Less destructive cleaning of web documents by using standoff annotation

Maik Stührenberg
Institut für Deutsche Sprache / Mannheim, Germany
maik@xstandoff.net

Abstract

Standoff annotation, that is, the separation of primary data and markup, can be an interesting option to annotate web pages since it does not demand the removal of annotations already present in web pages. We will present a standoff serialization that allows for annotating wellformed web pages with multiple annotation layers in a single instance, easing processing and analyzing of the data.

1 Introduction

Using web pages as primary data for linguistic corpora often includes the procedure of cleaning and normalizing the files. Tools such as POS taggers and linguistic parsers often require the input data to be raw text, that is, without any markup at all. In addition, adding markup layers on top of an already annotated file (such as an XHTML page) often results in markup overlaps – violating XML's wellformedness constraints (Bray et al., 2008).[1]

Since the original version of the web page is the origin of every further processing, we save this version unaltered. We call this version the "raw data". As a next step we create a primary data file containing all textual information but no annotation as input for the before-mentioned linguistic processing tools.[2] Every output of a processing step is stored in a separate folder, making each step of the pipeline reproducible. However, if we want to compare multiple annotation layers, it is preferable to not have to deal with a couple of files stored in a large number of folders. To combine both the original HTML annotation and additional annotation layers, standoff annotation can be an interesting option.

2 Standoff annotation

Standoff annotation is the separation of primary data and markup. The concept as such is not new at all, and there are several reasons to use this approach such as read-only primary data (which is the case as well when dealing with non-textual data) or copyright restrictions. Stührenberg and Jettka (2009) discuss some existing serialization formats, including XStandoff (XSF), which we will use in this paper to demonstrate its ability to process pre-annotated documents. An XStandoff instance roughly consists of the `corpusData` root element, underneath zero or more `primaryData` elements, a `segmentation`, and an `annotation` element can occur, amongst others – see Figure 1 for a graphical overview.

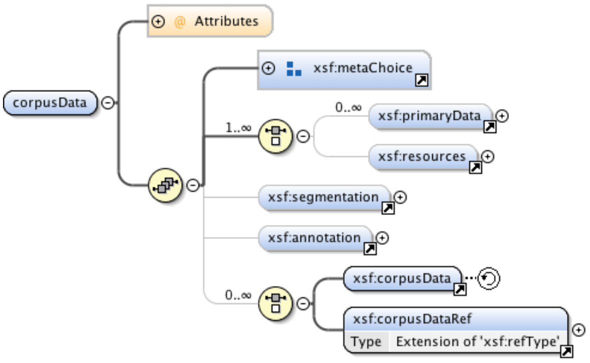

Figure 1: A graphical overview of XStandoff's root element

The two latter elements define two base constructs of standoff annotation formats: (1) the identification of regions of primary data (called segments in XStandoff) used as anchors for one or more annotations, and (2) the way in which annotations are stored.

[1] The discussion of this issue goes back to the days of SGML, including a large number of proposals for supporting overlapping markup not cited here due to space restrictions.

[2] Of course, this is only necessary, if the tool in question does not support pre-annotated input files.

Felix Bildhauer & Roland Schäfer (eds.), *Proceedings of the 9th Web as Corpus Workshop (WaC-9) @ EACL 2014*, pages 16–21,
Gothenburg, Sweden, April 26 2014. ©2014 Association for Computational Linguistics

2.1 Segmentation

In case of *textual primary data* such as web pages, segments can be identified by delimiting the character stream by means of tokenization methods (for example by splitting text into a stream of characters).

```
  T  h  i  s     i  s     a     w  o  r  d
00|01|02|03|04|05|06|07|08|09|10|11|12|13|14
```

The serialization in XStandoff can be seen below. In this example, we have selected the character span ranging from "0" to "4", resulting in the selection of the word "This".[3]

```
<segment xml:id="seg_text1" primaryData="txt" type="
    char" start="0" end="4"/>
```

