HAL
open science

Contribution à la conception, la réalisation et l'utilisation du système de bases de données S 0 M I N E Structuration des informations

Pierre Marty

▶ **To cite this version:**

Pierre Marty. Contribution à la conception, la réalisation et l'utilisation du système de bases de données S 0 M I N E Structuration des informations. Génie logiciel [cs.SE]. Ecole Nationale Supérieure des Mines de Saint-Etienne; Institut National Polytechnique de Grenoble - INPG, 1976. Français. NNT : . tel-00815854

HAL Id: tel-00815854

https://theses.hal.science/tel-00815854

Submitted on 19 Apr 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ECOLE NATIONALE SUPERIEURE
DES MINES DE SAINT.ETIENNE

INSTITUT NATIONAL POLYTECHNIQUE
DE GRENOBLE

N° d'ordre : 31S

THESE

présentée par

Pierre MARTY

pour obtenir

LE TITRE DE DOCTEUR DE 3° CYCLE

SPECIALITE SYSTEMES ET RESEAUX INFORMATIQUES

CONTRIBUTION A LA CONCEPTION, LA REALISATION ET L'UTILISATION DU SYSTEME DE BASES DE DONNEES SOMINE

STRUCTURATION DES INFORMATIONS

Soutenue à Saint-Etienne le 26 Janvier 1976, devant la Commission d'Examen

MM.	**L. BOLLIET**	**Président**
	R. BOUCHE	
	C. DELOBEL	
	S. GUIBOUD-RIBAUD	**Examinateurs**
	J. KOULOUMDJIAN	
	R. MAHL	

ECOLE NATIONALE SUPERIEURE
DES MINES DE SAINT.ETIENNE

INSTITUT NATIONAL POLYTECHNIQUE
DE GRENOBLE

N° d'ordre : 3IS

THESE

présentée par

Pierre MARTY

pour obtenir

LE TITRE DE DOCTEUR DE 3° CYCLE

SPECIALITE SYSTEMES ET RESEAUX INFORMATIQUES

CONTRIBUTION A LA CONCEPTION, LA REALISATION ET L'UTILISATION DU SYSTEME DE BASES DE DONNEES SOMINE

STRUCTURATION DES INFORMATIONS

Soutenue à Saint-Etienne le 26 Janvier 1976, devant la Commission d'Examen

MM.	**L. BOLLIET**	**Président**
	R. BOUCHE	
	C. DELOBEL	
	S. GUIBOUD-RIBAUD	**Examinateurs**
	J. KOULOUMDJIAN	
	R. MAHL	

ECOLE NATIONALE SUPERIEURE DES MINES DE SAINT.ETIENNE.

Directeur : M. Lucien VIELLEDENT

Sous-Directeur : M. Jacques BOISSÉ

PROFESSEURS DE 1ère CATEGORIE

MM. COINDE Alexandre	Gestion
GOUX Claude	Métallurgie
LEVY Jacques	Métallurgie
RIEU Jean	Mécanique-Résistance des Matériaux
SOUSTELLE Michel	Chimie

PROFESSEUR ASSOCIE DE 1ère CATEGORIE

M. FORMERY Philippe	Mathématiques Appliquées

PROFESSEURS DE 2ème CATEGORIE

MM. GUIBOUD-RIBAUD Serge	Informatique
LOWYS Jean Pierre	Physique
TOUCHARD Bernard	Physique Industrielle

PROFESSEURS ASSOCIES DE 2ème CATEGORIE

MM. FONTEILLES Michel	Géologie
SEROR Denis	Informatique

DIRECTEUR DE RECHERCHE

M. LESBATS Pierre	Métallurgie

MAITRES DE RECHERCHE

MM. BISCONDI Michel	Métallurgie
BOOS Jean-Yves	Métallurgie
Mlle. FOURDEUX Angéline	Métallurgie
MM. LALAUZE René	Chimie
LANCELOT Francis	Chimie
LE COZE Jean	Métallurgie
THEVENOT François	Chimie
TRAN MINH Canh	Chimie

o
o o
o

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

Président : M Louis NEEL

Vice-Présidents : MM. Jean BENOIT
Lucien BONNETAIN

PROFESSEURS TITULAIRES

MM. BENOIT Jean	Radioélectricité
BESSON Jean	Electrochimie
BLOCH Daniel	Physique du solide
BONNETAIN Lucien	Chimie Minérale
BONNIER Etienne	Electrochimie et Electrométallurgie
BRISSONNEAU Pierre	Physique du solide
BUYLE-BODIN Maurice	Electronique
COUMES André	Radioélectricité
FELICI Noël	Electrostatique
LESPINARD Georges	Mécanique
MOREAU René	Mécanique
PARIAUD Jean-Charles	Chimie-Physique
PAUTHENET René	Physique du solide
PERRET René	Servomécanisme
POLOUJADOFF Michel	Electrotechnique
SILBER Robert	Mécanique des Fluides

PROFESSEURS ASSOCIES

MM. RABINS Michaël	Automatique
ROUXEL Roland	Automatique

PROFESSEURS SANS CHAIRE

MM. BLIMAN Samuel	Electronique
COHEN Joseph	Electrotechnique
DURAND Francis	Métallurgie
FOULARD Claude	Automatique
LANCIA Roland	Electronique
VEILLON Gérard	Informatique fondamentale et appliquée
ZADWORNY François	Electronique

MAITRES DE CONFERENCES

MM. BOUDOURIS Georges	Radioélectricité
BOUVARD Maurice	Génie mécanique
CHARTIER Germain	Electronique
GUYOT Pierre	Chimie Minérale
IVANES Marcel	Electrotechnique
JOUBERT Jean-Claude	Physique du solide
LACOUME Jean Louis	Géophysique
MORET Roger	Electrotechnique Nucléaire
ROBERT François	Analyse numérique
SABONNADIERE Jean-Claude	Informatique fondamentale et appliquée
Mme. SAUCIER Gabrièle	Informatique fondamentale et appliquée

CHARGE DE FONCTIONS DE MAITRE DE CONFERENCES

MM. ANCEAU François	Mathématiques Appliquées
PIERRARD Jean-Marie	Hydraulique.

CHERCHEURS DU C.N.R.S.

MM. FRUCHARD Robert	Directeur de recherche
ANSARA Ibrahim	Maître de recherche
DRIOLE Jean	Mâitre de recherche
MATHIEU Jean-Claude	Maître de recherche
MUNIER Jacques	Maître de recherche

Je remercie,

Monsieur L. VIELLEDENT, Directeur de l'Ecole Nationale Supérieure des Mines de Saint-Etienne, qui a bien voulu m'accueillir dans son établissement pour effectuer le travail exposé dans ce mémoire.

Monsieur L. BOLLIET, Professeur à l'Université Scientifique et Médicale de Grenoble qui m'a fait l'honneur de présider ce jury .

Monsieur R. MAHL ,qui a guidé le démarrage de cette étude, alors qu'il était chef du département Informatique de l'E.M.S.E.

Monsieur S. GUIBOUD-RIBAUD, Professeur, Chef du Département Informatique de l'Ecole qui a permis à ce projet d'aboutir.

Monsieur R. BOUCHÉ, Professeur à l'Université de LYON I, qui a dirigé l'ensemble de ce travail et auprès de qui j'ai toujours trouvé aide efficace et encouragements.

Monsieur C. DELOBEL, Maître de Conférence à l'Université Scientifique et Médicale de GRENOBLE pour les conseils qu'il a bien voulu m'apporter notamment dans l'élaboration de la partie IV .

Monsieur J. KOULOUMDJIAN, Maître de Conférence à l'Université de LYON I pour l'intérêt qu'il a manifesté pour ce travail.

Monsieur R. THOMAS, Chef du Centre de Calcul de l'E.M.S.E., qui m'a toujours apporté une aide spontanée.

Tous les chercheurs, techniciens et étudiants du Département Informatique et du Centre de Calcul de l'Ecole pour leur collaboration et leur accueil sympathique.

Mes collègues de l'I.U.T. qui ont su m'aider et me supporter pendant la durée de ces travaux.

.../...

Tous ceux qui ont contribué à la réalisation matérielle de ce document, et principalement :

Mademoiselle J. DEFOUR de l'I.U.T.

Mesdames G. BONNEFOY, C. CHAMBON, M.C. MONTMARTIN, Messieurs LOUBET, BROSSARD de l'E.M.S.E.

Enfin mes deux complices, M. GAILLARD et C. SAYETTAT avec lesquels j'ai réalisé ce travail dans l'amitié et la bonne humeur.

SOMMAIRE DE LA THESE

PREFACE

PARTIE I — SOMMAIRE ET PRESENTATION GENERALE DU PROJET

PARTIE II — RESUMES SUR :
. LA GESTION DE MEMOIRE
. LES ACCES AUX BASES SOMINE

PARTIE III — STRUCTURE DES INFORMATIONS DANS SOMINE

PARTIE IV — DES INFORMATIONS A UNE STRUCTURE SOMINE

PARTIE V — CONCLUSION

FICHES SIGNALETIQUES

"L'outil est convivial dans la mesure où chacun peut l'utiliser, sans difficulté aussi souvent ou aussi rarement qu'il le désire, à des fins qu'il détermine lui-même. L'usage que chacun en fait n'empiète pas sur la liberté d'autrui d'en faire autant..."

Ivan ILLICH - La convivialité

PREFACE

Ce rapport résume le travail effectué pendant les deux dernières années par une équipe qui mène une recherche commune depuis trois ans. Nous sommes trois enseignants du département Génie Mécanique de l'Institut Universitaire de Technologie de Saint-Etienne. Réunis par nos travaux en informatique, nous avons été influencés dans le choix de notre sujet par notre profession commune et nos formations très diversifiées.

La réunion de l'informatique et de l'enseignement devait nous attirer à priori. Mais, tant dans notre pratique quotidienne (algorithmisation des solutions de problèmes, des raisonnements .... etc) que dans un travail de recherche de D.E.A. à l'Université de LyonI(2), nous avons rencontré les limites de l'apport de l'Enseignement Assisté par Ordinateur de forme tutorielle. Nous devions chercher autre chose !

Les rencontres avec Monsieur VIELLEDENT, Directeur de l'Ecole Supérieure des Mines de Saint-Etienne, très interessé par les problèmes pédagogiques et avec Monsieur MAHL, Directeur du département informatique de cette école, ont contribué à définir les buts de notre recherche : implémenter un système de banques de données accessibles à des utilisateurs "étudiants" (aide à l'enseignement) "ingenieurs et techniciens" (aide à la conception assistée par ordinateur) ou "gestionnaires" (aide à la gestion).

Cet objectif fixé, nous avons étudié les réalisations françaises et étrangères dans ces domaines. Ainsi, une importante bibliographie a été consultée. Son analyse critique nous a conduit à préciser les grandes lignes de notre recherche.

Ces éléments sont résumés dans la première partie (introduction générale) de ce mémoire qui montre comment notre travail s'est orienté suivant

.../...

deux axes principaux :

a) La conception et l'implémentation d'un système de gestion de bases de données (SOMINE)

b) Les recherches montrant comment ce système peut être appliqué à des domaines aussi divers que l'E.A.O., la C.A.O. ou l'optimisation de la structuration des informations.

Un tel travail ne pouvait être réalisé que par une équipe et c'est bien dans ce cadre que les choix principaux ont été fixés et les diverses solutions discutées. Mais, dès le début de notre travail, nous avons eu soin de donner à chacun la responsabilité d'une partie du travail.
Grâce à cela, nous présentons aujourd'hui trois thèses de DOCTORAT du TROISIEME CYCLE qui, issue d'un travail commun, sont cependant indépendantes.
Elles portent sur les sujets suivants :

* Interface entre SOMINE et ses utilisateurs et aide à la Conception Assistée par ordinateur (présentée par Cl. SAYETTAT)

* Structuration des informations dans SOMINE et recherche des structures optimales (présentée par P. MARTY)

* Gestion des mémoires de SOMINE et Enseignement Assisté par Ordinateur (présentée par M. GAILLARD)

Conscients d'avoir rencontré plus de questions que nous n'en avons résolues, nous espérons apporter cependant, par ce travail, une certaine contribution à la solution des problèmes concernant les banques de données et leurs interfaces avec des applications diverses.

PARTIE I

PRESENTATION GENERALE DU PROJET

SOMMAIRE DE LA PREMIERE PARTIE

(INTRODUCTION GENERALE)

PAGE

1 : LES BUTS DE NOTRE PROJET — I. 3

2 : DE LA "RECHERCHE DOCUMENTAIRE" aux "QUALITES SOUHAITABLES" — 7

2.1 - LES BANQUES DE DONNEES ET L'ENSEIGNEMENT — 7

2.2 - LES BANQUES DE DONNEES ET LA CONCEPTION DES PROJETS SCIENTIFIQUES ET INDUSTRIELS — 8

2.3 - LES BANQUES DE DONNEES ET LA GESTION — 10

2.4 - LES QUALITES SOUHAITABLES — 11

3 : DE SOCRATE A SOMINE — 13

3.1 - QUELQUES RAPPELS SUR SOCRATE — 13

3.1.1. - La gestion de mémoire — 14

3.1.2. - La structuration des informations — 14

3.1.3. - Le langage de requête — 15

3.1.4. - Le langage de commande — 16

3.1.5. - La programmation — 17

3.2 - SOCRATE ET NOS EXIGENCES — 17

3.2.1. - Choix de certains éléments de SOCRATE — 17

3.2.2. - Les différences indispensables — 18

3.3 - SOMINE — 20

4 : CARACTERISTIQUES GENERALES DU SYSTEME ET PLAN DU RAPPORT — 21

4.1 - LA PROGRAMMATION — 21

4.2 - LA CONFIGURATION GENERALE DU SYSTEME — 21

4.3 - VUE D'ENSEMBLE SUR LE TRAVAIL REALISE — 24

4.4 - CONTENU DU PRESENT RAPPORT — 27

ANNEXE : BIBLIOGRAPHIE DE LA PREMIERE PARTIE — 29

1. - LES BUTS DE NOTRE PROJET

De par sa composition et son travail antérieur, notre équipe était sensibilisée aux problèmes de :

- l'enseignement assisté par ordinateur,
- l'aide à la conception et la simulation des phénomènes scientifiques ou techniques.

Les ordinateurs ont été utilisés largement pour l'un ou l'autre de ces deux thèmes, car, dans chacun d'eux, on trouve une partie importante de recherche documentaire.

La documentation peut utiliser des supports bien différents. Le critère essentiel pour qualifier ces supports est le temps d'accès à l'information. D'où l'idée, aujourd'hui banale, d'utiliser des mémoires d'ordinateurs comme support d'information. Il y a trois avantages à cela :

- les mémoires, disques ou bandes, sont vastes
- les temps d'accès sont beaucoup plus courts que sur les supports traditionnels (livres, etc...)
- on peut facilement actualiser les informations (effacement, création, mise à jour ....)

Documenter étant au centre de nos préoccupations, deux voies pouvaient être choisies :

- créer des systèmes spécialisés et distincts pour assurer l'enseignement, l'aide à la conception ou la simulation (jeux d'entreprises..)

- créer un seul système composé de centres documentaires importants et de sous-systèmes assurant les interfaces de ces centres avec ses divers "clients", qu'ils soient des "étudiants" désirant un certain "cours", des "techniciens" voulant comparer un projet à l'ensemble des réalisations existantes, des "gestionnaires" ou des "scientifiques" voulant utiliser un grand nombre d'informations dynamiques.

.../...

Les systèmes spécialisés existent. Les avantages et les inconvénients des systèmes spécialisés dans l'enseignement assisté par ordinateur sont connus (1). Nous avons nous-même utilisé un tel système lors de notre travail de D.E.A. à Lyon (2). Notre cours de FORTRAN, opérationnel depuis deux ans, nous a permis de connaître les opinions des élèves ; ceux-ci sont généralement satisfaits mais se plaignent parfois d'un certain manque de souplesse. Les systèmes spécialisés dans l'aide au projet (voir notamment le remarquable travail de M. LATOMBE (3)) donnent plus de responsabilité (donc de créativité) aux étudiants mais demandent en parallèle, un cours, sur la spécialité de projet auxquels ils apportent leur aide. Quant aux jeux d'entreprises et autres simulations de phénomènes, ils font toujours appel à beaucoup d'informations que leurs programmeurs ont bien du mal à stocker dans les fichiers traditionnels.

D'où l'idée de construire un système unique, comprenant notamment :

- une banque de données qui pourrait contenir toutes les informations nécessaires à chacune des applications,

- un langage de cours permettant de gérer les relations entre les étudiants et la banque,

- des programmes utilisateurs faisant appel à la base et permettant de résoudre en calcul ou simulation des problèmes scientifiques ou techniques.

A partir de ces objectifs qui devaient conduire à :

- des réalisations en E.A.O. non directif,
- des applications nécessitant l'interrogation et la mise à jour de grandes quantités d'informations en temps réel (gestion, aide au projet, aide aux calculs scientifiques et techniques, etc...) il nous appartenait de :

- rechercher et étudier la documentation nécessaire,
- faire des choix et en déduire les différentes parties du projet global,
- écrire des programmes,
- trouver les applications concrètes et les mettre en oeuvre,
- juger les qualités de notre travail à partir de son efficacité dans ces applications.

.../...

A partir de la recherche documentaire qui précède toutes les réalisations, nous avons dégagé les idées générales et les caractéristiques essentielles de notre projet. Nous verrons de quelle manière dans le prochain chapitre.

2. - DE LA "RECHERCHE DOCUMENTAIRE" AUX ...

.. "QUALITES SOUHAITABLES"

L'utilisation des banques de données n'est pas nouvelle. Les constructeurs, eux-mêmes, ont mis au point, pour leurs machines, des logiciels spécifiques adaptés aux problèmes documentaires.

Cependant, leurs systèmes sont surtout adaptés aux problèmes de gestion (3). Ainsi, pour rechercher les qualités nécessaires à la satisfaction de nos différents objectifs, il est important de comprendre les questions qui se posent dans l'utilisation des banques de données pour :

- l'enseignement assisté par ordinateur,
- la mise au point des projets industriels ou scientifiques,
- la gestion.

