THESE

présentée

à l'Université des Sciences et de la technologie
HOUARI BOUMEDIENE

POUR OBTENIR LE GRADE DE
MAGISTER
EN
INFORMATIQUE

PAR

Mlle Mounira ZOUBEIDI

DÉFINITION DÉNOTATIONNELLE DE A . R . L . : LANGAGE APPLICATIF AVEC ORGANISATION RELATIONNELLE DES DONNÉES ET ÉTUDE DES PROBLÈMES LIÉS AUX RELATIONS

SOUTENUE LE 25 JUIN 1984 DEVANT LA COMMISSION D'EXAMEN

Messieurs :

C.B. BENYELLES	Professeur à l'USTHB	Président
A. AïNOUCHE	Maître de conférences à l'USTHB	Examinateurs
P.H. JORRAND	Directeur de recherche au CNRS	
S. A. LARIBI	Professeur à L'USTHB	

THESE

présentée

à l'Université des Sciences et de la technologie
HOUARI BOUMEDIENE

POUR OBTENIR LE GRADE DE
MAGISTER
EN
INFORMATIQUE

PAR

Mlle Mounira ZOUBEIDI

DÉFINITION DÉNOTATIONNELLE DE A . R . L . : LANGAGE APPLICATIF AVEC ORGANISATION RELATIONNELLE DES DONNÉES ET ÉTUDE DES PROBLÈMES LIÉS AUX RELATIONS

SOUTENUE LE 25 JUIN 1984 DEVANT LA COMMISSION D'EXAMEN

Messieurs :

C.B. BENYELLES	Professeur à l'USTHB	Président
A. AïNOUCHE	Maître de conférences à l'USTHB	Examinateurs
P.H. JORRAND	Directeur de recherche au CNRS	
S. A. LARIBI	Professeur à L'USTHB	

R E M E R C I E M E N T S

Je tiens à remercier Monsieur C.B. BENYELLES, Professeur à l'U.S.T.H.B. et Directeur de l'Institut de Mathématiques, pour l'intérêt qu'il a porté à mon travail et l'honneur qu'il me fait en acceptant la présidence de ce jury.

Que Monsieur Ph. JORRAND, Directeur de Recherches au C.N.R.S., soit assuré de ma profonde gratitude pour la formation qu'il m'a donnée en me dirigeant dans ce travail. Ses encouragements et ses critiques ont été pour moi autant de soutiens.

Que Monsieur S.A. LARIBI, Professeur à l'U.S.T.H.B., veuille bien accepter l'expression de ma profonde reconnaissance pour la formation qu'il m'a donnée et pour m'avoir constamment guidé et conseillé dans mon travail.

Je voudrais remercier Monsieur AINOUCHE, Maître de Conférences à l'U.S.T.H.B. pour l'intérêt qu'il témoigne à ce travail en acceptant de parcitiper à ce jury.

Mes remerciements s'adressent aussi à tous ceux qui ont participé à la réalisation matérielle de ce document, en particulier Madame MEKEBEL.

P L A N

Pages

INTRODUCTION

L'évolution parallèle des langages de programmation et de base de données a conduit à affranchir ces derniers du caractère procédural et des notions de variables et d'instructions (affectation, branchement, ...) qui les ont caractérisés jusqu'à lors.

Pour les langages de programmation, ces orientations ont donné naissance au développement des langages applicatifs [BAK 78], [TUR 81], où les notions de base sont perçues comme des fonctions et formes fonctionnelles.

Pour les langages de base de données, cette évolution est le résultat de l'introduction par E.F. CODD [COD 70] du modèle relationnel fondé sur la notion de relation, issue directement de la notion mathématique de relation.

Il est essentiel de prendre en compte l'évolution de ces deux types de langage pour solutionner certains problèmes, et dans ce travail nous nous sommes attachées à résoudre le problème posé par l'utilisation séparée des langages de programmation et de base de données. En effet, la plupart des langages actuels de programmation sont conçus pour traiter des données indépendantes les unes des autres, alors que les langages de base de données traitent des informations organisées en une structure globale et sont essentiellement conçus autour de cette structure.

Cependant, il est souvent nécessaire de faire coopérer ces deux types d'activité : après avoir extrait des informations d'une base de données, on peut vouloir exécuter un programme avec ces informations comme paramètres ou en sens inverse, les résultats d'un calcul complexe exprimé dans un langage de programmation doivent être rangés dans une base de données.

La solution généralement adoptée, est l'utilisation d'un langage de programmation (FORTRAN, PL1, ...) comme langage hôte. Cette solution, bien que techniquement satisfaisante, présente un certain nombre d'inconvénients dûs essentiellement à la lourdeur de sa mise en oeuvre.

