TigerXML

Quick Reference Guide

Wolfgang Lezius
Hannes Biesinger
Ciprian Gerstenberger

IMS, University of Stuttgart

January 9, 2002

Contents

1 About this document **1**

2 A walk through the TIGER-XML format **3**

2.1 Corpus header 3

2.2 Corpus body 5

3 An example corpus **8**

4 The TIGER-XML schema **15**

Bibliography **25**

Chapter 1

About this document

As there are various formats for the representation of linguistic corpora, there are also a number of formats for the encoding of syntactically annotated corpora or treebanks: Penn Treebank, Susanne, Negra, Tipster, and several other formats plus various kinds of parser output. Since applications like TIGERSearch cannot support all existing formats, it makes sense to define one single XML-based interface format for import and export. This format should be general enough to encode as many existing formats as possible.

The TIGER-XML format has been designed as an interface format. It is an XML-based equivalence of the corpus definition sublanguage of the TIGER description language presented in [1] and [2]. In addition to corpus definitions, the TIGER-XML format can also represent query results.

Any corpus to be processed by the TIGERSearch tool has to be converted into TIGER-XML first. For convenience, we have implemented corpus filters (i.e. converters to TIGER-XML) for many popular treebank and parser output formats like bracketing format, PennTreebank format, NeGra format etc.

Why should you read this quick reference guide? First of all, you might have corpora encoded in different formats. So knowledge about TIGER-XML is essential for their conversion. In addition, TIGERSearch supports exporting the matches of a query in the TIGER-XML format. If you like to transform the XML output, e.g. by XSLT stylesheets, you will have to know how the XML document has been designed.

This quick reference guide first gives you an impression of the TIGER-XML format by walking through an example (cf. chapter 2). In the second chapter a real-life example is presented (cf. chapter 3). Finally, you can find a description of the XML schema used to validate TIGER-XML documents (cf. chapter 4).

A former version of the TIGER-XML format has been presented at the LREC conference 2000 (cf. [5]). Though some notations have changed in TIGER-XML, this paper might be helpful to understand the basic ideas of the encoding format. If you are interested in the indexing of corpora using the TIGERRegistry tool or exporting the matches to TIGER-XML using the TIGERSearch tool, please consult the manuals of the tools (cf. [4] and [3]).

Chapter 2

A walk through the TIGER-XML format

The following introduction to TIGER-XML is a kind of guided tour. It walks through the encoding format and explains its properties.

A TIGER-XML document consists of two parts: In the header you can find the corpus declaration (cf. [1]) and some meta information (cf. section 2.1). The body comprises the definition of the corpus sentences and their annotation (cf. section 2.2).

2.1 Corpus header

The corpus header consists of two parts. General meta information about the corpus is encoded in the first part of the header: name of the corpus, author, date, short description, format and history. This corpus meta information is displayed by the TIGERSearch tool when browsing through the corpora. The ID of the corpus (cf. root element <corpus>) should be unique (with regard to all corpora you have prepared).

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>

<corpus id="TESTCORPUS"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation=
   "http://www.ims.uni-stuttgart.de/projekte/TIGER/public/TigerXML.xsd">
```

```
<head>

  <meta>
    <name>Test corpus</name>
    <author>Wolfgang Lezius</author>
    <date>November 2001</date>
    <description>Illustrates the TIGER-XML format.</description>
    <format>NeGra format, version 3</format>
    <history>first version</history>
  </meta>
```

The second part of the corpus header provides information about the features used in the corpus. This feature declaration is obligatory for corpora to be indexed by the TIGERRegistry tool (cf. [1]). Feature values and short explanations of the tags might be listed - this kind of meta information will be used by the TIGERSearch GUI as corpus documentation. In the following example, the feature word is declared as a feature of terminal nodes (T) and the feature cat as a feature of nonterminal nodes (NT). If a feature is used in both terminal and nonterminal nodes (e.g. case), its domain is called FREC. Potential edge labels are declared in an `<edgelabel>` element, secondary edges in an `<secedgelabel>` element.