2. 1 - LES BANQUES DE DONNEES ET L'ENSEIGNEMENT :

La place des banques de données dans l'enseignement est encore restreinte. Dans leur rapport "l'enseignement assisté par l'ordinateur en Amérique du Nord" (1), KAISER et COULON, en distinguent deux utilisations principales :

a) - banques de données utilisées comme instrument de recherche de faits (FACT RETRIEVAL). On désigne par là toutes les méthodes qui se bornent à retrouver, dans la banque de données, les informations à communiquer aux étudiants. Il n'y a aucun travail sur ces informations.
Cette méthode a notamment été utilisée par l'équipe de Jaime CARBONNEL à Cambridge (10). Son principal inconvénient est la rigidité : toutes les réponses, tous les textes sont "prévus par le maître" et fixés une fois pour toutes dans la banque. L'avantage par rapport à un cours d'E.A.O. tutoriel classique n'est pas évident.

b) - banques de données utilisées comme support d'instrument déductif. Il s'agit d'utiliser les données stockées dans la banque, non pas d'une manière "brute" comme ci-dessus (a), mais comme éléments de l'information à fournir à l'étudiant. Une réponse à donner, par exemple, sera déduite de la question de l'étudiant et des informations contenues dans la banque. On connait les travaux réalisés sur cette question par R.C. SCHANK (6), et son équipe (7). On sait aussi que, malgré des résultats remarquables le problème posé par la déduction n'est pas résolu de manière satisfaisante. Ou bien la déduction est complètement programmée et on retombe dans la fixité reprochée aux systèmes, vue en (a), ou bien elle s'effectue aveuglément ce qui demande des temps de réponse trop longs pour utilisation en mode conversationnel, c'est à dire en enseignement.
Un compromis semble pouvoir être recherché. C'est dans ce sens que vont les travaux de WINOGRAD (8) : il veut guider la recherche des informations dans la banque par une stratégie tenant compte, notamment, de ce qui "est possible dans le contexte donné". Appliquée à la pédagogie, il nous semble que cette voie peut être féconde, surtout si on la rapproche de l'idée émise par CHANDRASEKARAN (et rapportée par KAISER et COULON).

Il propose de "construire une banque de données conversationnelle qui demanderait à l'étudiant si la voie qu'elle emprunte pour accéder à sa requête lui parait bonne". Qu'importe, en effet, la rigidité des "atomes" à enseigner si on peut les parcourir dans un ordre choisi, avec possibilité de retour en arrière, de tests, de construction de problèmes à partir de données choisies par l'étudiant ....

2. 2 - LES BANQUES DE DONNEES ET LA CONCEPTION DES PROJETS SCIENTIFIQUES ET INDUSTRIELS :

Dans ce domaine également, il semble que les bases de données n'occupent pas la place qui pourrait être la leur.

Faire un "projet", c'est, presque toujours, réaliser un compromis entre l'abstrait et le réel, entre la rigueur des théories scientifiques et les exigences de la réalité. Un bon "projeteur" doit connaître les réalisations existant dans son domaine de travail et maitriser les théories scientifiques nécessaires. Tout cela constitue d'importantes masses de documentations, de connaissances qu'il faut actualiser sans cesse. Il est certain que l'utilisation de bases de données adaptées à ces problèmes faciliterait

ce travail. L'exemple des "projets d'électrotechniques" étudiés par M.LATOMBE (3) est significatif. Il faut stocker :

- des matériaux dont les caractéristiques et le nombre sont sans cesse variables,
- des coefficients modifiés chaque jour par de nouvelles découvertes techniques,
- des algorithmes de càlculs qu'il faut adapter à de nouvelles formes de machines ou de composants
- etc.....

Et tout cela doit être rapidement accessible et mis à jour, puis utilisable pour des calculs sur ordinateurs.

A travers ce qui précède se dessine le rôle des banques de données comme "puissants dictionnaires du passé technique et scientifique". Mais les banques de données peuvent intervenir dans la conception et la recherche de bien d'autres façons. Par exemple, on peut se servir des informations contenues dans une banque pour simuler le fonctionnement d'un prototype. Ont été simulés ainsi des mécanismes simples (11), les machines électrotechniques dans le programme "ESPACE" (3), les caractéristiques principales d'un moteur thermique à essence fonctionnant à différents régimes (10). Ce dernier exemple est particulièrement intéressant.On peut schématiquement le décrire de la manière suivante :

Dans la banque de données, on stocke les informations :

- type du moteur
- nombre de tours par minute du vilebrequin
- cylindrée
- consommation d'essence
- couple moteur
- puissance mécanique
- rendement
- etc..

La banque peut alors être utilisée de différentes manières :

a) - Un étudiant peut "manipuler" un moteur comme s'il était tout à fait réel. Par exemple, il peut tracer, en fonction de la vitesse de rotation du vilebrequin, les courbes classiques de consommation, de puissance et de rendement...

.../...

b) - Un constructeur peut consulter la banque pour connaître les principales caractéristiques de tous les moteurs existants qui donnent une puissance de 60 kw à 4000 tr/mn ....

c) - Un projeteur, ayant dessiné un nouveau moteur pourra, en fonction des dimensions et autres caractéristiques qu'il a imposées à son projet, connaître "où se trouvera approximativement son moteur dans la gamme des existants" .... Il pourra ainsi répondre partiellement à la question : "faut-il continuer l'étude ? " avant la construction d'un prototype...

d) - etc .....

Enfin, parlons de l'intervention des banques de données dans la création elle-même. Toute recherche commence généralement par une recherche d'idées. Les techniques de la créativité sont connues (brainstorming, listes de questions, méthodes analogiques, logiques ..... etc). Mais, toutes les idées étant trouvées la question qui se pose est la suivante : comment en dégager la solution optimale ? En général, c'est un "exploit" que l'on réserve aux personnes "d'expérience" ! Or, une approche possible est la comparaison des différentes solutions à l'aide de critères quantitatifs ou qualitatifs soigneusement hiérarchisés (MAUFIT (13), GEMINARD (14), SARAZIN (12)). Dans cette recherche l'analyse combinatoire joue un rôle important et on comprend la place que pourrait y tenir une banque de données dont les informations seraient accessibles directement par des programmes classiques de calcul.

2. 3 - LES BANQUES DE DONNEES ET LA GESTION :

Depuis 1960, le développement des S.G.B.D. est étroitement lié au développement de l'informatique de gestion.

L'apparition des premiers systèmes de gestion de base de données IDS (Integrated Date Store), TDMS (Time-Share Data Management System) (9), IMS (Information Management System) puis de SOCRATE et DBTG-CODASYL a permis de répondre en grande partie aux besoins des utilisateurs qui désiraient éviter les lacunes des méthodes traditionnelles de l'informatique de gestion (redondance des informations, rigidité des fichiers, difficultés d'accès aux informations...)

Actuellement, nous pouvons constater un intérêt grandissant pour les systèmes de gestion de bases de données comme en témoignent d'une

part la multiplication des rencontres organisées sur ce sujet (niveaux recherche ou utilisateur), d'autre part la rapide augmentation du nombre des logiciels disponibles et des installations existantes.

A titre d'exemple citons :

- TOTAL (développé par CINCOM système) qui compte près de 800 installations dans le monde entier dont une dizaine en France.

- I.M.S. qui est utilisé en France par de nombreux groupes bancaires, des assurances, des grandes entreprises nationalisées ou privées (4)

- I.D.S. qui compte plus de 500 installations dans le monde.

- SOCRATE plus récent dont le nombre d'installations est voisin de 40 (5).

- ADABAS, FORTE, D.M.S., GIM, SYSTEME 2000 , MISFIIT ....

Notons que la taille trop grande des systèmes disponibles ne permet pas encore un développement dans les petites et moyennes entreprises et que certains problèmes de performance (temps d'accès ...) seront résolus lorsque le " hardware" aura rattrapé son retard sur la technologie de conception des bases de données.

2. 4 - LES QUALITES SOUHAITABLES :

A travers la "recherche documentaire" concernant les trois types d'applications envisagées, il apparait clairement qu'un système de bases de données peut apporter des éléments nouveaux dans les recherches concernant ces trois domaines.

Mais pour cela, il est indispensable que le système de base de données utilisé possède un certain nombre de qualités principales qu'il paraît possible de résumer ainsi :

a) - Indépendance du système par rapport au matériel informatique utilisé : cette qualité assurera la "transportabilité" du système de bases de données et permettra de comparer éventuellement des matériels.

b) - Fiabilité et garantie du secret : ces qualités sont indispensables à tout système de bases de données.

.../...

c) - Adaptabilité du système : cette qualité est particulièrement importante. Il ne faut pas présenter un système "rigide", "non modifiable" mais, au contraire, un système qui, tout en étant complet, peut être "adapté" à des applications particulières : la modularité de la programmation et du système lui-même devront autoriser l'utilisateur à compléter, supprimer ou ajouter certaines fonctions.

d) Mesurabilité des caractéristiques du système : Ceci est nécessaire pour évaluer ses performances mais aussi, par exemple, pour mesurer les effets d'un changement de structuration ou pour procéder à certaines expériences à partir de la variation des valeurs des caractéristiques du système en vue de son optimisation future.

e) - Traitement efficace des informations : cela comporte notamment :

* la rapidité des opérations d'interrogation et de mise à jour
* la souplesse dans la structuration et la recherche des informations (plusieurs types de structures doivent être possibles, la recherche des informations doit être rapide et pouvoir utiliser plusieurs moyens : accès direct, critères de filtrages, etc...)
* la variété dans le choix des moyens de mise à jour et d'accès aux informations situées dans la base : les modes conversationnel et batch sont indispensables mais il faut surtout que des programmes puissent accèder aux informations, éventuellement les transformer puis les ranger à nouveau dans la base.
* la variété dans les centres d'informations possibles: listes de valeurs, mots, textes, numériques de tous types, chaînes de bits etc....

Cet ensemble de "qualités souhaitables" constitue une sorte de "CAHIER DES CHARGES" du système à réaliser. Il peut servir à comparer nos objectifs aux qualités présentées par les réalisations existantes.

.../...

3. - DE SOCRATE A SOMINE

A partir du "cahier des charges" défini à la fin du chapitre précédent, il convenait d'examiner les systèmes de bases de données existants pour les comparer et ..... pour éviter la reproduction de l'un ou de l'autre de ces systèmes.

Dans le paragraphe 2.3, nous avons donné la liste des systèmes de base de données dont nous avons consulté la documentation et, dans la conclusion de ce mémoire, nous reviendrons sur la comparaison de notre "produit" avec certains de ces systèmes.

Nous voulons cependant, dès cette introduction, traiter d'une manière spéciale le SYSTEME SOCRATE (5) car le groupe dans lequel nous avons travaillé au début de notre projet (MM. MAHL et THOMAS notamment) était en relation avec l'équipe SOCRATE de Grenoble. Nous avons donc pu :

- entrer en contact avec cette équipe,
- obtenir toute la documentation sur le système SOCRATE.

De ce point de vue, on peut dire que SOMINE est un des fils de SOCRATE..... ce qui ne veut pas dire que le fils est une "copie" du père !. Nous verrons au contraire, dès ce chapitre et dans toute la suite de ce mémoire, que ces deux produits sont très différents.

Remercions cependant les concepteurs et les réalisateurs de SOCRATE qui nous ont reçus plusieurs fois à Grenoble. Il est certain que leur réalisation nous a fortement influencés dans le début de nos travaux. Examinons leur produit.

3. 1 - QUELQUES RAPPELS SUR SOCRATE :

Ce paragraphe se contente de rappeler sommairement quelques éléments importants sur le système SOCRATE. Il suppose connues les nombreuses

brochures dans lesquelles ce système a été décrit et, principalement la publication citée en (5).

3. 1. 1 - La gestion de mémoire :

Une base de données SOCRATE utilise deux espaces mémoires secondaires : un espace "tambour" pour la structure et un espace "disque"pour les informations structurées (données).

A chacun de ces espaces secondaires correspond un espace "virtuel" de grande taille (2^{31} mots) dans lequel toute information a une adresse virtuelle. L'unité de passage de l'espace virtuel à l'espace mémoire secondaire est appelée "sous-page" (bloc de $2^n - 1$) mots consécutifs.

L'unité de passage entre les mémoires secondaires et la mémoire centrale de l'ordinateur est appelée "page" : bloc de m sous-pages consécutives.

Une information subit donc les transferts suivants :

Information ⟶ espace virtuel ⟶ espace disque ⟶ espace tore
(programmeur) (adresse virtuelle) (adresse disque) (adresse en mém. centrale)

Les applications permettant les passages entre les divers espaces mémoires sont réalisées par des fonctions de réallocations programmées, utilisant les méthodes de l'adressage dispersé, de l'adressage calculé et des tables (dictionnaire de gestion de mémoire). Elles sont entièrement transparentes à l'utilisateur de la banque.

3. 1. 2 - La structuration des informations :

La structure SOCRATE de base est arborescente. Elle ordonne des caractéristiques de divers types : valeurs numériques entières, listes de valeurs, mots, textes, entités. Ces caractéristiques peuvent être "simples" ou "complexes", regroupées en "blocs" si on le désire. Les "entités" sont des "blocs à caractère répétitif".

exemple :

```
ENTITE 100 PERSONNES
 DEBUT
 NOM   MOT 6
 FIN
```

Cette entité permettra le stockage des noms de 100 personnes (au plus).

Le COMPACTAGE DES DONNEES est une qualité essentielle de SOCRATE car :

- les valeurs numériques, définies par leur plage de variations sont codées sur des frontières de bit,
- les mots sont codés sur des frontières d'octet,
- les textes sont codés sur des frontières de mot.

On peut ainsi gagner de la place sur les fichiers traditionnels de trois façons :

- non redondance des informations (propriété de toutes les banques d'informations)
- compactage,
- choix de la meilleure structure à partir du critère : "meilleur compactage".

Les structures SOCRATE admettent un certain nombre de "caractéristiques particulières" qui accroissent de façon importante la souplesse des mises à jour et des interrogations et diminuent les temps nécessaires à ces opérations. Citons par exemple :

- la caractéristique "REFERENCE" : elle transforme la structure arborescente classique en une structure de GRAPHE
- les caractéristiques "discriminante rapide" et "inverse" qui diminuent les temps d'accès à certaines informations.
- certains filtres : "alors", "pour" .... qui permettent des sélections particulières d'information.

3. 1. 3 - Le langage de requête :

C'est le langage du dialogue entre la banque et ses utilisateurs. Dans ce langage, on utilise une technique de type COBOL (ou PL1) consistant à désigner tous les niveaux de blocs pour arriver à l'objet auquel on s'intéresse. Mais la présence des entités impose la création de "FILTRES" servant à caractériser plus précisement l'objet recherché. Les principaux filtres sont :

- AYANT : filtre de base généralement utilisé dans les requêtes. Il définit une réalisation possédant une ou plusieurs caractéristiques de valeur (de type) particulière .

- les filtres utilisés à la suite du filtre "AYANT" : " UN" "DERNIER" (positionnement dans l'entité), " LUI-MEME" et "EN COURS" (positionnement sur la caractéristique dont on vient de s'occuper ou sur laquelle on travaille actuellement), UN et TOUT (nombre d'éléments auxquels on veut s'adresser) ..... etc.

- les filtres combinant des expressions " BOOLEENNES" : "ET", "OU".

- les démonstratifs : ils permettent de caractériser l'élément par son numéro d'ordre dans l'entité ; exemple : NOM D'UNE PERSONNE X (I) fournira le nom de la 50 ième personne si X (I) = 50.

- filtre " TEL QUE" ...

Ce langage, assez complexe peut être allégé par l'utilisation de "MACROS". Un programme " MACROGENERATEUR " transforme alors le texte allégé en un texte normal, (rédigé en langage de citation ordinaire) qui est transmis au compilateur puis à l'interprèteur de SOCRATE.

3. 1. 4. - Le langage de commande :

Il complète le langage de requête en précisant le "mode" du travail à réaliser ou certaines particularités de ce travail. Il autorise les opérations de :

MISE A JOUR : création, effacement et mise à jour

ADDITION : on peut ajouter un nouveau représentant à une classe d'entité

INTERROGATION et DENOMBREMENT : permet de connaître le nombre des réalisations effectives d'une entité

SURVEILLANCE : en traitant une certaine partie de la structure de manière "préférentielle", cette commande autorise l'allègement du langage de citation.

POUR : permet l'utilisation de "groupements de commandes" : par exemple, plusieurs interrogations seront réalisées par une seule requête (interrogation itérative).

.../...

3. 1. 5. - La programmation :

Socrate a été programmé en autocode I.B.M. à Grenoble puis une version a été redéfinie pour les ordinateurs produits par la compagnie internationale d'informatique (série IRIS).

Dans les deux cas, le système n'est pas indépendant de la machine.

3. 2 - SOCRATE ET NOS EXIGENCES :

Comment le système SOCRATE se situe - t - il par rapport aux "qualités souhaitables" résumées dans notre cahier des charges ?
Répondre à cette question c'est à la fois :

- choisir certains éléments de Socrate qui pourront être utiles pour la conception et l'implémentation de SOMINE

- Montrer en quoi SOMINE doit différer de SOCRATE de manière fondamentale.

3. 2. 1. - Choix de certains éléments de Socrate utilisables dans SOMINE :

Certaines idées originales du système SOCRATE peuvent être conservées : il en est ainsi dans la GESTION DE MEMOIRE pour :

* l'utilisation d'un fichier-structure et un fichier-données
* la virtualisation de la mémoire et le partage de la mémoire réelle en espace disque et espace tore.

Dans la structuration des informations on peut conserver :

- la structuration arborescente dans le cas général,
- la caractéristique "référence" dans les cas où des structures de "graphes" sont indispensables.
- l'idée des "filtres" mais en transformant considérablement leur conception et leur utilisation car les filtres SOCRATE nous paraissent trop complexes, parfois redondant et pas tous utiles.

Au niveau du langage de requête, il faut retenir :

- la désignation successive de tous les niveaux de la structure qui permettent d'arriver jusqu'à l'objet (ou le groupe d'objets-entité) auquel on s'intéresse.

- les démonstratifs qui achèvent, de façon claire, cette détermination.

.../...

- certains éléments du langage (UN, TOUT, etc) mais il est absolument indispensable de simplifier et d'élaguer ce langage de requête si on veut le rendre facilement utilisable par des programmes.

Enfin, les modes "création", " interrogation", " mise à jour" "suppression", sont indispensables tandis que le "dénombrement" des réalisations effectivement créées est seulement utile.

La simplification du langage de requête devrait rendre sans objet les modes " surveillance" et "pour".

Au fil de cette analyse, on voit qu'il faut surtout transformer les parties qui sont plus ou moins directement en relation avec les utilisateurs. La comparaison des qualités de Socrate avec les qualités que nous avons estimées importantes confirme ce point de vue.

3. 2. 2. - Les différences indispensables :

Reprenons nos critères un à un en insistant maintenant sur les aspects nouveaux que SOMINE doit présenter par rapport à SOCRATE.

3.2. 2. I - Indépendance du système par rapport au matériel informatique :

Les réalisations actuelles du système SOCRATE ne sont pas transportables. Ainsi, une adaptation à un nouveau matériel impose une nouvelle programmation des algorithmes. SOMINE, au contraire, DOIT ETRE TRANSPORTABLE. Pour cela, nous devrons programmer dans un langage algorithmique.

3.2.2. II - Fiabilité et garantie du secret :

SOCRATE est fiable et le secret des informations y est correctement assuré. SOMINE n'apportera pas de nouveautés sur ce point.

3.2. 2.III - Adaptabilité du système :

SOCRATE est un système assez vaste et rapide. Ces qualités ont pour revers des difficultés d'adaptation en vue d'une application particulière. Sa programmation en langage autocode ne permet pas à l'utilisateur moyen de comprendre son fonctionnement.
La programmation de SOMINE devra être la plus COMPREHENSIBLE possible (choix du FORTRAN, modularité). Sur ce point SOMINE et SOCRATE seront donc très différents.