L'aboutissement à un mode d'expression homogène et cohérent pour les langages de programmation et les langages de base de données relationnelles est une voie qui nous a semblé appropriée pour la conception d'un langage applicatif couvrant les activités de gestion de base de données et de programmation.

Dans ce travail, nous présentons A.R.L. langage applicatif avec une organisation relationnelle des données.

Le premier chapitre de ce document décrit de manière informelle les objets et les fonctions de ce langage, ainsi que ses limitations.

Le deuxième chapitre rappelle certaines méthodes de description de langage et décrit plus particulièrement les fondements et le formalisme de la méthode dénotationnelle que nous avons adoptée pour décrire A.R.L.

Le troisième chapitre regroupe la description de la sémantique statique et dynamique du langage ainsi qu'une proposition d'une syntaxe concrète.

Le quatrième et dernier chapitre décrit en détail l'aspect logique et physique des relations, en mettant l'accent sur l'apport de A.R.L. dans la coopération entre les langages de programmation et de base de données, ainsi que l'apport de la définition formelle dans la conception d'un langage.

C O N C L U S I O N

Nous venons de présenter A.R.L. : langage destiné à une large communauté d'utilisateurs du fait qu'il assure les fonctions de programmation et de gestion de base de données relationnelles.

Après avoir défini formellement A.R.L. par la méthode dénotationnelle, nous avons proposé une structure de données, exprimée en PASCAL, pour l'ensemble de l'interpréteur.

L'implémentation des actions de définition de nom et de modification de relation nous ont permis d'apprécier particulièrement les avantages de la méthode dénotationnelle, qui tout en étant un outil agréable dans la phase de définition de langage, facilite le passage à l'implémentation.

Ne répondant pas encore à toutes les exigences d'un système de gestion de bases de données relationnelles, il serait souhaitable de développer A.R.L. en intégrant dans le langage les contraintes de confidentialité pour protéger la base de données contre certaines utilisations et en enrichissant la sémantique par :

- la fonction de gestion des accès concurrents pour permettre une multi-utilisation de la base,
- la fonction d'optimisation des requêtes,
- la fonction assurant la sécurité de fonctionnement pour une bonne reprise en cas de panne.

A.R.L. étant non séquentiel, l'implémentation d'un interpréteur "parallèle" est toute indiquée et une perspective intéressante à ce travail serait la réalisation automatique d'interpréteur "parallèle" sur base d'une définition dénotationnelle d'un langage applicatif quelconque.

ANNEXE 1 : STRUCTURE DE DONNEES

Ayant choisi PASCAL comme langage de programmation, les structures de données sont donc celles permises dans ce dernier.

Deux types de règles apparaissent dans la syntaxe abstraite :

1 - PARTGAUCH 1 :: PART1 PART 2 .... PARTK

2 - PA GAUCH 2 = P1 | P2 .... | PK

La première règle est représentée sous forme de record PASCAL où les différents éléments de la partie droite représentent les champs du record.

La deuxième règle est représentée sous forme de type scalaire énuméré.