```
  <annotation>

    <feature name="word" domain="T" />

    <feature name="pos" domain="T">
      <value name="ART">determiner</value>
      <value name="ADV">adverb</value>
      <value name="KOKOM">conjunction</value>
      <value name="NN">noun</value>
      <value name="PIAT">indefinite attributive pronoun</value>
      <value name="VVFIN">finite verb</value>
    </feature>

    <feature name="morph" domain="T">
      <value name="Def.Fem.Nom.Sg"/>
      <value name="Fem.Nom.Sg.*"/>
      <value name="Masc.Akk.Pl.*"/>
      <value name="3.Sg.Pres.Ind"/>
      <value name="--">not bound</value>
```

```
    </feature>

    <feature name="cat" domain="NT">
      <value name="AP">adjektive phrase</value>
      <value name="AVP">adverbial phrase</value>
      <value name="NP">noun phrase</value>
      <value name="S">sentence</value>
    </feature>

    <edgelabel>
      <value name="CC">comparative complement</value>
      <value name="CM">comparative concjunction</value>
      <value name="HD">head</value>
      <value name="MO">modifier</value>
      <value name="NK">noun kernel modifier</value>
      <value name="OA">accusative object</value>
      <value name="SB">subject</value>
    </edgelabel>

  </annotation>

</head>
```

2.2 Corpus body

The data model supported by TIGERSearch is based on so-called syntax graphs, i.e. directed acyclic graphs with exactly one root node (for a formal definition cf. [2]). Thus, the sentences cannot be encoded by (self-)embedding XML elements. As a solution, all terminal and nonterminal nodes are listed and edges are explicitly encoded. The following example illustrates the sentence encoding.

```
<body>

<s id="s5">
  <graph root="s5_504">
    <terminals>
      <t id="s5_1" word="Die" pos="ART" morph="Def.Fem.Nom.Sg" />
      <t id="s5_2" word="Tagung" pos="NN" morph="Fem.Nom.Sg.*" />
      <t id="s5_3" word="hat" pos="VVFIN" morph="3.Sg.Pres.Ind" />
```

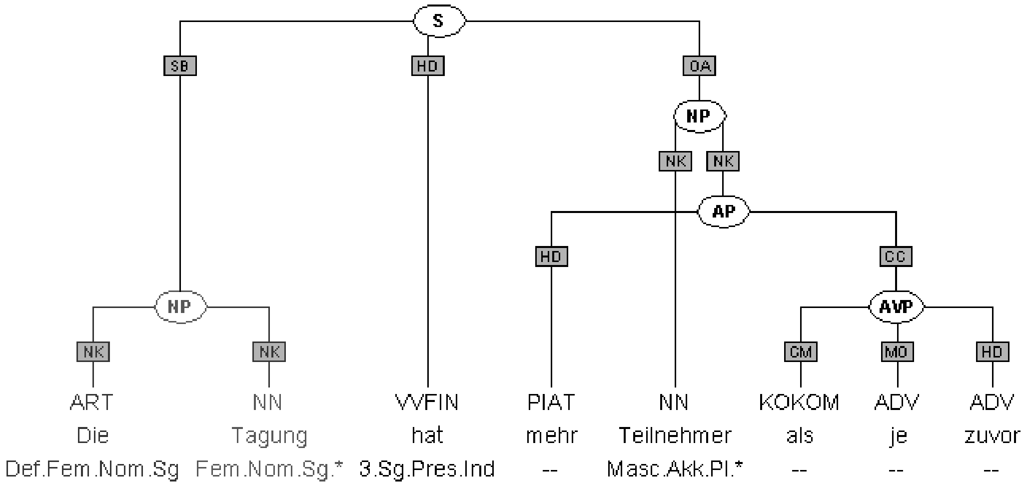


Figure 2.1: Example sentence

```
    <t id="s5_4" word="mehr" pos="PIAT" morph="--" />
    <t id="s5_5" word="Teilnehmer" pos="NN" morph="Masc.Akk.Pl.*" />
    <t id="s5_6" word="als" pos="KOKOM" morph="--" />
    <t id="s5_7" word="je" pos="ADV" morph="--" />
    <t id="s5_8" word="zuvor" pos="ADV" morph="--" />
  </terminals>
  <nonterminals>
    <nt id="s5_500" cat="NP">
      <edge label="NK" idref="s5_1" />
      <edge label="NK" idref="s5_2" />
    </nt>
    <nt id="s5_501" cat="AVP">
      <edge label="CM" idref="s5_6" />
      <edge label="MO" idref="s5_7" />
      <edge label="HD" idref="s5_8" />
    </nt>
    <nt id="s5_502" cat="AP">
      <edge label="HD" idref="s5_4" />
      <edge label="CC" idref="s5_501" />
    </nt>
    <nt id="s5_503" cat="NP">
      <edge label="NK" idref="s5_502" />
      <edge label="NK" idref="s5_5" />
    </nt>
    <nt id="s5_504" cat="S">
```

```
        <edge label="SB" idref="s5_500" />
        <edge label="HD" idref="s5_3" />
        <edge label="OA" idref="s5_503" />
      </nt>
    </nonterminals>
  </graph>
  <matches>
    <match subgraph="s5_500">
      <variable name="#w" idref="s5_2" />
      <variable name="#v" idref="s5_500" />
    </match>
    <match subgraph="s5_503">
      <variable name="#w" idref="s5_5" />
      <variable name="#v" idref="s5_503" />
    </match>
  </matches>
</s>

