

V L I S P - A I D



Harald WERTZ
Juillet 1977
R.T.-10-77

AID

«AID» est un ensemble de fonctions LISP destiné à aider le programmeur lors de la conception et la mise au point de ses programmes.

«AID» aide à retrouver l'endroit où a été utilisée telle ou telle variable ou fonction ; montre l'interaction des différentes parties du programme, ... ; et constitue avec DEBUG, EF, ADVISE et BREAK un excellent outil pour la mise au point.

Il existe deux manières de l'utiliser :

- comme programma d'indexation ;
- interactivement, en demandant des renseignements précis.

Exemple : soit une fonction (entre autres)

```
(DE FOO (X Y)
  (IF (NULL Z)
    (CONS (FOO (CAR X))(FOO (CDR Y))) ))
```

à l'exécution la machine donne l'indication suivante

```
Z
ER A8
```

Si vous ne savez plus (ou pas) où vous avez utilisé la variable Z, demandez (DESCR Z), la machine répond

```
xxxxx Z xxxxx
```

```
Z IS FREE IN FOO
```

Maintenant il ne reste plus qu'à chercher l'éditeur et debugger le programme .

COMMANDES DE AID

(AID) pour charger le système

(%NOTALL) toutes les fonctions définies à partir de l'instant de l'appel de %NOTALL ne sont pas connues. Pour revenir, il faut taper :

(%ALL) toutes les FEXPR ou EXPR définies à partir de maintenant sont prises en compte par AID (état standard)

(%I) donne l'arbre des appels pour toutes les fonctions que AID connaît et une liste de toutes les variables globales, où elles sont évaluées et où elles sont modifiées. Cette commande fournit la liste de toutes les variables locales, avec indication des fonctions dans lesquelles elles sont déclarées.

(DESCR) pour DESCRibe, livre la description de toute fonction et de toute variable connues par AID

(INDEX) livre les arbres d'appel de toutes les fonctions connues par le système

Si on donne des arguments (en nombre quelconque) à %I, DESCR ou INDEX, la machine ne donne les renseignements que pour ces arguments.

(IF $ftn_1 \dots ftn_n$) obligatoire pour que la machine connaisse de ftn_1 à ftn_n , si elles ont été définies soit avant d'utiliser AID, soit en mode %NOTALL

(VARDE) donne la description de toutes les variables connues

- (%UNDO) sans argument :
le système perd toute connaissance sur les fonctions et les variables
avec arguments :
le système perd toute connaissance sur les variables et/ou les fonctions données en argument
- (%FIN-AID) fait un (%UNDO) et efface toutes les fonctions de AID. Cette fonction aide à quitter DEFINITIVEMENT «AID», et libère tous les doublets utilisés par le système.

ATTENTION

Sous AID, vous pouvez travailler tout comme en LISP (les définitions de fonctions (DE, DF) se révèlent plus longues à l'exécution : le système redéfinit les deux fonctions DE et DF)

AID modifie les P-listes de tous les atomes qu'il connaît (il se produit donc des modifications sur les P-listes).

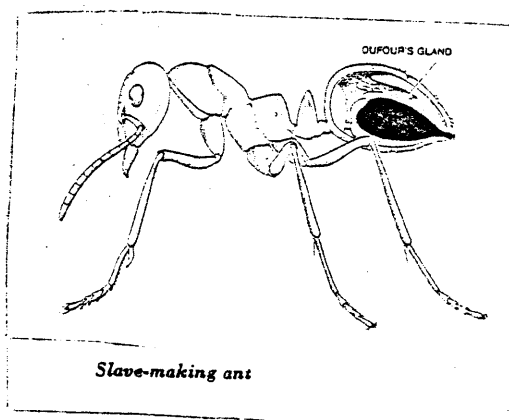
%UNDO efface toute la P-liste des atomes qui sont des variables.

Sauf pour MAP, MAPC, MAPCAR et APPLY, le système ne reconnaît pas les arguments fonctionnels et ne prend pas en compte l'argument d'un QUOTE

Les labels des ESCAPE sont, à la définition, considérés comme des variables et, à l'exit, comme des fonctions.

Les fonctions ou variables aux noms des fonctions standard ne sont pas prises en compte.

Eviter d'utiliser des variables ou fonctions dont le P-NAME commence par le caractère «%». Toutes les variables et fonctions critiques de AID ont un P-NAME commençant par ce caractère.