Since web pages consists of (Unicode) characters as well, it is possible to treat the markup as part of the character stream and in fact, this was the only way to segment primary data in XStandoff version 1 (and its predecessor SGF). However, this mechanism can be error-prone when using pre-annotated primary data because of the white space handling in XML. In this case, it is more promising to use the element node tree of an existing annotation as an initial traversal for the selection of the respective textual part. As an example we use a (valid) XHTML file, from which the first div element is selected by using an XPath 2.0 (Berglund et al., 2010) expression (the example can be seen in Listing 1 in Section 2.2). [4]

```
<segment xml:id="seg_html1" primaryData="pd1" target
    ="xhtml:html/xhtml:body/xhtml:div[1]"/>
```

This approach is limited to work on XML instances only, that is, documents that are at least well-formed according to the XML specification, including XHTML files and those HTML5 pages that use the XHTML syntax, see Chapter 9 of the HTML5 spec (Berjon et al., 2014). Since the larger part of the World Wide Web does not fulfill this requirement, tools such as TagSoup[5] or HTML Tidy[6] can be used to pre-process those web pages. This cleaning process is less aggressive since in most cases it only results in changes of the structural markup and since we have already saved the file in its original form, destructive changes can be detected afterwards.

2.2 Annotations

Standoff annotations may be stored in the same or a different file. XStandoff, as an integrated serialization format, not only combines segmentation and all annotation layers in a single instance, but sticks as close as possible to the original inline annotation format. Element and attribute names remain unchanged as well as the tree-like structure of the element nodes. Textual element content is deleted since it can be referenced via the corresponding segment, and additional attributes are added. The converted annotation layer is stored underneath one of XStandoff's layer elements.[7] The document grammar (defined by an XSD 1.1 schema file) does not require the subtree underneath the layer element to be valid (by using the value *lax* for the processContents attribute of the xs:any element wildcard), but is has to meet the well-formedness constraints defined in the XML specification.

Using the simple XHTML page shown in Listing 1 as primary data, we can select parts of the sentence with XPath 2.0 expressions – for example, the noun phrase (and the pronoun) "This" is selected by the expression xhtml:html/xhtml:body/substring(xhtml:div[1],1,4) using the substring() function (Malhotra et al., 2010).

Listing 1: Example XHTML page

```
<html xmlns="http://www.w3.org/1999/xhtml">
 <head><title>Instance</title></head>
 <body><div>This is a word.</div></body>
</html>
```

Listing 2 shows the XStandoff instance using this XHTML page as primary data. As an annotation layer, we have added a partial POS annotation (including sentence boundary detection).

Listing 2: XStandoff instance with XHTML primary data and POS annotation

```
<corpusData xml:id="c1" xmlns="http://www.xstandoff.
    net/2009/xstandoff/1.1"
  xmlns:xsf="http://www.xstandoff.net/2009/xstandoff
      /1.1">
  <primaryData xml:id="p1">
    <primaryDataRef uri="instance.html" mimeType="
        application/xhtml+xml" encoding="utf-8"/>
```

[3] The optional primaryData attribute's value refers to the corresponding primary data file via XML ID/IDREF identity constraints ((in case of multiple primary data files – in the example to the id "txt", not via a URI. It does not provide any hint about its MIME type, this information is stored in the respective primaryData element shown in Listing 2.

[4] Apart from XPath, the XPointer specification defined in DeRose et al. (2002a; 2002b) and used in XCES (see (Ide et al., 2000) and Section 5) would be another option. However, since XPointer support is very sparse, XPath is a more natural fit.

[5] See http://ccil.org/~cowan/XML/XML/tagsoup/ for further details.

[6] See http://tidy.sourceforge.net/ for further details.