</body>

</corpus>
```

Please note that feature values of the declared features (see above) always have to specified. If a feature value or edge label doesn't make sense for a token or inner node (example sentence: feature morph is unspecified), please use a conclusive symbol instead (in the example sentence the symbol is '`--`'). We recommend you to use the symbol '`--`' which is also used in our implemented import filters. When viewing a matching sentence using the TIGERGraphViewer, a feature value or edge label such as '`--`' can be suppressed.

This example sentence also illustrates the encoding of query results. It matches the query #v:[cat="NP"]>#w:[pos="NN"] twice. Both matches are represented by `<match>` elements. The `<variable>` elements refer to the corresponding graph nodes matching the variables #v and #w. Hence the IDs of the `<t>` and `<nt>` elements are essential for both the edge linking mechanism and match reference mechanism.

Chapter 3

An example corpus

The following listing shows the TIGER-XML representation of a small demo corpus. This corpus comprises two sentences of the *Wall Street Journal* corpus of the PennTreebank. It makes use of edge labels and also of secondary edges to represent coreference annotation.


```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<corpus id="DEMO"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation=
  "http://www.ims.uni-stuttgart.de/projekte/TIGER/public/TigerXML.xsd">

<head>
  <meta>
    <name>two sentences of Wall Street Journal corpus</name>
    <format>bracketing format</format>
  </meta>
  <annotation>
    <feature name="word" domain="T" />
    <feature name="pos" domain="T">
       <value name="," />
       <value name="-NONE-" />
       <value name="." />
       <value name="CC" />
       <value name="CD" />
       <value name="DT" />
       <value name="IN" />
       <value name="JJ" />
```

```
      <value name="MD" />
      <value name="NN" />
      <value name="NNP" />
      <value name="NNS" />
      <value name="VB" />
      <value name="VBD" />
      <value name="VBN" />
   </feature>
   <feature name="cat" domain="NT">
      <value name="ADJP" />
      <value name="NP" />
      <value name="PP" />
      <value name="S" />
      <value name="UCP" />
      <value name="VP" />
   </feature>
   <edgelabel>
      <value name="--">not bound</value>
      <value name="CLR" />
      <value name="PRD" />
      <value name="SBJ" />
      <value name="TMP" />
   </edgelabel>
   <secedgelabel>
      <value name="*" />
   </secedgelabel>
  </annotation>
</head>


<body>

<s id="s1">
  <graph root="s1_500">
    <terminals>
      <t id="s1_1" word="Pierre" pos="NNP" />
      <t id="s1_2" word="Vinken" pos="NNP" />
      <t id="s1_3" word="," pos="," />
      <t id="s1_4" word="61" pos="CD" />
      <t id="s1_5" word="years" pos="NNS" />
      <t id="s1_6" word="old" pos="JJ" />
      <t id="s1_7" word="," pos="," />
      <t id="s1_8" word="will" pos="MD" />
```

```
    <t id="s1_9" word="join" pos="VB" />
    <t id="s1_10" word="the" pos="DT" />
    <t id="s1_11" word="board" pos="NN" />
    <t id="s1_12" word="as" pos="IN" />
    <t id="s1_13" word="a" pos="DT" />
    <t id="s1_14" word="nonexecutive" pos="JJ" />
    <t id="s1_15" word="director" pos="NN" />
    <t id="s1_16" word="Nov." pos="NNP" />
    <t id="s1_17" word="29" pos="CD" />
    <t id="s1_18" word="." pos="." />
  </terminals>
  <nonterminals>
    <nt id="s1_502" cat="NP">
      <edge label="--" idref="s1_1" />
      <edge label="--" idref="s1_2" />
    </nt>
    <nt id="s1_504" cat="NP">
      <edge label="--" idref="s1_4" />
      <edge label="--" idref="s1_5" />
    </nt>
    <nt id="s1_503" cat="ADJP">
      <edge label="--" idref="s1_504" />
      <edge label="--" idref="s1_6" />
    </nt>
    <nt id="s1_501" cat="NP">
      <edge label="--" idref="s1_502" />
      <edge label="--" idref="s1_3" />
      <edge label="--" idref="s1_503" />
      <edge label="--" idref="s1_7" />
    </nt>
    <nt id="s1_507" cat="NP">
      <edge label="--" idref="s1_10" />
      <edge label="--" idref="s1_11" />
    </nt>
    <nt id="s1_509" cat="NP">
      <edge label="--" idref="s1_13" />
      <edge label="--" idref="s1_14" />
      <edge label="--" idref="s1_15" />
    </nt>
    <nt id="s1_508" cat="PP">
      <edge label="--" idref="s1_12" />
      <edge label="--" idref="s1_509" />
    </nt>
```

