UNIVERSITE DE NANCY I — U.E.R. DE SCIENCES MATHEMATIQUES

MODIFICATION DES FICHIERS

ET DES TEXTES DANS LE PROJET

"CIVA"

THESE

pour l'obtention du

DOCTORAT de SPECIALITE MATHEMATIQUES (3ème CYCLE)

Soutenue devant le Jury le 27 Avril 1972

par

MAHMOUD PAYAFAR

Jury : Mr J. LEGRAS , Président
Mr C. PAIR , Examinateur
Mr J.C. DERNIAME , Examinateur

UNIVERSITE DE NANCY I U.E.R. DE SCIENCES MATHEMATIQUES

MODIFICATION DES FICHIERS

ET DES TEXTES DANS LE PROJET

"CIVA"

par MAHMOUD PAYAFAR

Je tiens à exprimer ma profonde gratitude à

Monsieur le Professeur LEGRAS, Directeur de l'Institut Universitaire de Calcul Automatique de Nancy, pour l'enseignement qui m'a été dispensé à l'Institut et l'honneur qu'il me fait en présidant le Jury.

Je remercie vivement Monsieur le Professeur PAIR pour l'honneur qu'il me fait en participant au Jury.

Ce travail a été effectué sous la direction de Monsieur DERNIAME à qui j'exprime ma profonde reconnaissance pour la bienveillance et l'aide constante qu'il n'a cessée de me témoigner et pour la formation qu'il m'a donnée.

Que le personnel de l'I. U. C. A. trouve ici l'expression de mes remerciements pour l'aide qu'il m'a apportée.

INTRODUCTION

Notre étude porte essentiellement sur les problèmes de création, de consultation et de modification de fichiers que nous avons abordés en étudiant des systèmes existants et en faisant nous-même un essai de réalisation.

Ce travail se compose de trois parties.

La première consiste en une étude, et la réalisation correspondante des problèmes de modifications de fichiers en civa, étude développée dans le cadre du projet civa (1).

Pour cela, le chapitre I donnera une explication générale des informations et de la structure du fichier en civa. Puis le chapitre II expliquera un processeur chargé du traitement des modifications sur des fichiers internes en CIVA.

La deuxième partie est un exposé d'un module écrit en METASYMBOL et qui permet, comme un processeur autonome, la mise à jour sur un texte source rangé en mémoire secondaire (bande, disque magnétiques).

Le texte à modifier pourrait être écrit en civa ou en n'importe quel langage comme SYMBOL, METASYMBOL, FORTRAN,...

Enfin, la troisième partie est une étude du langage INFOL (Information-oriented-Language) particulièrement adapté aux problèmes de traitement de fichiers :

- déclaration de la structure des données,
- création, consultation et mise à jour des fichiers.

---

(1) Projet développé à l'Université de Nancy I, présenté dans "Introduction au projet CIVA"(doc. interne).

CHAPITRE I

GENERALITES SUR "CIVA" ET SES INFORMATIONS

SOMMAIRE

GENERALITES SUR "CIVA"

CIVA a pour projet de donner à l'utilisateur la possiblité d'employer une formulation unique, c'est à dire un vocabulaire commun, mais aussi une présentation commune pour les différentes phases de la mise en oeuvre d'un travail : leur conception , leur compte-rendu d'analyse par la description des actions à entreprendre, leur programmation, leur exécution et enfin leur maintenance.

L'objectif est donc de fournir un ensemble d'outils permettant de passer, sans heurt, de l'analyse à l'exploitation. Il s'agit, en effet, de définir un langage unique permettant de décrire à la fois les résultats de l'analyse, les programmes et leur exploitation et de construire un système (traducteur et ensemble de services) acceptant ce langage.

En CIVA , la notion de programme autonome disparaît au profit de celle de chaîne d'exploitation, dont la durée de vie n'est pas très courte, ce qui signifie que nous pouvons passer un temps important pour passer de la conception à l'exécution, mais que cette exécution, certainement répétitive, devra être rapide. Il ne s'agit donc pas de faciliter la mise en oeuvre d'un programme puis isolément, auquel cas les outils présentés seraient beaucoup trop lourds.

Il y est essayé de développer, dès l'analyse, l'état d'esprit informatique, qui consiste à tout préciser de manière formelle, donc dans un langage, de manière à ne rien laisser à l'interprétation sémantique. Important pour les rapports entre les hommes, cela permettra aussi de faire intervenir la machine plus tôt.

* En automatisant certaines tâches (regroupement, lancement, etc...).

* En lui confiant directement les résultats de l'analyse. Pour cela, il est prévu les outils suivants :

- programmes de service (aide à l'analyse)
- méthode de description des résultats de l'analyse :
  plus on s'approche des feuilles et plus on entre dans les détails de la méthode développée.

- langage de description, utilisé pour rédiger le dossier d'analyse.
- langage de description des résultats de l'analyse, celui du traducteur, tel que les textes sources se déduisent facilement des descriptions du dossier d'analyse. La description des informations est modulaire (classe) celle des traitements également (modules).
- bibliothèque de toutes les unités de nomenclature d'une application (description d'informations : "classes", descriptions des actions "modules", informations elles-mêmes : "fichier") et les modules de service gérant cette bibliothèque.
- un service d'acquisition, édition, interrogation et mise à jour de fichiers (bien entendu avec la possibilité de définir d'autres services, dans le même langage).
- un système d'exploitation permettant d'assurer l'exploitation des chaînes de traitement obtenues et fournissant des outils de mesure d'efficacité.
- des programmes d'optimisation s'appuyant sur les mesures faites au cours des premières exploitations d'une chaîne.

Notion d'application

La notion de "Programme" disparaissant au profit de celle de "chaîne de traitement" qui sera construite par le système à partir des composants situés en bibliothèque et décrits par l'utilisateur, il est préférable d'instaurer un niveau intermédiaire entre les objets CIVA (modules, classes, fichiers ...) et le système d'exploitation. Ce niveau est lié au domaine d'application à l'intérieur duquel on pourra prendre un certain nombre d'options implicites :

(constantes, types, valeurs des variables d'exploitation, descriptions des fichiers, protections, etc...)

C'est à ce niveau de l'application que l'on définira les bibliothèques. Les nomenclatures peuvent donc être différentes d'une application à l'autre. Ce découpage en application peut refléter le découpage en service d'une entreprise.

Notons l'existence d'une application particulière, celle du système.

I-1 GENERALITE SUR DES INFORMATIONS EN CIVA

En général, il y a quatre sortes d'informations en CIVA :

a) information ELEMENTAIRES qui nous permettent de composer toutes les autres.

b) ARTICLE

Ce sont des informations structurées ayant plusieurs composantes.

c) ENSEMBLE

Un ensemble est une collection d'objets de même nature.

Soit informations ELEMENTAIRES
" ARTICLE
" ENSEMBLE
" FILE

d) FILE

C'est un ensemble qui est muni d'un ordre total et on peut dire qu'une file est une représentation d'un ensemble.

Les définitions qu'on vient de citer sont récursives, par exemple, un article a un certain nombre de composantes qui peuvent être l'un des types cités ci-dessus c'est à dire des informations élémentaires, ou des articles ou des ensembles, ou des files.

On distingue un ensemble et une file parce que la plupart du temps, l'ordre est sans importance : c'est celui dans lequel les objets sont présentés sur leur support. Par exemple, un ordre peut-être donné en indiquant une clé de tri comme (éditer FICHCOM par ordre croissant de COMMAND).

Dans une file, l'accès peut-être séquentiel.

Dans une file ou un ensemble, l'accès peut-être séquentiel ou direct par l'intermédiaire d'une fonction booléenne sur les éléments (par exemple : égalité à un champ ou d'un article d'une file d'articles).

I-2 TYPE DES INFORMATIONS

Dans une suite d'informations ou un problème, interviennent un certain nombre de types :

- un type est un ensemble de valeurs et dire qu'une variable est d'un certain type, c'est dire qu'elle prend des valeurs appartenant à cet ensemble.

I-2-1 Analyse des types

a) Les types peuvent être communs à différentes parties du problème et définis indépendamment des traitements.

b) Ils ne se définissent qu'au fur et à mesure de l'analyse:
par exemple : on définit une file d'ARTICLES puis plus tard les noms de composantes de l'article et enfin la description des composantes.

Pour que l'on puisse travailler d'une telle façon, on est conduit à une description modulaire des types pouvant être indépendante des traitements, c'est à dire (notion de classe), avec renvoi d'une classe à l'autre pour compléter une définition.

Les types peuvent être communs à différents problèmes, ils sont alors définis au niveau de l'application ou au niveau du système s'ils sont communs à diverses applications.

I-3 DESCRIPTION EXPLICITE DES TYPES

I-3-1 Informations élémentaires :

En général, on connaît deux formats d'informations :
information interne et externe

Les objets élémentaires peuvent être déclarés de trois façons :

a) Définitions standards

Ils sont du type courant, comme entier, décimal ou réel en format interne.

On dit qu'un identificateur est de l'un de ces types par une déclaration de la forme : N entier , K réel , e, D décimal .

b) Définition par restriction

On attribue au nom du type des conditions restrictives qui définissent ainsi un nouvel ensemble de valeurs, donc un nouveau type par exemple : M type entier < 400.

c) Définition explicite par l'ensemble des valeurs

Si un identificateur ne peut prendre qu'un ensemble de valeurs restreint, on pourra définir son type par la donnée de cet ensemble qui est un code.

Ex : SITUATION code marié, célibataire, veuf, divorcé.

I-3-2 Définition explicite par concaténation : ensemble et file

Un ensemble ou une file K est constitué d'une suite d'éléments : si on veut définir l'ensemble des valeurs possibles de K, il suffit de définir l'ensemble des valeurs possibles (type) des éléments et leur nombre dans la file : (puissance $n^{ième}$ d'un ensemble), c'est à dire préciser la taille de la file.

Il y a trois types de files :

- les files de taille fixe comportent un nombre fixe d'éléments
- - les files de taille bornée comportent un nombre maximum d'éléments
- les files de taille variable comportent un nombre quelconque d'éléments

On doit alors préciser le nombre exact (file fixe) au maximum (file bornée) d'éléments, ou un moyen de la calculer.

Exemple :

TAB1	type	file (20)	caract
TAB2	type	(max 20)	caract
T	type	(30)	décimal
T	type	(j)	décimal
T	type	file	décimal

Dans une file utilisée en entrée (file bornée ou variable) la taille actuelle est fixée par les données d'entrée, le type externe correspondant devra comporter une condition restrictive.

Un type externe pourrait être $(y \neq 0)$ la file A serait constituée des caractères situés dans le fichier d'entrée à partir du début de la file jusqu'au premier point trouvé.

En cas d'une file de travail ou de sortie si la taille est donnée par une variable celle-ci devra avoir une valeur lors d'un appel du module qui la déclare.

I-3-3 ARTICLES : définition par composition

La définition du type d'un article se fait par la donnée du type de chacun de ses champs (produit d'un ensemble),

Personnel type (nom, adresse) ;

où adresse est adresse type file (max=50) caract.

La déclaration d'un identificateur du type Personnel se fera en précisant pour chaque champ un identificateur appelé sélecteur.

Personnel C Personnel (nom C, adresse C)

I-4 DEFINITIONS IMPLICITES DE TYPES

Les définitions explicites nous permettent d'éviter de redéfinir plusieurs fois les mêmes types, mais elles présentent quelques inconvénients:

* lourdeur d'écriture
* nombre d'identificateurs trop grand
* besoin de deux niveaux d'identificateurs (celui du type et celui du sélecteur) alors que bien souvent on ne veut utiliser que peu de sélecteurs du même type. On adaptera donc en plus des déclaration explicites, des définitions implicites de types :

I-4-1 Confusion type - sélecteur :

Lorsqu'on veut déclarer un seul identificateur d'un certain type on pourra prendre le type pour sélecteur, comme on le fait couramment.

I-4-2 Analogies

Lorsqu'on a déclaré un identificateur avec une définition implicite de type, et lorsqu'on veut déclarer un autre identificateur du même type il est nécessaire de redéfinir le type, ce qui peut paraître gênant, ou de procéder par analogie, nous l'accepterons donc de deux façons :

a) analogie par les identificateurs

Si "TAB" est un identificateur d'un certain type, alors tout identificateur s'écrivant TAB suivi d'un chiffre, "TAB2" par exemple, et dont le type n'est pas spécifié, sera considéré du même type que TAB.

b) Analogie par "comme"

Nous déclarons un nouvel identificateur du même type qu'un autre sans qu'il y ait ressemblance de nom, par exemple :

M (N file (10) entier , B booléen) ;
H comme M ;
a le même effet que type t (file (10) entier, booléen)
M, H t (N, B)

si M et N appartiennent au même type et si nous ne voulons pas qu'ils aient les mêmes noms de sélecteurs il suffira de préciser les sélecteurs désirés.

N comme M(Z, U) a le même effet que
N t (Z, U) .

I-5 LES CLASSES

Les déclarations pourront se trouver en tête de module, elles lui sont alors locales.

La description des informations (fichiers ou zone de travail) doit pouvoir être faite indépendamment de la description des actions, ceci est essentiel pour permettre la rédaction aisée de modules indépendants ou d'objets autonomes. De plus, comme elle doit se faire pendant l'analyse au fur et à mesure des possibilités, elle doit être modulaire.

La liaison entre une description de données et un module se fera à la compilation du module, la liaison avec les fichiers eux-mêmes se fera à l'exécution.

Une classe est un ensemble de déclarations portant un nom, c'est donc une unité de nomenclature. Elle figure en bibliothèque.

Pour pouvoir utiliser les identificateurs d'une classe A dans un module ou une autre classe B, on précisera dans ce module ou dans B :

UTILISE A ;

Exemple :

```
MODULE EXEMP ;
      UTILISE A ;
      FCOM fichier file COM
        :
      fin module

CLASSE A :
      COM type (CLIENT comme PERSONNE, NO COMMANDE,
            QUANTITE entier ; DESIGNATION)

      fin classe

CLASSE B ;
      PERSONNE type (NOM file (max=15) Car ; NUMERO entier) ;
      NOCOMMANDE type entier < 5000 ;
      DESIGNATION type (ref entier < 10000 ; file (Max =30) Car)
      fin classe
```

Notons que l'on pourra définir par application, une classe de l'application qui sera considérée comme un "prologue standard" pour toutes les chaînes de traitement de l'application (automatiquement déclarée) et c'est

dans cette classe qu'on déclarera la plupart du temps les types fréquemment utilisés. Elle peut donc constituer la "nomenclature" des information manipulées dans une application, donc, par exemple, dans un service de l'entreprise. D'autre part, elle peut être utilisée pour définir toutes les constantes caractéristiques de cette application et certaines options par l'intermédiaire de variables réservées.

I-6 DESCRIPTION DES INFORMATIONS

a) les informations sont décrites dans un module ou dans une classe

b) elles sont d'un certain type : on a vu qu'il peut être donné explicitement par un sélecteur ou par le nom d'un sélecteur suivi d'un nombre

c) Durée de vie

Il faut dire qu'un identificateur est local à un module : celui en tête duquel il est déclaré ou en tête duquel est déclarée la classe qui le contient.

d) Mise en mémoire

Pour l'analyste, toutes les informations sont également accessibles (du moins au début de son travail) donc tout est "interne" donc les descriptions des informations (classes) ne contiennent rien sur cette mise en mémoire et peuvent être écrites très tôt.

Cela signifie que pendant un traitement tous les fichiers sont internes : un fichier ne pourra être traité que s'il a été entièrement lu (rendre interne : acquisition de fichier) ; de même il est imprimé en une seule fois (édition).

I-7 PRESENTATION DES FICHIERS

I-7-1 Nature

Un fichier est une collection d'objets dotée d'une structure et il convient de distinguer la structure, qui est une constante, des objets eux-mêmes qui sont des valeurs, et essentiellement variables.

Pour caractériser un fichier il faut préciser :

a) son support ;

b) la répartition des objets sur le support (organisation physique) ;

c) la présentation de chacun des objets à l'extérieur du système (format) ;

d) la présentation des objets dans le système (pour qu'ils puissent être traités) ;

e) l'organisation des objets entre-eux (organisation logique) ; et d'autres renseignements concernant sa durée de vie, protections.

I-7-2 Description logique d'un fichier

C'est une description de structure, elle est faite dans une classe. Elle permet de décrire les enregistrements logiques du fichier : c'est à dire un ensemble d'informations auxquelles on veut pouvoir accéder "en même temps". Le nom d'un enregistrement logique sera précédé d'un * dans la description.

Elle contient les noms des différents champs, leur type, leur organisation.

Elle peut-être faite par niveaux (simplification d'écriture). Elle peut être donnée de façon modulaire, mais pour utiliser une telle description, elle devra être complète.

I-7-3 Répartition des objets sur le support (fichier interne)

Il s'agit là de l'organisation physique du fichier interne sur son support (bande ou disque).

Le découpage et la répartition dans les blocs physiques est une contrainte que l'utilisateur doit ignorer, c'est le système qui se préoccupe de cette répartition en utilisant un facteur de blocage : tous les blocs physiques auront la même taille et seront tous pleins, sauf, éventuellement, le dernier. Il n'y a pas de limite théorique à la taille d'un enregistrement logique, il peut être contenu dans un bloc, contenir plusieurs blocs, ou être à cheval sur deux blocs. Pratiquement, un enregistrement

logique nécessite un buffer de sa taille. Cependant, la disposition des enregistrements (fichier consécutif ou à clé devra être donné avec la description logique).

Exemple :

FICHIER EXEMP file article clé nom.

Remarque : Un enregistrement est donc composé de deux parties, la partie statique qui contient les objets de taille fixe dont on peut connaître l'adresse relative dans l'enregistrement (voir partie traduction classes), car elle est la même pour tous les enregistrements, la partie dynamique pour laquelle les objets (files variables) sont repérés par des pointeurs dans les parties fixes de chaque enregistrement.

I-7-4 Représentation des objets sur le support (fichier interne)

a) Informations élémentaires et structures de taille fixe : les entiers et les réels sont en binaire sur des mots de 32 bits, un complexe occupe 2 mots, un décimal est codé en condensé sous le format utilisé dans les calculs.

b) File dans un enregistrement logique

Pour les files de taille fixe, il n'y a aucun problème particulier, mais pour les files de taille bornée, sur le support elles sont précédées d'un entier égal à la taille actuelle. Elles occupent sur le support une zone correspondant à leur taille maximum.

b-1) File de taille variable

Elles seront toujours placées en fin d'enregistrement logique sur le support, quelle que soit leur place dans la description.

Un enregistrement qui possède des files de longueur variable, contiendra dans la partie statique un entier pour chacune d'elles : celui-ci désigne l'adresse relative au début de l'enregistrement de l'origine réelle de la file dans la partie dynamique.

Dans la partie dynamique, les morceaux contigüs de cette file sont chaînés entre eux de la façon suivante : chacun d'eux est précédé de

deux entiers occupant deux demi-mots consécutifs :

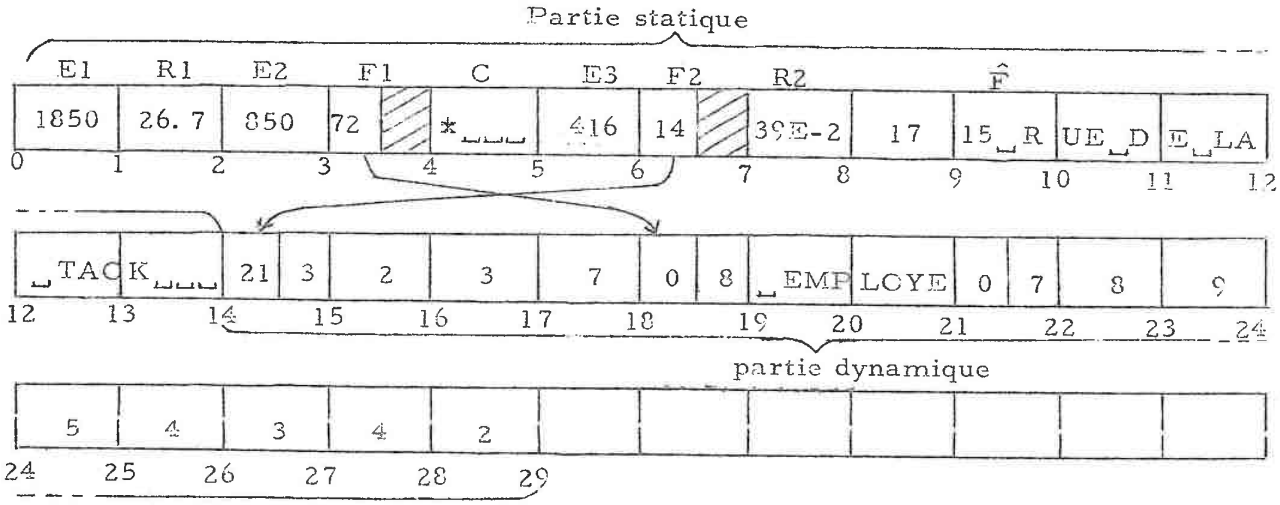
- le premier : contient un entier positif (adresse relative au début d'enregistrement de l'origine du morceau suivant de cette file) ou bien zéro signifiant que l'on attend le dernier morceau de cette file.
- le deuxième : contient une valeur entière positive différente, zéro indiquant le nombre d'éléments dans ce morceau.

Exemple :

Un enregistrement logique

E1	R1	E2	F1	C	E3
entier	réel	entier	file variable(caract)	carac	entier
1850	26.7	850	␣EMPLOYE	*	416

F2	R2	$\hat{F}$
file variable (entier)	réel	file bornée (max=20)
2 3 7 8 9 5 4 3 4 2	39E-2	15 rue de la TACK



I-8 TRADUCTION DES CLASSES

Pour chaque classe on construit les quatre tables suivantes :

1 - ZTAB (table des identificateurs)

2 - ZCC (table des constantes)

3 - ZTCLA (table des chaînages des appeles de classe)

4 - ZMCCLA (table des renseignements sur les identificateurs non simples)

Remarque : le nom de la classe est rangé dans la liste des classes existant pour l'application.

I-8-I ZTAB (table des identificateurs)

Cette table est organisée pour utiliser une méthode de recherche par "Hashing". On y rencontre par identificateur une zone de 4 mots :

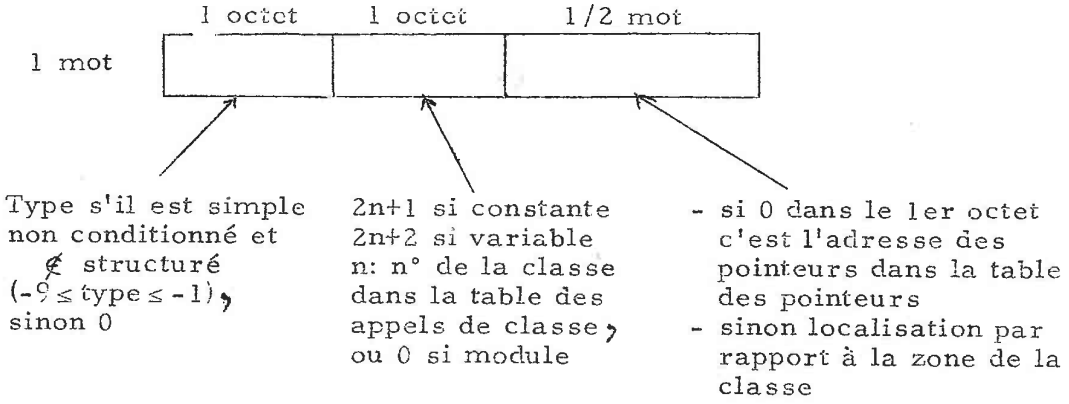
pointeur de table	descripteur	nom d'identificateur	

Le pointeur de table contient l'adresse relative du prochain identificateur de la table commençant par la même lettre que lui, ou il vaut zéro s'il est le dernier identificateur de la table commençant par cette lettre.

A cette table est associée une table qui, pour chaque lettre, donne l'adresse relative dans la table des identificateurs, du premier identificateur commençant par cette lettre ou zéro s'il n'y a pas d'identificateur dans la table des identificateurs par cette lettre.

I-8-2 Descripteur d'un identificateur ou d'une constante

Ce descripteur se trouve dans la table des constantes, dans le tableau des identificateurs, et dans la chaîne codée. Dans cette chaîne, c'est lui qui permet de définir le type et la localisation de l'information manipulée directement ou bien indirectement.



