Développer dans CASTEM2000 : LES OPERATEURS

T. CHARRAS

0.1 Introduction

La structure informatique de Castem2000 entraine qu'il n'est pas nécessaire de connaitre les autres opérateurs pour pouvoir en introduire un nouveau, ceux-ci sont en effet orthogonaux les uns aux autres. Par contre la communication avec le monde externe, via l'analyseur de syntaxe est absoluement primordiale de meme que la connaissance des structures d'informations (les objets) concernées par l'opérateur.
Rappelons que la syntaxe de la commande élémentaire dans Castem2000 est :

RESULTAT = OPERATEUR OPERANDES ;

Dans ce manuel nous regarderons le branchement de l'opérateur dans le gestionnaire de commandes puis nous préciserons quelques interfaces avec le monde externe ainsi que certaines règles de fonctionnement de l'analyseur de syntaxe.

0.2 Branchement d'un opérateur

Le sous-programme PILOT est le seul qui doit etre modifié. C'est lui qui connait la liste des opérateurs et qui leur donne le controle si l'analyseur de syntaxe a détecté le nom de l'opérateur dans la commande de l'utilisateur. Schématiquement un tableau de chaines de caractères de 4 lettres, initialisé par data, contient la liste des opérateurs. L'appel au sous-programme LIRMO2 retourne la position, dans le tableau de chaines de l'opérateur invoqué par la commande de l'utilisateur.Il suffit alors d'utiliser un GOTO indicé pour se brancher sur le sous-programme point d'entrée de l'opérateur.

0.2.1 Réalisation pratique

La liste du sous-programme PILOT est longue mais on peut en extraire (aujourd'hui 5 mai 1995) ces quelques lignes.

```
SUBROUTINE PILOT
  .
PARAMETER(NDIR3=49)
PARAMETER(NDIR2=178)
PARAMETER(NDIR1=179,NDIR1P=....,NDIR=NDIR1+NDIR2+NDIR3)
CHARACTER*4 MDIR(NDIR),MDIR1(NDIR1),...,MDIR3(NDIR3)
EQUIVALLENCE (MDIR(1),MDIR1(1)),(MDIR(NDIR1P)=MDIR2(1)),..
DATA MDIR1/'OPTI','FIN ','DUMP',...
   .
   .
```

```
      DATA MDIR2/'COPI','DIMN','SAUV',....
       .
      DATA MDIR3/'NUAG','WEIP','KHIS','.....
     *'KPRO','FFOR','RAYE','RAYN','VSUR','TRAJ'/
C l'ecriture compliquée avec plusieurs tableaux en equivallence
C n'est justifiée que par des raisons strictement informatiques
       .
       .
      CALL LIRMO2(MDIR,NDIR,0,I,ICOHCO)
      IF (I.NE.0) GO TO 100
       .
       .
100   LOCERR=MDIR(I)
       .
      IF(I.LE.100)
       .
      IF(I.LE.200)
       .
      IF(I.LE.300)
       .
      IF(I.LE.400)
       .
      IF (I.LE.500)
     *GOTO (501,502,503,504,505,506),I-400
       .
       .
506   CALL TRAJEC
      GO TO 1
      END
```

Les seules lignes à modifier pour introduire l'opérateur de nom TITI et de sous-programme principal TITISP sont :

```
C incrémentation de 1 de NDIR3
      PARAMETER(NDIR3=50)
       .
C ajout du nom TITI dans la liste des opérateurs
     *'KPRO','FFOR','RAYE','RAYN','VSUR','TRAJ','TITI'/
       .
C ajout d'un indice dans la liste du goto indicé
     *GOTO (501,502,503,504,505,506,507),I-400
       .
```

Enfin il faut ajouter juste avant l'instruction END les deux lignes :

```
507    CALL TITISP
       GO TO 1
       END
```

La seule précaution à prendre est de vérifier que le nom TITI n'est pas déjà le nom d'un opérateur.

Le gestionnaire de commandes fera en sorte que le sous-programme TITISP soit appelé si la commande invoque l'opérateur TITI. Il faut maintenant écrire la chaine de sous-programmes appelée par TITISP.

0.3 Interface opérateur-monde externe

L'opérateur a besoin de savoir sur quel objet il doit travailler et si la commande a des options particulières. Il lui faut donc connaitre les opérandes fournis par l'utilisateur dans sa commande. Ensuite, une fois le travail effectué l'opérateur devra rendre son ou ses résultats au monde externe.

