PDEN 2019

Corpus encoding

Simon Gabay

11 décembre 2019

XML

Le XML

Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable.

Cf. Wikipedia

Basic rules

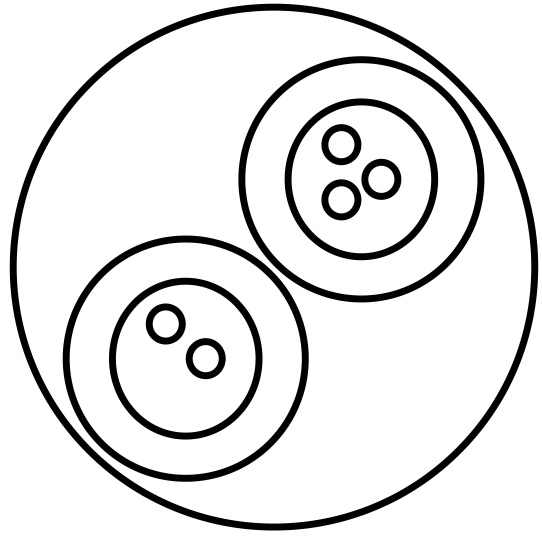
Very simple rules

```
<element attribute="value">data</element>
```

1. An `<element>` is surrounded by angle-brackets
2. A start `<tag>` is followed by an end `</tag>`
3. If we use a first tag `<tag1>` and then a second one `<tag2>` we cannot close the first `</tag1>` and then the second `</tag2>`
4. A `<tag/>` can be empty (self-closing tag)
5. An `<element>` can have an @attribute (cf. `@`)
6. The @attribute has a "value" (between by quotation marks)

From text to data

1. Data is inbetween two tags. For us, data are books, chapters, sentences, words, even characters...
2. These data are "encased" one in the other: a book contains chapters, whoch contains paragraphs, which contains sententes...



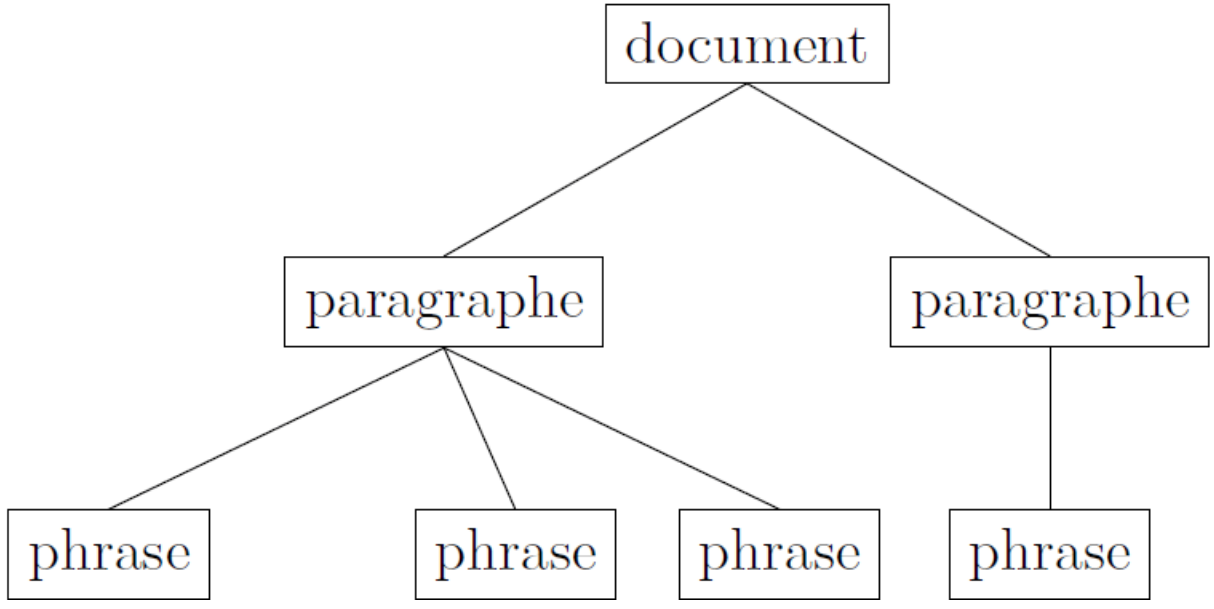
3. Doing so, we transform text into data

Tree structure

Exemple:

> On emploie a priori les italiques pour les termes empruntés à d'autres langues. On emploie les petites capitales pour les noms propres, comme Léopold Delisle. On emploie en revanche généralement le gras pour des raisons coupables.
> On retourne à la ligne pour un nouveau paragraphe.