I-8-3 ZCO (table des valeurs des constantes)

Elle contient deux mots mémoire par constante :

soit

descripteur	valeur de la constante

si la constante est un réel ou un entier (si elle tient sur un mot);

soit

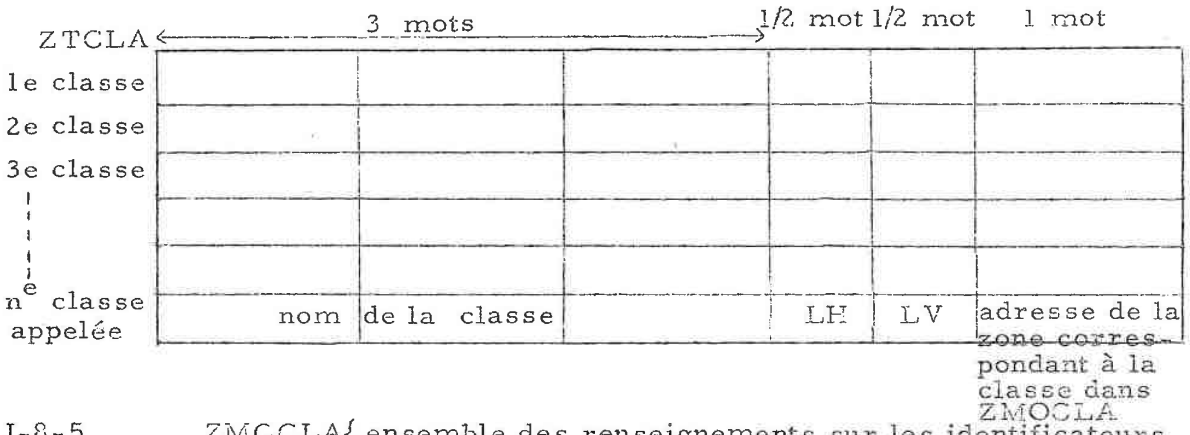
0	descripteur	valeur de la constante

si la constante est en double précision (si elle tient sur 2 mots) le descripteur est le même que l'on retrouve dans la table des identificateurs, si la constante a un nom, il suit le schéma général des descripteurs d'identificateurs.

La table des valeurs des constantes qui est transmise ne comporte que les valeurs des constantes rangées bout à bout dans ZCO.

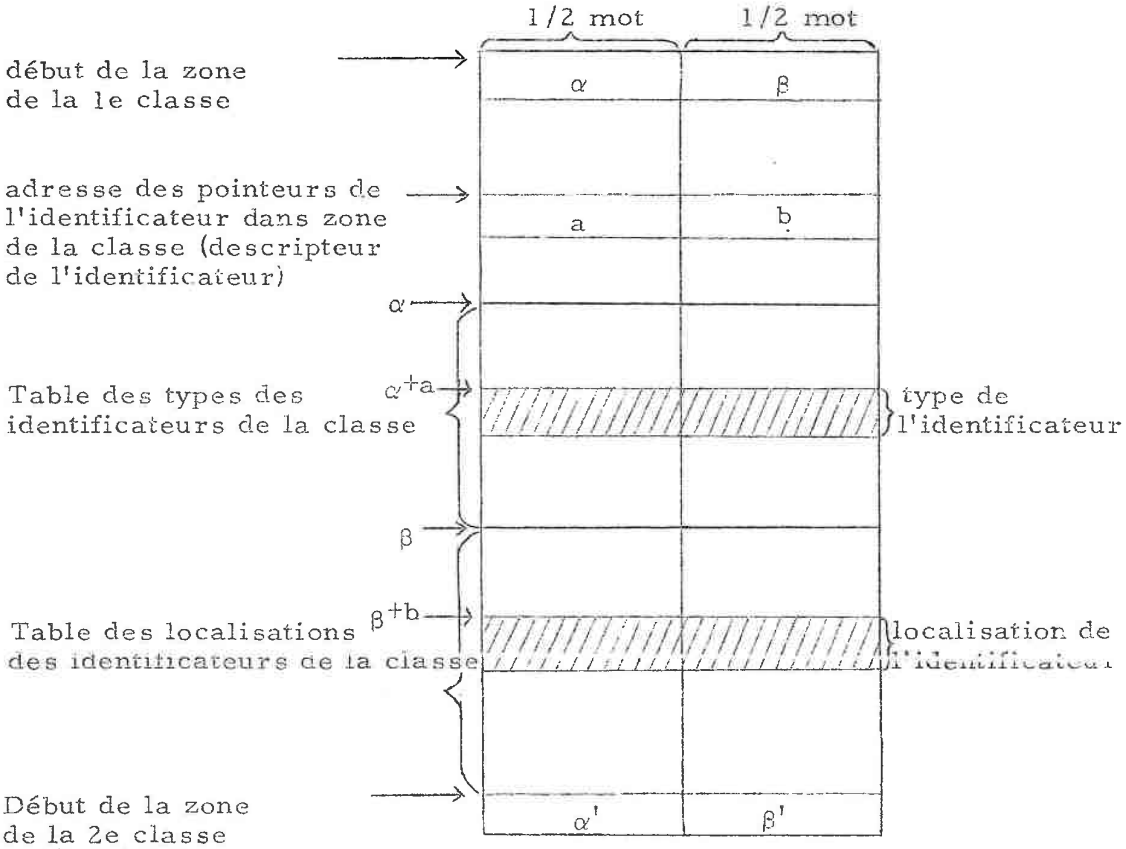
I-8-4 ZTCLA (table de chaînanage des appels de classes)

Cette table sert à donner un numéro à chaque classe utilisée par le module ou la classe, on intègre à cette table la table de chaînage des appels de chaque classe utilisée, en vérifiant que les classes qu'elle contient ne sont pas déjà dans la première classe.



I-8-5 ZMCCLA{ ensemble des renseignements sur les identificateurs non simples des classes appelées }

Elle est constituée de zones (une zone par classe) mises bout à bout.



I-9 REPRESENTATION DE DESCRIPTEUR DES DONNEES

I-9-1 Types :

En général il y a 5 catégories de types, chacun d'eux se compose de différents éléments. A chaque élément a été associée une valeur qui est son nom : on représente à la suite les différentes catégories avec les codes correspondant de chaque élément.

a) Types simples

TYPE	Code
Bit	-1
Caractère	-2
Booléen	-3
Entier	-4
Réel	-5
Décimal	-6
Complexe	-7
Double précision réel	-8
Double précision complexe	-9

b) Types file

TYPE	Code
File de longueur fixe ⊂ enregistrement	-10
File de longueur fixe = enregistrement	-11
File de longueur max ⊂ enregistrement	-12
File de longueur max = enregistrement	-13
File de longueur variable ⊂ enregistrement	-14
File de longueur variable = enregistrement	-15

c) Types STRUCTURE

TYPE	Code
STRUCTURE ⊂ enregistrement	-16
STRUCTURE = enregistrement	-17
STRUCTURE conditionnelle ⊂ enregistrement	-18
STRUCTURE conditionnelle = enregistrement	-19

d) Type fichier

TYPE	Code
Fichier ensemble structure consécutif	-20
Fichier ensemble structure à clé	-21
Fichier file consécutif	-22
Fichier file à clé	-23

e) Autres types

TYPE	Code
Code	-24
Constante	-25
Procédure	-26

I-9-2 Descripteurs de type

a) Types simples

Code Types -1 ou -2 : -8	lien horizontal	[Conditions]
1/2 mot	1:2 mot	1 mot

-7 ou -9	lien horizontal	0 ou 1	taille	Condition
1/2 mot	1/2 mot	1 bit	1/2 mot	

b) Types file

-10 ou -11	lien horizontal LH	taille fixe	type des éléments ou POINTEUR vers le type
1/2 mot	1/2 mot	1/2 mot	1/2 mot

-12 ou -13	lien horizontal LH	taille maximum	type des éléments ou POINTEUR vers le type
1/2 mot	1/2 mot	1/2 mot	1/2 mot

-14 ou -15	lien horizontal LH	type des éléments ou POINTEUR vers le type
1/2 mot	1/2 mot	1/2 mot

c) Types structure

-16 ou -17	lien horizontal LH	lien vertical LV
1/2 mot	1/2 mot	1/2 mot

-18 ou -19	lien horizontal LH	pointeur vers identificateur

	lien horizontal LH	lien vertical LV	valeur

d) Types fichier

-20	lien horizontal LH	lien vertical LV	
-21	lien horizontal LH	lien vertical LV	Pointeur vers identificateur clé
-22	LH lien horizontal	LV lien vertical	Type des éléments de la file
-23	lien horizontal LH	lien vertical LV	Type des éléments de la file / Pointeur vers identificateur clé
1/2 mot	1/2 mot	1/2 mot	1/2 mot

Descripteur des localisations

a) Type simple

Adresse relative de mot ou double-mots ou octet du champ sur le support selon le type.

Adresse relative dans le buffer
1/2 mot

b) Type structure

Adresse relative d'octets dans le buffer
1/2 mot

c) File fixe et bornée

taille	Adresse relative

d) File variable

Adresse relative selon le type dans le buffer

CHAPITRE II

MODIFICATION D'INFORMATIONS DES FICHIERS EN CIVA PROCESSEUR "MAJIC"

SOMMAIRE

PROCESSEUR "MAJIC"

II-1 INTRODUCTION

Au cours du chapitre précédent, on vient de mettre en évidence la notion de "classe" (description d'information) et la notion d'information elle-même : "fichier" en CIVA. On a cité, à titre d'exemple, quelques services de CIVA comme l'acquisition, la création, l'interrogation et l'édition de fichier, l'existence de modules de service gérant la bibliothèque d'une application et la bibliothèque du système.

On a cité aussi l'ensemble des descriptions d'informations "classe" et des descriptions d'actions "modules".

Dans la suite, on va étudier un autre service : mise à jour de fichier.

L'intérêt en est évident pour des fichiers permanents à grande durée de vie. Un fichier décrit en Civa est toujours en format interne c'est à dire qu'ils sont convertis et transformés de la forme externe en forme interne par l'intermédiaire du service d'acquisition*.

Pour pouvoir modifier les valeurs d'un ou plusieurs champs d'un enregistrement d'un fichier déjà créé, donc en format interne , il est indispensable de créer un module autonome : il fera la mise à jour des informations au niveau des champs à l'intérieur des enregistrements, c'est à dire qu'il attribuera les nouvelles valeurs aux différents champs d'un enregistrement déjà existant sans avoir besoin de passer par le service d'acquisition

Le travail essentiel de ce service appelé "MAJIC" (Mise A Jour d'Informations CIVA) est basé sur des fichiers en format interne et c'est grâce à la description d'informations (ensemble de classes) du fichier qu'on aura accès aux diverses informations sur leur support. Bien entendu les nouvelles données sont d'abord tranformées de la forme externe en format interne en effectuant toutes les conversions nécessaires puis elles remplacent les anciennes valeurs sur le support.

---

* Il faut bien préciser que la tâche essentielle du service d'acquisition est d'attribuer entièrement et en une seule fois les données correspondant à chaque fichier à l'aide de "description logique" et "description externe" du fichier.

Pour faire ce traitement, MAJIC doit avoir à sa disposition :

- le fichier à traiter
- la description d'informations (classes)
- le fichier de MODIFICATIONS (valeurs et noms des champs concernés)

Pour que le traitement soit bien clair, et avant de passer à l'organisation du processeur "MAJIC", nous allons exposer son rôle à propos d'un exemple très simple : apporter quelques modifications à un fichier de personnel.

On suppose qu'on dispose du fichier du personnel d'une société et que la structure ou description logique de ce fichier est donnée dans une "classe" de la façon suivante :

Description logique :

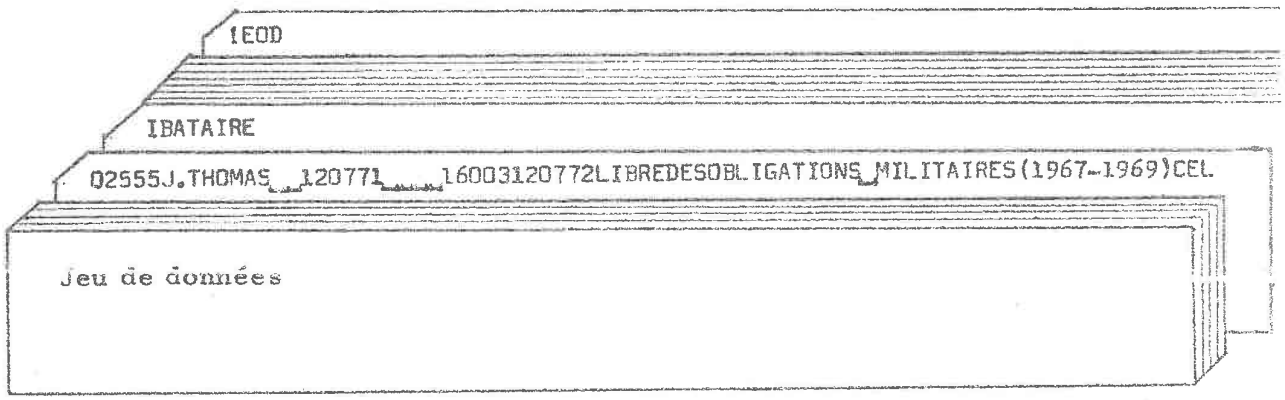
```
                classe EXEMP :

fichier PERSONNEL      file PERSONNEL ;

    * Personnel    01  N° DE FICHE DEC 9(5)
                   01  NOM (10)  CARACT
                   01  DATE D'ENGAGEMENT ENTIER
                   01  SALAIRE MENSUEL ENTIER
                   01  COTATION ENTIER CODE EXCELLENT - TRES BIEN -
                                            BIEN - SATISFAISANT
                   01  DATE DE REVISION ENTIER
                   01  SITUATION MILITAIRE (max=30) CARACT
                   01  SITUATION MARITALE (20)      CARACT
                    .
                    .
                    .
                   fin classe ;
```

Description du fichier externe :

PERSONNEL	: N° DE FICHE ≠ 0
N° de FICHE	: C5 < 10000
NOM	: A(10)
DATE D'ENGAGEMENT	: C6
SAUT	: (4)
SALAIRE MENSUEL	: C4
COTATION	: C1
DATE DE REVISION	: C6
SITUATION MILITAIRE	: ≠␣
SITUATION MARITALE	: A(20)
⋮	

Le service d'acquisition, à l'aide des deux descriptions qu'on vient de donner ci-dessus, transforme les données à partir du format externe en format interne tout le fichier en cause. Les données seront présentées sous forme d'une chaîne de caractère sur le support carte comme :

DONNEES



Ces données sont réparties sur le support comme ci-dessous

n° de fiche	Nom	Date	Saut	Salaire	Cotation	Date	Taille actuelle
2555	J.THOMAS	120771		1600	3	120772	42
1 mot	10 caractères	1 mot 4 caract		1 mot	1 mot	1 mot	1 mot

Situation militaire

LIBRE_DE_OBLIGATIONS_MILITAIRE(1967-1969)		

Situation maritale

CELIBATAIRE	

20 caractères

Maintenant ce fichier est complètement en format interne et il est utilisable.

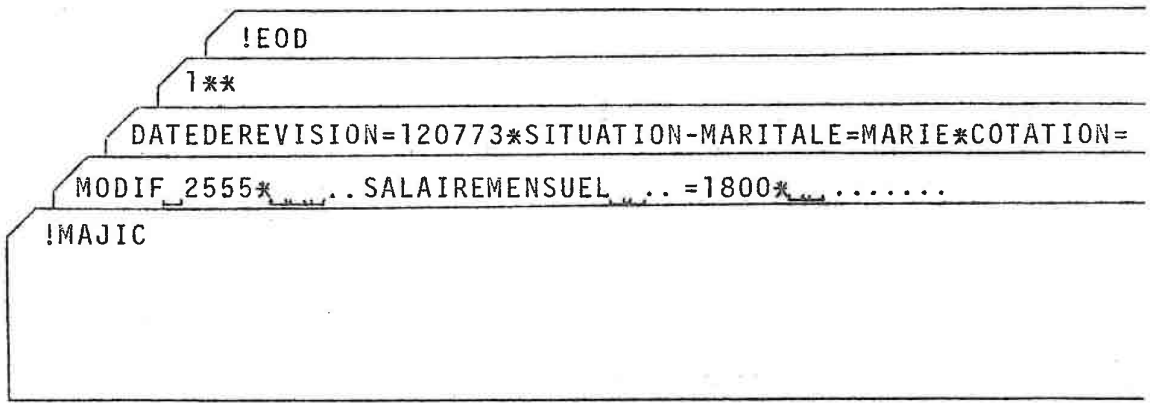
Plus tard, l'utilisateur aura l'intention de faire des modifications sur les valeurs de quelques champs de ce fichier. Il ne serait pas logique de reproduire un fichier complet de nouvelles données et de passer par le service d'acquisition, ce qui demanderait une perforation très coûteuse étant donné la masse des données. On utilisera donc le processeur de mise à jour qui fera une telle modification.

Exemple : on veut faire les modifications suivantes sur un enregistrement particulier (personnel qui porte le numéro 2555) du fichier PERSONNEL :

1 - le nouveau salaire : 1800 F
2 - la date de prochaine révision : 12 07 73
3 - le code de cotation : 1
4 - la situation maritale : MARIE

Les nouvelles valeurs seront représentées dans un fichier de modifications, avec le nom des champs auxquels on veut les attribuer. Ce fichier est toujours précédé d'une carte d'appel au processeur MAJIC qui sera unique pour tous les traitements de mise à jour du fichier à traite (Personnel).

Le traitement de modifications de chaque enregistrement de ce fichier commencera par une carte qui porte l'option MODIF.



Fichier modification

Le processeur fait une analyse des identificateurs comme

SALAIRE MENSUEL , DATE DE REVISION , etc... ,

puis il consulte le descripteur de la classe dans laquelle ces identificateurs sont déclarés (voir I-8) pour trouver le type de l'objet et sa localisation sur le support ; ensuite, à l'aide du type, il fait la conversion nécessaire pour la transformation des données du format externe en format interne ; enfin, grâce à la localisation, il range les nouvelles valeurs à leur place sur le support.

Remarque : Dans la suite, on parle d'IDENTIFICATEUR ou simplement IDEN, ce peut-être l'un des identificateurs :

SALAIRE MENSUEL , COTATION , etc... ,

on parle aussi de VALEUR qui peut-être l'une des valeurs attribuée à des identificateurs comme 1600 ou 120773, etc...

II-2 ORGANISATION GENERALE

"MAJIC" est un ensemble de programme et sous-programme écrits e langage METASYMBOL sur CII 10070. Ce programme est appelé en mémoire comme un processeur,ou comme un sous-programme afin d'effectuer des opérations de modifications et de mise à jour sur des informations élémentaires, files ou structures (en format interne) d'un fichier en CIVA.

II-2-1 Fichiers en CIVA

En CIVA, des fichiers (files externes) sont des files dont les éléments sont des enregistrements logiques ou comprennent des enregistrements logiques.

II-2-2 Désignation d'un enregistrement

Il y aura deux façons de désigner enregistrement.
Soit les enregistrements qui possèdent un nom (clé) qui est attribué à la création du fichier, s'il ne le possède pas, un enregistrement peut être désigné par sa place dans le fichier (un rang), ou par une condition à rempli on a choisi cette solution.
Un enregistrement pourra être désigner par l'égalité d'un de ses champs à une valeur donnée. Ce champ sera appelé SELECTION, son nom sera donné lors de l'appel de MAJIC, la valeur sera appelée "sélecteur". Elle sera donnée pour chaque modification.

II-2-3 TYPE DE FICHIERS

"MAJIC" est organisé en vue de traiter les fichiers en :

a) ACCES SEQUENTIEL SUR UN FICHIER A CLES

On procède à la mise à jour des enregistrements demandés par un sélecteur.

Dans ce cas, le traitement consiste en une ouverture du fichier et une présentation des enregistrements dans l'ordre des clés dans le répertoire des clés.

A la fin, il y a FERMETURE du fichier.

b) ACCES DIRECT DANS UN FICHIER A CLES

Le traitement dans ce cas consiste en une OUVERTURE du fichier. Puis on appelle l'enregistrement par son nom qui est clé ou sélecteur (un champfixe de l'enregistrement). Le système tient un répertoire des clés (qui donne l'adresse physique associée à ce nom). La fin de l'activation du processeur demande une FERMETURE du fichier.

c) ACCES SEQUENTIEL SUR UN FICHIER CONSECUTIF

Le traitement consiste en une ouverture du fichier et une présentation des enregistrements dans l'ordre où ils figurent dans le fichier. Le nom de champ qui suit MAJIC (SELECTION) indique le champ qui permettra de choisir les enregistrements à mettre à jour. On choisit le premier enregistrement rencontré pour lequel ce champ a pour valeur la valeur indiquée après MODIF (SELECTEUR).

Remarque 1 : Dans la suite, on utilise les noms symboliques

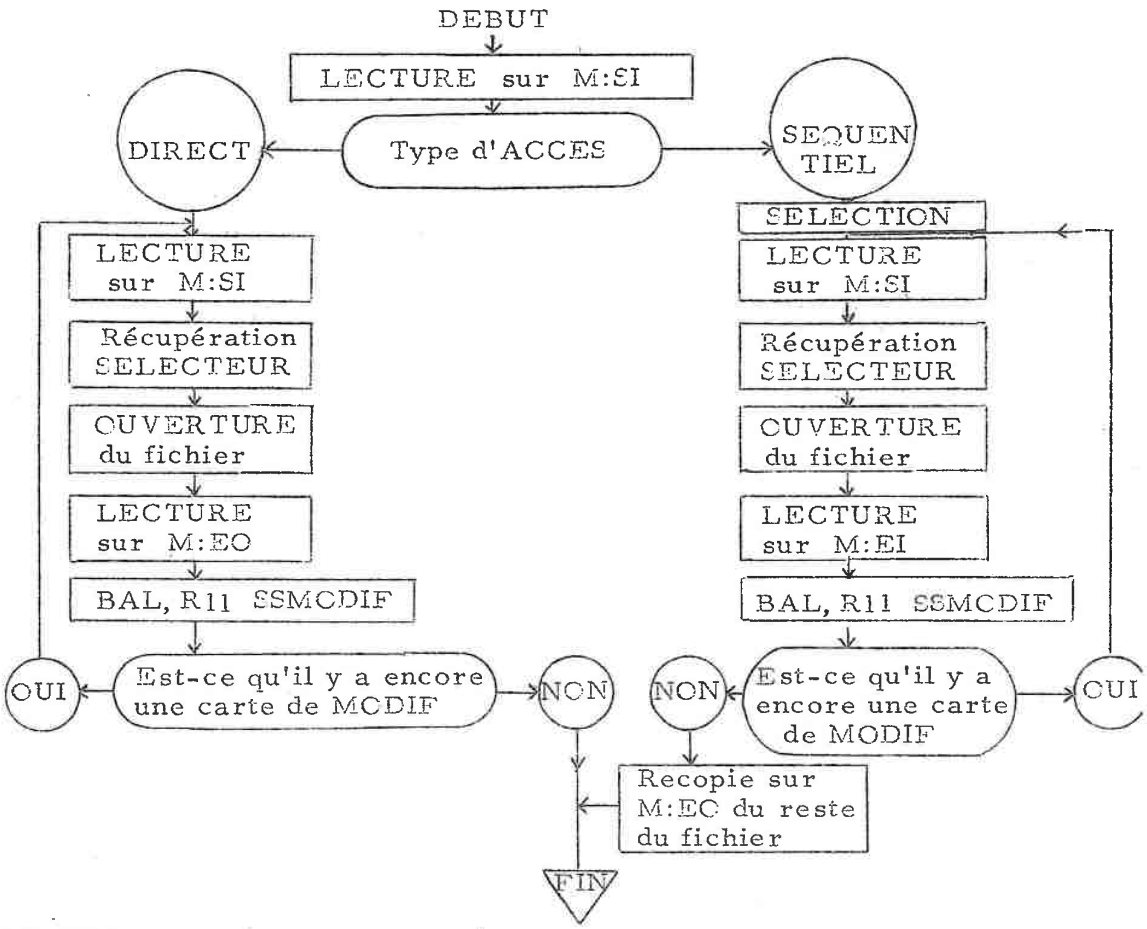
M:SI , M:EI , M:EC , M:BI , M:LO comme étiquette opérationnelle

Ce sont des adresses de tables DCB qui sont utilisées par le moniteur pour effectuer les entrées-sorties. Ils sont associés implicitement à un périphérique :

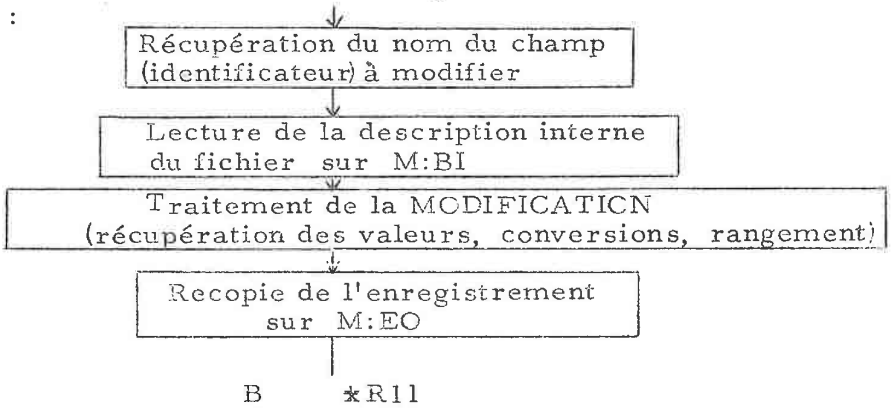
M:SI	(Symbolique Input)	est affecté au lecteur de carte
M:EI	(Elément Input)	fichier entrée
M:EO	(Elément Output)	fichier sortie
M:BI	(Binaire Input)	fichier entrée en binaire
M:LO	(Listing Output)	listing sortie.