[7] XML Namespaces (Bray et al., 2009) are used to differentiate between XStandoff's markup and foreign markup.

```
 </primaryData>
 <segmentation>
  <segment xml:id="seg1" target="xhtml:html/
     xhtml:body/xhtml:div[1]"/>
  <segment xml:id="seg2" target="xhtml:html/
     xhtml:body/substring(xhtml:div[1],1,4)"/>
  <!-- [...] -->
 </segmentation>
 <annotation>
  <level xml:id="pos">
   <layer>
    <s xmlns="http://www.xstandoff.net/pos"
       xsf:segment="seg1">
     <np xsf:segment="seg2">
      <pron xsf:segment="seg2"/>
     </np>
     <!-- [...] -->
    </s>
   </layer>
  </level>
 </annotation>
</corpusData>
```

Additional annotation levels and layers (see Witt (2004) for a discussion about the distinction of levels and layers) can be added any time. Since XStandoff supports not only multiple annotation layers but multiple primary data files as well, there are two alternative XSF representations possible, if we extract the written text from the XHTML file and use it as primary data file: (1) The TXT file is used as additional primary data file (and serves as input for other linguistic annotation tools, see Listing 3); (2) the TXT file serves as the single primary data file and both the XHTML and the POS annotation are stored as annotation levels and layers. For the second option it is again necessary to pre-process the XHTML file with the already mentioned tools.

Listing 3: XStandoff instance with two primary data files and POS annotation

```
<corpusData xml:id="c1" xmlns="http://www.xstandoff.
   net/2009/xstandoff/1.1"
 xmlns:xsf="http://www.xstandoff.net/2009/xstandoff
    /1.1">
 <primaryData xml:id="p1">
  <primaryDataRef uri="instance.html" mimeType="
     application/xhtml+xml" encoding="utf-8"/>
 </primaryData>
 <primaryData xml:id="txt">
  <primaryDataRef uri="instance.txt" mimeType="text
     /plain" encoding="utf-8"/>
 </primaryData>
 <segmentation>
  <segment xml:id="seg1" primaryData="p1" target="
     xhtml:html/xhtml:body/xhtml:div[1]"/>
  <segment xml:id="seg2" primaryData="p1" target="
     xhtml:html/xhtml:body/substring(xhtml:div
     [1],1,4)"/>
  <!-- [...] -->
  <segment xml:id="seg_txt1" primaryData="txt"
     start="0" end="4"/>
 </segmentation>
 <annotation>
  <level xml:id="pos">
   <layer>
    <s xmlns="http://www.xstandoff.net/pos"
       xsf:segment="seg1">
     <np xsf:segment="seg2">
      <pron xsf:segment="seg2␣seg_txt1"/>
     </np>
     <!-- [...] -->
    </s>
   </layer>
  </level>
 </annotation>
</corpusData>
```

Figure 2 shows the three possible representations.

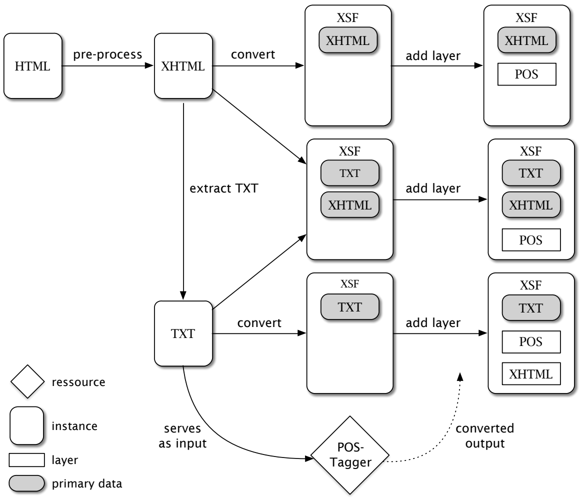

Figure 2: Possible XStandoff instances

3 Creating XStandoff instances

It it cumbersome to create XStandoff instances by hand due to its separation of primary data and annotation. In addition, most annotation tools create inline instances and can only use raw text as input files. Therefore, we have created a set of XSLT 2.0 transformation stylesheets (the XStandoff Toolkit) that allow for the easy conversion between an inline XML instance (containing a single annotation layer) to a single-layered XStandoff instance, and the merging of XStandoff instances over the very same primary data.