```
      <nt id="s1_510" cat="NP">
        <edge label="--" idref="s1_16" />
        <edge label="--" idref="s1_17" />
      </nt>
      <nt id="s1_506" cat="VP">
        <edge label="--" idref="s1_9" />
        <edge label="--" idref="s1_507" />
        <edge label="CLR" idref="s1_508" />
        <edge label="TMP" idref="s1_510" />
      </nt>
      <nt id="s1_505" cat="VP">
        <edge label="--" idref="s1_8" />
        <edge label="--" idref="s1_506" />
      </nt>
      <nt id="s1_500" cat="S">
        <edge label="SBJ" idref="s1_501" />
        <edge label="--" idref="s1_505" />
        <edge label="--" idref="s1_18" />
      </nt>
    </nonterminals>
  </graph>
</s>

<s id="s3">
  <graph root="s3_500">
    <terminals>
      <t id="s3_1" word="Rudolph" pos="NNP" />
      <t id="s3_2" word="Agnew" pos="NNP" />
      <t id="s3_3" word="," pos="," />
      <t id="s3_4" word="55" pos="CD" />
      <t id="s3_5" word="years" pos="NNS" />
      <t id="s3_6" word="old" pos="JJ" />
      <t id="s3_7" word="and" pos="CC" />
      <t id="s3_8" word="former" pos="JJ" />
      <t id="s3_9" word="chairman" pos="NN" />
      <t id="s3_10" word="of" pos="IN" />
      <t id="s3_11" word="Consolidated" pos="NNP" />
      <t id="s3_12" word="Gold" pos="NNP" />
      <t id="s3_13" word="Fields" pos="NNP" />
      <t id="s3_14" word="PLC" pos="NNP" />
      <t id="s3_15" word="," pos="," />
      <t id="s3_16" word="was" pos="VBD" />
      <t id="s3_17" word="named" pos="VBN" />
```

```xml
    <t id="s3_18" word="*" pos="-NONE-" />
    <t id="s3_19" word="a" pos="DT" />
    <t id="s3_20" word="nonexecutive" pos="JJ" />
    <t id="s3_21" word="director" pos="NN" />
    <t id="s3_22" word="of" pos="IN" />
    <t id="s3_23" word="this" pos="DT" />
    <t id="s3_24" word="British" pos="JJ" />
    <t id="s3_25" word="industrial" pos="JJ" />
    <t id="s3_26" word="conglomerate" pos="NN" />
    <t id="s3_27" word="." pos="." />
  </terminals>
  <nonterminals>
    <nt id="s3_502" cat="NP">
      <edge label="--" idref="s3_1" />
      <edge label="--" idref="s3_2" />
    </nt>
    <nt id="s3_505" cat="NP">
      <edge label="--" idref="s3_4" />
      <edge label="--" idref="s3_5" />
    </nt>
    <nt id="s3_504" cat="ADJP">
      <edge label="--" idref="s3_505" />
      <edge label="--" idref="s3_6" />
    </nt>
    <nt id="s3_507" cat="NP">
      <edge label="--" idref="s3_8" />
      <edge label="--" idref="s3_9" />
    </nt>
    <nt id="s3_509" cat="NP">
      <edge label="--" idref="s3_11" />
      <edge label="--" idref="s3_12" />
      <edge label="--" idref="s3_13" />
      <edge label="--" idref="s3_14" />
    </nt>
    <nt id="s3_508" cat="PP">
      <edge label="--" idref="s3_10" />
      <edge label="--" idref="s3_509" />
    </nt>
    <nt id="s3_506" cat="NP">
      <edge label="--" idref="s3_507" />
      <edge label="--" idref="s3_508" />
    </nt>
    <nt id="s3_503" cat="UCP">
```