0.3.1 Acquisition d'un objet

La première action d'un opérateur est souvent d'acquerir les structures de données (OBJET) à partir desquelles il travaillera. Pour cela les sous-programmes LIR*** ont été crés. En étant exhaustif, on peut citer :

1. SUBROUTINE LIRENT (IVAL,ICOND,IRETOU) Lectur d'un entier
2. SUBROUTINE LIRREE (XVAL,ICOND,IRETOU) Lecture d'un flottant
3. SUBROUTINE LIRCHA (CHAR,ICOND,IRETOU) Lecture d'une chaine
4. SUBROUTINE LIRLOG (LOGI,ICOND,IRETOU) Lecture d'un logique
5. SUBROUTINE LIROBJ (MTYP,IRET,ICOND,IRETOU) Lecture d'un objet
6. SUBROUTINE LIRTAB (MTYP,IRET,ICOND,IRETOU) Lecture d'une table sous-typée

- * ICOND est un entier de valeur 0 ou 1 précisant si la présence de l'objet est obligatoire ou non. Un message d'erreur est émis par le lecteur en cas de besoin, de plus la variable IERR du COMMON COPTIO prend une valeur différente de zéro.
- * IRETOU est un entier de valeur 1 ou 0 précisant en retour de la lecture si la demande a pu etre réalisée avec succés ou non. Dans le cas de la lecture d'une chaine IRETOU contient la longueur de la chaine lue.

* IVAL est un entier qui contient en retour la valeur lue.
* XVAL est un REAL*8 qui contient en retour la valeur lue.
* CHAR est une chaine de caractères, elle contient en retour, cadrée à gauche et tronquée ou complétée par des blancs, la chaine lue.
* LOGI est un logique (variable fortran LOGICAL)
* MTYP est une chaine de 8 caractères précisant le type de l'objet attendu (ou le sous-type de l'objet TABLE si appel à LIRTAB).
* IRET est la valeur associée à l'objet lu. Dans le cas d'objet crée par les opérateurs de Castem2000, c'est la valeur transmise lors de l'enregistrement de l'objet par les sous-programmes ECR*** qui suivent.

0.3.2 Ecriture d'un objet

Le nom des sous-programmes évoque l'écriture parce qu'il s'agit en fait d'écrire dans la pile des objets qui sont en attente de lecture. Les sous-programmes ECR***, qui servent aussi à rendre un résultat, sont :

1. SUBROUTINE ECRENT(IVAL) Pour rendre un entier
2. SUBROUTINE ECRREE(XVAL) Pour rendre un flottant
3. SUBROUTINE ECRCHA(CHAR) Pour rendre une chaine de caractères
4. SUBROUTINE ECRLOG(LOGI) Pour rendre un logique
5. SUBROUTINE ECROBJ(MTYP,IRET) Pour rendre un objet de type MTYP et de valeur associée IRET.

Les variables ont la meme signification que précédemment.

Pour lire ou écrire dans une table il faut utiliser les sous-programme ACCTAB ou ECCTAB qui ne font pas, à proprement parler, partie de l'analyseur de syntaxe. Ces deux sous-programmes sont détaillés dans la partie concernant les OBJETS.

0.3.3 Reconnaissance du type de la donnée

Certains opérateurs généraux peuvent traiter des objets de types variés. Un sous-programme permet de connaitre à l'avance le type de la prochaine donnée à lire.

– SUBROUTINE QUETYP (MTYP,ICOND,IRETOU)

En retour MTYP contient sur 8 caractères le type du premier objet en attente de lecture. Icond et IRETOU ont la meme signification que précédemment.

0.3.4 Lecture d'une chaine parmi une liste

Dans un opérateur, la lecture de mots-clés optionnels peut facilement se faire à l'aide de LIRMOT.

- SUBROUTINE LIRMOT (LISMO,NBMOT,IRAN,ICOND)
  - LISMO est un tableau de NBMOT chaines de caractères.
  - IRAN est le rang de la chaine lue dans le tableau LISMO.
  - ICOND demande ou non un message d'erreur si la demande n'a pas pu etre satisfaite.

0.4 Remarques

- Pour en savoir plus nous recommandons le rapport : "Langage GIBIANE : definition" DEMT/89-108 (il est repris dans le guide du developpement)
- Si un opérateur rend plusieurs résultats, il faut les écrire en commençant par le dernier (pile LIFO).
- Le lecteur explore les données tant qu'il n'a pas rencontré un objet du type désiré ou tant qu'il ne rencontre pas une chaine de caractère. Ceci permet la lecture non positionnelle des objets de type différents.
- L'accés aux informations contenues dans les objets de type TABLE est faite par le sous-programme ACCTAB et la mise d'informations dans un objet TABLE se fait à l'aide de ECCTAB.

0.5 Exemple complet et message d'erreur

Le common "COPTIO" que l'on peut mettre dans un sous-programme par l'instruction :
-INC CCOPTIO
contient certaines informations générales. En particulier la dimension de l'espace, les unités d'entrées-sorties et la communication avec le gestionnaire d'erreurs.
Ainsi la variable IERR, contenue dans le common, représente le niveau d'erreur atteint pendant l'exécution de la commande élémentaire. Les tableaux INTERR, REAERR et la chaine MOTERR permettent de communiquer avec les messages d'erreurs contenus dans le fichier gibi.erreur.
Nous nous proposons de faire un opérateur qui lit une syntaxe :
FLOT1 MOT1 FLOT2
Dans cet exemple MOT1 sera soit le mot SUPERIEUR soit le mot INFERIEUR.
L'opérateur rendra un logique vrai si la phrase est vrai.

