17 – Documentation technique

Bien que ce document soit axé sur l'utilisation de la TEI pour le codage de documents existant déjà sous forme électronique, la même approche peut également servir pour le codage de nouveaux documents. Dans la préparation de nouveaux documents (tels que celui-ci), SGML a beaucoup d'attraits : la structure du document peut être représentée clairement et le même texte électronique peut être réutilisé pour maintes fins (par exemple pour produire à la fois des versions en ligne – en hypertexte ou consultables – et des versions imprimées bien formatées et mises en pages, le tout à partir d'une source SGML commune).

Pour faciliter ceci, un nombre limité d'éléments supplémentaires sont inclus dans la TEI Lite en tant qu'extensions du DTD principal de la TEI ; ils serviront à marquer les caractéristiques particulières de documents techniques en général et des documents SGML en particulier.

17.1. Eléments supplémentaires pour les documents techniques

Les éléments suivants peuvent servir pour marquer les caractéristiques particulières de documents techniques :

<eg> regroupe un exemple court de notation technique, par exemple un code fragmenté ou un échantillon de codage SGML ;

<code> regroupe un fragment code écrit dans une certain langage formel (souvent un langage de programmation) ;

<ident> contient un identifiant d'un certain type, par exemple un nom variable ou le nom d'un élément ou d'un attribut SGML ;

<gi> contient un type spécial d'identifiant : un identifiant générique SGML, ou un nom d'élément ;

<kw> contient un mot clé dans une certaine langue formelle ;

<formula>

contient une formule chimique ou mathématique, présentée facultativement dans une quelconque notation non-SGML ; parmi les attributs possibles, citons :

notation précise la notation employée pour représenter le corps de la formule ; la valeur par défaut est `tex`, signifiant que la formule est représentée au moyen du système de composition TEX.

Les exemples suivantes indiquent une utilisation possible de ces éléments, à savoir, le codage d'un passage faisant partie d'un cours de présentation du langage de programmation Fortran :

```
<p>Il est de tradition de présenter un langage de
 programmation à l'aide d'un premier exemple comme :
 <eg>
    CHAR*20 GRTG
    GRTG = 'BONJOUR TOUT LE MONDE'
    PRINT *, GRTG
    END
 </eg></p>
 <p>Dans cet exemple, on commence par déclarer la variable
 <ident>GRTG</ident>, dans la ligne  <kw>CHAR*20 GRTG</kw>,
  qui identifie <ident>GRTG</ident> comme formée de 20 octets
  de type <kw>CHAR</kw>.  On affecte alors à cette variable
  la valeur  <mentioned>BONJOUR TOUT LE MONDE</mentioned>.
  Suivent alors l'ordre d'impression <kw>PRINT</kw> et
  l'instruction finale <kw>END</kw>.
```

Une application de formatage ayant à traiter un tel texte pourrait être programmée de façon à formater convenablement des exemples (par exemple, en conservant les coupures de ligne, ou en employant une police distinctive). Parallèlement, l'emploi de balises telles que *<ident>* et *<kw>* facilite grandement la création d'un index utile.

L'élément *<formula>* devrait servir à entourer une formule chimique ou mathématique présentée au sein du texte comme étant un passage distinct. Puisque les formules comportent généralement une grande variété de caractéristiques typographiques spéciales qui ne figurent pas ailleurs dans le texte courant, il sera habituellement nécessaire de présenter le corps de la formule dans une notation spécialisée. La notation employée devrait être spécifiée par l'attribut *notation*, comme dans l'exemple suivant :

```
 <formula notation=tex>
   \(E = mc^{2}\)
 </formula}
```

La notation *tex* est pre-définie pour la DTD de la TEI Lite ; d'autres notations peuvent être employées si besoin est, mais elles doivent être définies au préalable au moyen d'une mention du type «notation» dans la DTD.