```
    <edge label="--" idref="s3_504" />
    <edge label="--" idref="s3_7" />
    <edge label="--" idref="s3_506" />
  </nt>
  <nt id="s3_501" cat="NP">
    <edge label="--" idref="s3_502" />
    <edge label="--" idref="s3_3" />
    <edge label="--" idref="s3_503" />
    <edge label="--" idref="s3_15" />
    <secedge label="*" idref="s3_18" />
  </nt>
  <nt id="s3_513" cat="NP">
    <edge label="--" idref="s3_18" />
  </nt>
  <nt id="s3_515" cat="NP">
    <edge label="--" idref="s3_19" />
    <edge label="--" idref="s3_20" />
    <edge label="--" idref="s3_21" />
  </nt>
  <nt id="s3_517" cat="NP">
    <edge label="--" idref="s3_23" />
    <edge label="--" idref="s3_24" />
    <edge label="--" idref="s3_25" />
    <edge label="--" idref="s3_26" />
  </nt>
  <nt id="s3_516" cat="PP">
    <edge label="--" idref="s3_22" />
    <edge label="--" idref="s3_517" />
  </nt>
  <nt id="s3_514" cat="NP">
    <edge label="--" idref="s3_515" />
    <edge label="--" idref="s3_516" />
  </nt>
  <nt id="s3_512" cat="S">
    <edge label="SBJ" idref="s3_513" />
    <edge label="PRD" idref="s3_514" />
  </nt>
  <nt id="s3_511" cat="VP">
    <edge label="--" idref="s3_17" />
    <edge label="--" idref="s3_512" />
  </nt>
  <nt id="s3_510" cat="VP">
    <edge label="--" idref="s3_16" />
```

```
        <edge label="--" idref="s3_511" />
      </nt>
      <nt id="s3_500" cat="S">
        <edge label="SBJ" idref="s3_501" />
        <edge label="--" idref="s3_510" />
        <edge label="--" idref="s3_27" />
      </nt>
    </nonterminals>
  </graph>
</s>

</body>

</corpus>
```

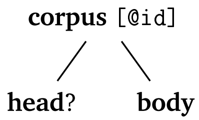
Chapter 4

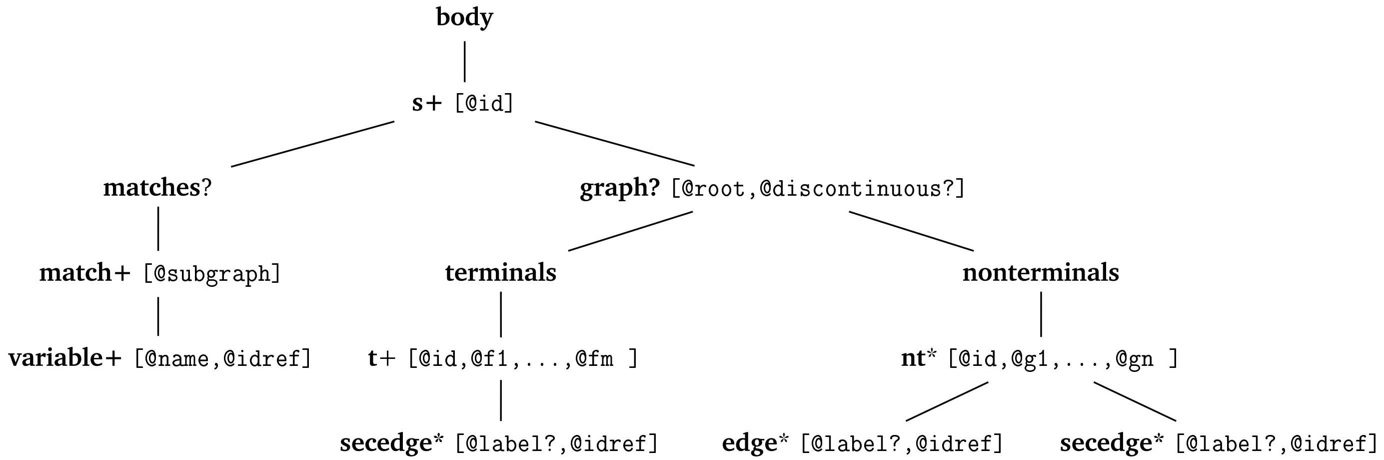
The TIGER-XML schema

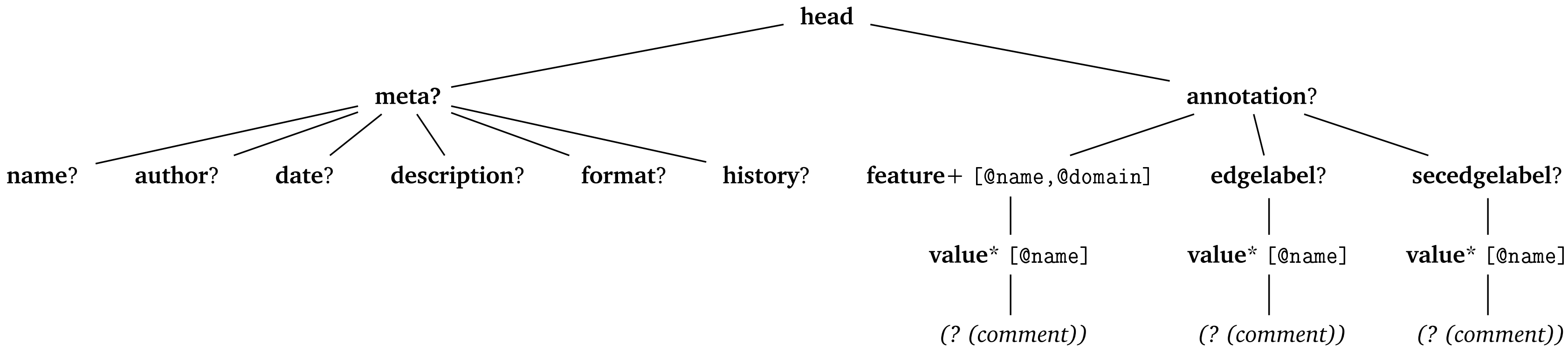
The TIGER-XML format is validated against an XML schema. These days XML schema validation is supported by all major XML parsers. This chapter comprises a graphical representation of the TIGER-XML format and a listing of the XML schema.

Here is a list of symbols used in the graphical representation:

symbol	description
`[]`	list of element attributes
`@name`	attribute named `name`
name	element named **name**
?	zero or one of the preceding item (0 or 1 times)
+	at least one of the preceding item (1 or more times)
$*$	zero or more of the preceding item (0 or more times)


corpus [@id]
head?
body


body
s+ [@id]
matches?
graph? [@root,@discontinuous?]
match+ [@subgraph]
terminals
nonterminals
variable+ [@name,@idref]
t+ [@id,@f1,...,@fm]
nt* [@id,@g1,...,@gn]
secedge* [@label?,@idref]
edge* [@label?,@idref]
secedge* [@label?,@idref]


head
meta?
annotation?
name?
author?
date?
description?
format?
history?
feature+ [@name,@domain]
edgelabel?
secedgelabel?
value* [@name]
value* [@name]
value* [@name]
(? (comment))
(? (comment))
(? (comment))

