LE FICHIER-TEXTE : UN FICHIER UTILE À LA PROGRAMMATION EN LINGUISTIQUE (application à la langue allemande)

Matériel utilisé : macintosh plus
Langage de programmation : turbo-pascal de borland

Robert LEGALERI

1) PROGRAMMATION LINGUISTIQUE ET FICHIERS

Quel que soit le type de recherche effectué en linguistique, il est probable que, tôt ou tard, il faudra avoir recours à des fichiers. Si l'on fait par exemple de la génération de formes (morphologie), on constatera vite que là où les règles grammaticales s'arrêtent et où par conséquent les exceptions commencent, la programmation devient impossible sans le recours à des listes contenant justement les items faisant exception. Si l'on veut par exemple générer un groupe nominal allemand en se donnant pour base de départ un déterminant, un adjectif épithète et un nom avec son genre, on pourra certes décliner correctement le déterminant et l'adjectif à l'aide d'un nombre restreint de règles, mais il faudra avoir recours à un vaste fichier pour pouvoir décliner le nom, en particulier au pluriel. Des travaux récents (1) à Reims, sous la direction de Jean Petit, utilisent un fichier de substantifs (3 genres réunis) de 3247 enregistrements contenant, outre les pluriels non prévisibles, un certain nombre de terminaisons déclenchant la génération de pluriels très fréquents. Dans certains cas, la présence d'un fichier semble moins impérative mais s'avère plus rentable lorsqu'on y regarde de plus près.

Exemple : la conjugaison des verbes. Les quelque 160 verbes forts que compte la langue allemande peuvent certes être regroupés en trois grands groupes, mais on constate qu'à l'intérieur de ces groupes demeurent quantité d'exceptions dont la programmation à grand renfort d'**IF** et de **CASE OF** serait extrêmement lourde et rendrait le

(1) Florence Wétischek. Modellierung der deutschen Nominalgruppe in der elektronischen Datenverarbeitung (Maîtrise. Reims 1987/88).

programme quasiment illisible. Il sera alors plus économique et plus clair de créer un fichier total des verbes forts contenant, outre l'infinitif, les temps primitifs et éventuellement quelques codes supplémentaires, tels que par exemple l'auxiliaire servant à les conjuguer aux formes composées.

2) LE FICHIER RECORD/END

Quand on pense fichier, on envisage généralement un fichier de type **RECORD/END**, pouvant éventuellement être exploité en accès direct. C'est en effet la forme de stockage qui correspond le mieux aux besoins du linguiste, en particulier dans la mesure où elle permet d'avoir plusieurs champs par enregistrement, chacun de ces champs pouvant contenir des renseignements de types divers (**string**, **integer**, **char**, etc). Exemple :

	champ 1	champ 2	champ 3	champ 4	etc
enregistrement 0	nom singulier	pluriel	particularité 1	particularité 2	
enregistrement 1	infinitif (springen)	prétérit (sprang)	participe 2 (gesprungen)	auxiliaire (sein)	
etc.					

Pourtant, la création et surtout la manipulation de tels fichiers est assez lourde et nécessite de nombreuses précautions :

a) Il faut tout d'abord faire un programme de création de fichier.

b) Il est également souhaitable de faire un programme de relecture du fichier et même un programme d'impression si le fichier est long et si l'on veut pouvoir le consulter à tête reposée dans son ensemble.

c) Une relecture de ce fichier ne sera profitable que si les enregistrements sont classés par ordre alphabétique. Un fichier ordonné est d'autre part indispensable, si l'on veut l'exploiter par la méthode du tri dichotomique (plus rapide que la consultation séquentielle dans le cas de très gros fichiers). Ceci nécessite à nouveau un programme assurant cette mise en ordre (qui sera remise en question en cas d'ajout d'un élément). Notons également qu'il faudra refaire ce travail pour chaque champ, si pour une raison quelconque on désire vérifier rapidement le contenu du champ 2, 3 ou 4 etc.

d) Il faut également prévoir un programme de corrections éventuelles, car dès qu'une ligne est tapée et validée par un retour chariot lors de la création, il est trop tard pour corriger.

e) Il sera enfin indispensable de pouvoir faire une extension, des insertions ou des suppressions, ce dernier aspect étant l'un des plus délicats à cause de la nécessité de "retasser" le fichier pour ne pas laisser d'enregistrements vides.