*** Structure de donnée relative à la syntaxe abstraite***

```
TYPE TOPACC = (TU, RE);
     TCD2   = (I, U, D);
     TCD1   = (CS, CB, CI, CD);
     TOPU   = (+ U, - U, NOT, LONG);
     TOPSET = (DIF, INC, IN, U, V);
     TOPNE  = (NOT = S, NOT = T, NOT = E, NOT = R);
     TOPE   = (= S, = T, = E, =R);
     TROPB  = (<, <=, >, >=);
     TBOPB  = (AND, OR, ⇒ );
     TNOPB  = ( +, -, *, DIV, CONC, **);
     TOPB   = (VNOPB, VBOPB, VROPB, VOPE, VOPNE, VOPSET);
     TOPC   = (AJOUT CONT, UPDATE CONT);
     TOPER  = (+ C, * C, ∧C, V C, MAX, MIN, CONT);
     TCOP   = (R, S);
     TUOP   = (UPDATE, UPDATE ALL);
     TOP    = (INSERT, DELETE, DELETE ALL);
     TCOND  = (COND1, COND2);
     TENS   = (SETEXT, SETINT);
     CSPEC  = (.,) ; : , / , ...);
     LETTRE = 'A' ....'Z'
     CHIFFRE = '0' ... '9'
     TCAR   = (VCHIFFRE, VLETTRE, VCSPEC);
     TCBOOL = (T, F);
     TC     = (VCENT, VCREEL, VCBOOL, VCTUPLE, VACCES, VCHAINE, VENS,
               VREL);
     TAPP   = (VREF, VLAMBDA);
     TEXPIMP = (VEXPR, VEIMP);
     TEXPC  = (MODIFC, DELETE CONT, GROSMOD, MAIGRMOD, SUPMOD);
     TEXP   = (EB, EU, APP, REF, CONSTR, COND, EXPREAD, EQUIJOIN, PROJ,
               SELECTR, SELECTS, OPEXIST, OPUNIV, CARTR, CARTE, JOIN,
               DIVR, ANTIPROJ, SOMR, MODIFR, UPDATER, COMB, EXTRCHAIN,
               RENOM, NEW, OLD, APPLY, WS);
     TEXPINT = (VEXP, VFCT);
     TYP1   = (ENT, BOOL, REEL, CHAINE);
     TYP2   = (VTELEM, VTENS, VTTUPLE);
     TYP3   = (VTBASE, VTREEL);
     TYP4   = (VTVAL, VTFCT);
     TYP5   = (VTYPE, VTYPR);
     TEXP1  = (VEXPR, VCOM);
```

```
PEXP    = ↑EXP
LEXP    =   ARRAY [1 ... MAXEXP]  OF PEXP;
T2 EXP  =   RECORD   EXP2 1 : P EXP;
                     EXP2 2 : P EXP
            END;
NOM     =   ARRAY  [1... 8]  OF CHAR;
LNOM    =   ARRAY  [1... MAXCHAMP]   OF NOM;
PPROD   =    ↑ LPROD;
LPROD   =   RECORD   ID      : NOM;
                     EXP     : P EXP;
                     PRODSUIV : PPROD
            END;

TLCOMP  = ↑ LCOMP;
LCOMP   =   RECORD   COMPL     : LEXP;
                     COMPSUIV  : TLCOMP
            END;

CTUPLE  =   ARRAY  [1 ... MAXCHAMP]   OF
               RECORD  CHAMP     : NOM ;
                       EXPCHAMP  : PEXP
            END;
PCHAINE = ↑CHAINE
CHAINE  =   RECORD  CARI      : TCAR;
                    CARSUIV   : PCHAIN
            END;
CENT    =   ARRAY [1 ... MAXCE]  OF CHIFFRE;
CREEL   =   RECORD  PARENT : ARRAY [1 ...MAXPE]  OF CHIFFRE;
                    DECPAR : ARRAY [1 ...MAXPD]  OF CHIFFRE
            END;
ENS     =   RECORD CASE  VR  :  TENS OF
                    SETEXT    : (CSETEXT : LEXP);
                    SETINT    : (CSETINT : T2EXP)
            END;
```

```
TCONSTR    =  RECORD CASE CV : TC OF
      VCENT    : (CVENT     : CENT);
      VCREEL   : (CVREEL    : CREEL);
      VCBOOL   : (CVBOOL    : TCBOOL);
      VCTUPLE  : (CVTUPLE   : CTUPLE);
      VACCES   : (CVACCES   : RECORD OPACC : TOPACC;
                                     EXPTUP : PEXP;
                                     CHAMPACCE : NOM
                              END;) ;
      VCHAINE  : (CVCHAIN   : PCHAINE);
      VREL     : (CVREL     : RECORD  LID    : LNOM;
                                      LCOMP  : LLCOMP;
                                      MDOM   : CTUPLE;
                              END;);
         END;

PTENS      = ↑TENS;
PTTUPLE    = ↑TTUPLE;
TBASE      =   RECORD  CASE VBASE : TYP2 OF
                       VTELEM    :  (CE       : TYP1);
                       VTENS     :  (CENS     : PTENS);
                       VTTUPLE   :  (NOTTUPLE: (TUPLE);
                                     CUPLE    : PTTUPLE)
                              END
         END;
TENS       = RECORD  NOTENS   : (ENS);
                     BASTYPE  : TBASE
         END;
TTUPLE     =  ARRAY [1 ... MAXCHAMP]   OF  RECORD
                                            CHAMPT    : NOM;
                                            TYPCHAMP  : TBASE
                                    END;
TREL       =  RECORD NOTREL    : (REL);