```
<?xml version="1.0"?>


 <!-- ================================================================
      XML Schema for the TIGER-XML format, 12/06/2001
      http://www.ims.uni-stuttgart.de/projekte/TIGER/TigerXML.xsd
      ================================================================
      TIGER project, Wolfgang Lezius, IMS, University of Stuttgart
      ================================================================ -->


<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">


  <!-- =======================================================
       DECLARATION OF THE DOCUMENT
       ======================================================= -->


  <!-- declaration of the root element: corpus -->

  <xsd:element name="corpus">

    <xsd:complexType>

      <xsd:sequence>

        <xsd:choice>
           <!-- header of the document is optional -->
           <xsd:element name="head" type="headType" minOccurs="0" maxOccurs="1"/>
        </xsd:choice>

        <xsd:element name="body" type="bodyType" minOccurs="1" maxOccurs="1"/>

      </xsd:sequence>

      <!-- corpus ID -->
      <xsd:attribute name="id"      type="idType"     use="required" />

      <!-- optional attribute: TigerXML version; used by TIGERSearch only -->
      <xsd:attribute name="version" type="xsd:string" use="optional" />

    </xsd:complexType>

  </xsd:element>



  <!-- declaration of the body type -->
```

```
<xsd:complexType name="bodyType">

  <xsd:sequence>
    <xsd:element name="s" type="sentenceType" minOccurs="1" maxOccurs="unbounded"/>
  </xsd:sequence>

</xsd:complexType>




<!-- =====================================================
     DECLARATION OF THE HEADER
     ===================================================== -->


<!-- declaration of the header type -->

<xsd:complexType name="headType">

   <xsd:sequence>
      <xsd:element name="meta"       type="metaType"       minOccurs="0" maxOccurs="1"/>
      <xsd:element name="annotation" type="annotationType" minOccurs="0" maxOccurs="1"/>
   </xsd:sequence>

   <!-- optional: reference to external header file

        The header of a TigerXML corpus can also be stored in separate file.
        This attribute points to the external header file. The pointer is
        an URI. Examples: file:relative.xml or file:/path/to/absolute.xml -->

   <xsd:attribute name="external" type="xsd:anyURI"/>

</xsd:complexType>



<!-- declaration of the meta information type -->

<xsd:complexType name="metaType">

  <xsd:sequence>
    <xsd:element name="name"        type="xsd:string"  minOccurs="0" maxOccurs="1"/>
    <xsd:element name="author"      type="xsd:string"  minOccurs="0" maxOccurs="1"/>
    <xsd:element name="date"        type="xsd:string"  minOccurs="0" maxOccurs="1"/>
    <xsd:element name="description" type="xsd:string"  minOccurs="0" maxOccurs="1"/>
    <xsd:element name="format"      type="xsd:string"  minOccurs="0" maxOccurs="1"/>
    <xsd:element name="history"     type="xsd:string"  minOccurs="0" maxOccurs="1"/>
  </xsd:sequence>
```

