# 9 Standards pour l'annotation des corpus

## 9.1 TEI

Text Encoding Initiative (TEI): projet international lancé en 1988 pour développer des conventions d'encodage des textes éléctroniques pour la recherche.

Publications des conventions en 1994. (voir page web du cours)

Conventions TEI en SGML, c'est-à-dire les conventions consistent en un DTD et des recommandations qui expliquent la sémantique des étiquettes dans le DTD.

Caractéristiques de TEI:

1. très détaillé et très varié: permet d'encoder beaucoup d'aspects différents et offre souvent plusieures façon d'annoter un aspect d'un texte
2. permet l'encodage des origines d'un texte d'une manière compatible avec des conventions suivies dans les bibliothèques
3. couvre différents types de textes (prose, poèmes, drames, lettres, dictionnaires, textes oraux, hypermédia) textes).

## 9.2 CES

CES (Corpus Encoding Standard), adaption de TEI aux besoins d'annotation des corpus, fait par EAGLES.

Un corpus annoté avec CES consiste d'un header suivi de

- soit un ou plusieurs document
- soit un ou plusieurs sous-corpus.

Un document consiste d'un header suivi d'un texte:

```
<!ELEMENT cesCorpus     - -  (cesHeader,(cesDoc+ | cesCorpus+)) >
<!ELEMENT cesDoc        - -  (cesHeader, text)                  >
```

### 9.2.1 Structure d'un header

Le fichier avec la définition du header se trouve sous:

`http://www.cs.vassar.edu/CES/sgml/header.elt`

Le header contient quatre éléments:

1. une déscription bibliographique du corpus ou du texte (`fileDesc`)
2. une déscription de la relation entre le texte éléctronique et le texte source (`encodingDesc`)
3. une partie qui contient des informations supplémentaires sur un texte (`profileDesc`)
4. l'histoire de révisions du corpus (`revisionDesc`)

Structure d'un `fileDesc`:

```
<!ELEMENT fileDesc      - -  (titleStmt, editionStmt?,
                             extent?, publicationStmt,
                             sourceDesc+ )                      >
```

**Exemple 28**

```
      <fileDesc>
```

```
        <titleStmt>
            <h.title>Machine-readable version of 1984, ch. 1</h.title>
            <respStmt>
                 <respType>annotated with CES tags </respType>
                 <respName>A. Student</respName>
            </respStmt>
        </titleStmt>
        <extent>
            <wordcount>6571 </wordcount>
            <bytecount units="bytes">6571 </bytecount>
        </extent>
        <publicationStmt>
            <distributor>Laboratoire Parole et Langage, CNRS</distributor>
            <pubAddress>...</pubAddress>
            ...
            <availability status=restricted>
                internal use only--cannot be distributed</availability>
            <pubDate>...</pubDate>
        </publicationStmt>
        <sourceDesc>
            <biblStruct>
                 <monogr>
                      <h.title>Nineteen Eighty-four</h.title>
                      <h.author>George Orwell</h.author>
                      ...
                 </monogr>
            </biblStruct>
        </sourceDesc>
    </fileDesc>
```

Dans la déscription de l'encodage (`encodingDesc`), on indique

- le projet et le but pour lequel le corpus a été annoté
- le choix des textes fait dans le corpus
- la façon d'éditer le corpus/texte, c'est-à-dire si on a fait des corrections ou des normalisations, si on a gardé la coupure des mots etc.
- les étiquettes utilisées dans le corpus/texte
- la façon de définir des références dans l'annotation
- une définition des classes de textes qui peuvent être utilisées pour classifier les textes dans le corpus

```
<!ELEMENT encodingDesc  - -  (projectDesc, samplingDecl*,
                             editorialDecl*, tagsDecl?,
                             refsDecl*, classDecl?)            >
```

La déscription du profile d'un corpus/texte (`profileDesc`) indique

- l'origine d'un texte
- les langues utilisées
- les caractères figurants dans le corpus/texte
- les classes de texte d'un corpus/texte
- s'il existe des traductions d'un texte
- et s'il existe des annotations dans des fichiers à part (p.e. un alignement entre le texte et une traduction)

Tous les éléments de `profileDesc` sont optionnels.