Remarque 2 : Un identificateur est appelé du type simple s'il n'est pas structure ou file.

Organigramme général de "MAJIC"



SSMODIF :



II-3 LES TACHES DU PROCESSEUR "MAJIC"

La tâche principale de "MAJIC" sera divisée en 6 étapes.

II-3-1 - Lecture sur M:SI de la première carte du fichier modification.
Cette carte a le type suivant :

```
!MAJIC␣[(SELECTION, nom)]
```

"!MAJIC" a été utilisé par le moniteur qui a chargé le processeur MAJIC.

- On recherche dans la carte précédente la présence du paramètre SELECTION.
  S'il n'existe pas, on passera à la phase suivante ;
  S'il existe, on récupère le paramètre de SELECTION.

II-3-2 - Lecture de la 2ème carte du fichier modification.

- Premier stade d'analyse de cette carte :
  Récupération de l'option "MODIF". Puis si le fichier est en accès direct, récupération de la valeur du SELECTEUR, sinon le fichier est alors en accès séquentiel, tout d'abord à l'aide de nom (identificateur) récupéré dans la première étape, on consulte avec le descripteur de la classe dans laquelle l'identificateur est déclaré (voir I-8) pour trouver sa localisation et sa taille puis on trie le fichier.

II-3-3 OUVERTURE du fichier

- En fonction du type d'accès, l'enregistrement à traiter est lu dans une zone de travail appelée TAMPON. Pour le fichier en accès direct, le traitement consiste simplement en une lecture de l'enregistrement en mémoire. Dans le cas de fichier en accès séquentiel, on doit avoir à notre disposition deux fichiers : l'un est le fichier à traiter (fichier source) et l'autre est utilisé pour faire la recopie des informations du fichier

Source sans modification ou après la modification sur le fichier mise à jour.

SSMODIF :

II-3-4

Récupération des identificateurs à modifier du fichier modification et lecture de la description des informations du fichier sur M:BI, pour avoir accès aux différents champs d'enregistrement à l'aide du descripteur de la localisation et du type de conversion grâce au descripteur de type.

II-3-5

- Traitement de la modification, mise à jour proprement dite et recopie sur M:EO (c'est la phase essentielle de MAJIC dont on parlera complètement dans la suite).

II-3-6

Fin du fichier modification. En cas de fichier en accès direct, c'est la fin du traitement, mais pour un fichier en accès séquentiel, on doit d'abord recopier les derniers enregistrements du fichier source s'il en reste.

II-4 REPRESENTATION DU FICHIER DE MODIFICATIONS

II-4-1 Un appel du processeur MAJIC peut demander la modification d'un certain nombre d'enregistrements. La modification d'un enregistrement sera demandée par le mot réservé MODIF suivie de la désignation de l'enregistrement à modifier, puis des modifications à effectuer. Une telle modification s'écrit :

MODIF ␣ sélecteur (sur N caractères) * [modification élémentaire *]^{n}

N maximum 11.
n signifie qu'on pourra répéter n fois des couples du type :

modification élémentaire *

une modification élémentaire s'écrira :

Identificateur = valeur

ou

Identificateur = suite de valeurs

La suite de valeurs est réduite à une seule valeur si l'identificateur est de type simple ; c'est une suite de valeurs séparées par des virgules si l'identificateur désigne un objet du type file ou structure.

La valeur ainsi décrite est attribuée par MAJIC à l'identificateur désigné.

Le fichier externe de modifications sera présenté sous la forme d'une chaîne de caractères. Pour traiter une telle chaîne on a introduit quelques caractères séparateurs entre les différents identificateurs comme :

MODIF , SELECTEUR , IDENTIFICATEUR et VALEURS.

II 4 2 SEPARATEURS

a) Au moins un blanc est obligatoire pour séparer l'option "MODIF" de la valeur du sélecteur. Des espaces sont aussi autorisés avant "MODIF" sur la carte comme :

1) MODIF ␣ SELECTEUR *

2) ␣␣␣␣ MODIF ␣␣␣␣ SELECTEUR *

b) "*" caractère astérisque

Il sert à séparer la valeur du SELECTEUR et l'IDENTIFICATEUR qui vient à la suite comme :

␣␣␣ MODIF ␣␣␣ SELECTEUR * ␣␣␣ IDENT =

Il peut également indiquer la fin d'une suite de valeurs :
dans le cas d'objets de type simple, c'est une valeur et dans le cas d'objets de type file ou structure, "*" suite la dernière valeur de la suite comme :

1) ␣␣␣ IDENT ␣␣␣ = ␣␣␣ VALEUR *
↑

2) IDENT1 ␣␣ = ␣␣ VALEUR1 ␣␣␣, ␣␣␣ VALEUR2, ␣␣␣ VALEUR3 ␣␣ *
↑

Les blancs(espaces) entre VALEUR et (,) ou (*) sont autorisés.

La fin de modification pour l'enregistrement demandée est repérée par deux astérisques consécutifs :

IDENT ␣␣␣ = ␣␣␣ VALEUR * *.
↑↑

La virgule ", "

Elle sert à séparer les différentes valeurs dans le cas où l'identificateur est de type file ou structure : (le dernier élément est suivi d'un "*").

Des espaces sont autorisés entre une valeur et le séparateur qui la suit, comme :

IDENT ␣␣ = ␣␣ VALEUR1 ␣␣, ␣␣ VALEUR2 ␣␣ , ␣␣ VALEUR3 ␣␣ *

le signe égal "="

Il sert à séparer un identificateur de la suite des valeurs correspondantes. Des espaces sont autorisés avant et après :

␣␣␣ IDENT ␣␣ = ␣␣␣ VALEUR1 *
↑

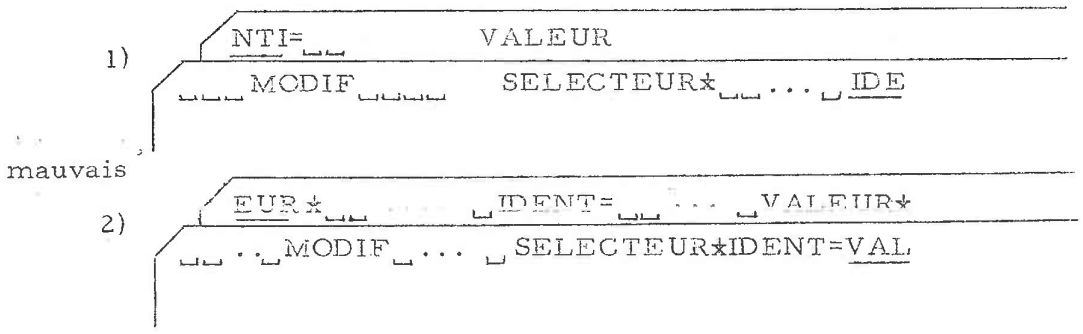
Remarque 1 :la taille maximum d'un (IDENTIFICATEUR) est huit caractères. En cas d'erreur (plus de huit caractères) la modification sera abandonnée par le processeur avec un message d'erreur (voir détection et message d'erreur).

Remarque 2 : la taille de la clé (SELECTEUR) est fixée conformément aux règles du système BPM sur 10070 et la taille maximum qu'on choisit est de 11 caractères. En cas d'incompatibilité, le processeur émet un message d'erreurs (voir détection et messages d'erreur) et il passe au traitement de l'enregistrement suivant.

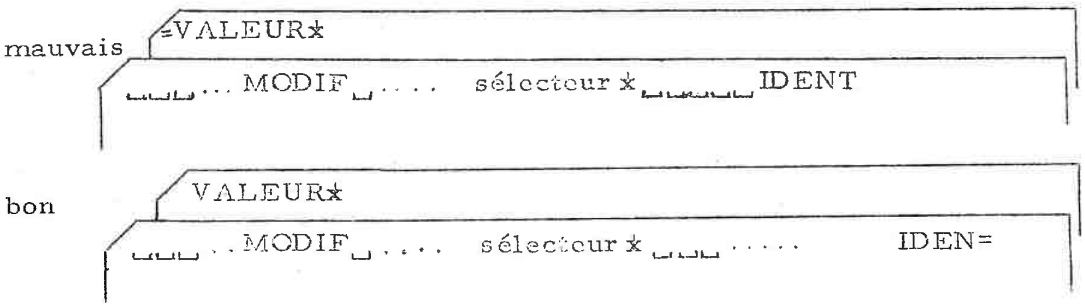
II-4-3 CONDITION D'EMPLACEMENT DES OBJETS SUR LA CARTE

Comme il est dit auparavant, le fichier de modifications sera préparé sous la forme d'une chaîne de caractères. La première carte est toujours une carte d'appel de processeur avec éventuellement comme paramètres (SELECTION, nom). Cette carte est valable pour la modification de tous les enregistrements de ce fichier. La chaîne de modifications pour chaque enregistrement qui commence par "MODIF" suivi par le SELECTEUR, précédé éventuellement d'espaces, se termine par deux "*" qui peuvent être sur plusieurs cartes. La fin de la carte ne sera pas prise en considération.

Il y a quelques restrictions que l'utilisateur devra respecter : on ne peut pas couper les identificateurs (MODIF, SELECTEUR, IDENTIFICATEUR et VALEUR) pour continuer sur la carte suivante par exemple les deux formes :



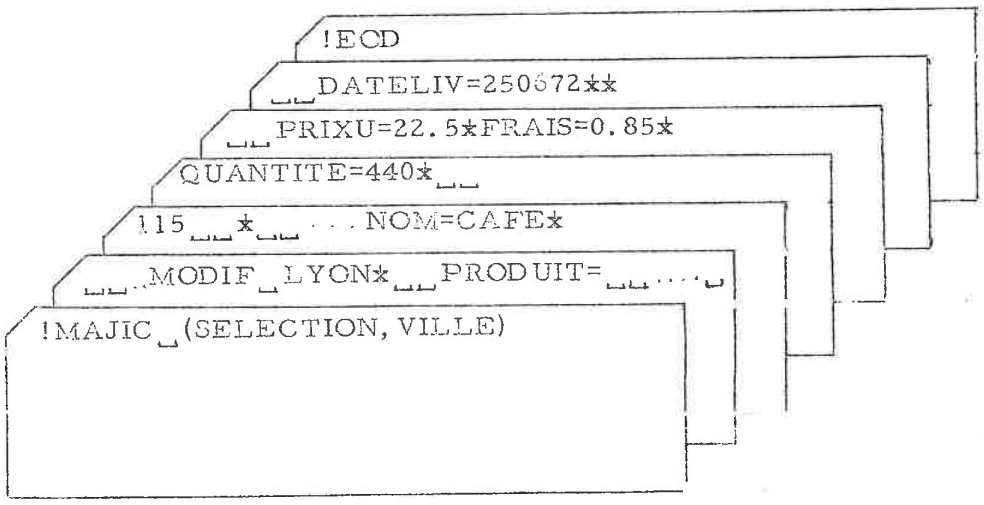
Autrement dit, chaque identificateur et le séparateur qui le suit doivent obligatoirement être sur la même carte comme



Dans les deux cas ci-dessus, si le processeur trouve une incompatibilité, il émet un message d'erreurs et il cherche l'enregistrement suivant à traiter Voici deux exemple de jeux de cartes de fichier de modifications.

Exemple 1 :

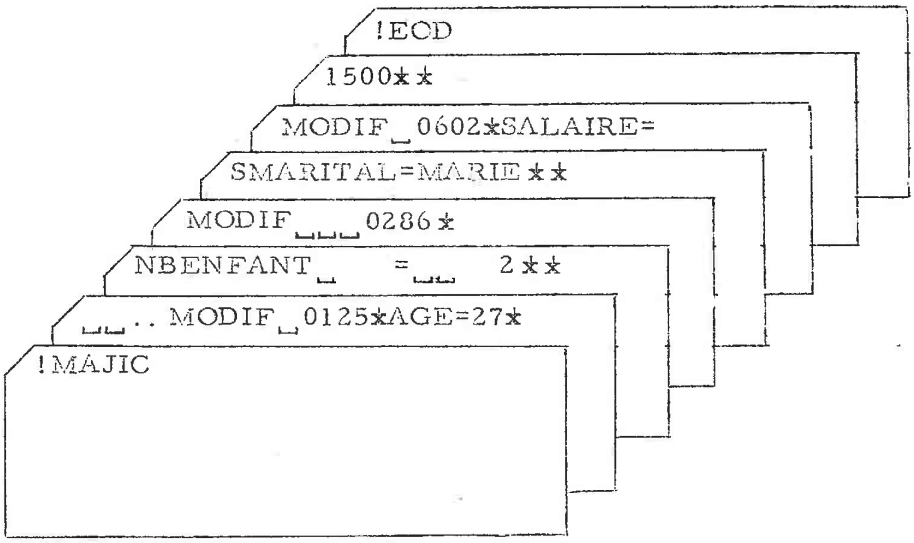
Modification des informations d'un enregistrement (sur un fichier en accès séquentiel)



Exemple 2 :

Modification des informations dans un fichier
(fichier en accès direct)

Dans cet exemple, on représente un fichier qui a modifié les objets qui appartiennent aux enregistrements avec les sélecteurs (clé) :
0125, 0286, 0602.



II-5 DIAGNOSTIC D'ERREURS ET MESSAGES CORRESPONDANTS

Le processeur MAJIC au cours de l'analyse et de l'exécution d'une mise à jour reconnaît un certain nombre d'erreurs d'utilisation et il émet un message pour chacune d'elles. Ce diagnostic se fait à l'aide d'un sous-programme prévu dans MAJIC.

II-5-1 Erreurs rencontrées pendant l'analyse

a) "OPTION INEXISTANTE "

erreur syntaxique sur "MODIF"

Exemple : manque de caractère de séparateur.

␣␣␣MODIF sélecteur *

b)

"LE CARACTERE DE LA FIN DE DECODAGE INEXISTANT"

c'est dans le cas d'incompatibilité ou d'abscence de caractères de la fin (les séparateurs, cf. paragraphe II-4-2) on aura le message ci-dessus.

II-5-2 Erreurs sur la longueur :

1 : si la longueur de la clé (SELECTEUR) dépasse 11 caractères le processeur émet le message :

" CLE TROP GRANDE "

2 : De même si la longueur d'identificateur est plus grande que 8 caractères on aura le message

" IDENTIFICATEUR TROP GRAND "

II-5-3 Erreurs sur l'existence

1 : On aura le message :

" ENREGISTREMENT INEXISTANT "

si on ne trouve pas la clé demandée parmi les clés des enregistrements.
2 : On aura le message :

" IDENTIFICATEUR INEXISTANT "

si le processeur ne trouve pas l'identificateur demandé parmi la liste des identificateurs fournis par le descripteur de classe dans laquelle il est déclaré.

II-5-4 ERREURS DE TYPE QUELCONQUE

a) On aura le message :

" UTILISATION D'* ILLEGALE "

si ce caractère (*) apparaît au milieu du texte MODIF ou IDENTIFICATEUP

b) Comme il est dit dans le paragraphe (II-4-3), on ne peut pas couper les identificateurs. Si le processeur trouve une telle situation il émet le message :

" IDENTIFICATEUR COUPE "

c) Dans le cas où on trouve au milieu d'une valeur du type entier ou décimal un caractère différent d'un chiffre, on aura le message :

" CODE NON NUMERIQUE "

Dans tous les cas qu'on vient de citer le processeur émet d'abord le message correspondant, puis il prépare le message :

d)

"LA MODIFICATION N'EST PAS FAITE POUR L'ENR. sélecteur"

A la fin de ce message, il fournit la valeur du SELECTEUR d'enregistrement. A l'aide de ces deux messages il a localisé l'erreur et de même l'utilisateur saura que la modification ne se fait pas sur enregistrement repéré par SELECTEUR, donc l'ancienne version d'enregistrement est toujours valable. Il passe pour traiter la modification pour un autre enregistrement si il est demandé.

e) "RECHERCHE IDENTIFICATEUR SUR LA CARTE SUIVANTE"

Ce message est donné par le processeur. Cela signifie que le programme est en déroulement.

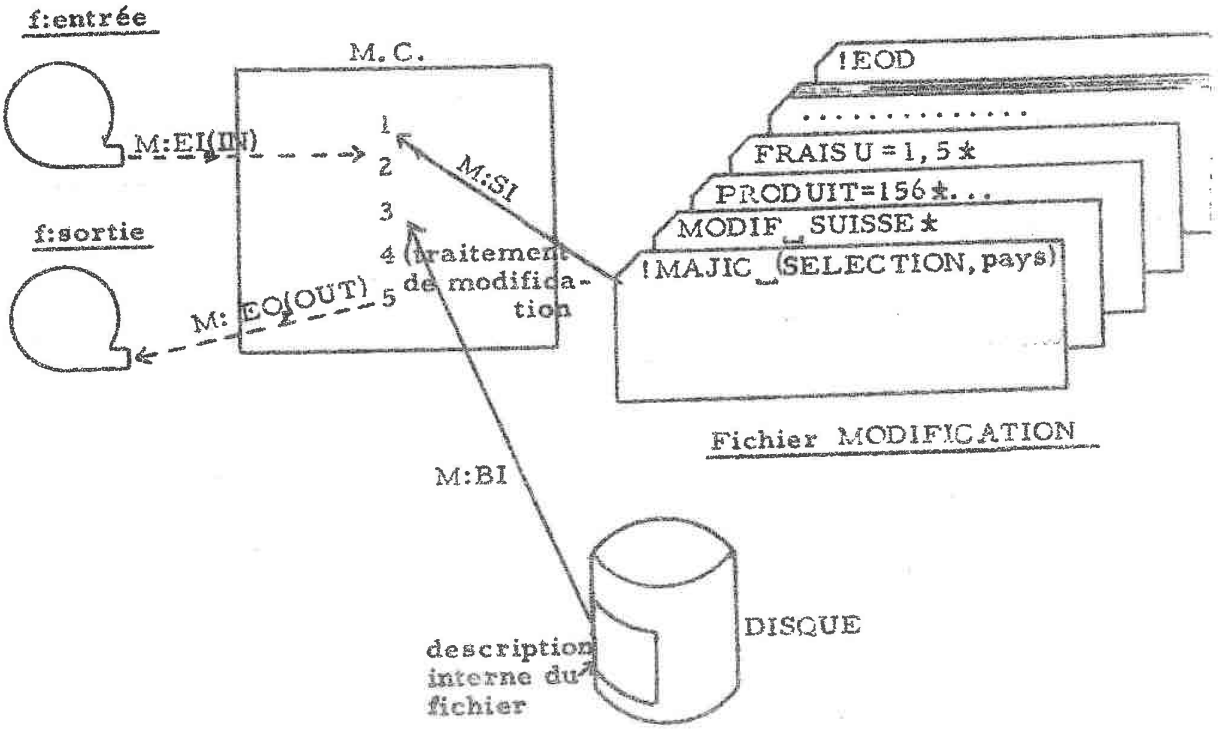
f) Si toutes les modifications se passent sans erreur pour un enregistrement le processeur imprime le message :

"FIN DE MODIFICATION POUR CET ENR. _ SELECTEUR"

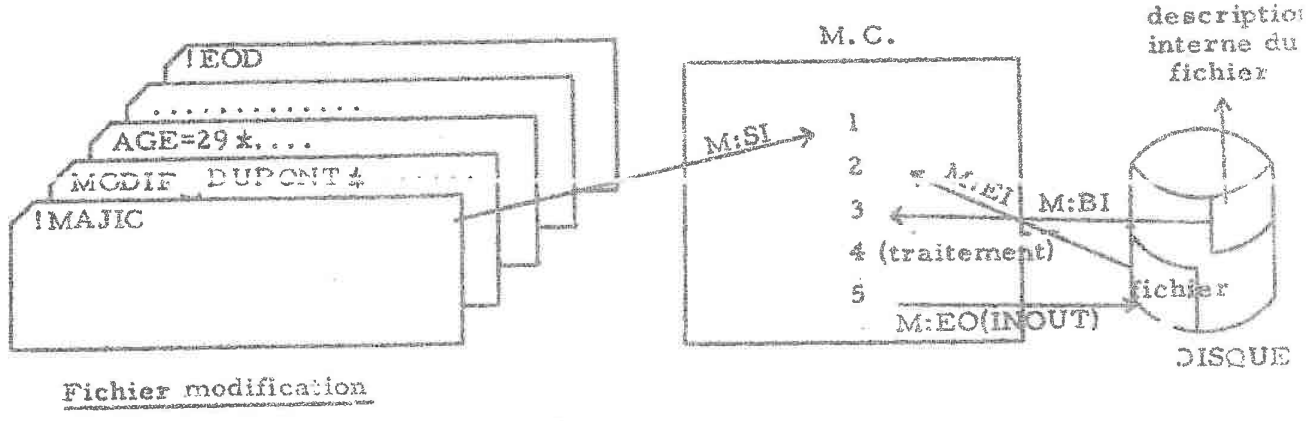
Dès cet instant, l'ancienne version de l'enregistrement n'est plus valable : la nouvelle version a pris sa place dans le fichier.

On peut schématiser le traitement du processeur MAJIC par la figure ci-dessous :

I - Cas du fichier en accès séquentiel



II - Cas du fichier à clé en accès direct (disque)



II-6 REALISATION DU PROCESSEUR MAJIC

MAJIC est un programme modulaire et nous allons décrire ces modules (ou sous-programmes).

La plupart de ces sous-programmes sont du type (ouvert). Les entrée-sorties et les messages se font par les procédures BPM.

I-6-I Paramètres communs de MAJIC

R10 = registre de liaison à l'intérieur de SSP RECHERCH.

R11 = registre de liaison entre le programme principal et RECHERCH.

IDEN = une zone de 8 mots mémoire (mémoire de travail).

B = une variable de sélection qui, chargée dans le programme principal, désigne les différents identificateurs comme : l'option, le sélecteur, l'identificateur et la valeur.

nom	valeur	option(MODIF)	sélecteur	identificateur
(B)	0	1	2	3

PARAM1 := variable de début de décodification.

PARAM2 := variable de fin de décodification.

MT1 := variable servant à garnir l'adresse du caractère de la fin de décodage, en cas de recherche des valeurs.

ZRECTAB:= une zone de 3 mots mémoire servant à transférer le nom d'identificateur au SSP2 pour déterminer son type et sa localisation.

ZENTAB := 1 mot de mémoire pour le paramètre de sortie de SSP2.

ZLEC := une zone de 20 mots mémoire (buffer de lecture).

TAMPON := une zone réservée de 4K-mots mémoire pour les enregistrements à traiter (zone du travail d'enregistrement).

BER := variable servant à la détermination du texte du message d'erreur d'une valeur entière négative comprise entre -1 et -15.

VECR := vecteur des renseignements de 10 mots mémoire utilisé surtout pour le traitement des valeurs.

ACCLE := zone de rangement de la valeur du sélecteur lue.

II-6-2 Module 1 (Programme principal)

But

- Lecture des cartes du fichier modification. Le fichier est repéré par l'étiquette opérationnelle M:SI. Récupération du paramètre sélecteur.
- Positionnement de B (variable de sélection) pour la récupération des identificateurs.
- Lecture de l'enregistrement concerné à l'aide du sélecteur.
- Lecture de la description d'information.

Conditions initiales

- Le fichier de modification qui est un fichier sur cartes.
- Buffer de lecture de cartes (ZLEC).
- Le fichier à modifier qui est un fichier sur bande ou disque magnétique repéré par l'étiquette opérationnelle M:EI et M:EO.