.../...

3.2.2.IV - Mesurabilité des caractéristiques du système :

Toutes les caractéristiques de SOCRATE (du nombre de pages en mémoire centrale au temps de réponse dans telle condition de fonctionnement) ont été mesurées. Il reste cependant un certain nombre d'expériences à réaliser. Nous pensons notamment à des expérimentations sur la taille des pages, sur l'effet de la présence ou de l'absence du cadre de manoeuvre, sur le nombre optimum de pages à garder simultanément en mémoire centrale.... etc. Les transformations de programmes nécessaires pour ces expériences ne sont pas faciles sur le système SOCRATE. Il faudra que SOMINE les permettent. D'autre part, il faudrait pouvoir compter les nombres d'interrogations et de mises à jour afin d'effectuer des recherches statistiques sur le fonctionnement de la base.

3.2.2.V - Traitement efficace des informations :

a) Le temps de réponse de SOCRATE est satisfaisant. SOMINE, dans sa version originale essaiera seulement de... faire aussi bien.

b) Les divers modes de structuration des informations ainsi que la variété des moyens d'accès sont un des atouts de SOCRATE. Par contre, la complexité du langage de requête est un défaut; il est souhaitable que l'utilisateur de la banque dispose d'un langage plus clair, plus proche du langage naturel. SOMINE s'efforcera de joindre cette qualité à une rapidité d'accès convenable.

c) SOCRATE peut être utilisé à partir de télétypes en mode conversationnel. Mais, les programmes ne peuvent pas utiliser directement la banque et la banque ne peut pas lancer l'exécution d'un programme. Au contraire, pour atteindre ces objectifs (notamment pour le calcul scientifique) il est INDISPENSABLE QUE LE SYSTEME SOMINE PUISSE A LA FOIS :

- être accessible par tout programme en langage évolué
- provoquer l'exécution de programmes.

A partir de ce critère apparaît une des différences essentielles entre les deux systèmes.

En effet, l'exécution de programmes à partir de la base elle-même est une propriété très importante car, à partir d'une seule requête on pourra déclencher l'exécution d'un nombre quelconque de requêtes et de programmes. Cela ouvre la voie à l'utilisation directe d'une banque de données dans les domaines de la conception assistée par ordinateur, du calcul scientifique et technique et résoud le problème des actions automatiques sur les informations.

.../...

d) Tous les types d'informations retenus dans SOCRATE le seront également ans SOMINE (entités, listes de valeurs, textes, mots, numériques entiers). Mais les calculs souhaités (aide à la conception, par exemple) imposent d'autres types d'informations : les valeurs numériques REELLES et, surtout, les PROGRAMMES.

3.3 - SOMINE :

SOMINE est donc fils de SOCRATE.

Mais SOMINE n'est pas SOCRATE car la discussion précédente fait apparaître trois sortes de différences :

-au niveau du mode et du langage de programmation

-au niveau des types d'informations et, surtout des types d'UTILISATEURS puisque SOMINE acceptera comme utilisateurs des PROGRAMMES EN LANGAGE EVOLUE.

-au niveau des langages de définition de structure et de requête : ces langages seront simplifiés et clarifiés afin de limiter au minimum le besoin de formation des utilisateurs de SOMINE.

Ayant défini ce que "nous souhaitions réaliser" dans les deux chapitres dont la lecture s'achève ici, nous devons présenter les caractéristiques générales de SOMINE et le plan du mémoire. Ce sera l'objet du chapitre suivant.

4. - CARACTERISTIQUES GENERALES DU SYSTEME ET PLAN DU RAPPORT

Le "cahier des charges" défini à la fin du précédent chapitre nous a guidé au cours de la conception et de l'implémentation de la base de données SOMINE et de ses applications. Il nous a imposé un certain nombre de choix essentiels au niveau de :

- la programmation,
- la configuration générale du système global d'aide à l'enseignement, à la conception et à la gestion,
- la conception et l'implémentation de SOMINE,
- la mise en oeuvre des applications.

Ce sont ces choix généraux que nous allons présenter en renvoyant chaque fois au rapport dans lesquel ils sont développés.

4. 1 - LA PROGRAMMATION :

Nous avons voulu réaliser un système indépendant du matériel utilisé. Pour cela, nous avons écrit tous les programmes (sauf deux sous-programmes de petite dimension) en FORTRAN.

Par souci de souplesse, la programmation a été réalisée de façon modulaire ce qui permet d'ajouter, d'enlever ou de modifier les fonctions de base du système.

4. 2 - LA CONFIGURATION GENERALE DU SYSTEME :

Le système global d'aide à l'enseignement, à la conception et à la gestion est construit autour d'un système de base de données (SOMINE) dont les "caractéristiques" très diversifiées (numérique entier et décimal),

.../...

texte, liste de valeurs, programmes d'écriture de cours ou programmes de tous types compatibles avec FORTRAN) permettent des applications multiples.

Conformément à la qualité principale souhaitée, ce système de bases de données peut être utilisé directement par des personnes (mode conversationnel sur écrans (16) alphanumériques et graphiques) ou par des programmes rédigés en langages compatibles avec FORTRAN.

Un "programme-tampon" de temps partagé assure l'interface entre les utilisateurs travaillant "simultanément" et le système de base de données, conformément à la figure de la page suivante.

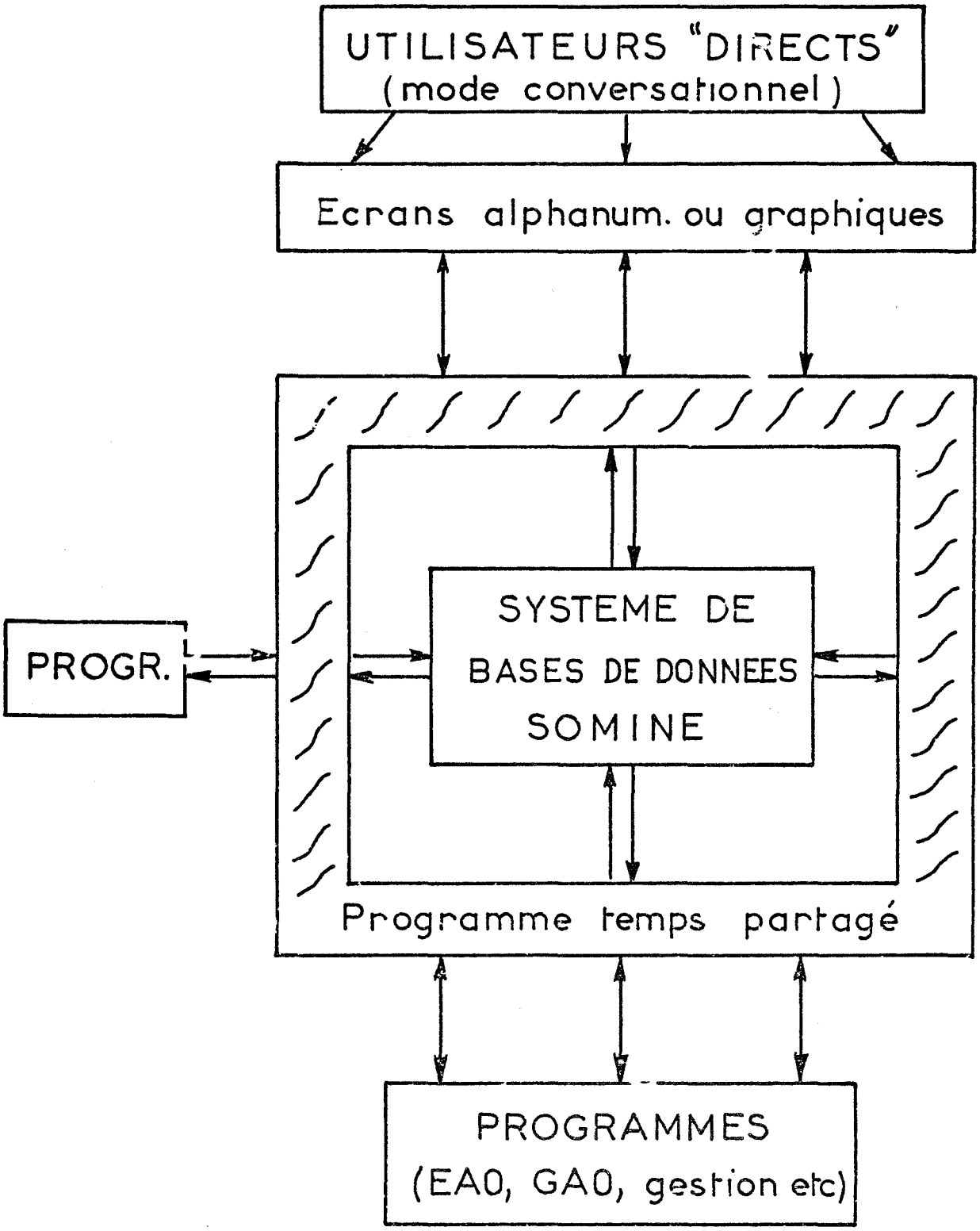


Figure I.1 : SOMINE CENTRE D'UN SYSTEME

4. 3 - VUE D'ENSEMBLE SUR LE TRAVAIL REALISE :

L'implémentation d'un système de gestion de bases de données pose des problèmes dans les domaines de :

- la gestion de la mémoire
- la structuration des informations
- l'interface entre le système et ses utilisateurs.

a) gestion des mémoires :

Une banque de données est un ensemble d'objets (informations et relations) reliés entre eux par une structure.

Ces objets ont des caractères dynamiques : leurs places et leurs tailles peuvent changer au cours d'un traitement.

La place physique nécessaire pour "loger" ces objets demande un espace important (de l'ordre de cent millions d'octets dans la plupart des applications de gestion). Il est donc impossible, dans l'état actuel de la technologie, d'implémenter un système de bases de données sans assurer la répartition des informations sur plusieurs niveaux de mémoire physique. Assurer cette répartition et les passages entre les divers espaces en donnant à l'utilisateur de la banque l'impression qu'il travaille dans un espace UNIQUE, de taille pratiquement ILLIMITEE, constituent les fonctions principales de la gestion de mémoire. La conception et la réalisation de la gestion des mémoires de SOMINE seront exposées dans la thèse de M. GAILLARD.

b) Structuration des informations :

Les performances d'un système de base de données dépendent largement des divers choix opérés lors de la définition de l'implémentation de la structure formelle des informations.

Les diverses structurations permises par SOMINE et le langage au demeurant extrêmement simple, qui permet leur implémentation seront définis et commentés dans ce mémoire.

c) SOMINE et ses utilisateurs :

La structure des informations étant définie et implémentée, il faut maintenant "remplir" la banque, puis l'interroger. Cela concerne le

traitement des informations c'est-à-dire, finalement, les relations entre SOMINE et les utilisateurs. L'accès à la banque se fait à travers un langage de requête qui doit être simple et adapté à tous les types d'applications.

La conception et l'implémentation du langage de requête ainsi que les diverses possibilités de traitement des informations que SOMINE peut offrir sont exposées dans la thèse de C. SAYETTAT.

D'autre part, puisque nous étions partis d'un "cahier des charges" il était important de vérifier que le système construit possédait les propriétés qui avaient motivé sa conception. Ainsi, nous avons développé trois recherches dans des domaines divers afin de mettre en évidence les qualités et les insuffisances de notre système et de montrer comment il était possible de le compléter pour pallier à ces insuffisances.

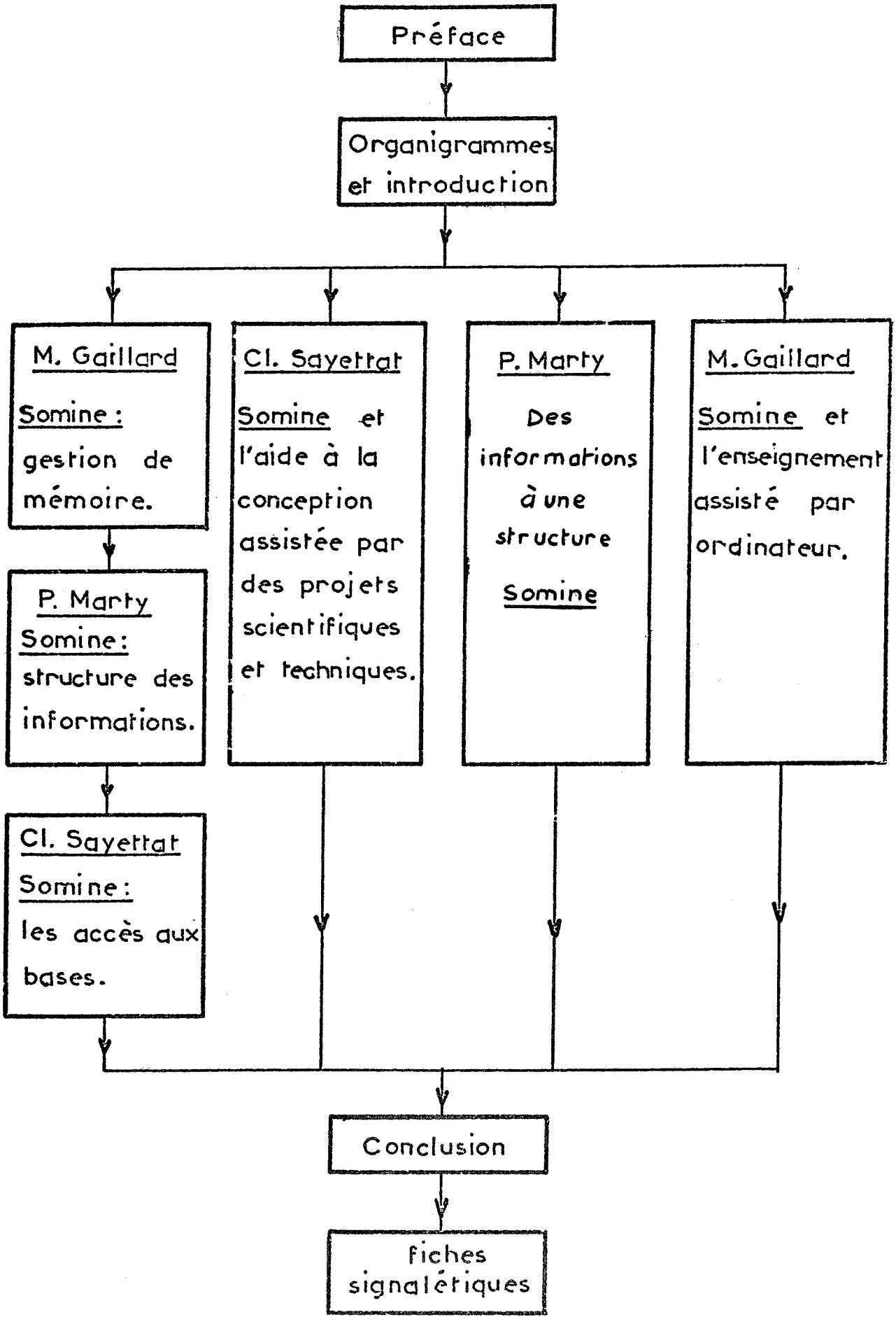
Les comptes-rendus de ces travaux sont exposés :

- dans la thèse de M. GAILLARD pour l'Enseignement Assisté par Ordinateur,
- dans la thèse de C. SAYETTAT pour la Conception Assistée par ordinateur,
- dans ce document pour une étude sur la structuration optimale des informations.

Ces trois thèses sont le résultat d'un travail d'équipe cependant, elles peuvent être lues indépendamment l'une de l'autre.

L'état actuel du projet SOMINE est résumé sur le schéma qui figure à la page suivante.

ETAT ACTUEL DU PROJET SOMINE



4.4 - CONTENU DU PRESENT RAPPORT :

Dans la suite de ce mémoire on trouvera :

- des résumés sur la gestion des mémoires et les accès aux bases SOMINE (partie II).
- la structure des informations dans SOMINE (partie III).
- des informations à une structure SOMINE (partie IV).
- une conclusion générale (partie V) dans laquelle nous donnons le bilan actuel des résultats obtenus par notre équipe.

Nous espérons donner ainsi à la fois une vue exacte de notre travail et une vue d'ensemble sur le projet dans lequel notre travail est inclu.

Nous sommes à la disposition de toutes les personnes qui désireraient des compléments d'information.

PARTIE II

ETAT ACTUEL DU PROJET SOMINE

RESUMES SUR :

-LA GESTION DE MEMOIRE

-LES ACCES AUX BASES SOMINE

S O M M A I R E

1 - LE SYSTEME DE GESTION DE MEMOIRE 5

2 - L'INTERFACE AVEC LES UTILISATEURS 7

2.1. - Le langage de requête 7

2.2. - La requête 8

2.2.1. - Forme générale 8

2.2.2. - Mode 10

2.2.2.1. - Création, suppression 10

2.2.2.2. - Mise à jour, interrogation 10

2.2.2.3. - Fréquence 10

2.2.3. - Citation 11

2.2.4. - Valeur 11

2.3. - Autres possibilités 12

2.4. - Les types d'accès 13

2.5. - Programmation 15

Dans cette partie nous allons brièvement exposer l'essentiel de ce qu'il faut retenir sur le système SOMINE en ce qui concerne :

* le système.de gestion de mémoire
* l'interface avec les utilisateurs.

pour la compréhension de la suite de notre rapport.

Pour plus de précisions sur ces sujets le lecteur pourra consulter respectivement les thèses de M. GAILLARD et de C. SAYETTAT.

1 - LE SYSTEME DE GESTION DE MEMOIRE

La gestion des mémoires réalise les passages de l'espace réel disque à l'espace virtuel et inversement ceci à partir de la mémoire centrale de l'ordinateur. Pour cela, il y a projection d'un espace virtuel de dimension très grande sur un fichier disque de taille plus petite. La méthode retenue est :

Une gestinn de mémoire paginée allouée de façon statique.

Elle réalise les fonctions suivantes :

* Création et préformatage des fichiers disques .

* Formatage et initialisation des fichiers disques (programme FORMFICH)

* Ouverture des fichiers (programme INFICH)

* Sauvegarde des informations et fermeture des fichiers (programme OUTFICH)

* Relations entre mémoire centrale et fichiers disques .

*Lecture des informations (programme LECTURE)

* Ecriture des informations (programme ECRIRE)

.../...

2 - L'INTERFACE AVEC LES UTILISATEURS

Les accès aux bases SOMINE se font à travers un langage de requête simple, souple et d'apprentissage rapide.

