

UNIVERSITE DE GENEVE

PRIMITIVES DE
CREATIONS ET SUPPRESSIONS
DE LA NOUVELLE
BASE DE DONNEES FINANCIERES
DU C E R N
(APPLICATION PBS)

FEVRIER 1985

NICOLAS-OLIVIER BARRELET

INTRODUCTION :

Toute cette étude est basée sur le "Planning, Budgeting System" (PBS). Or, pour comprendre ce système budgétaire, il est indispensable d'avoir un bref aperçu de la structure de fonctionnement du CERN.

1.L'Organisation du CERN :

Le CERN est organisé de manière assez complexe, car il est soumis à un double découpage (Cf. schéma 1) :

1. Celui qui tient compte de la structure des responsabilités, des Divisions, des différents services.
2. Celui qui ne s'occupe que des Domaines et des catégories de recherche.

En effet, pour le CERN, il est indispensable non seulement de savoir combien dépense une Division, un service, etc., mais aussi de connaître le montant des fonds nécessaires à un Domaine de recherche, un Projet, etc.

Ce double découpage s'avère nécessaire aussi bien pour l'établissement des budgets que pour le suivi des dépenses.

1.1 La structure de responsabilités :

Les Divisions représentent le plus haut niveau de regroupement administratif. Elles ont des sphères d'activité bien déterminées : l'une se consacre à la Physique théorique, l'autre à la Physique expérimentale...

Elles sont composées d'unités organiques ou centres de frais.

1.2 Les Domaines de recherche :

Les domaines de recherche (recherche à haute/moyenne/faible énergie...) peuvent être décomposés en différentes Activités Fonctionnelles telles que Détecteurs, Analyse de données, expériences LEP, ou en divers Projets.

Cependant il est évident que ce double découpage ne peut se faire de manière indépendante l'un de l'autre.

Les Activités Fonctionnelles sont exécutées par différentes unités organiques.

Ex.: la "Machine LEP" est une Activité Fonctionnelle.

Sa conception et sa réalisation relève de l'autorité de la "Division LEP", la préparation des expériences du LEP est confiée à la "Division de physique expérimentale" et son entretien incombe à la "Division services Techniques et Bâtiments".

Chaque partie d'une Activité Fonctionnelle confiée à une Division s'appelle une Activité Divisionnaire.

Chaque Activité Divisionnaire confiée à une unité organique s'appelle une Tâche. (Cf. schéma 2).

2. L'Etablissement des Budgets :

Suivant la philosophie du système PBS les budgets sont d'abord alloués par programmes. Ce n'est que dans une deuxième phase que les différents programmes sont ventilés par services.

2.1 Phase un :

Les budgets sont établis pour chaque domaine et chaque catégorie de recherches. Ils sont donc tout d'abord répartis sur les Activités Fonctionnelles.

2.2 Phase deux :

Sur la base des budgets par Activités Fonctionnelles, on peut calculer les budgets autorisés pour chaque Division.

Il sera nécessaire de les répartir à l'intérieur de la structure :

- par Activités Divisionnaires
- par unités organiques.

Là aussi, il y a un lien entre ces deux phases.

Ce lien peut être exprimé grâce à deux règles d'intégrité :

1. La somme des budgets alloués aux Activités Divisionnaires qui sont associées à une Activité Fonctionnelle doit être inférieure ou égale au budget de l'Activité Fonctionnelle.
2. La somme des budgets alloués aux Tâches qui sont associées à une Activité Divisionnaire doit être inférieure ou égale au budget de cette Activité Divisionnaire.

Il faut mentionner aussi que les informations du type budget sont revues mensuellement. Le fait que les budgets soient placés dans des enregistrements de type MTH permet de conserver en quelque sorte un "journal" des budgets qui ont été accordés.

3. Le suivi des dépenses :

Tout système budgétaire digne de ce nom doit non seulement permettre d'établir les budgets, mais aussi de connaître le montant des dépenses afin d'analyser les écarts entre budgets et dépenses réelles.

Ici les dépenses sont de deux types. :

1. les Commandes
2. les autres dépenses.

3.1 Les Commandes :

Les commandes regroupent sous ce nom :

- les demandes de travaux
- les travaux d'ateliers
- les commandes extérieures
- les contrats.

Chaque Commande se fait en deux phases :

1. passation de la commande et création d'un engagement, qui représente une "provision pour dépenses futures".
2. réception de la commande et paiement sur l'engagement.

Il peut arriver qu'un engagement soit insuffisant pour couvrir une dépense prévue; dans ce cas il est possible de modifier cet engagement.

3.2 Les autres dépenses :

Les autres dépenses regroupent :

- les dépenses de stockage
- les transactions inter-divisionnaires
- les dépenses en travail temporaire
- les dépenses de T.V.A.
- les dépenses diverses.

Elles se font sans nécessiter la création d'un engagement.

3.3 Affectation et imputation des dépenses :

Chaque dépense est affectée à une (ou plusieurs) Unité Organique. Celle-ci peut imputer cette dépense aux Tâches qui la constituent de deux manières :

- avec les taux de répartition directs
- avec les taux de répartition à priori.

Les taux de répartition directs servent à affecter une dépense à une seule Tâche, ou sur les mêmes Tâches mais avec des taux de répartition différents des taux à priori. Les taux de répartition à priori servent à imputer les dépenses sur les Tâches dans le cas où ces dépenses ne concernent pas une Tâche spécifique.

Il faut préciser que l'unité organique ne peut avoir que cinq clés de répartition directe et cinq clés de répartition à priori actives en même temps. Cela met en lumière une troisième contrainte d'intégrité :

A un moment donné, il ne peut y avoir que cinq Tâches actives associées à une unité organique.

A partir de cette Tâche, cette dépense est répartie sur tous les niveaux supérieurs :
- l'Activité Divisionnaire
- l'Activité Fonctionnelle
- le Projet.

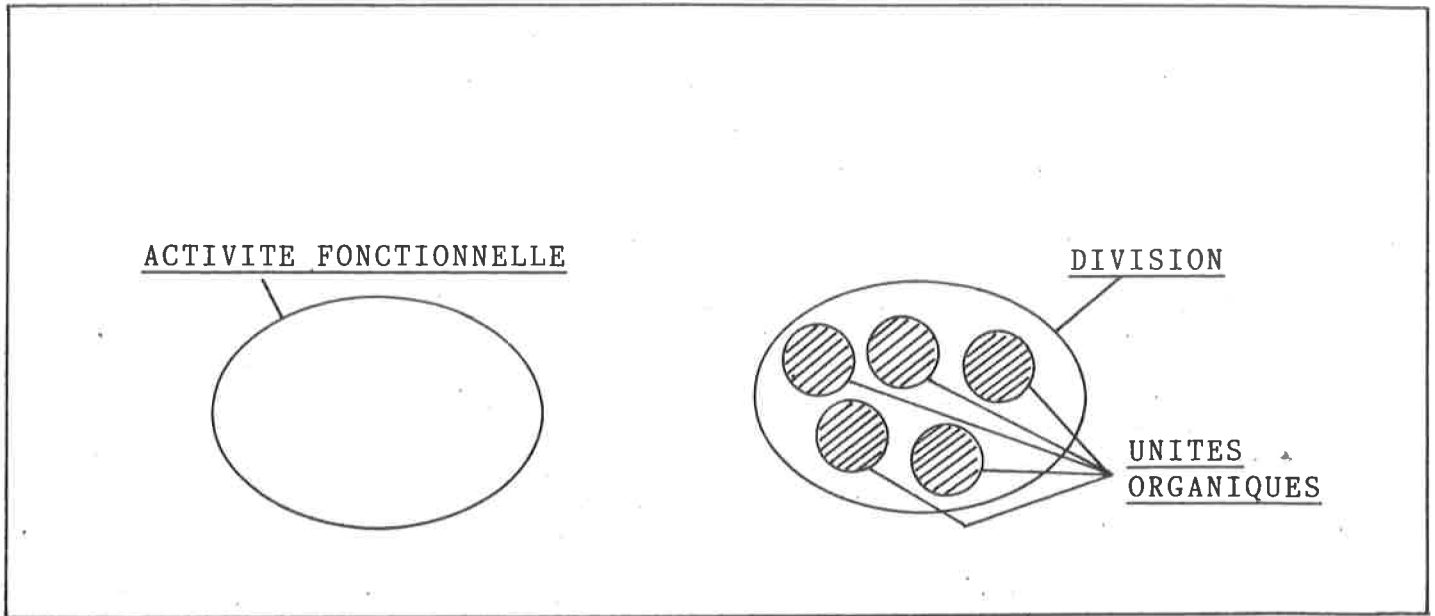
Remerciements :

Je tiens à remercier M. Réginald Bursens pour son aide et ses encouragements.

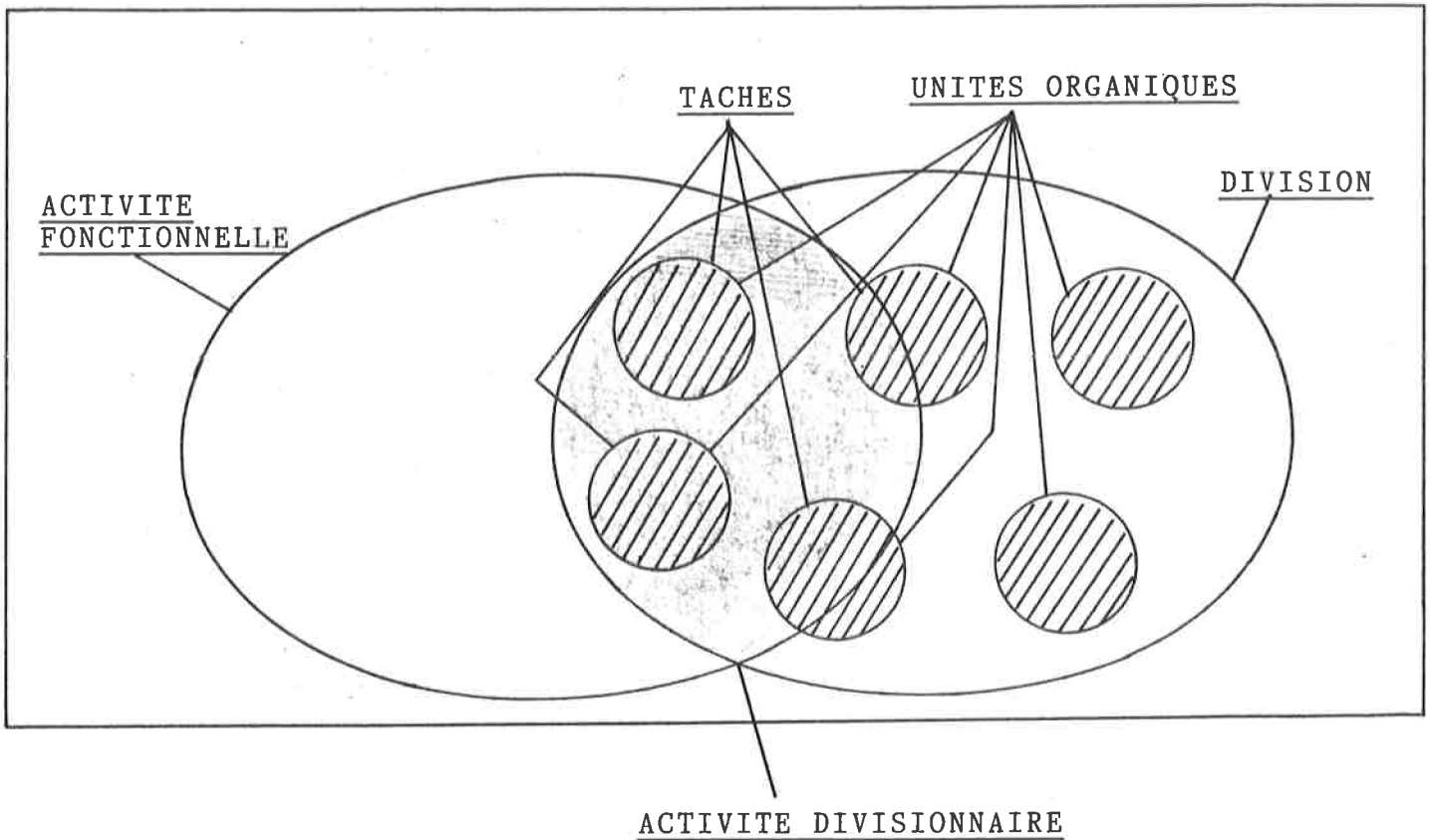
Bibliographie :

M. Leonard	mai 1978	Expression des résultats d'analyse au moyen de schémas de programmes.
M. Leonard	mars 1984	Expression des résultats d'analyse au moyen du modèle relationnel de données et au moyen de schémas de programmes.
B. T. Luong	mai 1983	Etude de la conception d'une nouvelle base de données financières.
Cullinane Database management system	mars 1983	IDMS Database Design and Definition Guide.

SCHEMA 1 : LA DOUBLE REPARTITION



SCHEMA 2 : LA LIAISON



SCHEMA DE PROGRAMME CREANT : FD-OOAK-PJ
FD-PROJECT
FD-PROJECT-STATE
FD-YR-PJ-INFO

FREQUENCE ESTIMEE :
DEGRE D'IMPORTANCE :

DONNEES : ----

RESULTATS : ----

REMARQUES : Ce schema cree la tete de liste FD-OOAK-PJ si elle n'existe pas deja. Il cree aussi un Projet, FD-PROJECT, a condition :

I. QUE DE TOUTE FACON :

1. sa date de fin de validite soit posterieure a sa date de debut de validite.
2. sa date de fin estimee soit posterieure a sa date de debut estimee.

II. QUE S'IL EXISTE UN PROJET AYANT MEME CODE :

1. Celui que l'on veut creer n'ait pas la meme date de debut de validite que celui qui existe.
2. Celui que l'on veut creer ne commence pas durant la periode de validite de celui qui existe deja.

Remarque : si la date de fin de validite du Projet qui existe est egale a Nil alors l'utilisateur ne pourra creer un nouveau Projet de meme code. Il devra tout d'abord aller preciser la date de fin de validite du Projet existant. Ensuite il pourra creer un "duplicate".

3. S'il existe un Projet de meme code et de date de debut de validite posterieure, celui que l'on veut creer finisse avant cette date.

Il cree aussi les FD-PROJECT-STATE associes au Projet (qui indiquent l'etat du Projet). Enfin il cree les YR-PJ-INFO associes aux FD-PROJECT-STATE et permet l'entree des valeurs correspondant au status "AUTHORISED" (FD-AMOUNT-STATUS = "A"), les valeurs pour les autres status ("S" ou "E") etant mises a zero.

dom DATE : article
CC : [0,99] entier
YY : [0,99] entier
MM : [0,99] entier
DD : [0,99] entier

fin
dom DATE-MOIS : article
CC : [0,99] entier
YY : [0,99] entier
MM : [0,99] entier
fin

dom BOOL : [faux, vrai]
dom CHEF : caracteres[30]
dom CODE : caracteres[4]
dom DEPENSES : [-999'999'999.99, 999'999'999.99] reel
dom NOM : caracteres[60]
dom TYPE-OF-STATUS : [A, S, E]

```

cst  FD-AMOUNT-STATUS      :TYPE-OF-STATUS
cst  FD-LAST-UPDATED       :DATE-MOIS
cst  FD-PERIOD-COMPL-DATE  :DATE
cst  FD-PERIOD-START-DATE  :DATE
cst  FD-PJ-CODE            :CODE
cst  FD-PJ-HEADER         :caracteres[2]
cst  FD-PJ-LAST-MANMTH-SMC-REV-DATE :DATE
cst  FD-PJ-LAST-MANMTH-TL-REV-DATE  :DATE
cst  FD-PJ-LAST-MANYR-REV-DATE      :DATE
cst  FD-PJ-LAST-MATL-COST-REV-DATE   :DATE
cst  FD-PJ-LEADER                  :CHEF
cst  FD-PJ-LEADER-DIV              :caracteres[3]
cst  FD-PJ-MANMTH-SMC              : tableau [10] de DEPENSES
cst  FD-PJ-MANMTH-TL               : tableau [10] de DEPENSES
cst  FD-PJ-MANYR                   : tableau [10] de DEPENSES
cst  FD-PJ-MATL-COST               : tableau [10] de DEPENSES
cst  FD-PJ-NAME                    :NOM]
cst  FD-PJ-REVISED                 :caractere[1]
cst  FD-VALID-PERIOD-END-DATE       :DATE et Nil
cst  FD-VALID-PERIOD-START-DATE     :DATE

rel  FD-OOAK-PJ (FD-PJ-HEADER)
      ayant cle  FD-PJ-HEADER

rel  FD-PROJECT (FD-PJ-CODE,FD-PJ-REVISED,FD-PJ-NAME,FD-PJ-LEADER,
      FD-PJ-LEADER-DIV,FD-VALID-PERIOD-START-DATE,
      FD-VALID-PERIOD-END-DATE,FD-PERIOD-START-DATE,
      FD-PERIOD-COMPL-DATE)
      ayant cle  FD-PJ-CODE

rel  FD-PROJECT-STATE (FD-PJ-CODE,FD-AMOUNT-STATUS,FD-VALID-PERIOD-START-DATE,
      FD-VALID-PERIOD-END-DATE)
      ayant cle  FD-PJ-CODE,FD-AMOUNT-STATUS

rel  FD-YR-PJ-INFO (FD-PJ-CODE,FD-AMOUNT-STATUS,FD-LAST-UPDATED,
      FD-PJ-LAST-MATL-COST-REV-DATE,
      FD-PJ-MATL-COST,FD-PJ-LAST-MANYR-REV-DATE,
      FD-PJ-MANYR,FD-PJ-LAST-MANMTH-TL-REV-DATE,
      FD-PJ-MANMTH-TL,FD-PJ-LAST-MANMTH-SMC-REV-DATE,
      FD-PJ-MANMTH-SMC)
      ayant cle  FD-PJ-CODE,FD-AMOUNT-STATUS,FD-LAST-UPDATED

cdd  CDD-OOAK-PJ           :FD-OOAK-PJ
cdd  CDD-PJ-1              :FD-PROJECT
cdd  CDD-PJ-2              :FD-PROJECT
cdd  CDD-PJ-3              :FD-PROJECT
cdd  CDD-PJ-STATE          :FD-PROJECT-STATE
cdd  CDD-PROJECT           :FD-PROJECT
cdd  CDD-PROJECT-1         :FD-PROJECT
cdd  CDD-YR-INFO           :FD-YR-PJ-INFO

var  bool                  :BOOL
var  compteur              :[0,9] entier
var  existant              :BOOL
var  existe-post           :BOOL
var  info-a                :FD-YR-PJ-INFO
var  info-e                :FD-YR-PJ-INFO
var  info-s                :FD-YR-PJ-INFO
var  oak                   :FD-OOAK-PJ et Nil

```

```

var pro                :FD-PROJECT
var pro-ex             :FD-PROJECT
var reponse            :caractere[1]
var sta-a              :FD-PROJECT-STATE
var sta-e              :FD-PROJECT-STATE
var sta-s              :FD-PROJECT-STATE

```

PROCEDURE INITIALISATION YR-PJ-INFO (info : FD-YR-PJ-INFO)

```

info.fd-last-updated := date actuelle /* Annee et Mois */
info.fd-pj-last-matl-cost-rev-date := date actuelle /* Annee, Mois, Jour */
compteur := 0
    Tant que compteur <= 9 faire
        info.fd-pj-matl-cost[compteur] := 0
        compteur := compteur + 1
    Refaire
info.fd-pj-last-manyr-rev-date := date actuelle /* Annee, Mois, Jour */
compteur := 0
    Tant que compteur <= 9 faire
        info.fd-pj-manyr[compteur] := 0
        compteur := compteur + 1
    Refaire
info.fd-pj-last-manmth-tl-rev-date := date actuelle /* Annee, Mois, Jour */
compteur := 0
    Tant que compteur <= 9 faire
        info.fd-pj-manmth-tl[compteur] := 0
        compteur := compteur + 1
    Refaire
info.fd-pj-last-manmth-smc-rev-date := date actuelle /* CC,YY,MM,DD */
compteur := 0
    Tant que compteur <= 9 faire
        info.fd-pj-manmth-smc[compteur] := 0
        compteur := compteur + 1
    Refaire
finprocedure

```

SCHEMA DE PROGRAMME CREATION

```

/* ENTREE DU CODE-PROJET */
/* ET CREATION DE LA TETE DE LISTE */
/* SI ELLE N'EXISTE PAS DEJA */
existant := faux
existe-post := faux
Afficher ('Entrez le code du Projet :')
Repondre (pro.fd-pj-code)
oak.fd-pj-header := ' '
oak := prendre dans CDD-OOAK-PJ avec cle = oak.fd-pj-header
    si oak = Nil alors
        oak est element de CDD-OOAK-PJ
    finsi

/* CREATION DU PROJET */

CDD-PROJECT-1 := vide /* La CDD-PROJECT-1 contiendra tous les Projets
                        ayant meme code que le Projet que l'on veut creer.*/
CDD-PROJECT-1 := CDD-PROJECT filtre par (FD-PJ-CODE = pro.fd-pj-code)
    si CDD-PROJECT-1 <> vide alors
        existant := vrai
    finsi
Afficher ('Entrez le statut du Projet (revise ou non) :')
Repondre (pro.fd-pj-revised)

```

```

Afficher ('Entrez le nom du Projet :')
Repondre (pro.fd-pj-name)
Afficher ('Entrez le nom du responsable du Projet :')
Repondre (pro.fd-pj-leader)
Afficher ('Entrez l'abreviation du nom de la Division responsable
de ce Projet :')
Repondre (pro.fd-pj-leader-div)
Afficher ('Entrez la date de debut de validite du Projet :')
ENTRER DATE (pro.fd-valid-period-start-date)

```

```

/* VERIFICATION QU'IL N'EXISTE PAS DEJA UN PROJET */
/* AYANT MEME CODE ET MEME DATE DE DEBUT DE VALIDITE */

```

```

si existant = vrai alors

```

```

CDD-PJ-1 := vide /* CDD-PJ-1 contiendra le Projet ayant meme code et
meme date de debut de validite, s'il existe. */
CDD-PJ-1 := CDD-PROJECT-1 filtre par (FD-VALID-PERIOD-START-DATE =
pro.fd-valid-period-start-date)

```

```

si CDD-PJ-1 <> vide alors
Afficher ('Un projet ayant meme code
et meme date de debut de validite
existe deja.Creation impossible.')

```

```

finschema

```

```

sinon

```

```

/* VERIFICATION QUE LE PROJET QUE L'ON VEUT CREER */
/* NE COMMENCE PAS DURANT LA PERIODE DE VALIDITE */
/* D'UN AUTRE PROJET */

```

```

CDD-PJ-2 := vide /* CDD-PJ-2 contiendra le Projet
valide pour cette periode, s'il
existe */

```

```

CDD-PJ-2 := CDD-PROJECT-1 filtre par
(FD-VALID-PERIOD-START-DATE < pro.fd-valid-period-start-date
et FD-VALID-PERIOD-END-DATE > pro.fd-valid-period-start-date
ou FD-VALID-PERIOD-END-DATE = Nil)

```

```

si CDD-PJ-2 <> vide alors
Afficher ('Le Projet que vous voulez creer serait
"contenu" dans un autre Projet deja
existant.Creation impossible.)

```

```

finschema

```

```

sinon

```

```

CDD-PJ-3 := vide /* CDD-PJ-3 contiendra les Projets
dont la date de debut de validite
est posterieure a celle du Projet
que l'on veut creer. */

```

```

CDD-PJ-3 := CDD-PROJECT-1 filtre par
(FD-VALID-PERIOD-START-DATE >
pro.fd-valid-period-start-date)
trie croissant par
(FD-VALID-PERIOD-START-DATE)

```

```

si CDD-PJ-3 <> vide alors
existe-post := vrai
pro-ex := prendre dans CDD-PJ-3

```

```

finsi

```

```

finsi

```

```

finsi

```

```

finsi

```

```

si existe-post = vrai alors

```



```
/* VERIFICATION QUE LA DATE DE FIN DE VALIDITE DU PROJET SOIT */
/* POSTERIEURE A SA DATE DE DEBUT DE VALIDITE */
```

```
Afficher ('Entrez la date de fin de validite du Projet :')
ENTRER DATE (pro.f-valid-period-end-date)
si pro.fd-valid-period-end-date <=
    pro.fd-valid-period-start-date
    alors
    Afficher ('La date de fin de validite est anterieure a la
        date de debut de validite.C'est impossible.')
    finschema
finsi
```

```
/* VERIFICATION : S'IL EXISTE UN PROJET DE MEME CODE ET DE DATE */
/* DE DEBUT DE VALIDITE POSTERIEURE, CELUI QUE L'ON VEUT CREER */
/* FINISSE AVANT CETTE DATE */
```

```
si pro-ex.fd-valid-period-start-date
    <= pro.fd-valid-period-end-date alors
    Afficher ('La date de fin de validite du Projet est
        posterieure ou egale a la date de debut de
        validite d'un Projet deja existant.
        C'est impossible.')
```

finschema

finsi

sinon

reponse := ' '

Tant que reponse <> 'O' et reponse <> 'N' faire

Afficher ('Voulez-vous rentrer la date de fin de validite
 du Projet ? (O/N)')

Repondre (reponse)

Refaire

si reponse = 'O' alors

Afficher ('Entrez la date de fin de validite du Projet')
 ENTRER DATE (pro.fd-valid-period-end-date)

si pro.fd-valid-period-end-date <=
 pro.fd-valid-period-start-date alors

Afficher ('La date de fin de validite
 est anterieure a la date
 de debut de validite.
 C'est impossible.')

finschema

finsi

sinon si reponse = 'N' alors
 pro.fd-valid-period-end-date := Nil

finsi

finsi

finsi

bool := faux

Tant que bool = faux faire

Afficher ('Entrez la date estimee de debut du Projet :')
 ENTRER DATE (pro.fd-period-start-date)

Afficher ('Entrez la date estimee de fin du Projet :')
 ENTRER DATE (pro.fd-period-compl-date)

si pro.fd-period-start-date >= pro.fd-period-compl-date
 alors

Afficher ('La date estimee de debut du Projet est
 posterieure ou egale a la date
 estimee de fin du Projet.')

C'est impossible.')

sinon bool := vrai
finsi

Refaire

pro est element de CDD-PROJET
LIER pro avec oak

/* ENTREE DES FD-PROJECT-STATE */

staa.fd-amount-status := 'A'
stae.fd-amount-status := 'E'
stas.fd-amount-status := 'S'
bool := faux

Tant que bool = faux faire

Afficher ('Veuillez entrer les dates de validite qui concernent
le Projet pour le status Autorise = A')

VALIDER DATE DEBUT-FIN (sta-a.fd-valid-period-start-date,
sta-a.fd-valid-period-end-date)

si sta-a.fd-valid-period-start-date < pro.fd-valid-period-start-date
alors

Afficher ('La date de debut de validite est inferieure a celle
du Projet.C'est impossible.')

sinon si sta-a.fd-valid-period-end-date < Nil et
pro.fd-valid-period-end-date < Nil et
sta-a.fd-valid-period-end-date < pro.fd-valid-period-end-date alors

Afficher ('La date de fin de validite est posterieure
a celle du Projet.C'est impossible.')

sinon bool := vrai
finsi

finsi

Refaire

sta-e.fd-valid-period-start-date := sta-a.fd-valid-period-start-date
sta-e.fd-valid-period-end-date := sta-a.fd-valid-period-end-date
sta-s.fd-valid-period-start-date := sta-a.fd-valid-period-start-date
sta-s.fd-valid-period-end-date := sta-a.fd-valid-period-end-date
sta-a est element de CDD-PJ-STATE
sta-e est element de CDD-PJ-STATE
sta-s est element de CDD-PJ-STATE
LIER pro avec sta-a
LIER pro avec sta-e
LIER pro avec sta-s

/* ENTREE DE FD-YR-PJ-INFO */

reponse := ' '

Tant que reponse <> 'O' ou reponse <> 'N' faire

Afficher ('Voulez-vous entrer les informations qui concernent
le Projet pour le status "Autorise" ? (O/N)')

Repondre (reponse)

Refaire

si reponse = 'O' alors

/* ENTREE MANUELLE DES VALEURS POUR LE STATUS = A */

info-a.fd-last-updated := date actuelle /* Annee et Mois/
info-a.fd-pj-last-matl-cost-rev-date := date actuelle
compteur := 0

Tant que compteur <= 9 faire

```

    Afficher ('Depenses materielles estimees pour
              1'annee :', annee actuelle + compteur)
    Repondre (info-a.fd-pj-matl-cost[compteur])
    compteur := compteur + 1
    Refaire
info-a.fd-pj-last-manyr-rev-date := date actuelle
compteur := 0
    Tant que compteur <= 9 faire
        Afficher ('Depenses en frais de personnel pour
                  1'annee :', annee actuelle + compteur)
        Repondre (info-a.fd-pj-manyr[compteur])
        compteur := compteur + 1
    Refaire
info-a.fd-pj-last-manmth-tl-rev-date := date actuelle
compteur := 0
    Tant que compteur <= 9 faire
        Afficher ('Depenses en travail temporaire exprime en
                  homme/mois pour 1'annee :',
                  annee actuelle + compteur)
        Repondre (info-a.fd-pj-manmth-tl[compteur])
        compteur := compteur + 1
    Refaire
info-a.fd-pj-manmth-smc-rev-date := date actuelle
compteur := 0
    Tant que compteur <= 9 faire
        Afficher ('Depenses de maintenance exprimees en
                  homme/mois pour 1'annee :',
                  annee actuelle + compteur)
        Repondre (info-a.fd-pj-manmth-smc[compteur])
        compteur := compteur + 1
    Refaire

    sinon INITIALISATION YR-PJ-INFO (info-a)
    finsi
INITIALISATION YR-PJ-INFO (info-e)
INITIALISATION YR-PJ-INFO (info-s)
info-a est element de CDD-YR-INFO
info-e est element de CDD-YR-INFO
info-s est element de CDD-YR-INFO
LIER sta-a avec info-a
LIER sta-e avec info-e
LIER sta-s avec info-s

finschema

```

SCHEMA DE PROGRAMME CREANT : FD-PJ-OU-STATUS
FD-YR-OU-COST

FREQUENCE ESTIMEE :
DEGRE D'IMPORTANCE :

DONNEES : ---

RESULTATS : ---

REMARQUES : Ce schema de programme cree un FD-PJ-OU-STATUS a condition que :

1. le Projet associe existe, et qu'il ne soit pas clos.
2. l'Unite Organique existe, et qu'elle ne soit pas close.
3. le Projet et l'Unite Organique ne soient pas deja associes.
4. la date de debut de validite du FD-PROJECT-OU-STATUS soit posterieure ou egale a celle du Projet.
5. la date de debut de validite du FD-PROJECT-OU-STATUS soit posterieure ou egale a celle de l'Unite Organique.

Cette procedure de creation manuelle pourrait etre remplacee par une creation automatique a partir des Taches, selon les types de depenses.

