

Annotation Interchange with XSLT*

Graham Wilcock
University of Helsinki

Abstract

The paper describes an XSLT stylesheet that transforms annotations from GATE XML to UIMA XML format. It extends an existing set of stylesheets that are freely available for download (Wilcock, 2009).

1 Introduction

One of the reasons XML was rapidly adopted as a standard format for data interchange ten years ago was that the details can be changed. Whatever specific format the data is in, if it's XML you can change it into the format you want using XSLT transformations. This was summed up in the joke *XML means never having to say you're sorry*.

For linguistic annotations, stand-off markup is normally used. The XML format for stand-off annotations is more complex than for in-line annotations. Nevertheless, the joke is not wrong: stand-off annotations in one specific format can be transformed into stand-off annotations in another specific format using XSLT stylesheets.

To demonstrate the truth of this claim in practice, not merely in theory, we wrote a set of XSLT stylesheets that transform between specific stand-off annotation formats. The formats are WordFreak XML (Morton & LaCivita, 2003), GATE XML (Cunningham *et al.*, 2002) and UIMA XML (Götz & Suhre, 2004). Stylesheets that transform GATE XML to WordFreak XML, and WordFreak XML to UIMA XML, are described by Wilcock (2009). This paper describes another example stylesheet that transforms GATE XML to UIMA XML.

2 Transforming GATE XML to UIMA XML

GATE XML format includes `Feature` elements that contain the feature's Name and its Value. The fragment of GATE XML in Figure 1 shows an `Annotation` for a token that includes a `Feature` whose Name is *string* and whose Value is *dun*. The same `Annotation` includes another `Feature` whose Name is *category* and whose Value is *NN*. This shows that GATE has tagged the token *dun* as NN.

The stylesheet `gate2uima.xsl` transforms GATE XML into UIMA XML (XMI). Namespace declarations are included for XMI, for the UIMA CAS (Common Analysis Structure) and for the type system used by the UIMA OpenNLP examples.

* Published in: *Proc. GSCL 2009* TODO . Seite ??-??.

```

<Annotation Id="100" Type="Token" StartNode="199" EndNode="202">
  <Feature>
    <Name className="java.lang.String">category</Name>
    <Value className="java.lang.String">NN</Value>
  </Feature>
  ...
  <Feature>
    <Name className="java.lang.String">string</Name>
    <Value className="java.lang.String">dun</Value>
  </Feature>
</Annotation>

```

Figure 1: GATE XML: An Annotation showing the token *dun* tagged as NN

The overall structure of the UIMA XML file is built by the template in Figure 2. This creates the root element `xmi:XMI`, the `cas:Sofa` and `cas:View` elements in `xmi:XMI`, and uses `<apply-templates select="AnnotationSet"/>` to produce the detailed annotations in between. The `cas:View` element includes a list of `xmi:id` numbers. Only annotations whose `xmi:id` numbers are listed as members of the view will be shown when the view is displayed by the UIMA Annotation Viewer. The stylesheet collects the list of `xmi:id` numbers for all annotation types (except `file`) by means of an `<xsl:for-each>` loop.

An `opennlp:Sentence` element is created for each GATE sentence annotation, and an `opennlp:Token` element for each GATE token annotation. The processing of tokens is shown in Figure 3. The values of the UIMA `begin` and `end` attributes are extracted from the GATE `StartNode` and `EndNode` attributes. The UIMA `posTag` attribute gets its value from the GATE Feature whose Name is *category*. For tokens, the UIMA `componentId` attribute is set to “GATE Tokenizer” and for sentences it is set to “GATE Sentence Splitter”. The GATE token *dun* (Figure 1) is shown after the transformation to UIMA XML in Figure 4, viewed in the UIMA Annotation Viewer.

3 Conclusion

Stylesheets that transform GATE XML to WordFreak XML, and WordFreak XML to UIMA XML, are described by Wilcock (2009). The stylesheets are available for download from <http://sites.morganclaypool.com/wilcock>. This paper describes another example stylesheet that transforms GATE XML to UIMA XML.