The XSLT stylesheet `inline2xsf` requires an input XML file ideally containing elements bound by XML namespaces since XStandoff uses XML namespaces for the layer separation (if no namespace is present, it will be generated). The process of converting an inline annotation to XSF is divided into two steps: After segments are built on the basis of the elements and the character stream of the underlying primary data, the annotation layer is produced by converting the former inline annotation and linking its elements to the according segments by `ID/IDREF` binding.

After at least two inline annotations have been transformed to single-layered XStandoff instances, it is possible to merge those into a single file. Due to the frequent use of the

ID/IDREF mechanism in XStandoff for establishing connections between segment elements and the corresponding annotation, manually merging of XStandoff files is quite unpromising. The mergeXSF XSLT stylesheet converts two XSF instances into a single one containing the annotation levels (or layers) from both input files and normalizing the corresponding segments.[8] The merge process leads to a complete reorganization of the segment list making it necessary to update the segment references of the elements in the XStandoff annotation layers. All that is done by applying the mergeXSF script.

Other stylesheets allow for the extraction and removal of single annotation layers, or a quick overview of overlapping annotations – see Stührenberg and Jettka (2009) for a detailed discussion. The current version of the stylesheet only supports the merging of two single XStandoff files at a time, additional files have to be merged successively. However, there is a web-based solution that uses the native XML database BaseX[9] as backend as well as a Java GUI that eases bulk transformation, merging and analyzing XStandoff instances.

In Jettka and Stührenberg (2011), different visualization options for concurrent markup (for example, the underlying XHTML annotation and one or more linguistic annotation layers) based on XStandoff are discussed, including newer web technologies such as WebGL for a three-dimensional visualization of overlapping subtrees. Although the examples given in this paper are quite short, Piez (2010; 2012) has already shown that the underlying concept is capable of visualizing larger instances (such as whole books) as well.

The full version of the XStandoff Toolkit can be obtained at XStandoff's website[10], although up to now it has not been adapted to support the additional segmentation mechanism for valid XHTML files described in Section 2.1.

4 Using XStandoff

The format as such has been successfully used in various projects for different purposes, such as storage format for multiple annotated corpora as part of an semi-automatic anaphora resolution (Stührenberg and Goecke, 2008), import/-export serialization of the web-based annotation tool Serengeti (Diewald et al., 2008; Poesio et al., 2011), and as annotation format for lexical chains (Waltinger et al., 2008), amongst others. Due to the fact, that the newly introduced segmentation for pre-annotated and multimodal primary data (Stührenberg, 2013) are still under development, XStandoff has not been used for larger web corpora yet.

Regarding the size of an XStandoff instance with multiple annotation layers compared to a number of inline annotation instances, it is hard to make a general expression about the increase/decrease in size. On the one hand, an XStandoff instance usually does not include the primary data (resulting in a smaller file size), on the other hand the meta information included in an XSF instance such as the additional segmentation mechanism add to the overall file size. Single heavily annotated XSF instances can take up to multiple megabytes in size, however, there have not been any problems to process these files with standard XML tools such as XSLT and XQuery. Densely annotated texts benefit from the fact that segments over a defined text span (or XHTML subtree) are only instantiated once, resulting in a state of processing in which additional annotation layer do only add very few if any segment elements to the resulting XStandoff instance. As a rule of thumb, it is highly recommended to use native XML databases such as the already-mentioned BaseX or eXist[11] as storage backends for analyzing large corpora.

5 XStandoff compared

Since the concept of standoff annotation as such is not new at all, a variety of serialization formats already exist. The most prominent candidate for a standoff serialization format supporting multiple annotations is the Graph Annotation Format (GrAF), the pivot format of the international standard ISO 24612:2012 (Linguistic Annotation Framework). However, there are different versions

[8] Especially this normalization can be problematic: On the one hand, there are segments spanning over the same string of the primary data (but with distinct IDs) that have to be replaced by a single segment element in the output instance. On the other hand, there are two segments with the same ID spanning over different character positions that have to get new unique IDs.

[9] See http://basex.org for further details.

[10] See http://xstandoff.net for further details.

[11] See http://exist-db.org for further details.

of the format: The partial document grammar in the ISO standard differs from the one that is available at its web site[12] while the first release of the GrAF-annotated Manually Annotated Sub-Corpus (MASC)[13] again uses different element and attribute names.