                     TYPCOMP   : PTTUPLE
         END;
```

```
TVAL      =   RECORD  CASE VTVAL  : TYP3  OF
                      VTBASE  : (CBASE : TBASE);
                      VTREL   : (CREL  : TREL)

                      END
              END;
PTYPARAM  =  ↑TYPPARAM;
TFCT      =   RECORD  CODFCT   : (FCT);
                      PARAMFCT : PTYPARAM;
                      TYPRES   : TVAL

              END;

TYP       =   RECORD CASE VTYP : TYP4  OF
                      VTVAL   : (CTVAL  :  TVAL);
                      VTFCT   : (CVTFCT : TFCT)

              END;

TYPARAM   =    RECORD  TYPPREM : TVAL;
                       TYPSUIV : PTYPARAM

              END;

EIMP      =    RECORD  PART1 : ENS;
                       PART2 : PEXP

              END;

EXPIMP    =    RECORD CASE VEIMP : TEXPIMP   OF
                           VEXPR  : (CEXPR  : PEXP);
                           VEIMP  : (CIMP   : EIMP)
                       END
              END;

PEXPL     =  ↑EXPL;
PLNOM     =  ↑NOML
NOML      =    RECORD NOM1     : NOM ;
                      NOMSUIV : PLNOM
              END;
```

```
EXP     =   RECORD  NOM1     : NOM;
                    NOMSUIV  : PLNOM
                END;

TMOD    =   RECORD  LREF     : PLNOM;
                    LUSE     : PLNOM
                END;

EXCP    =   RECORD CASE VEXPC  : TEXPC  OF
                   MODIFC      : (OPEXPC  : TOPC;
                                  CD1     : TCD1;
                                  CD2     : TCD2;
                                  IDC1    : NOM;
                                  IDC2    : NOM;
                                  EXPCOM  : PEXP );
                   DELETE CONT : (DC1     : TCD1;
                                  DC2     : TCD2;
                                  IDD1    : NOM;
                                  IDD2    : NOM
                   GROSMOD     : (NONMOD  : NOM;
                                  VMOD    : TMOD);
                   MAIGMOD     : (MOD     : NOM; NOMOBJET : NOM);
                   SUPMOD      : (MODSUP  : NOM);
                     END
                END;

MDEF    =   RECORD  MDNOM : NOM;
                    MODVAL : TMOD
                END;

MEXP    =   ARRAY [1 ... MAXCONT] OF RECORD
                                       NONC    : NOM ;
                                       EXPCONT : PEXP
                                   END;

MEXPIMP =   ARRAY [1 ... MAXCONT] OF RECORD
                                       NOMCI   : NOM;
                                       EXPCONT1: EXPIMP
                                   END;
```

```
BCONT     = RECORD KEY : ARRAY [1 ... MAXCHAMP  OF NOM
                   CONT STAT :  MEXP;
                   CONT BOOL :  RECORD SCD   : M EXP;
                                       SCI   : M EXP;
                                       SCU   : M EXPIMP;
                                END;
                   CONT IMP  :  RECORD SCDI  : MEXPIMP;
                                       SCII  : MEXPIMP;
                                       SCUI  : MEXPIMP;
                                END;
                   SDOM : MEXP
            END;
TYPR      = RECORD ST     : TREL;
                   S-CONT: BCONT;
            END;
TYPC      = RECORD CASE VTYPC   :  TYP5  OF
                        VTTYPE  :  (CT : TYP);
                        VTYPR   :  (CR : TYPR)
                    END
            END;
T1        = (ENT1, REEL1, BOOL1, CHAIN1, TUP1, ENS1, REL1, FONC);
EXP1      = RECORD CASE VEXP1     : TEXP1 OF
                        VEXPR     : (EXPR1 : PEXP);
                        VCOMMAND  : (EXPC1 : EXP)
                    END
            END;
PPARAM    =  ↑PARAM;
PARAM     = RECORD  NOMP     :  NOM;
                    TYPP     :  TVAL;
                    PARSUIV  :  PPARAM
            END;
```

```
EXPINT    =  RECORD CASE  VE       :  TEXPINT  OF
                          VEXP     : (CINT : PEXP);
                          VFCT     : (NOMFCT : NOM;
                                      LPARAM : PPARAM;