```
</xsd:complexType>



<!-- declaration of the annotation type -->

<xsd:complexType name="annotationType">

  <xsd:sequence>
    <xsd:element name="feature"      type="featureType"   minOccurs="1" maxOccurs="unbounded"/>
    <xsd:element name="edgelabel"    type="edgelabelType" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="secedgelabel" type="edgelabelType" minOccurs="0" maxOccurs="1"/>
  </xsd:sequence>

</xsd:complexType>



<!-- declaration of the feature type -->

<xsd:complexType name="featureType">

  <xsd:sequence>
     <xsd:element name="value" type="featurevalueType"  minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>

  <xsd:attribute name="name" type="featurenameType" use="required" />

  <xsd:attribute name="domain" use="required">
     <xsd:simpleType>
       <xsd:restriction base="xsd:string">
         <xsd:enumeration value="T"/>     <!-- feature for terminal nodes -->
         <xsd:enumeration value="NT"/>    <!-- feature for nonterminal nodes -->
         <xsd:enumeration value="FREC"/>  <!-- feature for both -->
       </xsd:restriction>
     </xsd:simpleType>
  </xsd:attribute>

</xsd:complexType>



<!-- declaration of the (secondary) edge label type -->

<xsd:complexType name="edgelabelType">

  <xsd:sequence>
     <xsd:element name="value" type="featurevalueType"  minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>

</xsd:complexType>
```

```
<!-- declaration of the feature value type -->

<xsd:complexType name="featurevalueType">

  <xsd:simpleContent>   <!-- element content: documentation of the feature value -->
    <xsd:extension base="xsd:string">
      <xsd:attribute name="name" type="xsd:string"/>
    </xsd:extension>
  </xsd:simpleContent>


</xsd:complexType>



<!-- =======================================================
     HEADER DECLARATIONS THAT SHOULD BE REFINED
     ======================================================= -->


<!-- declaration of the FEATURE NAMES used in the corpus header;
     this type is unrestricted, but should be refined by a
     specialised, corpus-dependent schema -->

<xsd:simpleType name="featurenameType">

  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1" />
    <xsd:maxLength value="20" />
    <xsd:whiteSpace value="preserve" />
  </xsd:restriction>

</xsd:simpleType>




<!-- =======================================================
     DECLARATION OF SENTENCES
     ======================================================= -->


<!-- declaration of the sentence type -->

<xsd:complexType name="sentenceType">

  <xsd:sequence>
    <xsd:element name="graph"   type="graphType"   minOccurs="0" maxOccurs="1"/>
```

```
    <xsd:element name="matches" type="matchesType" minOccurs="0" maxOccurs="1"/>
  </xsd:sequence>

  <xsd:attribute name="id" type="idType" use="required" />

</xsd:complexType>



<!-- declaration of the graph type -->

<xsd:complexType name="graphType">

  <xsd:sequence>
    <xsd:element name="terminals"    type="terminalsType"    minOccurs="1" maxOccurs="1"/>
    <xsd:element name="nonterminals" type="nonterminalsType" minOccurs="1" maxOccurs="1"/>
  </xsd:sequence>

  <xsd:attribute name="root" type="idrefType" use="required" />

  <!-- indicated that the exported sentence is discontinuous -->
  <xsd:attribute name="discontinuous" type="xsd:boolean" default="false" use="optional" />

</xsd:complexType>



<!-- declaration of the terminals type -->

<xsd:complexType name="terminalsType">

  <xsd:sequence>
    <xsd:element name="t" type="tType" minOccurs="1" maxOccurs="unbounded"/>
  </xsd:sequence>

</xsd:complexType>



<!-- declaration of the t element -->

<xsd:complexType name="tType">

  <!-- secondary edges possible -->
  <xsd:sequence>
    <xsd:element name="secedge" type="secedgeType" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>

  <xsd:attribute name="id" type="idType" use="required"/>
  <xsd:attributeGroup ref="tfeatureAttributes"/>
```

```
</xsd:complexType>



<!-- declaration of the nonterminals type -->

<xsd:complexType name="nonterminalsType">

  <xsd:sequence>
    <xsd:element name="nt" type="ntType" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>

</xsd:complexType>



<!-- declaration of the nt element -->

<xsd:complexType name="ntType">

  <!-- edge and secondary edges possible -->
  <xsd:sequence>
    <xsd:element name="edge"    type="edgeType"    minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="secedge" type="secedgeType" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>

