Case study: Preparing a TEI corpus for Canonical Text Services

Chris Becker, Jochen Tiepmar, Gerhard Heyer
University of Leipzig, Germany

September 30, 2016

Abstract

The CTS protocol offers features for distinct citations in text corpora. Additional examples were needed to further test the implementation and platform the University of Leipzig provides. In this paper we will both show a list of freely available works as well as a case study concerning the conversion of a TEI-encoded corpus for CTS.

1 Introduction

The vast majority of digitally available publications follow one of the well-established formats, e.g. the TEI (Text Encoding Initiative [1]). However, there exist protocols that further enhance the capability to process text resources. One of those is CTS (Canonical Text Services [2]). With CTS, it is possible to quote arbitrary passages of text - independent of pages, lines or sentences.

In order to do this, URNs (Unique Resource Name) are used to represent the hierarchical structure of a document. For instance, the URN

urn:cts:demo:shakespeare.sonnets.en.1:1.1

is a URN of type CTS in the namespace demo and refers to the first verse of the first sonnet in the first (English) edition of Shakespeare's sonnets. By leveraging URNs like this it is possible to quote passages down to specific words.

The University of Leipzig maintains a website dedicated to CTS [3] with a set of applications dedicated to browsing, reading and testing CTS instances.

2 Corpora

This section explains the search methodology and goes into detail about the different characteristics that are important for the transition to CTS. There is also a showcase for one of the results.

2.1 Search methodology

The task was to find text corpora which were not previously integrated into the CTS portal of Leipzig University. The primary goal was to find texts that can be integrated into the CTS repository without requiring a complete overhaul of the raw data. For this reason, we put emphasis on TEI-encoded data.

Another goal was to find concluded corpora and not content that was in the process of being created (social media would fall into the latter category).

The search itself can be best described as a two-pronged approach: On the one hand, existing collections of corpora were identified and the respective entries inspected with regards to their suitability. On the other hand, prevalent search engines were used to look for single corpora that might not show up in any pre-collected list.

Results mostly originated from the academical sector. Good collections for getting started as a beginner in the field can be found on the sites of the TEI foundation [4] and the HU Berlin [5]. Some promising candidates from the industry were found, but they did not fit our use case. A more detailed look into the reasons for this will be given in the next subsection.

2.2 Characteristics

There are several criteria that have to be considered for our use case. For instance, the texts should be available in plain text, as converting them from pdf or images would require more work than just being able to annotate the texts in preparation for CTS directly.

The major characteristics are:

- Structure of the files: How big is the percentage of metadata in comparison to the textual contents of the document?
- Licensing: Is the corpus free to use for other (in our case academic) purposes?
- Access policy: Is the content freely available? How much freedom does the search query interface provide?

- Download availability: Is there a dump available? Do we need to extrace html out of a website?

An initially promising find was the database of the "Truth Tobacco Industry Documents" [6]. In contrast to the mostly literature-based findings, this corpus contains advertisements targeted at costumers of the tobacco industry from several decades. Unfortunately, initial random sample tests have shown that the documents are mostly pdf only, without a plain text version. As such, the corpus was not what we were looking for.

A more extensive list of the searched corpora can be found in appendix A.

3 Case Study

As an exemplary corpus the project "Briefe und Texte aus dem intelektuellen Berlin um 1800[1]" [7] was chosen (primarily because it follows the TEI 5 guidelines and also for its manageable size which allows for quick validation of results). Apart from that, the corpus also fits the other criteria for the task really well, because it is licensed under the CC BY 3.0 license and it is readily available as a dump.

The website has an application for viewing high resolution scans of the original documents as well. As a technology the PowerShell scripting language was chosen for its ease of use. Originally a language exclusively for Microsoft operating systems, it has recently been open sourced [8].

3.1 Necessary steps

The first important step in preparing the TEI encoded corpus is the creation of a directory structure. CTS expects a three-level directory structure to follow its URN structure. For this case, the chosen structure looked like this:

```
$author / $title / $language
```

In order to do this, the three terms author, title and language had to be retrieved from the sources. This was achieved by using regular expressions. One of the difficulties was that several of the documents are formatted differently from the rest which was resolved by using two regular expressions, one for each annotation type.