On cree aussi le FD-YR-OU-COST associe au FD-PROJECT-OU-STATUS en initialisant toutes les valeurs a zero.

dom CODE-ORG : article
 FD-DIVISION-CODE : CODE-DIV
 FD-SECTION-CODE : CODE+SEC
 fin

dom DATE : article
 CC : [0,99] entier
 YY : [0,99] entier
 MM : [0,99] entier
 DD : [0,99] entier
 fin

dom DATE-MOIS : article
 CC : [0,99] entier
 YY : [0,99] entier
 MM : [0,99] entier
 fin

dom BOOL : [faux,vrai]
dom CAR-UN : caractere[1]
dom CHEF : caracteres[30]
dom CODE : caracteres[4]
dom CODE-DIV : [0,99] entier
dom CODE-SEC : [0,999] entier
dom DEPENSES : [-999'999'999.99,999'999'999.99] reel
dom FERME : [0,C]
dom LEVEL : [-999,999] entier
dom NOM : caracteres[30]
dom TYPE-DE-DEPENSE : [W,T,L,M,P,R,E,S]

cst FD-EXTRACTION-LEVEL : LEVEL
cst FD-DIVISION-CODE : CODE-DIV
cst FD-LAST-UPDATED : DATE-MOIS
cst FD-ORGANIC-CODE : CODE-ORG
cst FD-OU-CLOSED : FERME
cst FD-OU-LEADER : CHEF

```

cst  FD-OU-MATL-COST           : tableau [4] de DEPENSES
cst  FD-OU-NAME                : NOM
cst  FD-PERIOD-COMPL-DATE      : DATE
cst  FD-PERIOD-START-DATE      : DATE
cst  FD-PJ-CODE                : CODE
cst  FD-PJ-LEADER              : CHEF
cst  FD-PJ-LEADER-DIV          : caracteres[3]
cst  FD-PJ-NAME                : NOM
cst  FD-PJ-REVISED            : CAR-UN
cst  FD-PROJECT-EXPENSE-TYPE   : TYPE-DE-DEPENSE
cst  FD-SECTION-CODE           : CODE-SEC
cst  FD-SIGNING-LEVEL          : LEVEL
cst  FD-VALID-PERIOD-END-DATE   : DATE et Nil
cst  FD-VALID-PERIOD-START-DATE : DATE

rel  FD-PROJECT (FD-PJ-CODE,FD-PJ-REVISED,FD-PJ-NAME,FD-PJ-LEADER,
                  FD-PJ-LEADER-DIV,FD-VALID-PERIOD-START-DATE,
                  FD-VALID-PERIOD-END-DATE,FD-PERIOD-START-DATE,
                  FD-PERIOD-COMPL-DATE)
      ayant cle  FD-PJ-CODE

rel  FD-PJ-OU-STATUS (FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE,
                      FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE)
      ayant cle  FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE

rel  FD-YR-OU-COST (FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE,
                    FD-LAST-UPDATED,FD-OU-MATL-COST)
      ayant cle  FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE,
                    FD-LAST-UPDATED

rel  FD-ORGANIC-UNIT (FD-ORGANIC-CODE,FD-OU-NAME,FD-OU-LEADER,
                      FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE,
                      FD-EXTRACTION-LEVEL,FD-SIGNING-LEVEL,FD-OU-CLOSED)
      ayant cle  FD-ORGANIC-CODE

cdd  CDD-ORGANIC-UNIT          :FD-ORGANIC-UNIT
cdd  CDD-ORGUNIT-2             :FD-ORGANIC-UNIT
cdd  CDD-PJ-2                  :FD-PROJECT
cdd  CDD-PJ-OU-STATUS          :FD-PJ-OU-STATUS
cdd  CDD-PROJECT               :FD-PROJECT
cdd  CDD-YR-OU-COST            :FD-YR-OU-COST

var  bool                      :BOOL
var  compteur                  :[1,4] entier
var  lecture                    :BOOL-LECT
var  orgunit                   :FD-ORGANIC-UNIT
var  pro                       :FD-PROJECT
var  status                    :FD-PJ-OU-STATUS
var  yourcost                  :FD-YR-OU-COST

```

SCHEMA DE PROGRAMME CREATION

```

/* ENTREE DU CODE-PROJET */

```

```

bool := faux
  Tant que bool = faux faire
    Afficher ('Entrez le code du Projet :')
    Repondre (pro.fd-pj-code)
    Afficher ('Entrez la date de debut de validite du Projet :')
    ENTRER DATE (pro.fd-valid-period-start-date)

```

```

CDD-PJ-2 := vide /* CDD-PJ-2 contiendra le Projet qui a
                    ce code et cette date de debut de
                    valide, s'il existe. */
CDD-PJ-2 := CDD-PROJECT filtre par (FD-PJ-CODE =
    pro.fd-pj-code et FD-VALID-PERIOD-START-DATE =
    pro.fd-valid-period-start-date)
si CDD-PJ-2 = vide alors
    Afficher ('Il n'existe aucun Projet ayant ce
                code et cette date de debut de
                valide.Veuillez recommencer.')
sinon bool := vrai
    pro := prendre dans CDD-PJ-2
    status.fd-pj-code := pro.fd-pj-code
finsi

```

Refaire

/* VERIFICATION : LE PROJET NE DOIT PAS ETRE CLOS */

```

si pro.fd-valid-end-date <= date actuelle alors
    Afficher ('Le Projet est clos.Il n'est plus possible
                de lui ajouter un Project-status.')
finschema

```

sinon

/* ENTREE DU CODE DE L'UNITE ORGANIQUE */

bool := faux

Tant que bool = faux faire

```

    Afficher ('Entrez le code organique :')
    Afficher ('1! Entrez le code de la Division :')
    Repondre (orgunit.fd-division-code)
    Afficher ('2! Entrez le code de la Section :')
    Repondre (orgunit.fd-section-code)
    Afficher ('Entrez la date de debut de valide de
                l'Unite Organique :')
    ENTRER DATE (orgunit.fd-valid-period-start-date)
    CDD-ORGUNIT-2 := vide /* CDD-ORGUNIT-2 contiendra l'Unite
                            organique ayant ce code et cette
                            date de debut de valide, si elle
                            existe. */

```

```

CDD-ORGUNIT-2 := CDD-ORGANIC-UNIT filtre par
    (FD-ORGANIC-CODE = orgunit.fd-organic-code
     et FD-VALID-PERIOD-START-DATE =
     orgunit.fd-valid-period-start-date)

```

```

si CDD-ORGUNIT-2 = vide alors
    Afficher ('Il n'existe aucune Unite Organique
                ayant ce code et cette date de
                debut de valide.Veuillez
                recommencer.')

```

```

sinon bool := vrai
    orgunit := prendre dans CDD-ORGUNIT-2
    status.fd-organic-code
                := orgunit.fd-organic-code

```

finsi

Refaire

/* VERIFICATION : L'UNITE ORGANIQUE NE DOIT PAS ETRE CLOSE */

```

si orgunit.fd-valid-period-end-date <= date actuelle alors
    Afficher ('L'Unite Organique concerne est close.Il n'est

```

plus possible de lui associer un FD-PJ-OU-STATUS')

finschema

sinon

/* VERIFICATION : LE PROJET N'EST PAS DEJA */
/* ASSOCIE A CETTE UNITE ORGANIQUE */

Afficher ('Entrez le type de depenses :')
Repondre (status.fd-project-expense-type)
status := rche dans CDD-PJ-OU-STATUS
avec cle = status[fd-organic-code,fd-pj-code,
fd-project-expense-type]

si status <> Nil alors
Afficher ('Ce FD-PJ-OU-STATUS est deja associe
au Projet';pro.fd-pj-code;'.Il est
donc impossible de le creer.')

finschema

finsi

/* CREATION DE FD-PJ-OU-STATUS */

Afficher ('Entrez la date de debut de validee :')
ENTRER DATE (status.fd-valid-period-start-date)
si status.fd-valid-period-start-date <
pro.fd-valid-period-start-date alors
Afficher ('La date de debut de validee du
PROJECT-OU-STATUS est anterieure a la date
de debut de validee du Projet.C'est
impossible.')

finschema

sinon si status.fd-valid-period-start-date <
orgunit.fd-valid-period-start-date alors
Afficher ('La date de debut de validee du
PROJECT-OU-STATUS est anterieure a
la date de debut de validee de
l'Unite Organique.C'est impossible.')

finschema

finsi

finsi

status.fd-valid-period-end-date := Nil
status est element de CDD-PJ-OU-STATUS

LIER status avec pro

/* ENTREE DE FD-YR-OU-COST */

yroucost.fd-last-updated := date actuelle /* CC,YY,MM */
compteur := 1

Tant que compteur <= 4 faire
yroucost.fd-ou-matl-cost[compteur] := 0
compteur := compteur + 1

Refaire

yroucost est element de CDD-YR-OU-COST

LIER yroucost avec status

finsi

finsi

finschema

C'est impossible.')

sinon bool := vrai

finsi

Refaire

pro est element de CDD-PROJET

LIER pro avec oak

/* ENTREE DES FD-PROJECT-STATE */

staa.fd-amount-status := 'A'

stae.fd-amount-status := 'E'

stas.fd-amount-status := 'S'

bool := faux

Tant que bool = faux faire

Afficher ('Veuillez entrer les dates de validite qui concernent
le Projet pour le status Autorise = A')

VALIDER DATE DEBUT-FIN (sta-a.fd-valid-period-start-date,
sta-a.fd-valid-period-end-date)

si sta-a.fd-valid-period-start-date < pro.fd-valid-period-start-date
alors

Afficher ('La date de debut de validite est inferieure a celle
du Projet.C'est impossible.)

sinon si sta-a.fd-valid-period-end-date < Nil et
pro.fd-valid-period-end-date < Nil et
sta-a.fd-valid-period-end-date >

pro.fd-valid-period-end-date alors

Afficher ('La date de fin de validite est posterieure
a celle du Projet.C'est impossible.')

sinon bool := vrai

finsi

finsi

Refaire

sta-e.fd-valid-period-start-date := sta-a.fd-valid-period-start-date

sta-e.fd-valid-period-end-date := sta-a.fd-valid-period-end-date

sta-s.fd-valid-period-start-date := sta-a.fd-valid-period-start-date

sta-s.fd-valid-period-end-date := sta-a.fd-valid-period-end-date

sta-a est element de CDD-PJ-STATE

sta-e est element de CDD-PJ-STATE

sta-s est element de CDD-PJ-STATE

LIER pro avec sta-a

LIER pro avec sta-e

LIER pro avec sta-s

/* ENTREE DE FD-YR-PJ-INFO */

reponse := ' '

Tant que reponse <> 'O' ou reponse <> 'N' faire

Afficher ('Voulez-vous entrer les informations qui concernent
le Projet pour le status "Autorise" ? (O/N)')

Repondre (reponse)

Refaire

si reponse = 'O' alors

/* ENTREE MANUELLE DES VALEURS POUR LE STATUS = A */

info-a.fd-last-updated := date actuelle /* Annee et Mois/

info-a.fd-pj-last-matl-cost-rev-date := date actuelle

compteur := 0

Tant que compteur <= 9 faire


```

    Afficher ('Depenses materielles estimees pour
              1'annee :', annee actuelle + compteur)
    Repondre (info-a.fd-pj-matl-cost[compteur])
    compteur := compteur + 1
    Refaire
info-a.fd-pj-last-manyr-rev-date := date actuelle
compteur := 0
    Tant que compteur <= 9 faire
        Afficher ('Depenses en frais de personnel pour
                  1'annee :', annee actuelle + compteur)
        Repondre (info-a.fd-pj-manyr[compteur])
        compteur := compteur + 1
    Refaire
info-a.fd-pj-last-manmth-tl-rev-date := date actuelle
compteur := 0
    Tant que compteur <= 9 faire
        Afficher ('Depenses en travail temporaire exprime en
                  homme/mois pour 1'annee :',
                  annee actuelle + compteur)
        Repondre (info-a.fd-pj-manmth-tl[compteur])
        compteur := compteur + 1
    Refaire
info-a.fd-pj-manmth-smc-rev-date := date actuelle
compteur := 0
    Tant que compteur <= 9 faire
        Afficher ('Depenses de maintenance exprimees en
                  homme/mois pour 1'annee :',
                  annee actuelle + compteur)
        Repondre (info-a.fd-pj-manmth-smc[compteur])
        compteur := compteur + 1
    Refaire

    sinon INITIALISATION YR-PJ-INFO (info-a)
    finsi
INITIALISATION YR-PJ-INFO (info-e)
INITIALISATION YR-PJ-INFO (info-s)
info-a est element de CDD-YR-INFO
info-e est element de CDD-YR-INFO
info-s est element de CDD-YR-INFO
LIER sta-a avec info-a
LIER sta-e avec info-e
LIER sta-s avec info-s

```

finschema

SCHEMA DE PROGRAMME CREANT : FD-PJ-OU-STATUS
FD-YR-OU-COST

FREQUENCE ESTIMEE :
DEGRE D'IMPORTANCE :

DONNEES : ---

RESULTATS : ---

REMARQUES : Ce schema de programme cree un FD-PJ-OU-STATUS a condition que :

1. le Projet associe existe, et qu'il ne soit pas clos.
2. l'Unite Organique existe, et qu'elle ne soit pas close.
3. le Projet et l'Unite Organique ne soient pas deja associes.
4. la date de debut de validite du FD-PROJECT-OU-STATUS soit posterieure ou egale a celle du Projet.
5. la date de debut de validite du FD-PROJECT-OU-STATUS soit posterieure ou egale a celle de l'Unite Organique.

Cette procedure de creation manuelle pourrait etre remplacee par une creation automatique a partir des Taches, selon les types de depenses.

On cree aussi le FD-YR-OU-COST associe au FD-PROJECT-OU-STATUS en initialisant toutes les valeurs a zero.

dom CODE-ORG : article
FD-DIVISION-CODE : CODE-DIV
FD-SECTION-CODE : CODE+SEC
fin

dom DATE : article
CC : [0,99] entier
YY : [0,99] entier
MM : [0,99] entier
DD : [0,99] entier
fin

dom DATE-MOIS : article
CC : [0,99] entier
YY : [0,99] entier
MM : [0,99] entier
fin

dom BOOL : [faux,vrai]
dom CAR-UN : caractere[1]
dom CHEF : caracteres[30]
dom CODE : caracteres[4]
dom CODE-DIV : [0,99] entier
dom CODE-SEC : [0,999] entier
dom DEPENSES : [-999'999'999.99,999'999'999.99] reel
dom FERME : [0,C]
dom LEVEL : [-999,999] entier
dom NOM : caracteres[30]
dom TYPE-DE-DEPENSE : [W,T,L,M,P,R,E,S]

cst FD-EXTRACTION-LEVEL : LEVEL
cst FD-DIVISION-CODE : CODE-DIV
cst FD-LAST-UPDATED : DATE-MOIS
cst FD-ORGANIC-CODE : CODE-ORG
cst FD-OU-CLOSED : FERME
cst FD-OU-LEADER : CHEF

```

cst  FD-OU-MATL-COST           : tableau [4] de DEPENSES
cst  FD-OU-NAME                :NOM
cst  FD-PERIOD-COMPL-DATE      :DATE
cst  FD-PERIOD-START-DATE      :DATE
cst  FD-PJ-CODE                :CODE
cst  FD-PJ-LEADER              :CHEF
cst  FD-PJ-LEADER-DIV          :caracteres[3]
cst  FD-PJ-NAME                :NOM
cst  FD-PJ-REVISED            :CAR-UN
cst  FD-PROJECT-EXPENSE-TYPE   :TYPE-DE-DEPENSE
cst  FD-SECTION-CODE           :CODE-SEC
cst  FD-SIGNING-LEVEL          :LEVEL
cst  FD-VALID-PERIOD-END-DATE   :DATE et Nil
cst  FD-VALID-PERIOD-START-DATE :DATE

rel  FD-PROJECT (FD-PJ-CODE,FD-PJ-REVISED,FD-PJ-NAME,FD-PJ-LEADER,
                FD-PJ-LEADER-DIV,FD-VALID-PERIOD-START-DATE,
                FD-VALID-PERIOD-END-DATE,FD-PERIOD-START-DATE,
                FD-PERIOD-COMPL-DATE)
        ayant cle  FD-PJ-CODE

rel  FD-PJ-OU-STATUS (FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE,
                FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE)
        ayant cle  FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE

rel  FD-YR-OU-COST (FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE,
                FD-LAST-UPDATED,FD-OU-MATL-COST)
        ayant cle  FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE,
                FD-LAST-UPDATED

rel  FD-ORGANIC-UNIT (FD-ORGANIC-CODE,FD-OU-NAME,FD-OU-LEADER,
                FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE,
                FD-EXTRACTION-LEVEL,FD-SIGNING-LEVEL,FD-OU-CLOSED)
        ayant cle  FD-ORGANIC-CODE

cdd  CDD-ORGANIC-UNIT          :FD-ORGANIC-UNIT
cdd  CDD-ORGUNIT-2             :FD-ORGANIC-UNIT
cdd  CDD-PJ-2                  :FD-PROJECT
cdd  CDD-PJ-OU-STATUS          :FD-PJ-OU-STATUS
cdd  CDD-PROJECT               :FD-PROJECT
cdd  CDD-YR-OU-COST            :FD-YR-OU-COST

var  bool                      :BOOL
var  compteur                  :[1,4] entier
var  lecture                   :BOOL-LECT
var  orgunit                   :FD-ORGANIC-UNIT
var  pro                       :FD-PROJECT
var  status                    :FD-PJ-OU-STATUS
var  yroucost                  :FD-YR-OU-COST

```

SCHEMA DE PROGRAMME CREATION

/* ENTREE DU CODE-PROJET */

```

bool := faux
    Tant que bool = faux faire
        Afficher ('Entrez le code du Projet :')
        Repondre (pro.fd-pj-code)
        Afficher ('Entrez la date de debut de validite du Projet :')
        ENTRER DATE (pro.fd-valid-period-start-date)

```

```

CDD-PJ-2 := vide /* CDD-PJ-2 contiendra le Projet qui a
                    ce code et cette date de debut de
                    valide, s'il existe. */
CDD-PJ-2 := CDD-PROJECT filtre par (FD-PJ-CODE =
    pro.fd-pj-code et FD-VALID-PERIOD-START-DATE =
    pro.fd-valid-period-start-date)
si CDD-PJ-2 = vide alors
    Afficher ('Il n'existe aucun Projet ayant ce
                code et cette date de debut de
                valide.Veuillez recommencer.')
    sinon bool := vrai
        pro := prendre dans CDD-PJ-2
        status.fd-pj-code := pro.fd-pj-code
    finsi

```

Refaire

/* VERIFICATION : LE PROJET NE DOIT PAS ETRE CLOS */

```

si pro.fd-valid-end-date <= date actuelle alors
    Afficher ('Le Projet est clos.Il n'est plus possible
                de lui ajouter un Project-status.')
    finschema

```

sinon

/* ENTREE DU CODE DE L'UNITE ORGANIQUE */

bool := faux

Tant que bool = faux faire

```

    Afficher ('Entrez le code organique :')
    Afficher ('1! Entrez le code de la Division :')
    Repondre (orgunit.fd-division-code)
    Afficher ('2! Entrez le code de la Section :')
    Repondre (orgunit.fd-section-code)
    Afficher ('Entrez la date de debut de valide de
                1'Unite Organique :')

```

ENTRER DATE (orgunit.fd-valid-period-start-date)

```

CDD-ORGUNIT-2 := vide /* CDD-ORGUNIT-2 contiendra 1'Unite
                        organique ayant ce code et cette
                        date de debut de valide, si elle
                        existe. */

```

```

CDD-ORGUNIT-2 := CDD-ORGANIC-UNIT filtre par
    (FD-ORGANIC-CODE = orgunit.fd-organic-code
    et FD-VALID-PERIOD-START-DATE =
    orgunit.fd-valid-period-start-date)

```

```

si CDD-ORGUNIT-2 = vide alors
    Afficher ('Il n'existe aucune Unite Organique
                ayant ce code et cette date de
                debut de valide.Veuillez
                recommencer.')

```

```

    sinon bool := vrai
        orgunit := prendre dans CDD-ORGUNIT-2
        status.fd-organic-code
                    := orgunit.fd-organic-code

```

finsi

Refaire

/* VERIFICATION : L'UNITE ORGANIQUE NE DOIT PAS ETRE CLOSE */

```

si orgunit.fd-valid-period-end-date <= date actuelle alors
    Afficher ('L'Unite Organique concerne est close.Il n'est

```

plus possible de lui associer un FD-PJ-OU-STATUS')

finschema

sinon

/* VERIFICATION : LE PROJET N'EST PAS DEJA */
/* ASSOCIE A CETTE UNITE ORGANIQUE */

Afficher ('Entrez le type de depenses :')
Repondre (status.fd-project-expense-type)
status := reche dans CDD-PJ-OU-STATUS
avec cle = status[fd-organic-code,fd-pj-code,
fd-project-expense-type]

si status < Nil alors
Afficher ('Ce FD-PJ-OU-STATUS est deja associe
au Projet';pro.fd-pj-code;'.Il est
donc impossible de le creer.')

finschema

finsi

/* CREATION DE FD-PJ-OU-STATUS */

Afficher ('Entrez la date de debut de validite :')
ENTRER DATE (status.fd-valid-period-start-date)
si status.fd-valid-period-start-date <
pro.fd-valid-period-start-date alors
Afficher ('La date de debut de validite du
PROJECT-OU-STATUS est anterieure a la date
de debut de validite du Projet.C'est
impossible.')

finschema

sinon si status.fd-valid-period-start-date <
orgunit.fd-valid-period-start-date alors
Afficher ('La date de debut de validite du
PROJECT-OU-STATUS est anterieure a
la date de debut de validite de
l'Unite Organique.C'est impossible.')

finschema

finsi

finsi

status.fd-valid-period-end-date := Nil
status est element de CDD-PJ-OU-STATUS

LIER status avec pro

/* ENTREE DE FD-YR-OU-COST */

yroucost.fd-last-updated := date actuelle /* CC,YY,MM */
compteur := 1

Tant que compteur <= 4 faire
yroucost.fd-ou-matl-cost[compteur] := 0
compteur := compteur + 1

Refaire

yroucost est element de CDD-YR-OU-COST
LIER yroucost avec status

finsi

finsi

finschema

SCHEMA DE PROGRAMME DETRUISANT LOGIQUEMENT UN PROJET
ET METTANT A JOUR LES DRAPEAUX DE FD-PROJECT-STATE
ET DE FD-PJ-OU-STATUS

FREQUENCE ESTIMEE :

DEGRE D'IMPORTANCE :

DONNEES : --

RESULTATS : --

REMARQUES : Ce schema de programme detruit logiquement un Projet ainsi que tous les FD-PROJECT-STATE et tous les FD-PJ-OU-STATUS qui ne sont pas deja detruits logiquement.

Auparavant il verifie :

I. QUE DE TOUTES FACONS :

1. Le Projet existe.
2. Sa date de fin de validite soit posterieure ou egale a la date actuelle.
3. Sa date de fin de validite soit posterieure a sa date de debut de validite.
4. Sa date de fin de validite soit posterieure a la date de debut de validite de chaque :

1. FD-PROJECT-STATE

2. FD-PJ-OU-STATUS

qui lui est associe.

5. Les Taches qui sont associees a ce Projet soient closes a la date de fin de validite du Projet.

II. QUE S'IL EXISTE UN PROJET DE MEME CODE :

1. S'il existe un Projet de meme code et de date de debut de validite posterieure, celui que l'on veut creer finisse avant cette date.

dom CODE-ORG : article
FD-DIVISION-CODE :CODE-DIV
FD-SECTION-CODE :CODE-SEC
fin

dom DATE : article
CC :[0,99] entier
YY :[0,99] entier
MM :[0,99] entier
DD :[0,99] entier
fin

dom PBSKEY : article
FD-ORGANIC-CODE :CODE-ORG
FD-FA-CODE :CODE
FD-TASK-CODE :CODE
fin

dom BOOL :[faux,vrai]
dom BUDGET :[-999'999'999, 999'999'999] entier
dom CAR-UN :caractere[1]
dom CHEF :caracteres[30]
dom CODE :caracteres[4]
dom CODE-DIV :[00,99] entier
dom CODE-SEC :[000,999] entier
dom DEPENSES :[-999'999'999.99,999'999'999.99] reel
dom NOM :caracteres[60]
dom TYPE-DE-DEPENSE :[W,T,L,M,P,R,E,S]

dom TYPE-DE-STATUS :[A,E,S]

cst FD-AMOUNT-STATUS :TYPE-DE-STATUS
cst FD-OU-PBSKEY :PBSKEY
cst FD-PERIOD-COMPL-DATE :DATE
cst FD-PERIOD-START-DATE :DATE
cst FD-PJ-CODE :CODE
cst FD-PJ-LEADER :CHEF
cst FD-PJ-LEADER-DIV :caracteres[3]
cst FD-PJ-NAME :NOM
cst FD-PJ-REVISED :CAR-UN
cst FD-PROJECT-EXPENSE-TYPE :TYPE-DE-DEPENSE
cst FD-TASK-CURRENT-BUDGET :BUDGET
cst FD-TASK-NAME :NOM
cst FD-VALID-PERIOD-END-DATE :DATE et Nil
cst FD-VALID-PERIOD-START-DATE :DATE

rel FD-PROJECT (FD-PJ-CODE,FD-PJ-REVISED,FD-PJ-NAME,FD-PJ-LEADER,
FD-PJ-LEADER-DIV,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE,FD-PERIOD-START-DATE,
FD-PERIOD-COMPL-DATE)
ayant cle FD-PJ-CODE

rel FD-PROJECT-STATE (FD-PJ-CODE,FD-AMOUNT-STATUS,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE)
ayant cle FD-PJ-CODE,FD-AMOUNT-STATUS

rel FD-PJ-OU-STATUS (FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE
FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE)
ayant cle FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE

rel FD-YEAR-TASKLIST (FD-ORGANIC-CODE,FD-FA-CODE,FD-TASK-CODE,FD-TASK-NAME,
FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE,
FD-TASK-CURRENT-BUDGET)
ayant cle FD-OU-PBSKEY

cdd CDD-PJ-1 :FD-PROJECT
cdd CDD-PJ-2 :FD-PROJECT
cdd CDD-PJ-OU-STATUS :FD-PJ-OU-STATUS
cdd CDD-PJ-STATE :FD-PROJECT-STATE
cdd CDD-PROJECT :FD-PROJECT
cdd CDD-PROJECT-1 :FD-PROJECT
cdd CDD-YEAR-TASKLIST :FD-YEAR-TASKLIST

var bool :BOOL
var debute-apres :BOOL
var parcours-state :FD-PROJECT-STATE
var parcours-status :FD-PJ-OU-STATUS
var pro :FD-PROJECT
var pro-ex :FD-PROJECT
var reponse :caractere[1]

SCHEMA DE PROGRAMME DETRUISANT LOGIQUEMENT UN PROJET

/* ENTREE DU CODE DU PROJET */

Afficher ('Entrez le code du Projet :')

Repondre (pro.fd-pj-code)

CDD-PROJECT-1 := vide /* CDD-PROJECT-1 contiendra tous les Projets
ayant le meme code que celui que l'on veut

```

destruire logiquement. */
CDD-PROJECT-1 := CDD-PROJECT filtre par (FD-PJ-CODE = pro.fd-pj-code)
Afficher ('Entrez la date de debut de validite du Projet :')
Repondre (pro.fd-valid-period-start-date)
CDD-PJ-1 := vide /* CDD-PJ-1 contiendra le Projet ayant ce code et
cette date de debut de validite, s'il existe. */
CDD-PJ-1 := CDD-PROJECT-1 filtre par ( FD-VALID-PERIOD-START-DATE =
pro.fd-valid-period-start-date)
si CDD-PJ-1 = vide alors
Afficher ('Il n'existe aucun Projet ayant ce code qui commence
a cette date.Destruction logique impossible')
finschema
sinon pro := prendre dans CDD-PJ-1
finsi

si pro.fd-valid-period-end-date <> Nil alors
Afficher ('Ce Projet a deja une date de fin de validite.
C'est le :')
AFFICHER DATE (pro.fd-valid-period-end-date)
reponse := ' '
Tant que reponse <> 'O' et reponse <> 'N' faire
Afficher ('Voulez-vous la changer ? (O/N)')
Repondre (reponse)
Refaire
si reponse = 'N' alors
finschema
finsi
finsi

Afficher ('Entrez la date de fin de validite du Projet :')
ENTRER DATE (pro.fd-valid-period-end-date)

/* VERIFICATION : LA DATE DE FIN DE VALIDITE DOIT ETRE POSTERIEURE */
/* OU EGALE A LA DATE ACTUELLE */

si pro.fd-valid-period-end-date < date actuelle alors
Afficher ('Cette date de fin de validite est impossible
car elle est anterieure a la date actuelle.
La destruction aurait un effet retroactif.
C'est impossible.')
finschema
finsi

/* VERIFICATION QUE LA DATE DE FIN DE VALIDITE SOIT POSTERIEURE */
/* A LA DATE DE DEBUT DE VALIDITE */

si pro.fd-valid-period-end-date <= pro.fd-valid-period-start-date
alors
Afficher ('La date de fin de validite du Projet est anterieure
a sa date de debut de validite.C'est impossible.)
finschema
finsi

CDD-PJ-2 := vide /* CDD-PJ-2 contiendra les Projets de meme code dont
la date de debut de validite est posterieure
a celle que l'on veut destruire. */
CDD-PJ-2 := CDD-PROJECT-1 filtre par (FD-VALID-PERIOD-START-DATE >
pro.fd-valid-period-start-date)
trie croissant par
(FD-VALID-PERIOD-START-DATE)

```



```

si CDD-PJ-2 <> vide alors
    pro-ex := prendre dans CDD-PJ-2

/* VERIFICATION : S'IL EXISTE UN PROJET DE MEME CODE ET */
/* DE DATE DE DEBUT DE VALIDITE POSTERIEURE, CELUI QUE L'ON */
/* VEUT DETRUIRE FINISSE AVANT CETTE DATE */

    si pro-ex.fd-valid-period-start-date <=
        pro.fd-valid-period-end-date alors
            Afficher ('La date de fin de validite du Projet est
                posterieure ou egale a la date de debut
                validee d'un Projet de meme code qui existe
                deja.C'est impossible.')
            finschema
        finsi
    finsi

/* VERIFICATION QUE LA DATE DE FIN DE VALIDITE DU PROJET */
/* SOIT POSTERIEURE A LA DATE DE DEBUT DE VALIDITE DE CHAQUE */
/*
    1. FD-PROJECT-STATE */
/*
    2. FD-PJ-OU-STATUS */

debut-apres := faux
parcours-state.fd-pj-code := pro.fd-pj-code
parcours-state := premier
    Tant que parcours-state <> dernier ou parcours-state = dernier
        et debut-apres = faux faire
        parcours-state := prendre dans CDD-PJ-STATE
        si parcours-state.fd-valid-period-start-date >=
            pro.fd-valid-period-end-date alors
                debut-apres := vrai
            finsi
        parcours-state := suivant
    Refaire
    si debut-apres = vrai alors
        Afficher ('Il existe au-moins un FD-PROJECT-STATE associe
            a ce Projet dont la date de debut de validite est
            posterieure ou egale a la date de fin de validite
            du Projet.C'est impossible.')
        finschema
    finsi
parcours-status.fd-pj-code := pro.fd-pj-code
parcours-status := premier
    Tant que parcours-status <> dernier ou
        parcours-status = dernier et debut-apres = faux faire
        parcours-status := prendre dans CDD-PJ-OU-STATUS
        si parcours-status.fd-valid-period-start-date >=
            pro.fd-valid-period-end-date alors
                debut-apres = vrai
            finsi
        parcours-status := suivant
    Refaire
    si debut-apres = vrai alors
        Afficher ('Il existe au-moins un FD-PJ-OU-STATUS associe a
            ce Projet dont la date de debut de validite est
            posterieure ou egale a la date de fin de validite
            du Projet.C'est impossible.')
        finschema
    finsi

```


/* VERIFICATION QUE TOUTES LES TACHES ASSOCIEES A CE PROJET */
 /* SONT CLOSES A LA DATE DE FIN DE VALIDITE DU PROJET */

CDD-TASK-PAS-CLOSES := vide /* CDD-TASK-CLOSES contiendra toutes les Taches
 ayant ce code de Projet comme task-code et
 qui finissent apres la date de fin de validite
 du Projet. */

CDD-TASK-PAS-CLOSES := CDD-YEAR-TASKLIST filtre par
 (FD-TASK-CODE = pro.fd-pj-code et
 FD-VALID-PERIOD-END-DATE >
 pro.fd-valid-period-end-date ou
 FD-VALID-PERIOD-END-DATE = Nil)

si CDD-TASK-PAS-CLOSES <> vide alors
Afficher ('Il existe au-moins une Tache associee a ce Projet
 (par l'intermediaire du FD-PJ-OU-STATUS) qui se
 termine apres la date de fin de validite du Projet.
 C'est impossible. Pour detruire logiquement un
 Projet, il faut que toutes les Taches associees
 soient closes.')

finschema
finsi

/* DESTRUCTION LOGIQUE POSSIBLE */

/* DESTRUCTION LOGIQUE DES FD-PROJECT-STATE DONT LA DATE DE FIN */
 /* DE VALIDITE EST EGALE A Nil OU DONT LA DATE DE FIN DE VALIDITE */
 /* EST POSTERIEURE A CELLE DU PROJET */

parcours-state.fd-pj-code := pro.fd-pj-code

parcours-state := premier

Tant que parcours-state <> dernier ou parcours-state = dernier faire

parcours-state := prendre dans CDD-PJ-STATE

si parcours-state.fd-valid-period-end-date = Nil ou
 parcours-state.fd-valid-period-end-date >

pro.fd-valid-period-end-date alors

parcours-state.fd-valid-period-end-date =
 pro.fd-valid-period-end-date

MISE A JOUR de parcours-state dans CDD-PJ-STATE

finsi

parcours-state := suivant

Refaire

/* DESTRUCTION LOGIQUE DES FD-PJ-OU-STATUS DONT LA DATE DE FIN */
 /* DE VALIDITE EST EGALE A Nil OU DONT LA DATE DE FIN DE VALIDITE */
 /* EST POSTERIEURE A CELLE DU PROJET */

parcours-status.fd-pj-code := pro.fd-pj-code

parcours-status := premier

Tant que parcours-status <> dernier ou
 parcours-status = dernier faire

parcours-status := prendre dans CDD-PJ-OU-STATUS

si parcours-status.fd-valid-period-end-date = Nil ou
 parcours-status.fd-valid-period-end-date >

pro.fd-valid-period-end-date alors

parcours-status.fd-valid-period-end-date =
 pro.fd-valid-period-end-date

MISE A JOUR de parcours-status dans CDD-PJ-OU-STATUS

finsi

parcours-status := suivant

Refaire

/* DESTRUCTION LOGIQUE DU PROJET */

MISE A JOUR de pro dans CDD-PROJECT

finschema

SCHEMA DE PROGRAMME DETRUISANT PHYSIQUEMENT FD-PROJECT

<u>TOUS LES</u>	<u>FD-PROJECT-STATE</u>	
	<u>FD-YR-PJ-INFO</u>	
	<u>FD-PJ-OU-STATUS</u>	
	<u>FD-YR-OU-COST</u>	
	<u>FD-FA-PJ</u>	
	<u>FD-YEAR-TASKLIST</u>	<u>ASSOCIES</u>
<u>ET METANT A JOUR</u>	<u>FD-OOAK-PJ</u>	

FREQUENCE ESTIMEE :

DEGRE D'IMPORTANCE :

DONNEES : ---

RESULTATS : ---

REMARQUES : Ce schema de programme detruit physiquement un Projet et tous les FD-PROJECT-STATE,FD-YR-PJ-INFO,FD-PJ-OU-STATUS, FD-YR-OU-COST,FD-YEAR-TASKLIST qui lui sont associes.