Presque toutes les séquences de caractères sont admises dans le corps d'un élément <*formula*>, du moins du point de vue d'une application capable de traiter des données SGML. Les données sont transférées sans modification par l'analyseur syntaxique à toute application associé à la notation spécifiée. La seule exception à cette règle est que l'analyseur syntaxique reconnaîtra tout objet qui ressemble au début d'une fin-de-balise SGML, c'est-à-dire le caractère «plus petit que» (**<**) suivi aussitôt par une barre oblique (/) et un caractère alphabétique. En traitant l'exemple imaginaire ci-dessous, un analyseur syntaxique SGML génèrerait toute une suite de messages d'erreurs.

```
<formula notation=tex>
  \(E = mc^{2}</a\)
</formula>
```

Heureusement , la séquence «**</** » est plutôt rare dans la majorité des notations mathématiques utilisées actuellement. Néanmoins, si elle apparaît, il est nécessaire de prendre des dispositions qui dépassent le cadre de ce document (pour plus d'informations, se reporter au texte complet des recommandations de la TEI).

Ce problème existe sous une forme plus aiguë lorsque le codage SGML lui-même est le sujet de discussion à l'intérieur d'un document technique, lui-même codé en SGML. Dans un tel document, il est évidemment essentiel de faire une distinction entre le balisage SGML contenu dans les exemples, et celui qui est employé pour le balisage du document lui-même ; dans ces textes, en effet, l'emploi de balises de fin est très vraisemblable. La solution la plus générale consiste à marquer le corps de chaque exemple SGML comme contenant des données qui ne doivent pas être balayées par l'analyseur syntaxique pour des fins de balisage SGML. Ceci est fait en l'entourant d'une structure SGML spéciale appelée *CDATA marked section*, comme dans l'exemple suivant :

```
<p>Une liste devrait être codée comme suit:
<eg><![ CDATA [
   <list>
   <item>Premier élément de la liste</item>
   <item>Second élément</item>
   </list>
]]>
</eg>
Les éléments  <gi>list</gi> sont composés d'une série
d'éléments <gi>item</gi>.
```

L'élément <*list*> employé dans l'exemple ci-dessus ne sera pas considéré comme faisant partie du document proprement dit, parce qu'il est inséré à l'intérieur d'une section balisée (qui commence avec la mention spéciale de balisage `<! [ CDATA [`, et qui se termine avec `] ]>`).

À noter également l'utilisation de l'élément <*gi*> pour baliser les références à des noms d'éléments SGML (ou des *identifiants génériques*) au sein du corps du texte.

17.2. Divisions générées automatiquement par les outils bureautiques

La plupart des systèmes modernes de production de documents sont capables de générer automatiquement des sections entières telles qu'une table de matières ou un index. La TEI Lite fournit un élément pour marquer l'emplacement d'une section générée de cette façon.

<*divGen*>
: indique l'emplacement prévu pour une division textuelle générée automatiquement par une application de traitement de texte ; parmi les attributs possibles, citons :

  type : précise le type de la division de texte prévue (par exemple, un index, table de matières etc.) ; exemples : `index` (un index doit être généré et inséré à ce point), `toc` (une table des matières), `figlist` (une liste de figures) et `tablist` (une liste de tables).

L'élément <*divGen*> peut être placé partout où un élément de division serait autorisé, comme dans l'exemple suivant :

```
<front>
<titlePage>... </titlePage>
<divGen type=toc>
<div type='Preface'><head>Preface</head>... </div>
</front>
<body>... </body>
<back>
<div1><head>Appendix</head>... </div1>
<divGen type=index n='Index'>
</back>
```

Cet exemple montre également l'emploi de l'attribut *type* pour distinguer les différentes sortes de division à générer : dans le premier cas une table des matières (*toc*) et dans le second un index.

Lorsqu'une table des matières ou un index existant doit être codé (plutôt que généré) pour une raison quelconque, il convient d' employer l'élément <*liste*> présenté dans la section 12 «Listes».