Communication

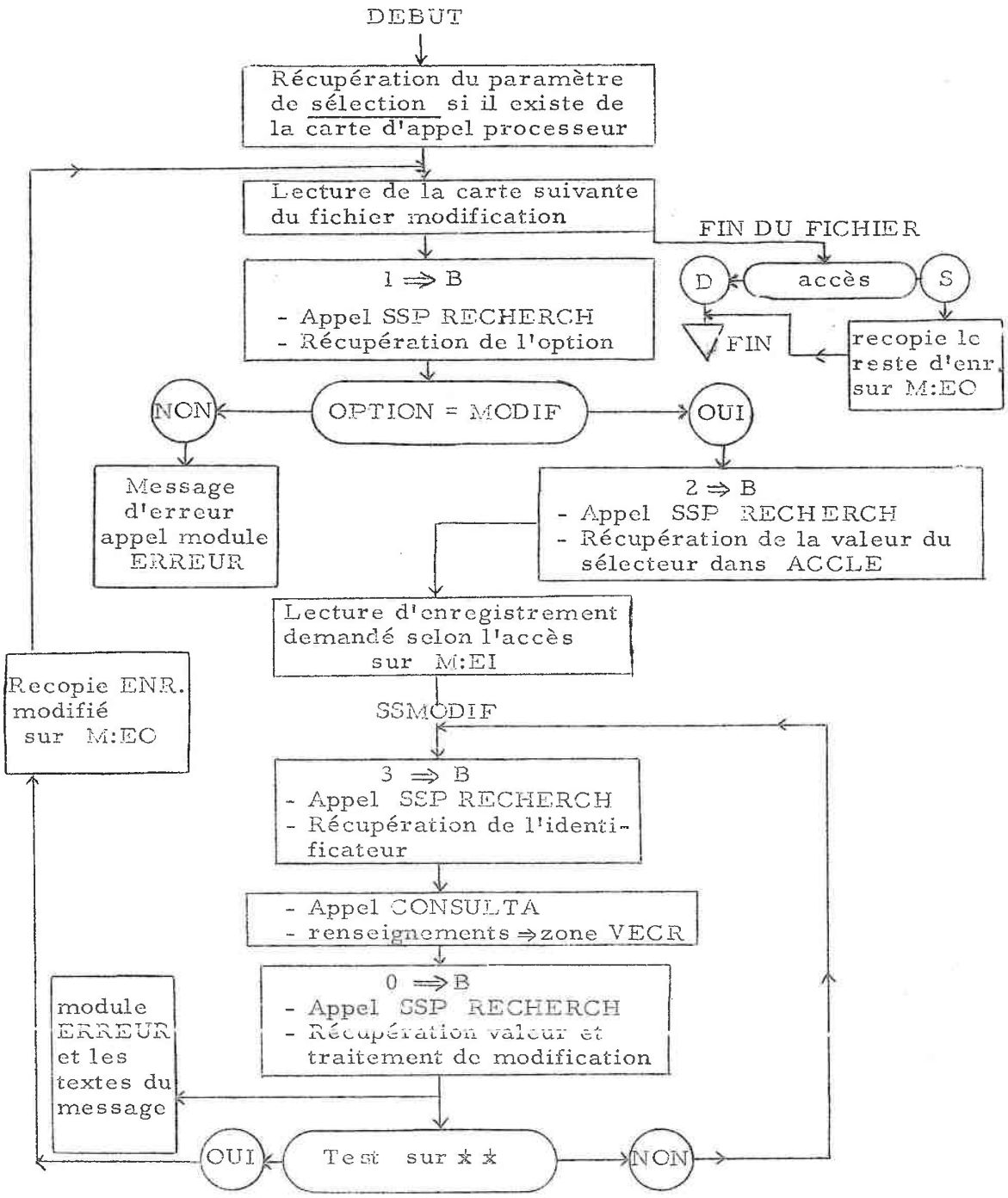
Il communique avec le sous-programme RECHERCH et les modules CONSULTA et ERREUR.

Description

Ses tâches essentielles sont de lire le fichier de modifications : récupération paramètre de sélection de la carte d'appel processeur (1ère carte du fichier modification), si il existe, sinon lecture de la carte suivante et de charger les valeurs (0, 1, 2 ou 3) dans B. Ces valeurs servant à analyser le texte de la carte de modification afin de récupérer l'option, la valeur, le sélecteur ou l'identificateur.

- Le traitement d'erreur à l'aide de module ERREUR et de même la consultation avec le module CONSULTA se fait pour déterminer le type d'identificateur à modifier et sa localisation sur le support et à la fin, on recopie l'enregistrement modifié.

Organigramme général



II-6-3 Module (ERREUR)

But

- Impression du message d'erreur.
- Interruption du traitement des modifications sur l'enregistrement demandé.
- Préparation de traitement d'un nouvel enregistrement à modifier.

Conditions initiales

- La valeur de BER (variable du code d'erreurs).
- La table des messages d'erreurs.
- La chaîne de caractères représentant le sélecteur d'enregistrement (le contenu de la zone ACCLE).

Communication

Ce module est appelé par tous les modules de MAJIC.

Description

BER est une variable qui est positionnée à une valeur négative comprise entre -1 et -15 et en fonction de sa valeur on aura le message correspondant d'erreur. Ce module se compose essentiellement de deux étapes :
I - La valeur de BER qui doit être positionnées avant l'appel au module d'ERREUR, est testé. Si elle vaut -12 ou -15 avant d'imprimer le message correspondant, on transfère la chaîne du tampon ACCLE à la suite du texte de chaque message de la façon suivante :

a) pour (-12) : "FIN DE MODIF POUR CETTE CLE x ..... x " (contenu de ACCLE)

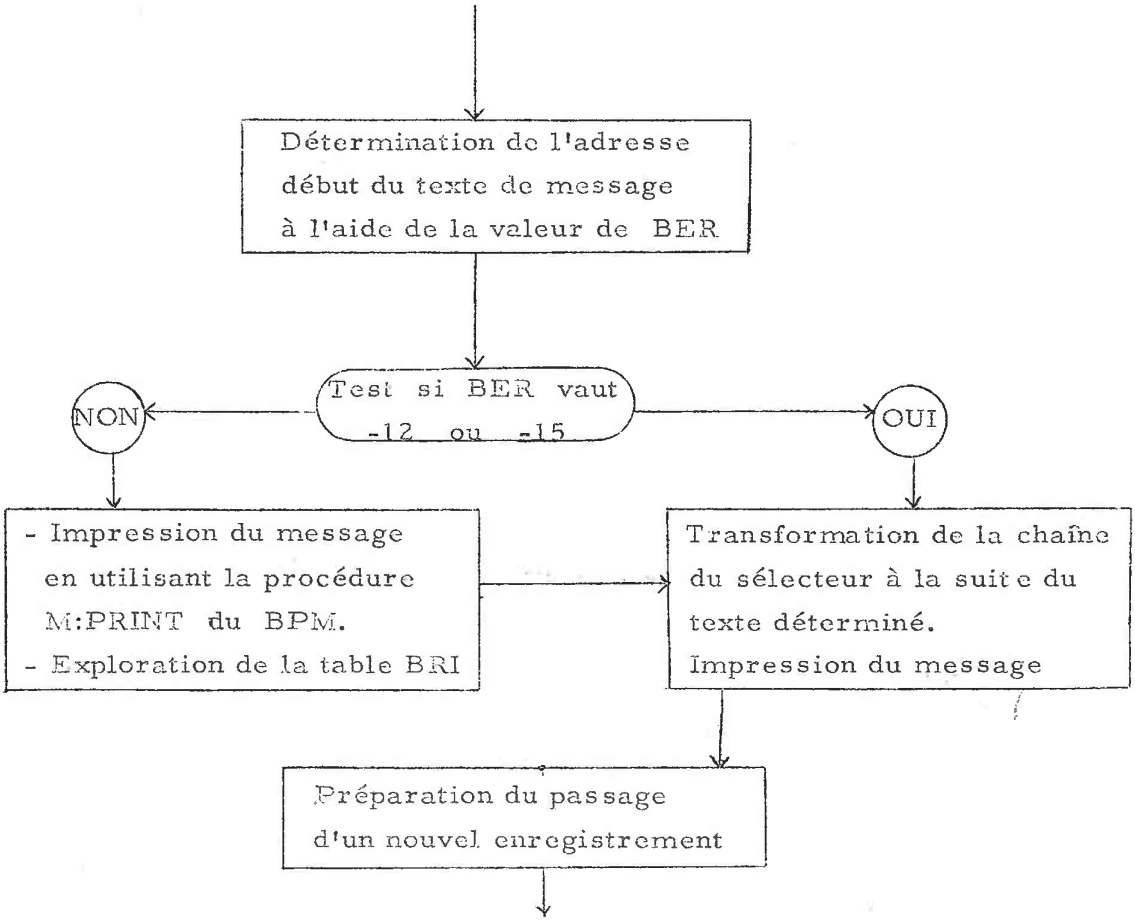
b) pour (-15) : "MODIFICATION N'EST PAS FAITE POUR ENR. xx... xx"

Le premier cas (-12) n'est pas un cas d'erreur, cela veut seulement dire que le processeur a repéré (**) ou la modification pour cet enregistrement est entièrement faite.
Si BER a une valeur autre que -12 ou -15 le processeur passe à l'étape suivante pour imprimer le message d'erreur correspondant, puis il imprime le message de BER=15.

II - Traitement de la table BRI et impression du message.
Table des messages d'erreur et des codes correspondants.
(tous les messages sont sur 48 caractères)

texte / BER	Messages d'erreur
-1	"OPTION INEXISTANTE "
-2	"RECHERCHE IDENTIFICATEUR SUR LA CARTE SUIVANTE"
-3	"IDENTIFICATEUR COUPE "
-4	"IDENTIFICATEUR TROP GRANDE "
-5	"CLE TROP GRANDE "
-6	"TERMINATEUR INEXISTANT "
-7	"CODE NON NUMERIQUE "
-8	"TERMINATEUR TYPE CARACTERE INEXISTANT "
-9	"ENREGISTREMENT INEXISTANT "
-10	"IDENTIFICATEUR INEXISTANT "
-11	"UTILISATION DE * ILLEGALE "
-12	"FIN DE MODIF POUR CETTE CEL "
-13	"ERR. DU TYPE BOOLEENNE "
-14	"ERR. PARAMETER DE SELECTION "
-15	"LA MODIFICATION N'EST PAS FAITE POUR CET ENR. "

Organigramme général



II-6-4 Module 3 (CONSULTA)

But

- Détermination du type, de la localisation et éventuellement de la taille de l'identificateur à l'aide de la description d'information.

Conditions initiales

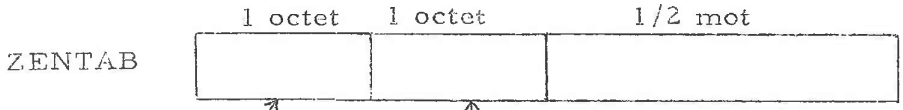
- Les conditions initiales pour ce module sont :
- le nom de l'identificateur
- le sous-programme SSP2

Communication

CONSULTA communique avec les modules SSP2* et STRUCTURE.

Description

Après récupération de l'identificateur (max. 8 caractères), on le transfère cadré à gauche dans la zone ZRETAB (paramètre d'entrée) de SSP2. Puis on fait appel à SSP2.
A la sortie, on aura dans ZENTAB les renseignements suivants.



type :
- s'il est simple ∉ structure ($-9 \leq type \leq -1$)
- sinon 0

1) 2n+1 : si constante
2) 2n+2 : si variable
3) 0 si module
n : n° de la classe dans la table de classe

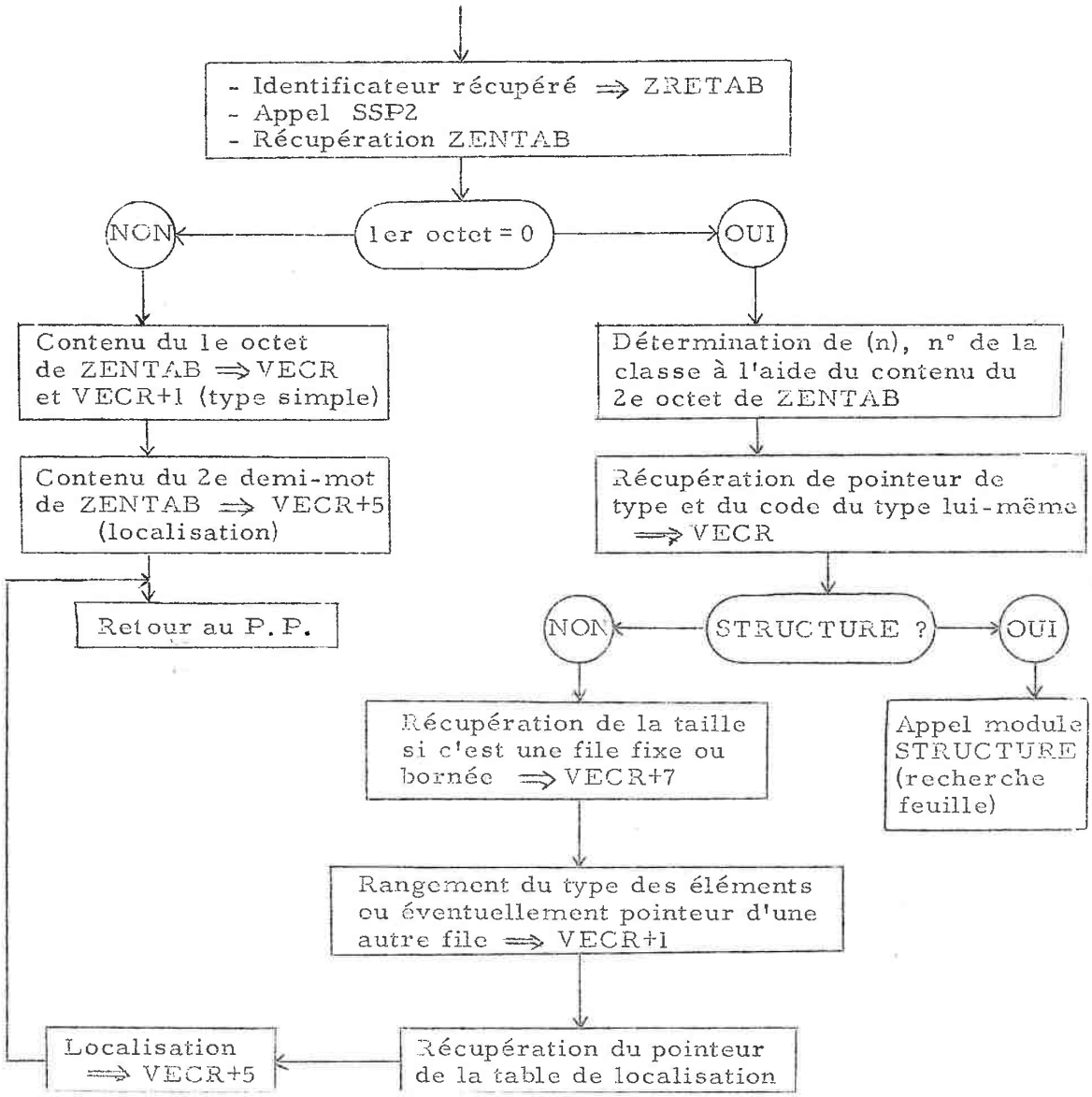
- si 0 dans 1er octet c'est l'adresse du pointeur dans la table des pointeurs.
- sinon localisation/ zone de la classe.

Les informations de genre type, localisation, taille, lien horizontal (LH) et lien vertical (LV) propres à chaque identificateur qu'on a ainsi reçues sont rangées à partir de VECR (zone réservée pour les vecteurs de renseignemen afin de les utiliser dans le module de traitement de modification.

---

* SSP2 est un sous-programme qui a le rôle de déterminer la situation de chaque identificateur dans la chaîne codée.

Organigramme général



II-6-5 Module 4 (SOUS PROGRAMME DE RECHERCH)

But

- Analyse du texte du fichier modification et détermination des identificateurs
- Traitement de la valeur
- Traitement des conversions
- Traitement des modifications

Conditions initiales

Les conditions initiales pour le traitement de ce modules sont :
- la valeur de B est positionnée dans le programme principal.
- buffer de ZLEC (tampon de lecteur de carte).

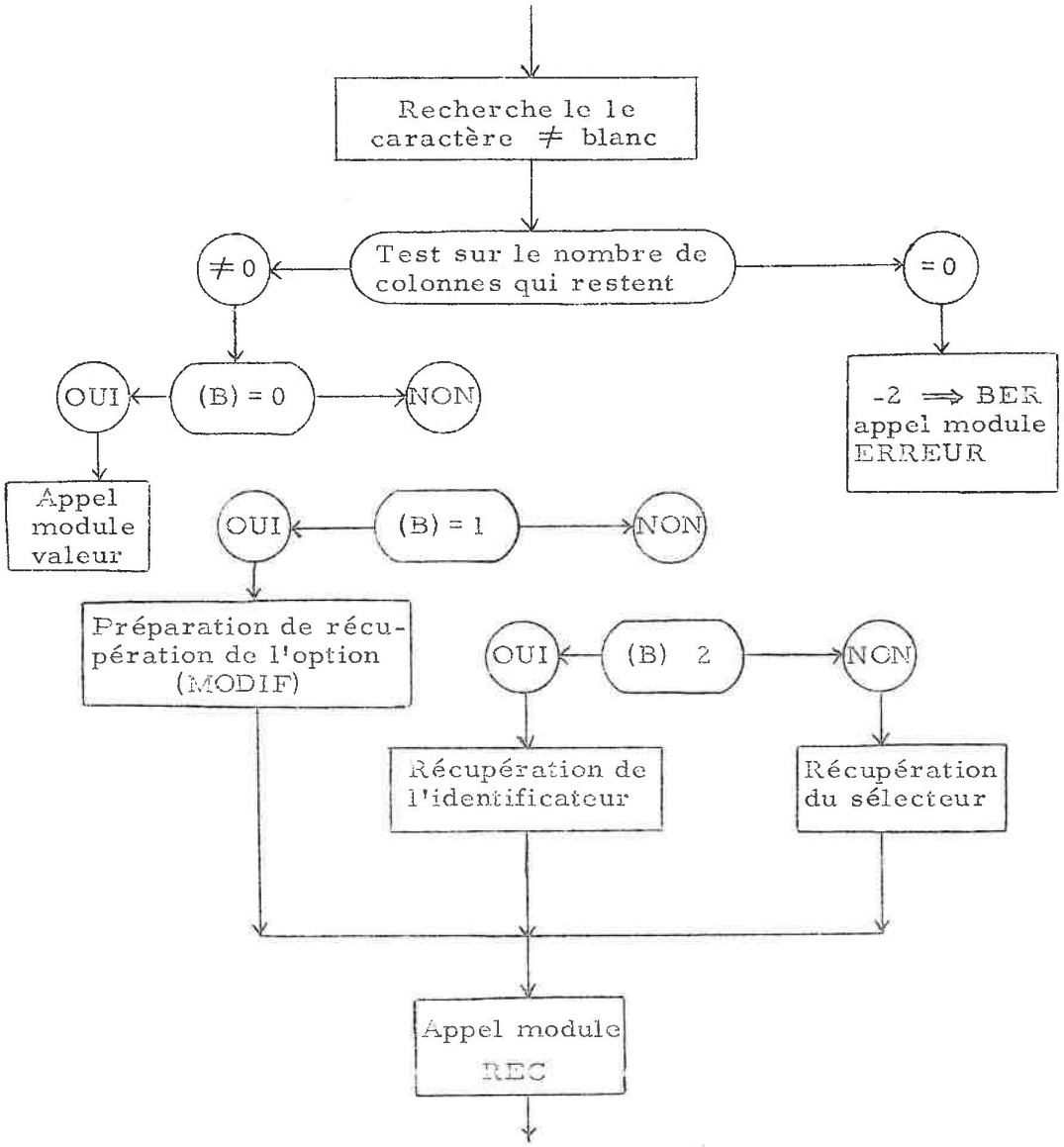
Communication

RECHERCH appelle les modules d'ERREUR, REC, VALEUR et il est appelé par le programme principal.

Description

En fonction de la valeur de B, RECHERCH appelle l'un des deux modules REC ou VALEUR. Si le nombre des colonnes de la carte lue dans ZLEC devient zéro, on met dans BER la valeur -2 et puis on se branche au module ERREUR.

Organigramme général



II-6-6 Module 6 (REC)

But

Détermination et récupération : l'option MODIF et rangement dans IDEN, l'identificateur et rangement dans ZRETAB et le sélecteur dans ACCLE.

Conditions initiales

Les conditions initiales pour traitement de ce module sont :
- (PARAM1) := l'adresse du 1er caractère différent de blanc
- (PARAM2) := le caractère de la fin de décodage
- la valeur de B.

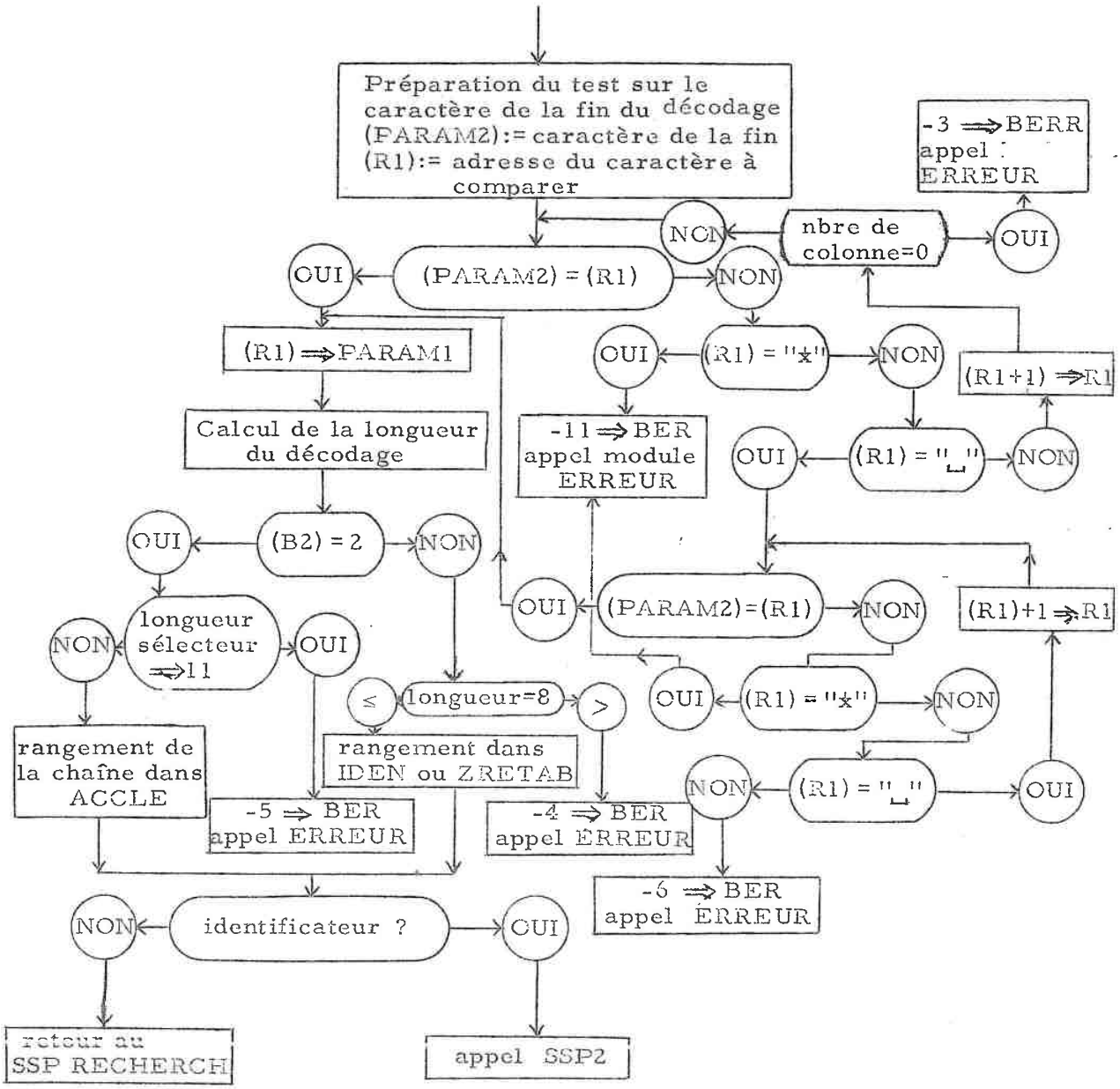
Communication

REC appelle le module ERREUR et SSP2. Il est appelé par le sous-programme RECHERCH.

Description

On teste d'abord le caractère de fin de décodage (*), (=) ou (␣) , de même en fonction de la valeur de B et la longueur de l'identificateur récupéré, on distingue l'option, l'identificateur et le sélecteur. A la fin, on range la chaîne récupérée dans IDEN , ZRETAB ou ACCEL.
En cas d'incompatibilité, d'abord BER est positionné et puis on appelle le module de traitement d'erreur.

Organigramme général



II-6-7 Module 7 (VALEUR)

But

En fonction du type de l'identificateur, on sélectionne le type de traitement de modification (conversion et rangement des informations à modifier).