Auparavant il verifie que :

1. Ce Projet existe et que sa date de fin de validite soit anterieure a la date actuelle.
2. Tous les FD-PROJECT-STATE associes a ce Projet sont clos.
3. Tous les FD-PJ-OU-STATUS associes a ce Projet sont clos.
4. Toutes les Taches associees a ce Projet sont closes.

dom CODE-ORG : article

FD-DIVISION-CODE :CODE-DIV

FD-SECTION-CODE :CODE-SEC

fin

dom DATE : article

CC :[0,99] entier

YY :[0,99] entier

MM :[0,99] entier

DD :[0,99] entier

fin

dom DATE-MOIS : article

CC :[0,99] entier

YY :[0,99] entier

MM :[0,99] entier

fin

dom PBSKEY : article

FD-ORGANIC-CODE :CODE-ORG

FD-FA-CODE :CODE

FD-TASK-CODE :CODE

fin

dom BOOL :[faux,vrai]

dom CAR-UN :caractere[1]

dom CHEF :caracteres[30]

dom CODE :caracteres[4]

dom CODE-DIV :[00,99] entier

dom CODE-SEC :[000,999] entier

dom DEPENSES :[-999'999'999.99,999'999'999.99] reel

dom NOM :caracteres[30]

```

dom  TYPE-OF-EXPENSE   :[ W,T,L,M,P,R,E,S ]
dom  TYPE-OF-STATUS    :[ A,S,E ]

cst  FD-FAPJ-DUMMY      :caracteres[2]
cst  FD-LAST-UPDATED    :DATE-MOIS
cst  FD-OU-MATL-COST     : tableau [4] de DEPENSES
cst  FD-OU-PBS-KEY       :PBSKEY
cst  FD-PERIOD-COMPL-DATE :DATE
cst  FD-PERIOD-START-DATE :DATE
cst  FD-PJ-CODE          :CODE
cst  FD-PJ-HEADER        :caracteres[3]
cst  FD-PJ-LAST-MANMTH-SMC-REV-DATE :DATE
cst  FD-PJ-LAST-MANMTH-TL-REV-DATE  :DATE
cst  FD-PJ-LAST-MANYR-REV-DATE       :DATE
cst  FD-PJ-LAST-MATL-COST-REV-DATE    :DATE
cst  FD-PJ-LEADER                     :CHEF
cst  FD-PJ-LEADER-DIV                  :caracteres[3]
cst  FD-PJ-MANMTH-SMC                   : tableau [10] de DEPENSES
cst  FD-PJ-MANMTH-TL                     : tableau [10] de DEPENSES
cst  FD-PJ-MANYR                         : tableau [10] de DEPENSES
cst  FD-PJ-MATL-COST                     : tableau [10] de DEPENSES
cst  FD-PJ-NAME                          :NOM
cst  FD-PJ-REVISED                      :CAR-UN
cst  FD-PROJECT-EXPENSE-TYPE              :TYPE-OF-EXPENSE
cst  FD-TASK-CURRENT-BUDGET               :BUDGET
cst  FD-TASK-NAME                        :NOM
cst  FD-VALID-PERIOD-END-DATE             :DATE et Nil
cst  FD-VALID-PERIOD-START-DATE           :DATE

rel  FD-OOAK-PJ (FD-PJ-HEADER)
      ayant cle FD-PJ-HEADER

rel  FD-PROJECT (FD-PJ-CODE,FD-PJ-REVISED,FD-PJ-NAME,FD-PJ-LEADER,
      FD-PJ-LEADER-DIV,FD-VALID-PERIOD-START-DATE,
      FD-VALID-PERIOD-END-DATE,FD-PERIOD-START-DATE,
      FD-PERIOD-COMPL-DATE)
      ayant cle FD-PJ-CODE

rel  FD-PROJECT-STATE (FD-PJ-CODE,FD-AMOUNT-STATUS,FD-VALID-PERIOD-START-DATE,
      FD-VALID-PERIOD-END-DATE)
      ayant cle FD-PJ-CODE,FD-TYPE

rel  FD-YR-PJ-INFO (FD-PJ-CODE,FD-TYPE,FD-AMOUNT-STATUS,FD-LAST-UPDATED,
      FD-PJ-LAST-MATL-COST-REV-DATE,
      FD-PJ-MATL-COST,FD-PJ-LAST-MANYR-REV-DATE,
      FD-PJ-MANYR,FD-PJ-LAST-MANMTH-TL-REV-DATE,
      FD-PJ-MANMTH-TL,FD-PJ-LAST-MANMTH-SMC-REV-DATE,
      FD-PJ-MANMTH-SMC)
      ayant cle FD-PJ-CODE,FD-TYPE,FD-AMOUNT-STATUS,FD-LAST-UPDATED

rel  FD-PJ-OU-STATUS (FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE,
      FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE)
      ayant cle FD-PJ-CODE,FD-ORGANIC-CODE

rel  FD-YR-OU-COST (FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE,
      FD-LAST-UPDATED,FD-OU-MATL-COST)
      ayant cle FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE,
      FD-LAST-UPDATED

rel  FD-FA-PJ (FD-FA-CODE,FD-PJ-CODE,FD-FAPJ-DUMMY)

```

ayant cle FD-FA-CODE,FD-PJ-CODE

rel FD-YEAR-TASKLIST (FD-OU-PBSKEY,FD-TASK-NAME,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE,FD-TASK-CURRENT-BUDGET)

ayant cle FD-OU-PBSKEY

cdd CDD-FA-PJ :FD-FA-PJ
cdd CDD-OOAK-PJ :FD-OOAK-PJ
cdd CDD-PJ-1 :FD-PROJECT
cdd CDD-PJ-OU-STATUS :FD-PJ-OU-STATUS
cdd CDD-PJ-STATE :FD-PROJECT-STATE
cdd CDD-PROJECT :FD-PROJECT
cdd CDD-TASK-A-NIL :FD-YEAR-TASKLIST
cdd CDD-YEAR-TASKLIST :FD-YEAR-TASKLIST
cdd CDD-YR-INFO :FD-YR-PJ-INFO
cdd CDD-YR-OU-COST :FD-YR-OU-COST

var compteur :[1,10] entier
var oak :FD-OOAK-PJ
var parcours-fapj :FD-FA-PJ
var parcours-info :FD-YR-PJ-INFO
var parcours-oucoss :FD-YR-OU-COST
var parcours-state :FD-PROJECT-STATE
var parcours-status :FD-PJ-OU-STATUS
var parcours-tasklist :FD-YEAR-TASKLIST
var pas-clos :BOOL
var pro :FD-PROJECT

SCHEMA DE PROGRAMME DETRUISANT PHYSIQUEMENT UN PROJET

/* ENTREE DU CODE DU PROJET */

Afficher ('Entrez le code du Projet :')

Repondre (pro.fd-pj-code)

Afficher ('Entrez la date de debut de validite du Projet :')

ENTRER DATE (pro.fd-valid-period-start-date)

CDD-PJ-1 := vide /* CDD-PJ-1 contiendra le Projet ayant ce code et cette
date de debut de validite s'il existe. */

CDD-PJ-1 := CDD-PROJECT filtre par (FD-PJ-CODE = pro.fd-pj-code et
FD-VALID-PERIOD-START-DATE =
pro.fd-valid-period-start-date)

si CDD-PJ-1 = vide alors

Afficher ('Il n'existe aucun Projet ayant ce code qui commence
a cette date.Destruction physique impossible.')

finschema

finsi

/* VERIFICATION : CE PROJET DOIT DEJA ETRE */

/* LOGIQUEMENT DETRUIT */

si pro.fd-valid-period-end-date := Nil alors

Afficher ('Ce projet n'a pas de date de fin de
validite.Il n'a pas ete logiquement detruit.

Il est donc impossible de le detruire physiquement.')

finschema

finsi

si pro.fd-valid-period-end-date >= date actuelle alors

Afficher ('Ce projet n'est pas encore arrive a sa
date de fin de validite.Il est donc impossible

de le detruire physiquement.')

finschema
finsi

/* VERIFICATION QUE TOUS LES FD-PROJECT-STATE */
/* ASSOCIES A CE PROJET SONT CLOS */

pas-clos := faux

parcours-state.fd-pj-code := pro.fd-pj-code

parcours-state := premier

Tant que parcours-state <> dernier ou parcours-state = dernier
et pas-clos = faux faire
parcours-state := prendre dans CDD-PROJECT-STATE
si parcours-state.fd-valid-period-end-date = Nil
alors

pas-clos := vrai

finsi

parcours-state := suiuant

Refaire

si pas-clos = vrai alors

Afficher ('Ce Projet n'a pas ete logiquement detruit.Il reste
des FD-PROJECT-STATE qui lui sont associes et
dont la date de fin de validite est egale a Nil .
Destruction physique impossible.')

finschema
finsi

/* VERIFICATION QUE TOUS LES FD-PJ-OU-STATUS */
/* ASSOCIES A CE PROJET SONT CLOS */

parcours-status.fd-pj-code := pro.fd-pj-code

parcours-status := premier

Tant que parcours-status <> dernier ou
parcours-status = dernier et pas-clos = faux faire
parcours-status := prendre dans CDD-PJ-OU-STATUS
si parcours-status.fd-valid-period-end-date = Nil
alors

pas-clos := vrai

finsi

parcours-status := suiuant

Refaire

si pas-clos = vrai alors

Afficher ('Ce Projet n'a pas ete logiquement detruit.Il reste
des FD-PJ-OU-STATUS dont la date de fin de validite
est egale a Nil .Destruction physique impossible.')

finschema
finsi

/* VERIFICATION : TOUTES LES TACHES ASSOCIEES A CE PROJET */
/* DOIVENT ETRE CLOSES */

CDD-TASK-A-NIL := vide /* CDD-TASK-A-NIL contiendra toutes les Taches
associees au Projet (fd-task-code = fd-pj-code)
dont la date de fin de validite est egale a Nil)
CDD-TASK-A-NIL := CDD-YEAR-TASKLIST filtrer par (FD-TASK-CODE = pro.fd-pj-code
et FD-VALID-PERIOD-END-DATE = Nil)

si CDD-TASK-A-NIL <> vide alors

Afficher ('Ce Projet n'a pas ete detruit logiquement.Il reste
des Taches associees a ce Projet dont la date de
fin de validite est egale a Nil .Destruction

physique impossible.')

finschema
finsi

/* DESTRUCTION PHYSIQUE PERMISE */

/* RECHERCHE DES PROJECT-STATE ET DES FD-YR-PJ-INFO */
/* ASSOCIES ET DESTRUCTION PHYSIQUE */

```
parcours-state.fd-pj-code := pro.fd-pj-code
parcours-state := premier
  Tant que parcours-state <> dernier ou parcours-state = dernier faire
    parcours-state := prendre dans CDD-PJ-STATE
    parcours-info.fd-pj-code := pro.fd-pj-code
    parcours-info := premier
      Tant que parcours-info <> dernier ou
        parcours-info = dernier faire
        parcours-info := prendre dans CDD-YR-INFO
        SUPPRIMER parcours-info
        parcours-info := suivant
      Refaire
    SUPPRIMER parcours-state
    parcours-state := suivant
  Refaire
```

/* RECHERCHE DES PJ-OU-STATUS ET DES FD-YR-OU-COST */
/* ET DES FD-YEAR-TASKLIST ASSOCIES ET DESTRUCTION PHYSIQUE */

```
parcours-status.fd-pj-code := pro.fd-pj-code
parcours-status := premier
  Tant que parcours-status <> dernier ou
    parcours-status = dernier faire
    parcours-status := prendre dans CDD-PJ-OU-STATUS
    parcours-oucost.fd-pj-code := pro.fd-pj-code
    Tant que parcours-oucost <> dernier ou
      parcours-oucost = dernier faire
      parcours-oucost := prendre dans CDD-YR-OU-COST
      SUPPRIMER parcours-oucost
      parcours-oucost := suivant
    Refaire
    parcours-tasklist.fd-task-code := pro.fd-pj-code
    parcours-tasklist := premier
      Tant que parcours-tasklist <> dernier ou
        parcours-tasklist = dernier faire
        parcours-tasklist := prendre dans CDD-YEAR-TASKLIST
        DETUIRE TACHE (parcours-tasklist)
        parcours-tasklist := suivant
      Refaire
    SUPPRIMER parcours-status
  parcours-status := suivant
  Refaire
```

/* RECHERCHE DES FA-PJ ASSOCIES ET DESTRUCTION */

```
parcours-fapj.fd-pj-code := pro.fd-pj-code
parcours-fapj := premier
  Tant que parcours-fapj <> dernier ou parcours-fapj = dernier
```


faire

parcours-fapj := prendre dans CDD-FA-PJ
SUPPRIMER parcours-fapj
parcours-fapj := suivant

Refaire

/* MISE A JOUR DE LA TETE DE LISTE */

DELIER pro avec oak
 si CDD-OOAK-PJ = vide alors
 SUPPRIMER oak
 sinon MISE A JOUR de oak
 finsi
SUPPRIMER pro

finschema

SCHEMA DE PROGRAMME CREANT UNE ACTIVITE DIVISIONNAIRE
ET LE PREMIER DA-MTH ASSOCIE

FREQUENCE ESTIMEE :
DEGRE D'IMPORTANCE :

DONNEES : ---

RESULTATS : ---

REMARQUES : Ce schema de programme cree une Activite Divisionnaire.

Auparavant il verifie :

I. QUE DE TOUTE FACON :

1. l'Activite Fonctionnelle a laquelle est associee l'Activite Divisionnaire existe deja dans la base et que celle-ci n'est pas close (les doubles ne sont pas autorises).
2. la Division a laquelle est associee l'Activite Divisionnaire existe deja dans la base et qu'elle n'est pas close (les doubles ne sont pas autorises).
3. le budget de l'Activite Fonctionnelle est superieur ou egal a la somme des budgets des Activites Divisionnaires associees a cette Activite Fonctionnelle.
4. la date de debut de validite de l'Activite Divisionnaire est egale ou posterieure a celle :
 1. de l'Activite Fonctionnelle
 2. de la Division
5. la date de fin de validite de l'Activite Divisionnaire est egale ou anterieure a celle :
 1. de l'Activite Fonctionnelle
 2. de la Division
6. la date de fin de validite de l'Activite Divisionnaire est posterieure a sa date de debut de validite.

II. QUE S'IL EXISTE UNE ACTIVITE DIVISIONNAIRE AYANT MEME CODE

1. Celle que l'on veut creer n'ait pas la meme date de debut de validite que celle qui existe deja.
2. Celle que l'on veut creer ne commence pas durant la periode de validite de celle qui existe deja.

Remarque : si la date de fin de validite de l'A.D.

qui existe est egale a Nil alors l'utilisateur ne pourra creer une nouvelle A.D. de meme code. Il devra tout d'abord aller preciser la date de fin de validite de l'A.D. existante. Ensuite il pourra creer un "duplicate".

3. S'il existe une Activite Divisionnaire de meme code et de date de debut de validite posterieure, celle que l'on veut creer finisse avant cette date.

dom BOOL : [faux, vrai]

dom DATE : article
CC : [0,99] entier
YY : [0,99] entier
MM : [0,99] entier
DD : [0,99] entier
fin

dom DATE-MOIS : article
CC : [0,99] entier
YY : [0,99] entier

MM:[0,99] entier

fin

dom BUDGET :[-999'999'999,999'999'999] entier
dom CAR-TROIS :caracteres[3]
dom CHEF :caracteres[30]
dom CODE :caracteres[4]
dom DEPENSES :[-999'999'999.99, 999'999'999.99] reel
dom NOM :caracteres[60]

cst FD-DA-CREATION-MTH :DATE-MOIS
cst FD-DA-CUMUL-EXP :DEPENSES
cst FD-DA-CUMUL-OPEN-COMMIT :DEPENSES
cst FD-DA-CUMUL-TID :DEPENSES
cst FD-DA-CURRENT-BUDGET :BUDGET
cst FD-DA-LEADER :CHEF
cst FD-DA-MTH-COMMIT :DEPENSES
cst FD-DA-MTH-EXP :DEPENSES
cst FD-DA-NAME :NOM
cst FD-DIVISION-ID :CAR-TROIS
cst FD-DIVISION-LEADER :CHEF
cst FD-FA-CODE :CODE
cst FD-FA-CUMUL-TID :DEPENSES
cst FD-FA-CUMUL-OPEN-COMMIT :DEPENSES
cst FD-FA-CURRENT-BUDGET :BUDGET
cst FD-FA-MTH-COMMIT :DEPENSES
cst FD-FA-MTH-EXP :DEPENSES
cst FD-FA-NAME :NOM
cst FD-FA-YEAR-EXP :DEPENSES
cst FD-LAST-UPDATED :DATE-MOIS
cst FD-VALID-PERIOD-END-DATE :DATE et Nil
cst FD-VALID-PERIOD-START-DATE :DATE

rel FD-FCTNAL-ACTIV (FD-FA-CODE,FD-FA-NAME,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE)
ayant cle FD-FA-CODE

rel FD-FA-MTH (FD-FA-CODE,FD-LAST-UPDATED,
FD-FA-CURRENT-BUDGET,FD-FA-YEAR-EXP,FD-FA-MTH-EXP,
FD-FA-CUMUL-TID,FD-FA-CUMUL-OPEN-COMMIT,FD-FA-MTH-COMMIT)
ayant cle FD-FA-CODE,FD-LAST-UPDATED

rel FD-DIVISION (FD-DIVISION-ID,FD-DIVISION-LEADER,
FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE)
ayant cle FD-DIVISION-ID

rel FD-DIV-ACTIV (FD-FA-CODE,FD-DIVISION-ID,FD-DA-NAME,FD-DA-LEADER,
FD-DA-CREATION-MTH,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE)
ayant cle FD-FA-CODE,FD-DIVISION-ID

rel FD-DA-MTH (FD-FA-CODE,FD-DIVISION-ID,FD-LAST-UPDATED,
FD-DA-CURRENT-BUDGET,FD-DA-CUMUL-EXP,FD-DA-MTH-EXP,
FD-DA-CUMUL-TID,FD-DA-CUMUL-OPEN-COMMIT,FD-DA-MTH-COMMIT)
ayant cle FD-FA-CODE,FD-DIVISION-ID,FD-LAST-UPDATED

cdd CDD-DA-2 :FD-DIV-ACTIV
cdd CDD-DA-3 :FD-DIV-ACTIV
cdd CDD-DA-4 :FD-DIV-ACTIV
cdd CDD-DA-MTH :FD-DA-MTH

```

cdd CDD-DIV-ACT-1           :FD-DIV-ACTIV
cdd CDD-DIV-ACTIV          :FD-DIV-ACTIV
cdd CDD-DIVISION           :FD-DIVISION
cdd CDD-FA-MTH             :FD-FA-MTH
cdd CDD-FCTNAL-ACTIV       :FD-FCTNAL-ACTIV

var bool                   :BOOL
var budget-tot-fa          :BUDGET
var da                     :FD-DIV-ACTIV et Nil
var da-ex                  :FD-DIV-ACTIV
var damth                  :FD-DA-MTH
var div                    :FD-DIVISION et Nil
var existant               :BOOL
var existe-post            :BOOL
var famth                  :FD-FA-MTH
var fctnal                 :FD-FCTNAL-ACTIV et Nil
var parcours-da           :FD-DIV-ACTIV
var parcours-damth         :FD-DA-MTH
var parcours-famth         :FD-FA-MTH
var reponse                :caractere[1]

```

SCHEMA DE PROGRAMME CREANT UNE ACTIVITE DIVISIONNAIRE

/* RECHERCHE DE L'ACTIVITE FONCTIONNELLE */

```

existant := faux
existe-post := faux
bool := faux
  Tant que bool = faux faire
    Afficher ('Entrez le code de l'Activite Fonctionnelle :')
    Repondre (fctnal.fd-fa-code)
    fctnal := prendre dans CDD-FCTNAL-ACTIV
               avec cle = fctnal.fd-fa-code
    si fctnal = Nil alors
      Afficher ('Ce code est invalide. Il ne
                  correspond a aucune Activite
                  Fonctionnelle.')
    sinon bool := vrai
    finsi
  Refaire

/* VERIFICATION : L'ACTIVITE FONCTIONNELLE NE DOIT PAS */
/* ETRE CLOSE */

```

```

si fctnal.fd-valid-period-end-date <= date actuelle alors
  Afficher ('Cette Activite Fonctionnelle est close.
              Il est donc impossible de lui associer une
              Activite Divisionnaire.')
  finschema

```

sinon

/* RECHERCHE DE LA DIVISION */

```

bool := faux
Tant que bool = faux faire
  Afficher ('Entrez le code de la Division :')
  Repondre (div.fd-division-id)
  div := prendre dans CDD-DIVISION
          avec cle = div.fd-division-id
  si div = Nil alors

```

```

        Afficher ('Ce code de Division est invalide.
                    Il ne correspond a aucune
                    Division.')
```

sinon bool := vrai
finsi

Refaire

/* VERIFICATION : LA DIVISION NE DOIT PAS ETRE CLOSE */

si div.fd-valid-period-end-date <= date actuelle alors
 Afficher ('Cette Division est close.
 Il est donc impossible de lui associer une
 Activite Divisionnaire.')

finschema
sinon

/* RECHERCHE DE L'ACTIVITE DIVISIONNAIRE */

da.fd-fa-code := fctnal.fd-fa-code
 da.fd-division-id := div.fd-division-id
 CDD-DIV-ACT-1 := vide /* CDD-DIV-ACT contiendra toutes
 les Activites Divisionnaires
 ayant meme code que celle
 que l'on veut creer */
 CDD-DIV-ACT-1 := CDD-DIV-ACTIV filtre par
 (FD-FA-CODE = da.fd-fa-code,
 FD-DIVISION-ID = da.fd-division-id)
si CDD-DIV-ACT-1 <> vide alors
 existant := vrai
 finsi
 Afficher ('Entrez le nom de l'Activite Divisionnaire :')
 Repondre (da.fd-da-name)
 Afficher ('Entrez le nom du chef de l'Activite
 Divisionnaire :')
 Repondre (da.fd-da-leader)
 Afficher ('Entrez la date de creation de l'Activite
 Divisionnaire :')
 ENTRER DATE-MOIS (da.fd-da-creation-mth)
 Afficher ('Entrez le budget de l'Activite Divisionnaire :')
 Repondre (damth.fd-da-current-budget)

/* VERIFICATION : LA SOMME DES BUDGETS DES ACTIVITES */
 /* DIVISIONNAIRES ASSOCIEES A UNE ACTIVITE FONCTIONNELLE */
 /* DOIT ETRE INFERIEURE OU EGALE AU BUDGET DE L'ACTIVITE */
 /* FONCTIONNELLE */

parcours-famth.fd-fa-code := fctnal.fd-fa-code
 parcours-famth := dernier
 parcours-famth := prendre dans CDD-FA-MTH
 parcours-da.fd-fa-code := fctnal.fd-fa-code
 parcours-damth.fd-fa-code := parcours-da.fd-fa-code
 parcours-da := premier
 budget-tot-fa := 0
Tant que parcours-da <> dernier ou parcours-da = dernier faire
 parcours-da := prendre dans CDD-DIV-ACTIV
 parcours-damth.fd-division-id := parcours-da.fd-division-id
 parcours-damth := dernier
 parcours-damth := prendre dans CDD-DA-MTH
 budget-tot-fa := budget-tot-fa

```

+ parcours-damth.fd-da-current-budget
parcours-da := suivant
Refaire
budget-tot-fa := budget-tot-fa + damth.fd-da-current-budget
si budget-tot-fa > parcours-famth.fd-fa-current-budget
  alors
    Afficher ('Le Budget total des Activites Divisionnaires
               associees a cette Activite Fonctionnelle
               serait superieur au Budget de l'Activite
               Fonctionnelle si on lui rajoutait ce
               Budget.C'est impossible.')
    finschema
  finsi
Afficher ('Entrez la date de debut de validite de
            l'Activite Divisionnaire :')
ENTRER DATE (da.fd-valid-period-start-date)
si da.fd-valid-period-start-date <
  fctnal.fd-valid-period-start-date alors
    Afficher ('La date de debut de validite de
               l'Activite Divisionnaire est anterieure
               a celle de l'Activite Fonctionnelle.
               C'est impossible.')
    finschema
  sinon si da.fd-valid-period-start-date <
    div.fd-valid-period-start-date alors
      Afficher ('La date de debut de validite
                 de l'Activite Divisionnaire est
                 anterieure a celle de la
                 Division.C'est impossible')
      finschema
    finsi
  finsi
si existant = vrai alors

/* VERIFICATION QU'IL N'EXISTE PAS UNE AUTRE A.D. AYANT MEME */
/* DATE DE DEBUT DE VALIDITE */

CDD-DA-2 := vide /* CDD-DA-2 contiendra l'Activite
                  Divisionnaire qui a la meme
                  date de debut de validite, si
                  elle existe. */
CDD-DA-2 := CDD-DIV-ACT-1 filtre par
(FD-VALID-PERIOD-START-DATE =
  da.fd-valid-period-start-date)
si CDD-DA-2 <> vide alors
  Afficher ('Il existe deja une Activite
              Divisionnaire ayant meme code
              et meme date de debut de
              validite.Creation
              impossible.')
  finschema
finsi
/* VERIFICATION QUE L'A.D. QUE L'ON VEUT CREER */
/* NE COMMENCE PAS DURANT LA PERIODE DE VALIDITE */
/* D'UNE AUTRE ACTIVITE DIVISIONNAIRE */

CDD-DA-3 := vide /* CDD-DA-3 contiendra l'A.D.
                  valide pour cette periode,
                  si elle existe. */

```

```

CDD-DA-3 := CDD-DIV-ACT-1 filtre par
              (FD-VALID-PERIOD-START-DATE <
                da.fd-valid-period-start-date
                et FD-VALID-PERIOD-END-DATE >
                da.fd-valid-period-start-date
                ou FD-VALID-PERIOD-END-DATE = Nil )
si CDD-DA-3 <> vide alors
    Afficher ('L'Activite Divisionnaire
              que vous voulez creer
              serait "contenue" dans
              une autre A.D. deja
              existante.C'est impossible.')
    finschema
  finsi
CDD-DA-4 := vide /* CDD-DA-4 contiendra les
                  Activites Divisionnaires dont
                  la date de debut de validite
                  est posterieure a celle de l'A.D.
                  que l'on veut creer. */
CDD-DA-4 := CDD-DIV-ACT-1 filtre par
              (FD-VALID-PERIOD-START-DATE >
                da.fd-valid-period-start-date)
              trie croissant par
                (FD-VALID-PERIOD-START-DATE)
si CDD-DA-4 <> vide alors
    existe-post := vrai
    da-ex := prendre dans CDD-DA-4
    finsi
  finsi
si existe-post = vrai alors
    Afficher ('Entrez la date de fin de validite
              de l'Activite Divisionnaire :')
    ENTRER DATE (da.fd-valid-period-end-date)
    si da.fd-valid-period-end-date <=
        da.fd-valid-period-start-date alors
        Afficher ('La date de fin de validite est
                  anterieure a la date de debut de
                  validite.C'est impossible.')
        finschema
      finsi
    si da-ex.fd-valid-period-start-date <=
        da.fd-valid-period-end-date alors
        Afficher ('La date de fin de validite de
                  l'Activite Divisionnaire est
                  posterieure ou egale a la date de
                  debut de validite d'une A.D.
                  existante. C'est impossible.')
        finschema
      finsi
    sinon
      reponse := ' '
      Tant que reponse <> 'O' et reponse <> 'N' faire
        Afficher ('Voulez-vous rentrer la date de fin de
                  validite de l'Activite Divisionnaire ?
                  (O/N)')
        Repondre (reponse)
        si reponse = 'O' alors
          Afficher ('Entrez la date de fin de
                    validite de l'Activite
                    Divisionnaire :')
        finsi
      finsi
    finsi
  finsi

```



```

ENTRER DATE (da.fd-valid-period-end-date)
  si da.fd-valid-period-end-date <=
    da.fd-valid-period-start-date alors
      Afficher ('La date de fin de validite
        est anterieure a la date
        de debut de validite.C'est
        impossible.')
      finschema
    finsi
  sinon si reponse = 'N' alors
    da.fd-valid-period-end-date := Nil
    finsi
  finsi
  Refaire
finsi
si da.fd-valid-period-end-date <> Nil et
  fctnal.fd-valid-period-end-date <> Nil et
  da.fd-valid-period-end-date > fctnal.fd-valid-period-end-date
    alors
      Afficher ('La date de fin de validite de L'Activite
        Divisionnaire est posterieure a celle de l'Activite
        Fonctionnelle.C'est impossible.')
      finschema
    finsi
  si da.fd-valid-period-end-date <> Nil et
    div.fd-valid-period-end-date <> Nil et
    da.fd-valid-period-end-date > div.fd-valid-period-end-date
      alors
        Afficher ('La date de fin de validite de l'Activite
          Divisionnaire est posterieure a celle de
          la Division.C'est impossible.')
        finschema
      finsi
    da est element de CDD-DIV-ACTIV
    LIER fctnal avec da
    LIER div avec da

    /* CREATION DU DA-MTH ASSOCIE */

    damth.fd-last-updated := date actuelle /* Annee et mois */
    damth.fd-da-cumul-exp := 0
    damth.fd-da-mth-exp := 0
    damth.fd-da-cumul-tid := 0
    damth.fd-da-cumul-open-commit := 0
    damth.fd-da-mth-commit := 0
    damth est element de CDD-DA-MTH
    LIER damth avec da

  finsi
finschema

```

SCHEMA DE PROGRAMME DETRUISANT LOGIQUEMENT UNE ACTIVITE DIVISIONNAIRE

FREQUENCE ESTIMEE :

DEGRE D'IMPORTANCE :

DONNEES : ---

RESULTATS : ---

REMARQUES : Ce schema de programme detruit logiquement une Activite Divisionnaire, c'est-a-dire qu'il permet d'entrer ou de modifier la date de fin de validite de cette Activite Divisionnaire. Auparavant il verifie que :

I. DE TOUTE FACON :

1. L'Activite Fonctionnelle existe et qu'elle ne soit pas close.
2. La Division existe et qu'elle ne soit pas close.
3. L'Activite Divisionnaire existe.
4. Sa date de fin de validite soit posterieure ou egale a la date actuelle.
5. La date de fin de validite de l'Activite Divisionnaire soit posterieure a sa date de debut de validite.
6. La date de fin de validite de l'Activite Divisionnaire soit anterieure a celle :
 1. de l'Activite Fonctionnelle
 2. de la Division
7. Toutes les Taches associees a cette Activite Divisionnaire finissent avant cette date.

