Jérôme FRARESSO
Frédéric GRANDCLAUDE
Stéphane MAGNIER
Emmanuel SIMHA

Traducteur SGML vers LaTeX

SPECIFICATIONS

Dans le cadre de la traduction d'un langage vers un autre, la littérature nous indique que le traducteur doit avoir certaines caractéristiques que l'on qualifiera de génériques. En effet, tout traducteur se doit de :

- savoir reconnaître les commandes des données proprement dites,
- savoir traduire les commandes,
- être capable de transcrire les données intelligemment, c'est à dire en tenant compte du contexte défini par les commandes.

Notre analyse du fichier source SGML nous a permis de comprendre la structure d'un fichier SGML. Ce dernier est scindé en deux parties :

- une première contient les déclarations de macro-commandes,
- la deuxième contient les données proprement dites (en ce qui nous concerne, le texte et les symboles mathématiques codés par leur équivalent ASCII) et des appels aux commandes SGML de mise en page prédéfinies (l'appel à une commande se trouvant dans des balises délimitées par des crochets “>” et “<”).

Cette disposition nous a permis de définir une stratégie de traduction du document SGML en LaTeX, à savoir une conversion en deux étapes. En effet, tout en répondant aux caractéristiques génériques des traducteurs, cette stratégie s'est avérée bien adaptée au travail en équipe de conception sur le papier ainsi qu'à la multiprogrammation. De plus, une telle stratégie nous a permis de scinder le problème, facilitant ainsi sa résolution.

1. En commençant l'analyse du document SGML à la suite des déclarations (le début de la zone à traduire étant signalé par la

balise <“paragraphe” >), on sépare les données des appels au commandes. On écrit ensuite dans un fichier temporaire (baptisé fich.tmp), les données telles qu'elles sont écrites, et à la place des commandes SGML (de mise en page ou de passage en mode mathématique) on place un équivalent tel qu'il puisse lever toute ambiguité. Chaque commande SGML est ainsi codée par un caractère d'une table ASCII étendue, caractère que l'on ne doit bien sûr pas trouver dans les données.

2. On parcourt le fichier temporaire en lecture. On traduit chaque code de mise en page par son équivalent LaTeX, les données étant quant à elles réecrites:

   - tel quel en ce qui concerne le texte,
   - avec le symbole mathématique LaTeX ad hoc lorsqu'un symbole mathématique codé par son code ASCII équivalent est reconnu.

I. Description de la première passe

On utilise:

- Une fonction SGML2TMP
  - prenant comme argument un fichier SGML (fich.sgml),
  - créant un fichier fich.tmp,

  qui réalise la conversion du fichier fich.sgml en un fichier fich.tmp
- une fonction TESTCHAINE
  - prenant comme argument une chaîne de caractères
  - renvoyant un entier.

a) SGML2TMP

- Ouvrir fich.tmp en écriture.
- C1 $\leftarrow$ 1 (début du fichier fich.sgml) .
- Drapeau $\leftarrow$ False. (drapeau indique si nous sommes dans la zone du fichier SGML contenant les données).
- Tant que fin du fichier fich.sgml n'est pas atteinte, faire:
  - Tant que caractère lu à la position C1 est différent de "<" faire:
    - Si drapeau=True, ecrire caratère à la position C1 du fichier fich.sgml dans fich.tmp et C1 $\leftarrow$ C1 + 1.
    - Si caractère lu à la position C1 du fichier fich.sgml = fin de fichier alors Exit.
  - chaîne $\leftarrow$ NULL.
  - C2 $\leftarrow$ C1 + 1.
  - Tant que le caractère lu à la position C2 dans fich.sgml est $\neq$ ">" et chaîne $\neq$ "frame" et caractère lu à la position C2 est $\neq$ fin de fichier, faire:
    - C2 $\leftarrow$ C2 + 1.
  - chaîne $\leftarrow$ extraitchaîne(C1,C2)
  - code $\leftarrow$ TESTCHAINE(chaîne).
  - Si code=0 et drapeau=vrai faire
    - Si fich.sgml(C1 à C1+6)="Frame", alors faire
      - chaîne2$\leftarrow$""
      - C3$\leftarrow$C1

            - Tant que caractère lu à la position C3$\neq$ “ “ ” faire C3$\leftarrow$C3+1
            - C4$\leftarrow$C3+1
            - Tant que caractère lu à la position C4$\neq$ “ ” ” faire C4$\leftarrow$C4+1
            - chaîne2 $\leftarrow$ extraitchaîne(C3,C4)
            - code2$\leftarrow$TESTCHAINE(chaîne2)
            - Si code2=0 écrire chaîne2 dans fich.tmp
            - Si code2$>$127 écrire code2 dans fich.tmp
            - C1$\leftarrow$C2+1
        - Sinon faire
            - écrire “$<$” dans fich.tmp
            - C1$\leftarrow$C1+1
    - Si code=1 faire drapeau$\leftarrow$VRAI
    - Si code$>$127 écrire code dans fich.tmp
- Fermer fich.tmp et fich.sgml.