Another issue is that the standard is quite indifferent in terms of the segmentation over the primary data. While anchors are defined via string values, the standard states that, "[a]pplications are expected to know how to parse the string representation of an anchor into a location in the artifact being annotated" (Table 3, in the standard document). Although pre-annotated primary data is supported[14], one either may include markup as part of the character stream when referring to character positions, or use a combination of an XPath 2.0 expression to select the element containing the text, and an offset to select the corresponding part of the character string (see Section 3.3.4 of the standard) – XPath 2.0's `substring()` function shown in Listing 2 is not used.

Concerning the annotation itself, GrAF uses a feature structure format that resembles the serialization standardized in ISO 24610-1 and Chapter 18 of the TEI P5 (Burnard and Bauman, 2014). Converting existing annotation into this format can be considered as a more complex task and the resulting subtrees may become quite large (see Stegmann and Witt (2009) for a discussion of TEI feature structures as serialization for multiple annotated XML instances).

6 Conclusion and future development

Standoff annotation can be a valuable means in annotating web corpora, especially when combined with a strict policy of storing both the raw data and the primary data as non-altered files. With its segmentation mechanism supporting XPath 2.0 expressions, XStandoff can use only slightly processed XHTML pages together with their respective annotation layers, allowing for less destructive cleaning of web pages.

Since the segmentation mechanism discussed in this paper have been added to XStandoff only recently, non-textual primary data is not yet supported by the current version of the XStandoff Toolkit. Although it is much easier to identify the respective subtrees of valid XHTML pages (for example by using XPath visualization and/or selection tools such as the one included in the oXygen XML Editor[15]) compared to computing character positions, an automatic instantiation of segments is preferred. We plan to include the segmentation over pre-annotated files in one of the next iterations of the XStandoff Toolkit.

References

Anders Berglund, Scott Boag, Don Chamberlin, Mary F. Fernández, Michael Kay, Jonathan Robie, and Jérôme Siméon. 2010. XML Path Language (XPath). Version 2.0 (Second Edition). W3C Recommendation, World Wide Web Consortium.

Robin Berjon, Steve Faulkner, Travis Leithead, Erika Doyle Navara, Edward O'Connor, Silvia Pfeiffer, and Ian Hickson. 2014. Html5. a vocabulary and associated apis for html and xhtml. W3C Candidate Recommendation, World Wide Web Consortium.

Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, and François Yergeau. 2008. Extensible Markup Language (XML) 1.0 (Fifth Edition). W3C Recommendation, World Wide Web Consortium.

Tim Bray, Dave Hollander, Andrew Layman, Richard Tobin, and Henry S. Thompson. 2009. Namespaces in XML 1.0 (third edition). W3C Recommendation, World Wide Web Consortium.

Lou Burnard and Syd Bauman, editors. 2014. *TEI P5: Guidelines for Electronic Text Encoding and Interchange*. Text Encoding Initiative Consortium, Charlottesville, Virginia. Version 2.6.0. Last updated on 20th January 2014, revision 12802.

Steven J. DeRose, Ron Jr. Daniel, Paul Grosso, Eve Maler, Jonathan Marsh, and Norman Walsh. 2002a. XML Pointer Language (XPointer). W3C Working Draft, World Wide Web Consortium.

Steven J. DeRose, Eve Maler, and Ron Jr. Daniel. 2002b. XPointer xpointer() Scheme. W3C Working Draft, World Wide Web Consortium.

Nils Diewald, Maik Stührenberg, Anna Garbar, and Daniela Goecke. 2008. Serengeti – webbasierte Annotation semantischer Relationen. *Journal for Language Technology and Computational Linguistics*, 23(2):74–93.

Shudi (Sandy) Gao, C. M. Sperberg-McQueen, and Henry S. Thompson. 2012. W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures. W3C Recommendation, World Wide Web Consortium.

[12] See `http://www.xces.org/ns/GrAF/1.0/` for further details.