II. QUE S'IL EXISTE UNE A.D. AYANT MEME CLE :

1. S'il existe une Activite Divisionnaire de meme code et de date de debut de validite posterieure, celle que l'on veut detruire finisse avant cette date.

dom CODE-ORG : article

FD-DIVISION-CODE :CODE-DIV

FD-SECTION-CODE :CODE-SEC

fin

dom DATE : article

CC :[0,99] entier

YY :[0,99] entier

MM :[0,99] entier

DD :[0,99] entier

fin

dom DATE-MOIS : article

CC:[0,99] entier

YY:[0,99] entier

MM:[0,99] entier

fin

dom PBSKEY : article

FD-ORGANIC-CODE :CODE-ORG

FD-FA-CODE :CODE

FD-TASK-CODE :CODE

fin

dom BOOL :[faux,vrai]

dom BUDGET :[-999'999'999,999'999'999] entier
dom CAR-TROIS :caracteres[3]
dom CHEF :caracteres[30]
dom CODE :caracteres[4]
dom CODE-DIV :[0,99] entier
dom CODE-SEC :[0,999] entier
dom NOM :caracteres[60]

cst FD-DA-CREATION-MTH :DATE-MOIS
cst FD-DA-LEADER :CHEF
cst FD-DA-NAME :NOM
cst FD-DIVISION-ID :CAR-TROIS
cst FD-DIVISION-LEADER :CHEF
cst FD-FA-CODE :CODE
cst FD-FA-NAME :NOM
cst FD-OU-PBS-KEY :PBSKEY
cst FD-TASK-CURRENT-BUDGET :BUDGET
cst FD-TASK-NAME :NOM
cst FD-VALID-PERIOD-END-DATE :DATE et Nil
cst FD-VALID-PERIOD-START-DATE :DATE

rel FD-FCTNAL-ACTIV (FD-FA-CODE,FD-FA-NAME,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE)
ayant cle FD-FA-CODE

rel FD-DIVISION (FD-DIVISION-ID,FD-DIVISION-LEADER,
FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE)
ayant cle FD-DIVISION-ID

rel FD-DIV-ACTIV (FD-FA-CODE,FD-DIVISION-ID,FD-DA-NAME,FD-DA-LEADER,
FD-DA-CREATION-MTH,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE)
ayant cle FD-FA-CODE,FD-DIVISION-ID

rel FD-YEAR-TASKLIST (FD-OU-PBSKEY,FD-TASK-NAME,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE,FD-TASK-CURRENT-BUDGET)
ayant cle FD-OU-PBSKEY

cdd CDD-DA-1 :FD-DIV-ACTIV
cdd CDD-DA-2 :FD-DIV-ACTIV
cdd CDD-DIV-ACT :FD-DIV-ACTIV
cdd CDD-DIV-ACT-1 :FD-DIV-ACTIV
cdd CDD-DIVISION :FD-DIVISION
cdd CDD-FCTNAL-ACTIV :FD-FCTNAL-ACTIV
cdd CDD-YEAR-TASKLIST :FD-YEAR-TASKLIST

var da :FD-DIV-ACTIV
var div :FD-DIVISION et Nil
var fctnal :FD-FCTNAL-ACTIV et Nil
var parcours-tasklist :FD-YEAR-TASKLIST
var pas-clos :BOOL
var reponse :caractere[1]

SCHEMA DE PROGRAMME DETRUISANT LOGIQUEMENT UNE ACTIVITE DIVISIONNAIRE

/* RECHERCHE DE L'ACTIVITE FONCTIONNELLE */

Afficher ('Entrez le code de l'Activite Fonctionnelle :')
Repondre (fctnal,fd-fa-code)

```

fctnal := prendre dans CDD-FCTNAL-ACTIV avec cle = fctnal.fd-fa-code
si fctnal = Nil alors
    Afficher ('Ce code est invalide. Il ne correspond a aucune
                Activite Fonctionnelle. Destruction impossible.')
    finschema
finsi

/* VERIFICATION : L'ACTIVITE FONCTIONNELLE NE DOIT PAS */
/*                               ETRE CLOSE                               */

si fctnal.fd-valid-period-end-date <> Nil et
    fctnal.fd-valid-period-end-date <= date actuelle alors
    Afficher ('Cette Activite Fonctionnelle est close. Toute
                Activite Divisionnaire lui etant associee
                est donc deja logiquement detruite.')
    finschema
finsi

/* RECHERCHE DE LA DIVISION */

Afficher ('Entrez le code de la Division :')
Repondre (div.fd-division-id)
div := prendre dans CDD-DIVISION avec cle = div.fd-division-id
si div = Nil alors
    Afficher ('Ce code de Division est invalide.
                Il ne correspond a aucune Division. Destruction
                impossible.')
    finschema
finsi

/* VERIFICATION : LA DIVISION NE DOIT PAS ETRE CLOSE */

si div.fd-valid-period-end-date <> Nil et
    div.fd-valid-period-end-date <= date actuelle alors
    Afficher ('Cette Division est close. Toute Activite
                Divisionnaire qui lui etait associee
                est donc deja logiquement detruite.')
    finschema
finsi

/* RECHERCHE DE L'ACTIVITE DIVISIONNAIRE */

da.fd-fa-code := fctnal.fd-fa-code
da.fd-division-id := div.fd-division-id
CDD-DIV-ACT-1 := vide /* CDD-DIV-ACT-1 contiendra toutes les Activites
                        Divisionnaires ayant cette cle. */
CDD-DIV-ACT-1 := CDD-DIV-ACTIV filtre par (FD-FA-CODE = da.fd-fa-code,
                                           FD-DIVISION-ID = da.fd-division-id)
Afficher ('Entrez la date de debut de validite de l'Activite Divisionnaire:')
ENTRER DATE (da.fd-valid-period-start-date)
CDD-DA-1 := vide /* CDD-DA-1 contiendra l'Activite Divisionnaire ayant
                  cette cle et cette date de debut de validite, si elle
                  existe. */
CDD-DA-1 := CDD-DIV-ACT-1 filtre par (FD-VALID-PERIOD-START-DATE =
                                       da.fd-valid-period-start-date)
si CDD-DA-1 = vide alors
    Afficher ('Il n'existe aucune Activite Divisionnaire ayant
                ce code et cette date de debut de validite.
                Destruction logique impossible.')
    finschema

```

sinon da := prendre dans CDD-DA-1
finsi

/* CAS OU L'ACTIVITE DIVISIONNAIRE EXISTE */

```

si da.fd-valid-period-end-date <> Nil alors
    Afficher ('Cette Activite Divisionnaire a deja une
                date de fin de validite.C'est le :')
    AFFICHER DATE (da.fd-valid-period-end-date)
    reponse := ' '
    Tant que reponse <> 'O' et reponse <> 'N' faire
        Afficher ('Voulez-vous la changer ? (O/N)')
        Repondre (reponse)
    Refaire
si reponse = 'N' alors
    finschema
finsi

```

Afficher finsi ('Entrez la date de fin de validite de l'Activite Divisionnaire :')
ENTRER DATE (da,fd-valid-period-end-date)

```
/* VERIFICATION QUE LA DATE DE FIN DE VALIDITE SOIT POSTERIEURE *
/* OU EGALE A LA DATE ACTUELLE */
```

```

si da.fd-valid-period-end-date < date actuelle alors
Afficher ('Cette date de fin de validite est impossible
car elle est anterieure a la date actuelle.La
destruction aurait un effet retroactif.
C'est impossible.')

```

finschema
finsi

```
/* VERIFICATION QUE LA DATE DE FIN DE VALIDITE SOIT POSTERIEURE */
/* A LA DATE DE DEBUT DE VALIDITE */
```

```

si da.fd-valid-period-end-date <= da.fd-valid-period-start-date
  Afficher ('La date de fin de validite est anterieure
             a la date de debut de validite. C'est impossible.')
  finschema
finsi

```

```
/* VERIFICATION QUE LA DATE DE FIN DE VALIDITE DE L'A.D. SOIT */
/* ANTERIEURE OU EGALE A CELLE DE L'ACTIVITE FONCTIONNELLE */
```

```

si fctnal.fd-valid-period-end-date <> Nil et
da.fd-valid-period-end-date > fctnal.fd-valid-period-end-date alors
Afficher ('La date de fin de validite de l'Activite Divisionnaire
est posterieure a celle de l'Activite Fonctionnelle.
C'est impossible.')

```

finschema
finsi

```

/* VERIFICATION QUE LA DATE DE FIN DE VALIDITE DE L'A.D. SOIT */
/* ANTERIEURE OU EGALE A CELLE DE LA DIVISION */

```

```

si   div.fd-valid-period-end-date <> Nil   et
da.fd-valid-period-end-date > div.fd-valid-period-end-date alors
Afficher ('La date de fin de validite de l'Activite Divisionnaire
            est posterieure a celle de la Division.

```

C'est impossible')

```

      finschema
    finsi
CDD-DA-2 := vide /* CDD-DA-2 contiendra les Activites Divisionnaires
                  dont la date de debut de validite est posterieure
                  a celle que l'on veut detruire. */
CDD-DA-2 := CDD-DIV-ACT-1 filtre par (FD-VALID-PERIOD-START-DATE >
                                     da.fd-valid-period-start-date)
                                     trie croissant par
                                     (FD-VALID-PERIOD-START-DATE)

```

```

    si CDD-DA-2 <> vide alors
      da-ex := prendre dans CDD-DA-2

```

```

/* VERIFICATION : S'IL EXISTE UNE A.D. DE MEME CODE ET DE DATE */
/* DE DEBUT DE VALIDITE POSTERIEURE, CELLE QUE L'ON VEUT DETRUIRE */
/* FINISSE AVANT CETTE DATE */

```

```

    si da-ex.fd-valid-period-start-date <=
      da.fd-valid-period-end-date alors
      Afficher ('La date de fin de validite de l'Activite
                  Divisionnaire est posterieure ou egale a
                  la date de debut de validite d'une
                  Activite Divisionnaire ayant meme code
                  qui existe deja.C'est impossible.')
```

```

      finschema
    finsi
  finsi

```

```

/* VERIFICATION : IL FAUT QUE TOUTES LES TACHES ASSOCIEES */
/* A CETTE A.D. SOIENT CLOSES POUR POUVOIR DETRUIRE CETTE A.D. */

```

```

pas-clos := faux
parcours-tasklist.fd-fa-code := da.fd-fa-code
parcours-tasklist := premier
  Tant que parcours-tasklist <> dernier ou
    parcours-tasklist = dernier et pas-clos = faux faire
    parcours-tasklist := prendre dans CDD-YEAR-TASKLIST
      si parcours-tasklist.fd-valid-period-end-date >
        da.fd-valid-period-end-date ou
        parcours-tasklist.fd-valid-period-end-date = Nil
          alors
            pas-clos := vrai

```

```

      finsi
    parcours-tasklist := suivant
  Refaire
  si pas-clos = vrai alors
    Afficher ('Il existe au-moins une Tache associee a cette
                Activite Divisionnaire qui se termine apres
                la date de fin de validite de l'Activite
                Divisionnaire.C'est impossible.')
```

```

      finschema
    finsi

```

```

/* DESTRUCTION LOGIQUE DE L'ACTIVITE DIVISIONNAIRE */

```

```

MISE A JOUR de da dans CDD-DIV-ACTIV

```

```

finschema

```

SCHEMA DE PROGRAMME DETRUISANT PHYSIQUEMENT UNE ACTIVITE DIVISIONNAIRE
ET TOUS LES FD-DA-MTH
ET TOUTES LES TACHES QUI LUI SONT ASSOCIES

FREQUENCE ESTIMEE :
DEGRE D'IMPORTANCE :

DONNEES : ---

RESULTATS : ---

REMARQUES : Ce schema de programme detruit physiquement une Activite Divisionnaire ainsi que tous les DA-MTH et toutes les Taches (YR-TASKLIST) qui lui etaient associees.
 Auparavant il verifie que :
 1. l'Activite Fonctionnelle existe (elle peut etre close ou non).
 2. la Division existe (elle peut etre close ou non).
 3. l'Activite Divisionnaire existe et qu'elle soit close.
 4. toutes les Taches associees a cette Activite Divisionnaire aient une date de fin de validite differente de Nil (qu'elles soient donc toutes closes).

dom CODE-ORG : article
 FD-DIVISION-CODE :CODE-DIV
 FD-SECTION-CODE :CODE-SEC
 fin

dom DATE : article
 CC :[0,99] entier
 YY :[0,99] entier
 MM :[0,99] entier
 DD :[0,99] entier
 fin

dom DATE-MOIS : article
 CC:[0,99] entier
 YY:[0,99] entier
 MM:[0,99] entier
 fin

dom PBSKEY : article
 FD-ORGANIC-CODE :CODE-ORG
 FD-FA-CODE :CODE
 FD-TASK-CODE :CODE
 fin

dom BOOL :[faux,vrai]
dom BUDGET :[-999'999'999,999'999'999] entier
dom CAR-TROIS :caracteres[3]
dom CHEF :caracteres[30]
dom CODE :caracteres[4]
dom CODE-DIV :[0,99] entier
dom CODE-SEC :[0,999] entier
dom DEPENSES :[-999'999'999.99,999'999'999.99] reel
dom NOM :caracteres[60]

cst FD-DA-CREATION-MTH :DATE-MOIS
cst FD-DA-CUMUL-EXP :DEPENSES


```

cst  FD-DA-CUMUL-OPEN-COMMIT      :DEPENSES
cst  FD-DA-CUMUL-TID              :DEPENSES
cst  FD-DA-CURRENT-BUDGET         :BUDGET
cst  FD-DA-LEADER                 :CHEF
cst  FD-DA-MTH-EXP                :DEPENSES
cst  FD-DA-MTH-COMMIT             :DEPENSES
cst  FD-DA-NAME                   :NOM
cst  FD-DIVISION-ID               :CAR-TROIS
cst  FD-DIVISION-LEADER           :CHEF
cst  FD-FA-CODE                   :CODE
cst  FD-FA-NAME                   :NOM
cst  FD-LAST-UPDATED              :DATE-MOIS
cst  FD-OU-PBS-KEY                :PBSKEY
cst  FD-TASK-CURRENT-BUDGET       :BUDGET
cst  FD-TASK-NAME                 :NOM
cst  FD-VALID-PERIOD-END-DATE     :DATE et Nil
cst  FD-VALID-PERIOD-START-DATE   :DATE

rel  FD-FCTNAL-ACTIV (FD-FA-CODE,FD-FA-NAME,FD-VALID-PERIOD-START-DATE,
                    FD-VALID-PERIOD-END-DATE)
      ayant cle FD-FA-CODE

rel  FD-DIVISION (FD-DIVISION-ID,FD-DIVISION-LEADER,
                  FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE)
      ayant cle FD-DIVISION-ID

rel  FD-DIV-ACTIV (FD-FA-CODE,FD-DIVISION-ID,FD-DA-NAME,FD-DA-LEADER,
                  FD-DA-CREATION-MTH,FD-CURRENT-BUDGET,
                  FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE)
      ayant cle FD-FA-CODE,FD-DIVISION-ID

rel  FD-DA-MTH (FD-FA-CODE,FD-DIVISION-ID,FD-LAST-UPDATED,
               FD-DA-CUMUL-EXP,FD-DA-MTH-EXP,FD-DA-CUMUL-TID,
               FD-DA-CUMUL-OPEN-COMMIT,FD-DA-MTH-COMMIT)
      ayant cle FD-FA-CODE,FD-DIVISION-ID,FD-LAST-UPDATED

rel  FD-YEAR-TASKLIST (FD-OU-PBSKEY,FD-TASK-NAME,FD-VALID-PERIOD-START-DATE,
                     FD-VALID-PERIOD-END-DATE,FD-TASK-CURRENT-BUDGET)
      ayant cle FD-OU-PBSKEY

cdd  CDD-DA-1                     :FD-DIV-ACTIV
cdd  CDD-DA-MTH                   :FD-DA-MTH
cdd  CDD-DIV-ACTIV                 :FD-DIV-ACTIV
cdd  CDD-DIVISION                  :FD-DIVISION
cdd  CDD-FCTNAL-ACTIV              :FD-FCTNAL-ACTIV
cdd  CDD-YEAR-TASKLIST             :FD-YEAR-TASKLIST

var  da                           :FD-DIV-ACTIV et Nil
var  div                          :FD-DIVISION et Nil
var  fctnal                       :FD-FCTNAL-ACTIV et Nil
var  parcours-damth               :FD-DA-MTH
var  parcours-tasklist            :FD-YEAR-TASKLIST
var  pas-clos                     :BOOL

```

SCHEMA DE PROGRAMME DETRUISANT PHYSIQUEMENT UNE ACTIVITE DIVISIONNAIRE

/* RECHERCHE DE L'ACTIVITE FONCTIONNELLE */

Afficher ('Entrez le code de l'Activite Fonctionnelle :')
Repondre (fctnal.fd-fa-code)


```

fctnal := prendre dans CDD-FCTNAL-ACTIV avec cle = fctnal.f-d-fa-code
      si fctnal = Nil alors
          Afficher ('Ce code est invalide. Il ne correspond a aucune
                    Activite Fonctionnelle. Destruction physique
                    impossible.')
      finschema
finsi

```

/* RECHERCHE DE LA DIVISION */

```

Afficher ('Entrez le code de la Division :')
Repondre (div.f-d-division-id)
div := prendre dans CDD-DIVISION avec cle = div.f-d-division-id
      si div = Nil alors
          Afficher ('Ce code de Division est invalide.
                    Il ne correspond a aucune Division.
                    Destruction physique impossible.')
      finschema
finsi

```

/* RECHERCHE DE L'ACTIVITE DIVISIONNAIRE */

```

da.f-d-fa-code := fctnal.f-d-fa-code
da.f-d-division-id := div.f-d-division-id
Afficher ('Entrez la date de debut de validite de
          l'Activite Divisionnaire :')
ENTRER DATE (da.f-d-valid-period-start-date)
CDD-DA-1 := vide /* CDD-DA-1 contiendra cette Activite Divisionnaire,
                  si elle existe. */
CDD-DA-1 := CDD-DIV-ACTIV filtre par (FD-FA-CODE = da.f-d-fa-code et
                                     FD-DIVISION-ID = da.f-d-division-id et
                                     FD-VALID-PERIOD-START-DATE = da.f-d-valid-period-start-date)

```

/* VERIFICATION QUE CETTE ACTIVITE DIVISIONNAIRE EXISTE */

```

si CDD-DA-1 = vide alors
    Afficher ('L'Activite Divisionnaire correspondant
              aux cles donnees n'existe pas. Destruction
              physique impossible.')
    finschema
sinon da := prendre dans CDD-DA-1
finsi

```

/* VERIFICATION QUE CETTE ACTIVITE DIVISIONNAIRE EST CLOSE */

```

si da.f-d-valid-period-end-date >= date actuelle ou
   da.f-d-valid-period-end-date = Nil alors
    Afficher ('Cette Activite Divisionnaire n'a pas ete logiquement
              detruite. Il est donc impossible de la detruire
              physiquement.')
    finschema
finsi

```

/* VERIFICATION QUE TOUTES LES TACHES ASSOCIEES A CETTE A.D. */
/* SOIENT CLOSES */

```

pas-clos := faux
parcours-tasklist.f-d-fa-code := da.f-d-fa-code
parcours-tasklist := premier
Tant que parcours-tasklist <> dernier ou

```

```

    parcours-tasklist = dernier et pas-clos = faux faire
    parcours-tasklist := prendre dans CDD-YEAR-TASKLIST
    si parcours-tasklist.fd-valid-period-end-date = Nil
        alors
            pas-clos := vrai
        finsi
    parcours-tasklist := suivant

```

Refaire

```

si pas-clos = vrai alors
    Afficher ('Cette Activite Divisionnaire n'a pas ete detruite
        logiquement. Il reste au-moins une Tache associee a
        cette Activite Divisionnaire dont la date de fin de
        validee est egale a Nil .Destruction physique
        impossible.')

```

finschema

finsi

/* DESTRUCTION PHYSIQUE PERMISE */

/* DESTRUCTION DE TOUS LES DA-MTH ASSOCIES*/

```

    parcours-damth.fd-fa-code := fctnal.fd-fa-code
    parcours-damth.fd-division-id := div.fd-division-id
    parcours-damth := premier
    Tant que parcours-damth <> dernier ou parcours-damth = dernier faire
        SUPPRIMER parcours-damth
        parcours-damth := suivant

```

Refaire

/* DESTRUCTION DE TOUTES LES TACHES ASSOCIEES */

```

    parcours-tasklist.fd-fa-code := da.fd-fa-code
    parcours-tasklist := premier
    Tant que parcours-tasklist <> dernier ou
        parcours-tasklist = dernier faire
        DETRUIRE TACHE (parcours-tasklist)
        parcours-tasklist := suivant

```

Refaire

/* DESTRUCTION DE L'ACTIVITE DIVISIONNAIRE */

SUPPRIMER da

finschema

SCHEMA DE PROGRAMME CREANT UN FA-PJ

FREQUENCE ESTIMEE :
DEGRE D'IMPORTANCE :

DONNEES : ---

RESULTATS : ---

REMARQUES : Ce schema de programme cree une association entre un Projet et une Activite Fonctionnelle (FA-PJ).

Auparavant il verifie que :

1. l'Activite Fonctionnelle existe dans la base et qu'elle n'est pas close (les doubles ne sont pas autorises).
2. le Projet existe dans la base et qu'un des doubles n'est pas clos (les doubles sont autorises).
3. l'association FA-PJ n'existe pas deja dans la base.

dom DATE : article
 CC : [0,99] entier
 YY : [0,99] entier
 MM : [0,99] entier
 DD : [0,99] entier
 fin

dom BOOL : [faux,vrai]
dom CHEF : caracteres[30]
dom CODE : caracteres[4]
dom NOM : caracteres[60]

cst FD-FA-CODE : CODE
cst FD-FA-NAME : NOM
cst FD-FAPJ-DUMMY : caracteres[2]
cst FD-PERIOD-COMPL-DATE : DATE
cst FD-PERIOD-START-DATE : DATE
cst FD-PJ-CODE : CODE
cst FD-PJ-LEADER : CHEF
cst FD-PJ-LEADER-DIV : caracteres[3]
cst FD-PJ-NAME : NOM
cst FD-PJ-REVISED : caractere[1]
cst FD-VALID-PERIOD-END-DATE : DATE et Nil
cst FD-VALID-PERIOD-START-DATE : DATE

rel FD-FCTNAL-ACTIV (FD-FA-CODE,FD-FA-NAME,FD-VALID-PERIOD-START-DATE,
 FD-VALID-PERIOD-END-DATE)
 ayant cle FD-FA-CODE

rel FD-PROJECT (FD-PJ-CODE,FD-PJ-REVISED,FD-PJ-NAME,FD-PJ-LEADER,
 FD-PJ-LEADER-DIV,FD-VALID-PERIOD-START-DATE,
 FD-VALID-PERIOD-END-DATE,FD-PERIOD-START-DATE,
 FD-PERIOD-COMPL-DATE)
 ayant cle FD-PJ-CODE

rel FD-FA-PJ (FD-FA-CODE,FD-PJ-CODE,FD-FAPJ-DUMMY)
 ayant cle FD-FA-CODE,FD-PJ-CODE

cdd CDD-FAPJ : FD-FA-PJ
cdd CDD-FCTNAL-ACTIV : FD-FCTNAL-ACTIV
cdd CDD-PJ-2 : FD-PROJECT
cdd CDD-PROJECT : FD-PROJECT

```

var bool :BOOL
var fapj :FD-FA-PJ et Nil
var fctnal :FD-FCTNAL-ACTIV et Nil
var pro :FD-PROJECT et Nil

```

SCHEMA DE PROGRAMME CREATION

```

/* ENTREE DU CODE DE L'ACTIVITE FONCTIONNELLE */
/* ET VERIFICATION QU'IL CORRESPOND BIEN A UNE A.F. */
/* DEJA CONTENUE DANS LA BASE */

```

bool := faux

```

Tant que bool = faux faire
Afficher ('Entrez le code de l'Activite Fonctionnelle :')
Repondre (fctnal.fd-fa-code)
fctnal := prendre dans CDD-FCTNAL-ACTIV
avec cle = fctnal.fd-fa-code
si fctnal = Nil alors
Afficher ('Ce code est invalide. Il ne correspond
a aucune Activite Fonctionnelle.')
sinon
bool := vrai
finsi
Refaire

```

```

/* VERIFICATION QUE CETTE A.F. N'EST PAS CLOSE */

```

```

si fctnal.fd-valid-period-end-date <> Nil
et fctnal.fd-valid-period-end-date <= date actuelle
alors
Afficher ('Cette Activite Fonctionnelle est close.
Il est impossible de lui associer un Projet
(FA-PJ)')
finschema

```

sinon

```

/* ENTREE DU CODE PROJET ET VERIFICATION */
/* QU'IL CORRESPOND BIEN A UN PROJET DEJA */
/* CONTENU DANS LA BASE */

```

bool := faux

```

Tant que bool = faux faire
Afficher ('Entrez le code du Projet :')
Repondre (pro.fd-pj-code)
Afficher ('Entrez sa date de debut de validite :')
ENTRER DATE (pro.fd-valid-period-start-date)

```

```

CDD-PJ-2 := vide /* CDD-PJ-2 contiendra le Projet qui a ce code
et cette date de debut de validite, s'il existe */

```

```

CDD-PJ-2 := CDD-PROJECT filtre par (FD-PJ-CODE = pro.fd-pj-code
et FD-VALID-PERIOD-START-DATE =
pro.fd-valid-period-start-date)

```

```

si CDD-PJ-2 = vide alors
Afficher ('Creation impossible. Il n'existe aucun Projet
qui a ce code et cette date de debut de
validite.')
finschema

```

```

sinon pro := prendre dans CDD-PJ-2

```

bool := vrai

finsi
Refaire

/* VERIFICATION QUE CE PROJET N'EST PAS CLOS */

si pro.fd-valid-period-end-date <> Nil et
pro.fd-valid-period-end-date <= date actuelle alors
Afficher ('Ce Projet est clos. Il est impossible
de lui associer un FD-FA-PJ.')
finschema
finsi

/* VERIFICATION QUE CETTE ASSOCIATION FA-PJ N'EXISTE PAS */
/* DEJA DANS LA BASE */

fapj.fd-pj-code := pro.fd-pj-code
fapj.fd-fa-code := fctnal.fd-fa-code
fapj := prendre dans CDD-FAPJ
avec cle fapj.[fd-pj-code, fd-fa-code]
si fapj <> Nil alors
Afficher ('Cette association entre
le Projet', pro.fd-pj-code,
'et l'Activite Fonctionnelle'
fctnal.fd-fa-code, 'est deja
contenue dans la base.
Creation impossible.')
finschema
finsi

/* CREATION DE L'ASSOCIATION FA-PJ */

fapj.fd-fapj-dummy := ' '
fapj est element de CDD-FAPJ
LIER fapj avec pro
LIER fapj avec fctnal

finsi

finsi
finschema

SCHEMA DE PROGRAMME DETRUISANT UNE ASSOCIATION FA-PJ

FREQUENCE ESTIMEE :

DEGRE D'IMPORTANCE :

DONNEES : ---

RESULTATS : ---

REMARQUES : Ce schema de Programme cree une association FA-PJ a condition que :

1. 1'Activite Fonctionnelle existe et qu'elle ne soit pas close.
2. le Projet existe et qu'il ne soit pas clos.
3. 1'association FA-PJ existe

dom DATE : article
 CC : [0,99] entier
 YY : [0,99] entier
 MM : [0,99] entier
 DD : [0,99] entier
 fin

dom BOOL : [faux,vrai]
dom CHEF : caracteres[30]
dom CODE : caracteres[4]
dom NOM : caracteres[60]

cst FD-FA-CODE :CODE
cst FD-FA-NAME :NOM
cst FD-FAPJ-DUMMY :caracteres[2]
cst FD-PERIOD-COMPL-DATE :DATE
cst FD-PERIOD-START-DATE :DATE
cst FD-PJ-CODE :CODE
cst FD-PJ-LEADER :CHEF
cst FD-PJ-LEADER-DIV :caracteres[3]
cst FD-PJ-NAME :NOM
cst FD-PJ-REVISED :caractere[1]
cst FD-VALID-PERIOD-END-DATE :DATE et Nil
cst FD-VALID-PERIOD-START-DATE :DATE

rel FD-FCTNAL-ACTIV (FD-FA-CODE,FD-FA-NAME,FD-VALID-PERIOD-START-DATE,
 FD-VALID-PERIOD-END-DATE)
 ayant cle FD-FA-CODE

rel FD-PROJECT (FD-PJ-CODE,FD-PJ-REVISED,FD-PJ-NAME,FD-PJ-LEADER,
 FD-PJ-LEADER-DIV,FD-VALID-PERIOD-START-DATE,
 FD-VALID-PERIOD-END-DATE,FD-PERIOD-START-DATE,
 FD-PERIOD-COMPL-DATE)
 ayant cle FD-PJ-CODE

rel FD-FA-PJ (FD-FA-CODE,FD-PJ-CODE,FD-FAPJ-DUMMY)
 ayant cle FD-FA-CODE,FD-PJ-CODE

cdd CDD-FAPJ :FD-FA-PJ
cdd CDD-FCTNAL-ACTIV :FD-FCTNAL-ACTIV
cdd CDD-PJ-1 :FD-PROJECT
cdd CDD-PROJECT :FD-PROJECT

```

var bool :BOOL
var fapj :FD-FA-PJ et Nil
var fctnal :FD-FCTNAL-ACTIV et Nil
var pro :FD-PROJECT

```

} pointer d'entité

SCHEMA DE PROGRAMME DESTRUCTION

```

/* ENTREE DU CODE DE L'ACTIVITE FONCTIONNELLE */
/* ET VERIFICATION QU'IL CORRESPOND BIEN A UNE A.F. */
/* DEJA CONTENUE DANS LA BASE */

```

```

Afficher ('Entrez le code de l'Activite Fonctionnelle :')
Repondre (fctnal.fd-fa-code)
fctnal := prendre dans CDD-FCTNAL-ACTIV avec cle = fctnal.fd-fa-code
      si fctnal = Nil alors
        Afficher ('Ce code est invalide. Il ne correspond
                  a aucune Activite Fonctionnelle.')
      finschema
    finsi

```

```

/* VERIFICATION QUE CETTE A.F. N'EST PAS CLOSE */

```

```

si fctnal.fd-valid-period-end-date <> Nil
  et fctnal.fd-valid-period-end-date <= date actuelle
    alors
      Afficher ('Cette Activite Fonctionnelle est close.
                Il est donc impossible de detruire une
                de ses associations avec un Projet.
                (FA-PJ)')
    finschema

```

sinon

```

/* ENTREE DU CODE PROJET ET VERIFICATION */
/* QU'IL CORRESPOND BIEN A UN PROJET DEJA */
/* CONTENU DANS LA BASE */

```

```

Afficher ('Entrez le code du Projet :')
Repondre (pro.fd-pj-code)
Afficher ('Entrez la date de debut de validite du
          Projet :')
ENTRER DATE(pro.fd-valid-period-start-date)
CDD-PJ-1 := vide /* CDD-PJ-1 contiendra le Projet
                  ayant ce code et cette date de debut
                  de validite, s'il existe */
CDD-PJ-1 := CDD-PROJECT filtre par (FD-PJ-CODE =
  pro.fd-pj-code et FD-VALID-PERIOD-START-DATE =
  pro.fd-valid-period-start-date)
si CDD-PJ-1 = vide alors
  Afficher ('Il n'existe aucun Projet ayant un tel
            code et une telle date de debut de
            validite. Destruction impossible')
  finschema
sinon pro := prendre dans CDD-PJ-1
finis

```

```

/* VERIFICATION QUE CE PROJET N'EST PAS CLOS */

```

```

si pro.fd-valid-period-end-date <> Nil
  et pro.fd-valid-period-end-date <= date

```

```

                                actuelle alors
Afficher ('Ce Projet est clos. Il est
                                impossible de detruire cette
                                FA-PJ.')
                                finschema
                                finsi
finsi
fapj.fd-fa-code := fctnal.fd-fa-code
fapj.fd-pj-code := pro.fd-pj-code
fapj := prendre dans CDD-FAPJ avec cle = fapj[fd-fa-code,fd-pj-code]
        si fapj = Nil alors
            Afficher ('Il n'existe pas de FA-PJ de tel code.
                        Destruction impossible.')
        finschema
        sinon
            SUPPRIMER fapj
        finsi
finschema
```


SCHEMA DE PROGRAMME CREANT UN OOAK-FA
UNE ACTIVITE FONCTIONNELLE

FREQUENCE ESTIMEE :
DEGRE D'IMPORTANCE :

DONNEES : ---

RESULTATS : ---

REMARQUES : Ce schema de programme cree une Activite Fonctionnelle et la tete de liste (OOAK-FA) qui permet de les atteindre si elle n'existait pas deja.
Auparavant il verifie que l'Activite Fonctionnelle n'existait pas deja dans la base.

dom DATE : article
CC :[0,99] entier
YY :[0,99] entier
MM :[0,99] entier
DD :[0,99] entier
fin

dom BOOL :[faux,vrai]
dom CAR-DEUX :caracteres[2]
dom CODE :caracteres[4]
dom NOM :caracteres[60]

cst FD-FA-CODE :CODE
cst FD-FA-HEADER :CAR-DEUX
cst FD-FA-NAME :NOM
cst FD-VALID-PERIOD-END-DATE :DATE et Nil
cst FD-VALID-PERIOD-START-DATE :DATE

rel FD-OOAK-FA (FD-FA-HEADER)
ayant cle FD-FA-HEADER

rel FD-FCTNAL-ACTIV (FD-FA-CODE,FD-FA-NAME,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE)
ayant cle FD-FA-CODE

cdd CDD-FCTNAL-ACTIV :FD-FCTNAL-ACTIV
cdd CDD-OOAK-FA :FD-OOAK-FA

var bool :BOOL
var fctnal :FD-FCTNAL-ACTIV et Nil
var oak :FD-OOAK-FA et Nil

SCHEMA DE PROGRAMME CREATION D'UNE FD-FCTNAL-ACTIV

/* ENTREE DU CODE DE L'ACTIVITE FONCTIONNELLE */
/* CREATION DE LA TETE DE LISTE SI ELLE N'EXISTE PAS */

oak.fd-fa-header := ' '
oak := prendre dans CDD-OOAK-FA avec cle = oak.fd-fa-header
si oak := Nil alors
oak est element de CDD-OAK-FA
finsi
bool := faux
Tant que bool = faux faire

```
Afficher ('Entrez le code de l'Activite Fonctionnelle :')
Repondre (fctnal.fd-fa-code)
fctnal := prendre dans CDD-FCTNAL-ACTIV
           avec cle = fctnal.fd-fa-code
           si fctnal <> Nil alors
               Afficher ('Ce code d'Activite Fonctionnelle
                           est deja actif.
                           Creation impossible.')
           sinon bool := vrai
           finsi
```