SUBROUTINE EXEMPL

```
      IMPLICIT REAL*8(A-H,O-Z)
-INC CCOPTIO
      CHARACTER*9 MOT(2)
      DATA MOT/'SUPERIEUR','INFERIEUR'/
      CALL LIRMOT(MOT,2,0,ICAS)
      IF(ICAS.EQ.0) THEN
        MOTERR(1 :9) = MOT(1)
        MOTERR(10 :18) = MOT(2)
        CALL ERREUR ( 777)
        RETURN
      ENDIF
      CALL LIRREE(XV1,1,IRETOU)
      IF(IERR.NE.0) RETURN
      CALL LIRREE(XV2,1,IRETOU)
      IF(IERR.NE.0) RETURN
      IF( ICAS.EQ.1) THEN
        IF ( XV1.GT.XV2) THEN
          CALL ECRLOG(.TRUE.)
        ELSE
          CALL ECRLOG(.FALSE.)
        ENDIF
      ELSE
        IF ( XV1.GT.XV2) THEN
          CALL ECRLOG(.FALSE.)
        ELSE
          CALL ECRLOG(.TRUE.)
        ENDIF
      ENDIF
      RETURN
      END
```

Le message d'erreur 777 est supposé faire une impression qui ressemble à :
On attend un des mots : SUPERIEUR ou INFERIEUR
Le tableau INTERR permet de passer des valeurs entières et REAERR des valeurs réelles. Dans le fichier gibi.erreur, un message d'erreur est composé de deux ou quatre lignes. La première et la troisième contiennnent le numèro de l'erreur et son niveau (le niveau sert à initialiser la variable IERR et à partir de 2 le lecteur interne passe à l'instruction suivante en sortant du bloc REPETER ou de la procédure). La deuxième et la quatrième ligne servent à définir le message qui va etre imprimé. Un numéro d'erreur négatif est utilisé pour les messages normaux, par exemple pour les textes des listes d'objets.Ci-joint l'en-tete du fichier et un ou deux extraits.

```
9999 0 C GIBI      ERREUR    CHAT      96/03/18    21:15:02     2086
Liste des messages à traduire pour castem2000
9999 0
une ligne introduite par 9999 est commentaire: par 9998 est la
9999 0
liste des langues: par 9997 est la def de la langue  : par
9999 0
 9996 la fin du fichier 9995 introduit la langue utilisée par défaut
9999 0
Format du fichier : lrecl=80
9999 0
 première ligne numéro message I4,1x,niveau erreur I1
9999 0
deuxième ligne texte A80
9999 0
prendre un numéro inférieur ou supérieur ( consécutif)
9999 0
%i1 représente interr(1)  common coptio
9999 0
%r3 représente reaerr(3)  common coptio
9999 0
%m3:8 représente moterr(3:8)  common coptio
9998 0 ceci est la liste des langues utilisées(FRAN en premier)
FRAN ANGL
9995 0  langue par défaut à changer si besoin est
FRAN
9997 0 ceci est la langue qui va etre utilisée
FRAN
-307 0
Matrice élémentaire non-symétrique
-306 0
*****     FIN DE LA PROCEDURE INITIALE    ******
-305 0
*****  EXECUTION D'UNE PROCEDURE INITIALE ******
-304 0
ATTENTION : On ne prend pas en compte la moyenne pour un bruit blanc
-304 0
à distribution exponentielle (valeur lue : %r1 )
-303 0
Impossible de regénérer l'élément %i5 ayant deux fois le meme noeud.
-302 0
```

```
Liste des procedures surchargées par l'utilisateur :
-302 0
****************************************************


 188 2
On cherche un chpoint qui contient des contributions modales
 189 2
Pas de fréquence propre dans l'intervalle fourni
 190 2
On veux lire un entier supérieur ou égal à %i1 au lieu de %i2
 191 2
On veux lire un flottant supérieur ou égal à %r1 au lieu de %r2
 192 2
Impossible d'orienter les forces de pression: direction donnée
 192 2
perpendiculaire à la force de pression au point indiqué
 193 2
Impossible d'utiliser cet opérateur pour la formulation %m1:8
 194 2
Impossible de calculer les contraintes pour la formulation %m1:8
 195 2
Changement de signe du jacobien dans l'élément %i1. Maillage incorrect
 196 2
Le type d'un des opérandes n'est pas compatible avec l'opérateur %m1:8
 197 2
Le mot %m1:4 n'est pas un nom de composante reconnu
```