Conditions initiales

Ce module a besoin pour son traitement :

- la valeur du code type (C(VECR)) et le type d'éléments, dans le cas d'objets du type file (C(VECR+1)).
- la localisation et éventuellement la taille d'objet à traiter.

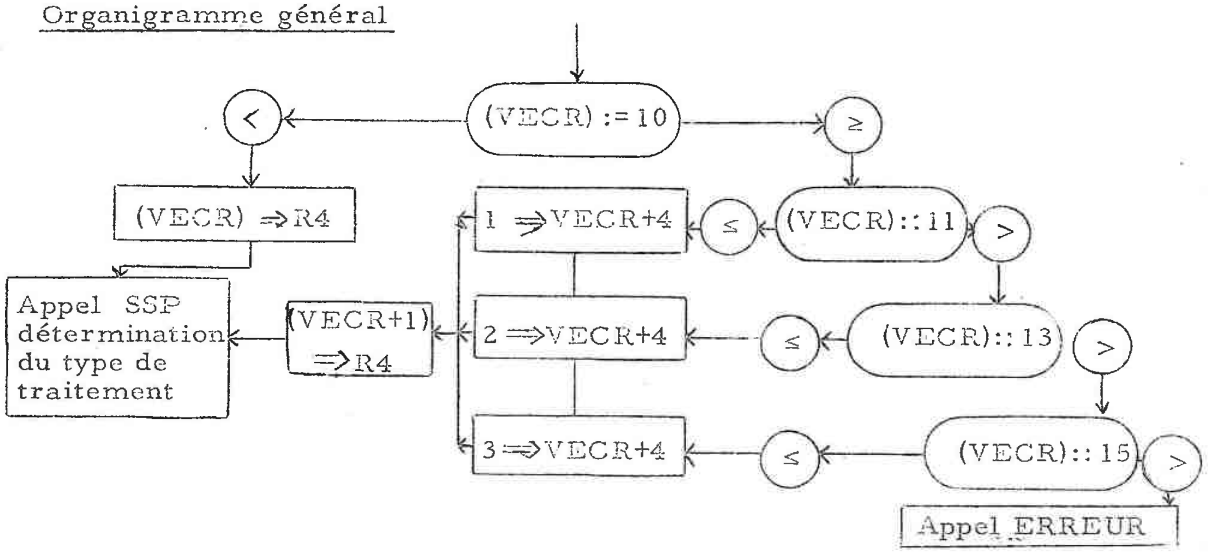
Communication

VALEUR appelle les modules REC1 et ERREUR . Il est appelé par le module RECHERCH.

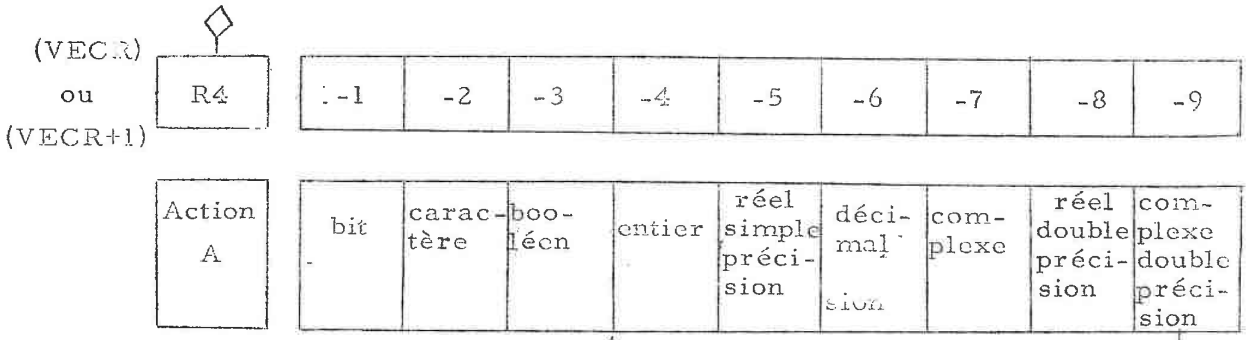
Description

C'est un module de sélection. Au cours du déroulement de ce module, on distingue les types files et simples.
Le type de conversion pour changer la valeur du format externe en format interne.
La sélection sera faiteà l'aide de C(VECR) et C(VECR+1) qui sont positionnés dans le module CONSULTA. Le C(VECR) est une valeur qui donne la possibilité de distinguer les objets du type file de simple.
Le C(VECR+1) détermine toujours le type des éléments dans le cas de file.
Dans ce module, on fait aussi appel à un petit sous-programme qui sélectionne le type de traitement, sous-programme SSP.

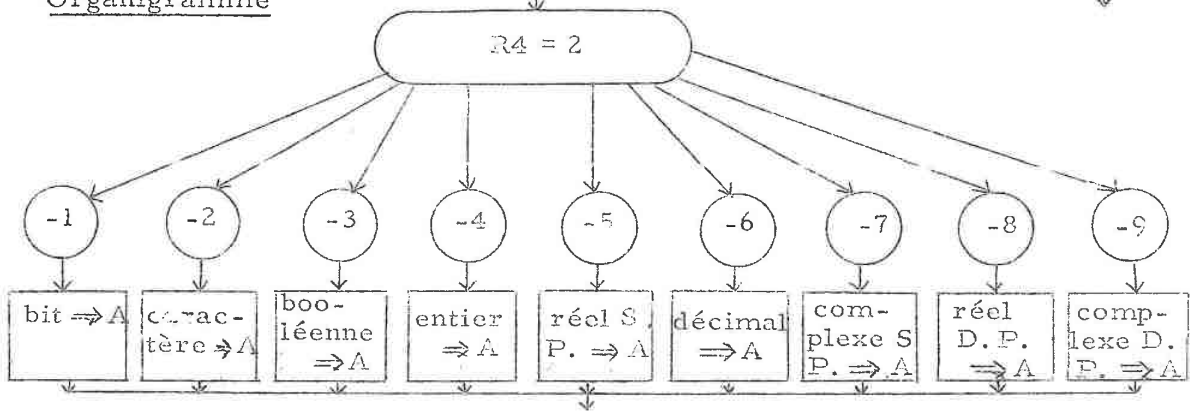
Organigramme général



SSP : C'est un ensemble de tests sur la valeur de C(VECR) ou C(VECR+1). Après entrée dans la table de sélection, on aura un branchement vers un traitement particulier soit entier, réel simple précision, réel double précision, caractère, booléen, décimal, ...



Organigramme



II-6-8 Module 8 (REC1)

But

- Conversion de la valeur entier en binaire
- Conversion de la valeur décimal en forme interne (PACK)
- Conversion de la valeur réel (simple précision ou double précision) en forme interne.

Conditions initiales

Ce module doit avoir à sa disposition :
- Adresse début de la chaîne à convertir
- La valeur du type d'élément C(VECR+1)

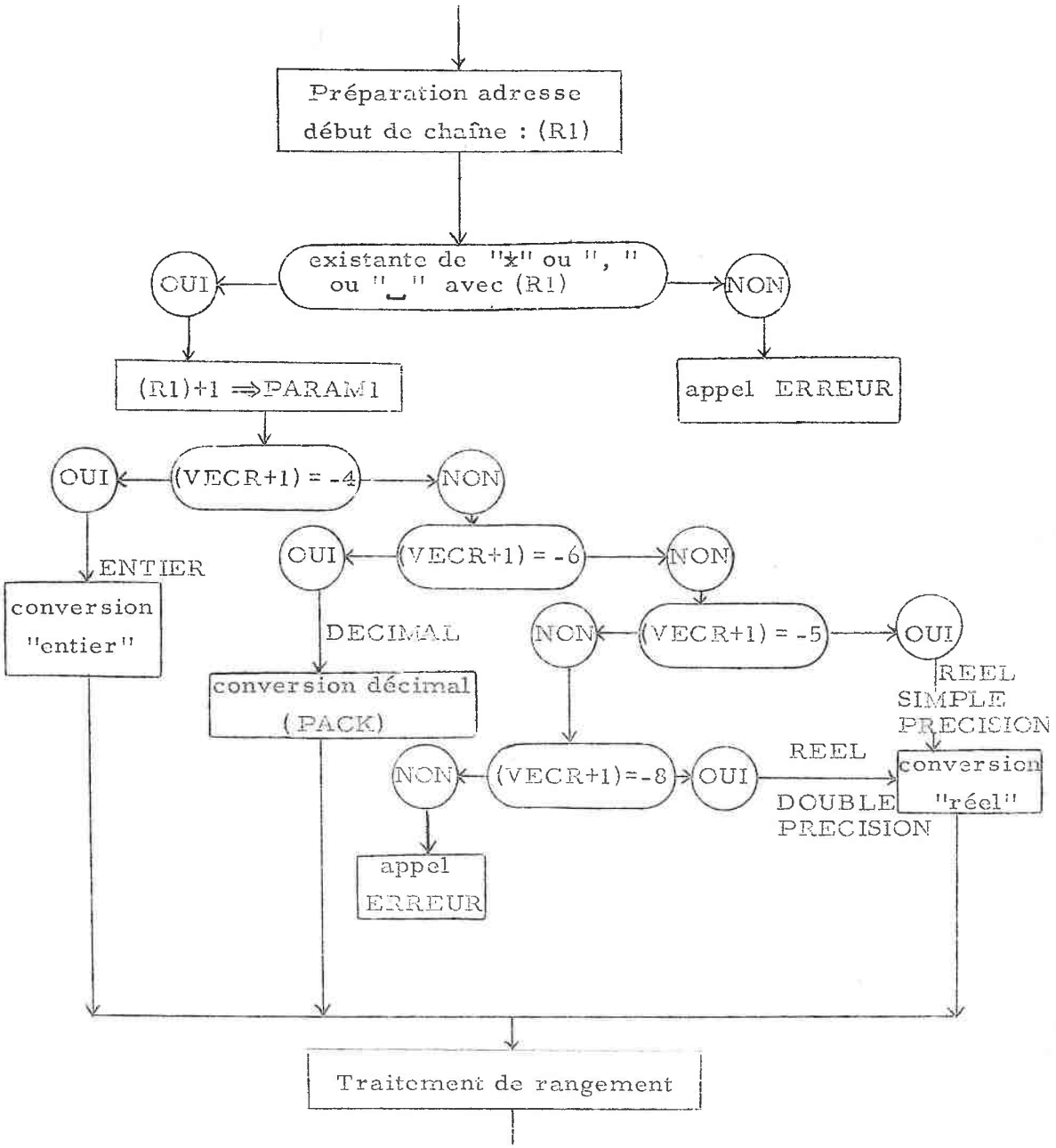
Communication

Il communique avec les modules ERREUR, RED et VALEUR.

Description

A l'aide de ce module, on fait d'abord une décodification sur le caractère de la fin de la partie "valeur", c'est à dire qu'on distingue un élément du type simple d'un élément du type file. Ensuite, en fonction de la valeur de C(VECR+1), on choisit l'une des trois tâches : entier, décimal, réel, pour la conversion correspondante.
A la fin, on effectue le rangement.

Organigramme général



II-6-9 Module 9 : STRUCTURE (Recherche feuille)

But

- Traitement sur des objets de type structure.

Conditions initiales

Ce module a besoin pour son traitement :
- la table de description du type des éléments
- rang de l'identificateur, défini à l'aide de nom de structure (nom identificateur).

Communication

STRUCTURE communique avec le module CONSULTA et RECHERCH.

Description

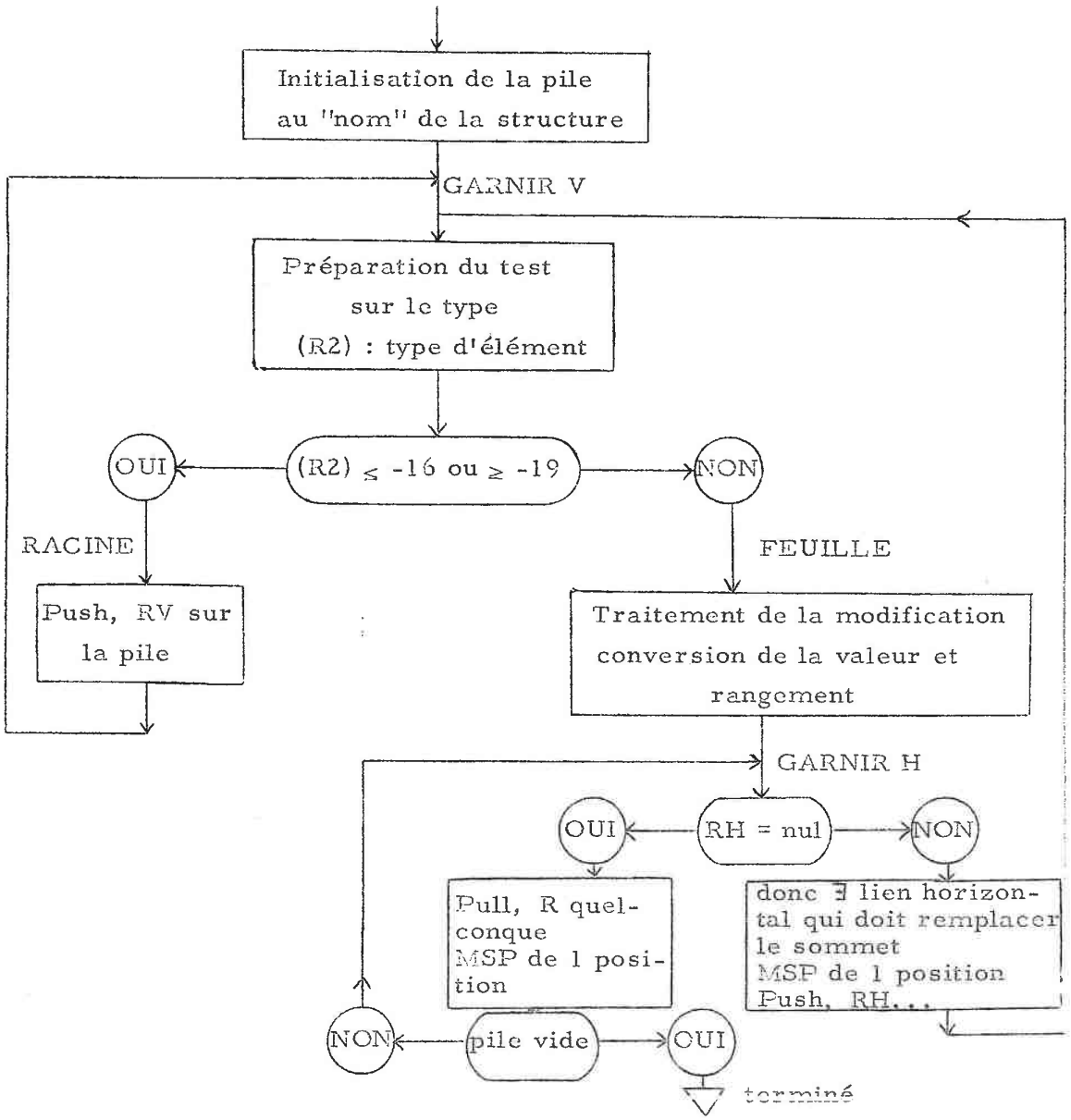
Dans une représentation arborescente, assurée par le descripteur de type, un objet de type structure se représente à l'aide d'un lien horizontal et d'un lien vertical. Si on n'a pas de lien vertical, il s'agit d'une feuille qu'on doit exploiter, sinon c'est un noeud.

Dans STRUCTURE , pour exploiter l'arbre de notre structure, on utilise une pile. Au début, on initialise la pile avec le rang du nom de la structure, puis on teste le type d'objet. Si c'est différent de (-16, -17, -18 ou -19) c'est un objet non structuré, donc on fait le traitement correspondant à cet objet. Ensuite, on teste le lien horizontal. S'il vaut zéro, on fait le traitement prévu d'empilement, sinon on met ce lien horizontal au sommet de la pile et on continue.
Si le type d'objet est (-16, 17, 18 ou -19), on empile le lien vertical et on continue dans le rang correspondant.

RV utilisé comme support de lien vertical.

RH utilisé comme support de lien horizontal.

Organigramme général



On peut aussi traduire le traitement du module STRUCTURE à l'aide de la table de décision comme suit :

I : est un indicateur qui dit si l'opération précédente sur la pile est une entrée ou une sortie de la pile.
Initialisation de la pile ; I : entrée ;

I = entier	y	y	y	N	N
$-16 \leq R2 \geq -19$	y	N	N	-	-
RH = 0	-	y	N	-	N
Pile vide	NON	N	N	y	N

Push, RV	1				
MSP de 1		2	2	1	1
I:=entrée			4		
I:=sortie Pull,		3			
Push, RH			3		2
RE	2	4	5		3
traitement feuille		1	1		
terminé				2	

CHAPITRE III

Mise à jour du TEXTE SOURCE

"CORRECTION TEXTE"

CORTEX

SOMMAIRE

III-1 MODULE CORTEX (CORRECTION TEXTE)

Ce module permet la mise à jour du texte source d'un programme écrit en civa ou en n'importe quel langage (Symbol, Métasymbol, Fortran, Cobol, ...) et rangé en mémoire secondaire (bande magnétique, disque, etc..)

Le module CORTEX est un programme METASYMBOL qui, comme processeur autonome, permet les modifications suivantes du texte source d'un programme :

1) SUPPRESSION
2) ADJONCTION
3) REMPLACEMENT

Les modifications sont effectuées par un fichier sur cartes appelé fichier modifications.

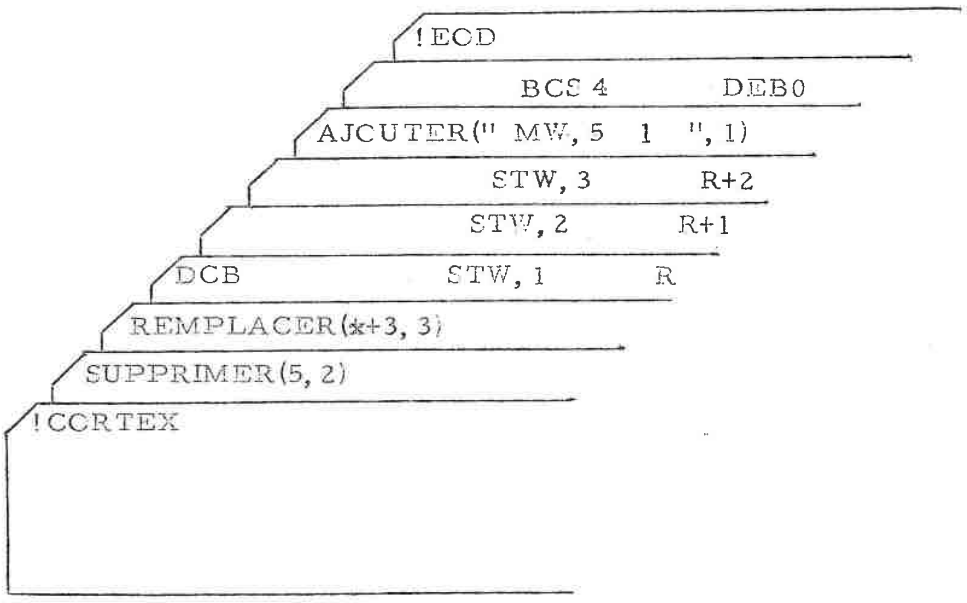
"CORTEX" ne touche pas au contenu des lignes du programme à modifier, mais il supprime, remplace ou ajoute des lignes complètes.

III-1-1 Exemple

Un programme symbol (représenté page III-5, à gauche) est sur disque ou bande magnétique et on a l'intention de le modifier de la façon suivante :

1er : supprimer les lignes numéros 5 et 6.
2e : remplacer les lignes numéros 10, 11 et 12 par les contenus de trois cartes du fichier modifications.
3e : ajouter après la ligne dont le contenu est : MW, 5 1 un nouveau texte (sur une carte donnée du fichier modifications).

Le fichier modifications sera représenté sous la forme suivante :



Les versions des deux textes du programme source avant et après mise à jour sont données ci-dessous :

L'ANCIENNE VERSION

(Avant mise à jour)

```
00 TAB      DATA    X'01000000'
01          DATA    X'80000000'
02          DATA    MESS
03 R        RES     5
04 ADX      RES     1
05 APPEL    RES     1          2 lignes
06 APPEL1   RES     1          à supprimer
07 ADB      RES     1
08 NB       RES     1
09 CAD      DATA    X'F0'
10  DCB     LW,1    R
11          LW,2    R+1        3 lignes
12          LW,3    R+3        à modifier
13          STW,4   R+3
14          STW,5   R+4
15          LW,1    *13
16          STW,1   ADX
17          AI,13   1
18          LW,1    *13
19          STW,1   ADB
20          AI,13   1
21          LW,1    *13
22          STW,1   NB
23          AI,13   1
24          LI,1    1
25          LI,3    0
26          LW,2    ADB
27          AW,2    NB
28          LW,4    NB
29 B0UC     AI,2    -1
30          LB,5    0,2
31          CI,5    X'60'      ***
32          BE      PF
33          SW,5    CAD
34          MW,5    1
35          AW,3    5
36          BCS,4   DEB0
37          MI,1    X'A'
38          BDR,4   B0UC
39          B       FIN
40 DEB0     CAL1,2  TAB
41          B       $+3
42 PF       MI,3    -1
43 FIN      STW,3   *ADX
44          LW,1    R
45          LW,2    R+1
46          LW,3    R+2
47          LW,4    R+3
48          LW,5    R+4
49 MESS     TEXTC   'ERR DEB0RD DCB '
50          B       *13
```

LA NOUVELLE VERSION

(Après mise à jour)

```
TAB      DATA    X'01000000'
         DATA    X'80000000'
         DATA    MESS
R        RES     5
ADX      RES     1
ADB      RES     1
NB       RES     1
CAD      DATA    X'F0'
DCB      STW,1   R
         STW,2   R+1        3 lignes
         STW,3   R+2        remplacée
         STW,4   R+3
         STW,5   R+4
         LW,1    *13
         STW,1   ADX
         AI,13   1
         LW,1    *13
         STW,1   ADB
         AI,13   1
         LW,1    *13
         STW,1   NB
         AI,13   1
         LI,1    1
         LI,3    0
         LW,2    ADB
         AW,2    NB
         LW,4    NB
B0UC     AI,2    -1
         LB,5    0,2
         CI,5    X'60'      ***
         BE      PF
         SW,5    CAD
         MW,5    1
         BCS,4   DEB0       1 ligne
                            ajoutée
         AW,3    5
         BCS,4   DEB0
         MI,1    X'A'
         BDR,4   B0UC
         B       FIN
DEB0     CAL1,2  TAB
         B       $+3
PF       MI,3    -1
FIN      STW,3   *ADX
         LW,1    R
         LW,2    R+1
         LW,3    R+2
         LW,4    R+3
         LW,5    R+4
MESS     TEXTC   'ERR DEB0RD DCB '
         B       *13
```

III-2 REPRESENTATION DU FICHIER MODIFICATION (et MODE D'UTILISATION

La modification et la mise à jour d'un texte source seront demandées par une option qui s'écrira sur une ou plusieurs cartes.

OPTION(LOCALISATION, nombre N2 de buffers à traiter)
[dans la suite "buffer" ou "tampon" sera utilisé à la place, soit d'une ligne du texte source, soit d'une carte du fichier modification].

III-2-1 OPTION (commande)

Les options seront soit AJOUTER (adjonction), SUPPRIMER (suppression) ou REMPLACER (remplacement). Le nombre de caractères de chaque option est fixé :

7 caractères pour la première et 9 caractères pour les deuxième et troisième. En cas d'incompatibilité en cours d'analyse de la carte option, le programme en question est rejeté par le processeur avec un message d'erreur.

III-2-2 LOCALISATION

Pour déterminer l'endroit où l'on veut faire le traitement, on a prévu trois sortes de localisation.

III-2-2-1 <RANG>

C'est le numéro de la première ligne du buffer à traiter avec les options SUPPRIMER ou REMPLACER ou du buffer après lequel on ajoute . donc c'est un nombre entier positif décimal (RANG $\in \mathbb{N}$).

$0 \leq$ RANG $\leq$ le numéro de la dernière ligne du texte source.

Exemple :

OPTION(9, N2)

Si OPTION est SUPPRIMER, on supprime N2 lignes à partir de la neuvième comprise. Si OPTION est REMPLACER, on remplace N2 lignes à partir de la neuvième tandis que, avec .AJOUTER , on ajoute entre la neuvième et la dixième ligne N2 cartes .

III-2-2-2 <*> astérisque :

a) On désigne l'emplacement ou l'on est actuellement, autrement dit * désigne (LOC) , compteur d'emplacement. De même, dans la même catégorie de localisation on a prévu :

b) *+α où α est un nombre entier positif décimal (α ∈ ℕ) qui désigne l'emplacement de la ligne à traiter par rapport à * (à (LOC)) autrement dit c'est une adresse relative par rapport à * .