References

- Cunningham, H., Maynard, D., Bontcheva, K., & Tablan, V. (2002). GATE: A framework and graphical development environment for robust NLP tools and applications. In *40th Anniversary Meeting of the Association for Computational Linguistics*, Philadelphia.
- Götz, T. & Suhre, O. (2004). Design and implementation of the UIMA Common Analysis System. *IBM Systems Journal*, **43**(3), 476–489.
- Morton, T. & LaCivita, J. (2003). Wordfreak: An open tool for linguistic annotation. In *Proceedings of HLT-NAACL 2003, Demonstrations*, pages 17–18, Edmonton.
- Wilcock, G. (2009). *Introduction to Linguistic Annotation and Text Analytics*. Morgan and Claypool.

```

<!-- Template to process top-level document element -->
<xsl:template match="GateDocument">
  <xsl:element name="xmi:XMI">
    <xsl:attribute name="xmi:version">2.0</xsl:attribute>

    <xsl:element name="cas:NULL">
      <xsl:attribute name="xmi:id">0</xsl:attribute>
    </xsl:element>

    <!-- Make UIMA sofaString from GATE TextWithNodes -->
    <xsl:element name="cas:Sofa">
      <xsl:attribute name="xmi:id">1</xsl:attribute>
      <xsl:attribute name="sofaNum">1</xsl:attribute>
      <xsl:attribute name="sofaID">_InitialView</xsl:attribute>
      <xsl:attribute name="mimetype">text</xsl:attribute>
      <xsl:attribute name="sofaString">
        <xsl:value-of select="TextWithNodes"/>
      </xsl:attribute>
    </xsl:element>

    <xsl:apply-templates select="AnnotationSet"/>

    <xsl:element name="cas:View">
      <xsl:attribute name="sofa">1</xsl:attribute>
      <xsl:attribute name="members">
        <xsl:text>999998</xsl:text>
        <xsl:text> </xsl:text>
        <xsl:text>999999</xsl:text>
        <xsl:for-each select="//Annotation[@Type='Sentence']">
          <xsl:text> </xsl:text>
          <xsl:value-of select="@Id"/>
        </xsl:for-each>
        <xsl:for-each select="//Annotation[@Type='Token']">
          <xsl:text> </xsl:text>
          <xsl:value-of select="@Id"/>
        </xsl:for-each>
      </xsl:attribute>
    </xsl:element>

  </xsl:element>
</xsl:template>

```

Figure 2: gate2uima.xsl: Transforming the document from GATE XML to UIMA XML

```

<!-- Template to process tokens -->
<xsl:template match="Annotation[@Type='Token']">
  <xsl:element name="opennlp:Token">
    <xsl:attribute name="xmi:id">
      <xsl:value-of select="@Id"/>
    </xsl:attribute>
    <xsl:attribute name="sofa">1</xsl:attribute>
    <xsl:attribute name="begin">
      <xsl:value-of select="@StartNode"/>
    </xsl:attribute>
    <xsl:attribute name="end">
      <xsl:value-of select="@EndNode"/>
    </xsl:attribute>
    <xsl:attribute name="posTag">
      <xsl:value-of select="Feature[Name='category']/Value"/>
    </xsl:attribute>
    <xsl:attribute name="componentId">
      <xsl:text>GATE Tokenizer</xsl:text>
    </xsl:attribute>
  </xsl:element>
</xsl:template>

```

Figure 3: gate2uima.xsl: Transforming tokens from GATE XML to UIMA XML

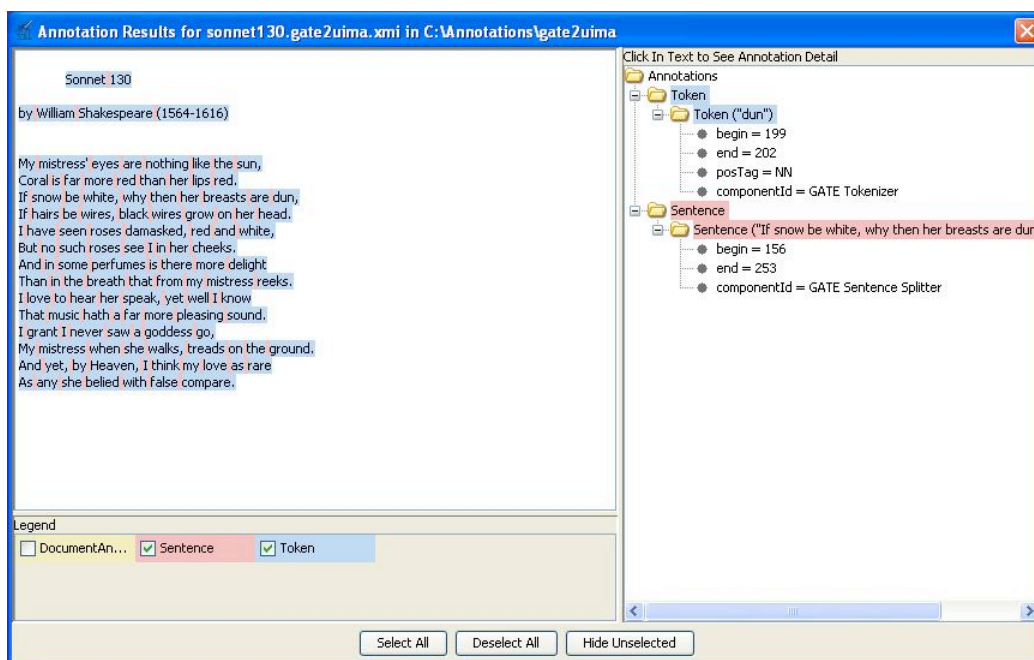


Figure 4: UIMA Annotation Viewer showing the token *dun* tagged NN by GATE