                                      EXPFCT : PEXP)
                    END
             END;

NDEF      =  RECORD  S-TYPE : TYPC;
                     S-EXP  : PEXP
             END;

PTREPTYP  = ↑REPTYP;
PENSE     = ↑ENSEMB;
TMAP      = ↑MAP;
MAP       =  RECORD  CANCIEN  :  NOM;
                     CNOUV    :  NOM;
                     CSUIV    :  TMAP;
             END;

ENSEM     =  RECORD CASE BASTYP : T1  OF
                     ENT1    : (CE : ARRAY [1 ...MAXENS] OF  INTEGER);
                     REEL1   : (CR : ARRAY [1 ...MAXENS] OF REAL);
                     BOOL1   : (CB : ARRAY [1 ... 2] OF BOOLEAN);
                     CHAIN1  : (CCH : ARRAY [1 ... MAXENS] OF PCHAINE);
                     TUP1    : (CTUP: ARRAY [1 ... MAXENS] OF PTREPTYP);
                     ENS1    : (CENS: ARRAY [1 ... MAXENS] OF PENSE)

                     END;

REPTYP    =  RECORD CLONG : INTEGER;
                     CREPTYP : ARRAY [1... MAXCHAMP] OF
                       RECORD NOM CHAMP : NOM ;
                              CASE CATCHAMP : T1 OF

                             ENT1   : (VALE   : INTEGER);
                             REEL1  : (VALR   : REAL);
                             BOOL1  : (VALB   : BOOLEAN);
                             CHAIN1 : (VALCH  : PCHAINE);
                             TUP1   : (VALT   : PTREPTYP);
                             ENS1   : (VALENS : PENSE)

                              END
             END;
```

```
TRES    =  RECORD CASE CODTYP : T1  OF
                  ENT1    : (RES1   : INTEGER);
                  REEL1   : (RES2   : REAL);
                  BOOL1   : (RES3   : BOOLEAN);
                  CHAIN1  : (RES4   : PCHAINE);
                  TUP1    : (RES5   : PTREPTYP);
                  ENS1    : (RES6   : PENSE);
                  REL1    : (REPTYPR : PTREPTYP, ADR : INTEGER)
                  END
           END;

EXP     =  RECORD CATTYP  : T1;
                  REPTYP  : PTREPTYP
                  CASE CVAR : TEXP  OF
                EB : (EXP1 : PEXP, EXP2 : PEXP) CASE OPVAR OF
                     VNOPB  : (NOPB  : TNOPB);
                     VBOPB  : (BOPB  : TBOPB);
                     VROPB  : (ROPB  : TROPB);
                     VOPE   : (OPE   : TOPE);
                     VOPNE  : (OPNE  : TOPNE);
                     VOPSET : (OPSET : TOPSET)
                     END)
           END;

EU      :  (EXPU : PEXP; OPU : TOPU);
APP     :  (CASE VAPP : TAPP OF
                  VREF : (EXPAPP1 : NOM, PAREF : PLEXP);
                  VLAMBDA : (PARF : PPARAM, EXPAPP2 : PEXP, PAREF2 : PLEXP
                  END);
REF     : (ID : NOM);
CONSTR  : (CCONSTR : TCONSTR);
COND    : (CASE VCOND : TCOND OF
                COND1 : (EXPCOND11  : PEXP, EXPCOND12 :PEXP,
                         EXPCOND13  : PEXP);
                COND2 : (EXPCOND2   : T2 EXP)  END);

EXPREAD : (VLU : TVAL);
```

```
EQUIJOIN  : (ID1 : NOM, EXPJ1 : PEXP, ID2 : NOM, COPE : TOPE,
             ID3 : NOM, EXPJ2 : PEXP);
PROJ      : (CPROJ : T2 EXP);
SELECTR   : (CSELECTR : T2 EXP);
SELECTS   : (CLIBRE : NOM, CSELECTS : T2 EXP);
OPEXIST   : (L1 : PLNOM, L2 : PLEXP, EXPEXIST : PEXP);
OPUNIV    : (LE1 : PLNON, LE2 : PLEXP, EXPUNIV : PEXP);
CARTR     : (CCARTR : T2 EXP);
CARTE     : (CCARTE : PPROD);
JOIN      : (JEXP1 : PEXP, IDJ1 : NOM, CROPB : TROPB, IDJ2 : NOM,
             JEXP2 : PEXP);
DIVR      : (DEXP1 : PEXP, LC1 : PLNOM, LC2 : PLNOM, DEXP2 : PEXP);
ANTIPROJ  : (CANTIPROJ : T2 EXP);
SOMR      : (CSOMR : T2 EXP);
MODIFR    : (OP1 : TOP, IDMODIF : NOM, EXPMODIF : PEXP);
UPDATER   : (UOPU : TUOP, IDEX : NOM, EXPUP : PEXP, VL1B1 : NOM,
             CTUPLEUP : CTUPLE);
COMB      : (COP1 : TCOP, ID1 : NOM, COPER : TOPER, EXPCOMB : PEXP);
EXTRCHAIN : (EXCHAIN : PEXP, BINF : PEXP, BSUP : PEXP);
RENOM     : (DMAP : TMAP, EXPRENOM : PEXP);
NEW       : (CNEW : NOM);
OLD       : (COLD : NOM);
APPLY     : (CL1B : NOM, CTUP : CTUPLE, EXPAPPLY : PEXP);
WS        : (CWS  : (WS))
                END
     END;
```