There were also some special cases where author, title or date of the publication were missing or for some reason not extracted by the RegEx. This

[1] Letters and texts: Intellectual Berlin around 1800

only affected a handful of examples so manual changes were a suitable solution. In two cases, the resulting file name would have been too long for the development system (Win7) to create the file name, so it was reasonably shorted.

In a next step, missing metadata was added that some of the documents were lacking. This included the download source for the respective document and that for the dump as well as the publication date and the author information. Most of the documents in the corpus are letters which contain the author information inside a

```
<correspAction type="sent">
```

tag, which was copied into a porper author tag.

Lastly, any special characters were cleaned from the file names and replaced by underscores.

Appendix B contains the script used to prepare the corpus for CTS integration to allow for a complete source code inspection. The result of the preparation can be found online[2].

4 Conclusion

The general résumé is positive. It is possible to prepare a TEI-encoded corpus for CTS with relative ease. Problems arise frequently because the annotation differs from file to file, even if they are all part of the same corpus. This can in part be attributed to the fact that the annotation for the example corpus has been done by different persons that each had another preference in annotating the documents.

4.1 Challenges

Several key challenges rose up in the process of solving the task. Regarding the discovery of corpora there is no amount of central repositories that cover even a considerable majority of all available text corpora.

In fact most of the projects rely on their own websites, even sites that maintain a collection of one form or another can only count as glimpses compared to the stand-alone collections.

The determination of the characteristics of a single corpus turned out to be a labour intensive and time consuming task, once again caused by the heterogeneous nature of the corpus landscape on the web. If every site looks

[2] cts.informatik.uni-leipzig.de/teidumps/berlinletters.zip

different, all of them have to be carefully checked individually to retrieve the wanted characteristics.

Other facts worth considering are:

- Crawling: Be wary of query limits, website changes or javascript hiding content
- Licensing: Many older projects dating back to the 1990's do not have any licensing information present on their websites
- For some projects there is a lack of crucial information like missing licensing information which makes the corpus ineligible
- Some corpora are only accessible after a sign-up account creation. These were not tested in this paper because one of the goals was to use corpora that are freely available.

The manual search with different prominent engines certainly leaves room for improvements when it comes to finding more and/or better results.

It can be concluded that most corpora cover narrow topics based on literature, e.g. using the works of a single author or medium. There are no perfect fits for our use case but we managed to find enough that are usable with a little effort.

A future endeavour in this field might also delve into a half-automatic search for corpora in large online hosted repositories like GitHub.

References

[1] *The TEI Project*. 2016. URL: `http://www.tei-c.org/Activities/Projects` (visited on 06/04/2016).

[2] *Specification of the Canonical Text Services Protocol*. 2013. URL: `https://github.com/cite-architecture/cts_spec` (visited on 06/04/2016).

[3] *Canonical Text Service at Leipzig University*. 2016. URL: `http://cts.informatik.uni-leipzig.de/Canonical_Text_Service.html` (visited on 09/29/2016).

[4] *Projects Using the TEI*. 2016. URL: `http://www.tei-c.org/Activities/Projects/` (visited on 09/28/2016).

[5] *Korpora - Korpuslinguistik und Morphologie*. 2016. URL: `https://www.linguistik.hu-berlin.de/de/institut/professuren/korpuslinguistik/links/korpora_links` (visited on 09/28/2016).

[6] *Truth Tobacco Industry Documents*. 2016. URL: https://www.industrydocumentslibrary.ucsf.edu/tobacco/? (visited on 09/29/2016).

[7] *Letters and texts Intellectual Berlin around 1800*. 2015. URL: http://www.berliner-intellektuelle.eu/ (visited on 07/05/2016).

[8] *PowerShell for every system!* 2016. URL: https://github.com/PowerShell/PowerShell (visited on 09/29/2016).