Soit un exemple «bidon»

```

?(DE F00BAR (X Y)
? (SETQ A (CDR A))
? (COND
? ((F00 X A)(BAR Y (NC0NC B C)))
? ( T (PR0G (A B)(F00BAR (C0NS A (LIST B X Y)) B))))
F00BAR
?(DE F00 (A B)
? (IF (CDR A)(F00 (F00BAR (NC0NC1 X A)(F00 Y (C0PY B))) B)
? B))
F00
?(DE BAR (A B)
? (IF A B (BAR (F00BAR A B)(F00 (SETQ Y B) A))))
BAR
?(DE C0PY (X)
? (IF X (C0NS (CAR X)(C0PY (CDR X))))
C0PY
?(%I)

```

On a défini les 4 fonctions sous AID, regardons d'abord les arbres d'appel

```

C0PY      C0PY
BAR        BAR
           F00      C0PY
                   F00
                   F00BAR  BAR
                               F00
                               F00BAR
F00BAR     F00BAR
F00        C0PY
           F00
           F00BAR
F00BAR     BAR
           F00
           F00BAR

```

Noter que AID ne reconstruit pas les arbres déjà présentés

puis les variabls

```

VARIABLE GL0BALES :
A      IN F00BAR
B      IN F00BAR
C      IN F00BAR
X      IN F00
Y      IN BAR F00

```

```

VARIABLE GL0BALES M0DI FIEE :
A      IN F00BAR
B      IN F00BAR
C      IN
X      IN F00
Y      IN BAR

```

C n'est nulle part modifiée

```

VARIABLE L0CALES :
A      IN BAR F00 F00BAR
B      IN BAR F00 F00BAR
X      IN C0PY F00BAR
Y      IN F00BAR
NIL

```

?(DESCR)

***** A *****

D'abord les variables

A IS FREE IN F00BAR
 A IS MODIFIED IN F00BAR
 A IS BOUND IN BAR F00 F00BAR

***** B *****

B IS FREE IN F00BAR
 B IS MODIFIED IN F00BAR
 B IS BOUND IN BAR F00 F00BAR

***** C *****

C IS FREE IN F00BAR

***** X *****

X IS FREE IN F00
 X IS MODIFIED IN F00
 X IS BOUND IN COPY F00BAR

***** Y *****

Y IS FREE IN BAR F00
 Y IS MODIFIED IN BAR
 Y IS BOUND IN F00BAR

puis les fonctions

***** BAR *****

TYPE = EXPR
 ARGS = (A B)
 VARGLOB = Y
 VARGLOB MODIF = Y
 USING = BAR F00 F00BAR
 USED BY = BAR F00BAR

***** COPY *****

TYPE = EXPR
 ARGS = (X)
 USING = COPY
 USED BY = COPY F00

***** F00 *****

TYPE = EXPR
 ARGS = (A B)
 VARGLOB = X Y
 VARGLOB MODIF = X
 USING = COPY F00 F00BAR
 USED BY = BAR F00 F00BAR

***** F00BAR *****

TYPE = EXPR
 ARGS = (X Y)
 VARGLOB = A B C
 VARGLOB MODIF = A B
 USING = BAR F00 F00BAR

Demandons l'arbre d'appel de F00

(INDEX F00)

```
F00      F00BAR      F00BAR
          F00
          BAR          F00BAR
                   F00
                   BAR
          F00
          C0PY      C0PY
```

NIL

? (ZNOTALL)

DF

? (DE PREP (X L))

? (COND

? ((NULL L) NIL)

? ((EQ (CAR L) X) (PREP X (CDR L)))

? ((CONS (CAR L) (PREP X (CDR L))))))

PREP

? (DESCR PREP)

Sortons provisoirement de AID et
définissons une fonction

***** PREP *****

Pas de description, puisque inconnue à AID

NIL

? (II PREP)

PREP PREP

l'arbre d'appel pour PREP

NIL

? (DESCR PREP)

***** PREP *****

maintenant il est connu

TYPE = EXPR

ARGS = (X L)

USING = PREP

USED BY = PREP

NIL

? (DESCR X)

***** X *****

X IS FREE IN F00

X IS MODIFIED IN F00

X IS BOUND IN PREP C0PY F00BAR

NIL

? (PREP 'X '(X Y X Z))

(Y Z)

?

OK, bonne chance