En résumé, les commandes entre $<$ et $>$ sont codées par un entier entre 128 et 255 lorsqu'elles ont reconnues.
Si elles ne sont pas reconnues, on les transcrit telles quelles en ASCII dans le fichier temporaire.

- Dans le cas d'une commande $< Frame, \ldots >$, si le programme ne localise pas de chaîne entre guillemets, elle est oubliée (on passe à la suite);
- dans le cas contraire (il existe une chaîne entre les guillemets), cette chaîne est codée par un entier entre 128 et 255 si elle est présente dans la liste de hachage, ou transcrite telle quelle en ASCII dans fich.tmp si elle n'y figure pas.

b) TESTCHAINE

Cette fonction s'applique sur une chaîne de caractères et renvoie un entier qui prendra pour valeur:

- 0 si la chaîne n'est pas une commande SGML.
- 1 si la chaîne est égale à “paragraphe”.
- 2 si la chaîne est égale à “Frame,”.
- un entier entre 128 et 255 correspondant au numero de code attribué à la commande en question sinon.

L'algorithme de recherche de chaîne dans la liste des commandes SGML reconnues se fera par l'intermédiaire d'une table de hachage.
La fonction de hachage f sera la partie entière de la somme des codes ASCII des caractères composant la chaîne modulo 128 .
La table de hachage est un tableau de listes chaînées de structures, chaque structure comportant 3 champs.

- Un contenant la commande SGML en ASCII,

- Un autre contenant le numéro correspondant à cette commande,

- Le dernier, un pointeur sur la structure suivante de la liste.

II. Description de la deuxième passe

On utilise:

- Une fonction TMP2TEX
  - prenant comme argument le fichier fich.tmp,
  - créant un fichier fich.tex,
    qui réalise la conversion de fich.tmp en un fichier fich.tex.
- Un tableau SymbSGML2TEX
  - ayant deux champs: mode et code,
  - dont les cases contiennent des chaînes de caractères,
- Un tableau de commandes de mise en page LaTeX Tableau-com[code].

a) TMP2TEX

- ouvrir fich.tmp en lecture.
- ouvrir fich.tex en ecriture.
- mise en forme du fich.tex (écriture des préambules, déclarations...)
- C1 $\leftarrow$ 1 (début de fich.tmp).
- mode $\leftarrow$ 0 (mode texte par défaut).
- Tant que C1 $\neq$ de fin de fichier, faire:
  - code $\leftarrow$ code ASCII du caractère lu à C1.
    Si <u>code < 128</u>
    - si mode = 0 alors
      - si le caractère fait partie de la liste suivante : $, _ , # , & , { , } , % , alors, il s'agit d'un caractère spécial de LaTeXet il faut écrire "\caractère" dans fich.tex,
      - si caractère $\neq$ "<"
        drapeau-inférieur $\leftarrow$ 0.
        écrire caractère dans fich.tex.
      - si caractère = "<"
        si drapeau-inférieur = 0
        écrire "<"; drapeau-inférieur $\leftarrow$ 1.
        sinon drapeau-inférieur $\leftarrow$ 0.

    - sinon (mode $\neq$ 0) écrire la chaîne rendue par symbSGML2TEX(mode,code).

Sinon (code $>$ 128)

  - code←code-128
  - écrire dans fich.tex le premier champ de Tableau-com[code].
  - mode ← le deuxième champ de Tableau-com[code].

- C1 ← C1 + 1
- fin de la mise en forme de fich.tex (écriture de

  `\end{document}`

  ,etc...).

b) SymbSGML2TEX

- En argument:
  - mode: Un entier 1,2 ... pour le symbole 1, symbole 2 ...
  - code: Le code du caractère ASCII, i.e un entier entre 0 et 127.
- En retour: Un pointeur sur une chaîne de caractères se trouvant à la ligne mode et à la colonne code d'une matrice 4 lignes et 128 colonnes. Cette chaîne de caractères est la commande LaTeX devant écrire le symbole mathématique ad hoc.

c) Tableau-com[code]

C'est un tableau:

- de taille 128 car le code varie ici de 128 à 255,
- de structure à deux champs:
  1. Un pointeur sur une chaîne de caractères correspondant à la commande LaTeX de mise en page. Cette chaîne est vide si c'est une "commande" de passage en mode mathématique.
  2. Un entier qui vaut:
     - 0 si c'est une commande de mise en page.
     - 1,2 ... si c'est une "commande" de passage en mode mathématique.