Exemple :

OPTION(*+4, N2)

Par exemple, si * vaut 5, *+4 vaut 9 et le traitement se fait comme avec OPTION(9, N2).

III-2-2-3 <"TEXTE">

C'est un texte du programme source qui est représenté entre les caractères ("). C'est le texte de la ligne qu'on veut localiser. Dans ce cas, la suite du texte peut ne pas tenir sur une carte ; elle se continue sur la carte suivante, le processeur ne tient pas compte des espaces entre les différents morceaux du texte et la taille maximum du texte est 80 caractères.

Exemple : On veut faire une mise à jour dans un programme SYMBOL et on demande :

OPTION ("LW, RX2 TAMPON, RX1", N2)

c'est à dire qu'à l'endroit où se trouve ce texte dans le programme source on fait un traitement de mise à jour en fonction de l'option correspondante.

Remarque : L'utilisateur doit présenter son fichier modification, au point de vue de la localisation, par ordre croissant car le processeur ne recule jamais en arrière.

III-2-3 Description N2

C'est un nombre entier décimal positif qui indique le nombre de cartes à ajouter ou à remplacer par une nouvelle version du texte, ou encore, le nombre de lignes du texte source à supprimer.

III-2-3-1 SEPARATEURS

a) La parenthèse ouverte (sert à séparer l'option et l'argument de localisation. Les espaces ne sont pas autorisés ni avant ni après.

b) La virgule sert à séparer les deux arguments localisation et N2. Les espaces avant et après la virgule sont interdits.

c) La parenthèse fermée c'est un caractère qui sert à la fois pour la fin de décodage sur N2 et aussi pour désigner la fin de spécification de chaque opération de mise à jour.

Remarques :

1 - Les cartes données qui suivent les cartes d'options AJOUTER et REMPLACER sont perforées selon la règle propre du langage du programme source :

2 - Des espaces sur la carte, avant OPTION, sont autorisés à condition que l'option et ses paramètres se trouvent entièrement sur la même carte.

III-2-4 ERREURS et messages correspondants :

Erreurs rencontrées par le processeur pendant l'analyse d'un paquet de mise à jour :

a) Erreur dans le nombre de caractères de la commande

7 pour AJOUTER et 9 pour REMPLACER et SUPPRIMER.

b) Erreur si la parenthèse ouverte (est mal placée ou si elle n'est pas accolée au dernier caractère de la commande.

c) Erreur si la valeur de la commande n'est pas correcte.

On aura le message . unique .

COMMANDE NON RECONNUE

d) En cas d'anomalie ou fin du fichier on aura le message

FIN DU FICHIER ARRET

c) Erreur sur la partie localisation :
si le processeur en cours d'analyse de cette partie ne trouve pas le caractère de fin, c'est à dire la virgule ; on aura le message

TERMINATEUR DE LA LOCALISATION INEXISTANT

e) Erreur sur la partie N2 (nombre de buffer) :

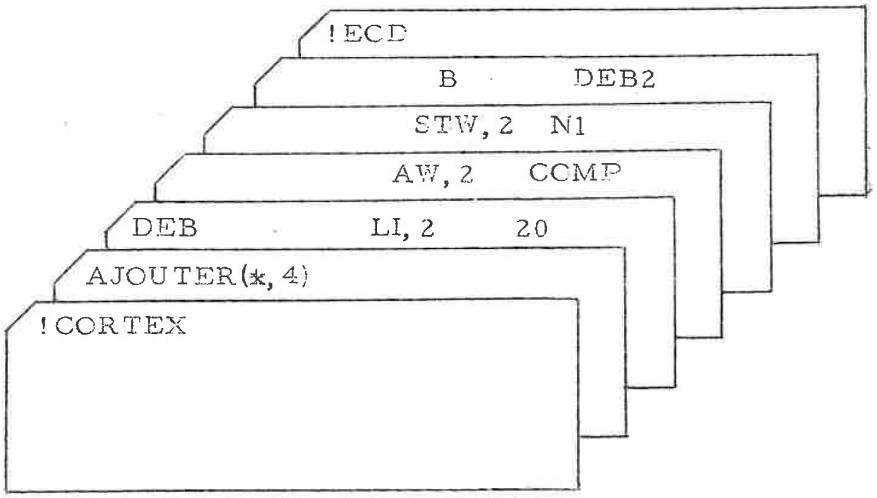
On aura le message :

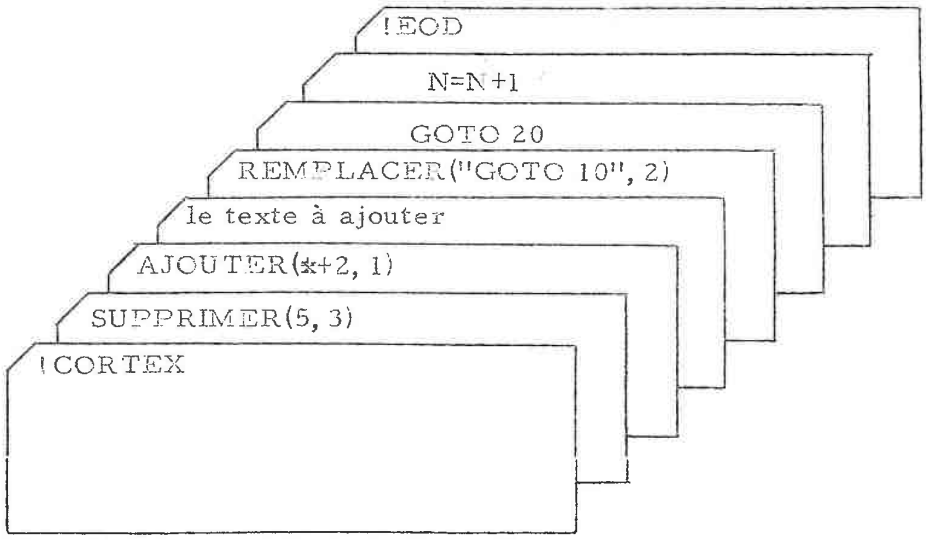
' ERR DEBORD DCB '

si il y a un débordement au moment du décodage sur N2.

Exemple : jeux de cartes de mise à jour

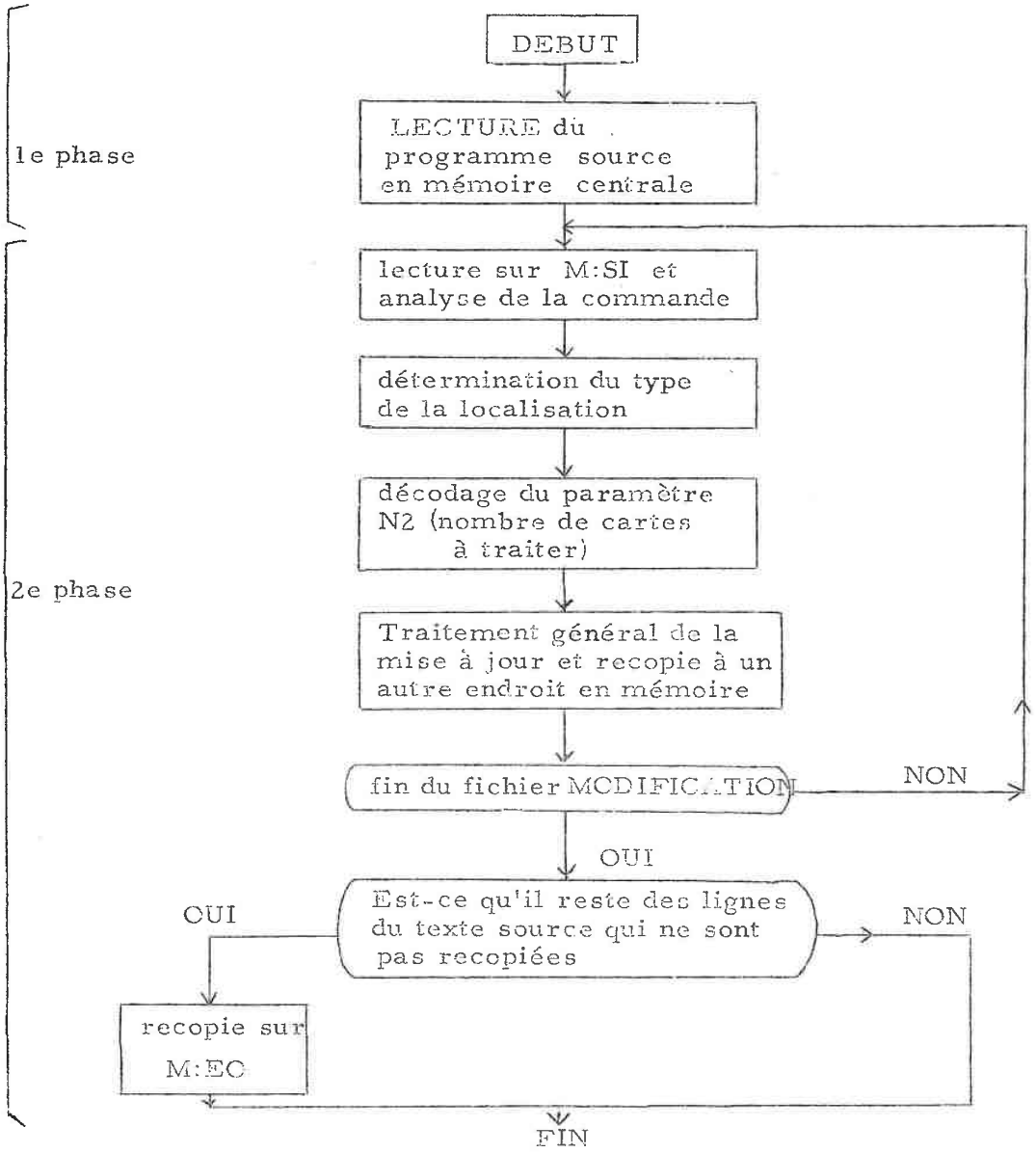
Ci-dessous, on donne quelques exemples de structures usuelles de jeux de cartes de mise à jour.





III-3 REALISATION

ORGANISATION GENERALE DU MODULE CORTEX :



Ce programme est écrit en langage "Symbolique" mais pour faciliter les opérations d'entrée-sortie on a utilisé les procédures de BPM qui sont définies dans le langage METASYMBOL.

Ce programme se compose de deux phases essentielles

a)LECTURE du programme source sur M:EI et mise en mémoire centrale à partir de l'adresse symbolique TEXSOU. En cours de LECTURE on fait un comptage du nombre de lignes du texte source. Le registre 15 est réservé pour la taille du texte source (nombre de lignes) et celui-ci est intouchable pendant l'exécution de la mise à jour.

b) Mise à jour proprement dite.

Dans cette phase, on a deux tâches essentielles :

b-1) Analyse de la carte de modification qui porte la commande de modification. Ses paramètres sont la localisation et N2 : à la fin de cette tâche on connaît le type du traitement, l'emplacement à mettre à jour et le nombre de cartes ou de lignes à modifier.

b-2) Traitement de mise à jour qu'il s'agisse de suppression, remplacement ou adjonction et recopie des lignes du texte source qui ne sont pas touchées .

Les registres 10, 13 sont les registres de liaison entre le programme principal et les sous-programmes :

RECC et sous-programme de conversion (entier décimal en représentation binaire).

CARTE := TAMPON de lecture de la carte de modification

CARTE1 := TAMPON de la carte donnée.

(N1):= rang de la carte du programme source à partir de laquelle on veut faire des modifications (localisation).

(N2) := nombre de cartes ou d'enregistrements à traiter

CARACT := emplacement du caractère qui est sur la carte, au cours de l'analyse.

CARACT := soit CARTE, R1
soit CARTE, R6
(R1) := nombre de caractères explorés à partir de CARTE.
(R6) := nombre de blancs
(TRUC) := adresse de début de la zone localisée
(COMP) := (LOC) ou le compteur d'emplacement. Au début, il vaut zéro mais à chaque fois qu'on fait un traitement de mise à jour sur le texte source, on le met à jour aussi.

III-3-1 REALISATION DU PROGRAMME CORTEX

MODULE 1 : (LECTURE DU TEXTE SOURCE)

But :

- LECTURE du texte source en mémoire afin d'effectuer un traitement de mise à jour à l'aide d'un fichier modification
- comptage du nombre de lignes du texte à traiter

Conditions initiales

- le programme à modifier est sur une mémoire secondaire rapide du type (bande ou disque magnétique)
- TEXSOU adresse de début d'implantation du texte du programme en mémoire
- R15 garni par le nombre de lignes

Communication :

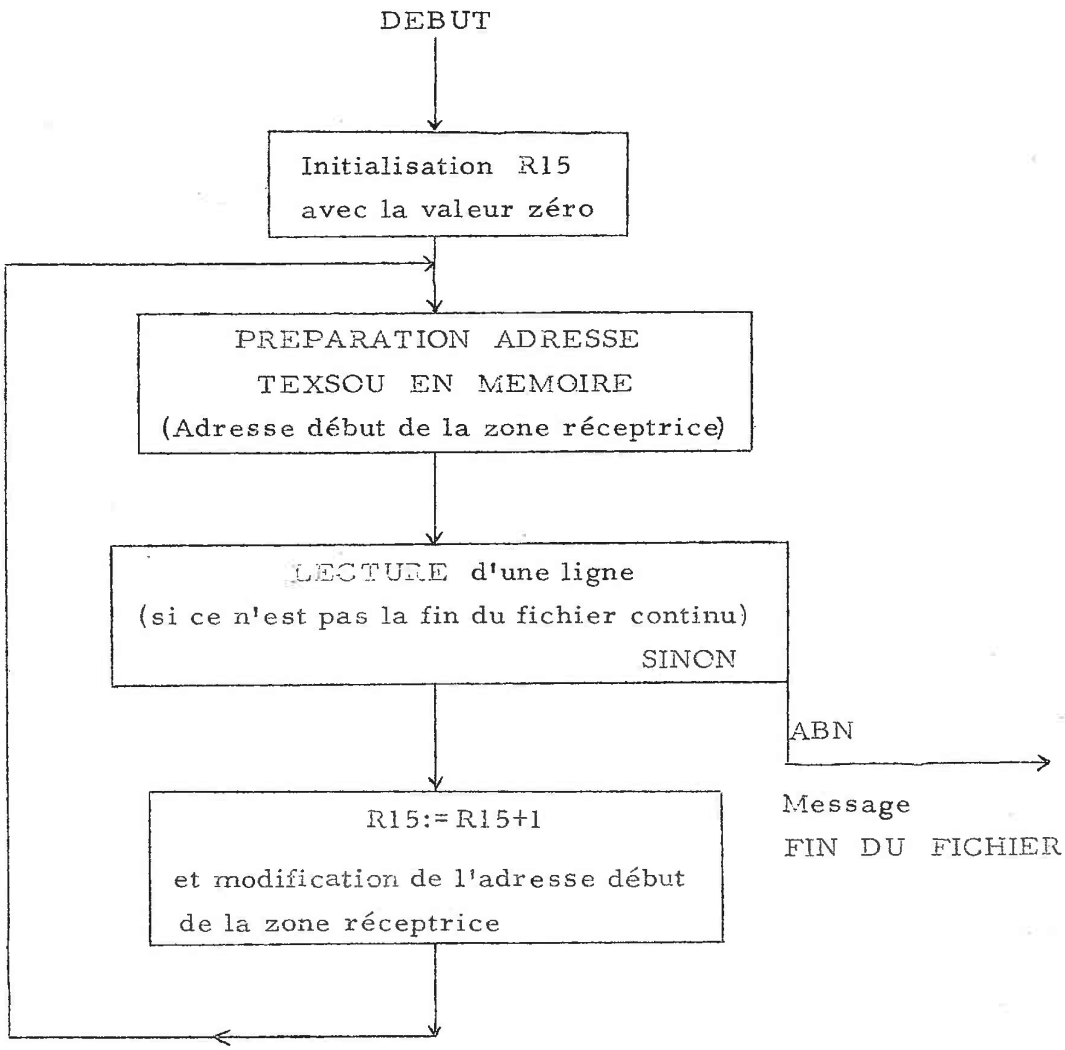
Le module 1 ne communique avec aucun des autres modul

Description

Le rôle de ce module est de lire le texte source qui est en mémoire secondaire sur M:EI , de l'implanter en mémoire centrale à partir de l'adresse symbolique TEXSOU et en même temps de faire un comptage des lignes dans le registre 15.

Le contenu de R15 reste inchangé en cours d'exécution du programme.

ORGANIGRAMME GENERAL



III-3-2 MODULE 2

But

- Lecture d'une carte du fichier modification
- Analyse du texte pour détermination de l'OPTION

Condition initiale

- Le fichier modification est sur cartes .

Communication

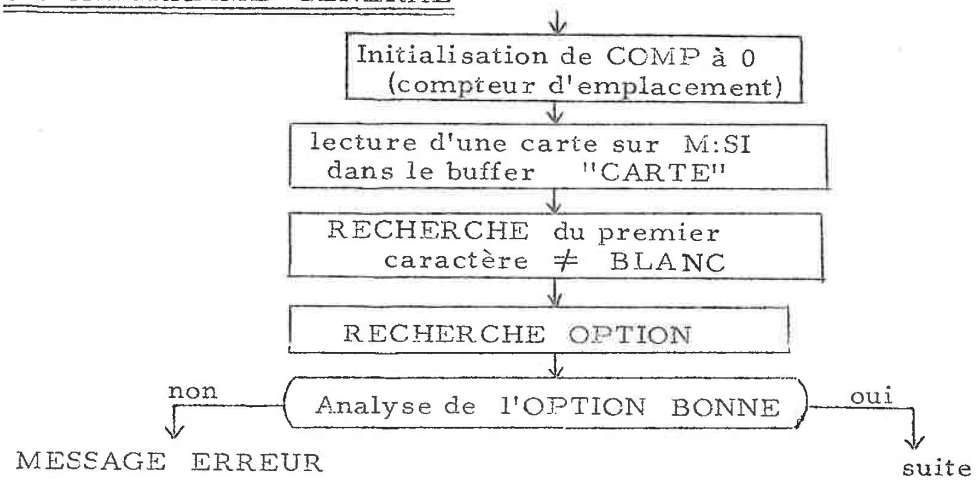
Ce module communique avec le module 3.

Description

- Dans ce module on initialise à zéro le compteur d'emplacement (COMP), on lit une carte du fichier MODIFICATION sur M:SI. Puis on cherche le premier caractère différent de blanc.
Alors on détermine l'option :

(AJOUTER, SUPPRIMER ou REMPLACER)

ORGANIGRAMME GENERAL



III-3-3 MODULE 3

But

- Analyse du paramètre de LOCALISATION (recherche du rang de la ligne du texte source à traiter)

Conditions initiales

- buffer de 'carte'
- adresse relative du premier caractère de la partie localisation par rapport au début de "CARTE'
- virgule (caractère de séparation)
- TAMPON (zone de 20 mots mémoire garnie par le texte compacté)
- APPEL et APPEL+1 (deux mots mémoire pour transfert du paramètre de décodage)
- N1 (mot qui contiendra la valeur de la localisation au rang de la ligne en mémoire en binaire)

Communication

Il appelle le module (7) DCB (décimal codé binaire)

Il est appelé par le module (6) RECO (5), (TRAITEMENT DE MISE A JOUR) et module 2.

Description

Sa tâche consiste en différents traitements sur le paramètre de localisation selon le type :

a) Si cet argument est du type <rang> le traitement consiste seulement en décodage à l'aide du module DCB et localisation du rang de la ligne à trait dans N1 en forme binaire.

b) Si cet argument est du type <*> : le traitement est de ranger la valeur de COMP dans N1 (COMP) ⟹ N1

b-1) Si il est du type <*+α> :

le traitement consiste d'abord en une conversion sur la valeur α qui est un nombre entier décimal positif en appelant le module DCB, puis on ajoute la valeur obtenue dans N1 à la valeur de COMP :

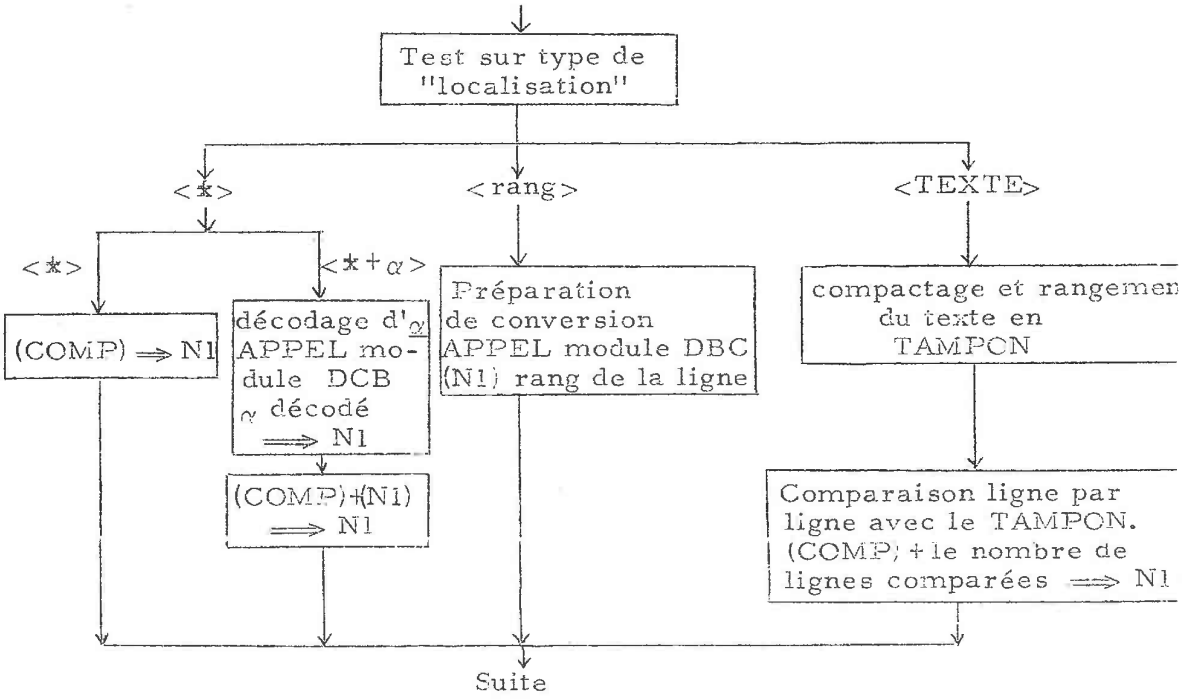
(COMP) + (N1) $\Longrightarrow$ N1

c) Si il est du type <TEXTE> :

on fait d'abord un compactage et rangement du <texte> de la carte dans la zone TAMPON et puis la comparaison entre deux chaînes de caractères d'une part la chaîne dans TAMPON et d'autre part le texte de chaque ligne du programme source.

- Pour la comparaison on commence toujours à partir du rang dans COMP . Dans ce cas N1 contient la somme de la valeur de COMP et du nombre de lignes comparées.

ORGANIGRAMME GENERAL



III-3-4 MODULE 4

But

- Décodage du deuxième paramètre de la carte de Commande.

Conditions initiales

- adresse relative de début de la chaîne à décoder par rapport au début du buffer "CARTE"
- la parenthèse) caractère de fin de décodage
- Appel1 et APPEL1+1 (deux mots de mémoire pour transfert des paramètres)
- N2 (mot qui contiendra le nombre de lignes ou cartes à traiter en binaire)

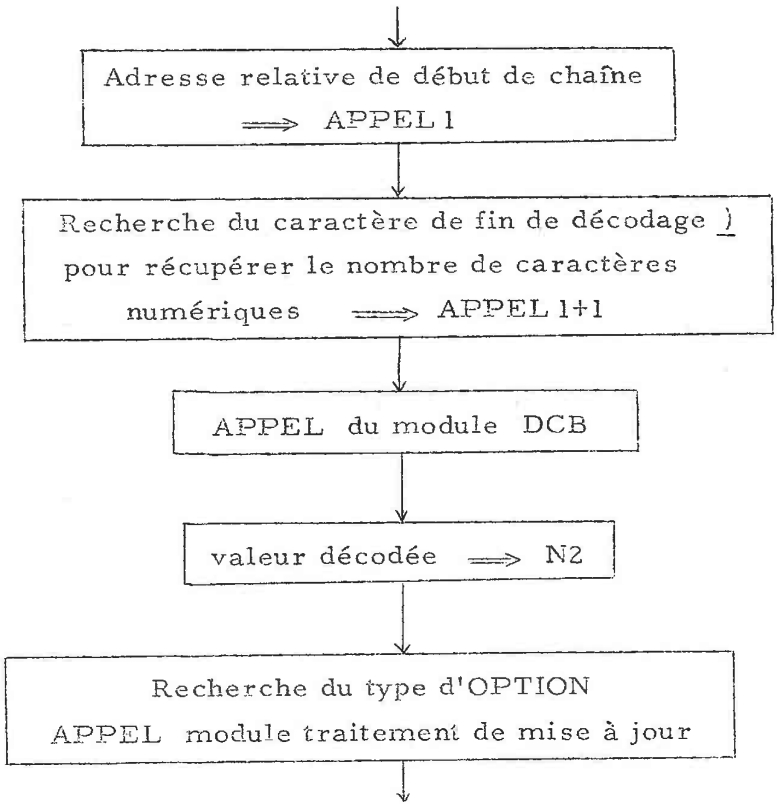
Communication

Il appelle le module (7) DCB et module (5) (traitement de mise à jour).