Refaire

/* CREATION DE L'ACTIVITE FONCTIONNELLE */

```
Afficher ('Entrez le nom de l'Activite Fonctionnelle :')
Repondre (fctnal.fd-fa-name)
VALIDER DATE DEBUT-FIN (fctnal.fd-valid-period-start-date,
                        fctnal.fd-valid-period-end-date)
LIER oak avec fctnal
finschema
```

SCHEMA DE PROGRAMME DETRUISANT LOGIQUEMENT UNE ACTIVITE FONCTIONNELLE
ET DETRUISANT LOGIQUEMENT TOUS LES FD-DIV-ACTIV ASSOCIES

FREQUENCE ESTIMEE :

DEGRE D'IMPORTANCE :

DONNEES : ---

RESULTATS : ---

REMARQUES : Ce schema de programme detruit logiquement une Activite Fonctionnelle et toutes les Activites Divisionnaires associees qui ne sont pas deja detruites logiquement. Auparavant il verifie :

I. QUE DE TOUTES FACONS :

1. L'Activite Fonctionnelle que l'on veut detruire existe.
2. Sa date de fin de validite soit posterieure ou egale a la date actuelle.
3. Sa date de fin de validite soit posterieure a sa date de debut de validite.
4. Sa date de fin de validite soit posterieure a la date de debut de validite de toutes les Activites Divisionnaires qui lui sont associees.
5. Les Taches qui sont associees a cette Activite Fonctionnelle soient toutes closes a cette date (date de fin de validite de l'Activite Fonctionnelle).

dom CODE-ORG : article
FD-DIVISION-CODE :CODE-DIV
FD-SECTION-CODE :CODE-SEC
fin

dom DATE : article
CC :[0,99] entier
YY :[0,99] entier
MM :[0,99] entier
DD :[0,99] entier
fin

dom DATE-MOIS : article
CC:[0,99] entier
YY:[0,99] entier
MM:[0,99] entier
fin

dom PBSKEY : article
FD-ORGANIC-CODE :CODE-ORG
FD-FA-CODE :CODE
FD-TASK-CODE :CODE
fin

dom BOOL :[faux,vrai]
dom BUDGET :[-999'999'999,999'999'999] entier
dom CAR-TROIS :caracteres[3]
dom CHEF :caracteres[30]
dom CODE :caracteres[4]
dom CODE-DIV :[0,99] entier
dom CODE-SEC :[0,999] entier
dom NOM :caracteres[60]

```

cst  FD-DA-CREATION-MTH           :DATE-MOIS
cst  FD-DA-LEADER                 :CHEF
cst  FD-DA-NAME                   :NOM
cst  FD-DIVISION-ID              :CAR-TROIS
cst  FD-FA-CODE                   :CODE
cst  FD-FA-NAME                   :NOM
cst  FD-OU-PBS-KEY                :PBSKEY
cst  FD-TASK-CURRENT-BUDGET       :BUDGET
cst  FD-TASK-NAME                 :NOM
cst  FD-VALID-PERIOD-END-DATE     :DATE et Nil
cst  FD-VALID-PERIOD-START-DATE   :DATE

rel  FD-FCTNAL-ACTIV (FD-FA-CODE,FD-FA-NAME,FD-VALID-PERIOD-START-DATE,
                        FD-VALID-PERIOD-END-DATE)
      ayant cle  FD-FA-CODE

rel  FD-DIV-ACTIV (FD-FA-CODE,FD-DIVISION-ID,FD-DA-NAME,FD-DA-LEADER,
                    FD-DA-CREATION-MTH,FD-VALID-PERIOD-START-DATE,
                    FD-VALID-PERIOD-END-DATE)
      ayant cle  FD-FA-CODE,FD-DIVISION-ID

rel  FD-YEAR-TASKLIST (FD-OU-PBSKEY,FD-TASK-NAME,FD-VALID-PERIOD-START-DATE,
                       FD-VALID-PERIOD-END-DATE,FD-TASK-CURRENT-BUDGET)
      ayant cle  FD-OU-PBSKEY

cdd  CDD-DIV-ACTIV                :FD-DIV-ACTIV
cdd  CDD-FCTNAL-ACTIV             :FD-FCTNAL-ACTIV
cdd  CDD-TASK-PAS-CLOSES          :FD-YEAR-TASKLIST
cdd  CDD-YEAR-TASKLIST            :FD-YEAR-TASKLIST

var  bool                         :BOOL
var  fctnal                       :FD-FCTNAL-ACTIV et Nil
var  parcours-da                  :FD-DIV-ACTIV
var  pas-clos                     :BOOL
var  reponse                      :caractere[1]

```

SCHEMA DE PROGRAMME DETRUISANT LOGIQUEMENT UNE ACTIVITE FONCTIONNELLE

/* ENTREE DU CODE DE L'ACTIVITE FONCTIONNELLE */

```

Afficher ('Entrez le code de l'Activite Fonctionnelle :')
Repondre (fctnal.fd-fa-code)
fctnal := prendre dans CDD-FCTNAL-ACTIV avec cle = fctnal.fd-fa-code
si fctnal = Nil alors
  Afficher ('Ce code ne correspond a aucune Activite
              Fonctionnelle.Destruction logique impossible.')
  finschema
finsi

```

/* CAS OU L'A.F. A DEJA UNE DATE DE FIN */
/* DE VALIDITE */

```

si fctnal.fd-valid-period-end-date <> Nil alors
  Afficher ('Cette Activite Fonctionnelle a deja une date de
              fin de validite. Cette date est le :')
  AFFICHER DATE (fctnal.fd-valid-period-end-date)
  reponse := ' '
  Tant que reponse <> 'O' et reponse <> 'N' faire
    Afficher ('Voulez-vous la changer ? (O/N)')

```

[illegible]

```

                                la date de fin de validite de l'A.F. */
CDD-TASK-PAS-CLOSES := CDD-YEAR-TASKLIST filtre par
                                (FD-FA-CODE = fctnal.fd-fa-code et
                                FD-VALID-PERIOD-END-DATE >
                                fctnal.fd-valid-period-end-date ou
                                FD-VALID-PERIOD-END-DATE = Nil )
si CDD-TASK-PAS-CLOSES <> vide alors
    Afficher ('Il existe au-moins une Tache associee a cette
    Activite Fonctionnelle (par l'intermediaire d'une
    Activite Divisionnaire) qui se termine apres la
    date de fin de validite de l'Activite
    Fonctionnelle.C'est impossible.')
    finschema
finsi

/* DESTRUCTION LOGIQUE DES ACTIVITES DIVISIONNAIRES DONT */
/* LA DATE DE FIN DE VALIDITE EST EGALE A NIL OU DONT LA */
/* DATE DE FIN DE VALIDITE EST SUPERIEURE A CELLE DE */
/* L'ACTIVITE FONCTIONNELLE */

parcours-da.fd-fa-code := fctnal.fd-fa-code
parcours-da := premier
    Tant que parcours-da <> dernier ou parcours-da = dernier faire
        parcours-da := prendre dans CDD-DIV-ACTIV
        si parcours-da.fd-valid-period-end-date = Nil ou
            parcours-da.fd-valid-period-end-date >
                fctnal.fd-valid-period-end-date alors
                    parcours-da.fd-valid-period-end-date :=
                        fctnal.fd-valid-period-end-date
        MISE A JOUR de parcours-da dans CDD-DIV-ACTIV
        parcours-da := suivant
    Refaire

    /* DESTRUCTION LOGIQUE DE L'ACTIVITE FONCTIONNELLE */

MISE A JOUR de fctnal dans CDD-FCTNAL-ACTIV

finschema

```

SCHEMA DE PROGRAMME DETRUISANT PHYSIQUEMENT UNE FD-FCTNAL-ACTIV
AINSI QUE TOUS LES FD-FA-MTH
FD-FA-PJ
FD-DIV-ACTIV
FD-DA-MTH
FD-YEAR-TASKLIST
ASSOCIES

FREQUENCE ESTIMEE :
DEGRE D'IMPORTANCE :

DONNEES : ---

RESULTATS : ---

RÉMARQUES : Ce schema de programme detruit physiquement une Activite Fonctionnelle et :
 1. tous les FA-MTH associes.
 2. tous les FA-PJ associes.
 3. toutes les Activites Divisionnaires associees, qui elles-memes obligent a detruire :
 3.1 tous les DA-MTH associes.
 3.2 toutes les Taches associees.
 4. met a jour ou detruit la tete de liste.
 Auparavant il verifie que :
 1. l'Activite Fonctionnelle existe et qu'elle soit close.
 2. toutes les Activites Divisionnaires associees soient closes.
 3. toutes les Taches Associees soient closes.

dom CODE-ORG : article
 FD-DIVISION-CODE :CODE-DIV
 FD-SECTION-CODE :CODE-SEC
 fin

dom DATE : article
 CC :[0,99] entier
 YY :[0,99] entier
 MM :[0,99] entier
 DD :[0,99] entier
 fin

dom DATE-MOIS : article
 CC:[0,99] entier
 YY:[0,99] entier
 MM:[0,99] entier
 fin

dom PBSKEY : article
 FD-ORGANIC-CODE :CODE-ORG
 FD-FA-CODE :CODE
 FD-TASK-CODE :CODE
 fin

dom BUDGET :[-999'999'999,999'999'999] entier
dom CAR-DEUX :caracteres[2]
dom CAR-TROIS :caracteres[3]
dom CHEF :caracteres[30]
dom CODE :caracteres[4]

dom CODE-DIV :[0,99] entier
dom CODE-SEC :[0,999] entier
dom DEPENSES :[-999'999'999.99,999'999'999.99] reel
dom NOM :caracteres[60]

cst FD-DA-CREATION-MTH :DATE-MOIS
cst FD-DA-CUMUL-EXP :DEPENSES
cst FD-DA-CUMUL-OPEN-COMMIT :DEPENSES
cst FD-DA-CUMUL-TID :DEPENSES
cst FD-DA-CURRENT-BUDGET :BUDGET
cst FD-DA-LEADER :CHEF
cst FD-DA-MTH-COMMIT :DEPENSES
cst FD-DA-MTH-EXP :DEPENSES
cst FD-DA-NAME :NOM
cst FD-DIVISION-ID :CAR-TROIS
cst FD-FA-CODE :CODE
cst FD-FA-CUMUL-OPEN-COMMIT :DEPENSES
cst FD-FA-CUMUL-TID :DEPENSES
cst FD-FA-CURRENT-BUDGET :BUDGET
cst FD-FA-HEADER :CAR-DEUX
cst FD-FA-MTH-COMMIT :DEPENSES
cst FD-FA-MTH-EXP :DEPENSES
cst FD-FA-NAME :NOM
cst FD-FAPJ-DUMMY :CAR-DEUX
cst FD-FA-YEAR-EXP :DEPENSES
cst FD-LAST-UPDATED :DATE-MOIS
cst FD-OU-PBS-KEY :PBSKEY
cst FD-TASK-CURRENT-BUDGET :BUDGET
cst FD-TASK-NAME :NOM
cst FD-VALID-PERIOD-START-DATE :DATE
cst FD-VALID-PERIOD-END-DATE :DATE et Nil

rel FD-OOAK-FA (FD-FA-HEADER)
ayant cle FD-FA-HEADER

rel FD-FCTNAL-ACTIV (FD-FA-CODE,FD-FA-NAME,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE)
ayant cle FD-FA-CODE

rel FD-FA-MTH (FD-FA-CODE,FD-LAST-UPDATED,FD-FA-CURRENT-BUDGET,
FD-FA-YEAR-EXP,FD-FA-MTH-EXP,FD-FA-CUMUL-TID,
FD-FA-CUMUL-OPEN-COMMIT,FD-FA-MTH-COMMIT)
ayant cle FD-FA-CODE,FD-LAST-UPDATED

rel FD-DIV-ACTIV (FD-FA-CODE,FD-DIVISION-ID,FD-DA-NAME,FD-DA-LEADER,
FD-DA-CREATION-MTH,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE)
ayant cle FD-FA-CODE,FD-DIVISION-ID

rel FD-DA-MTH (FD-FA-CODE,FD-DIVISION-ID,FD-LAST-UPDATED,FD-DA-CUMUL-EXP,
FD-DA-CURRENT-BUDGET,FD-DA-MTH-EXP,FD-DA-CUMUL-TID,
FD-DA-CUMUL-OPEN-COMMIT,FD-DA-MTH-COMMIT)
ayant cle FD-FA-CODE,FD-DIVISION-ID,FD-LAST-UPDATED

rel FD-YEAR-TASKLIST (FD-OU-PBSKEY,FD-TASK-NAME,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE,FD-TASK-CURRENT-BUDGET)
ayant cle FD-OU-PBSKEY

rel FD-FA-PJ (FD-FA-CODE,FD-PJ-CODE,FD-FAPJ-DUMMY)
ayant cle FD-FA-CODE,FD-PJ-CODE


```

cdd CDD-DA-A-NIL :FD-DIV-ACTIV
cdd CDD-DA-MTH :FD-DA-MTH
cdd CDD-DIV-ACTIV :FD-DIV-ACTIV
cdd CDD-FA-MTH :FD-FA-MTH
cdd CDD-FA-PJ :FD-FA-PJ
cdd CDD-FCTNAL-ACTIV :FD-FCTNAL-ACTIV
cdd CDD-OOAK-FA :FD-OOAK-FA
cdd CDD-TASK-A-NIL :FD-YEAR-TASKLIST
cdd CDD-YEAR-TASKLIST :FD-YEAR-TASKLIST

```

```

var fctnal :FD-FCTNAL-ACTIV et Nil
var parcours-da :FD-DIV-ACTIV
var parcours-damth :FD-DA-MTH
var parcours-famth :FD-FA-MTH
var parcours-fapj :FD-FA-PJ
var parcours-tasklist :FD-YEAR-TASKLIST
var oak :FD-OOAK-FA

```

SCHEMA DE PROGRAMME DETRUISANT PHYSIQUEMENT UNE FD-FCTNAL-ACTIV

/* ENTREE DU CODE DE L'ACTIVITE FONCTIONNELLE */

Afficher ('Entrez le code de l'Activite Fonctionnelle :')

Repondre (fctnal.fd-fa-code)

fctnal := prendre dans CDD-FCTNAL-ACTIV avec cle = fctnal.fd-fa-code

si fctnal = Nil alors

Afficher ('Ce code ne correspond a aucune Activite Fonctionnelle.
Destruction impossible.')

finschema

finsi

/* VERIFIER QUE L'ACTIVITE FONCTIONNELLE EST CLOSE */

si fctnal.fd-valid-period-end-date = Nil ou

fctnal.fd-valid-period-end-date >= date actuelle alors

Afficher ('Cette Activite Fonctionnelle n'est pas encore
close.Destruction physique impossible.')

finschema

finsi

/* VERIFIER QUE TOUTES LES ACTIVITES DIVISIONNAIRES */

/* ASSOCIEES A CETTE ACTIVITE FONCTIONNELLE SONT CLOSES */

CDD-DA-A-NIL := vide /* CDD-DA-A-NIL contiendra toutes les Activites
Divisionnaires dont la date de fin de validite
est egale a Nil .S'il en existe une, cela veut
dire que l'A.F. n'a pas ete logiquement detruite. */

CDD-DA-A-NIL := CDD-DIV-ACTIV filtre par (FD-FA-CODE = fctnal.fd-fa-code et
FD-VALID-PERIOD-END-DATE = Nil)

si CDD-DA-A-NIL <> vide alors

Afficher ('Cette Activite Fonctionnelle n'a pas ete detruite
logiquement.Il reste des Activites Divisionnaires qui
lui sont associees et dont la date de fin de validite
est egale a Nil . Destruction physique impossible.')

finschema

finsi

/* VERIFICATION QUE TOUTES LES TACHES ASSOCIEES A CETTE */

/* ACTIVITE FONCTIONNELLES SONT CLOSES */

```

CDD-TASK-A-NIL := vide /* CDD-TASK-A-NIL contiendra toutes les Taches
                        dont la date de fin de validite = Nil .S'il
                        en existe une cela veut dire que l'A.F. n'a
                        pas ete logiquement detruite. */
CDD-TASK-A-NIL := CDD-YEAR-TASKLIST filtre par (FD-FA-CODE =
                                                fctnal.fd-fa-code et
                                                FD-VALID-PERIOD-END-DATE = Nil )
si CDD-TASK-A-NIL <> vide alors
    Afficher ('Cette Activite Fonctionnelle n'a pas ete detruite
                logiquement.Il reste des Taches associees a cette
                Activite Fonctionnelle dont la date de fin de
                validite est egale a Nil .Destruction physique
                impossible.')
    finschema
finsi

/* DESTRUCTION PHYSIQUE PERMISE */

/* DESTRUCTION DE TOUS LES FD-FA-MTH ASSOCIES */

parcours-famth.fd-fa-code := fctnal.fd-fa-code
parcours-famth := premier
    Tant que parcours-famth <> dernier ou parcours-famth = dernier
                                                faire
        SUPPRIMER parcours-famth
        parcours-famth := suivant
    Refaire

/* DESTRUCTION DE TOUS LES FD-FA-PJ ASSOCIES */

parcours-fapj.fd-fa-code := fctnal.fd-fa-code
parcours-fapj := premier
    Tant que parcours-fapj <> dernier ou parcours-fapj = dernier
                                                faire
        SUPPRIMER parcours-fapj
        parcours-fapj := suivant
    Refaire

/* DESTRUCTION DE TOUS LES FD-DIV-ACTIV ET */
/* DE TOUS LES FD-DA-MTH */
/* ET DE TOUTES LES TACHES */
/* ASSOCIES */

parcours-da.fd-fa-code := fctnal.fd-fa-code
parcours-da := premier
    Tant que parcours-da <> dernier ou parcours-da = dernier
                                                faire
        parcours-damth.fd-fa-code := fctnal.fd-fa-code
        parcours-damth.fd-division-id := parcours-da.fd-division-id
        parcours-damth := premier
            Tant que parcours-damth <> dernier ou
                                                parcours-damth = dernier faire
                SUPPRIMER parcours-damth
                parcours-damth := suivant
            Refaire
        parcours-tasklist.fd-fa-code := fctnal.fd-fa-code
        parcours-tasklist := premier
            Tant que parcours-tasklist <> dernier ou
                                                parcours-tasklist = dernier faire

```

DETRUIRE TACHE (parcours-tasklist)
parcours-tasklist := suivant

Refaire
SUPPRIMER parcours-da
parcours-da := suivant
Refaire

/* DESTRUCTION OU MISE A JOUR DE LA TETE DE LISTE */

DELIER oak avec fctnal
si CDD-OOAK-FA = vide alors
SUPPRIMER oak
finsi
SUPPRIMER fctnal
finschema

SCHEMA DE PROGRAMME CREANT UNE TACHE

FREQUENCE ESTIMEE :

DEGRE D'IMPORTANCE :

DONNEES : ---

RESULTATS : ---

REMARQUES : Ce schema de programme cree une Tache.

Auparavant il verifie :

I. QUE DE TOUTE FACON :

1. L'Unite Organique a laquelle est associee cette Tache existe deja dans la base et qu'elle ne soit pas close.
2. L'Activite Fonctionnelle a laquelle est indirectement associee cette Tache existe deja dans la base.
3. L'Activite Divisionnaire a laquelle est associee cette Tache existe deja dans la base et qu'elle ne soit pas close.
4. Le Projet auquel est associe la Tache (par l'intermediaire de PJ-OU-STATUS) existe deja dans la base et qu'il ne soit pas clos.
5. La date de debut de validite de la Tache soit posterieure ou egale a celle :
 1. de l'Unite Organique
 2. de l'Activite Divisionnaire
 3. du Projet
6. La date de fin de validite de la Tache soit posterieure a sa date de debut de validite.
7. La date de fin de validite de la Tache soit anterieure a celle :
 1. de l'Unite Organique
 2. de l'Activite Divisionnaire
 3. du Projet
8. Il n'y ait pas deja 5 Taches actives (durant la periode de validite de la Tache) associee a l'Unite Organique.

Remarque : une Tache "active" designe une Tache qui a au-moins un jour de validite durant la periode de validite de la Tache que l'on veut creer.

Remarque : si la date de fin de validite de la Tache que l'on veut creer est egale a Nil alors toutes les Taches associees a cette Unite Organique dont les dates de debut de validite sont posterieures seront considerees comme actives.

9. Le Budget de l'Activite Divisionnaire est superieur ou egal a la somme des budgets des Taches associees a cette Activite Divisionnaire.

II. QUE S'IL EXISTE UNE TACHE AYANT MEME CLE :

1. Celle que l'on veut creer n'ait pas la meme date de debut de validite que celle qui existe deja.
2. Celle que l'on veut creer ne commence pas durant la periode de validite de celle qui existe deja.
3. S'il existe Tache de meme code et de date de debut de validite posterieure, celle que l'on veut creer finisse avant cette date.

dom CODE-ORG : article

FD-DIVISION-CODE :CODE-DIV
FD-SECTION-CODE :CODE-SEC

fin

dom DATE : article
CC :[0,99] entier
YY :[0,99] entier
MM :[0,99] entier
DD :[0,99] entier

fin

dom DATE-MOIS : article
CC :[0,99] entier
YY :[0,99] entier
MM :[0,99] entier

fin

dom PBSKEY : article
FD-ORGANIC-CODE :CODE-ORG
FD-FA-CODE :CODE
FD-TASK-CODE :CODE

fin

dom BOOL :[faux,vrai]
dom BUDGET :[-999'999'999,999'999'999] entier
dom CODE :caracteres[4]
dom CODE-DIV :[0,99] entier
dom CODE-SEC :[0,999] entier
dom CHEF :caracteres[30]
dom DEPENSES :[-999'999'999.99, 999'999'999.99] reel
dom FERME :[O,C]
dom LEVEL :[-999,999] entier
dom NOM :caracteres[60]
dom NOM1 :caracteres[30]
dom TYPE-OF-EXPENSE :[W,T,L,M,P,R,E,S]

cst FD-DA-CREATION-MTH :DATE-MOIS
cst FD-DA-CUMUL-EXP :DEPENSES
cst FD-DA-CUMUL-OPEN-COMMIT :DEPENSES
cst FD-DA-CUMUL-TID :DEPENSES
cst FD-DA-CURRENT-BUDGET :BUDGET
cst FD-DA-LEADER :CHEF
cst FD-DA-MTH-COMMIT :DEPENSES
cst FD-DA-MTH-EXP :DEPENSES
cst FD-DA-NAME :NOM
cst FD-DIVISION-ID :caracteres[3]
cst FD-DIVISION-LEADER :CHEF
cst FD-DIVISION-NAME :NOM1
cst FD-EXTRACTION-LEVEL :LEVEL
cst FD-FA-CODE :CODE
cst FD-FA-NAME :NOM
cst FD-LAST-UPDATED :DATE-MOIS
cst FD-OU-CLOSED :FERME
cst FD-OU-LEADER :CHEF
cst FD-OU-NAME :NOM1
cst FD-OU-PBSKEY :PBSKEY
cst FD-PERIOD-COMPL-DATE :DATE
cst FD-PERIOD-START-DATE :DATE
cst FD-PJ-CODE :CODE

```

cst  FD-PJ-LEADER                :CHEF
cst  FD-PJ-LEADER-DIV            :caracteres[3]
cst  FD-PJ-NAME                  :NOM
cst  FD-PJ-REVISED               :caractere[1]
cst  FD-SIGNING-LEVEL            :LEVEL
cst  FD-TASK-CURRENT-BUDGET      :BUDGET
cst  FD-TASK-NAME                 :NOM
cst  FD-VALID-PERIOD-END-DATE    :DATE et Nil
cst  FD-VALID-PERIOD-START-DATE  :DATE

rel  FD-YEAR-TASKLIST (FD-OU-PBSKEY,FD-TASK-NAME,FD-VALID-PERIOD-START-DATE,
                        FD-VALID-PERIOD-END-DATE,FD-TASK-CURRENT-BUDGET)
      ayant cle  FD-OU-PBSKEY

rel  FD-ORGANIC-UNIT (FD-ORGANIC-CODE,FD-OU-NAME,FD-OU-LEADER,
                        FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE,
                        FD-EXTRACTION-LEVEL,FD-SIGNING-LEVEL,FD-OU-CLOSED)
      ayant cle  FD-ORGANIC-CODE

rel  FD-FCTNAL-ACTIV (FD-FA-CODE,FD-FA-NAME,FD-VALID-PERIOD-START-DATE,
                        FD-VALID-PERIOD-END-DATE)
      ayant cle  FD-FA-CODE

rel  FD-DIV-CODE (FD-DIVISION-CODE,FD-DIVISION-NAME,
                  FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE)
      ayant cle  FD-DIVISION-CODE

rel  FD-DIVISION (FD-DIVISION-ID,FD-DIVISION-LEADER,
                  FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE)
      ayant cle  FD-DIVISION-ID

rel  FD-DIV-ACTIV (FD-FA-CODE,FD-DIVISION-ID,FD-DA-NAME,FD-DA-LEADER,
                  FD-DA-CREATION-MTH,FD-VALID-PERIOD-START-DATE,
                  FD-VALID-PERIOD-END-DATE)
      ayant cle  FD-FA-CODE,FD-DIVISION-ID

rel  FD-DA-MTH (FD-FA-CODE,FD-DIVISION-ID,FD-LAST-UPDATED,
                FD-DA-CURRENT-BUDGET,FD-DA-CUMUL-EXP,FD-DA-MTH-EXP,
                FD-DA-CUMUL-TID,FD-DA-CUMUL-OPEN-COMMIT,FD-DA-MTH-COMMIT)
      ayant cle  FD-FA-CODE,FD-DIVISION-ID,FD-LAST-UPDATED

rel  FD-PROJECT (FD-PJ-CODE,FD-PJ-REVISED,FD-PJ-NAME,FD-PJ-LEADER,
                 FD-PJ-LEADER-DIV,FD-VALID-PERIOD-START-DATE,
                 FD-VALID-PERIOD-END-DATE,FD-PERIOD-START-DATE,
                 FD-PERIOD-COMPL-DATE)
      ayant cle  FD-PJ-CODE

cdd  CDD-DA-MTH                  :FD-DA-MTH
cdd  CDD-DIV-ACT-2               :FD-DIV-ACTIV
cdd  CDD-DIV-ACTIV               :FD-DIV-ACTIV
cdd  CDD-DIV-CODE                :FD-DIV-CODE
cdd  CDD-DIVISION                :FD-DIVISION
cdd  CDD-FCTNAL-ACTIV            :FD-FCTNAL-ACTIV
cdd  CDD-PROJECT                 :FD-PROJECT
cdd  CDD-ORGANIC-UNIT            :FD-ORGANIC-UNIT
cdd  CDD-ORGUNIT-2               :FD-ORGANIC-UNIT
cdd  CDD-YEAR-TASK-ACTIVES       :FD-YEAR-TASKLIST
cdd  CDD-YEAR-TASKLI-1           :FD-YEAR-TASKLIST
cdd  CDD-YEAR-TASKLI-2           :FD-YEAR-TASKLIST
cdd  CDD-YEAR-TASKLI-3           :FD-YEAR-TASKLIST

```

```

cdd CDD-YEAR-TASKLIST          :FD-YEAR-TASKLIST
cdd CDD-YEAR-TASKLIST-1        :FD-YEAR-TASKLIST

var bool                        :BOOL
var budget-tot-da              :BUDGET
var compteur-taches-actives    :[1..5] entier
var divactiv                   :FD-DIV-ACTIV
var divcode                     :FD-DIV-CODE
var division                   :FD-DIVISION
var existant                   :BOOL
var existe-post                :BOOL
var fctnal                     :FD-FCTNAL-ACTIV
var parcours-damth            :FD-DA-MTH
var parcours-task-2           :FD-YEAR-TASKLIST
var parcours-tasklist         :FD-YEAR-TASKLIST
var pro                        :FD-PROJECT
var taskli-ex                  :FD-YEAR-TASKLIST
var tasklist                   :FD-YEAR-TASKLIST

```

SCHEMA DE PROGRAMME CREANT UNE TACHE

/* ENTREE DU CODE DE L'UNITE ORGANIQUE */

```

existant := faux
existe-post := faux
bool := faux
Tant que bool = faux faire
    Afficher ('Entrez le code organique :')
    Afficher ('1. Entrez le code de la division :')
    Repondre (tasklist.fd-division-code)
    Afficher ('2. Entrez le code de la section :')
    Repondre (tasklist.fd-section-code)
    Afficher ('Entrez la date de debut de validite de l'UNITE
              ORGANIQUE :')
    ENTRER DATE (orgunit.fd-valid-period-start-date)
    CDD-ORGUNIT-2 := vide /* CDD-ORGUNIT-2 contiendra l'Unite Organique
                          si elle existe. */
    CDD-ORGUNIT-2 := CDD-ORGANIC-UNIT filtre par
                        (FD-ORGANIC-CODE = tasklist.fd-organic-code
                        et FD-VALID-PERIOD-START-DATE =
                        orgunit.fd-valid-period-start-date)
    si CDD-ORGUNIT-2 = vide alors
        Afficher ('Il n'existe aucune Unite Organique ayant ce
                  code et cette date de debut de validite.
                  Veuillez recommencer.')
    sinon orgunit := prendre dans CDD-ORGUNIT-2
        bool := vrai
    finsi

```

Refaire

/* VERIFICATION : L'U.O NE DOIT PAS ETRE CLOSE */

```

si orgunit.fd-valid-period-end-date <> Nil et
    orgunit.fd-valid-period-end-date <= date actuelle alors
        Afficher ('Cette Unite Organique est close. Il est
                  impossible de lui associer une tache.')
        finschema
    finsi

```


/* ENTREE DU CODE DE L'ACTIVITE FONCTIONNELLE */

```
bool := faux
Tant que bool := faux faire
    Afficher ('Entrez le code de l'Activite Fonctionnelle :')
    Repondre (tasklist.fd-fa-code)
    fctnal := prendre dans CDD-FCTNAL-ACTIV avec cle =
                                                tasklist.fd-fa-code
        si fctnal = Nil alors
            Afficher ('Il n'existe aucune Activite Fonctionnelle
                        ayant un tel code.Veuillez recommencer.')
        sinon bool := vrai
    finsi
```

Refaire

/* VERIFICATION QUE L'ACTIVITE DIVISIONNAIRE EXISTE */

```
divcode := prendre dans CDD-DIV-CODE avec cle = tasklist.fd-division-code
division := prendre parent de divcode dans CDD-DIVISION
divactiv.fd-fa-code := tasklist.fd-fa-code
divactiv.fd-division-id := division.fd-division-id
Afficher ('Entrez la date de debut de validite de l'ACTIVITE DIVISIONNAIRE ')
ENTRER DATE (divactiv.fd-valid-period-start-date)
CDD-DIV-ACT-2 := vide /* CDD-DIV-ACT-2 contiendra l'Activite Divisionnaire
                        ayant cette cle et cette date de debut de validite
                        si elle existe. */
CDD-DIV-ACT-2 := CDD-DIV-ACTIV filtre par (FD-FA-CODE = divactiv.fd-fa-code
                                           et FD-DIVISION-ID = divactiv.fd-division-id
                                           et FD-VALID-PERIOD-START-DATE =
                                           divactiv.fd-valid-period-start-date)
    si CDD-DIV-ACT-2 = vide alors
        Afficher ('Il n'y a aucune Activite Divisionnaire de tel
                    code et de telle date de debut de validite.
                    Creation impossible.')
        finschema
    sinon divactiv := prendre dans CDD-DIV-ACT-2
    finsi
```

/* VERIFICATION QUE L'ACTIVITE DIVISIONNAIRE N'EST PAS CLOSE */

```
si divactiv.fd-valid-period-end-date <> Nil et
   divactiv.fd-valid-period-end-date < date actuelle alors
    Afficher ('Cette Activite Divisionnaire est close.Il est
              donc impossible de lui associer une Tache.')
    finschema
finisi
```