* REPRESENTATION DES ENVIRONNEMENTS *

* 1 ENVIRONNEMENT *

```
VPAGE    =  RECORD PAGESUIV : INTEGER;
                   PLIBRE   : INTEGER;
                   VBUFFER  : ARRAY [1... MAXPAG]  OF CHAR
            END;
PREC1    = ↑REC1;
```

```
REC1   =   RECORD NOM OBJET : NOM;
                  CASE CATOBJET : T1  OF

                  ENT1   : (VALE    : INTEGER);
                  REEL1  : (VALR    : REAL);
                  BOOL1  : (VALB    : BOOLEAN);
                  CHAIN1 : (VALCH   : PCHAINE);
                  TUP1   : (VALREC  : PTREPTYP);
                  ENS1   : (VENSE   : PENSE);
                  REL1   : (VALBUF  : VPAGE, ADRDISK : INTEGER,
                             FICHUT : (0,1), REPTYPR : PTREPTYP);
                  FONC   : (CEXP : PEXP; ENVFCT : ARRAY [1...MAXVAR] OF
                             PREC1);
                  END
           END;

TRO    =   ARRAY [1... MAXRO]  OF REC1;
TRELAT =   FILE OF VPAGE;
```

* 2 ENVIRONNEMENT ρ CONT *

```
REC3   =  RECORD INDIC1    : (0,1);
                 NONCONT1  : NOM;
                 EXPCONT1  : PEXP
          END;

REC4   =  RECORD INDIC2    : (0,1);
                 NONCONT2  : NOM;
                 EXPCONT2  : EXP IMP
          END;
```

```
REC1   =   RECORD NOM OBJET : NOM;
                  CASE CATOBJET : T1  OF

                  ENT1   : (VALE    : INTEGER);
                  REEL1  : (VALR    : REAL);
                  BOOL1  : (VALB    : BOOLEAN);
                  CHAIN1 : (VALCH   : PCHAINE);
                  TUP1   : (VALREC  : PTREPTYP);
                  ENS1   : (VENSE   : PENSE);
                  REL1   : (VALBUF  : VPAGE, ADRDISK : INTEGER,
                             FICHUT : (O,1), REPTYPR : PTREPTYP);
                  FONC   : (CEXP : PEXP; ENVFCT : ARRAY [1...MAXVAR] OF
                             PREC1);
                  END
           END;

TRO    =   ARRAY [1... MAXRO]  OF REC1;
TRELAT =   FILE OF VPAGE;
```

* 2 ENVIRONNEMENT ρ CONT *

```
REC3   =  RECORD INDIC1    : (O,1);
                 NONCONT1  : NOM;
                 EXPCONT1  : PEXP
          END;

REC4   =  RECORD INDIC2    : (O,1);
                 NONCONT2  : NOM;
                 EXPCONT2  : EXP IMP
          END;
```

```
REC2     =  RECORD NONREL : NOM;
                   BKEY   : ARRAY  1 ... MAXCHAMP  OF NOM;
                   BSTAT  : ARRAY  1 ... MAXCONT   OF REC3;
                   BDYNAM : RECORD
                              UP1    : ARRAY  1 ... MAXCONT  OF REC3;
                              DE1    : ARRAY  1 ... MAXCONT  OF REC3;
                              INS1   : ARRAY  1 ... MAXCONT  OF REC3;
                            END;
                   BIMP   : RECORD
                              UP2    : ARRAY  1 ... MAXCONT   OF REC4 ;
                              DE2    : ARRAY  1 ... MAXCONT   OF REC4;
                              INS2   : ARRAY  1 ... MAXCONT   OF REC4
                            END;
                   BDOM   : ARRAY  1 ... MAXCHAMP  OF
                              RECORD : NDOM : NOM;
                                       VDOM : PENSE
                            END;
               END;
TROCONT  =  ARRAY  1 ... MAXROCONT   OF  REC2;
```