  <xsd:attribute name="id" type="idType" use="required"/>
  <xsd:attributeGroup ref="ntfeatureAttributes"/>

</xsd:complexType>



<!-- declaration of the edge type -->

<xsd:complexType name="edgeType">

  <xsd:attribute name="idref" type="idrefType" use="required" />

  <xsd:attributeGroup ref="edgelabelAttribute" />

</xsd:complexType>



<!-- declaration of the secondary edge type -->

<xsd:complexType name="secedgeType">

  <xsd:attribute name="idref" type="idrefType" use="required"/>
```

```
    <xsd:attributeGroup ref="secedgelabelAttribute" />

  </xsd:complexType>



  <!-- declaration of the matches type -->

  <xsd:complexType name="matchesType">

    <xsd:sequence>
      <xsd:element name="match" type="matchType" minOccurs="1" maxOccurs="unbounded"/>
    </xsd:sequence>

  </xsd:complexType>



  <!-- declaration of the match type -->

  <xsd:complexType name="matchType">

    <xsd:sequence>
      <xsd:element name="variable" type="varType" minOccurs="1" maxOccurs="unbounded"/>
    </xsd:sequence>

    <xsd:attribute name="subgraph" type="idrefType" use="required"/>

  </xsd:complexType>



  <!-- declaration of the variable type -->

  <xsd:complexType name="varType">

    <xsd:attribute name="name" type="xsd:string" use="required"/>

    <xsd:attribute name="idref" type="idrefType" use="required"/>

  </xsd:complexType>



  <!-- ======================================================
       SENTENCE DECLARATIONS THAT SHOULD BE REFINED
       ====================================================== -->


  <!-- declaration of the TERMINAL FEATURE ATTRIBUTES;
       this group is unrestricted, but should be refined by a
```

```
     specialised, corpus-dependent schema -->

<xsd:attributeGroup name="tfeatureAttributes">

  <xsd:anyAttribute processContents="skip"/>

</xsd:attributeGroup>



<!-- declaration of the NONTERMINAL FEATURE ATTRIBUTES;
     this group is unrestricted, but should be refined by a
     specialised, corpus-dependent schema -->

<xsd:attributeGroup name="ntfeatureAttributes">

  <xsd:anyAttribute processContents="skip" />

</xsd:attributeGroup>



<!-- declaration of the EDGE-LABEL ATTRIBUTE;
     the label attribute is optional which should be refined by a
     specialised, corpus-dependent schema -->

<xsd:attributeGroup name="edgelabelAttribute">

  <xsd:attribute name="label" type="xsd:string" use="optional"/>

</xsd:attributeGroup>



<!-- declaration of the SECONDARY-EDGE-LABEL ATTRIBUTE;
     the label attribute is optional which should be refined by a
     specialised, corpus-dependent schema -->

<xsd:attributeGroup name="secedgelabelAttribute">

  <xsd:attribute name="label" type="xsd:string" use="optional"/>

</xsd:attributeGroup>



<!-- ======================================================
     ID and IDREF TYPE DECLARATIONS
     ====================================================== -->

<!-- Even though XML Schema are a W3C Recommendation, schema
```

```
       support of XML parsers is still restricted. Using some
       parsers you might have problems with the ID and IDREF
       attributes in combination with an "anyAttribute"
       declaration. In this case, just modify the base type
       of the following two declarations to "xsd:string".  -->


  <!-- declaration of idType -->

  <xsd:simpleType name="idType">

    <xsd:restriction base="xsd:ID"/>

  </xsd:simpleType>


  <!-- declaration of idrefType -->

  <xsd:simpleType name="idrefType">

    <xsd:restriction base="xsd:IDREF"/>

  </xsd:simpleType>


</xsd:schema>
```

Bibliography

[1] Esther König and Wolfgang Lezius. The TIGER language - a description language for syntax graphs. Part 1: User's guidelines. Technical report, IMS, University of Stuttgart, 2001.

[2] Esther König and Wolfgang Lezius. The TIGER language - a description language for syntax graphs. Part 2: Formal definition. Technical report, IMS, University of Stuttgart, 2001.

[3] Wolfgang Lezius, Hannes Biesinger, and Ciprian Gerstenberger. *TIGERRegistry Manual*. IMS, University of Stuttgart, 2002.

[4] Wolfgang Lezius, Hannes Biesinger, and Ciprian Gerstenberger. *TIGERSearch Manual*. IMS, University of Stuttgart, 2002.

[5] Andreas Mengel and Wolfgang Lezius. An XML-based encoding format for syntactically annotated corpora. In *Proceedings of the Second International Conference on Language Resources and Engineering (LREC)*, volume 1, pages 121–126, Athens, Greece, 2000.