2. 1. - LE LANGAGE DE REQUETE :

En utilisant le formalisme dérivé de la forme normale de Bacchus-Naur et décrit dans la partie III (§ 2.2.1.), les règles de grammaire peuvent être présentées ainsi (les symboles du langage sont soulignés):

<REQUETE> :: = < MODE> < CITATION> (= < VALEUR>) #

<MODE> :: = CREATION | SUPPRESSION | MISE A JOUR | INTERROGATION | FREQUENCE

<CITATION> :: = <IDENTIFICATEUR> (<NOMBRE> | X (<N>))
([DU | DE LA | DE L' | DE | AYANT] <CITATION>)

<VALEUR> :: = <NOMBRE> | <IDENTIFICATEUR> | '<CHAINE ALPHANUMERIQUE>' |
<NOMBRE> . <NOMBRE>

<N> :: = <NOMBRE> | <IDENTIFICATEUR>

<NOMBRE> :: = CHIFFRE { CHIFFRE }

<IDENTIFICATEUR> :: = LETTRE { CARACTERE ALPHANUMERIQUE }

.../...

LETTRE $\in$ $\{$ A,B, ... Z $\}$

CHIFFRE $\in$ $\{$ 0,1, ... 9 $\}$

CARACTERE ALPHANUMERIQUE est un élément de l'ensemble des caractères alphanumériques.

Les automates qui correspondent à cette description sont donnés fig.II.1 (page suivante). Ils ont été utilisés pour l'analyse syntaxique.

2. 2. - LA REQUETE :

Les exemples qui seront donnés dans ce paragraphe et les suivants porteront tous sur la structure " EXEMPLE " de la partie III § 2.2.2. (fig. III.6 et III.7).

2.2.1. - Forme générale :

Une requête SOMINE comporte toujours :

- un "mode" qui définit l'action à entreprendre
- une "citation" qui désigne la caractéristique (cf. III § 2.2) objet de la requête.

Elle peut comporter, en outre, une valeur.

exemple :

$$\underset{\text{mode}}{\underset{\uparrow}{M}} \quad \underbrace{\text{NOM DE LA PERSONNE } 5}_{\text{citation}} \quad = \quad \underset{\text{valeur}}{\underset{\uparrow}{\text{DURAND}}} \ \#$$

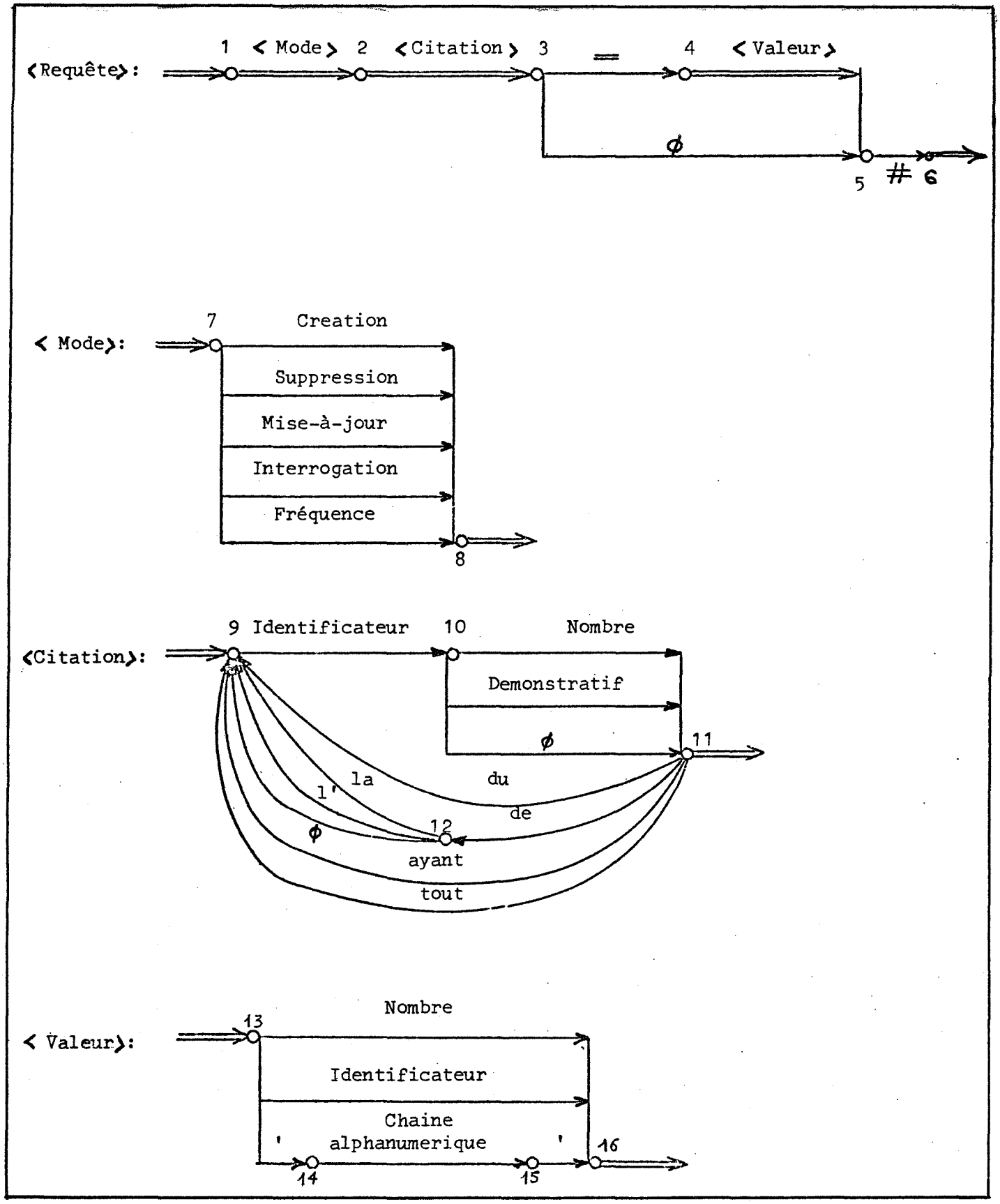


Figure II.1 - Les automates de la requête

.../...

2.2.2. - Mode :

2.2.2.1. - Création, suppression (C, S) :

* cas des entités ou des entités-choix
(Cf. III § 2.2.2.7. - § 2.2.2.8)

Ces modes permettent respectivement de réserver ou de libérer pour des réalisations d'entité la place physique qui leur est nécessaire.

exemple : C PERSONNE 10 #

* Cas des références : (Cf. III § 2.2.2.9)

La création (resp. la suppression) provoque l'établissement (resp. la disparition) d'un lien entre la caractéristique et la réalisation d'entité citée par addition (resp. soustraction) d'un maillon de la chaîne des références. (Cf. III 3.3.5.)

exemple : C PROPRIETAIRE DE LA VOITURE 7 = 58 #

(le PROPRIETAIRE de la voiture 7 est alors la PERSONNE correspondant à la 58ème réalisation de l'entité PERSONNE).

2.2.2.2. - Mise à jour, interrogation (M, I) :

Ces deux modes sont utilisés pour les caractéristiques simples MOT, TEXTE, LISTE DE VALEURS, etc..... La mise à jour fait entrer une information dans la base à la place qui lui est réservée. L'interrogation permet d'obtenir en sortie une information déjà contenue dans la base.

exemple : M COULEUR DE LA VOITURE 5 = VERT #
I ADRESSE DE LA PERSONNE 34 #

.../...

2.2.2.3. - Fréquence (F) :

Ce mode permet de connaître la valeur des compteurs introduits dans le fichier-structure pour connaître le nombre des opérations effectuées sur une caractéristique (Cf. III 3.2.2.5.)

2.2.3. - Citation :

Elle définit la caractéristique qui est l'objet de la requête en la reliant au sommet de l'arbre structurel par désignation de tous les niveaux intermédiaires. Elle se présente comme une suite d'identificateurs isolés par des séparateurs ("de" , "de la ", "du", "de", "tout(e)", " ayant "). Les identificateurs correspondant à des entités (ou des entités-choix) peuvent être suivis du numéro de la réalisation citée. Ce numéro figure explicitement ou est représenté par une variable indicée X (I) appelée démonstratif.

exemples : C VOITURE DE LA PERSONNE 3 #

L'identificateur VOITURE n'étant pas suivi d'un numéro, la citation désigne la première réalisation non encore créée de l'entité VOITURE (il y a recherche du premièr bit nul dans la chaîne des bits de présence cf. III 3.3.4.).

I MARQUE DE LA VOITURE X(3) DE LA PERSONNE X(1) #

démonstratifs

La dernière citation n'est correcte que si des valeurs ont été affectées préalablement à X(3) et X (1).

2.2.4. - Valeur :

La valeur n'apparaît que dans les requêtes de mise à jour des caractéristiques simples, elle sert alors de véhicule à l'information qui doit entrer dans la base.

Les types de valeurs possibles (chaînes alphanumériques, nombres....) correspondent à la nature des caractéristiques intervenant dans la requête.

exemples :

M JOUR DE DATE = 12 #

Caractéristique NUMERIQUE E — nombre entier

M COULEUR DE LA VOITURE X(5) = JAUNE #

Caractéristique LISTE DE VALEURS — identificateur appartenant à la liste de valeurs

M ADRESSE DE LA PERSONNE 27 = ' COURS FAURIEL 42100 ' #

Caractéristique TEXTE — chaîne alphanumérique (elle doit être entre 2 apostrophes)

2. 3. - AUTRES POSSIBILITES :

Dans les paragraphes précédents, nous avons donné un apperçu des possibilités du langage de requêtes sous sa forme initiale. Au cours de la mise en oeuvre des premières applications de SOMINE, des améliorations ont été apportées à l'interpréteur de base en étudiant :

.../...

* les requêtes groupées
* les requêtes booléennes
* la synonimie
* les requêtes à valeurs variables
* les requêtes contenant le séparateur TOUT
* les requêtes discriminantes.

L'introduction de ces possibilités a eu pour objectifs :

- la diminution du temps d'accès à la base pour un travail donné (ex : requêtes booléennes).

- la simplification de la formulation des requêtes (ex : synonimie).

2. 4. - LES TYPES D'ACCES (fig. II. 2 ,p. suivante)

Les types d'accès aux bases SOMINE peuvent varier en fonction des applications envisagées. Ainsi, il est permis d'utiliser des **requêtes** :

* Dans des sessions d'utilisation des bases :
(une requête se présente dans ce cas comme une donnée constante "hollerith" du programme utilisateur)

- en mode conversationnel (le programme BANKECR réalise l'interface entre l'écran et le système SOMINE)

- en traitement par lots (par utilisation d'un programme standard)

* Dans tout programme écrit dans un langage compatible avec FORTRAN :
une requête se présente comme l'argument du sous-programme d'analyse et d'interprétation INTER. Elle réalise de fait l'interface entre les langages évolués comme FORTRAN ou COBOL et SOMINE. Par exemple, si A est une chaîne alphanumérique représentant une requête, nous pouvons écrire :

.../...

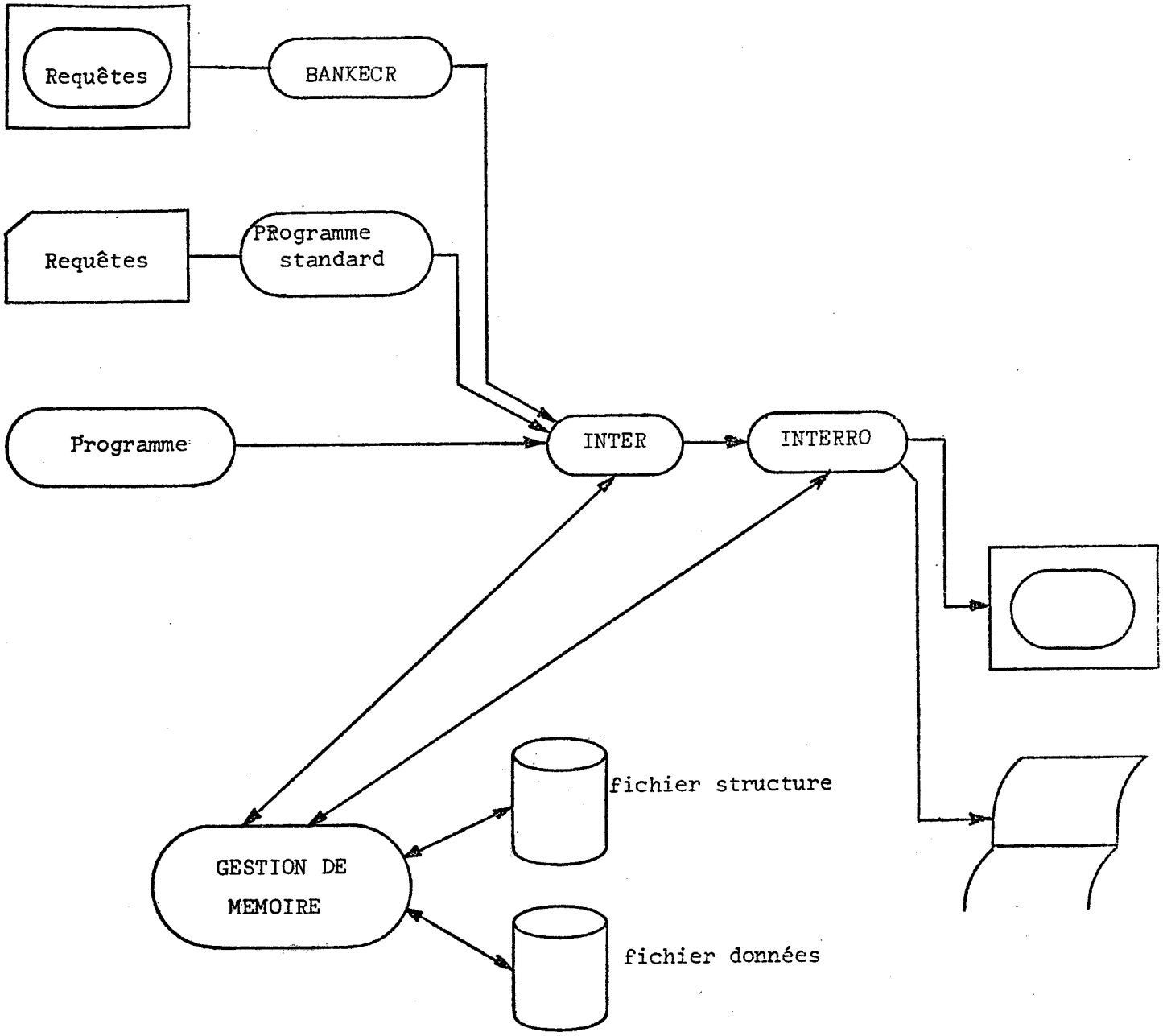


Figure II.2 - Les accès aux bases

- dans un programme FORTRAN :

```
CALL  INTER     (A)
```

- dans un programme COBOL :

```
ENTER INTER  USING  A
```

2. 5. - PROGRAMMATION :

L'exécution d'une requête met en oeuvre des programmes qui nécessitent une exploration de la structure par la méthode depth-first (C. SAYETTAT III. § 2.3.) et un calcul d'adresses dans le fichier - données.:

Ces programmes ont pu être réalisés simplement grâce à la représentation que nous avons adoptée pour la structure dans le fichier virtuel et que nous décrivons dans la partie III (§ 3) de ce rapport.

PARTIE III

STRUCTURE DES INFORMATIONS DANS SOMINE

"Nous ignorons quelles formes de vie ont pu évoluer sur d'autres planètes, mais nous pouvons être assurés que partout la vie doit être organisée hiérarchiquement"

Arthur KOESTLER - Le cheval dans la locomotive

SOMMAIRE

1 - INTRODUCTION 7

1.1. - Modèle 7
1.2. - Les différentes natures de modèles 8
1.3. - Modèle hiérarchisé 8
1.4. - Modèle réseau 9
1.5. - Modèle relationnel 10
1.6. - Le modèle adopté dans SOMINE 11

2 - DEFINITION DE LA STRUCTURE 13

2.1. - Graphe 13
2.2. - Langage de définition de structure (LDS) 15

2.2.1. Grammaire du LDS 15
2.2.2. Description des caractéristiques 19

2.2.2.1. - Type MOT 19
2.2.2.2. - Type TEXTE 19
2.2.2.3. - Type NUMERIQUE 22
2.2.2.4. - Type LISTE DE VALEURS 23
2.2.2.5. - Type IDEM 24
2.2.2.6. - Type BLOC 24
2.2.2.7. - Type ENTITE 25
2.2.2.8. - Type ENTITE-CHOIX 26
2.2.2.9. - Type REFERENCE 30
2.2.2.10. - Type INVERSE 32

.../...

3 - REPRESENTATION DE LA STRUCTURE SUR LE SUPPORT PHYSIQUE 33

3.1. - L'organisation de la structure sur le support physique 33

3.1.1. Technique statique 33
3.1.2. Les fichiers 33

3.2. - Le fichier-structure 34

3.2.1. Le plexe 34
3.2.2. Format d'un grain 35

3.2.2.1. - Schéma d'un grain
3.2.2.2. - Identification et nature de la caractéristique
3.2.2.3. - Pointeurs vers les autres grains du plexe
3.2.2.4. - Fonctions de calcul
3.2.2.5. - Compteurs
3.2.2.6. - Remarque

3.3. - Le fichier - données 39

3.3.1. Les types d'organisation 39
3.3.2. Bloc, caractéristiques simples 44
3.3.3. Idem 45
3.3.4. Entité, entité-choix 45
3.3.5. Référence 46

3.3.5.1. - Chaîne des références 46
3.3.5.2. - Mise à jour de la chaîne des références 48

3.3.6. Inverse 50

4 - IMPLEMENTATION DE LA STRUCTURE 51

4.1. - Analyse syntaxique 51

.../...

4.1.1. Analyse d'un mot-clé du langage 52
4.1.2. Analyse d'un identificateur 52
4.1.3. Analyse d'un nombre 52

4.2. - Construction du plexe 53

4.2.1. Du résultat de l'analyse lexicologique au plexe 53
4.2.2. Détermination des pointeurs vers la structure 55
4.2.3. Calcul de la taille des blocs et des entités 57
4.2.4. Calcul des adresses relatives 60

5 - CONCLUSION 63

6 - BIBLIOGRAPHIE 67

.../...

Dans de nombreux domaines (gestion, enseignement, recherche scientifique, applications techniques,....) l'information à utiliser donc à gérer peut-être de nature très diverse (nombres, chaines alphanumériques, dessins,....) mais, quel que soit le support (livre, disque, fiche, mémoire de calculateur,....), dès qu'elle atteint une certaines complexité, son traitement nécessite qu'elle soit organisée.

De ce besoin sont nées les banques d'informations. Les performances de ces dernières sont conditionnées par les différents choix opérés lors de la définition et de l'implémentation de la structure formelle suivant laquelle l'information est conservée.

1 - INTRODUCTION

1. 1. - MODELE :

On peut définir une base de données comme étant le modèle d'un monde physique en perpétuelle évolution, l'état de ce modèle, à un instant donné, représentant la connaissance qu'il a acquis de ce monde (cf. ABRIAL /12/).

Il est évident qu'un modèle parfait n'existe pas. "Le modèle n'est pas la réalité, il en a seulement une connaissance partielle " /12/.