* 3 ENVIRONNEMENT $_{m}$ *

```
REPTYP   =  RECORD CLONG : INTEGER; CVAR : ARRAY  1 ...MAXB  OF CHAR  END;
REC5     =  RECORD  CASE CATOB : T1 OF
                      ENT1    : (VALE1 : INTEGER);
                      REEL1   : (VALR1 : REAL);
                      BOOL1   : (VALB1 : BOOLEAN);
                      CHAIN1  : (VALCH1 : ARRAY  1 ... MAXCH  OF CSPEC);
                      TUP1    : (VALRE  : ARRAY  1 ... MAXTUP OF CHAR,
                                                    REPTYP T1 : REPTYP);
                      ENS1    : (VENS : ARRAY  1 ... MENS  OF CHAR,
                                 REPENS : ARRAY  1 ... MREPENS  OF CHAR);
                      REL1    : (VALBUF1 : VPAGE, ADRDISK : INTEGER,
                                 REPTYPR1 : REPTYP)

               END;
```

```
OBJET    =  RECORD INDIC2  : (0,1);
                   NOBJET  : NOM;
                   CASE COBJET  OF

                   ENT1    :  (VALE : INTEGER);
                   REEL1   :  (VALR : REAL);
                   BOOL1   :  (VALB : BOOLEAN);
                   CHAIN1  :  (VALCH: ARRAY [1 ...MAXCH]  OF CSPEC);
                   TUP1    :   (VALREC : ARRAY [1...MAXTUP] OF CHAR,
                                REPTYPT : REPTYP);
                   ENS1    :  (VENS :ARRAY  1...MENS  OF CHAR,
                                REPENS : ARRAY 1...MREPENS  OF CHAR);
                   REL1    :  (VALBUF  : VPAGE, ADRISK : INTEGER,
                               REPTYPR : REPTYP);
                   FONC    : (CEXP : ARRAY [1 ... MAXEXP]  OF CHAR,
                               ENVGLOB :[ARRAY  1 ... MAXVAR] OF
                                     RECORD  NONGLOB : NOM;
                                             VALGLOB : REC5
                   END)
              END;

CONTENU  = RECORD : INDICATEUR : INTEGER;
                    NOMMOD     : NOM;
                    NBREO      : INTEGER;
                    OBJMOD     : ARRAY [1 ... MAXOBJET] OF OBJET
              END;

TROM     = FILE OF CONTENU;

CATALOGUE = FILE OF VPAGE ;
```

* 4 ENVIRONNEMENT ρ_m CONT *

```
REC6     =  RECORD  INDIC   : (0,1);
                    NOMCONT : NOM;
                    EXPCONT :[ARRAY  1 ... MAXEXP] OF CHAR
              END;
```

```
CONTCAT    = RECORD    INDIC :  (0,1);
                       NONREL : NOM;
                       CONTREL : RECORD
                                   BKEY  : ARRAY [1 ... MAXCHAMP] OF NOM;
                                   BSTAT : ARRAY [1 ... MAXCONT]  OF REC5;
                                   BDYNAM RECORD
                                          UP1 : ARRAY [1...MAXCONT] OF REC5;
                                          DE1 : ARRAY [1...MAXCONT] OF REC5;
                                          INS1: ARRAY [1...MAXCONT] OF REC5;
                                        END;
                                   BIMP  RECORD
                                          UP2  : ARRAY [1 ...MAXCONT] OF REC5;
                                          DE2  : ARRAY [1 ...MAXCONT] OF REC5;
                                          INS2 : ARREY [1 ...MAXCONT] OF REC5)
                                        END;
                                   BDOM  ARRAY [1 ... MAXCHAMP] OF
                                          RECORD NDOM : NOM;
                                                 VDOM : (VENS:ARRAY [1...MENS
                                                  OF CHAR],
                                                        REPENS :ARRAY [1...MREPENS
                                                  OF CHAR]);
                                        END;
                         END;
CAT CONT = FILE OF CONT CAT;
```

ANNEXE 2 : DICTIONNAIRE DES MOTS SYMBOLIQUES

ACCES : accès à un champ d'un tuple

ANTIPROJ : antiprojection

APPEL : appel d'une fonction

APPLY : fonction définie dans le chapitre I

BCONT : bloc de contraintes d'intégrité

BOPB : opérateurs binaires booléens

CAR : caractères

CARTE : produit cartésien d'ensembles

CARTR : produit cartésien de relations

CBOOL : constante booléenne

CCHAIN : constante chaîne

CD1 : code prenant l'une des valeurs CS, CB, CI, CD, respectivement pour contrainte statique, contrainte booléenne, contrainte impérative, contraine domaine.

CD2 : code prenant l'une des valeurs I, U, D, respectivement pour INSERT, UPDATE, DELETE.

CDEC : constante décimale

CENT : constante entière

CHIFFRE : caractères compris entre '0' et '9'

CREEL : constante réelle

CSPEC : caractère spécial

CST : constantes

COMB : combinateur (fonction définie dans le chapitre I)

COMP : composant

COND : expression conditionnelle

COND1 : expression conditionnelle if then else

COND2 : expression conditionnelle if then

COP : opérateur prenant l'une des valeurs R, S respectivement, pour relation, ensemble.