Appendix A: Overview of corpus results

Corpus	Url
African Languages Lexicon	http://www.edd.uio.no/allex/
African American Women Writers of the 19th century	http://digital.nypl.org/schomburg/writers_aa19/
American Verse Project	http://quod.lib.umich.edu/a/amverse/
Austrian Academy Corpus	http://www.aac.ac.at/index.html
C4 Korpus	http://www.korpus-c4.org/index.php/de/
CELT - Corpus of Electronic Texts	http://www.ucc.ie/celt/index.html
Letters and texts: Intellectual Berlin around 1800	http://tei.ibi.hu-berlin.de/berliner-intellektuelle/?de
Project Gutenberg	https://www.gutenberg.org/
Public Library of Science Journals	http://journals.plos.org/
Sumerian Literature	http://etcsl.orinst.ox.ac.uk/
The Digital Walters	http://thedigitalwalters.org/
Truth Tobacco Industry Documents	https://industrydocumentslibrary.ucsf.edu/tobacco

Appendix B: Source code for preparation script

```
#input of raw data
$inputData = "C:\tmp\berliner-intellektuelle-manuscripts\raw\"

#target output directory
$outputData = "C:\tmp\berliner-intellektuelle-manuscripts\target\"

#convert every document in the target input directory
Get-ChildItem $inputData | % {

#resetting values between files
$author = $null
$date = $null
$lang = $null
$title = $null
$urlSource = $null

#necessary regex
$authorRegex = '(?s)<author>.+?(?=<persName)<persName\sref="#p\d\d\d\d">(.*)<\/persName>.+?(?=<\/author)<\/author>'
$briefRegex = '(?s)<correspAction\stype="sent">.+?(?=<persName)<persName\sref="#p\d\d\d\d">(.{5,50})<\/persName>'
$dateRegex = '<docDate\swhen="(1\d\d\d-?\d?\d?-?\d?\d?)"'
$letterDateRegex = '(?s)<correspAction\stype="sent">.+?(?=<date)<date\swhen-iso="(1\d\d\d-?\d?\d?-?\d?\d?\/?\d?\d?|\d\d\d\d\/\d\d\d\d|1\d\d\d-?\d?\d?-?\d?\d?\/?1\d\d\d-?\d?\d?-?\d?\d?)"\/>.+?(?=<\/corr)<\/correspAction>'
$langRegex = '<language\sident="(\w\w)"'
$titleRegex = '(?s)<titleStmt>\s+<title.{0,20}>(.{5,200})<\/title>'

#retrieve file content
$file = Get-Content ("$inputData" + $_.Name) -Encoding utf8 | Out-String

#author
```

```
29 if ($_.Name -ne "Boeckh_Buchkatalog.xml") {
30 if ($file -match "$briefRegex") {
31 $author = ( [Regex]::Escape($matches[1]), '' )
32 } elseif ($file -match "$authorRegex") {
33 $author = ( [Regex]::Escape($matches[1]), '' )
34 }
35 }
36 $author = $author -replace '\\',''
37 $author = $author.Trim()
38
39 #date
40 if ( $file -match "$dateRegex" ) {
41 $date = ( [Regex]::Escape($matches[1]), '' )
42 } elseif ($file -match "$letterDateRegex" ) {
43 $date = ( [Regex]::Escape($matches[1]), '' )
44 }
45
46 $date = $date -replace '\\',''
47 $date = $date.Trim()
48
49 #language
50 if ( $file -match "$langRegex" ) {
51 $lang = ( [Regex]::Escape($matches[1]), '' )
52 }
53 $lang = $lang -replace '\\',''
54 $lang = $lang.Trim()
55
56 #title
57 if ( $file -match "$titleRegex" ) {
58 $title = ( [Regex]::Escape($matches[1]), '' )
59 }
60 $title = $title -replace '\\r|\\n|\\t|\\|<lb/>',''
61 $title = $title -replace '\s+',' '
62 $title = $title.Trim()
63
64 #sometimes hard-coding easier than complicating the
    RegEx
65 switch ($_.Name){
66 Boeckh_Buchkatalog.xml {$author="August Boeckh";