### 9.2.2 Structure d'un texte

Le DTD CES avec les définitions pour les documents se trouve sous:

`http://www.cs.vassar.edu/CES/sgml/cesDoc.dtd`

Dans le DTD on a d'abord défini des entités qui peuvent apparaître à plusieures positions dans le DTD:

```
<!ENTITY % m.inter ' bibl | quote | list |
                    poem | note | caption | figure | table '    >

<!ENTITY % m.token  'abbr | date | num |
                     measure | name | term | time |'            >

<!ENTITY % m.phrase '%m.token; corr | distinct | foreign |
                     gap | hi | list | mentioned | ptr |
                     q | ref | reg | s | title'                 >

<!ENTITY % base.seq   '(#PCDATA | num | abbr)*'       >

<!ENTITY % phrase.seq '(%m.phrase; | #PCDATA)*'                 >

<!ENTITY % par.seq    '(p | sp | %m.inter;)*'                   >
```

Dans les défintions d'éléments on réfère à ces entités:

```
<!ELEMENT text          - -  (body | group)                     >
<!ELEMENT body          - -  (%par.seq;, div*)                  >
<!ELEMENT group         - -  (%par.seq;, body+)                 >
```

Une division (`div`) se compose de

- 0 ou plusieurs dates, mots de clés, titres ou noms d'auteurs,
- soit une suites de paragraphes peut-être suivies d'autres divisions, soit 1 ou plusieurs divisions
- et 0 ou plusieurs dates, mots de clés, phrases ou noms d'auteurs.

Au niveau des paragraphes, CES définit les éléments suivants:

1. Captions (titres dans des tableaux, figures, etc.)
2. citations `quote`
3. dialogues (spoken paragraphes `sp`)
4. poèmes
5. listes,
6. figures,
7. annotations (notes)
8. tableaux.

Au niveau sous les paragraphes (niveau des phrases) on trouve

1. des éléments linguistiques
2. information de layout
3. des corrections

4. phrases et citations q
5. des pointeurs et des références.

### 9.2.3 Chevauchements d'éléments

Une limitation de SGML est qu'on doit toujours obtenir une structure hiérarchique (structure d'arbe). Mais parfois on a des chevauchements d'éléments, p.e. pour phrases et citations.

**Exemple 29** *D'après le directeur l'économie va "mieux que jamais. Elle va en fait très bien."*

On ne peut ni annoter des phrases s comme sous-éléments des citations q ni des citations comme sous-éléments des phrases.

Solution adoptée dans CES: Coupure des phrases ou des citations qui s'étendent sur plus qu'une citation/phrase.

```
<!ELEMENT s            - -  (%phrase.seq;)                  >
<!ATTLIST s                 %a.text;
          next              IDREF              #IMPLIED
          prev              IDREF              #IMPLIED
          broken            (yes | no)         no           >


<!ELEMENT q            - -  (%phrase.seq;)                  >
<!ATTLIST q                 %a.text;
          next              IDREF              #IMPLIED
          prev              IDREF              #IMPLIED
          broken            (yes | no)         no           >
```

Dans %a.text; il y a un attribut id.

**Exemple 30**

```
<s>D'apr&egrave;es le directeur l'&eacute;conomie va
        <q id=q1 next=q2 broken=yes>mieux que jamais.</q></s>
<s><q id=q2 prev=q1 broken=yes>Elle va en fait tr&egrave;s bien.</q></s>.
```

### 9.2.4 Annotation morpho-syntaxique dans CES

2 solutions possibles:

1. Définir un élément pour une étiquette:

```
<tok>
   <orth>il</orth>
   <ctag>CL</ctag>
</tok>
```

2. Définir un attribut pour l'étiquette:

```
<w ctag="CL">il</w>
```

Avantage de la deuxième solution: texte sans markup = texte original

# 10 XML

XML = eXtensible Markup Language

## 10.1 Le DTD

XML est un sous-ensemble de SGML, plus stricte que SGML

Différences entre XML et SGML:

1. Dans XML les tags fermants sont toujours obligatoires, dans SGML cela dépend du DTD.
2. Les éléments vides en XML consistent d'un seul tag qui se compose de `<` suivi du nom du tag suivi de `/>`.

   SGML:

   ```
   <editionStmt version='1'>
   ```

   XML:

   ```
   <editionStmt version='1'/>
   ```

3. En XML, toutes les valeurs d'attributs doivent être mises entre guillemets (`"` ou `'`)
4. XML distingue entre minuscules et majuscules á l'égard des noms d'éléments, SGML ne le fait pas.
5. XML ne permet pas d'exception (*inclusion* ou *exclusion exception*) comme le fait SGML. Une d'éfinition de `sp` comme dans CES n'est pas possible en SGML.

   Incorrect en XML:

   ```
   <!ELEMENT sp  - -   (speaker?, p+)      +(stage) >
   ```

## 10.2 XSL

XSL = Extensible Style Language: langage utilisé pour définir les feuilles de style associées aux documents XML. C'est le fichier XSL qui permet de définir que tel élément XML doit être affiché avec telle fonte, de telle couleur, etc.

XSL est souvent utilisé pour afficher HTML.

XSL permet aussi de formuler une transformation d'une structure XML en une autre.