Les recherches effectuées ces dernières années pour élaborer de nouveaux modèles ont toujours eu pour objectif une meilleure représentation du monde physique ; on peut ainsi préciser, en fonction de cette finalité, la définition d'une base de données en disant que son but est " de représenter sur des supports un grand nombre d'informations munies de relations internes qui les lient afin de pouvoir grâce à un logiciel approprié lui faire subir des opérations définies à l'avance (cf. LOUIS-GAVET /11/).

1. 2. - LES DIFFERENTES CLASSES DE MODELES :

Tout système de gestion de bases de données, quelle que soit sa nature, met à la disposition de l'utilisateur deux outils fondamentaux :

* un langage de description de données qui permet de décrire les informations, leurs propriétés et leurs liens .

* un langage de manipulation de données qui permet les opérations d'extraction, d'addition, de suppression,..... sur les données d'une base.

Les S.G.B.D. peuvent être classés suivant la nature des langages employés (langage procédural ou langage non procédural) ou, suivant les possibilités de manipulation de données offertes à l'utilisateur (type des requêtes, actions spontanées...) . Mais ce qui caractérisera le mieux un S.G.B.D. c'est le type d'organisation logique des informations quil permet.

En prenant ce dernier critère, nous pouvons distinguer 3 grandes catégories de modèles de S.G.B.D. qui sont dans l'ordre chronologique de leur apparition :

* le modèle hiérarchisé (IMS /13/, TDMS /4/)

* le modèle réseau (SOCRATE /3/, ADABAS /14/, IDS /15/ D.B.T.G - CODASYL /16/)

* le modèle relationnel (modèles de laboratoire inspirés par les travaux comme ceux de CODD /17/ ou ABRIAL /12/)

1. 3. - MODELE HIERARCHISE :

Dans un modèle hiérarchisé la structure peut-être représentée par un arbre (cf. exemple de la figure III.1)

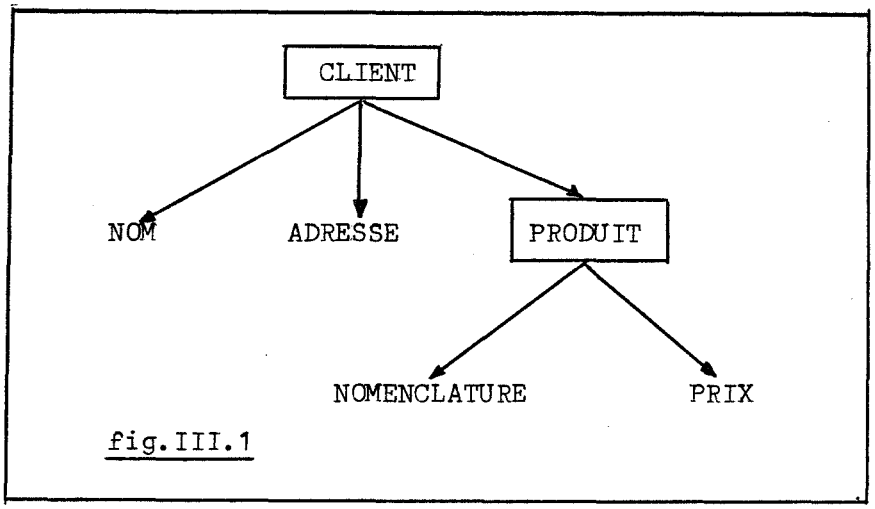


fig.III.1

Les informations sont découpées en un certain nombre de groupes répétitifs (ou entités) composés eux-mêmes de constituants (ou caractéristiques) . Il n'existe qu'un seul lien entre deux entités (par exemple entre CLIENT et PRODUIT) ce qui entraîne, entre autres inconvénients :

* une répétition de données : si plusieurs clients utilisent le même produit, la description de ce dernier est répétée

* une difficulté dans la réalisation des programmes d'accès aux informations. Ainsi, s'il est aisé de rechercher les nomenclatures des produits utilisés par un client donné (exemple fig. III.1.) il est, par contre, nécessaire de balayer l'ensemble des clients pour mettre à jour le prix d'un produit déterminé.

1. 4. - MODELE RESEAU :

Un modèle de ce type peut, soit s'appuyer sur la définition d'une relation binaire entre des groupes de données (système DBTG /22/), soit étendre la notion de modèle hiérarchisé en introduisant le concept de référence (système SOCRATE /3/).

Ce type de modèle permet d'utiliser des structures plus complexes que dans le cas précédent; une meilleure représentation d'un système d'informations peut donc être obtenue. On pourra notamment éviter des répétitions de données ou diversifier les accès à un groupe de données.

.../...

Reprenons la structure de la fig. III.1 en la modifiant de façon à faire apparaître deux liaisons entre les entités CLIENT et PRODUIT ; ainsi, la mise à jour du prix d'un produit ne nécessitera plus le balayage de l'ensemble " CLIENT " (fig.III.2).

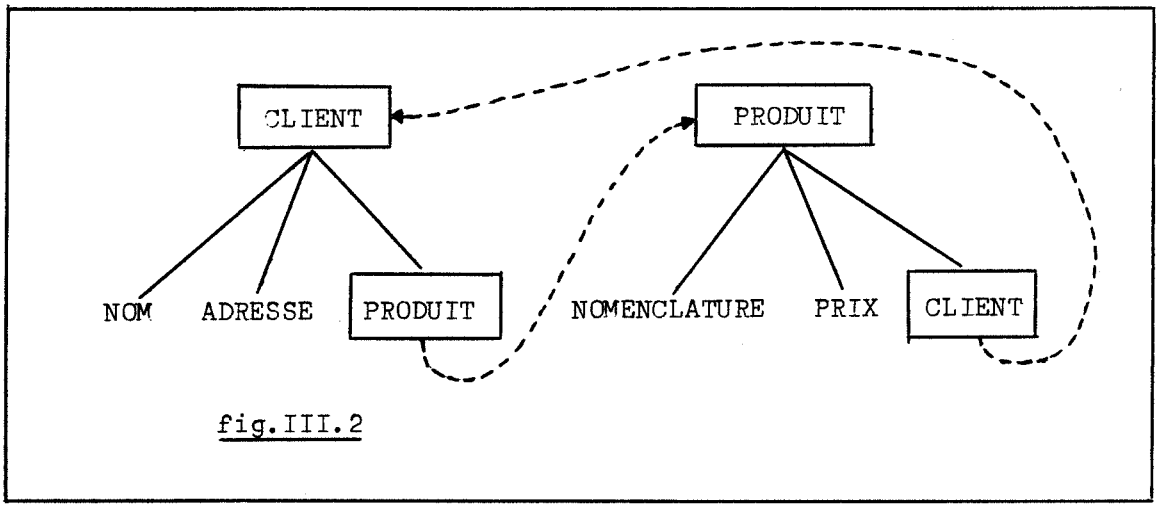


fig.III.2

1. 5. - MODELE RELATIONNEL

Les remarques que nous avons faites dans 1.3. et 1.4. montrent que la définition d'une structure dans un modèle de type hiérarchisé ou de type réseau fixe les chemins d'accès aux informations.

Dans un modèle rationnel il est défini deux ensembles : un ensemble de relations, un ensemble de constituants. Les chemins d'accès aux informations ne sont pas déterminés ce qui donne à l'utilisateur:

- une plus grande liberté dans la manipulation des informations
- la possibilité d'introduire de nouvelles informations ou de nouveaux liens.

.../...

Les modèles relationnels semblent avoir une meilleure "data indépendance" que les autres ; toutefois, comme le fait remarquer DELOBEL dans /19/ , " il est possible qu'on paiera ce plus haut degré d'adaptabilité des modèles relationnels par une baisse des performances".

1. 6. - LE MODELE ADOPTE DANS SOMINE :

Lorsque nous avons commencé notre étude dont le but, rappelons-le, était de concevoir et réaliser un SGBD capable de répondre à un certain nombre de critères (cf. partie 1) l'avancement des recherches en cours sur les modèles relationnels était tel que nous ne pouvions pas espérer à court terme déboucher sur un système opérationnel.

Nous nous sommes donc référés aux logiciels qui semblaient alors les plus aptes à représenter les informations les plus diverses avec leurs propriétés et leurs liens. Il s'agissait des modèles de SGBD de type réseau. Nous avons ainsi réalisé un modèle du type réseau en utilisant l'expérience acquise par les équipes ayant travaillé dans cette direction et en particulier sur le projet SOCRATE /3/.

2 - DEFINITION DE LA STRUCTURE

2. 1. - GRAPHE :

L'organisation de base est un arbre orienté. Des définitions précises de liens de parenté entre les éléments de l'arbre permettent ultérieurement l'accès aux différentes informations représentées par cet arbre.

SOMINE devant servir à des applications de types très divers, nous nous sommes attachés à définir des liens (ou fonctions d'accès) permettant une exploration rapide de l'arbre suivant plusieurs méthodes (préordre, ordre terminal, largeur d'abord) /1/.

Les principaux de ces liens, représentés par des flèches dans l'exemple de la figure III.3, sont les suivants :

- fonction PERE : elle permet une remontée au noeud ascendant adjacent :

exemple : B ⟶ A, G ⟶ B,.....

- fonction FILS : elle permet l'accès au premier fils du noeud (s'il existe), c'est à dire au fils le plus à gauche :

exemple : A ⟶ B, D ⟶ E, B ⟶ C, H ⟶ I

- fonction FRERE : elle permet de retrouver le premier frère c'est-à-dire le noeud adjacent à droite d'un même niveau ayant même père :

exemple : B ⟶ H, C ⟶ D, D ⟶ G, E ⟶ F

.../...

La relation entre une feuille et la racine ("ancêtre" de la feuille) se fait en composant plusieurs fois la fonction PERE.

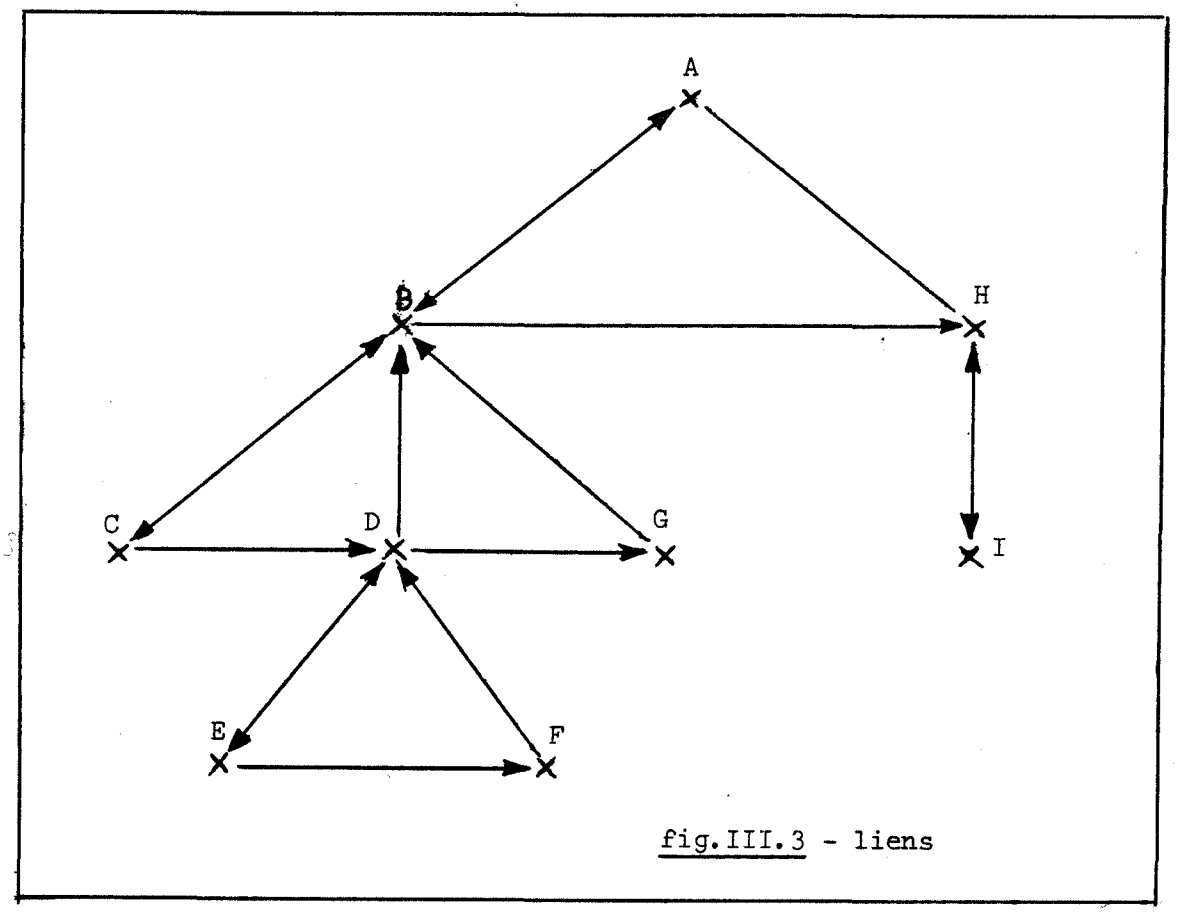


fig.III.3 - liens

Outre ces trois liens fondamentaux qui, définis pour chaque noeud de l'arbre, rendent possible son exploration complète, nous serons amenés à définir quelques fonctions spéciales permettant de relier directement certaines informations entre elles. La structure sera alors un véritable graphe.

Pour permettre à l'utilisateur d'implémenter la structure formelle des informations, il faut lui donner un moyen de décrire l'arbre qui la représente.

Nous avons défini un langage de définition de structure

.../...

(L.D.S.) qui autorise une présentation sous forme linéaire des éléments de la structure. Cette présentation, simple à comprendre, correspond à l'exploration dite en préordre de l'arbre.

2. 2. - LANGAGE DE DEFINITION DE STRUCTURE (L.D.S.)

Après une présentation de la grammaire du langage, nous nous attacherons à étudier en détail chacun des éléments du L.D.S. en précisant sa présentation syntaxique, sa signification sémantique et l'intérêt qui a justifié son introduction.

2.2.1. - Grammaire du L.D.S.

a) Nous présentons d'abord les règles de grammaire en utilisant un formalisme dérivé de la forme normale de la métalangue de BACKUS-NAUR et voisin de la description linguistique utilisée dans les manuels de référence de PHILIPS /7/ p.115.
La description du L.D.S. (fig.III.4) est composée d'un certain nombre de règles de réécriture qui comportent :

- des éléments non-terminaux entre crochets pointus (exemple : <BLOC>)

- des éléments terminaux qui sont soulignés. Ils appartiennent au L.D.S.
(exemple : <u>CHOIX</u>, <u>MOT</u>)

- les symboles

:: = ; | ; ( ) ; [ ] ; { }

Dans chaque règle de réécriture le non-terminal à remplacer figure à gauche du symbole " :: = " tandis que le remplacement à effectuer est défini à droite de ce symbole.

La bande verticale " | " correspond à un "ou" exclusif, son champ d'action est éventuellement limité à gauche (resp. à droite) par une autre barre verticale ou une ouverture (resp. fermeture) de parenthèse, crochet ou accolade.

.../...

```
< STRUCTURE > :: = < BLOC >  ***
< BLOC > :: = DEBUT < CARAC > { < CARAC > | OU < CARAC > } FIN
< CARAC > :: = < IDENT > [ < BLOC > | < TYPE-ELEM > ] | ENTITE < NB > < IDENT >
               ( CHOIX < IDENT > < LV > < NB > ) < BLOC >
< TYPE-ELEM > :: = [ MOT | TEXTE | < LV > ] < NB > | NUMERIQUE [ E | R | D ]
                   [ [ référence | INVERSE ] UN | IDEM ] < IDENT >
                   { DE (UN) < IDENT > }
< LV > :: = ( < IDENT > { < IDENT > } )
< IDENT > :: = lettre { carac-alpha. }
< NB > :: = chiffre { chiffre }
  lettre ∈ { A, B, C, ........ , Z }
  carac-alpha est un élément de l'ensemble des caractères alpha-
  numériques , chiffre ∈ { 0, 1, 2, ......, 9 }
```

fig.III.4 - grammaire du L.D.S.

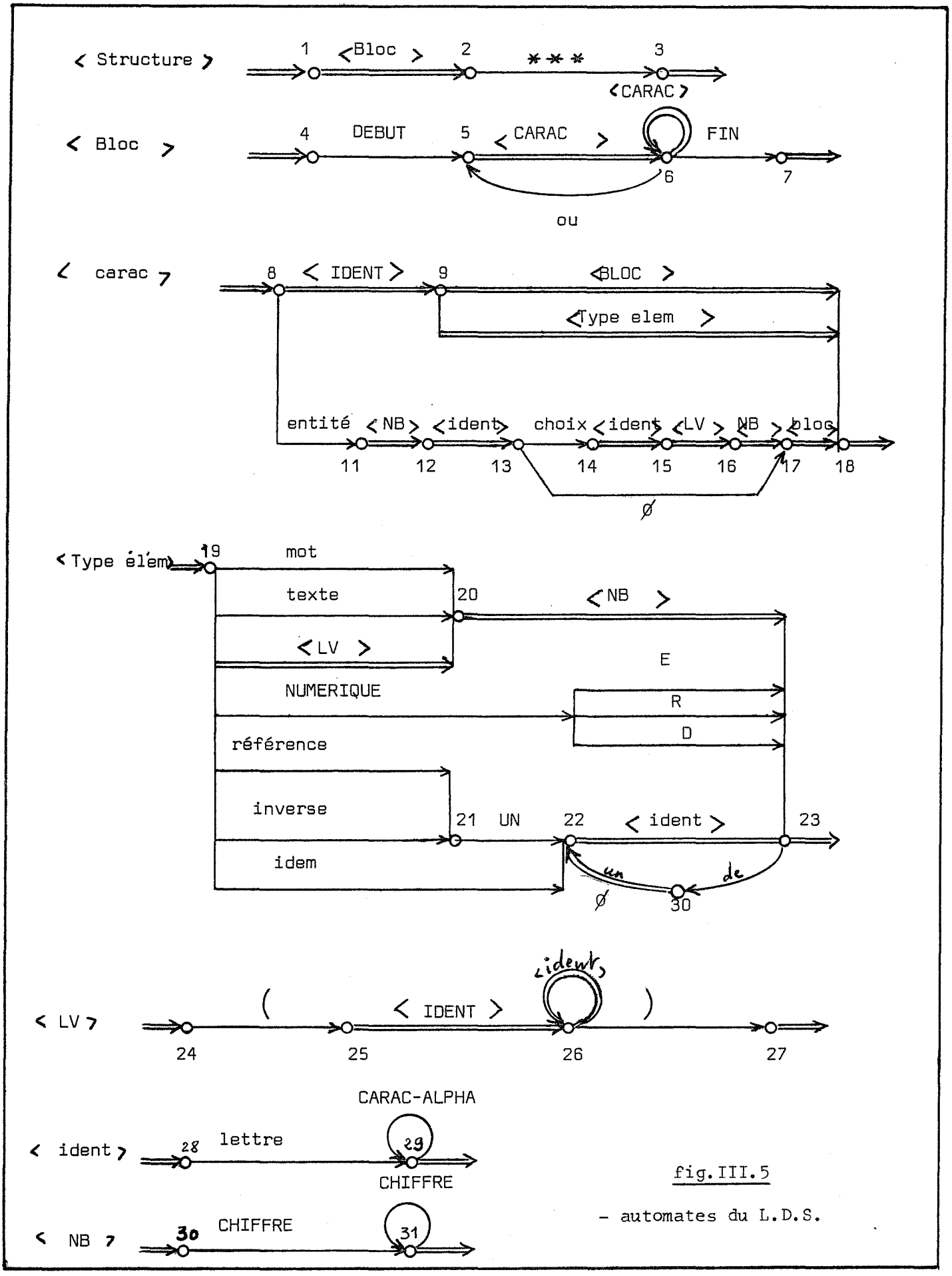


fig.III.5

- automates du L.D.S.