3) LE FICHIER-TEXTE

Pour la clarté de l'exposé, nous reprendrons point par point les différentes étapes évoquées plus haut :

a) La création du fichier-texte est immédiate, sans utilisation d'un programme spécial de création. Il suffit de disposer d'un support compatible avec le type **text**. Ce sera par exemple :

- l'éditeur de Turbo-Pascal : on ouvre un fichier de travail vierge (dont le nom de sauvegarde sera également le nom de notre fichier-texte) et on tape les enregistrements avec un retour chariot à la fin de chaque enregistrement.
- un traitement de texte permettant de faire une sauvegarde en format "text" (**MacWrite**, **Works** par exemple).
- une base de données, un traitement de fichier, un tableur type **Excel**, avec option sauvegarde en format **text**.

b) La relecture se fait instantanément puisque nous sommes en fonction de traitement de texte. On peut donc relire son fichier comme une lettre, en utilisant les flèches de déplacement ou les "ascenseurs" si la liste est longue. On peut l'imprimer en totalité (ou en imprimer seulement une sélection) en utilisant la fonction d'impression du logiciel qui a servi à le créer.

c) Etant donné que le fichier peut être créé directement dans une base de données ou exporté dans cette base s'il a été créé ailleurs, il va par conséquent pouvoir être soumis à tous les tris que ces logiciels proposent à l'utilisateur, et ce sur n'importe quel champ. On pourra en particulier classer par ordre alphabétique, utiliser des fonctions telles que :

commençant/finissant par

contenant/ne contenant pas

plus grand que (valeur ASCII), plus petit que, égal, différent, etc.

poser des conditions, lier les fonctions par des opérateurs logiques (et, ou etc.). Ceci peut être d'un grand secours, quand, après avoir découvert une anomalie lors du fonctionnement d'un programme (de conjugaison par

exemple), on veut trouver rapidement tous les verbes qui risquent de poser le même problème (ex. présence d'un ß, radical terminé par **-d**, **-t**, **-z**, **-el**, **-er**, **-ier**).

Ces fonctions de traitement de fichier règlent du même coup les problèmes que nous avons soulevés sous les rubriques (d) et (e). Corrections, insertions, extensions, suppressions sont immédiates. Lorsque le fichier-texte est au point, un simple programme de transfert, que l'on peut aisément fabriquer soi-même, permettra de le transformer en fichier séquentiel type **record/end**, donc susceptible d'être exploité "en accès direct" à l'aide de l'instruction **seek**. Ce nouveau fichier n'aura jamais à être relu, ni corrigé, ni augmenté, ni diminué. Son seul travail consistera à être appelé par le programme d'accueil.

En cas de correction ou autre changement, on agira directement sur le fichier-texte d'origine et le programme de transfert écrasera la première version pour la remplacer par la nouvelle.

Ajoutons enfin que le fichier-texte peut aussi être utilisé avantageusement tel quel si l'enregistrement ne contient qu'un seul champ (ex : fichier d'adverbes, de prépositions, d'adjectifs, etc.) et si l'on a pas besoin d'accès direct. On peut même, le cas échéant, simuler des champs en séparant, dans chaque enregistrement-texte, les items par un espace ou par tout autre caractère démarcatif. Si un enregistrement-texte (Etx) est par exemple constitué de trois formes verbales (Inf, Pret, Part2) séparées par des espaces :

laufen lief gelaufen

on accédera à chacune d'entre elles par les formules suivantes (bien entendu après les déclarations de variables et initialisations qui s'imposent) :

FOR i := 1 **TO** 3 **do** {exécuter 3 fois de suite}

begin

Tableau[i] := **COPY**(Etx,1,**POS**(' ',ETX)-1); {mettre dans un tableau depuis le début de l'enregistrement jusqu'au premier espace}

DELETE(Etx,**LENGTH**(Tableau[i])+1) {détruire la partie de l'enregistrement qui a été stockée dans le tableau et recommencer l'opération}

end.

ce qui permettra d'avoir Inf dans Tableau[1], Pret dans Tableau[2] et Part2 dans Tableau[3].

Nous dirons en conclusion que le fichier-texte peut être considéré comme le pivot de toute utilisation de fichier en programmation avec Turbo-Pascal, grâce à sa souplesse d'emploi et à la possibilité de transfert en fichiers d'autres types. Il constitue une base de données toujours accessible, dont on peut à tout moment extraire une sélection et la transférer en fichier à accès direct pour les besoins de tel ou tel programme.

R. LEGALERI
REIMS