    $title="Katalog meiner Bücher"; $date="unknown";
    break}
67 Brief002InnenministeriumanBoeckh.xml {$title="Preuß
    Innenministerium Kultus an August Boeckh 3.9.1812";
     $author="Preußisches Innenministerium, Sektion für
```

```
        Kultus und öffentlichen Unterricht" ; break}
68  Brief003ChamissoandeLaFoye.xml {$title="Blumen lese
        von 1804"; $date="1804"; break}
69  Brief003fakultaetssitzung.xml {$title="Brief von
        Friedrich Wilken an die Philosophische Fakultät
        (17. Dezember 1817)"; break}
70  Brief005InnenministeriumanBoeckh.xml {$title="Preuß
        Innenministerium Kultus an August Boeckh
        28.5.1812"; $author="Preußisches Innenministerium,
        Sektion für Kultus und öffentlichen Unterricht" ;
        break}
71  Brief006KultusministeriumanBoeckh.xml {$author="
        Preußisches Kultusministerium, Unterrichtsabteilung
        " ; break}
72  Brief008KultusministeriumanBoeckh.xml {$author="
        Preußisches Kultusministerium, Unterrichtsabteilung
        " ; break}
73  Brief27TieckanRaumer.xml {$title="Brief von Ludwig
        Tieck an Friedrich von Raumer (Dresden, 4. Januar
        1834)" ; break}
74  IHFichte_Die_Aufgabe_der_Philosophie.xml {$title="Über
        den Begriff der Philosophie pp."; break}
75  Reinbold_Literaturbrief.xml {$date="unknown"; break}
76  Roxane.xml {$date="1789/1790/1791"; break}
77  }
78
79  Write-Host "Dateiname: $_"
80  #Write-Host "Autor: $author"
81  #Write-Host "Datum: $date"
82  #Write-Host "Sprache: $lang"
83  #Write-Host "Titel: $title"
84
85  #add missing labels for metadata
86  $urlSource = '<idno type="URLWeb">http://www.berliner-
        intellektuelle.eu/manuscript?' + ($_ -replace ".xml
        ","") + '</idno>'
87  $publication = '<date type="publication">' + $date +
        '</date>' + "`r`n"
88
89  if ($file -like "*<titleStmt>*" -and $_.Name.
        StartsWith("Brief")) {
90  $file = $file -replace "<titleStmt>", ("<titleStmt>`r`
        n<author>`r`n" + $author + "`r`n</author>")
91  }
```

```
92
93 if ($file -like "*</publicationStmt>*") {
94 $file = $file -replace "</publicationStmt>",(
      $publication + $urlSource + "`r`n</publicationStmt
      >")
95 }
96
97 #filter special characters from file names
98 $author = $author -replace
      '\\|\/|\||\<|\>|\"|\?|\*|\:|\.|\(|\)|,| ',"_"
99 $title = $title -replace
      '\\|\/|\||\<|\>|\"|\?|\*|\:|\.|\(|\)|,| ',"_"
100 $title = $title -replace 'wahrscheinlich',''
101 $lang = $lang -replace
      '\\|\/|\||\<|\>|\"|\?|\*|\:|\.|\(|\)|,| ',"_"
102
103 $author = $author -replace "_+","_"
104 $title = $title -replace "_+","_"
105 $lang = $lang -replace "_+","_"
106
107 $author = $author.TrimEnd('_')
108 $title = $title.TrimEnd('_')
109 $lang = $lang.TrimEnd('_')
110
111 #Three level directory structure
112 if (Test-Path $outputData) {
113 #create folders
114 New-Item -ItemType Directory -Force -Path ("
      $outputData" + "$author\$title\$lang" ) | Out-Null
115
116 #create line.xml
117 New-Item -ItemType File -Force -Path ("$outputData" +
      "$author\$title\$lang\line.xml" ) -Value $file |
      Out-Null
118 }
119 }
```