/* VERIFICATION QUE LE PROJET EXISTE */

```
Afficher ('Entrez le code de tache :')
Repondre (tasklist.fd-task-code)
CDD-YEAR-TASKLIST-1 := vide /* CDD-YEAR-TASKLIST-1 contiendra toutes
                             les taches ayant meme cle que celle
                             que l'on veut creer. */
CDD-YEAR-TASKLIST-1 := CDD-YEAR-TASKLIST filtre par
    (FD-ORGANIC-CODE = tasklist.fd-organic-code et
     FD-FA-CODE = tasklist.fd-fa-code et
     FD-TASK-CODE = tasklist.fd-task-code)
    si CDD-YEAR-TASKLIST-1 <> vide alors
```



```

    existant := vrai
  finsi
  pro.fd-pj-code := tasklist.fd-task-code
  pro := prendre dans CDD-PROJECT avec cle = pro.fd-pj-code
    si pro = Nil alors
      Afficher ('Il n'y a aucun Projet qui ait cette cle.
        Creation impossible.')
    finschema
  finsi

  /* VERIFICATION QUE CE PROJET NE SOIT PAS CLOS */

  si pro.fd-valid-period-end-date <> Nil
    et pro.fd-valid-period-end-date < date actuelle alors
      Afficher ('Ce Projet est clos. Il est impossible
        de lui associer une Tache.')
    finschema
  finsi

  Afficher ('Entrez le nom de la tache :')
  Repondre (tasklist.fd-task-name)
  Afficher ('Entrez la date de debut de validite de la tache :')
  ENTRER DATE (tasklist.fd-valid-period-start-date)
    si existant = vrai alors

    /* VERIFICATION QU'IL N'EXISTE PAS DEJA UNE TACHE AYANT */
    /* MEME CLE ET MEME DATE DE DEBUT DE VALIDITE */

    CDD-YEAR-TASKLI-1 := vide /* CDD-YEAR-TASKLI-1 contiendra la
      Tache ayant la meme cle et la meme
      date de debut de validite, si elle
      existe. */
    CDD-YEAR-TASKLI-1 := CDD-YEAR-TASKLIST-1 filtre par
      (FD-VALID-PERIOD-START-DATE =
        tasklist.fd-valid-period-start-date)
    si CDD-YEAR-TASKLI-1 <> vide alors
      Afficher ('Une Tache ayant meme cle et meme date
        de debut de validite existe deja.
        Creation impossible.')
    finschema
  finsi

  /* VERIFICATION QUE LA TACHE QUE L'ON VEUT CREER NE COMMENCE */
  /* PAS DURANT LA PERIODE DE VALIDITE D'UNE AUTRE TACHE */

  CDD-YEAR-TASKLI-2 := vide /* CDD-YEAR-TASKLI-2 contiendra
    la Tache valide pour cette
    periode, si elle existe. */
  CDD-YEAR-TASKLI-2 := CDD-YEAR-TASKLIST-1 filtre par
    (FD-VALID-PERIOD-START-DATE < tasklist.fd-valid-period-start-date
    et FD-VALID-PERIOD-END-DATE > tasklist.fd-valid-period-start-date)
    ou FD-VALID-PERIOD-END-DATE = Nil )
    si CDD-YEAR-TASKLI-2 <> vide alors
      Afficher ('La Tache que vous voulez creer serait
        "contenue" dans une autre Tache deja
        existence. Creation impossible')
    finschema
  finsi

  CDD-YEAR-TASKLI-3 := vide /* CDD-YEAR-TASKLI-3 contiendra
    les Taches dont la date de debut

```

```

de validee est posterieure a celle
de la Tache que l'on veut creer. */
CDD-YEAR-TASKLI-3 := CDD-YEAR-TASKLIST-1 filtre par
(FD-VALID-PERIOD-START-DATE > tasklist.fd-valid-period-start-date)
trie croissant par (FD-VALID-PERIOD-START-DATE)
si CDD-YEAR-TASKLI-3 <> vide alors
    existe-post := vrai
    taskli-ex := prendre dans CDD-TASKLI-3
finsi
finsi

/* VERIFICATION QUE LA DATE DE DEBUT DE VALIDITE DE */
/* LA TACHE SOIT POSTERIEURE OU EGALE A CELLE : */
/*      1. DE L'UNITE ORGANIQUE      */
/*      2. DE L'ACTIVITE DIVISIONNAIRE */
/*      3. DU PROJET                  */

si tasklist.fd-valid-period-start-date <
    orgunit.fd-valid-period-start-date alors
    Afficher ('La date de debut de validee de la Tache est
        anterieure a celle de l'Unite Organique.
        C'est impossible.')
    finschema
finsi
si tasklist.fd-valid-period-start-date <
    divactiv.fd-valid-period-start-date alors
    Afficher ('La date de debut de validee de la Tache est
        anterieure a celle de l'Activite Divisionnaire.
        C'est impossible.')
    finschema
finsi
si tasklist.fd-valid-period-start-date <
    pro.fd-valid-period-start-date alors
    Afficher ('La date de debut de validee de la Tache est
        anterieure a celle du Projet.
        C'est impossible.')
    finschema
finsi
reponse = ' '
Tant que reponse <> 'O' et reponse <> 'N' faire
    Afficher ('Voulez-vous rentrer la date de fin de validee
        de la Tache ? (O/N)')
    Repondre (reponse)
Refaire
si reponse = 'O' alors
    Afficher ('Entrez la date de fin de validee de la Tache :')
    ENTRER DATE (tasklist.fd-valid-period-end-date)

/* VERIFICATION QUE LA DATE DE FIN DE VALIDITE SOIT POSTERIEURE */
/* A LA DATE DE DEBUT DE VALIDITE */

si tasklist.fd-valid-period-end-date <=
    tasklist.fd-valid-period-start-date alors
    Afficher ('La date de fin da validee est anterieure a la
        date de debut de validee.C'est impossible.')
    finschema
finsi

/* VERIFICATION QUE S'IL EXISTE UNE TACHE DE MEME CLE ET DE */
/* DATE DE DEBUT DE VALIDITE POSTERIEURE, CELLE QUE L'ON VEUT */

```

/* CREER FINISSE AVANT CETTE DATE */

```

si existe-post = vrai alors
  si taskli-ex.fd-valid-period-start-date <=
    taslist.fd-valid-period-end-date alors
    Afficher ('La date de fin de validite de la Tache est
      egale ou posterieure a la date de debut de
      validite d'une Tache (de meme cle) deja
      existante.C'est impossible.')
```

finschema

finsi

finsi

/* VERIFICATION QUE LA DATE DE FIN DE VALIDITE DE LA TACHE */

/* EST ANTERIEURE A LA DATE DE FIN DE VALIDITE DE : */

```

/*          1. L'UNITE ORGANIQUE          */
/*          2. L'ACTIVITE DIVISIONNAIRE    */
/*          3. DU PROJET                    */
```

```

si orgunit.fd-valid-period-end-date <> Nil et
  tasklist.fd-valid-period-end-date >
    orgunit.fd-valid-period-end-date alors
    Afficher ('La date de fin de validite de la Tache est
      posterieure a celle de l'Unite Organique.
      C'est impossible.')
```

finschema

finsi

```

si divactiv.fd-valid-period-end-date <> Nil et
  tasklist.fd-valid-period-end-date >
    divactiv.fd-valid-period-end-date alors
    Afficher ('La date de fin de validite de la Tache est
      posterieure a celle de l'Activite Divisionnaire.
      C'est impossible.')
```

finschema

finsi

```

si pro.fd-valid-period-end-date <> Nil et
  tasklist.fd-valid-period-end-date >
    pro.fd-valid-period-end-date alors
    Afficher ('La date de fin de validite de la Tache est
      posterieure a celle du Projet.
      C'est impossible.')
```

finschema

finsi

```

sinon tasklist.fd-valid-period-end-date := Nil
finsi
```

/* VERIFICATION QU'IL N'Y A PAS DEJA 5 TACHES ACTIVES POUR */
 /* CETTE PERIODE ET CETTE UNITE ORGANIQUE */

CDD-YEAR-TASK-ACTIVES := vide /* CDD-YEAR-TASK-ACTIVES contiendra toutes
 les Taches actives (a un moment donne)
 durant la periode de validite de la Tache
 que l'on veut creer. */

```

si tasklist.fd-valid-period-end-date <> Nil alors
  CDD-YEAR-TASK-ACTIVES := CDD-YEAR-TASKLIST filtre par
    (FD-ORGANIC-CODE = tasklist.fd-organic-code et
    (FD-VALID-PERIOD-START-DATE <=
      tasklist.fd-valid-period-start-date
    et (FD-VALID-PERIOD-END-DATE >=
```

```

                                tasklist.fd-valid-period-start-date
ou FD-VALID-PERIOD-END-DATE = Nil ))
ou (FD-VALID-PERIOD-START-DATE >
                                tasklist.fd-valid-period-start-date
et FD-VALID-PERIOD-START-DATE <=
                                tasklist.fd-valid-period-end-date))

sinon
    CDD-YEAR-TASK-ACTIVES := CDD-YEAR-TASKLIST filtre par
        (FD-ORGANIC-CODE = tasklist.fd-organic-code et
        (FD-VALID-PERIOD-START-DATE <=
            tasklist.fd-valid-period-start-date
        et (FD-VALID-PERIOD-END-DATE >=
            tasklist.fd-valid-period-start-date
        ou FD-VALID-PERIOD-END-DATE = Nil ))
        ou FD-VALID-PERIOD-START-DATE >
            tasklist.fd-valid-period-start-date)

    finsi
compteur-taches-actives := 0
parcours-task-2 := premier
    Tant que parcours-task-2 < > dernier ou parcours-task-2 = dernier faire
        parcours-task-2 := prendre dans CDD-YEAR-TASK-ACTIVES
        compteur-taches-actives := compteur-taches-actives + 1
        parcours-task-2 := suivant
    Refaire
        si compteur-taches-actives = 5 alors
            Afficher ('Il y a deja 5 Taches actives pour cette periode
                qui sont associees a cette Unite Organique. Il est
                donc impossible de lui en associer une de plus
                pour cette periode.')
            finschema
        finsi

/* VERIFICATION QUE LE BUDGET TOTAL DES TACHES ASSOCIEES */
/* A CETTE ACTIVITE DIVISIONNAIRE SOIT INFERIEUR AU BUDGET */
/* DE CETTE ACTIVITE DIVISIONNAIRE */

Afficher ('Entrez le Budget de la Tache :')
Repondre (tasklist.fd-task-current-budget)
parcours-damth.fd-fa-code := divactiv.fd-fa-code
parcours-damth.fd-division-id := divactiv.fd-division-id
parcours-damth := dernier
parcours-damth := prendre dans CDD-DA-MTH
parcours-tasklist.fd-fa-code := divactiv.fd-fa-code
budget-tot-da := 0
parcours-tasklist := premier
    Tant que parcours-tasklist < > dernier ou
        parcours-tasklist = dernier faire
        parcours-tasklist := prendre dans CDD-YEAR-TASKLIST
        budget-tot-da := budget-tot-da +
            parcours-tasklist.fd-task-current-budget
        parcours-tasklist := suivant
    Refaire
        si budget-tot-da + tasklist.fd-task-current-budget >
            parcours-damth.fd-da-current-budget alors
                Afficher ('Le Budget total des Taches associees a cette
                    Activite Divisionnaire serait superieur au
                    Budget de l'Activite Divisionnaire si on lui
                    rajoutait ce Budget. C'est impossible.')
                finschema
        finsi

```

finsi

/* CREATION DE LA NOUVELLE TACHE */

tasklist est element de CDD-YEAR-TASKLIST

LIER tasklist avec orgunit

LIER tasklist avec divactiv

finschema

SCHEMA DE PROGRAMME CREANT LES FD-HIST-AP-TASK ASSOCIES A UNE UNITE ORGANIQUE

FREQUENCE ESTIMEE :

DEGRE D'IMPORTANCE :

DONNEES : ---

RESULTATS : ---

REMARQUES : Ce schema de Programme cree les taux de repartition a priori.
Cette creation doit se faire a partir de l'Unite Organique pour
valider les regles d'integrite suivantes :

1. il y a au plus 5 taux de repartition a priori par Unite Organique.
2. la somme de ces taux doit etre egale a 100 % (ce qui interdit la creation a partir des Taches).

Auparavant il verifie que :

1. L'Unite Organique existe et qu'elle ne soit pas close.
2. Le mois de validite des taux de repartition soit posterieur au mois actuel.
3. Le mois de validite des taux soit posterieur ou egal au mois de debut de validite de l'Unite Organique.
4. Le mois de validite des taux soit anterieur ou egal au mois de fin de validite de l'Unite Organique.
5. Il ne soit pas deja asocie des taux de repartition a priori a cette Unite Organique pour ce mois la.
6. Il existe une ou plusieurs Taches associees a cette Unite Organique.
7. Le nombre des taux de repartition a priori associes a cette Unite Organique soit au minimum de 1 et au maximum de 5.
8. Chaque Tache a laquelle on associe un taux de repartition a priori existe.
9. Chaque Tache a laquelle on associe un taux de repartition a priori ait une date de debut de validite anterieure ou egale a la date de debut du mois de validite de ces taux.
10. Chaque Tache a laquelle on associe un taux de repartition a priori ait une date de fin de validite posterieure ou egale a la date de fin du mois de validite de ces taux.
11. Chaque taux de repartition a priori soit compris entre 0.01 % et 100 %.
12. La somme de ces taux de repartition soit egale a 100 %.

dom CODE-ORG : article
FD-DIVISION-CODE :CODE-DIV
FD-SECTION-CODE :CODE-SEC
fin

dom DATE : article
DM :DATE-MOIS
DD :[0,99] entier
fin

dom DATE-MOIS : article
CC :[0,99] entier
YY :[0,99] entier
MM :[0,99] entier
fin

dom PBSKEY : article

FD-ORGANIC-CODE :CODE-ORG
FD-FA-CODE :CODE
FD-TASK-CODE :CODE

fin

dom BOOL :[faux,vrai]
dom CHEF :caracteres[30]
dom CODE :caracteres[4]
dom CODE-DIV :[0,99] entier
dom CODE-SEC :[0,999] entier
dom FERME :[O,C]
dom INUSE :[Y,N]
dom LEVEL :[-999,999]
dom NOM :caracteres[60]
dom POURCENT :[0,999.99] reel

cst FD-EXTRACTION-LEVEL :LEVEL
cst FD-KEY-IS-USED :INUSE
cst FD-OU-CLOSED :FERME
cst FD-OU-LEADER :CHEF
cst FD-OU-NAME :NOM
cst FD-OU-PBS-KEY :PBSKEY
cst FD-PBS-KEY-VALID-MONTH :DATE-MOIS
cst FD-RATE :POURCENT
cst FD-SIGNING-LEVEL :LEVEL
cst FD-TASK-CURRENT-BUDGET :BUDGET
cst FD-TASK-NAME :NOM
cst FD-VALID-PERIOD-END-DATE :DATE
cst FD-VALID-PERIOD-START-DATE :DATE

rel FD-ORGANIC-UNIT (FD-ORGANIC-CODE,FD-OU-NAME,FD-OU-LEADER,
FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE,
FD-EXTRACTION-LEVEL,FD-SIGNING-LEVEL,FD-OU-CLOSED)

rel FD-YEAR-TASKLIST (FD-OU-PBSKEY,FD-TASK-NAME,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE,FD-TASK-CURRENT-BUDGET)
ayant cle FD-OU-PBSKEY

rel FD-HIST-AP-TASK (FD-OU-PBSKEY,FD-PBS-KEY-VALID-MONTH,FD-RATE,
FD-KEY-IS-USED)
ayant cle FD-OU-PBS-KEY,FD-PBS-KEY-VALID-MONTH

cdd CDD-HIST-AP-TASK :FD-HIST-AP-TASK
cdd CDD-ORGANIC-UNIT :FD-ORGANIC-UNIT
cdd CDD-TASK-1 :FD-YEAR-TASKLIST
cdd CDD-YEAR-TASKLIST :FD-YEAR-TASKLIST
cdd CDD-YEAR-TASKLIST-1 :FD-YEAR-TASKLIST

var compteur :[1,10] entier
var debut-mois-de-validite :DATE
var fin-mois-de-validite :DATE
var histap :FD-HIST-AP-TASK
var mois-de-validite :DATE-MOIS
var orgunit :FD-ORGANIC-UNIT
var parcours-tasklist :FD-YEAR-TASKLIST
var reponse-numerique :[0,5] entier
var somme-des-taux :POURCENT
var tasklist : tableau [5] de FD-YEAR-TASKLIST
var taux : tableau [5] de POURCENT
var taux-deja-associe :BOOL

SCHEMA DE PROGRAMME CREANT LES FD-HIST-AP-TASK ASSOCIES A UNE UNITE ORGANIQUE

```

Afficher ('Entrez le code de l'Unite Organique :')
Repondre (orgunit.fd-organic-code)
Afficher ('Entrez la date de debut de validite de l'Unite Organique :')
ENTRER DATE (orgunit.fd-valid-period-start-date)
CDD-ORGUNIT-1 := vide /* CDD-ORGUNIT-1 contiendra l'Unite Organique ayant
                        ce code et cette date de debut de validite, si elle
                        existe. */
CDD-ORGUNIT-1 := CDD-ORGANIC-UNIT filtre par
                  (FD-ORGANIC-CODE = orgunit.fd-organic-code et
                   FD-VALID-PERIOD-START-DATE =
                     orgunit.fd-valid-period-start-date)
si CDD-ORGUNIT-1 = vide alors
  Afficher ('Il n'existe aucune Unite Organique ayant ce code
            et cette date de debut de validite.Creation des
            taux a priori impossible.')
  finschema
finsi

/* VERIFICATION QUE L'UNITE ORGANIQUE N'EST PAS CLOSE */

si orgunit.fd-valid-period-end-date <= date actuelle alors
  Afficher ('Cette Unite Organique est close.Il est impossible
            de lui associer des taux de repartition a priori.')
  finschema
finsi

Afficher ('Entrez le mois de validite des taux de repartition a priori :')
ENTRER DATE-MOIS (mois-de-validite)

/* VERIFICATION QUE CE MOIS DE VALIDITE DES TAUX FAIT BIEN */
/* PARTIE DE LA PERIODE DE VALIDITE DE L'UNITE ORGANIQUE */

si mois-de-validite <= date actuelle /* ANNEE et MOIS */ alors
  Afficher ('Ce mois de validite est anterieur ou egal au mois
            actuel.Cette creation aurait un effet retro-actif.
            C'est impossible.')
  finschema
finsi

si mois-de-validite <
  orgunit.fd-valid-valid-period-start-date[dm] alors
  Afficher ('Ce mois de validite est anterieur a la date de debut
            de validite de l'Unite Organique.C'est impossible.')
  finschema
finsi

si orgunit.fd-valid-period-end-date <> Nil et
  mois-de-validite >
  orgunit.fd-valid-period-end-date[dm] alors
  Afficher ('Ce mois de validite est posterieur a la date de
            fin de validite de l'Unite Organique.C'est
            impossible.')
  finschema
finsi

/* VERIFICATION QU'IL N'Y A AUCUN FD-HIST-AP-TASK ASSOCIE */
/* A CETTE UNITE ORGANIQUE POUR CE MOIS DE VALIDITE */

```



```

CDD-YEAR-TASKLIST-1 := vide /* CDD-YEAR-TASKLIST-1 contiendra toutes les
                                Taches associees a cette Unite Organique. */
taux-deja-associe := faux
parcours-tasklist.fd-organic-code := orgunit.fd-organic-code
parcours-tasklist := premier
    Tant que parcours-tasklist <> dernier ou
        parcours-tasklist = dernier et taux-deja-associe = faux
                                alors
        parcours-tasklist := prendre dans CDD-YEAR-TASKLIST
        parcours-histap.fd-ou-pbs-key :=
            parcours-tasklist.fd-ou-pbs-key
        parcours-histap.fd-pbs-key-valid-month := mois-de-validite
        parcours-histap := rche dans CDD-HIST-AP-TASK
            si parcours-histap <> Nil alors
                taux-deja-associe := vrai
            finsi
        parcours-tasklist est element de CDD-YEAR-TASKLIST-1
        parcours-tasklist := suivant
    Refaire
    si taux-deja-associe = vrai alors
        Afficher ('Cette Unite Organique a deja des taux de repartition
                    a priori pour ce mois de validite. Il n'est donc
                    pas possible d'en re-creeer d'autres pour le meme
                    mois.')
        finschema
    finsi
    si CDD-YEAR-TASKLIST-1 = vide alors
        Afficher ('Il n'y a aucune Tache associee a cette Unite
                    Organique. Creation des taux a priori impossible.')
        finschema
    finsi

    /* ENTREE DU NOMBRE DE TACHES SUR LESQUELLES */
    /* IL FAUDRA REPARTIR LES DEPENSES AVEC LES TAUX */
    /* A PRIORI */

reponse-numerique := 0
    Tant que reponse-numerique < 1 ou reponse-numerique > 5 faire
        Afficher ('A combien de Taches voulez-vous
                    imputer ces taux ? (1 a 5)')
        Repondre (reponse-numerique)
    Refaire

    /* ENTREE DES DIFFERENTES TACHES ET VALIDATION QUE */
    /* CES TACHES SONT ACTIVES DURANT LA PERIODE DE VALIDITE */
    /* DE CES TAUX DE REPARTITION A PRIORI */

DONNER DATE-DEBUT-FIN (mois-de-validite, debut-mois-de-validite,
                        fin-mois-de-validite)

somme-des-taux := 0
compteur := 1
    Tant que compteur <= reponse-numerique faire
        Afficher ('Entrez le code d'Activite Fonctionnelle
                    de la Tache n.', compteur, ' :')
        Repondre (tasklist[compteur].fd-fa-code)
        Afficher ('Entrez le code de Tache de la
                    Tache n.', compteur, ' :')
        Repondre (tasklist[compteur].fd-task-code)
        Afficher ('Entrez la date de debut de validite de

```

```

    la Tache n.',compteur,' :')
ENTRER DATE (tasklist[compteur].fd-valid-period-start-date)
    si tasklist[compteur].fd-valid-period-start-date >
        debut-mois-de-validite alors
        Afficher ('Cette Tache a une date de debut de
            validee posterieure au debut du mois
            de validee des taux a priori.
            C'est impossible.')
        finschema
    finsi
CDD-TASK-1 := vide /* CDD-TASK-1 contiendra la Tache
    qui correspond a ce code et a
    cette date de debut de validee,
    si elle existe. */
CDD-TASK-1 := CDD-YEAR-TASKLIST-1 filtre par
    (FD-FA-CODE = tasklist[compteur].fd-fa-code
    et FD-TASK-CODE = tasklist[compteur].fd-task-code
    et FD-VALID-PERIOD-START-DATE =
        tasklist[compteur].fd-valid-period-start-date)
si CDD-TASK-1 = vide alors
    Afficher ('Il n'existe aucune Tache ayant ce
        code et cette date de debut de
        validee.')
    finschema
sinon tasklist[compteur] := prendre dans CDD-TASK-1
finsi
si tasklist[compteur].fd-valid-period-end-date <
    Nil et
    tasklist[compteur].fd-valid-period-end-date <
        fin-mois-de-validite alors
        Afficher ('Cette Tache a une date de fin de
            validee anterieure a la fin du mois
            de validee des taux a priori.
            C'est impossible.')
        finschema
    finsi
taux[compteur] := 0
    Tant que taux[compteur] < 0.01 ou
        taux[compteur] > 100 alors
        Afficher ('Entrer le taux de repartition a
            priori (compris entre 0.01% et
            100%) :')
        Repondre (taux[compteur])
    Refaire
    somme-des-taux := somme-des-taux + taux[compteur]
    compteur := compteur + 1
Refaire

/* VERIFICATION QUE LA SOMME DES TAUX EST EGALE A 100% */

si somme-des-taux <> 100 alors
    Afficher ('La somme des taux de repartition n'est pas egale
        a 100 %.C'est impossible.')
    finschema
finsi

/* CREATION EFFECTIVE DES FD-HIST-AP-TASK */

histap.fd-pbs-key-valid-month := mois-de-validee
histap.fd-key-is-used := 'N'

```

```
compteur := 1
  Tant que compteur <= reponse-numerique faire
    histap.fd-ou-pbs-key := tasklist[compteur].fd-ou-pbs-key
    histap.fd-rate := taux[compteur]
    histap est element de CDD-HIST-AP-TASK
    LIER tasklist[compteur] avec histap
    compteur := compteur + 1
  Refaire
finschema
```

SCHEMA DE PROGRAMME MODIFIANT LES FD-HIST-AP-TASK ASSOCIES

A UNE UNITE ORGANIQUE

FREQUENCE ESTIMEE :

DEGRE D'IMPORTANCE :

DONNEES : ---

RESULTATS : ---

REMARQUES : Ce schema de Programme modifie les taux de repartition a priori.
Auparavant il verifie que :

1. L'Unite Organique existe et qu'elle ne soit pas close.
2. Le mois de validite des taux de repartition soit posterieur au mois actuel.
3. Le mois de validite des taux soit posterieur ou egal au mois de debut de validite de l'Unite Organique.
4. Le mois de validite des taux soit anterieur ou egal au mois de fin de validite de l'Unite Organique.
5. Il existe une ou plusieurs Taches associees a cette Unite Organique.
6. Il y ait deja des taux de repartition a priori associes a cette Unite Organique pour ce mois la.
7. Le nombre des taux de repartition a priori associes a cette Unite Organique soit au minimum de 1 et au maximum de 5.
8. Chaque Tache a laquelle on associe un taux de repartition a priori existe.
9. Chaque Tache a laquelle on associe un taux de repartition a priori ait une date de debut de validite anterieure ou egale a la date de debut du mois de validite de ces taux.
10. Chaque Tache a laquelle on associe un taux de repartition a priori ait une date de fin de validite posterieure ou egale a la date de fin du mois de validite de ces taux.
11. Chaque taux de repartition a priori soit compris entre 0.01 % et 100 %.
12. La somme de ces taux de repartition soit egale a 100 %.

dom CODE-ORG : article

FD-DIVISION-CODE :CODE-DIV

FD-SECTION-CODE :CODE-SEC

fin

dom DATE : article

DM :DATE-MOIS

DD :[0,99] entier

fin

dom DATE-MOIS : article

CC :[0,99] entier

YY :[0,99] entier

MM :[0,99] entier

fin

dom PBSKEY : article

FD-ORGANIC-CODE :CODE-ORG

FD-FA-CODE :CODE

FD-TASK-CODE :CODE

fin

```

var   compteur           :[1,10]  entier
var   debut-mois-de-validite :DATE
var   fin-mois-de-validite  :DATE
var   hist-a-det          :FD-HIST-AP-TASK
var   histap              :FD-HIST-AP-TASK
var   mois-de-validite     :DATE-MOIS
var   orgunit              :FD-ORGANIC-UNIT
var   parcours-tasklist    :FD-YEAR-TASKLIST
var   reponse-numerique    :[0,5]  entier
var   somme-des-taux        :POURCENT
var   tasklist             : tableau [5] de FD-YEAR-TASKLIST
var   taux                 : tableau [5] de POURCENT
var   taux-deja-associe    :BOOL

```

A UNE UNITE ORGANIQUE

```

Afficher ('Entrez le code de l'Unite Organique :')
Repondre (orgunit.fd-organic-code)
Afficher ('Entrez la date de debut de validite de l'Unite Organique :')
ENTRER DATE (orgunit.fd-valid-period-start-date)
CDD-ORGUNIT-1 := vide /* CDD-ORGUNIT-1 contiendra l'Unite Organique ayant
                        ce code et cette date de debut de validite, si elle
                        existe. */
CDD-ORGUNIT-1 := CDD-ORGANIC-UNIT filtre par
                  (FD-ORGANIC-CODE = orgunit.fd-organic-code et
                   FD-VALID-PERIOD-START-DATE =
                     orgunit.fd-valid-period-start-date)

  si CDD-ORGUNIT-1 = vide alors
    Afficher ('Il n'existe aucune Unite Organique ayant ce code
              et cette date de debut de validite.Modification des
              taux a priori impossible.')
    finschema
  finsi

/* VERIFICATION QUE L'UNITE ORGANIQUE N'EST PAS CLOSE */

  si orgunit.fd-valid-period-end-date <= date actuelle alors
    Afficher ('Cette Unite Organique est close.Il est impossible
              de lui associer des taux de repartition a priori.')
    finschema
  finsi

Afficher ('Entrez le mois de validite des taux de repartition a priori :')
ENTRER DATE-MOIS (mois-de-validite)

/* VERIFICATION QUE CE MOIS DE VALIDITE DES TAUX FAIT BIEN */
/* PARTIE DE LA PERIODE DE VALIDITE DE L'UNITE ORGANIQUE */

  si mois-de-validite <= date actuelle /* ANNEE et MOIS */ alors
    Afficher ('Ce mois de validite est anterieur ou egal au mois
              actuel.Cette modification aurait un effet
              retro-actif.C'est impossible.')
    finschema
  finsi

  si mois-de-validite <
    orgunit.fd-valid-valid-period-start-date[dm] alors
    Afficher ('Ce mois de validite est anterieur a la date de debut
              de validite de l'Unite Organique.C'est impossible.')
    finschema
  finsi
  si orgunit.fd-valid-period-end-date <> Nil et
    mois-de-validite >
    orgunit.fd-valid-period-end-date[dm] alors
    Afficher ('Ce mois de validite est posterieur a la date de
              fin de validite de l'Unite Organique.C'est
              impossible.')
    finschema
  finsi

/* VERIFICATION QU'IL Y A DES FD-HIST-AP-TASK ASSOCIES */
/* A CETTE UNITE ORGANIQUE POUR CE MOIS DE VALIDITE */

CDD-YEAR-TASKLIST-1 := vide /* CDD-YEAR-TASKLIST-1 contiendra toutes les
                             Taches associees a cette Unite Organique. */

```

```

CDD-YEAR-TASKLIST-1 := CDD-YEAR-TASKLIST filtrer par
                        (FD-ORGANIC-CODE = orgunit.fd-organic-code)

si CDD-YEAR-TASKLIST-1 = vide alors
    Afficher ('Il n'y a aucune Tache associee a cette Unite
                Organique.Modification des taux a priori
                impossible.')
    finschema
finsi
CDD-HIST-A-DET := vide /* CDD-HIST-A-DET contiendra les FD-HIST-AP-TASK
                        a detruire apres modification */
taux-deja-associe := faux
parcours-tasklist := premier
    Tant que parcours-tasklist <> dernier ou
        parcours-tasklist = dernier faire
        parcours-tasklist := prendre dans CDD-YEAR-TASKLIST-1
        parcours-histap.fd-ou-pbs-key :=
            parcours-tasklist.fd-ou-pbs-key
        parcours-histap.fd-pbs-key-valid-month := mois-de-validite
        parcours-histap := rche dans CDD-HIST-AP-TASK
        si parcours-histap <> Nil alors
            parcours-histap est element de CDD-HIST-A-DET
            taux-deja-associe := vrai
        finsi
        parcours-tasklist := suiuant
    Refaire
si taux-deja-associe = faux alors
    Afficher ('Cette Unite Organique n'a pas de taux de repartition
                a priori pour ce mois de validite.Il n'est donc
                pas possible de les modifier.')
    finschema
finsi

    /* ENTREE DU NOMBRE DE TACHES SUR LESQUELLES */
    /* IL FAUDRA REPARTIR LES DEPENSES AVEC LES TAUX */
    /* A PRIORI */

reponse-numerique := 0
    Tant que reponse-numerique < 1 ou reponse-numerique > 5 faire
        Afficher ('A combien de Taches voulez-vous
                    imputer ces taux ? (1 a 5)')
        Repondre (reponse-numerique)
    Refaire

    /* ENTREE DES DIFFERENTES TACHES ET VALIDATION QUE */
    /* CES TACHES SONT ACTIVES DURANT LA PERIODE DE VALIDITE */
    /* DE CES TAUX DE REPARTITION A PRIORI */

DONNER DATE-DEBUT-FIN (mois-de-validite,debut-mois-de-validite,
                        fin-mois-de-validite)

somme-des-taux := 0
compteur := 1
    Tant que compteur <= reponse-numerique faire
        Afficher ('Entrez le code d'Activite Fonctionnelle
                    de la Tache n.',compteur,' :')
        Repondre (tasklist[compteur].fd-fa-code)
        Afficher ('Entrez le code de Tache de la
                    Tache n.',compteur,' :')
        Repondre (tasklist[compteur].fd-task-code)
        Afficher ('Entrez la date de debut de validite de

```



```

    la Tache n.',compteur,' :')
ENTRER DATE (tasklist[compteur].fd-valid-period-start-date)
    si tasklist[compteur].fd-valid-period-start-date >
        debut-mois-de-validite alors
        Afficher ('Cette Tache a une date de debut de
            validee posterieure au debut du mois
            de validee des taux a priori.
            C'est impossible.')
        finschema
    finsi
CDD-TASK-1 := vide /* CDD-TASK-1 contiendra la Tache
    qui correspond a ce code et a
    cette date de debut de validee,
    si elle existe. */
CDD-TASK-1 := CDD-YEAR-TASKLIST-1 filtre par
    (FD-FA-CODE = tasklist[compteur].fd-fa-code
    et FD-TASK-CODE = tasklist[compteur].fd-task-code
    et FD-VALID-PERIOD-START-DATE =
        tasklist[compteur].fd-valid-period-start-date)
    si CDD-TASK-1 = vide alors
        Afficher ('Il n'existe aucune Tache ayant ce
            code et cette date de debut de
            validee.Modification impossible.')
        finschema
    sinon tasklist[compteur] := prendre dans CDD-TASK-1
    finsi
    si tasklist[compteur].fd-valid-period-end-date <
        Nil et
        tasklist[compteur].fd-valid-period-end-date <
            fin-mois-de-validite alors
            Afficher ('Cette Tache a une date de fin de
                validee anterieure a la fin du mois
                de validee des taux a priori.
                C'est impossible.')
            finschema
        finsi
    taux[compteur] := 0
    Tant que taux[compteur] < 0.01 ou
        taux[compteur] > 100 alors
        Afficher ('Entrer le taux de repartition a
            priori (compris entre 0.01% et
            100%) :')
        Repondre (taux[compteur])
        Refaire
    somme-des-taux := somme-des-taux + taux[compteur]
    compteur := compteur + 1
Refaire

/* VERIFICATION QUE LA SOMME DES TAUX EST EGALE A 100% */

    si somme-des-taux <> 100 alors
        Afficher ('La somme des taux de repartition n'est pas egale
            a 100 %.C'est impossible.')
        finschema
    finsi

    /* DESTRUCTION DES ANCIENS TAUX DE REPARTITION */
    /* A PRIORI ASSOCIES A CETTE UNITE ORGANIQUE POUR */
    /* CE MOIS DE VALIDITE */

```



```

hist-a-det.fd-organic-code := orgunit.fd-organic-code
parcours-histap := premier
    Tant que parcours-histap <> dernier ou
        parcours-histap = dernier faire
        parcours-histap := prendre dans CDD-HIST-A-DET
        hist-a-det.fd-fa-code := parcours-histap.fd-fa-code
        hist-a-det.fd-task-code := parcours-histap.fd-task-code
        hist-a-det := rche dans CDD-HIST-AP-TASK avec cle =
            hist-a-det[fd-organic-code,fd-fa-code,fd-task-code]
            SUPPRIMER hist-a-det
        parcours-histap := suivant
    Refaire

```

/* CREATION EFFECTIVE DES FD-HIST-AP-TASK */

```

histap.fd-pbs-key-valid-month := mois-de-validite
histap.fd-key-is-used := 'N'
compteur := 1
    Tant que compteur <= reponse-numerique faire
        histap.fd-ou-pbs-key := tasklist[compteur].fd-ou-pbs-key
        histap.fd-rate := taux[compteur]
        histap est element de CDD-HIST-AP-TASK
        LIER tasklist[compteur] avec histap
        compteur := compteur + 1
    Refaire
finschema

```

SCHEMA DE PROGRAMME DETRUISANT LOGIQUEMENT UNE TACHE

FREQUENCE ESTIMEE :

DEGRE D'IMPORTANCE :

DONNEES : ---

RESULTATS : ---

REMARQUES : Ce schema de programme detruit logiquement une Tache.