.../...

Les parenthèses " ( )" indiquent que leur contenu est optionnel. Les crochets " [ ] " indiquent que la prise en compte de l'un des éléments entre crochets est obligatoire.

Les portions de phrase placées entre accolade " { } " peuvent êtres omises, ou bien répétées un nombre quelconque de fois.

La description du langage par cette méthode permet une construction aisée des automates qui seront utilisés pour l'analyse syntaxique.

b) Les automates du L.D.S. sont rassemblés sur la figure III.5 Certaines transitions sont des appels à un automate. Un automate peut s'appèler lui-même. Ce sont donc des automates récursifs.

Ils permettent une analyse déterministique car ils satisfont aux deux conditions suivantes /1/ :

- il n'y a jamais deux transitions issues d'un état qui génèrent le même symbole de base.
- il y a au plus une transition de type $\emptyset$ issue d'un état donné de l'automate.

A chaque unité d'information ou caractéristique sont associés :

- un identificateur pour la désigner
- un type qui permet de connaitre sa nature avec éventuellement quelques paramètres la précisant.

Pour ne pas compliquer la lecture des automates, nous n'avons pas fait figurer les blancs qui servent de séparateurs entre les mots du langage

Un identificateur est une chaîne alphanumérique sans blanc (le blanc étant un séparateur) dont le premier caractère est une lettre. Il est utile de noter que seuls les 16 premiers caractères d'un identificateur sont pris en compte lors de l'analyse syntaxique.

2.2.2. - Description des caractéristiques :

Cette description fait référence à l'exemple donné figure III.6 qui représente l'arbre structurel dessiné figure III.7. (p.20 et p.21).

Il existe des caractéristiques simples qui correspondent à un seul élément d'information (mot, texte,....), des caractéristiques plus complexes qui groupent un certain nombre de caractéristiques (BLOC, ENTITE, ...)

2.2.2.1. - TYPE MOT :

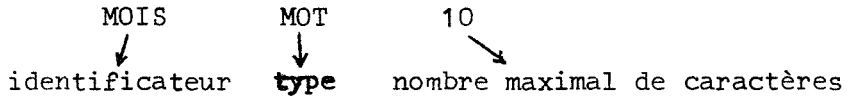
- définition syntaxique :

<IDENT> MOT <NB>

- explication :

La valeur de cette caractéristique simple est une chaîne de caractères alphanumériques (sans espace) le premier étant une lettre. Le nombre maximal de caractères est spécifié par <NB>

- exemple :



- cette caractéristique sert pour le stockage d'informations brèves comme des noms de personnes, des titres de publications, des marques de voitures....

Le nombre maximal de caractères doit être précisé car il détermine la réservation d'une place suffisante sur le support physique.

Il est possible de faire une recherche sur un constituant ayant ce type dans une requête ou un traitement.

2.2.2.2. - TYPE TEXTE :

- définition syntaxique :

<IDENT> TEXTE <NB>

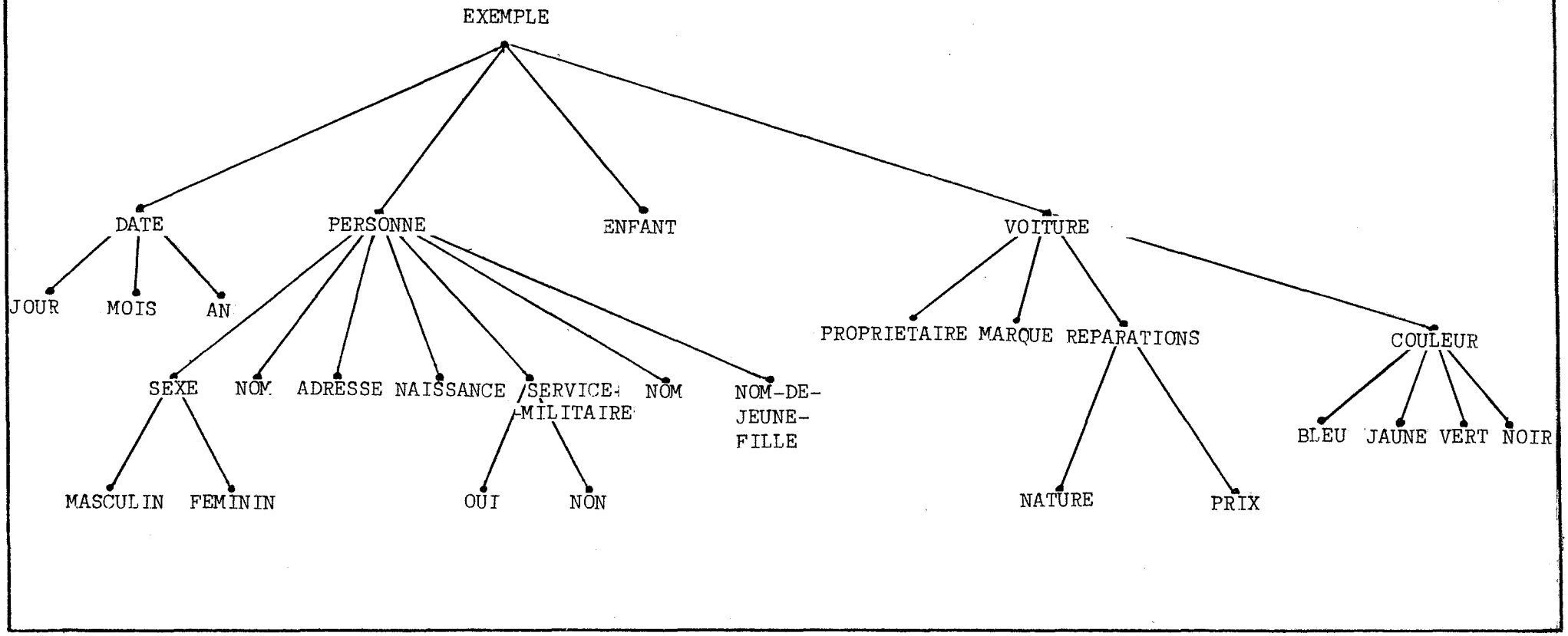
.../...

```
EXEMPLE
DEBUT
   DATE  DEBUT
          JOUR NUMERIQUE E
          MOIS MOT 10
          AN NUMERIQUE  E
       FIN
   ENTITE 100 PERSONNE CHOIX SEXE ( MASCULIN FEMININ) 2
       DEBUT
          NOM MOT 10
          ADRESSE TEXTE 2
          NAISSANCE IDEM DATE
          SERVICE-MILITAIRE ( OUI NON) 2
       OU
          NOM MOT 10
          NOM-DE-JEUNE-FILLE  IDEM NOM
      FIN
  ENFANT INVERSE UNE PERSONNE
  ENTITE 15 VOITURE
      DEBUT
          PROPRIETAIRE REFERENCE UNE PERSONNE
          MARQUE MOT 12
          ENTITE 25 REPARATIONS
            DEBUT
               NATURE MOT 16
               PRIX NUMERIQUE R
            FIN
          COULEUR (BLEU JAUNE VERT NOIR ) 11
       FIN
FIN  ***
```

fig.III.6 - "EXEMPLE" en L.D.S.

.../...

fig.III.7 - structure "EXEMPLE"



.../

- explication :

La valeur de cette caractéristique simple est une chaîne de caractères alphanumériques quelconques. <NB> indique le nombre maximal autorisé de lignes de 60 caractères.

- exemple :

ADRESSE	TEXTE	2
↓	↓	↘
identificateur	type	nombre de lignes de 60 caractères (15 mots)

La valeur de ADRESSE sera une chaine alphanumérique d'au plus 2 lignes soit 120 caractères.

Comme son nom l'indique, cette caractéristique est adaptée aux informations qui se présentent sous la forme d'un texte : résumé d'une publication, article de code, adresse, théorème, courbe, dessin ...

L'indication du nombre de lignes de 60 caractères est obligatoire pour la réservation de place en mémoire.

Cette caractéristique sert essentiellement à l'édition.

2.2.2.3. - TYPE NUMERIQUE :

- Définition syntaxique :

<IDENT> NUMERIQUE [E | R | D]

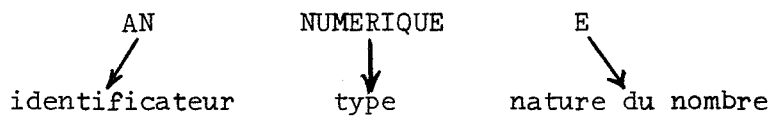
- explication :

Suivant la lettre qui suit NUMERIQUE, la représentation interne du nombre permettra d'exprimer :

* soit des entiers ———————— lettre = E
* soit des réels en simple précision ——— lettre = R
* soit des réels en double précision ——— lettre = D

.../...

- exemple :



Dans la plupart des applications on peut être amené à mémoriser puis traiter des données numériques, d'où la nécessité d'introduire ce type de caractéristique.

2.2.2.4. - TYPE LISTE DE VALEURS (LV)

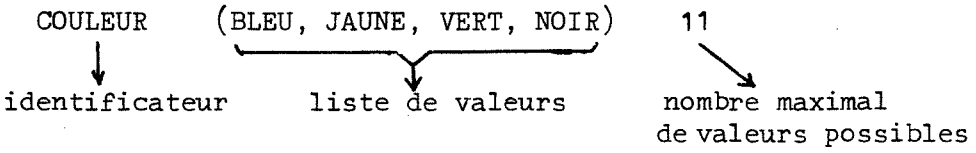
- définition syntaxique :

<IDENT> <LV> <NB>

- explication :

La valeur de cette caractéristique simple est l'une des chaînes alphanumériques composant <LV> . Le nombre maximal de chaînes possibles est spécifié par <NB>

- exemple :



La valeur de COULEUR est l'une des chaînes alphanumériques précisées dans la liste contenue dans le jeu de parenthèses (les espaces servant de séparateurs). Dans cet exemple le nombre total des chaînes possibles dans une mise à jour ultérieure de la structure est 11.

- L'utilisation de cette caractéristique est justifiée lorsque l'information à conserver est toujours choisie parmi une liste de valeurs définies à l'avance. Cela permet une vérification lors de la mise à jour car il n'est alors possible de ranger que des valeurs permises. Ces valeurs sont mémorisées sous une forme codée transparente à l'utilisateur.

.../...

2.2.2.5. - TYPE IDEM :

- définition syntaxique :

<IDENT> IDEM <IDENT>

- explication :

Cette caractéristique désignée par le <IDENT>de gauche est du type de la caractéristique désignée par le <IDENT> de droite.

- exemple :

NOM-DE-JEUNE-FILLE	IDEM	NOM
↓	↓	↘
identificateur	type	identificateur d'une autre caractéristique

La caractéristique NOM-DE-JEUNE-FILLE aura le même type que la 1ère caractéristique NOM rencontrée dans la structure c'est-à-dire sera un mot de 10 caractères .

Le type IDEM ne définit pas une nouvelle forme de stockage d'information mais évite la répétition de spécifications déjà précisées à un autre endroit de la structure.

2.2.2.6. - TYPE BLOC :

- définition syntaxique :

<IDENT> DEBUT <CARAC>{<CARAC>} FIN

- explication :

Un BLOC est un ensemble de caractéristiques (pouvant être elles-mêmes du type BLOC) auquel est associé un identificateur. La liste des caractéristiques constituant un BLOC est encadrée par les mots DEBUT et FIN.

- exemple :

DATE DEBUT JOUR..... NUMERIQUE E FIN

identificateur caractéristiques composant le bloc

DATE est un bloc constitué par les informations JOUR, MOIS, AN.

L'examen de la grammaire du LDS (fig.III.4) montre que toute structure est un BLOC.

Cette caractéristique est utilisée chaque fois qu'il est nécessaire de désigner globalement un ensemble de caractéristiques. Il est important de noter que son introduction crée un niveau supplémentaire dans l'arbre structurel utilisé et qu'il faut en tenir compte dans les citations qui permettent l'accès aux informations (/9/ - La promenade dans la structure par la méthode" depth-first). Par exemple, toute mise à jour ou interrogation de la caractéristique AN du bloc DATE fait intervenir l'élément de citation AN DE LA DATE.

- Remarque :

Dans l'entité choix nous serons amenés à utiliser une notion de bloc dont la syntaxe est légèrement plus complexe (cf. type ENTITE CHOIX 2.2.2.8)

2.2.2.7. - TYPE ENTITE

- définition syntaxique :

ENTITE <NB> <IDENT> <BLOC>

- explication :

Une entité représente un ensemble de blocs ayant même structure. L'ensemble des informations associées à un de ces blocs est appelé réalisation de l'entité. Chaque réalisation est numérotée.

<NB> désigne le nombre maximal de réalisations possibles de l'entité.

.../...

- exemple :

ENTITE	15	VOITURE	DEBUT	MARQUE..... FIN
↓	↓	↓		
type	nombre d'objets	identificateur		

L'entité VOITURE permet de stocker 15 réalisations numérotées de 1 à 15 et contenant chacune les informations MARQUE, REPARATIONS, COULEUR.
On considère, par exemple, les informations :

MARQUE DE LA VOITURE 7

COULEUR DE LA VOITURE 14

- Cet exemple montre que l'entité, analogue au "Repeating Group" défini dès 1967 dans le système TDMS /4/, est fondamentale dans la définition d'une base de données. Le fait qu'elle permette de traiter un ensemble d'objets de structure identique peut être exploité avec profit dans beaucoup d'applications telles que : fichiers de clientèle, fichier de cartes grises, documentation, propriétés physique de métaux .....

Le nombre de réalisations possibles doit être obligatoirement précisé afin de pouvoir procéder au calcul des adresses dans le fichier - données (cf. 3.3) ; Cette contrainte n'a pas de signification conceptuelle mais provient de la réalisation statique qui consiste à affecter une adresse virtuelle définitive à toute information. (cf.3.3)

2.2.2.8. - TYPE ENTITE CHOIX :

- définition syntaxique

ENTITE <NB> <IDENT> CHOIX <LV> <BLOC>

avec

<BLOC> :: = DEBUT <CARAC>{<CARAC> | OU <CARAC>} FIN

- explication :

Une entité choix représente un ensemble de blocs.
Chaque réalisation de l'entité correspond à un bloc qui comprend la caractéristique liste de valeurs placée après le mot CHOIX et, suivant

la valeur de cette caractéristique, les informations indiquées soit entre DEBUT et OU, soit entre deux OU successifs, soit entre OU et FIN.

Les flèches du schéma explicatif (III.8) indiquent cette correspondance.

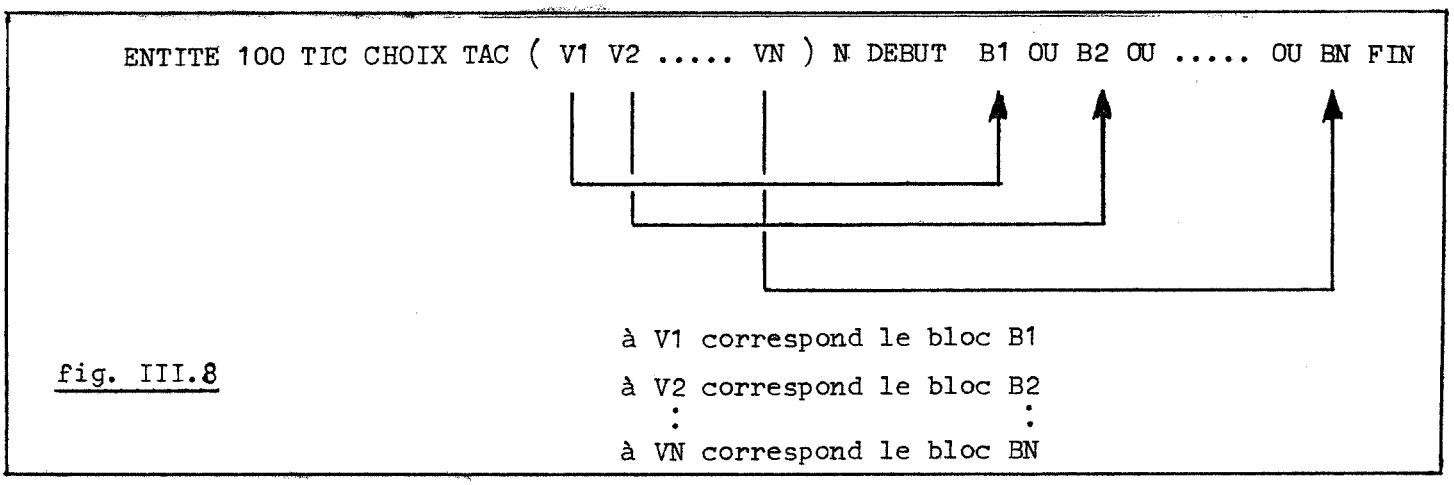
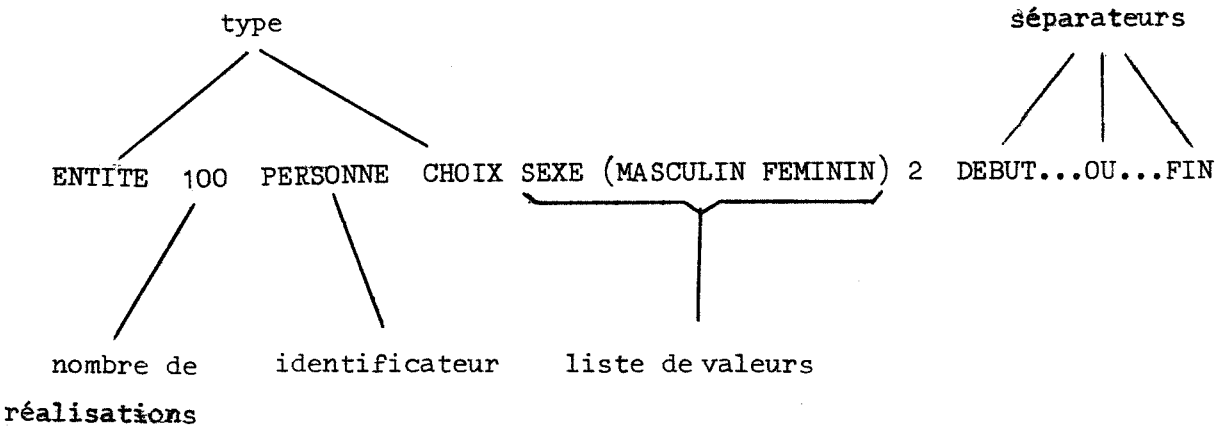


fig. III.8

- exemple :



Dans cet exemple, les réalisations pour lesquelles la caractéristique SEXE aura la valeur MASCULIN comprendront les informations NOM, ADRESSE, NAISSANCE, SERVICE-MILITAIRE, tandis que celles pour lesquelles SEXE aura la valeur FEMININ comprendront les informations NOM et NOM-DE-JEUNE-FILLE.