Underlying structure:



XML as a structured language

```
<document>
  <paragraphe>
    <phrase>
      On emploie <locutionÉtrangère>a
      priori</locutionÉtrangère> les italiques pour les
      termes empruntés à d’autres langues.
    </phrase>
    <phrase>
    On emploie les petites capitales pour les noms
    propres,comme <nom>Léopold Delisle</nom> ou
    <nom>Jules Quicherat</nom>.
    </phrase>
    <phrase>
      On emploie en revanche généralement le gras pour
      des raisons coupables.
    </phrase>
  </paragraphe>
  <paragraphe>
    <phrase>
      On retourne à la ligne pour un nouveau paragraphe.
    </phrase>
  </paragraphe>
</document>
```

An important question

1. Until now we have use `<paragraphe>` or `<phrase>`, but we could have chosen something else.
2. If we were italian, we could have used `<paragrafo>` or `<frase>`
3. But then documents are encoded differently: how can we choose `<elements>` and `@attributes` that have the same name everywhere?

TEI

The TEI

- TEI means *Text Encoding Initiative*
- Invented in 1987 (before internet)
- Directed by a consortium which collectively develops and maintains a standard for the representation of texts in digital form.
- These guidelines are in constant evolution
- They are freely available online: http://www.tei-c.org/guidelines/

Between vocabulary and language:

There are other vocabularies

- EAD (*Encoded Archival Description*) for archivists
- DC (*Dublin Core*) for librarian
- TMX (*Translation Memory eXchange*) for translators

Some of these vocabularies can be expressed with other languages (*e.g.* RDF-DC in turtle).
For this reason, we talk about *XML-TEI* (a long time ago, there was a *SGML-TEI*)

Three things to know

1. It is an english vocabulary : we use a `<w>` (*word*) for a `<w>word</w>`
2. It is limited: we (almost) cannot invent new tags
3. It offers a semantic way to encode texts (not like LaTeX).

Sémantique et procédural

> On emploie *a priori* les italiques pour les locutions et termes empruntés à d'autres langues.

Presentational markup

```
On emploie <italique>a priori</italique> les italiques pou
locutions et termes empruntés à d'autres langues.
```

Descriptive markup

```
On emploie <locutionEtrangère>a priori</locutionEtrangère>
italiques…
```

Descriptive markup II

```
On emploie <latin>a priori</latin> les italiques...
```

In TEI

```
On emploie <foreign xml:lang="la">a priori</foreign> les italiques...
```

ELTeC Transcription Guidelines

The objective

We need to capture a minimum of features for each text:

- significant structural features (chapters, headings, paragraphs...)
- descriptive metadata (bibliographic and non bibliographic)

Considering the amount of text that will be encoded, we cannot offer the finest granularity: we need to choose elements in the TEI (c. 30 out of 450)

Obviously such a project raises many questions as to which features should be captured.

What is essential?

- Are we interested in the physical structure ? (book, page, line…)
- Are we interested in the logical structure ? (parts, chapters, paragraphs…)
- Both?
- Do we want to capture polyphonic effects (direct and indirect speech……)

Basic elements: structure

- `<pb>` when a page begins
- `<front>` for liminal material (preface, forword...)
- `<div>` for major structural divisions (parts, sections, chapters etc.)
- `<div>` can have a `@type` to indicate the function of a structural division with the following values: `liminal`, `titlepage`, `notes`, `part`, `chapter`

Basic elements 2: the text

- `<head>` for the title of a chapter, or of any other subdivision
- `<p>` element is used for everything which is typeset as a separate block on the page, including both paragraphs and list items
- `<l>` element is used for verse lines or similar, typically set off from the rest of the text
- `<hi>` indicates the presence of typographic salience (superscript, italics…)

Reference scheme

- text identifier : every text will have an identifier consisting of its three letter language code in upper case and a five digit serial number, for example `FRA00042`
- chapter identifier: each chapter or equivalent will have an identifier concatenating the text identifier and a three digit serial number, for example `FR042012` is the twelfth chapter of the 42nd French novel.
- If sub-chapter segmentation is implemented, then the segments will append a further four digit serial number.
- The identifier will be supplied as the value of an xml:id attribute on each `<text>`, `<div>`, `<front>`, `<back>`, or `<s>` element as appropriate.

Additional remarks

We do not preserve the lineation of running prose in our source texts, since this is always purely an artefact of the source edition. For the same reason we reassemble words broken across a line break, silently removing any hyphen present.