Auparavant il verifie :

I. QUE DE TOUTES FACONS :

1. La Tache existe.
2. Sa date de fin de validite ne soit pas anterieure a la date actuelle .
3. Sa date de fin de validite soit posterieure a sa date de debut de validite.
4. Sa date de fin de validite soit anterieure ou egale a celle :

1. de l'Activite Divisionnaire
2. de l'Unite Organique
3. du FD-PJ-OU-STATUS

qui lui sont associees.

4. Tous les taux de repartition FD-HIST-AP-TASK ne soient plus valides a cette date.
5. Tous les taux de repartition qui concernent les Commandes sont associes a des Commandes fermees.(Sinon voir II.2)

II. QUE S'IL EXISTE UNE TACHE AYANT MEME CODE :

1. S'il existe un Tache ayant meme code et de date de debut de validite posterieure,celle que l'on veut detruire finisse avant cette date.
2. Si toutes les Commandes ne sont pas closes a la date de fin de validite de la Tache et qu'elle possede un "double" qui commence exactement un jour apres sa date de fin de validite , on associe a ce "double" les cles de repartition qui etaient imputes a cette Tache.

dom CODE-ORG : article

FD-DIVISION-CODE :CODE-DIV

FD-SECTION-CODE :CODE-SEC

fin

dom DATE : article

DM :DATE-MOIS

DD :[0,99] entier

fin

dom DATE-MOIS : article

YR :YEAR

MM :[0,99] entier

fin

dom PBSKEY : article

FD-ORGANIC-CODE :CODE-ORG

FD-FA-CODE :CODE

FD-TASK-CODE :CODE

fin

dom YEAR : article

CC :[0,99] entier

YY :[0,99] entier

fin

dom BOOL :[faux,vrai]
dom BUDGET :[-999'999'999,999'999'999] entier
dom CODE :caracteres[4]
dom CODE-DIV :[0,99] entier
dom CODE-SEC :[0,999] entier
dom CHEF :caracteres[30]
dom DEPENSES :[-999'999'999.99,999'999'999.99] reel
dom FERME :[O,C]
dom INUSE :[Y,N]
dom LEVEL :[-999,999] entier
dom NOM :caracteres[60]
dom POURCENT :[0,999.99] reel
dom TYPE-OF-EXPENSE :[W,T,L,M,P,R,E,S]
dom TYPE-OF-RATE :[D,P]

cst FD-BOOKCLOSED-YEAR :YEAR
cst FD-DA-CREATION-MTH :DATE-MOIS
cst FD-DA-LEADER :CHEF
cst FD-DA-NAME :NOM
cst FD-DIVISION-ID :caracteres[3]
cst FD-EXTRACTION-LEVEL :LEVEL
cst FD-FA-CODE :CODE
cst FD-KEY-IS-USED :INUSE
cst FD-LAST-UPDATED :DATE-MOIS
cst FD-ORDER-CODE :caracteres[9]
cst FD-ORDER-CREATION-DATE :DATE
cst FD-ORGANIC-CODE :CODE-ORG
cst FD-OU-CLOSED :FERME
cst FD-OU-LEADER :CHEF
cst FD-OU-NAME :NOM
cst FD-OU-PBSKEY :PBSKEY
cst FD-PBS-KEY-VALID-MONTH :DATE-MOIS
cst FD-PERIOD-COMPL-DATE :DATE
cst FD-PERIOD-START-DATE :DATE
cst FD-PJ-CODE :CODE
cst FD-PJ-LEADER :CHEF
cst FD-PJ-LEADER-DIV :caracteres[3]
cst FD-PJ-REVISED :caractere[1]
cst FD-PROJECT-EXPENSE-TYPE :TYPE-OF-EXPENSE
cst FD-RATE :POURCENT
cst FD-RATE-TYPE :TYPE-OF-RATE
cst FD-SIGNING-LEVEL :LEVEL
cst FD-TASK-CUMUL-EXP :DEPENSES
cst FD-TASK-CUMUL-OPEN-COMMIT :DEPENSES
cst FD-TASK-CURRENT-BUDGET :BUDGET
cst FD-TASK-MTH-COMMIT :DEPENSES
cst FD-TASK-MTH-EXP :DEPENSES
cst FD-TASK-NAME :NOM
cst FD-VALID-PERIOD-END-DATE :DATE
cst FD-VALID-PERIOD-START-DATE :DATE

rel FD-YEAR-TASKLIST (FD-OU-PBSKEY,FD-TASK-NAME,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE,FD-TASK-CURRENT-BUDGET,
FD-KEY-IS-USED)

ayant cle FD-OU-PBSKEY

rel FD-ORGANIC-UNIT (FD-ORGANIC-CODE,FD-OU-NAME,FD-OU-LEADER,
FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE,

FD-EXTRACTION-LEVEL,FD-SIGNING-LEVEL,FD-OU-CLOSED)

ayant cle FD-ORGANIC-CODE

rel FD-DIV-ACTIV (FD-FA-CODE,FD-DIVISION-ID,FD-DA-NAME,FD-DA-LEADER,
FD-DA-CREATION-MTH,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE)

ayant cle FD-FA-CODE,FD-DIVISION-ID

rel FD-PROJECT (FD-PJ-CODE,FD-PJ-REVISED,FD-PJ-NAME,FD-PJ-LEADER,
FD-PJ-LEADER-DIV,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE,FD-PERIOD-START-DATE,
FD-PERIOD-COMPL-DATE)

ayant cle FD-PJ-CODE

rel FD-PJ-OU-STATUS (FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE,
FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE)

ayant cle FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE

rel FD-HIST-AP-TASK (FD-OU-PBSKEY,FD-PBS-KEY-VALID-MONTH,FD-RATE,
FD-KEY-IS-USED)

ayant cle FD-OU-PBSKEY,FD-PBS-KEY-VALID-MONTH

rel FD-ORDER-RATE (FD-OU-PBSKEY,FD-ORDER-CODE,FD-RATE-TYPE,FD-RATE)

ayant cle FD-OU-PBSKEY,FD-ORDER-CODE

rel FD-ORDER (FD-ORDER-CODE,...,FD-ORDER-CREATION-DATE,FD-BOOKCLOSED-YEAR,
...)

ayant cle FD-ORDER-CODE

<u>cdd</u> CDD-DIV-ACTIV	:FD-DIV-ACTIV
<u>cdd</u> CDD-HIST-AP-TASK	:FD-HIST-AP-TASK
<u>cdd</u> CDD-ORDER	:FD-ORDER
<u>cdd</u> CDD-ORDER-RATE	:FD-ORDER-RATE
<u>cdd</u> CDD-ORGANIC-UNIT	:FD-ORGANIC-UNIT
<u>cdd</u> CDD-PJ-OU-STATUS	:FD-PJ-OU-STATUS
<u>cdd</u> CDD-PROJECT	:FD-PROJECT
<u>cdd</u> CDD-TASK-1	:FD-YEAR-TASKLIST
<u>cdd</u> CDD-TASK-2	:FD-YEAR-TASKLIST
<u>cdd</u> CDD-YEAR-TASKLIST	:FD-YEAR-TASKLIST
<u>cdd</u> CDD-YEAR-TASKLIST-1	:FD-YEAR-TASKLIST

<u>var</u> continuite	:BOOL
<u>var</u> da	:FD-DIV-ACTIV
<u>var</u> date-fin-invalide	:BOOL
<u>var</u> debut-validite	:DATE
<u>var</u> fin-validite	:DATE
<u>var</u> mois-de-validite	:DATE-MOIS
<u>var</u> nbrjours	:ENTIER
<u>var</u> order	:FD-ORDER
<u>var</u> orgunit	:FD-ORGANIC-UNIT
<u>var</u> parcours-histap	:FD-HIST-AP-TASK
<u>var</u> parcours-rate	:FD-ORDER-RATE
<u>var</u> pro	:FD-PROJECT
<u>var</u> reponse	:caractere[1]
<u>var</u> status	:FD-PJ-OU-STATUS
<u>var</u> tasklist	:FD-YEAR-TASKLIST
<u>var</u> task-ex	:FD-YEAR-TASKLIST

SCHEMA DE PROGRAMME DETRUISANT PHYSIQUEMENT UNE TACHE

```

Afficher ('Entrez le code organique de la Tache :')
Repondre (tasklist.fd-organic-code)
Afficher ('Entrez le code d'Activite Fonctionnelle de la Tache :')
Repondre (tasklist.fd-fa-code)
Afficher ('Entrez le code de tache :')
Repondre (tasklist.fd-task-code)
CDD-YEAR-TASKLIST-1 := vide /* CDD-YEAR-TASKLIST-1 contiendra toutes les
                             Taches de meme code que celle que l'on
                             veut detruire. */
CDD-YEAR-TASKLIST-1 := CDD-YEAR-TASKLIST filtrer par
                        (FD-ORGANIC-CODE = tasklist.fd-organic-code et
                         FD-FA-CODE = tasklist.fd-fa-code et
                         FD-TASK-CODE = tasklist.fd-task-code)
Afficher ('Entrez la date de debut de validite de la Tache :')
ENTRER DATE (tasklist.fd-valid-period-start-date)

/* VERIFICATION QUE CETTE TACHE EXISTE */

CDD-TASK-1 := vide /* CDD-TASK-1 contiendra la Tache ayant cette cle et
                    cette date de debut de validite, si elle existe. */
CDD-TASK-1 := CDD-YEAR-TASKLIST-1 filtrer par (FD-VALID-PERIOD-START-DATE =
                                                tasklist.fd-valid-period-start-date)
si CDD-TASK-1 = vide alors
    Afficher ('Il n'existe aucune Tache ayant cette cle et cette
              date de debut de validite.Destruction logique
              impossible.')
    finschema
    sinon tasklist := prendre dans CDD-TASK-1
    finsi
    si tasklist.fd-valid-period-end-date <> Nil alors
        Afficher ('Cette Tache a deja une date de fin de validite.
                  C'est le :')
        AFFICHER DATE (tasklist.fd-valid-period-end-date)
        reponse := ' '
        Tant que reponse <> 'O' et reponse <> 'N' faire
            Afficher ('Voulez-vous la changer ? (O/N).')
            Repondre (reponse)
        Refaire
        si reponse = 'N' alors
            finschema
        finsi
    finsi
Afficher ('Entrez la date de fin de validite de la Tache :')
ENTRER DATE (tasklist.fd-valid-period-end-date)

/* VERIFICATION : LA DATE DE FIN DE VALIDITE DOIT ETRE */
/* POSTERIEURE OU EGALE A LA DATE ACTUELLE */

si tasklist.fd-valid-period-end-date < date actuelle alors
    Afficher ('Cette date de fin de validite est impossible
              car elle est anterieure a la date actuelle.
              La destruction aurait un effet retroactif.')
    finschema
finsi

/* VERIFICATION QUE LA DATE DE FIN DE VALIDITE SOIT POSTERIEURE */
/* A LA DATE DE DEBUT DE VALIDITE */

```

```

si tasklist.fd-valid-period-end-date <=
    tasklist.fd-valid-period-start-date alors
    Afficher ('La date de fin de validite de la Tache est anterieure
        a sa date de debut de validite.C'est impossible.')
    finschema
finsi

/* VERIFICATION QUE LA DATE DE FIN DE VALIDITE DE LA TACHE SOIT */
/* ANTERIEURE OU EGALE A CELLE : */
/* 1. DE L'ACTIVITE DIVISIONNAIRE */
/* 2. DE L'UNITE ORGANIQUE */
/* 3. DU PJ-OU-STATUS OU DU */
/* PROJET */

da := prendre parent de tasklist dans CDD-DIV-ACTIV
    si da.fd-valid-period-end-date <> Nil et
        tasklist.fd-valid-period-end-date > da.fd-valid-period-end-date
            alors
            Afficher ('La date de fin de validite de la Tache serait
                posterieure a celle de l'Activite Divisionnaire
                a laquelle elle est associee.C'est impossible.')
            finschema
        finsi

orgunit := prendre parent de tasklist dans CDD-ORGANIC-UNIT
    si orgunit.fd-valid-period-end-date <> Nil et
        tasklist.fd-valid-period-end-date >
            orgunit.fd-valid-period-end-date alors
            Afficher ('La date de fin de validite de la Tache serait
                posterieure a celle de l'Unite Organique a laquelle
                elle est associee.C'est impossible.')
            finschema
        finsi

status := prendre parent de tasklist dans CDD-PJ-OU-STATUS
    si status = Nil alors
        CDD-PJ-1 := vide /* CDD-PJ-1 contiendra le Projet auquel
            est rattachee la Tache. */
        CDD-PJ-1 := CDD-PROJECT filtre par (FD-PJ-CODE =
            tasklist.fd-task-code et
            FD-VALID-PERIOD-START-DATE <=
            tasklist.fd-valid-period-start-date et
            FD-VALID-PERIOD-END-DATE = Nil ou
            FD-VALID-PERIOD-END-DATE >
            tasklist.fd-valid-period-start-date)
        pro := prendre dans CDD-PJ-1
            si pro.fd-valid-period-end-date <> Nil et
                tasklist.fd-valid-period-end-date >
                    pro.fd-valid-period-end-date alors
                    Afficher ('La date de fin de validite de la Tache
                        serait posterieure a celle du Projet.
                        C'est impossible.')
                    finschema
                finsi
            sinon si status.fd-valid-period-end-date <> Nil et
                tasklist.fd-valid-period-end-date >
                    status.fd-valid-period-end-date alors
                    Afficher ('La date de fin de validite de la Tache
                        serait posterieure a celle du
                        FD-PJ-OU-STATUS.C'est impossible.')
                    finschema
                finsi
    
```

finsi

```
CDD-TASK-2 := vide /* CDD-TASK-2 contiendra les Taches de meme code dont
                        la date de debut de validee est posterieure a celle
                        que l'on veut detruire. */
CDD-TASK-2 := CDD-YEAR-TASK-1 filtre par (FD-VALID-PERIO-START-DATE >
                                           tasklist.fd-valid-period-start-date)
                        trie croissant par (FD-VALID-PERIOD-START-DATE)
si CDD-TASK-2 <> vide alors
    task-ex := prendre dans CDD-TASK-2

/* VERIFICATION : S'IL EXISTE UNE TACHE DE MEME CODE ET DE DATE */
/* DE DEBUT DE VALIDEE POSTERIEURE , CELLE QUE L'ON VEUT DETRUIRE */
/* FINISSE AVANT CETTE DATE */
```

```
si task-ex.fd-valid-period-start-date <=
    tasklist.fd-valid-period-end-date alors
    Afficher ('La date de fin de validee de la Tache est
                posterieure ou egale a la date de debut de validee
                d'une Tache de meme code qui existe deja.
                C'est impossible.')
```

finschema

finsi

```
/* VERIFICATION QUE TOUS LES FD-HIST-AP-TASK ASSOCIES A CETTE */
/* TACHE NE SONT PLUS VALIDES A CETTE DATE */
```

```
date-fin-invalide := faux
parcours-histap.fd-ou-pbskey := tasklist.fd-ou-pbskey
parcours-histap := premier
    Tant que parcours-histap <> dernier ou
        parcours-histap = dernier et date-fin-invalide = faux faire
        parcours-histap := prendre dans CDD-HIST-AP-TASK
        mois-de-validee := parcours-histap.fd-pbs-key-valid-month
        DONNER DATE-DEBUT-FIN(mois-de-validee,debut-validee,
                                fin-validee)
        si fin-validee > tasklist.fd-valid-period-end-date alors
            date-fin-invalide := vrai
            finsi
```

parcours-histap := suivant

Refaire

```
si date-fin-invalide = vrai alors
    Afficher ('Il existe au-moins un FD-HIST-AP-TASK (taux de
                repartition a priori) dont le mois de validee se
                termine apres la date de fin de validee de la
                Tache.C'est impossible.')
```

finschema

finsi

```
/* VERIFICATION QUE LES TAUX DE REPARTITION DIRECTS ET A PRIORI */
/* QUI CONCERNENT LES COMMANDES SONT ASSOCIES A DES COMMANDES QUI */
/* SONT CLOSES.SI CE N'EST PAS LE CAS, ET QUE CETTE TACHE AIT */
/* UN "DUPLICATE" QUI COMMENCE APRES LA DATE DE FIN DE VALIDEE */
/* VOIR SI LA DIFFERENCE ENTRE CES DEUX DATES EST DE 1 JOUR.*/
/* SI OUI ON AFFECTE CES TAUX A CE "DUPLICATE".*/
```

continue := faux


```

NBR-JOURS-ENTRE-DATES(tasklist.fd-valid-period-end-date,
                        task-ex.fd-valid-period-start-date,nbrjours)
  si nbrjours = 1 alors
    continuite := vrai
  finsi

parcours-rate.fd-ou-pbskey := tasklist.fd-ou-pbskey
parcours-rate := premier
  Tant que parcours-rate (<) dernier ou parcours-rate = dernier faire
    parcours-rate := prendre dans CDD-ORDER-RATE
    order := prendre parent de parcours-rate
    si order.fd-bookclosed-year >
      tasklist.fd-valid-period-end-date[yr] alors
      si continuite = faux alors
        Afficher ('Il existe une Commande associee a
                    cette Tache (par FD-ORDER-RATE) qui
                    se termine apres la date de fin de
                    validee cette Tache.
                    Comme de plus il n'existe
                    pas de "duplicate" de cette Tache qui
                    commence immediatement apres sa date
                    de fin de validee,il est impossible
                    de detruire logiquement cette
                    Tache.')
      finschema
    sinon
      orderate.fd-ou-pbskey := parcours-rate.fd-ou-pbskey
      orderate.fd-order-code := order.fd-order-code
      orderate.fd-rate-type := parcours-rate.fd-rate-type
      orderate.fd-rate := parcours-rate.fd-rate
      orderate est element de CDD-ORDER-RATE
      LIER orderate avec task-ex
      LIER orderate avec order
    finsi
  parcours-rate := suiuant
  Refaire
MISE A JOUR de tasklist dans CDD-YEAR-TASKLIST
finschema

```


SCHEMA DE PROGRAMME DETRUISANT PHYSIQUEMENT UNE TACHE
ET TOUS LES FD-TASK-MTH
FD-HIST-AP-TASK
FD-ORDER-RATE ASSOCIES

FREQUENCE ESTIMEE :
DEGRE D'IMPORTANCE :

DONNEES : ---

RESULTATS : ---

REMARQUES : Ce schema de programme detruit physiquement une Tache et tous les FD-TASK-MTH, FD-HIST-AP-TASK,FD-ORDER-RATE qui lui sont associes.Auparavant il verifie que :
 1. La Tache existe et que sa date de fin de validite soit anterieure a la date actuelle.

dom CODE-ORG : article
 FD-DIVISION-CODE :CODE-DIV
 FD-SECTION-CODE :CODE-SEC
 fin

dom DATE : article
 CC :[0,99] entier
 YY :[0,99] entier
 MM :[0,99] entier
 DD :[0,99] entier
 fin

dom DATE-MOIS : article
 CC :[0,99] entier
 YY :[0,99] entier
 MM :[0,99] entier
 fin

dom PBSKEY : article
 FD-ORGANIC-CODE :CODE-ORG
 FD-FA-CODE :CODE
 FD-TASK-CODE :CODE
 fin

dom BOOL :[faux,vrai]
dom BUDGET :[-999'999'999,999'999'999] entier
dom CODE :caracteres[4]
dom CODE-DIV :[0,99] entier
dom CODE-SEC :[0,999] entier
dom DEPENSES :[-999'999'999.99,999'999'999.99] reel
dom INUSE :[Y,N]
dom NOM :caracteres[60]
dom POURCENT :[0,999.99] reel
dom TYPE-OF-RATE :[D,P]

cst FD-KEY-IS-USED :INUSE
cst FD-LAST-UPDATED :DATE-MOIS
cst FD-ORDER-CODE :caracteres[9]
cst FD-OU-PBSKEY :PBSKEY
cst FD-PBS-KEY-VALID-MONTH :DATE-MOIS
cst FD-RATE :POURCENT
cst FD-RATE-TYPE :TYPE-OF-RATE

```

cst  FD-TASK-CUMUL-EXP           :DEPENSES
cst  FD-TASK-CUMUL-OPEN-COMMIT   :DEPENSES
cst  FD-TASK-CURRENT-BUDGET      :BUDGET
cst  FD-TASK-MTH-COMMIT          :DEPENSES
cst  FD-TASK-MTH-EXP             :DEPENSES
cst  FD-TASK-NAME                :NOM
cst  FD-VALID-PERIOD-END-DATE    :DATE
cst  FD-VALID-PERIOD-START-DATE  :DATE

rel  FD-YEAR-TASKLIST (FD-OU-PBSKEY,FD-TASK-NAME,FD-VALID-PERIOD-START-DATE,
                      FD-VALID-PERIOD-END-DATE,FD-TASK-CURRENT-BUDGET,
                      FD-KEY-IS-USED)
      ayant cle  FD-OU-PBSKEY

rel  FD-HIST-AP-TASK (FD-OU-PBSKEY,FD-PBS-KEY-VALID-MONTH,FD-RATE,
                      FD-KEY-IS-USED)
      ayant cle  FD-OU-PBSKEY,FD-PBS-KEY-VALID-MONTH

rel  FD-TASK-MTH (FD-OU-PBSKEY,FD-LAST-UPDATED,FD-TASK-CUMUL-EXP,
                  FD-TASK-MTH-EXP,FD-TASK-CUMUL-OPEN-COMMIT,
                  FD-TASK-MTH-COMMIT)
      ayant cle  FD-OU-PBSKEY,FD-LAST-UPDATED

rel  FD-ORDER-RATE (FD-OU-PBSKEY,FD-ORDER-CODE,FD-RATE-TYPE,FD-RATE)
      ayant cle  FD-OU-PBSKEY,FD-ORDER-CODE

cdd  CDD-YEAR-TASKLIST           :FD-YEAR-TASKLIST
cdd  CDD-HIST-AP-TASK            :FD-HIST-AP-TASK
cdd  CDD-TASK-1                  :FD-YEAR-TASKLIST
cdd  CDD-TASK-MTH                :FD-TASK-MTH
cdd  CDD-ORDER-RATE              :FD-ORDER-RATE

var  parcours-histap             :FD-HIST-AP-TASK
var  parcours-orderate           :FD-ORDER-RATE
var  parcours-taskmth            :FD-TASK-MTH
var  tasklist                    :FD-YEAR-TASKLIST

```

SCHEMA DE PROGRAMME DETRUISANT PHYSIQUEMENT UNE TACHE

```

Afficher ('Entrez le code organique de la Tache :')
Repondre (tasklist.fd-organic-code)
Afficher ('Entrez le code d'Activite Fonctionnelle de la Tache :')
Repondre (tasklist.fd-fa-code)
Afficher ('Entrez le code de tache :')
Repondre (tasklist.fd-task-code)
Afficher ('Entrez la date de debut de validite de la Tache :')
ENTRER DATE (tasklist.fd-valid-period-start-date)

/* VERIFICATION QUE CETTE TACHE EXISTE */

CDD-TASK-1 := vide /* CDD-TASK-1 contiendra la Tache ayant cette cle et
                  cette date de debut de validite, si elle existe. */
CDD-TASK-1 := CDD-YEAR-TASKLIST filtre par
              (FD-ORGANIC-CODE = tasklist.fd-organic-code et
               FD-FA-CODE = tasklist.fd-fa-code et
               FD-TASK-CODE = tasklist.fd-task-code et
               FD-VALID-PERIOD-START-DATE =
                  tasklist.fd-valid-period-start-date)

si CDD-TASK-1 = vide alors
    Afficher ('Il n'existe aucune Tache ayant cette cle et cette

```

date de debut de valideite.Destruction physique impossible.')

finschema
sinon tasklist := prendre dans CDD-TASK-1
finsi

/* VERIFICATION QUE CETTE TACHE EST DEJA LOGIQUEMENT DETRUITE */

si tasklist.fd-valid-period-end-date = Nil ou
tasklist.fd-valid-period-end-date >= date actuelle alors
Afficher ('Cette Tache n'a pas ete logiquement detruite.
Il est donc impossible de la detruire physiquement.')

finschema
finsi

/* DESTRUCTION PHYSIQUE PERMISE */

/* DESTRUCTION DE TOUS LES TASK-MTH ASSOCIES */

parcours-taskmth.fd-ou-pbskey := tasklist.fd-ou-pbskey
parcours-taskmth := premier
Tant que parcours-taskmth <> dernier ou
parcours-taskmth = dernier faire
parcours-taskmth := prendre dans CDD-TASK-MTH
SUPPRIMER parcours-taskmth
parcours-taskmth := suivant
Refaire

/* DESTRUCTION DE TOUS LES HIST-AP-TASK ASSOCIES */

parcours-histap.fd-ou-pbskey := tasklist.fd-ou-pbskey
parcours-histap := premier
Tant que parcours-histap <> dernier ou
parcours-histap = dernier faire
parcours-histap := prendre dans CDD-HIST-AP-TASK
SUPPRIMER parcours-histap
parcours-histap := suivant
Refaire

/* DESTRUCTION DE TOUS LES ORDER-RATE ASSOCIES */

parcours-orderate.fd-ou-pbskey := tasklist.fd-ou-pbskey
parcours-orderate := premier
Tant que parcours-orderate <> dernier ou
parcours-orderate = dernier faire
parcours-orderate := prendre dans CDD-ORDER-RATE
SUPPRIMER parcours-orderate
parcours-orderate := suivant
Refaire

SUPPRIMER tasklist

finschema

BIBLIOTHEQUE DES PROCEDURESPROCEDURE ENTRER DATEFREQUENCE ESTIMEE :DEGRE D'IMPORTANCE :DONNEES : ---RESULTATS :var da :DATE

REMARQUES : Cette Procedure permet l'entree d'une date.Elle verifie que cette date ne puisse pas etre impossible (cas du 30 Fevrier, etc.).

Remarque : le siecle est toujours egal a 19.

var limite :[28,31] entiervar siecle :[19,20] entier

limite := 31

siecle := 19

da.cc := siecle

Afficher ('Rentrez l'annee :')Repondre (da.yy)Afficher ('Rentrez le mois :')Repondre (da.mm)Tant que da.mm<1 ou da.mm>12 faire

Afficher ('Mois invalide.Le mois doit etre compris entre 1 et 12')

Repondre (da.mm)RefaireAfficher ('Rentrez le jour :')Repondre (da.dd)si da.mm=4 ou da.mm=6 ou da.mm=9 ou da.mm=11alors limite:=30finsisi da.mm=2 et (da.yy/4 = INT(da.yy/4))alors limite:=29sinon si da.mm=2alors limite:=28finsiTant que da.dd <1 ou da.dd > limite faireAfficher (da.mm;'a au plus ';limite;' jours')Repondre (da.dd)RefairefinprocedurePROCEDURE ENTRER DATE-MOIS (datem :DATE-MOIS)FREQUENCE ESTIMEE :DEGRE D'IMPORTANCE :DONNEES : ---RESULTATS :var datem :DATE-MOIS

REMARQUES :

var siecle :[19,20] entier

siecle := 19

date.m.cc := 19

Afficher ('Entrez l'annee :')

Repondre (date.m.yy)

Afficher ('Entrez le mois :')

Repondre (date.m.mm)

Tant que date.m.mm <1 ou date.m.mm >12 faire

Afficher ('Le mois doit etre compris entre 1 et 12')

Afficher ('Entrez le mois :')

Repondre (date.m.mm)

Refaire

finprocedure

PROCEDURE AFFICHER DATE

FREQUENCE ESTIMEE :

DEGRE D'IMPORTANCE :

DONNEES :

var da :DATE

RESULTATS : ---

REMARQUES : Cette Procedure affiche simplement une date que le Programme principal lui fourni.