.../...

A l'aide de cet exemple, nous pouvons essayer de dégager l'intérêt de la création d'entités de type CHOIX.

En effet, si les informations contenues dans cet exemple, sont prises en compte en utilisant seulement la caractéristique ENTITE, la place correspondant aux caractéristiques :

SEXE
NOM
ADRESSE
NAISSANCE
SERVICE-MILITAIRE
NOM-DE-JEUNE-FILLE

doit être réservée pour chaque réalisation alors que, dès l'élaboration de la structure, il est prévisible que certaines de ces caractéristiques n'auront aucun sens pour une partie des réalisations. Certaines de ces caractéristiques s'excluent mutuellement. Ainsi aucune réalisation de l'exemple cité ne pourra comporter des informations à la fois pour SERVICE-MILITAIRE et pour NOM-DE-JEUNE-FILLE.

Il nous est donc apparu souhaitable de pouvoir associer dans une même entité des réalisations ne correspondant pas à la même structure, c'est-à-dire au même bloc de caractéristiques, mais ayant cependant un lien commun.

L'introduction de cette possibilité au niveau de la définition de la structure entraîne un accroissement de la complexité des programmes d'implantation de la structure et d'interprétation des requêtes d'accès aux informations.

Le gain de place dans le fichier-données (voir 3.3) peut dans certains cas être très important mais il se situe au niveau virtuel et n'a pas d'incidence au niveau physique. (Il peut dans certains cas éviter la saturation de l'espace virtuel).

La possibilité de pouvoir désigner sous le même nom et de faire figurer au même endroit de l'arbre structurel des objets ne possèdant pas obligatoirement les mêmes caractéristiques permet de

simplifier, pour l'utilisateur, la formulation des requêtes d'accès aux informations.

D'autre part, l'économie d'un niveau dans l'arbre (f.III.9) se traduit par un gain de temps appréciable lors de l'exécution des requêtes comme le montrent les résultats des mesures effectuées. (voir partie IV).

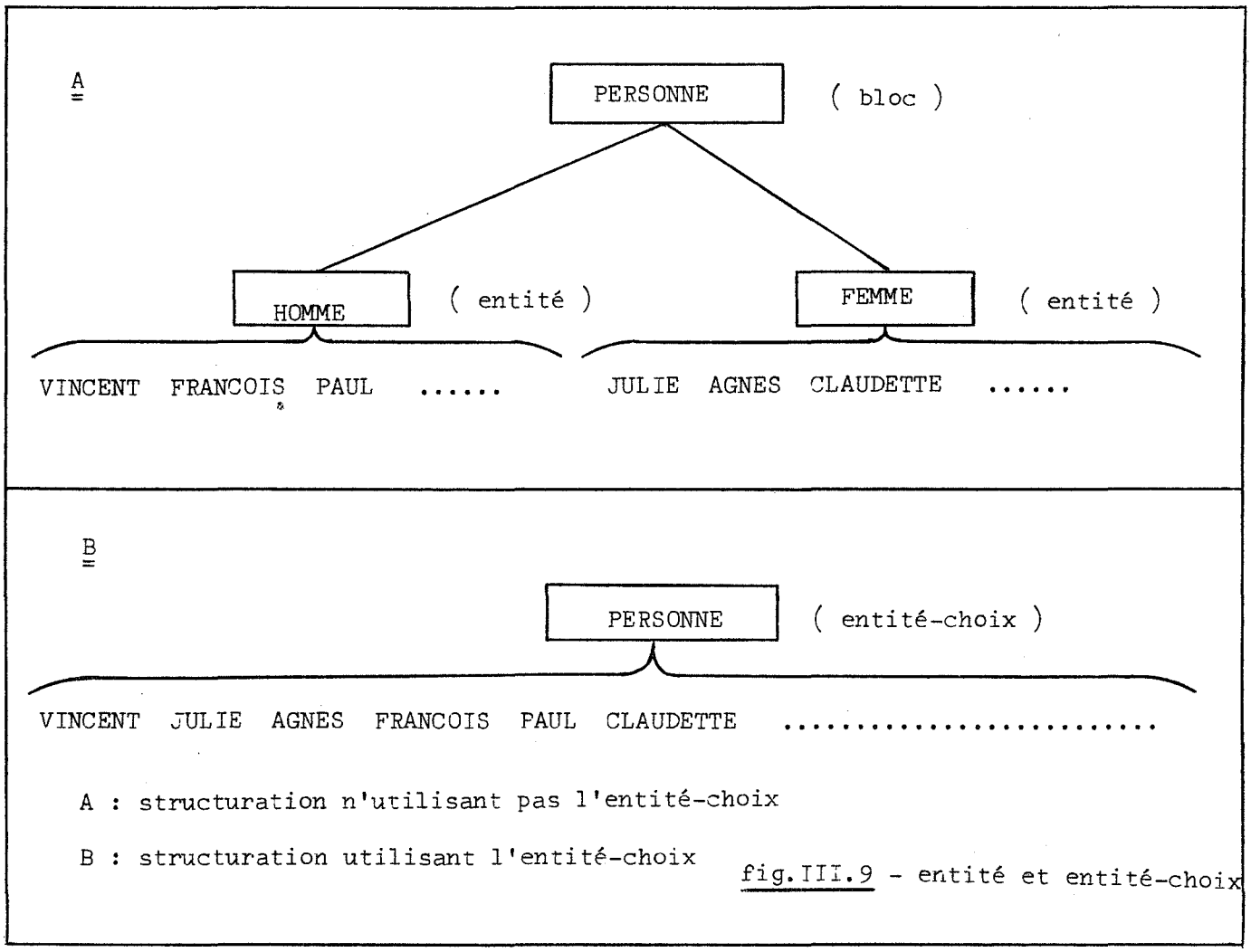


fig.III.9 - entité et entité-choix

L'exemple donné au début de ce paragraphe et la fig.III.8 montrent que le "choix" entre les diverses structures de bloc possibles pour une réalisation s'opère à partir d'une caractéristique du type "liste de valeurs" dont la valeur sera toujours en tête des informations de chaque réalisation de l'entité choix.

.../...

2.2.2.9. - TYPE REFERENCE :

- définition syntaxique :

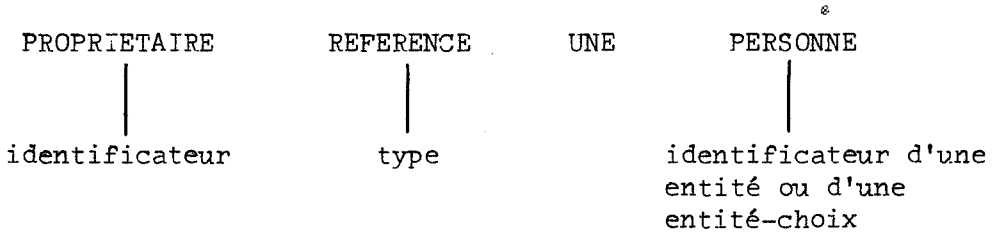
<IDENT> REFERENCE [UN | UNE] <IDENT>

Cette caractéristique désignée par le <IDENT> de gauche est définie par référence à une entité ou une entité-choix dont l'identificateur est cité par le <IDENT> de droite.

La REFERENCE ne peut être définie (par conséquent n'a de sens) que si la caractéristique citée a été définie antérieurement dans la structure.

La caractéristique REFERENCE n'a de valeur que si la caractéristique citée en a déjà une.

- exemple :



PROPRIETAIRE désigne une réalisation de l'entité PERSONNE.

Les caractéristiques de la structure arborescente décrite jusqu'à présent ne nous permettent de définir que des relations d'appartenance. Par exemple, dans la structure de la figure III.7 :

- UNE PERSONNE donnée appartiendra à l'ensemble nommé EXEMPLE
- UNE DATE-DE-NAISSANCE appartiendra à l'ensemble d'informations caractéristique d'une personne donnée.
- UN NOM et un NOM-DE-JEUNE-FILLE donnés appartiendront tous les deux au même ensemble PERSONNE etc....

.../...

Le type REFERENCE enrichit la structure formelle en lui donnant la forme d'un graphe.

Dans l'ensemble donné ci-dessus, la caractéristique PROPRIETAIRE dans une réalisation de l'entité VOITURE est liée à l'existence d'une réalisation déterminée de l'entité PERSONNE. Un lien est ainsi créé entre les deux entités. Lors de la disparition d'une PERSONNE donnée , la mise à jour de la caractéristique PROPRIETAIRE devra se faire automatiquement (cf. 3.3.4.).

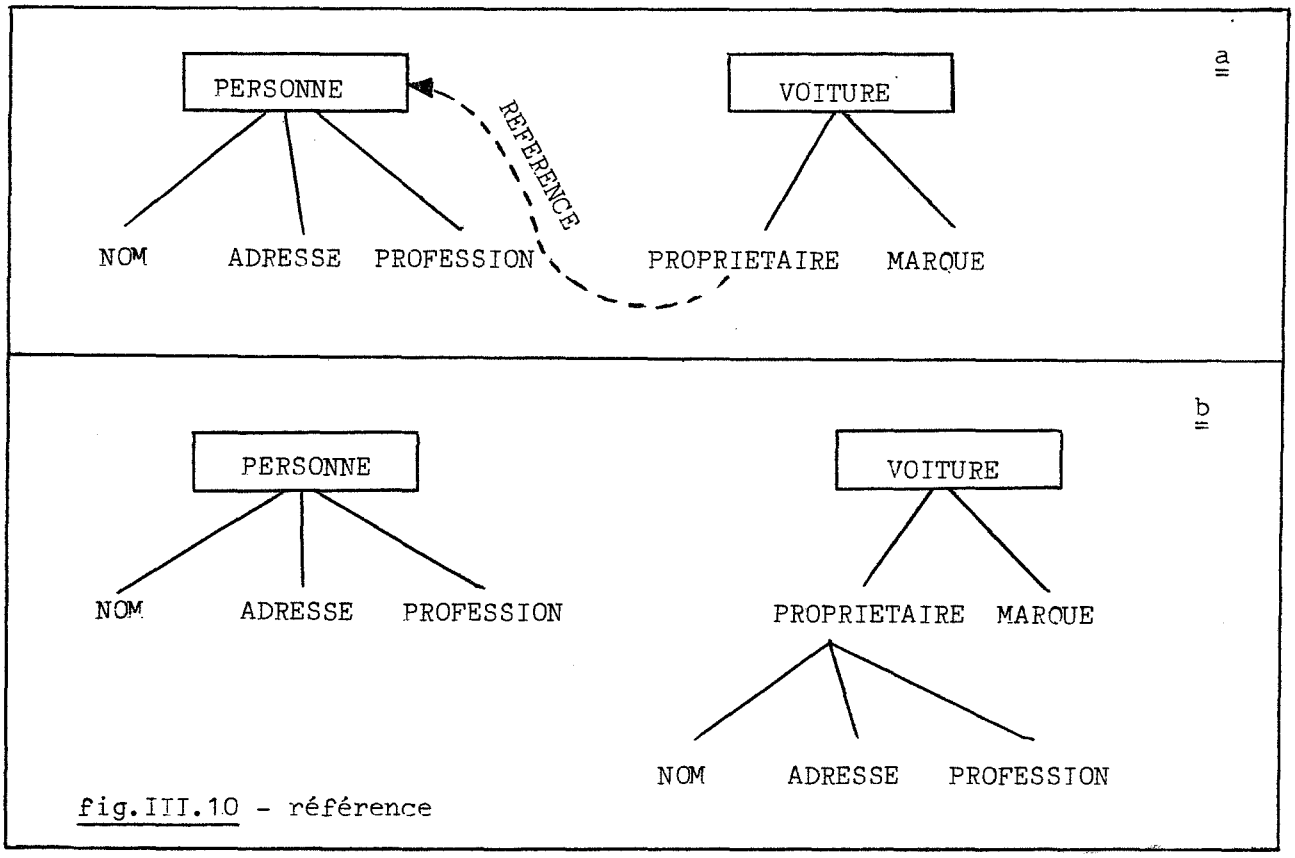


fig.III.10 - référence

Au point de vue des accès, la structure utilisant une référence et représentée fig.III.10.a est semblable à la structure de la fig.III.10.b . La présence d'une référence correspond donc en fait à un niveau supplémentaire dans la structure. Elle permet, tout en évitant des duplications avec tous les problèmes que cela comporte notamment pour les mises à jour, d'accèder à des informations par plusieurs niveaux logiques de la structure. Signalons, en outre, qu'à la non redondance des informations correspond toujours un gain de place dans les fichiers.

.../...

2.2.2.10. - TYPE INVERSE :

- Définition syntaxique :

<IDENT> INVERSE [UN | UNE] <IDENT>

L'introduction du type INVERSE correspond à la nécessité de pouvoir retrouver rapidement les objets d'une entité (dont l'identificateur correspond à l' <IDENT> de droite) possèdant une certaine propriété (désignée par l' <IDENT> de gauche)

- exemple :

ENFANT	INVERSE	UNE	PERSONNE
identificateur	type		identificateur d'une entité

La valeur de ENFANT sera une chaîne de bits de dimension égale au nombre de réalisations possibles de l'entité PERSONNE. La Ième réalisation de l'entité PERSONNE possèdera la propriété ENFANT si le Ième bit de la chaîne inverse est à 1. (Rappelons que chaque réalisation d'une entité est numérotée. Cf TYPE ENTITE).

3 - REPRESENTATION DE LA STRUCTURE SUR LE SUPPORT PHYSIQUE

3.1. - L'ORGANISATION DES DONNEES EN MEMOIRE VIRTUELLE :

3.1.1 - Technique statique :

Alors que dans la technique d'organisation dite dynamique, les adresses sont définies dynamiquement à la création des valeurs, dans la technique statique à chaque donnée potentielle est affectée une adresse fixe et une place de longueur fixe dans la mémoire virtuelle.

La gestion de mémoire adoptée permettant de disposer d'une mémoire virtuelle de dimension très grande nous avons préféré la deuxième technique (plus simple) à la première.

L'organisation statique commande donc de travailler avec des informations de structure et de taille fixes. Ces éléments étant connus et mémorisés lors de la définition de la structure il est possible alors de déterminer, en y faisant référence, l'adresse virtuelle de toute information et d'y accèder directement sans avoir besoin de suivre une chaîne de pointeurs.

3.1.2. - Les fichiers :

Une base SOMINE utilise deux fichiers créés et gérés à travers le système de gestion de mémoire défini dans le chapitre 2.

Le premier, que nous appellerons le FICHIER-STRUCTURE, est destiné à contenir la structure formelle suivant laquelle

.../...

les informations seront mémorisées dans le deuxième fichier appelé FICHIER-DONNEES.

Nous étudierons d'abord la méthode retenue pour représenter la structure dans le premier fichier puis, pour chaque caractéristique, la manière dont les informations sont organisées dans le fichier-données à partir des renseignements figurant dans le fichier-structure.

3. 2. - LE FICHIER-STRUCTURE :

3.2.1. - Le plexe :

Le plexe est une structure bien adaptée pour représenter des arbres et des graphes dirigés dans un ordinateur /2/. Il est composé d'un ensemble d'éléments appelés grains ; chaque grain est un vecteur comportant un nombre constant de mots de mémoire.

Dans une telle structure, les informations et les pointeurs relatifs à un noeud donné sont contenus dans un vecteur ayant toujours le même format. Cela permet de faciliter la programmation par l'utilisation d'incrémentations à pas constants.

Nous avons choisi d'utiliser des grains de 15 mots pour les deux raisons principales suivantes :

- la mémoire virtuelle peut-être facilement décomposée en sous-pages de 15 mots

- 15 mots suffisent pour contenir l'identificateur d'une caractéristique, les index nécessaires pour l'exploration de l'arbre, les variables utiles au calcul de la taille des informations et de leur adresse dans le fichier-données.

3.2.2. - Format d'un grain :

Nous appelons ADRESSE STRUCTURELLE d'une caractéristique l'adresse virtuelle du vecteur qui la représente dans le fichier-structure .

3.2.2.1. - Schéma d'un grain :

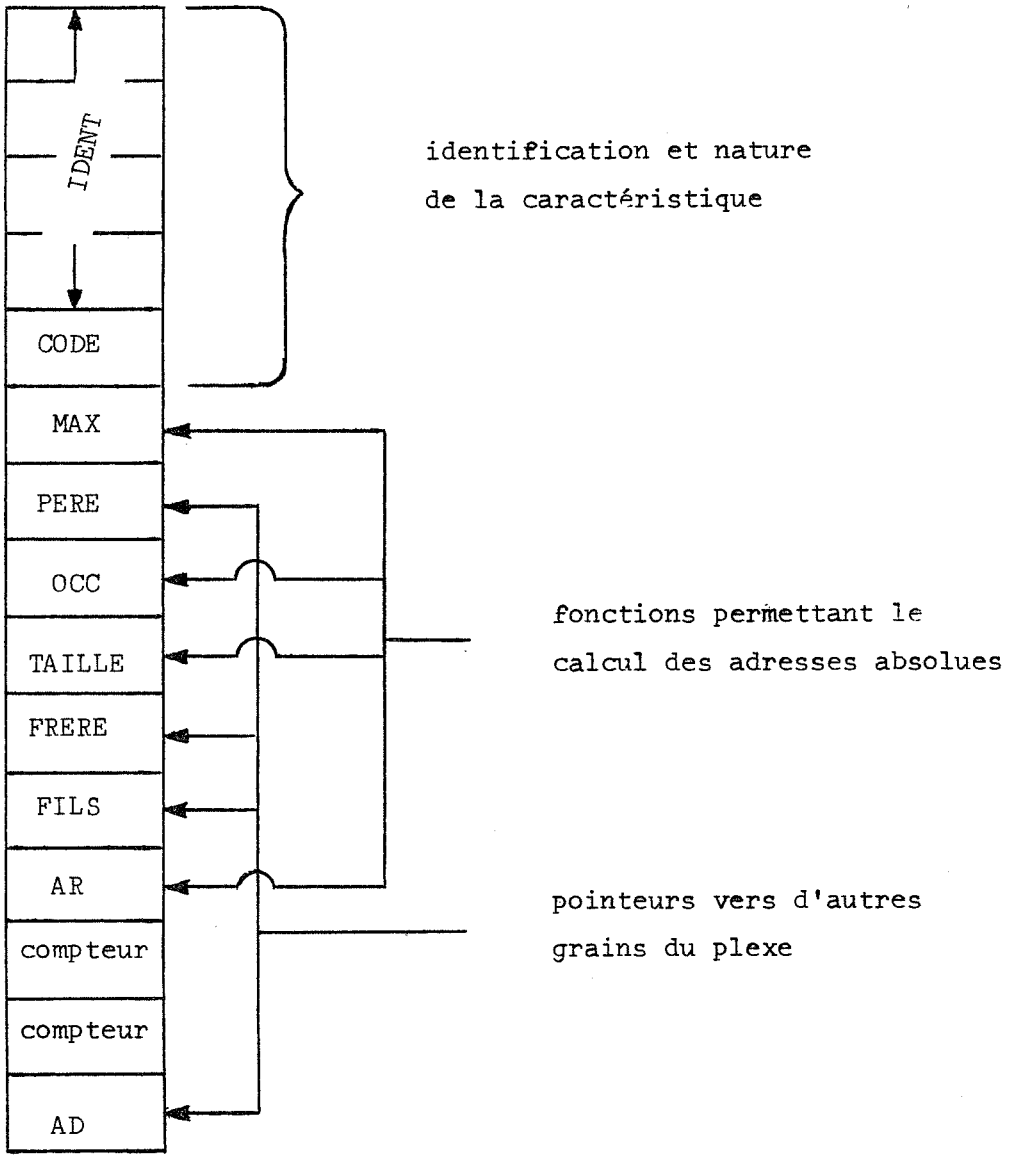


Fig. III. 11

.../...

