4, the transformation to TEI is still incomplete, meaning the output is not yet suitable for direct use without further curation. The RDF component likewise remains under development, which limits the extent to which the pipeline can currently be considered fully end-to-end.

Acknowledgements

This work was funded by the H2IOSC Project - Humanities and cultural Heritage Italian Open Science Cloud funded by the European Union NextGenerationEU - National Recovery and Resilience Plan (NRRP) - Mission 4 "Education and Research" Component 2 "From research to business" Investment 3.1 "Fund for the realization of an integrated system of research and innovation infrastructures" Action 3.1.1 "Creation of new research infrastructures strengthening of existing ones and their networking for Scientific Excellence under Horizon Europe" - Project code IR0000029 - CUP B63C22000730005. Implementing Entity CNR. It was also supported by CLARIN-IT and by the I+D+i projects PID2024-159530OB-I00 (funded by MCIN/AEI/10.13039/501100011033) and UZ2024-IyA-02 (funded by Univ. Zaragoza).

7. Bibliographical References

Ben Adida, Mark Birbeck, Shane McCarron, and Ivan Herman. 2015. RDFa Core 1.1. Syntax and processing rules for embedding RDF through attributes. Technical report, W3C Recommendation. Third edition.

Jack Bowers. 2019. TEI encoding of a classical Mixtec dictionary using GROBID-Dictionaries. *HAL (Le Centre pour la Communication Scientifique Directe)*.

John R Clark Hall. 1916. *A concise Anglo-Saxon*

[14] https://clark-hall-tei-rdfa.netlify.app/

[15] https://github.com/max-ionov/vitepress-plugin-sparql.

[16] https://teic.github.io/CETEIcean/.

dictionary: for the use of students, second edition. Swan Sonnenschein & Company.

Fahad Khan, Maxim Ionov, Christian Chiarcos, Laurent Romary, Gilles Sérasset, and Besim Kabashi. 2024. On modelling corpus citations in computational lexical resources. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 12385–12394, Torino, Italia. ELRA and ICCL.

Mohamed Khemakhem. 2020. *Standard-based lexical models for automatically structured dictionnaries*. Ph.D. thesis, Université Paris Cité.

Mohamed Khemakhem, Luca Foppiano, and Laurent Romary. 2017. Automatic extraction of TEI structures in digitized lexical resources using conditional random fields. In *electronic lexicography, eLex 2017*.

John P McCrae, Julia Bosque-Gil, Jorge Gracia, Paul Buitelaar, and Philipp Cimiano. 2017. The Ontolex-Lemon model: development and applications. In *Proceedings of eLex 2017 conference*, pages 19–21.

John P McCrae, Carole Tiberius, Anas Fahad Khan, Ilan Kernerman, Thierry Declerck, Simon Krek, Monica Monachini, and Sina Ahmadi. 2019. The elexis interface for interoperable lexical resources. In *Proceedings of the sixth biennial conference on electronic lexicography (eLex)*. eLex 2019.

Emrah Özcan. 2018. Retro-digitizing turkish dictionaries using GROBID-Dictionaries. In *Lexical Data Masterclass Symposium*.

Laurent Romary and Toma Tasovac. 2018. TEI Lex-0: A target format for TEI-encoded dictionaries and lexical resources. In *TEI Conference and Members' Meeting*.