17.3. Génération d'index

Tandis que la génération d'une table des matières à partir d'un document correctement balisé se passe généralement sans problèmes pour un processeur automatique, la production d'un index de bonne qualité nécessitera dans bien des cas un balisage plus réfléchi. Il peut ne pas être suffisant de produire une simple liste de toutes les parties balisées d'une certaine façon, bien que le fait d'extraire (par exemple) toutes les occurrences d'éléments tels que <*terme*> ou <*nom*> soit souvent un bon point de départ pour un index.

La DTD de la TEI fournit un balise <*index*> spéciale qui peut servir pour indiquer à la fois les parties du document qui devrait figurer dans l'index, et la façon dont l'indexage devrait être fait.

<*index*>
: marque un emplacement à indexer pour une certaine raison ; parmi les attributs possibles, citons :

  level1 donne la forme principale de l'entrée d'index ;

  level2 donne la forme du second niveau, s'il existe ;

  level3 donne la forme du troisième niveau, s'il existe ;

  level4 donne la forme du quatrième niveau, s'il existe ;

  index indique à quel index (entre plusieurs) appartient l'entrée d'index.

Par exemple, le deuxième paragraphe de la présente section pourrait présenter le balisage suivant :

```
...
La DTD de la TEI fournit une balise <gi>index</gi> tag
<index level1='indexing'>
<index level1='index (tag)' level2='use in index generation'>
 spéciale qui peut servir ...
```

L'élément <*index*> peut également servir pour fournir une forme d'information analytique ou interprétative. Par exemple, dans une étude d'Ovide, on pourrait vouloir enregistrer toutes les références du poète concernant les différents personnages, pour des besoins d'étude stylistique comparative. Dans les lignes suivantes des *Métamorphoses*, une telle étude enregistrerait les références du poète à Jupiter (comme *deus*, *se* et en tant que sujet de *confiteor* [sous la forme inflectionnelle 227]), à Jupiter en guise de taureau (*imago tauri fallacis* et sujet de *teneo*), et ainsi de suite[1]. L'exemple a été légèrement simplifié.

```
<l n=3.001>iamque deus posita fallacis imagine tauri
<l n=3.002>se confessus erat Dictaeaque rura tenebat
```

Cet objectif pourrait être atteint au moyen de l'élément <*note*> présentée dans la section 7 « Notes » ou au moyen de l'élément <*interp*> présenté dans la section 16 «Interprétation et Analyse». Ici nous montrons le moyen d'obtenir le même résultat avec l'élément <*index*>.

Nous supposons que l'objet doit générer plus d'un index : un pour des noms de divinités (appelé *dn*), un autre pour des références onomastiques (appelé *on*), un troisième pour les références pronominales (appelées *pr*), et ainsi de suite. Une façon d'y parvenir est indiquée ci-dessous :

```
<l n=3.001>iamque deus posita fallacis imagine tauri
     <index index="dn" level1="Iuppiter" level2="deus">
     <index index="on" level1="Iuppiter (taurus)"
                       level2="imago tauri fallacis"></l>
<l n=3.002>se confessus erat Dictaeaque rura tenebat
     <index index="pr"    level1="Iuppiter" level2="se">
     <index index="v"     level1="Iuppiter" level2="confiteor
                          (v227)">
     <index index="mons"  level1="Dicte" level2="rura Dictaea">
     <index index="regio" level1="Creta" level2="rura Dictaea">
     <index index="v"     level1="Iuppiter (taurus)"
                          level2="teneo (v9)"></l}
```

Pour chaque élément <*index*> ci-dessus, une entrée sera générée dans l'index approprié, en employant comme mot principal la valeur de l'attribut *level1* et comme mot clé secondaire celui de l'attribut *level2* qui contient le mot cité sous sa forme nominative. La référence elle-même sera prise dans le contexte où figure l'élément <*index*>, c'est-à-dire dans le cas présent, l'identifiant de l'élément <*l*> qui le contient.

1. L'analyse est empruntée, avec autorisation, à Willard McCarty et Burton Wright, *An Analytical Onomasticon to the Metamorphoses of Ovid* (Princeton: Princeton University Press, à paraître).