Afficher (da.dd, '-' ,da.mm, '-' ,da.cc;da.yy)

finprocedure

PROCEDURE VALIDER DATE DEBUT-FIN

FREQUENCE ESTIMEE :

DEGRE D'IMPORTANCE :

DONNEES : ---

RESULTATS :

var dal :DATE

var da2 :DATE

REMARQUES : Cette Procedure valide la regle d'integrite qui veut que la date de debut de validite d'un enregistrement soit anterieure a sa date de fin.

var reponse :[N,0]

var permis :[faux,vrai]

permis := faux

Tant que permis = faux faire

```

Afficher ('Entrez la date de debut de valideite :')
ENTRER DATE (dal)
Afficher ('Voulez-vous entrer la date de fin de valideite ? (O/N)')
Repondre ( reponse )
    si reponse = 'O' alors
        Afficher ('Entrez la date de fin de valideite :')
        ENTRER DATE (da2)
        si dal >= da2 alors
            Afficher ('La date de debut de valideite
                        est posterieure a la date de fin
                        de valideite.C'est impossible.
                        Veuillez recommencer.')
        sinon permis := vrai
        finsi
    sinon si reponse = 'N' alors
        da2 := Nil
        permis := vrai
    sinon
        Afficher ('Reponse incorrecte.Veuillez repondre
                    par O (oui) ou par N (non)')
    finsi
finsi

```

Refaire

finprocedure

PROCEDURE DONNER DATE-DEBUT-FIN

FREQUENCE ESTIMEE :
DEGRE D'IMPORTANCE :

DONNEES :

var datem :DATE-MOIS

RESULTATS :

var dal :DATE
var da2 :DATE

REMARQUES : Cette Procedure donne pour un mois donne (datem) sa date de
 debut (dal) et sa date de fin (da2)

var limite :[1,31] entier

limite := 31
 dal.dm := datem
 dal.dd := 1

```

si datem.mm = 4 ou datem.mm = 6 ou datem.mm = 9
    ou datem.mm = 11 alors
        limite := 30
    sinon si da.mm = 2 et (da.yy/4 = INT(da.yy/4) alors
        limite := 29
        sinon si da.mm = 2 alors
            limite := 28
        finsi
    finsi
finsi

```

da2.dm := datem
da2.dd := limite

finprocedure

PROCEDURE NBR-JOURS-ENTRE-DATES

FREQUENCE ESTIMEE :
DEGRE D'IMPORTANCE :

DONNEES :

var dal :DATE
var da2 :DATE

RESULTATS :

var nbr :entier

REMARQUES :

var diff1 :entier
var diff2 :entier
var diff3 :entier
var jbis :entier
var nbrjr :entier
var compteur :entier
var totjr :entier

/* REMARQUE : CETTE PROCEDURE UTILISE DEUX FONCTIONS :*/
/* INT(x) DONNE LA PARTIE ENTIERE D'UN NOMBRE */
/* SGN(x) DONNE LE SIGNE D'UN NOMBRE (-1 ou 1) */

diff1 := da2.yy - dal.yy

/* COMPTEUR D'ANNEES BISEXTILES */

si diff1 <> 0 alors
 compteur := 0
 Tant que compteur <= diff1 faire
 annee := dal.yy + compteur
 si annee/4 = INT(annee/4) alors
 jbis := jbis + 1
 finsi
 compteur := compteur + 1
 Refaire
 si dal.yy/4 = INT(dal.yy/4) et dal.mm > 2 ou
 da2.yy/4 = INT(da2.yy/4) et da2.mm <= 2 alors
 jbis := jbis - 1
 finsi
finsi

diff2 := da2.mm - dal.mm

/* COMPTEUR DU NOMBRE DE JOURS (POUR LA DIFF. ENTRE LES MOIS) */

si diff2 <> 0 alors
 compteur := 1
 mois := dal.mm


```

an := dal.yy
  Tant que compteur <= diff2 faire
    nbrjr := 31
    si mois = 4 ou mois = 6 ou mois = 9 ou
      mois = 11 alors
        nbrjr := 30
    finsi
    si mois = 2 et an/4 = INT(an/4) alors
      nbrjr := 29
    sinon si mois = 2 alors
      nbrjr := 28
    finsi
  finsi
  totjr := totjr + nbrjr
  mois := mois + 1
  si mois > 12 alors
    mois := 1
    an := an + 1
  finsi
  compteur := compteur + 1
Refaire
finsi

```

/* CALCUL DE LA DIFFERENCE ENTRE LES DEUX JOURS DES DATES */

diff3 := da2.dd - dal.dd

/* CALCUL DE LA DIFFERENCE TOTALE DE JOURS ENTRE CES */
/* DATES */

nbr := diff1 * 365 + jbis + SGN(diff2) * totjr + diff3

finprocedure

PROCEDURE DETRUIRE TACHE

FREQUENCE ESTIMEE :
DEGRE D'IMPORTANCE :

DONNEES :

var tasklist :FD-YEAR-TASKLIST

RESULTATS : ---

REMARQUES : Cette Procedure detruit une Tache ,FD-YEAR-TASKLIST, ainsi que tous les FD-TASK-MTH,FD-HIST-AP-TASK,FD-ORDER-RATE qui lui etaient associes.Les verifications ont du etre faites avant l'appel a la Procedure.

dom CODE-ORG : article

FD-DIVISION-CODE :CODE-DIV

FD-SECTION-CODE :CODE-SEC

fin

dom DATE : article

CC :[0,99] entier

YY :[0,99] entier

MM :[0,99] entier

DD :[0,99] entier

fin

dom DATE-MOIS : article

CC :[0,99] entier

YY :[0,99] entier

MM :[0,99] entier

fin

dom PBSKEY : article

FD-ORGANIC-CODE :CODE-ORG

FD-FA-CODE :CODE

FD-TASK-CODE :CODE

fin

dom CODE :caracteres[4]

dom CODE-DIV :[0,99] entier

dom CODE-SEC :[0,999] entier

dom DEPENSES :[-999'999'999.99,999'999'999.99] reel

dom INUSE :[Y,N]

dom POURCENT :[0,999.99] reel

dom TYPE-OF-RATE :[D,P]

cst FD-KEY-IS-USED :INUSE

cst FD-LAST-UPDATED :DATE-MOIS

cst FD-ORDER-CODE :caracteres[9]

cst FD-OU-PBSKEY :PBSKEY

cst FD-PBS-KEY-VALID-MONTH :DATE-MOIS

cst FD-RATE :POURCENT

cst FD-RATE-TYPE :TYPE-OF-RATE

cst FD-TASK-CUMUL-EXP :DEPENSES

cst FD-TASK-CUMUL-OPEN-COMMIT :DEPENSES

cst FD-TASK-MTH-COMMIT :DEPENSES

cst FD-TASK-MTH-EXP :DEPENSES

cst FD-VALID-PERIOD-END-DATE :DATE

cst FD-VALID-PERIOD-START-DATE :DATE

rel FD-HIST-AP-TASK (FD-OU-PBSKEY,FD-PBS-KEY-VALID-MONTH,FD-RATE,
FD-KEY-IS-USED)

ayant cle FD-OU-PBSKEY,FD-PBS-KEY-VALID-MONTH

rel FD-TASK-MTH (FD-OU-PBSKEY,FD-LAST-UPDATED,FD-TASK-CUMUL-EXP,
FD-TASK-MTH-EXP,FD-TASK-CUMUL-OPEN-COMMIT,
FD-TASK-MTH-COMMIT)

ayant cle FD-OU-PBSKEY,FD-LAST-UPDATED

rel FD-ORDER-RATE (FD-OU-PBSKEY,FD-ORDER-CODE,FD-RATE-TYPE,FD-RATE)

ayant cle FD-OU-PBSKEY,FD-ORDER-CODE

cdd CDD-HIST-AP-TASK :FD-HIST-AP-TASK

cdd CDD-TASK-MTH :FD-TASK-MTH

cdd CDD-ORDER-RATE :FD-ORDER-RATE

var parcoures-histap :FD-HIST-AP-TASK

var parcoures-orderate :FD-ORDER-RATE

var parcoures-taskmth :FD-TASK-MTH

/* DESTRUCTION DE TOUS LES TASK-MTH ASSOCIES */

parcoures-taskmth.fd-ou-pbskey := tasklist.fd-ou-pbskey

```
parcours-taskmth := premier
  Tant que parcours-taskmth <> dernier ou
    parcours-taskmth = dernier faire
    parcours-taskmth := prendre dans CDD-TASK-MTH
    SUPPRIMER parcours-taskmth
    parcours-taskmth := suivant
  Refaire

  /* DESTRUCTION DE TOUS LES HIST-AP-TASK ASSOCIES */

parcours-histap.fd-ou-pbskey := tasklist.fd-ou-pbskey
parcours-histap := premier
  Tant que parcours-histap <> dernier ou
    parcours-histap = dernier faire
    parcours-histap := prendre dans CDD-HIST-AP-TASK
    SUPPRIMER parcours-histap
    parcours-histap := suivant
  Refaire

  /* DESTRUCTION DE TOUS LES ORDER-RATE ASSOCIES */

parcours-orderate.fd-ou-pbskey := tasklist.fd-ou-pbskey
parcours-orderate := premier
  Tant que parcours-orderate <> dernier ou
    parcours-orderate = dernier faire
    parcours-orderate := prendre dans CDD-ORDER-RATE
    SUPPRIMER parcours-orderate
    parcours-orderate := suivant
  Refaire

SUPPRIMER tasklist

finprocedure
```

PROCEDURE REPARTITION DEPUIS TACHE

FREQUENCE ESTIMEE :
DEGRE D'IMPORTANCE :

DONNEES :

var tasklist :FD-YEAR-TASKLIST
var exty :FD-EXPENSE-TYPE
var mvt :FD-MVT-CODE
var somme :DEPENDENSES

RESULTATS :

code-erreur := 1 /* si le Project-State est clos */
0 /* si tout s'est bien passe */

REMARQUES : Ce schema de Programme repartit une somme depuis une Tache sur :

1. 1e FD-TASK-MTH
2. 1e FD-FA-MTH
3. 1e FD-DA-MTH
4. 1e FD-YR-OU-COST
5. 1e FD-YR-PJ-INFO

Remarque : si ces enregistrements n'existent pas ils sont
crees automatiquement par la Procedure.

dom CODE-ORG : article
FD-DIVISION-CODE :CODE-DIV
FD-SECTION-CODE :CODE-SEC
fin

dom DATE : article
CC :[0,99] entier
YY :[0,99] entier
MM :[0,99] entier
DD :[0,99] entier
fin

dom PBSKEY : article
FD-ORGANIC-CODE :CODE-ORG
FD-FA-CODE :CODE
FD-TASK-CODE :CODE
fin

dom BOOL :[faux,vrai]
dom BUDGET :[-999'999'999, 999'999'999] entier
dom CAR-UN :caractere[1]
dom CHEF :caracteres[30]
dom CODE :caracteres[4]
dom CODE-DIV :[00,99] entier
dom CODE-SEC :[000,999] entier
dom DEPENDENSES :[-999'999'999.99,999'999'999.99] reel
dom NOM :caracteres[60]
dom TYPE-DE-DEPENDENSE :[W,T,L,M,P,R,E,S]
dom TYPE-DE-STATUS :[A,E,S]

cst FD-AMOUNT-STATUS :TYPE-DE-STATUS
cst FD-DA-CREATION-MTH :DATE-MOIS
cst FD-DA-CUMUL-EXP :DEPENDENSES
cst FD-DA-CUMUL-OPEN-COMMIT :DEPENDENSES
cst FD-DA-CUMUL-TID :DEPENDENSES
cst FD-DA-CURRENT-BUDGET :BUDGET

<u>cst</u>	FD-DA-LEADER	:CHEF
<u>cst</u>	FD-DA-MTH-COMMIT	:DEPENDSES
<u>cst</u>	FD-DA-MTH-EXP	:DEPENDSES
<u>cst</u>	FD-DA-NAME	:NOM
<u>cst</u>	FD-DIVISION-ID	:caracteres[3]
<u>cst</u>	FD-FA-CODE	:CODE
<u>cst</u>	FD-FA-CUMUL-OPEN-COMMIT	:DEPENDSES
<u>cst</u>	FD-FA-CUMUL-TID	:DEPENDSES
<u>cst</u>	FD-FA-CURRENT-BUDGET	:BUDGET
<u>cst</u>	FD-FA-MTH-COMMIT	:DEPENDSES
<u>cst</u>	FD-FA-MTH-EXP	:DEPENDSES
<u>cst</u>	FD-FA-NAME	:NOM
<u>cst</u>	FD-FA-YEAR-EXP	:DEPENDSES
<u>cst</u>	FD-OU-MATL-COST	: <u>tableau</u> [4] de DEPENDSES
<u>cst</u>	FD-OU-PBSKEY	:PBSKEY
<u>cst</u>	FD-PERIOD-COMPL-DATE	:DATE
<u>cst</u>	FD-PERIOD-START-DATE	:DATE
<u>cst</u>	FD-PJ-CODE	:CODE
<u>cst</u>	FD-PJ-LAST-MANMTH-SMC-REV-DATE	:DATE
<u>cst</u>	FD-PJ-LAST-MANMTH-TL-REV-DATE	:DATE
<u>cst</u>	FD-PJ-LAST-MANYR-REV-DATE	:DATE
<u>cst</u>	FD-PJ-LAST-MATL-COST-REV-DATE	:DATE
<u>cst</u>	FD-PJ-LEADER	:CHEF
<u>cst</u>	FD-PJ-LEADER-DIV	:caracteres[3]
<u>cst</u>	FD-PJ-MANMTH-SMC	: <u>tableau</u> [10] de DEPENDSES
<u>cst</u>	FD-PJ-MANMTH-TL	: <u>tableau</u> [10] de DEPENDSES
<u>cst</u>	FD-PJ-MANYR	: <u>tableau</u> [10] de DEPENDSES
<u>cst</u>	FD-PJ-MATL-COST	: <u>tableau</u> [10] de DEPENDSES
<u>cst</u>	FD-PJ-NAME	:NOM
<u>cst</u>	FD-PJ-REVISED	:CAR-UN
<u>cst</u>	FD-PROJECT-EXPENSE-TYPE	:TYPE-DE-DEPENDSE
<u>cst</u>	FD-TASK-CUMUL-EXP	:DEPENDSES
<u>cst</u>	FD-TASK-CUMUL-OPEN-COMIT	:DEPENDSES
<u>cst</u>	FD-TASK-MTH-COMMIT	:DEPENDSES
<u>cst</u>	FD-TASK-MTH-EXP	:DEPENDSES
<u>cst</u>	FD-VALID-PERIOD-END-DATE	:DATE <u>et Nil</u>
<u>cst</u>	FD-VALID-PERIOD-START-DATE	:DATE

rel FD-TASK-MTH (FD-OU-PBSKEY,FD-LAST-UPDATED,FD-TASK-CUMUL-EXP,
FD-TASK-MTH-EXP,FD-TASK-CUMUL-OPEN-COMMIT,
FD-TASK-MTH-COMMIT)
ayant cle FD-OU-PBSKEY,FDLAST-UPDATED

rel FD-FCTNAL-ACTIV (FD-FA-CODE,FD-FA-NAME,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE)
ayant cle FD-FA-CODE

rel FD-FA-MTH (FD-FA-CODE,FD-LAST-UPDATED,FD-FA-CURRENT-BUDGET,
FD-FA-YEAR-EXP,FD-FA-MTH-EXP,FD-FA-CUMUL-TID,
FD-FA-CUMUL-OPEN-COMMIT,FD-FA-MTH-COMMIT)
ayant cle FD-FA-CODE,FD-LAST-UPDATED

rel FD-DIV-ACTIV (FD-FA-CODE,FD-DIVISION-ID,FD-DA-NAME,FD-DA-LEADER,
FD-DA-CREATION-MTH,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE)
ayant cle FD-FA-CODE,FD-DIVISION-ID

rel FD-DA-MTH (FD-FA-CODE,FD-DIVISION-ID,FD-LAST-UPDATED,FD-DA-CUMUL-EXP,
FD-DA-CURRENT-BUDGET,FD-DA-MTH-EXP,FD-DA-CUMUL-TID,

FD-DA-CUMUL-OPEN-COMMIT,FD-DA-MTH-COMMIT)

ayant cle FD-FA-CODE,FD-DIVISION-ID,FD-LAST-UPDATED

rel FD-PROJECT (FD-PJ-CODE,FD-PJ-REVISED,FD-PJ-NAME,FD-PJ-LEADER,
FD-PJ-LEADER-DIV,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE,FD-PERIOD-START-DATE,
FD-PERIOD-COMPL-DATE)

ayant cle FD-PJ-CODE

rel FD-PROJECT-STATE (FD-PJ-CODE,FD-AMOUNT-STATUS,FD-VALID-PERIOD-START-DATE,
FD-VALID-PERIOD-END-DATE)

ayant cle FD-PJ-CODE,FD-TYPE

rel FD-YR-PJ-INFO (FD-PJ-CODE,FD-TYPE,FD-AMOUNT-STATUS,FD-LAST-UPDATED,
FD-PJ-LAST-MATL-COST-REV-DATE,
FD-PJ-MATL-COST,FD-PJ-LAST-MANYR-REV-DATE,
FD-PJ-MANYR,FD-PJ-LAST-MANMTH-TL-REV-DATE,
FD-PJ-MANMTH-TL,FD-PJ-LAST-MANMTH-SMC-REV-DATE,
FD-PJ-MANMTH-SMC)

ayant cle FD-PJ-CODE,FD-TYPE,FD-AMOUNT-STATUS,FD-LAST-UPDATED

rel FD-PJ-OU-STATUS (FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE,
FD-VALID-PERIOD-START-DATE,FD-VALID-PERIOD-END-DATE)

ayant cle FD-PJ-CODE,FD-ORGANIC-CODE

rel FD-YR-OU-COST (FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE,
FD-LAST-UPDATED,FD-OU-MATL-COST)

ayant cle FD-PJ-CODE,FD-ORGANIC-CODE,FD-PROJECT-EXPENSE-TYPE,
FD-LAST-UPDATED

<u>cdd</u>	CDD-DA-MTH	:FD-DA-MTH
<u>cdd</u>	CDD-DIV-ACTIV	:FD-DIV-ACTIV
<u>cdd</u>	CDD-FA-MTH	:FD-FA-MTH
<u>cdd</u>	CDD-FCTNAL-ACTIV	:FD-FCTNAL-ACTIV
<u>cdd</u>	CDD-PJ-OU-STATUS	:FD-PJ-OU-STATUS
<u>cdd</u>	CDD-PROJECT	:FD-PROJECT
<u>cdd</u>	CDD-PROJECT-STATE	:FD-PROJECT-STATE
<u>cdd</u>	CDD-TASK-MTH	:FD-TASK-MTH
<u>cdd</u>	CDD-YR-OU-COST	:FD-YR-OU-COST
<u>cdd</u>	CDD-YR-PJ-INFO	:FD-YR-PJ-INFO

<u>var</u>	code-erreur	:BOOL
<u>var</u>	damth	:FD-DA-MTH
<u>var</u>	divactiv	:FD-DIV-ACTIV
<u>var</u>	famth	:FD-FA-MTH
<u>var</u>	fctnal	:FD-FCTNAL-ACTIV
<u>var</u>	parcours-damth	:FD-DA-MTH
<u>var</u>	parcours-famth	:FD-FA-MTH
<u>var</u>	parcours-taskmth	:FD-TASK-MTH
<u>var</u>	parcours-yroucost	:FD-YR-OU-COST
<u>var</u>	parcours-yrpj-info	:FD-YR-PJ-INFO
<u>var</u>	pro	:FD-PROJECT
<u>var</u>	state	:FD-PROJECT-STATE
<u>var</u>	status	:FD-PJ-OU-STATUS
<u>var</u>	tasmth	:FD-TASK-MTH
<u>var</u>	yroucost	:FD-YR-OU-COST
<u>var</u>	yrpj-info	:FD-YR-PJ-INFO

Debut Procedure

```

code-erreur := 0
taskmth.fd-organic-code := tasklist.fd-organic-code
taskmth.fd-fa-code := tasklist.fd-fa-code
taskmth.fd-task-code := tasklist.fd-task-code
taskmth.fd-last-updated := date actuelle /* ANNEE et MOIS */
taskmth := rche dans CDD-TASK-MTH avec cle =
    taskmth[fd-organic-code,fd-fa-code,fd-task-code,fd-last-updated]
    si taskmth = Nil alors

        /* CREATION DU TASK-MTH S'IL N'EXISTE PAS */

        parcours-taskmth.fd-organic-code := taskmth.fd-organic-code
        parcours-taskmth.fd-fa-code := taskmth.fd-fa-code
        parcours-taskmth.fd-task-code := taskmth.fd-task-code
        parcours-taskmth := dernier /* On veut trouver le dernier
                                         TASK-MTH qui ait ete associe
                                         a cette Tache. */
        parcours-taskmth := rche dans CDD-TASK-MTH
        taskmth.fd-task-cumul-exp := parcours-taskmth.fd-task-cumul-exp
        taskmth.fd-task-cumul-open-commit :=
            parcours-taskmth.fd-task-cumul-open-commit
        taskmth.fd-task-mth-commit := 0
        taskmth.fd-task-mth-exp := 0
        taskmth est element de CDD-TASK-MTH
        LIER tasklist avec taskmth
    finsi

    si mvt = 530 ou mvt = 550 alors
        taskmth.fd-task-cumul-open-commit :=
            taskmth.fd-task-cumul-open-commit + somme
        taskmth.fd-task-mth-commit := taskmth.fd-task-mth-commit + somme
    sinon
        taskmth.fd-task-cumul-exp := taskmth.fd-task-cumul-exp + somme
        taskmth.fd-task-mth-exp := taskmth.fd-task-mth-exp + somme
    finsi
MISE A JOUR de taskmth dans CDD-TASK-MTH

fctnal.fd-fa-code := tasklist.fd-fa-code
fctnal := prendre dans CDD-FCTNAL-ACTIV avec cle = fctnal.fd-fa-code
famth.fd-fa-code := tasklist.fd-fa-code
famth.fd-last-updated := date actuelle /* ANNEE et MOIS */
famth := rche dans CDD-FA-MTH avec cle = famth[fd-fa-code,fd-last-updated]
    si famth = Nil alors

        /* CREATION DU FA-MTH S'IL N'EXISTE PAS */

        parcours-famth.fd-fa-code := fctnal.fd-fa-code
        parcours-famth := dernier /* On veut trouver le dernier
                                         FA-MTH qui ait ete associe a cette
                                         Activite Fonctionnelle. */
        parcours-famth := rche dans CDD-FA-MTH
        famth.fd-fa-current-budget := parcours-famth.fd-fa-current-budget
        famth.fd-fa-year-exp := parcours-famth.fd-fa-year-exp
        famth.fd-fa-cumul-tid := parcours-famth.fd-fa-cumul-tid
        famth.fd-fa-cumul-open-commit :=
            parcours-famth.fd-fa-cumul-open-commit
        famth.fd-fa-mth-exp := 0
        famth.fd-fa-mth-commit := 0
        famth est element de CDD-FA-MTH
        LIER fctnal avec famth
    
```


finsi

```

si mvt = 522 alors
    famth.fd-fa-cumul-tid := famth.fd-fa-cumul-tid + somme
sinon si mvt = 530 ou mvt = 550 alors
    famth.fd-fa-cumul-open-commit :=
        famth.fd-fa-cumul-open-commit + somme
    famth.fd-fa-mth-commit := famth.fd-fa-mth-commit + somme
    sinon
        famth.fd-fa-year-exp := famth.fd-fa-year-exp + somme
        famth.fd-fa-mth-exp := famth.fd-fa-mth-exp + somme
    finsi

```

finsi

MISE A JOUR de famth dans CDD-FA-MTH

```

divactiv := prendre parent de tasklist dans CDD-DIV-ACTIV
damth.fd-fa-code := tasklist.fd-fa-code
damth.fd-division-id := divactiv.fd-division-id
damth.fd-last-updated := date actuelle /* ANNEE et MOIS */
damth := rche dans CDD-DA-MTH avec cle =
    damth[fd-fa-code,fd-division-id,fd-last-updated]
si damth = Nil alors

```

/* CREATION DU DA-MTH S'IL N'EXISTE PAS DEJA */

```

parcours-damth.fd-fa-code := damth.fd-fa-code
parcours-damth.fd-division-id := damth.fd-division-id
parcours-damth := dernier /* On veut trouver le dernier DA-MTH
    qui ait ete associe a cette
    Activite Divisionnaire. */
parcours-damth := rche dans CDD-DA-MTH
damth.fd-da-cumul-exp := parcours-damth.fd-da-cumul-exp
damth.fd-da-cumul-tid := parcours-damth.fd-da-cumul-tid
damth.fd-da-cumul-open-commit :=
    parcours-damth.fd-da-cumul-open-commit
damth.fd-da-mth-exp := 0
damth.fd-da-mth-commit := 0
damth est element de CDD-DA-MTH
    LIER divactiv avec damth

```

finsi

```

si mvt = 522 alors
    damth.fd-da-cumul-tid := damth.fd-da-cumul-tid + somme
sinon si mvt = 530 ou mvt = 550 alors
    damth.fd-da-cumul-open-commit :=
        damth.fd-da-cumul-open-commit + somme
    damth.fd-da-mth-commit :=
        damth.fd-da-mth-commit + somme
    sinon
        damth.fd-da-cumul-exp := damth.fd-da-cumul-exp + somme
        damth.fd-da-mth-exp := damth.fd-da-mth-exp + somme
    finsi

```

finsi

MISE A JOUR de damth dans CDD-DA-MTH

```

pro.fd-pj-code := tasklist.fd-task-code
pro := prendre dans CDD-PROJECT
    si exty = 'W' ou exty = 'L' ou exty = 'M' ou exty = 'P' alors
        status.fd-pj-code := tasklist.fd-task-code
        status.fd-organic-code := tasklist.fd-organic-code

```



```

status.fd-project-expense-type := exty
status := rche dans CDD-PJ-OU-STATUS avec cle =
           status[fd-pj-code,fd-organic-code,fd-project-expense-type]
si status = Nil alors

    /* CREATION DU PJ-OU-STATUS S'IL N'EXISTE PAS DEJA */
    /* ET DU PREMIER YR-OU-COST QUI LUI EST ASSOCIE */

    status.fd-valid-period-start-date :=
                                   pro.fd-valid-period-start-date
    status.fd-valid-period-end-date :=
                                   pro.fd-valid-period-end-date
    status est element de CDD-PJ-OU-STATUS
    LIER pro avec status
    LIER status avec tasklist
    youcost.fd-pj-code := status.fd-pj-code
    youcost.fd-organic-code := status.fd-organic-code
    youcost.fd-project-expense-type :=
                                   status.fd-project-expense-type
    youcost.fd-last-updated := date actuelle /* ANNEE, MOIS */
    compteur := 1
    Tant que compteur <= 4 faire
        youcost.fd-ou-mat1-cost[compteur] := 0
        compteur := compteur + 1
    Refaire
    youcost est element de CDD-YR-OU-COST
    LIER status avec youcost
finsi

    youcost.fd-pj-code := status.fd-pj-code
    youcost.fd-organic-code := status.fd-organic-code
    youcost.fd-project-expense-type := status.fd-project-expense-type
    youcost.fd-last-updated := date actuelle /* ANNEE et MOIS */
    youcost := rche dans CDD-YR-OU-COST
    si youcost := Nil alors
        parcours-yroucost.fd-pj-code := youcost.fd-pj-code
        parcours-yroucost.fd-organic-code := youcost.fd-organic-code
        parcours-yroucost.fd-project-expense-type :=
                                   youcost.fd-project-expense-type
        parcours-yroucost := dernier /* On veut trouver le dernier
                                   YR-OU-COST qui ait ete associe
                                   au PJ-OU-STATUS. */
        parcours-yroucost := rche dans CDD-YR-OU-COST
        si parcours-yroucost.yy - youcost.yy = 1 alors
            youcost.fd-mat1-cost[1] :=
                                   parcours-yroucost.fd-mat1-cost[2]
            youcost.fd-mat1-cost[2] :=
                                   parcours-yroucost.fd-mat1-cost[3]
            youcost.fd-mat1-cost[3] := 0
            youcost.fd-mat1-cost[4] := youcost.fd-mat1-cost[4] +
                                   parcours-yroucost.fd-mat1-cost[1]
        sinon
            compteur := 1
            Tant que compteur <= 4 faire
                youcost.fd-mat1-cost[compteur] :=
                                   parcours-yroucost.fd-mat1-cost[compteur]
                compteur := compteur + 1
            Refaire
        finsi
    youcost est element de CDD-YR-OU-COST

```

```

    LIER status avec yourcost
  finsi
  yourcost.fd-matl-cost[1] := yourcost.fd-matl-cost[1] + somme
  MISE A JOUR de yourcost dans CDD-YR-OU-COST

  /* RECHERCHE DU PROJECT-STATE ET VERIFICATION QU'IL */
  /* N'EST PAS CLOS */

  state.fd-pj-code := pro.fd-pj-code
  state.fd-amount-status := 'S' /* Correspond au statut "present" */
  state := rche dans CDD-PROJECT-STATE avec cle =
           state[fd-pj-code,fd-amount-status]
  si state.fd-valid-period-end-date < Nil et
      state.fd-valid-period-end-date < date actuelle alors
      code-erreur := 1
      finprocedure
  finsi

  /* RECHERCHE DU YR-PJ-INFO VALIDE POUR LE MOIS ACTUEL */
  /* ET CREATION S'IL N'EXISTE PAS DEJA */

  yrpj-info.fd-project-code := pro.fd-pj-code
  yrpj-info.fd-amount-status := 'S' /* Correspond au status "present" */
  yrpj-info.fd-last-updated := date actuelle /* ANNEE et MOIS */
  yrpj-info := rche dans CDD-YR-PJ-INFO
  si yrpj-info = Nil alors
    parcours-yrpj-info.fd-pj-code := pro.fd-pj-code
    parcours-yrpj-info.fd-amount-status := 'S'
    parcours-yrpj-info := dernier
    parcours-yrpj-info := prendre dans CDD-YR-PJ-INFO
    yrpj-info.fd-pj-matl-cost-rev-date :=
      parcours-yrpj-info.fd-pj-last-matl-rev-date
    yrpj-info.fd-pj-last-manyr-rev-date :=
      parcours-yrpj-info.fd-pj-last-manyr-rev-date
    yrpj-info.fd-pj-last-manmth-tl-rev-date :=
      parcours-yrpj-info.fd-pj-last-manmth-tl-rev-date
    yrpj-info.fd-pj-last-manmth-smc-rev-date :=
      parcours-yrpj-info.fd-pj-last-manmth-smc-rev-date
    si yrpj-info.yy - parcours-yrpj-info.yy = 1 alors
      compteur := 2
      Tant que compteur <= 10 faire
        yrpj-info.fd-pj-matl-cost[compteur - 1] :=
          parcours-yrpj-info.fd-pj-matl-cost[compteur]
        yrpj-info.fd-pj-manyr[compteur - 1] :=
          parcours-yrpj-info.fd-pj-manyr[compteur]
        yrpj-info.fd-pj-manmth-tl[compteur - 1] :=
          parcours-yrpj-info.fd-pj-manmth-tl[compteur]
        yrpj-info.fd-pj-manmth-smc[compteur - 1] :=
          parcours-yrpj-info.fd-pj-manmth-smc[compteur]
        compteur := compteur + 1
      Refaire
      yrpj-info.fd-pj-matl-cost[10] := 0
      yrpj-info.fd-pj-manyr[10] := 0
      yrpj-info.fd-pj-manmth-tl[10] := 0
      yrpj-info.fd-pj-manmth-smc[10] := 0
    sinon
      compteur := 1
      Tant que compteur <= 10 faire
        yrpj-info.fd-pj-matl-cost[compteur] :=
          parcours-yrpj-info.fd-pj-matl-cost[compteur]

```

```
yrpj-info.fd-pj-manyr[compteur] :=  
    parcours-yrpj-info.fd-pj-manyr[compteur]  
yrpj-info.fd-pj-manmth-tl[compteur] :=  
    . parcours-yrpj-info.fd-pj-manmth-tl[compteur]  
yrpj-info.fd-pj-manmth-smc[compteur] :=  
    parcours-yrpj-info.fd-pj-manmth-smc[compteur]  
compteur := compteur + 1
```

Refaire

finsi

yrpj-info est element 'de CDD-YR-PJ-INFO
LIER state avec yrpj-info

finsi

yrpj-info.fd-pj-mat1-cost[1] := yrpj-info.fd-pj-mat1-cost[1] + somme
MISE A JOUR de yrpj-info dans CDD-YR-PJ-INFO

finprocedure

CONCLUSION :

Tous les schémas de programmes qui composent ce mémoire découlent de la modélisation relationnelle. Cependant, la base de données n'étant pas relationnelle, il a été nécessaire d'inclure dans ces schémas des instructions qui se rapprochent des instructions réelles utilisées par le système de gestion de base de données IDMS. Cela facilitera, je l'espère, leur transcription.

Il aurait été possible d'imbriquer des schémas de programmes, une destruction physique d'une Activité Fonctionnelle entraînant automatiquement la destruction physique d'une Activité Divisionnaire par exemple. Je ne l'ai pas fait, car cela permet de mettre en évidence les validations à effectuer avant de détruire une entité.

REMARQUE :

Les informations de type budgétaire sont toutes mensuelles, car cela permet de tenir un "journal" des budgets qui ont été accordés. Cependant le budget d'une Tâche est resté annuel. N'y a-t-il pas là une particularité à revoir ?

Physique expérimentale

Physique à faible/moyenne énergie

Expériences Isolde

Autres expériences type SC

Physique à énergie intermédiaire

Expériences PS 25 GeV

Expériences LEAR

Expériences Neutrino .

Physique à haute énergie, cibles fixes

Expériences WA

Expériences NA

Physique à haute énergie, collisions pp

Expériences ISR

Physique à haute énergie, collisions $p\bar{p}$

Expériences ISR

Expériences UA

Physique à haute énergie, collisions e^+e^-

Expériences LEP

Autres physique

Expériences extérieures : Serpukhov, Proton decay, SIN, USA, ...

Expériences in situs : Gravitation, monopole, jet moléculaire (?)

Physique théoriqueAcquisition et analyse des données expérimentalesDétecteurs

Détecteurs SC (Omicron, ...)

Détecteurs PS

Détecteurs ISR

SFM

AFS

BEBC

EHS

Détecteurs LEP

OPAL

DELPHI

LR 3

Détecteurs spéciaux

Electronique et traitement des données en ligne

Analyse des données

Erasmus, ,,,

Ordinateurs centraux

Réseaux d'ordinateurs

Installations de recherche et support

Accélérateurs

- Complexe injecteur
 - Linacs, Booster, AA
 - Machine PS et contrôle
- Machine SC
- Machine ISR
- Machine SPS
- Machine LEP
 - Linacs
 - EPA
 - Machine LEP et contrôle

Zones expérimentales

- Zones à faible/moyenne énergie
- Zones à énergie intermédiaire
- Zones à haute énergie, cibles fixes
- Zones à haute énergie, collisions pp
- Zones à haute énergie, collisions p $\bar{p}$
- Zones à haute énergie, collisions e $^+$ e $^-$

Support scientifique

- Engineering
- Cryogénie
- Aides aux expériences et administration divisions scientifiques
- CAD

Assistance technique et administrative

Services techniques

- Liquefacteurs
- Transport et manutention
- Ateliers
- Magasins
- Information scientifique
- Autres

Services administratifs

- Direction, gestion
- Finances
- Achats
- Personnel
- Autres

Infrastructure générale

Entretien et exploitation du site

- Entretien exploitation EM
- Nettoyage
- Entretien des bâtiments, des routes, des parkings
- Exploitation amphithéâtre, restaurants, ...

Energie et eau

Assurances

Communications

LISTE DES DIVISIONS :

Division Physique théorique
Division Physique expérimentale
Division Données et Documents
Division LEP
Division Super Synchrotron à Protons
Division Synchrotron à Protons
Division Service Technique et Bâtiments
Département Documentation
Département Finances
Département Information de Gestion
Département Personnel
Commission d'Inspection Technique et de Sécurité