CTUPLE : constante tuple

DELETE CONT : destruction d'une contrainte

DIVR : division de deux relations

EB : expression binaire

EIMP : expression impérative

ENS : ensemble

EQUIJOIN : équijoin

EXP : expression

EXP1 : expression ou commande

EXPAPP : expression d'appel de fonction

EXPC : commande

EXPIMP : expression simple ou impérative

EXPINT : expression simple ou fonction

EXPREAD : expression de lecture

EXTRCHAIN : extraction d'une sous-chaîne d'une chaîne

EU : expression unaire

FCT : fonction

GROSMOD : ajout d'objet dans un module

ID : identificateur

JOIN : jonction de deux relations

LAMBDA : application de fonction

LETTRE : lettres de l'alphabet

MAIGRMOD : suppression d'objet d'un module

MDEF : définition d'un module

MODIFR : suppression ou insertion de tuples dans une relation

MODIFC : modification de contrainte

NDEF : définition de nom

NE : nombre entier

NEW : nouvelle valeur d'un champ

NOPB : opérateurs binaires numériques

OLD : ancienne valeur d'un champ

OP : opérateur

OPACC : opérateur indiquant le type d'accès

OPB : opérateurs binaires

OPC : opérateurs d'ajout ou de mise à jour de contrainte

OPNE : opérateurs de non égalité

OPER : opérateurs de la fonction COMB

OPEXIST : opérateur existentiel

OPUNIV : opérateur universel

PROJ : projection

REF : référence

REL : relation

RENOM : opération renommant les champs d'une relation

ROPB : opérateurs binaires de relation

SELECTR : sélection sur une relation

SELECTS : sélection sur un ensemble

SESS : session

SETEXT : ensemble défini en extension

SETINT : ensemble défini en intention

SOMR : somme de deux relations

S-SESS : sous-session

SURNOM : suppression d'un module

TBASE : type de base

TELEM : type élémentaire

TENS : type ensemble

TFCT : type fonction

TREL : type relation

TTUPLE : type tuple

TVAL : type de base ou relation

TYPC : type TYPE ou type RELATION

TYPE : type VAL ou TFCT

TYPR : type RELATION avec contraintes d'intégrité

UOP : opérateurs unaires

UPDATER : modification d'un tuple

WS : espace de travail

B I B L I O G R A P H I E

/AHO 76/ A.V.AHO and J.D.ULLMAN
Universality of data retrieval languages.

/ALA 81/ S. ALAGIC and KULENOVIC
Relational Pascal database interface.
The Computer Journal, Vol. 24, n° 2, 1981.

/BAK 78/ J.J. BACKUS
Can Programming be libertaded from the VON-NEWMAN style ?
A functional style and its algebra of programs.
ACM, vol. 21, n° 2, 1978.

/COD 70/ E.F. CODD
A relational model of data for lar[illegible] shared data banks.
CAMC, vol.13, n° 16, 1970.

/COD 81/ E.F. CODD
The capabilites of relational database management systems.
RJ 3132 (38505) 5.11.81, Computer Science

/DAT 77/ C.J. DATE
An introduction to database systems IBM (UK) laboratories
Ltd, Second Edition.

/DEL 82/ DELOBEL et M. ADIBA
Bases de données et systèmes relationnels.
DUNOD, 1982.

/JON 82/ C.B. JONES
The formal definition of languages.
Ecole d'été d'informatique.
CEA - INRIA - EDF, juillet 1982.

/LIV 78/ C. LIVERCY
Théorie des programmes.
DUNOD, 1978.

/McC 63/ Mc CARTY
A basis for a mathematical theorie computation.
Computer programming and formal system.
North-Holland, Amsterdam.

/SCH 77/ J.W. SCHMIDT
Some high livel language constructs for data of type relation.
ACM, vol. 2, n° 3, 77.

/SMI 77/ J. MILES
Database abstractions aggregation.
ACM, vol. 20, n° 6, 1977.

/SHO 79/ J. SHOPIRO
Theseus : A programming language for relational database.
ACM, vol. 4, n° 4, 1979.

/TEN 76/ R.D. TENNENT
The denotational semantics of programming languages.

/TEN 84/ R.D. TENNENT
Principles of programming languages

/THO 81/ M. THORIN
Langage ADA.
Manuel complet du langage avec exemples.
Edition EYROLLES, 1981

/TUR 76/ D. TURNER
SASL : Language manual
CS/75/1 revised 2.12.76.

/TUR 81/ D. TURNER
The future of applicative programming.
Computing Laboratory. University of Kent
Canterbury, England.