Description

Cet argument décodé est un nombre entier décimal positif donc le seul traitement qu'on doit faire est la conversion de cet argument en forme exécutable pour l'ordinateur (en binaire) et puis détermination de l'option exacte.

ORGANIGRAMME GENERAL



III-3-5 MODULE 5 : TRAITEMENT DE MISE A JOUR

But

- Traitement général de mise à jour et recopie du nouveau text

Condition initiale

- valeur de N1
- valeur de COMP
- valeur de N2

Communication

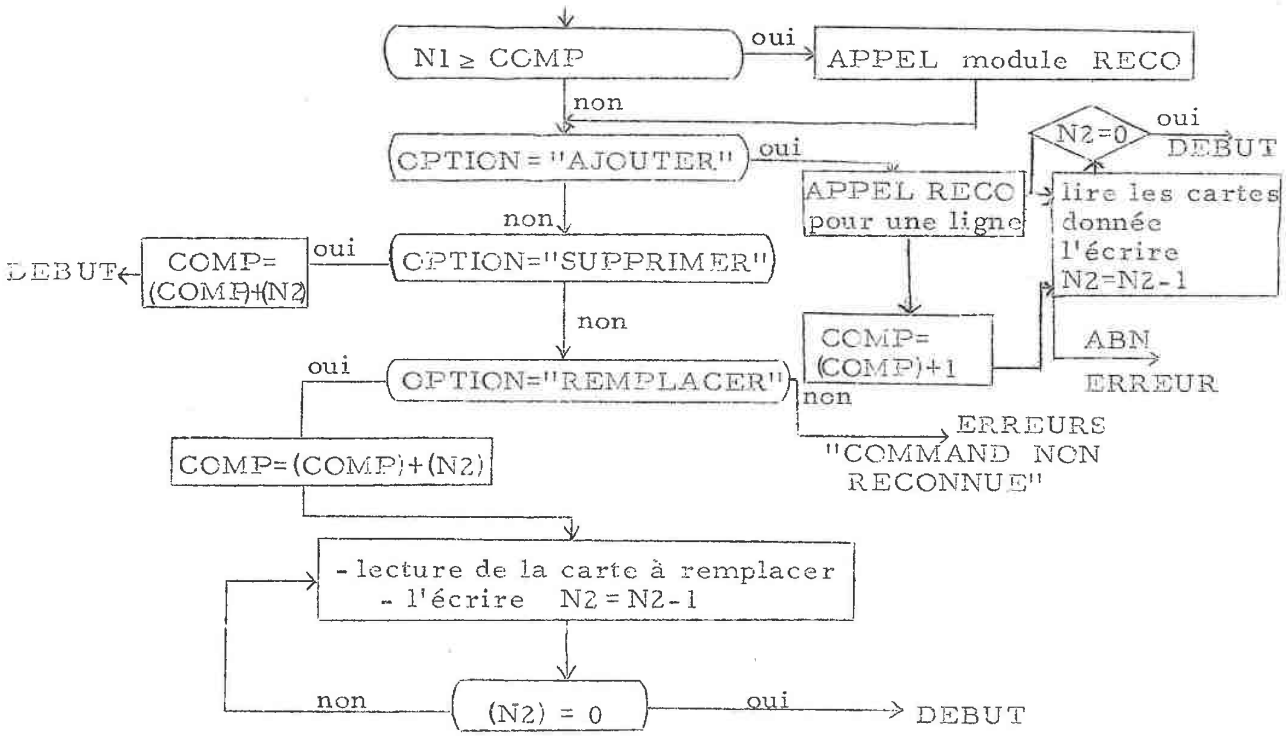
Il appelle le module RECO et il est appelé par le module (4).

Description

Dans cette partie, avant tout, on doit recopier les lignes du texte source qui se trouvent entre (N1) et (COMP), puis faire les traitements de mise à jour selon le type d'option comme suit :

- en cas de suppression, mise à jour de COMP avec la valeur N2.
- en cas d'adjonction, on recopie d'abord une ligne puis mise à jour de COMP et à la fin lecture des cartes données et écriture de ces cartes.
- pour remplacement, on fait d'abord la mise à jour de COMP, puis la lecture des cartes données et remplacement aux lignes correspondantes.

ORGANIGRAMME GENERAL



III-3-6 MODULE 6 : RECO

But

- Traitement de la recopie des lignes du programme source qui restent inchangées.

Condition initiale

- valeur de N1
- valeur de COMP
- adresse début d'implantation du texte source en mémoire TEXSOU.

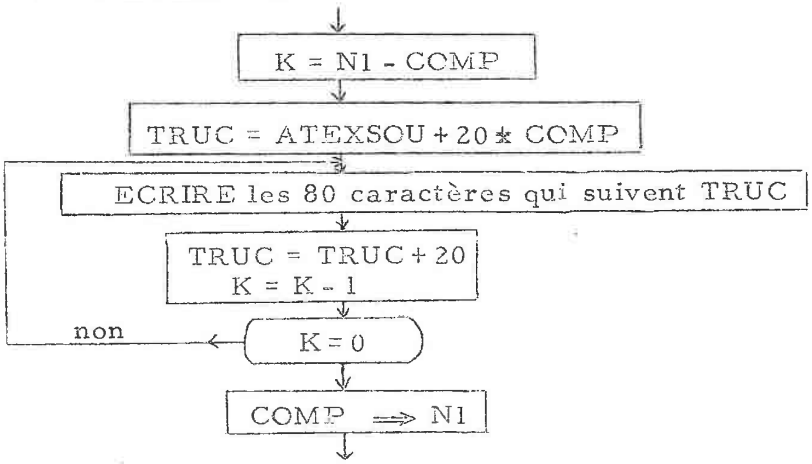
Communication

Il appelle le module (3) et le module (1) et il est appelé par le module (5) (traitement de mise à jour).

Description

Dans ce module on fait le traitement de recopie des lignes de programme source en fonction de la différence entre la valeur de N1 et la valeur de COMP (l'ancienne valeur du compteur d'emplacement) et l'adresse début de la zone en mémoire du programme en cours.

ORGANIGRAMME GENERAL



III-3-7 MODULE 7 (DCB)

But

- Conversion d'un nombre entier décimal en binaire

Condition initiale

- adresse début de la chaîne à convertir
- nombre de chiffres à traiter

Communication

Il est appelé par le module (3) et le module (4)

Description

En général, dans deux cas distincts en cours d'exécution du programme CORTEX on doit faire la conversion.

a) si le type de localisation (module 3) est <RANG> ou <*+α>, pour obtenir la valeur de N1.

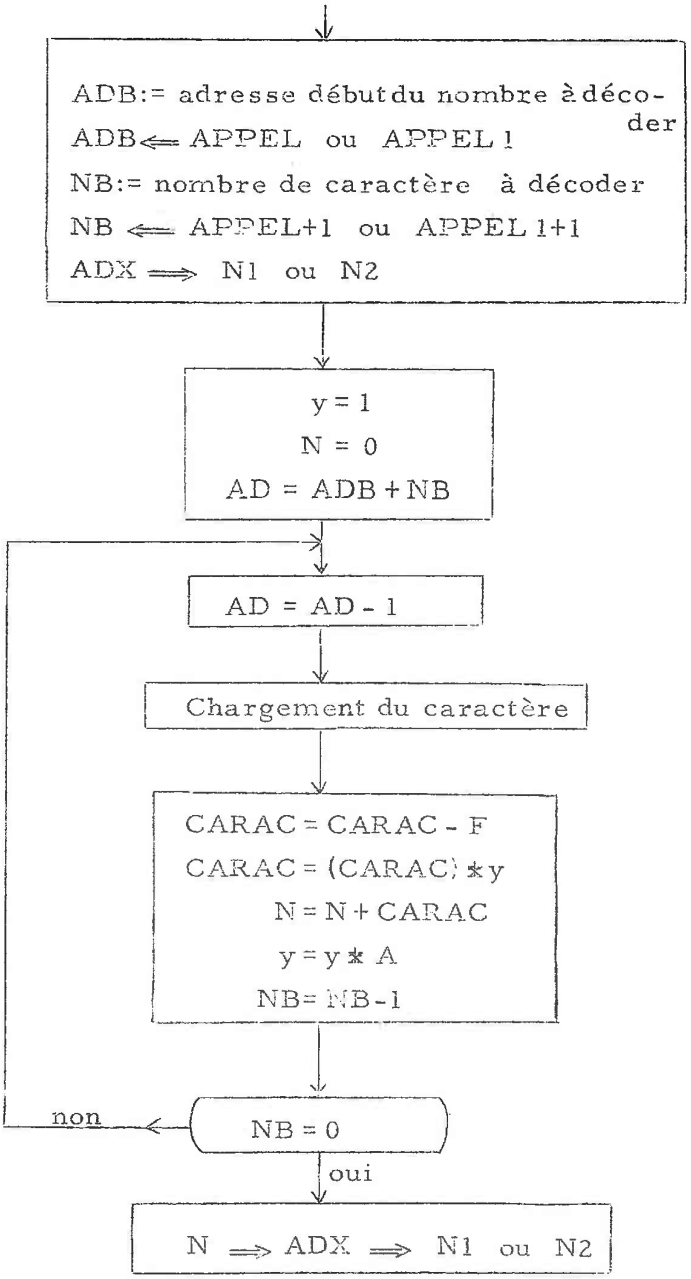
b) Pour convertir la valeur de N2 en binaire :

- dans le premier cas, en entrant dans ce module l'adresse début de chaîne est dans APPEL et le nombre de chiffres à convertir dans APPEL+1 ; en sortant la valeur décodée est dans N1.
- dans le deuxième cas :

  en entrant : l'adresse début de la chaîne est dans APPEL 1
  et le nombre de chiffres dans APPEL 1+1

  en sortant : la valeur décodée est dans N2

ORGANIGRAMME GENERAL



CHAPITRE IV

PRESENTATION DU LANGAGE

INFOL

SOMMAIRE

INTRODUCTION

INFOL (Information Oriented Language) est un système d'organisation de l'information généralisée et d'extraction.

Il est fonctionnel sous le système SCORE (bandes ou tambours magnétiques).

A l'aide d'INFOL, les informations sélectionnées sont extraites du fichier et présentées rapidement et facilement à l'utilisateur, dans leur formats d'origine.

Le langage INFOL (sous-ensemble de l'anglais) a été élaboré pour des utilisateurs n'ayant pas de connaissance en programmation ni même sur les ordinateurs - le concept de champ, impliquant la limitation de taille de mots, a été éliminé en INFOL2, l'utilisateur n'a pas à se préoccuper du nombre de caractères à assigner aux informations - toutes les informations pouvant être de longueur variable.

L'information sortie peut-être un état imprimé ou un fichier bande pouvant être utilisé comme fichier d'entrée pour un autre traitement

Le processeur lui-même se compose de 183 sous-programmes écrits en langage symbolique (COMPASS).

L'ordinateur utilisé est doté d'une mémoire de 32K mots

IV-1 - GENERALITES

En INFOL, le fichier est la plus grande unité d'information , un fichier n'agit que sur un sujet, tel que médical, étudiant, documents, employés, etc...

Un fichier est constitué d'éléments (FICHE). Un élément est constitué d'une liste d'ITEMS identiques se rapportant à un seul sujet.

Un ITEM description définit chaque ITEM.

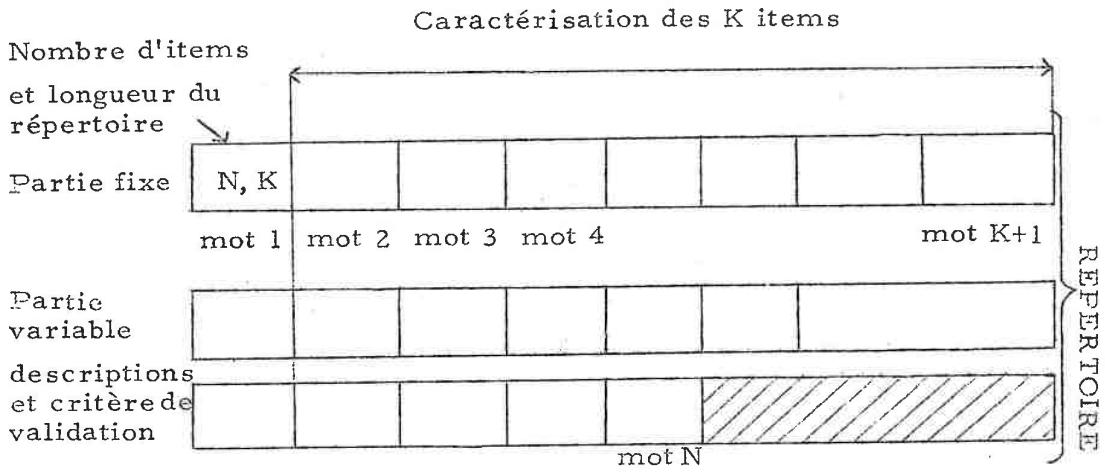
En INFOL, on peut utiliser les ITEM MULTIPLES, mais, pour cela il faut les définir lors de la définition de l'ITEM, un item-multiple comprend plusieurs sous-ITEMS ; par exemple, dans un fichier de documents "authors G. Thomas, E. Alain" l'utilisateur n'a d'ailleurs pas besoin de spécifier le nombre maximum de sous-items ni le nombre de caractères réservés pour chaque sous-ITEM.

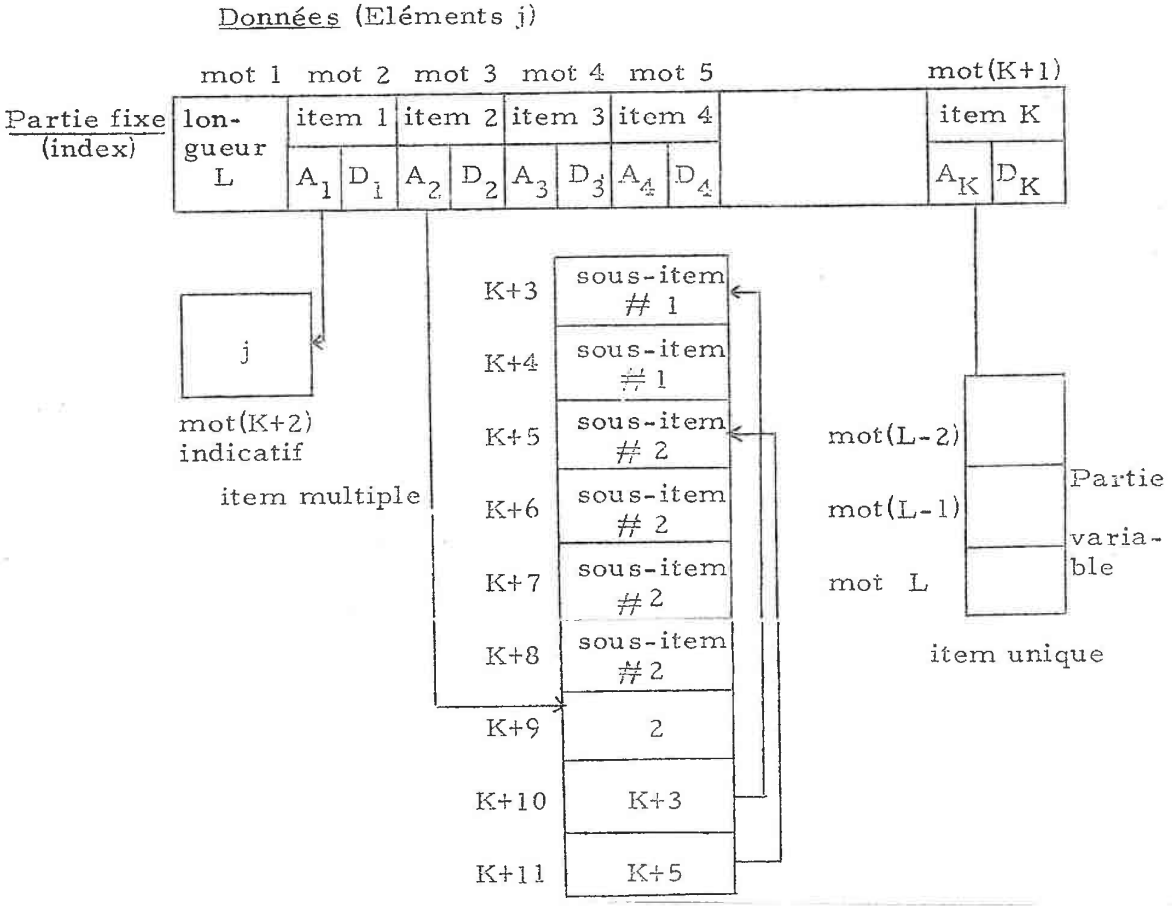
Les éléments dépendant des items sont donc de longueur variable.

Sur les ordinateurs 3600/3800 CONTROL DATA, les items peuvent avoir jusqu'à 16000 caractères.

Structure du fichier

Les éléments composant le fichier total inscrits sur bande en longueur variable et précédés d'un répertoire décrivant les caractéristiques des items, la structure du fichier est spécifiée dans le schéma page suivante :


Caractérisation des K items
Nombre d'items
et longueur du
répertoire
Partie fixe
N, K
mot 1
mot 2
mot 3
mot 4
mot K+1
Partie
variable
descriptions
et critère de
validation
mot N
REPERTOIRE


Données (Eléments j)
mot 1
mot 2
mot 3
mot 4
mot 5
mot(K+1)
Partie fixe
(index)
lon-
gueur
L
item 1
item 2
item 3
item 4
item K
A1
D1
A2
D2
A3
D3
A4
D4
AK
DK
j
mot(K+2)
indicatif
item multiple
K+3
K+4
K+5
K+6
K+7
K+8
K+9
K+10
K+11
sous-item
1
sous-item
1
sous-item
2
sous-item
2
sous-item
2
2
K+3
K+5
mot(L-2)
mot(L-1)
mot L
Partie
variable
item unique

FORME STANDARD

Pour la phase de préparation (INPUT), l'utilisateur a,à sa disposition,une forme standard qui se compose de :

a) ITEM DESCRIPTION

Une liste des ITEMS dans chaque élément.

b) NOMBRE associé à un ITEM

Chaque ITEM s'identifie par un nombre entier entre deux astérisques , lorsque les items-valeurs sont présents, les items-descriptions sont facultatifs ; cependant les nombres items doivent toujours exister et doivent précéder toutes les informations, sauf les items-descriptions.

LES CARACTERES

Les caractères en INFOL se trouvent essentiellement en code standard BCD mais il y a trois caractères spéciaux, l'astérisque, la parenthèse gauche et la parenthèse droite.

Un astérisque est exclusivement utilisée comme séparateur.

BLANC

En INFOL , les blancs précédés ou suivis d'un astérisque sont sans signification, sauf s'il s'agit d'une phase d' "EXTRACTION".

FORMAT LIBRE

Il n'est pas nécessaire d'utiliser des formes spécifiques sur cartes perforées, car l'information en entrée est traitée sous forme de chaînes de caractères continues.

IV-2 STRUCTURE GENERALE DU LANGAGE

INFOL utilise six phases principales identifiées par les mots de contrôle suivant :

ESTABLISHEMENT(création)
INTERROGATION
UPDATE (mise à jour)
VALIDATION
BOOKKEEPING (comptabilité)
REVISION

Les phases majeures sont ESTABLISHEMENT , INTERROGATION, et UPDATE.

Les autres phases, du point de vue de l'utilisateur, sont considérées comme des phases mineures.

Le système doit toujours reconnaître comme première information l'un de ces mots de contrôle sinon il y a émission d'un message d'erreur et arrêt.

IV-2-1 ESTABLISHEMENT

Dans cette phase, le fichier est créé ; les items valeurs sont identifiés par l'item-description, type, catégorie.

IV-2-2 INTERROGATION

Ce mot de contrôle recouvre deux fonctions.

"RETRIEVAL CRITERIA"

et "EXTRACTION"

Dans cette phase les critères fournis par l'utilisateur déterminent l'extraction d'éléments formant un sous-fichier.

IV-2-3 UPDATE (mise à jour)

Il existe deux types de mise à jour

- DISCRETE
- SELECTIVE

a) DISCRETE : s'applique pour la modification, le remplacement d'un élément particulier du fichier.

b) SELECTIVE : quand elle affecte tous les éléments dont les items vérifient des critères d'extraction.

IV-2-4 VALIDATION

Le système compare les déclarations avec les données ; si le système relève une incompatibilité, il y a émission de messages d'erreurs.

IV-2-5 BOOKKEEPING (comptabilité)

Dans cette phase le processeur tient à jour une table de dimension maximale, de toutes les informations du fichier.

Cette comptabilité se fait automatiquement pour chaque item pendant l'ESTABLISHEMENT et UPDATE phases. Cette table est imprimée en clair en tête du rapport final.

IV-2-6 REVISION

Cette phase est utilisée pour changer la structure du fichier ; pour modifier les items.

IV-3 ITEMS

Nous venons de voir que chaque fichier est composé d'éléments de base, que chaque élément est composé d'une liste d'items.

Nous allons maintenant voir les caractéristiques d'item.

IV-3-1 ITEM-CATEGORIE

En INFOL, chaque ITEM doit être caractérisé par un item-catégorie (UNIQUE ou MULTIPLE) ; par exemple : la date de naissance est un item unique, auteur dans un fichier document est un item multiple, parce que l'on peut trouver un document ayant plusieurs auteurs, chaque auteur est alors un sous-item d'item-multiple.

Généralement, il n'y a aucune limite sur le nombre de sous-items.

Deux ou plusieurs items-multiples consécutifs peuvent être attachés en un groupe d'associations s'il ont toujours le même nombre de sous-item.

IV-3-2 ITEM TYPES

Il y a quatre types différents pour chaque item.

IV-3-2-1 Numérique

On peut utiliser un nombre entier ou réel de 1e chiffres maximum sans compter le signe et la virgule, s'il y en a.

IV-3-2-2 Alphanumérique

Un item alphanumérique est une chaîne de caractères quelconques sauf un astérisque ou une parenthèse

IV-3-2-3 Date

Pour permettre aux utilisateurs de manipuler n'importe quelle date.

La date en INFOL est identifiée comme un type unique. En général, on utilise deux formes.

La forme interne est fixe sous une forme standard de six chiffres

yy mm dd

ou

aa mm jj

mais la forme externe est variée: par exemple

FORME EXTERNE	FORME INTERNE
5 sept 1969	69 09 05
11, 24, 67	67 11 24
4-20-70	70 04 20
71 01 11	71 01 11
12/27/70	70 12 27
jan 1969	69 01 00
To DAY	la date courante

Si la forme externe se compose de deux parties, il faut qu'il y ait le mois et l'année, pour le siecle seulement 19 est acceptable.

IV-3-2-4 CODE

Lorsqu'on utilise une information codée sous forme numérique ou mnémonique.

Dans le cas numérique, le système assigne automatiquement un nombre pour chaque code, en commençant par le nombre UN et incrémente de Un séquentiellement.

Dans le cas mnémonique, le type alphanumérique (maximum de 8 caractères) est assigné.

Par exemple :

a) FORME NUMERIQUE	LE TEXTE
1	célibataire, divorcé
2	veuf avec enfant
3	marié sans enfant
4	" avec "

b) FORME MNEMONIQUE	LE TEXTE
EX	excellent
TB	très bien
B	bien
SA	satisfait
IS	inférieur au standard
TF	très faible

NOMBRE associé aux ITEMS

A chaque item de la liste est associé un nombre toujours encadré par des astérisques ; le premier nombre est initialisé à 1 et ce nombre est incrémenté de 1 pour divers items jusqu'au dernier item de la liste. On ne peut pas savoir le même nombre pour deux items différents de la liste. L'item nombre 1 doit toujours être unique et numérique car il donne le nombre d'éléments. Il faut savoir qu'automatiquement au moment de l'entrée des données à chaque item est associée une date d'entrée.

IV-4 CREATION, INTERROGATION et MISE A JOUR DU FICHIER

Dans ce chapitre, nous allons donner une explication plus détaillée des phases essentielles d'INFOL ; et puis, on applique sur un fichier d'étudiant.

IV-4-1 ESTABLISHEMENT

Pour créer un fichier, il faut toujours commencer par le mot de contrôle "ESTABLISHEMENT" (niveau supérieur).