[13] See `http://www.anc.org/MASC/About.html` for further details.

[14] The preferred primary data format is raw text.

[15] See `http://oxygenxml.com` for further details

Nancy M. Ide, Patrice Bonhomme, and Laurent Romary. 2000. XCES: An XML-based Encoding Standard for Linguistic Corpora. In *Proceedings of the Second International Language Resources and Evaluation (LREC 2000)*, pages 825–830, Athens. European Language Resources Association (ELRA).

ISO/TC 37/SC 4/WG 1. 2006. Language Resource Management — Feature Structures – Part 1: Feature Structure Representation. International Standard ISO 24610-1:2006, International Organization for Standardization, Geneva.

ISO/TC 37/SC 4/WG 1. 2012. Language Resource Management — Linguistic annotation framework (LAF). International Standard ISO 24612:2012, International Organization for Standardization, Geneva.

Daniel Jettka and Maik Stührenberg. 2011. Visualization of concurrent markup: From trees to graphs, from 2d to 3d. In *Proceedings of Balisage: The Markup Conference*, volume 7 of *Balisage Series on Markup Technologies*, Montréal.

Ashok Malhotra, Jim Melton, Norman Walsh, and Michael Kay. 2010. XQuery 1.0 and XPath 2.0 Functions and Operators (Second Edition). W3C Recommendation, World Wide Web Consortium.

Wendell Piez. 2010. Towards Hermeneutic Markup: An architectural outline. In *Digital Humanities 2010 Conference Abstracts*, pages 202–205, London. The Alliance of Digital Humanities Organisations and The Association for Literary and Linguistic Computing and The Association for Computers and the Humanities and The Society for Digital Humanities – Société pour l'étude des médias interactif.

Wendell Piez. 2012. Luminescent: parsing LMNL by XSLT upconversion. In *Proceedings of Balisage: The Markup Conference*, volume 8 of *Balisage Series on Markup Technologies*, Montréal.

Massimo Poesio, Nils Diewald, Maik Stührenberg, Jon Chamberlain, Daniel Jettka, Daniela Goecke, and Udo Kruschwitz. 2011. Markup Infrastructure for the Anaphoric Bank: Supporting Web Collaboration. In Alexander Mehler, Kai-Uwe Kühnberger, Henning Lobin, Harald Lüngen, Angelika Storrer, and Andreas Witt, editors, *Modeling, Learning and Processing of Text Technological Data Structures*, volume 370 of *Studies in Computational Intelligence*, pages 175–195. Springer, Berlin and Heidelberg.

Jens Stegmann and Andreas Witt. 2009. TEI Feature Structures as a Representation Format for Multiple Annotation and Generic XML Documents. In *Proceedings of Balisage: The Markup Conference*, volume 3 of *Balisage Series on Markup Technologies*, Montréal.

Maik Stührenberg and Daniela Goecke. 2008. SGF – an integrated model for multiple annotations and its application in a linguistic domain. In *Proceedings of Balisage: The Markup Conference*, volume 1 of *Balisage Series on Markup Technologies*, Montréal.

Maik Stührenberg and Daniel Jettka. 2009. A toolkit for multi-dimensional markup: The development of SGF to XStandoff. In *Proceedings of Balisage: The Markup Conference*, volume 3 of *Balisage Series on Markup Technologies*, Montréal.

Maik Stührenberg. 2013. What, when, where? Spatial and temporal annotations with XStandoff. In *Proceedings of Balisage: The Markup Conference*, volume 10 of *Balisage Series on Markup Technologies*, Montréal.

Ulli Marc Waltinger, Alexander Mehler, and Maik Stührenberg. 2008. An integrated model of lexical chaining: application, resources and its format. In Angelika Storrer, Alexander Geyken, Alexander Siebert, and Kay-Michael Würzner, editors, *KONVENS 2008 – Ergänzungsband Textressourcen und lexikalisches Wissen*, pages 59–70, Berlin.

Andreas Witt. 2004. Multiple hierarchies: New Aspects of an Old Solution. In *Proceedings of Extreme Markup Languages*, Montréal.