Formats and standards for metadata, coding and tagging

Paul Meurer
The FAIR principles

FAIR principles for resources (data and metadata):

• **Findable** (-> persistent identifier, metadata, registered/indexed)

• **Accessible** (-> retrievable by pid, standardized protocol; metadata always accessible)

• **Interoperable** (use formal, accessible, shared, broadly applicable language for knowledge representation; vocabularies following FAIR principles)

• **Reusable** (metadata have accurate and relevant attributes; clear license; information on provenance; metadata meet domain relevant community standards)
CLARIN: CMDI metadata

CMDI (Component MetaData Infrastructure): XML-based metadata format (ISO standard) to describe a resource and make it findable

- CMDI templates to choose from
- Persistent identifiers
- Standardized hierarchical description
- Makes resource findable through e.g. VLO (Virtual Language Observatory)
L2 corpora, ASK

**L2 corpora**: Particularly important to link text to information about the learner and the text production task context

- -> encoding of core (learner and task) metadata

Can be achieved by e.g. using XML/TEI.

- XML: machine-independent text-based coding standard, much used to encode corpus data
- TEI: XML-based guidelines for structural encoding with meaningful (semantic) markup
Encoding format: TEI

TEI header: encoding of metadata
Person- and task-related metadata is encoded in profileDesc/particDesc/person, as a list of <p> elements:

<profileDesc>
  <particDesc>
    <person>
      <p id="01" n="pid">h0004</p>
      <p id="02" n="testyear">2000</p>
      <p id="03" n="testlevel">Høyere nivå</p>
      <p id="04" n="country">Nederland</p>
      <p id="05" n="language">nederlandsk</p>
      <p id="06" n="age">45</p>
      <p id="07" n="gender">kvinne</p>
  </person>
</particDesc>
</profileDesc>
Encoding: possible improvements

- Should stick to English or multilingual attributes and values
- Use a restricted vocabulary for atts and values
  - either agreed-on in the L2 community,
  - or based on some public external vocabulary like OpenSKOS
- Using `<p>` elements to encode flat attribute-value pairs seems OK, since TEI does not provide structured elements that fit our purpose.
- They are also straight-forward to feed into corpus applications
- But distinguishing person, task-related and bookkeeping metadata could be advantageous
- Also attributes that are constant for a given corpus/project should be encoded
Error encoding in ASK

Encoding device: `<sic>` elements, with attributes `type`, `desc` and `corr`.

```xml
<sic type="F" desc="AGR" corr="stilling">stillingen</sic>
```

`<sic>` elements can nest:

```xml
<sic type="O">før var kvinner
    <sic type="INFL">undertrukket</sic>
</sic>
```

Improvements:
- Better attribute names (‘desc’ means subtype)
- Different error classification/granularity?
Grammatical annotation

ASK: Text is tokenized and split into sentences
Morphosyntactic annotation with POS, morphological features, syntactic relations
(Obs.: morphosyntactastic annotation is not always accurate, is based on corrected version; it should only be used to guide your searches)

<word lemma="dette" features="pron nøyt ent pers @subj">
    dette
</word>
Grammatical annotation: improvements

• **ASK**: Bag of tags, Norwegian names for grammatical features (taken from Oslo-Bergen Tagger)

• Other corpora:
  – **Šolar**: array of characters, each position encoding one feature (e.g., ‘Sometn’). Cryptic.
  – **Teitok**: POS only?

• **But**: Better interoperability obtained using standardized tags

• Possible choices:
  – Eagles tagset (outdated?)
  – Universal dependency POS, feature set and relations
    • e.g., VERB tense=Past
  – Some bag of tags solution

• Advantage of bag of tags: easy to use in corpus tool (but less self-explanatory)
ASK: Addressing of text positions

Sentence encoding: <s sid="h0004:s19">

• Document id (pid), sentence id (sid) and word position in sentence are used as reference points to uniquely address a word in a text.

• Address stays the same also when grammatical or error coding is changed.

• Important for user-generated corpus annotation when corpus is reindexed
User-generated annotation

Annotate phenomena that are not coded in the corpus and cannot be automatically searched for.

Two variants:

• Free-text annotation

• Annotation with a restricted vocabulary (feature-value pairs)

Important: ability to search for annotations
Annotation: use case

Annotate stranded prepositions (simplified)

• Devise an annotation feature ‘stranded’ with two values: yes, no
• Search for prepositions followed by comma or full stop
• Go through list in KWIC and classify
• Search like [feature=("stranded:yes")]] etc.