Ce mot de contrôle indique au système qu'un fichier va être créé.

Après "ESTABLISHEMENT", nous trouvons les mots de contrôle de niveaux secondaires :

ITEM DESCRIPTION n	n est un nombre entier indiquant le nombre d'éléments dans l'ITEM DESCRIPTION
CATEGORY-TYPE	
CODES	
VALIDATION	

A la fin des données sont entrées avec le mot de contrôle "INITIAL INPUT".

Parmi ces mots de contrôle, CODES et VALIDATION sont facultatifs mais les autres sont obligatoires , c'est à dire que, lorsque le système finit le contrôle de

ITEM DESCRIPTION, CATEGORY et TYPE

si aucun ITEM n'est défini en forme codée.

INFOL fait le contrôle pour "VALIDATION" ou bien "INITIAL INPUT" sinon il cherche le mot de contrôle "CODE" après "CATEGORY TYPE". Maintenant nous allons étudier "VALIDATION et INITIAL INPUT".

IV-4-2 VALIDATION

Un utilisateur peut spécifier les critères de validation qui doivent être satisfaits avant d'entrer chaque item ou sous-item d'information dans le fichier.

Les critères de validation sont appliqués pendant les phases "ESTABLISHEMENT ou UPDATE".

S'il n'y a pas de critère, le système accepte l' "INITIAL INPUT".

Les critères de validation sont :

IV-4-2-1 NECESSARY

Impose une condition à un item quelconque dans tous les éléments, ceci signifie que, si un item sur lequel on a imposé une condition, "NECESSARY" ne figure pas dans un élément, tout l'élément est rejeté.

IV-4-2-2 CHARACTERS X

Avec ce critère le nombre de caractères alphanumériques d'un item ou sous-item est limité à x caractères (x est un entier < 4096).

IV-4-2-3 ALPHABETIC

Ce critère impose aux items alphanumériques de n'avoir que des items ou sous-items n'ayant que les caractères de A jusqu'à Z, blanc et virgule.

IV-4-2-4 "MAXIMUM m"

Cette condition est utilisée essentiellement pour de multiples items (m est un nombre jusqu'à 511).

On l'utilise pour indiquer le maximum de sous-items dans un multiple item. Si il y a plus de 511 sous-items, l'élément est refusé.

IV-4-2-5 "RANGE"

Ce critère s'applique pour imposer une limite.
Cette limitation de "RANGE" peut être imposée pour lesitems numériques ou date.

IV-4-2-6 "NON NUMERIC"

Cette condition interdit l'emploi de chiffres de 0 à 9 dans les items alphanumériques.

IV-2-2-7 "INTEGER"

Ce critère s'applique numériquement aux items numériques.

IV-5 "INITIAL INPUT"

Le mot de contrôle d' "INITIAL-INPUT" sur une seule carte, indique au système que des données sur les cartes suivantes sont utilisées pour établir la version initiale du fichier. Pour la préparation de l'entrée des données il faut respecter les conditions suivantes :

- des nombres d'items sont entourés par des astérisques *N*
- chaque sous-item est suivi d'un astérisque.

Cette règle s'applique aussi au dernier sous-item:

- un astérisque en plus doit être mis après le dernier item
- une chaîne de caractères blancs suivie d'un astérisque n'a aucune signification.

maintenant la création du fichier est terminée.

On peut mettre sur bande magnétique ce fichier depuis les cartes, il est prêt à être exploiter.
Voir schéma page suivante.

IV-6 INTERROGATION

Pendant cette phase une recherche est effectuée sur la bande, chaque élément est examiné et soumis à des critères conduisant à l'extraction possible des valeurs de certains items.

Dans cette phase, il y a deux fonctions, sous le nom de

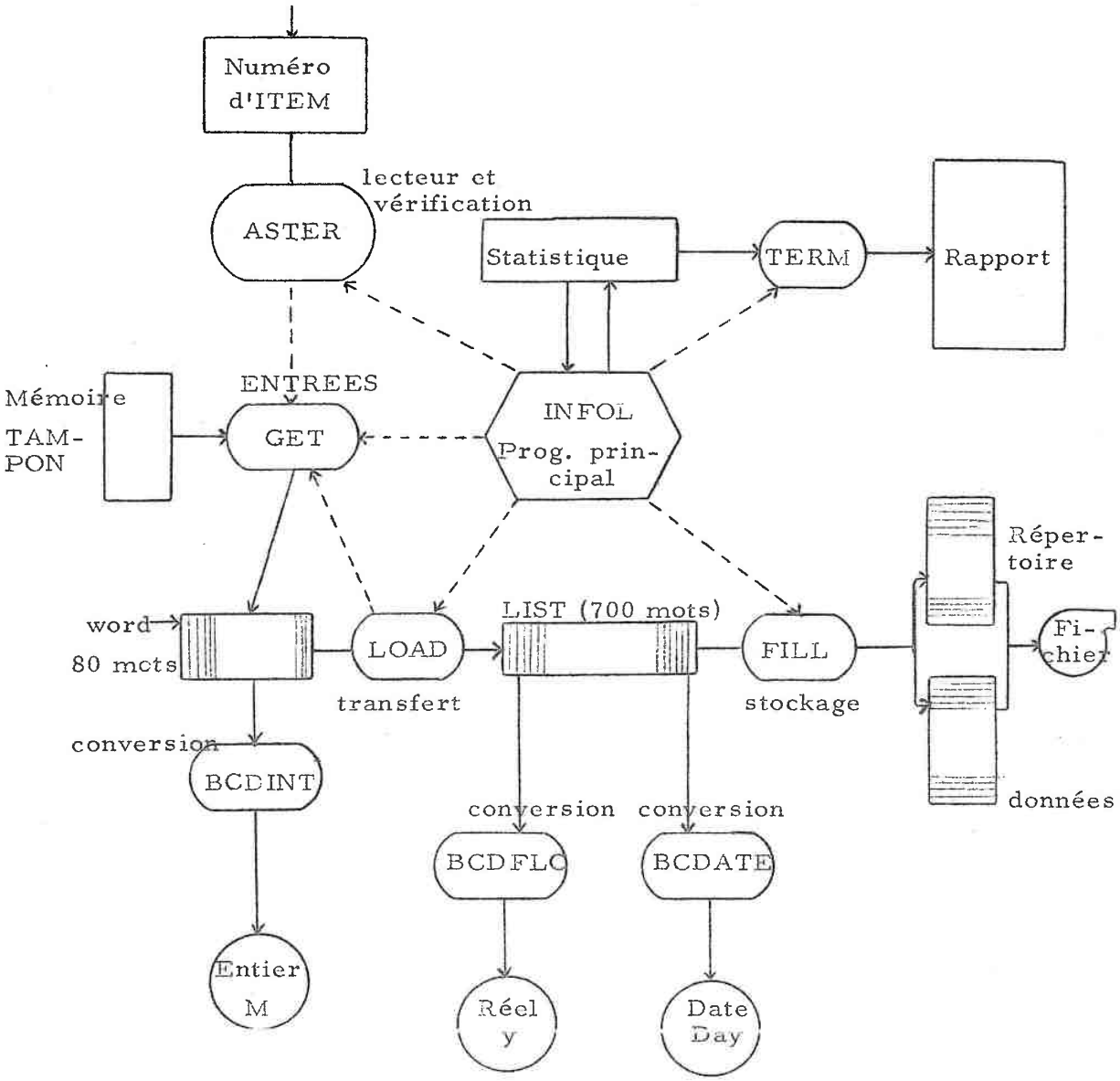
"RETRIVAL CRITERIA"

"EXTRACTION"

L'utilisateur peut spécifier quelques ensembles de "RETRIVAL CRITERIA" pour chaque passage d'interrogation et chaque ensemble doit être suivi par un ou plusieurs ensembles d' "EXTRACTION".

CREATION D'UN FICHIER AVEC INFOL phase ETABLICHEMENT :

On donne une organisation simplifiée du processeurINFOL ci-dessous



données : ————————

contrôle : - - - - - - - -

Ces deux mots de contrôles peuvent être suivis par le même nombre entier qui identifie à l'utilisateur un certain ensemble défini.

Dans la phase "INTERROGATION" il y a quelques points importants qu'il faut d'abord définir. Premièrement, "ITEM SUB-CRITERION" c'est une condition singulière qui peut être appliquée à tous les items d'un élément. Une combinaison de deux ou plusieurs sous-critères définit un "CRITERION".

a - Sous critère d'existence qui est une condition de présence ou d'abscence d'un certain item dans le fichier.

b. - Sous critère de relation. Il impose aux items valeurs l'une de ces conditions :

EQ(égal), NE(non égal), GT(supérieur à)
LT(inférieur à), GE(supérieur ou égal), LE(inférieur ou égal)

et à la fin on compare la valeur d'item avec une valeur externe fournie par l'utilisateur.

c - "ITEM CRITERION" :

C'est un ensemble de sous-critères d'existence et de relation.
Dans la comparaison des nombres et des dates, tous les opérateurs relations dont nous avons parlé en "sous-critère de relation" sont permis et en plus dans la désignation des sous-items et des quantités dérivées, on peut spécifier :

- "FIRST et LAST" le premier et le dernier sous-items.
- "LAST but n" le $(n+1)^{ième}$ sous-item avant la fin.
  "LAST but 1" est un sous-item avant la dernière.
- "SUB ITEM n" le $n^{ième}$ sous-item.
- "MINIMUM et MAXIMUM" peut être utilisé seulement avec des items multiples du type numérique ou bien date .
- "TOTAL" Il refère le nombre de sous-items. On peut l'utiliser avec tous les types d items multiples.

- "SUM et MEAN" peuvent être utilisés avec des items multiples du type numérique (SUM refère les sommes arithmétiques de sous-items VALEUR et MEAN aux moyennes de sous-items valeur)

- "ANY n" Il est utilisé pour lier un ensemble de valeurs externes à un ensemble de sous-items où n est un nombre entier, donnant le nombre de facteurs nécessaires (seulement des opérateurs, EQ et NE peuvent être utilisés après "ANY n")

Par exemple :

* 8 * ANY 2 EQ * NANCY * OR * METZ * OR * PARIS *

Cette condition sur l'item 8 est vraie si au moins deux des trois valeurs (NANCY, METZ, PARIS) sont satisfaites.

Exemple :

Supposons que l'on utilise INFOL pour implanter le fichier des étudiants de 3ème cycle de l'Université :

- on reconnaît un étudiant individuel comme élément de base, caractérisé par un certain nombre d'item,
- maintenant, nous pouvons suivre la fiche particulière suivante qui montre les déclarations nécessaires pour un fichier simple.

INFOL
ESTABLISHEMENT
ITEM DESCRIPTION
NUMERO DE FICHE * 1 * NOM * 2 * DATE D'INSCRIPTION * 3 *
NATIONALITE * 4 * SITUATION DE FAMILLE * 5 *
ADRESSE * 6 * SITUATION MILITAIRE * 7 * SPECIALITE
* 8 * DATE D'OBTENTION *9 * PREPARATION DEMANDEE * 10 *
CATEGORY - TYPE
* 1 * UNARY NUMERIC
* 2 * UNARY ALPHANUMERIC
* 3 * UNARY DATE
* 4 * UNARY ALPHANUMERIC
* 5 * UNARY CODE
* 6 * UNARY ALPHNUMERIC
* 7 * UNARY ALPHNUMERIC
* 8 * MULTIPLE ALPHANUMERIC
* 9 * MULTIPLE DATE
10 UNARY ALPHANUMERIC
CODES
* 5 *NUMBRE CELIBATAIRE * DIVORCE * VEUF AVEC ENFANT *
MARIE SANS ENFANT * MARIE AVEC ENFANT *
VALIDATIONS
* 1 * RANG 99999 INTEGER * 2 * NECESSARY CHARACTERS
30 ALPHABETIC * 3 * RANG SEP 1971 * 7 *
NECESSARY CHARACTERS 30 ALPHANUMERIQUE
* 9 * RANG JULY 1973
INITIAL INPUT
* 1 * 325 * 2 * ALBERT THOMAS * 3 * 2 SEP 1971
* 4 * LUXEMBOURGEOIS * 5 * 1 * 6 * 27 RUE FRANCE 54-NANCY
* 7 * LIBRE DES OBLIGATIONS MILITAIRES 196[illegible]-1970
* 8 * Diplôme ingénieur * 9 * JUNE 1967 * 10 *
DOCTORAT INGENIEUR *

INTERROGATION

Pendant la phase d'interrogation, une recherche est effectuée sur la bande. Chaque élément est examiné et soumis à des critères conduisant à l'extraction possible des valeurs de certains items, comme le montrent les exemples suivants ; supposons que l'on s'adresse au fichier ci-dessous :

a) Pour la liste des noms de tous les étudiants dont la fiche ne comporte pas la SITUATION MILITAIRE, sous forme de rapport.
L'instruction suivante répond à la question :

RETRIVAL CRITERIA 1 * 7 * DOES NOT EXIST
EXTRACTION 1 * 2 * REPORT

b) Si on veut les numéros des fiches (éléments) des étudiants dont la date d'inscription a été intégrée au fichier en nov. 1971.

RETRIVAL CRITERIA 2 * 3 * DATE D'INSCRIPTION
EQ NOV-1971
EXTRACTION 2 * 1 * REPORT

c) Mais si on veut obtenir un rapport donnant les noms de tous les étudiants dont la spécialité est plus que une :

RETRIVAL CRITERIA 3 * 8 * SUB ITEM 2 .GE1
EXTRACTION 3 * 2 * REPORT

d) Si on veut avoir le nom des étudiants qui sont inscrits pour doctorat d'ETAT et qui sont entrés avant 1969.

RETRIVAL CRITERIA 4 * 10 * ALL DOCTORAT D'ETAT
* 3 * LE 1969
EXTRACTION 4 * 2 * REPORT

IV-7 UPDATE (mise à jour)

On peut supprimer, remplacer ou modifier des information dans un fichier. On peut faire ces modifications au niveau d'éléments d'items de sous-items.

En plus, comme nous en avons parlé avant, la modification peut se faire en mode "DISCRETE" quand elle s'applique à un élément particulier du fichier ou "SELECTIVE" lorsqu'elle s'applique à tous les éléments dont les items vérifient des critères d'exécution.

Pour la mise à jour, la première carte du jeu de cartes est la carte "UPDATE" suivi du mot de contrôle au niveau secondaire comme :

DISCRETE	UPDATES
SELECTIVE	CRITERIA
SELECTIVE	UPDATES

IV-7-1 "DISCRETE UPDATES"

Cette mise à jour se fait sur un élément qui est identifié par la valeur d'item * 1 * et l'un des mots de contrôle suivants

a) "NEW ELEMENT", "REMOVE ELEMENT" et "MODIFY-ELEMENT".

le premier "NEW ELEMENT" (le nouveau élément)

Avec ce mot de contrôle, on peut introduire un nouvel élément dans le fichier, l'action de préparation ; un nouvel élément est exactement comme pour "INITIAL-INPUT"

b) "REMOVE ELEMENT" (enlève élément)

En utilisant ce mot de contrôle pour enlever un élément du fichier ; après ce mot de contrôle, il faut mettre le nombre d'éléments par exemple :

REMOVE ELEMENT * 1 * 510 *

c) "MODIFY ELEMENT" :

Un élément du fichier peut être modifié pour ajouter, enlever ou changer chacun de ses items ou sous-items. Pour distinguer les différents modes de modification, il est nécessaire d'utiliser au moins un de ces mots de contrôle.

DELETE, INSERT, SUBSTIT, ELIMINATE, ADD ou REPLACE.

On peut aussi utiliser le premier caractère de chacun d'eux comme D, I, S, E, A, R.

a) DELETE (effacer ou supprimer) :

est utilisé pour supprimer un item d'un élément, ce mot de contrôle peut être utilisé avec tous les "catégories" et "types" d'item. Si un item spécifie "NECESSARY" en "ESTABLISHEMENT" on ne peut pas les supprimer.

b) INSERT ou (I) :

est utilisé pour insérer un entier item dans un élément. On suppose que cet item n'existait pas dans l'élément auparavant sinon le système remet un diagnostic et ne fait pas d'insertion.

c) "SUBSTITUTE" :

Il est analogue au mode "INSERT" mais on l'utilise lorsque l'item du fichier à une valeur. Le "SUBSTITUTE" se refère à un item entier.

d) "ELIMINATE" :

est utilisé pour enlever un ou plusieurs sous-items particuliers d'un multiple item. Le mot de contrôle est suivi par un ou plusieurs sous-items du même type que l'item.

Lorsque cette mise à jour se fait, la liste des sous-items du fichier est testée avec la condition imposée.

Il faut faire attention lorsque des sous-items sont éliminés d'un multiple item d'une "association".

Il faut qu'il y ait le même nombre de sous-item après l'élimination d'un item d'un groupe, des autres membres de groupe doivent être éliminés.

e) ADD :

ajoute des items à la fin de la liste de sous-items dans un item multiple. Les sous-items doivent être du type correct et satisfaire le critère de "VALIDATION" avant d'entrer dans le fichier.

f) "REPLACE" :

est utilisé pour remplacer des sous-items particuliers d'un item multiple. "REPLACE" est suivi par une paire ou plusieurs sous-items valeur, chaque valeur est séparée par un astérisque. Le premier est l'ancienne valeur et le second est la nouvelle valeur qui doit remplacer l'ancienne.

IV-7-2 - "SELECTIVE UPDATE"

L'utilisateur peut spécifier un ensemble de critères et tester chaque élément du fichier. Si l'élément satisfait à des critères, on peut effectuer une certaine mise à jour sélective (enlever ou modifier l'élément).
Il n'est pas possible d'ajouter de nouveaux éléments sans pré-spécifier les critères :

Le mot de contrôle UPDATE peut être suivi par le mot de contrôle "SELECTIVE CRITERIA" pouvant aussi être suivi d'un ensemble de critères qui se préparent de la même façon que "RETRIVAL CRITERIA".

L'ensemble des critères est terminé par le

"SELECTIVE UPDATE"

et à la suite viennent les options de

"REMOVE ELEMENT"

ou

"MODIFY ELEMENT"

La mise à jour sélective est très puissante mais il existe un risque de perdre des informations du fichier.

Un exemple de "SELECTIVE UPDATE"

```
SELECTIVE CRITERIA 5
* 9 * EQ * NAN. FRA. OR * NANCY. FRA. * OR * NAN. FRANCE *
SELECTIVE UPDATE 5
MODIFY ELEMENT
* 9 * SUBSTITUTE NANCY FRANCE *
```

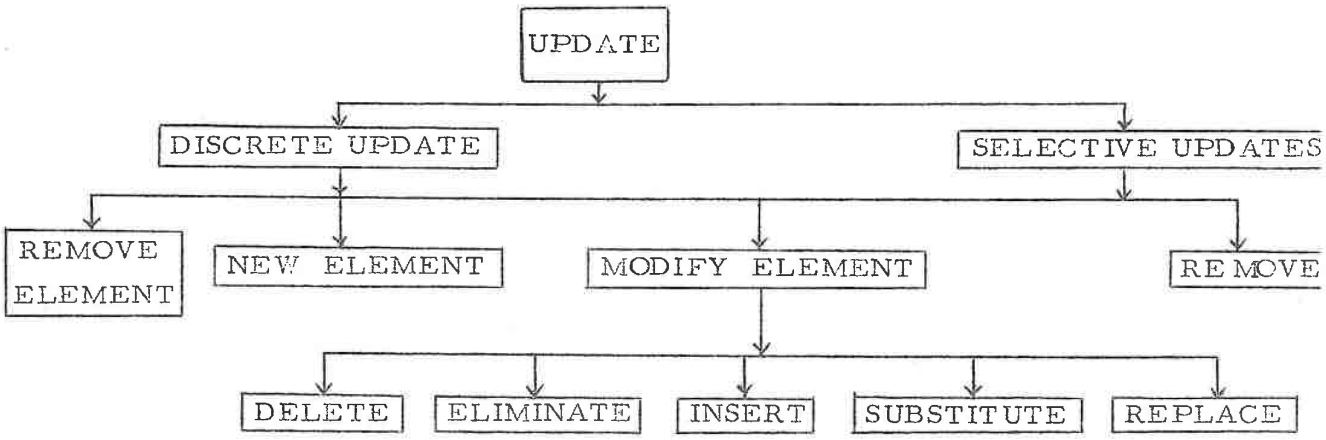
Dans l'exemple ci-dessus, l'item est un item unique mais si l'item 9 est un item multiple, nous pouvons faire comme suit :

```
SELECTIVE CRITERIA 5
* 9 * EQ * NAN. FRA. *
SELECTIVE UPDATES 5
MODIFY ELEMENT * 9 * REPLACE NAN. FRA. * NANCY FRANCE * *
SELECTIVE CRITERIA 6
* 9 * EQ * NANCY FRA. *
SELECTIVE UPDATES 6
MODIFY ELEMENT * 9 * REPLACE NANCY. FRA. * NANCY FRANCE * *
SELECTIVE CRITERIA 7
* 9 * EQ * NAN. FRANCE *
SELECTIVE UPDATE 7
MODIFY ELEMENT * 9 * NAN. FRANCE * NANCY. FRANCE * *
```

Un exemple plus complet :

```
UPDATE
SELECTIVE CRITERIA
* 15 * NE 135
SELECTIVE UPDATES
MODIFY ELEMENT
* 22 * DELETE *
DISCRETE UPDATES
REMOVE ELEMENT * 1 * 425 *
MODIFY ELEMENT * 1 * 375 * * 7 * REPLACE NANCI * NANCY * *
```

NEW ELEMENT
* 1 * 528
* 2 * ASISSTANT PROFESSEUR
* 3 * 3 SEP 70
* 4 * 1500.00
* 5 * JACK DUPONT
* 6 * Fev, 12, 1947
* 17 * 84 boulevard [illegible] PARIS 5e
MODIFY ELEMENT * 1 * 492
* 1 * ELIMINATE 113940 * 114952 *
* 11 * ADD 11328 *
* 7 * SUBSTITUTE 4040000 *



ORGANIGRAMME "UPDATE"

IV-8 REVISION

Si on veut changer la structure du fichier, c'est le mot de contrôle de REVISION qui doit être utilisé. Les révisions peuvent être :

* définir de nouveaux items, ou changer des items description pour des items qui sont déjà créés :

* changer la catégorie d'un item unique ou multiple (sauf pour les items alphanumériques).

* ajouter de nouveaux codes ou changer les codes existants.

* ajouter des critères de validation ou changer les anciens critères de validation. La plupart des mots de contrôle de niveau secondaire de "ESTABLISHEMENT" sont utilisés dans cette phase.

a) "ITEM DESCRIPTION"

De nouveaux items peuvent être ajoutés ou alors les items descriptions sont changés avec

ITEM DESCRIPTION n

n est le nombre total des items après revision.

b) Catégorie- type

Ce mot de contrôle est utilisé pour changer la catégorie d'un item unique ou multiple qui existe dans un fichier et aussi pour spécifier la catégorie et le type de nouveaux items.

c) Validation .

On peut imposer des critères de validation aux nouveaux items ou modifier ou enlever les critères existants .

BIBLIOGRAPHIE

1 - INFCL : Référence manuel, CDC Publ. N° 60170300

2 - VALLEE J. F., CHALICER et MITTMAN-B
≪A Progress Report on INFCL2≫ Northwestern University Report, nov. 1967.

3 - VALLEE J. F. KRULEE G. K. et GRAU A. A.
≪Retrival formula for inquiry systems≫ Information storage and Retrival journal, March 1968.

4 - VALLEE J. F.
Les langages de gestion et l'implantation rationelle des fichiers. AFIRC-DUNOD, n° 9 , Mai/Juin 1968.

5 - PAIR C. - Structure de donnée, Cours école d'été d'Alèse, 1971.

6 - DERNIAME J. C.
Introduction au projet CIVA (doc. interne).

7 - BAZERQUE G.
Implantation des objets de langage PL1 sur une calculatrice, AFCET-DUNOD, n° B-1, Mai 1969.

8 - ARSAC - Les systèmes de conduite des ordinateurs - DUNOD, 197(

9 - BAZERQUE G.
Programmation système Tome I et II.

10 - DEBRAINE P.
Machines de traitement de l'information Tome II.
Programmation Principes et langages d'assemblage.
Masson - Paris 1969.

11 - REILLY E. D., Jr. and FEDERIGHT E. D
The èlements of digital Computer programming.
Holden-Day, Inc, San Francisco - California.

NOM DE L'ETUDIANT : PAYAFAR Mahmoud

Nature de la thèse : Doctorat de Spécialité en Mathématiques Appliquées

Vu, Approuvé

et permis d'imprimer

NANCY, le 19 Avril 1979

Le Président du Conseil de l'Université de NANCY I

J.R. HELLUY