3.2.2.2. - Identification et nature de la caractéristique :

IDENT : Les quatre premiers mots du vecteur sont réservés au stockage de l'identificateur de la caractéristique, ce qui implique que seuls les 16 premiers caractères d'un identificateur sont pris en compte.

CODE : le 5ème mot contient un nombre entier compris entre 1 et 14 et qui indique la nature de la caractéristique conformément au tableau suivant:

1	FICHIER
2	BLOC
3	ENTITE
4	REFERENCE
5	INVERSE
6	TEXTE
7	NUMERIQUE E
8	LISTE DE VALEURS
9	MOT
10	ENTITE - CHOIX
11	IDEM
12	NUMERIQUE R
13	NUMERIQUE D
14	LISTE DE VALEURS CHOIX

Fig. III.12 - Code des caractéristiques

.../...

3.2.2.3. - Pointeurs vers les autres grains du plexe :

PERE- FRERE-FILS :

Les mots correspondant contiennent respectivement les adresses structurelles du PERE, du 1er FRERE, du 1er FILS (fonctions définies dans le paragraphe 2-1) ou à défaut , - 1.

AD : Le pointeur AD ne prend une valeur que pour les caractéristiques de type REFERENCE, INVERSE ou IDEM . Dans ces cas, la valeur prise par AD est l'adresse structurelle de la caractéristique citée par la REFERENCE, l'INVERSE ou l'IDEM.

3.2.2.4. - Fonctions de calcul :

Nous appellerons ADRESSE ABSOLUE d'une information valeur d'une caractéristique son adresse virtuelle dans le fichier-données. Nous dirons souvent pour simplifier "adresse absolue de la caractéristique", aucune confusion n'étant possible avec l'adresse structurelle.

MAX : La fonction MAX sert au calcul de la taille maximale d'une caractéristique . Sa valeur est :

- le nombre maximum de réalisations pour une entité ou une entité-choix
- le nombre maximum de caractères pour un MOT
- le nombre maximum de lignes d'un TEXTE
- le nombre maximum d'éléments d'une LISTE de VALEURS.

TAILLE : Cette fonction a pour valeur la dimension évaluée en mots de la place occupée par la caractéristique dans le fichier-données.

Pour les entités nous avions le choix entre noter la taille de l'entité prise dans son ensemble et conserver la taille d'une réalisation.

.../...

La nécessité, au niveau des requêtes, de connaître pour les différents accès la taille des réalisations nous a fait adopter la deuxième solution. Le calcul de la taille d'une entité connaissant le nombre de réalisations et la taille d'une réalisation ne pose par ailleurs aucun problème.

Pour les entités-choix, c'est la taille de la plus grande des réalisations possibles qui est retenue.

OCC (Occurrence)
La fonction OCC possède la valeur 0 sauf pour les caractéristiques d'une entité-choix. Elle prend alors pour valeur le numéro du "choix". L'introduction de cette notion d'occurrence a été indispensable pour permettre de différencier les caractéristiques appartenent à des structures de bloc distinctes dans le cas d'une entité-choix.

AR

AR sert à conserver l'Adresse Relative dans le fichier-données d'une caractéristique par rapport à sa caractéristique mère. Il n'est en effet pas possible, du fait de la présence d'entités, de prévoir le stockage d'adresses absolues. Le calcul de ces dernières devra intervenir lors de chaque opération d'accès à l'information.

3.2.2.5. - Compteurs :

L'exécution des requêtes se faisant toujours à travers les grains de plexe du fichier-structure correspondant à l'information considérée, il est possible d'utiliser les deux mots restants comme compteurs.

Ces compteurs que l'utilisateur pourra relever grâce à des requêtes spéciales /8/, permettent de connaitre le nombre d'accès effectués pour une information donnée (interrogation et mise à jour) et, en conséquence, de pouvoir éventuellement modifier l'organisation de la banque pour en améliorer les services (voir chapitre IV).

.../...

3.2.2.6. - Remarque :

En exception avec ce qui précède, les caractéristiques LISTE de VALEURS et LISTE de VALEURS CHOIX utilisent des vecteurs supplémentaires pour stocker les éléments des listes de valeurs et les adresses structurelles de début des blocs choix.

3. 3. - LE FICHIER - DONNEES :

Après avoir analysé la composition du fichier-structure, nous allons étudier maintenant comment sont organisées physiquement les informations dans le fichier-données.

Les informations vont être affectées en mémoire virtuelle à partir des indications contenues dans le fichier-structure .

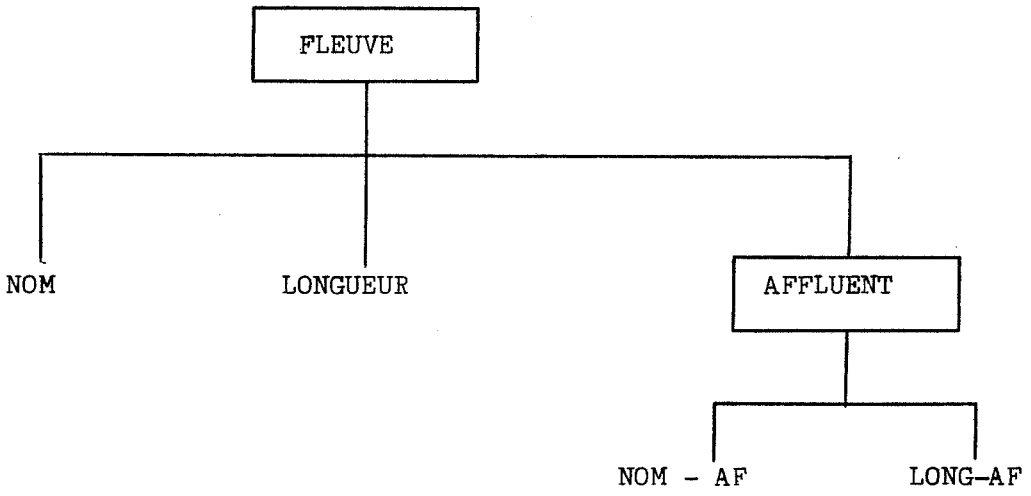
3.3.1. - Les types d'organisation :

Deux types principaux d'organisation peuvent être distingués /3/ :

l'organisation dispersée et

l'organisation verticale

Pour les exposer brièvement nous nous appuierons sur l'exemple de structure de la figure III.13. (p.40) .



FLEUVE et AFFLUENT sont des ENTITES

NOM, LONGUEUR, NOM- AF et LONG-AF sont des caractéristiques simples

fig.III.13 - structure "FLEUVE"

a) Organisation dispersée :

Dans cette forme d'organisation, les informations relatives à un même BLOC ou à une même réalisation d'ENTITE sont regroupées. Les informations concernant différentes réalisations sont juxtaposées. Ce qui donne le schéma de la fig.III.14. a pour l'exemple de la figure III.13.

Cette manière de procéder permet d'obtenir l'ensemble des données relatives à un fleuve en lisant un petit nombre de pages réelles puisque des données qui se suivent en mémoire virtuelle ont de grandes chances de se suivre aussi en mémoire réelle.

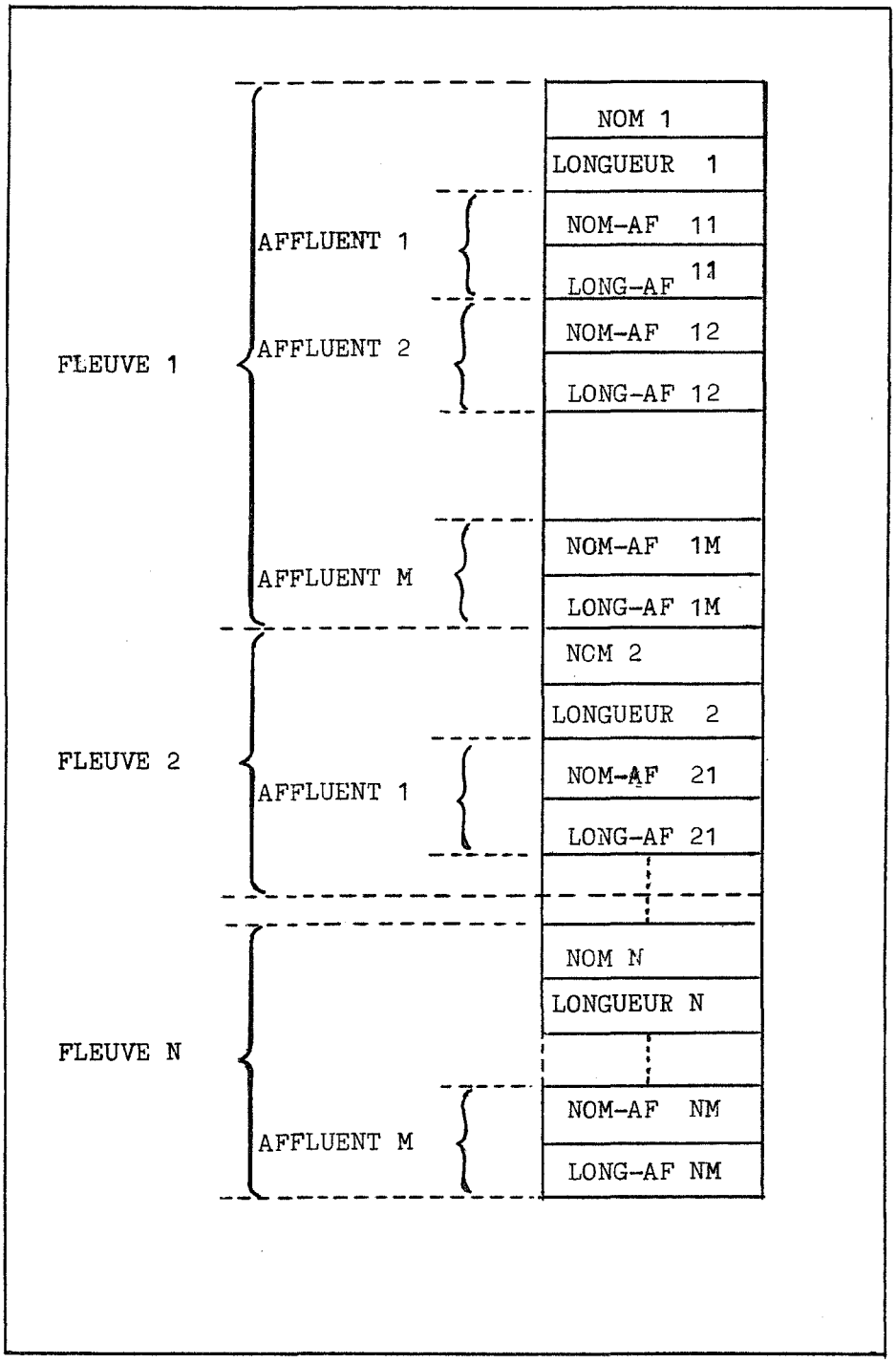


Fig.III.14.a - Organisation dispersée

b) Organisation verticale :

Dans ce type d'organisation, les informations correspondant à une même caractéristique sont regroupées entre elles. Pour la structure de la figure III.13 nous trouvons par exemple (fig.III.14.b) la suite des noms des fleuves, puis la liste ordonnée de leurs longueurs puis la liste des noms des affluents (NOM-AF) du 1er fleuve, puis du 2ème ....

Cette disposition facilite l'exploration séquentielle des éléments d'une rubrique. On peut en effet, par un accès à un nombre réduit de pages physiques, obtenir les valeurs d'une caractéristique pour toutes les réalisations d'une entité (LONGUEUR de tous les FLEUVES) et par exemple sélectionner les réalisations qui satisfont à certaines conditions portant sur cette caractéristique (FLEUVES dont la longueur est supérieure à 300 KM).

Par contre, les renseignements concernant une réalisation donnée (FLEUVE 2 ou AFFLUENT 3 du FLEUVE 4) sont éparpillés dans la mémoire virtuelle donc dans la mémoire réelle et leur rassemblement éventuel nécessiterait des accès à de nombreuses pages physiques et un cheminement faisant intervenir de nombreux pointeurs.

c) Organisation mixte :

Il est possible également de réaliser un compromis entre les deux options précédentes et d'utiliser un type d'organisation mixte dans lequel une partie des informations est en organisation dispersée dans une certaine zone de mémoire et l'autre partie en organisation verticale dans une autre zone de mémoire.

Pour notre projet, nous avons adopté essentiellement la méthode de l'organisation dispersée car il nous a semblé plus intéressant pour les accès envisagés dans différentes applications de laisser regroupées entre-elles les informations relatives à une même entité. Toutefois, compte-tenu des services qu'elle peut rendre, nous avons estimé nécessaire de permettre l'organisation verticale pour certaines

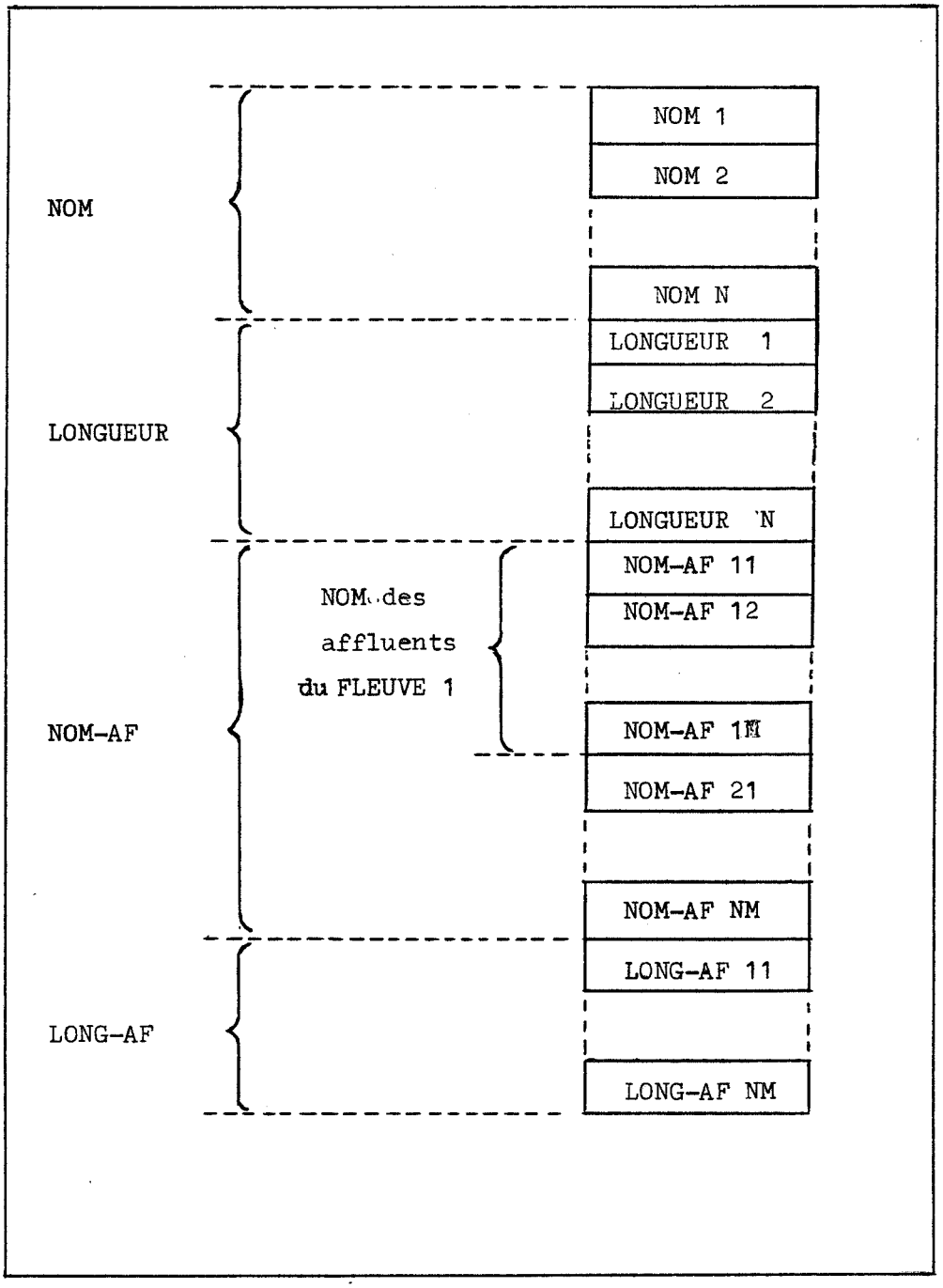


Fig. III. 14 b - Organisation verticale

.../...

données en créant notamment les caractéristiques de type INVERSE. Dans la même optique, nous avons mis à l'étude l'introduction d'une caractéristique DISCRIMINANTE basée sur la verticalité et permettant un accès semi-direct (la valeur de la caractéristique servant de clé de comparaison).

Voyons maintenant le détail de la représentation des données correspondant à chaque type de caractéristique dans le fichier-données.

3.3.2. - BLOC - Caractéristiques simples :

Pour ces caractéristiques nous allons simplement donner un exemple, aucune difficulté spéciale n'étant observée.

```
DEBUT
  NOM MOT 10
  ADRESSE TEXTE  2
  SERVICE MILITAIRE ( OUI NON) 2
  AGE NUMERIQUE   E
FIN
```

(cf.fig.III.15)

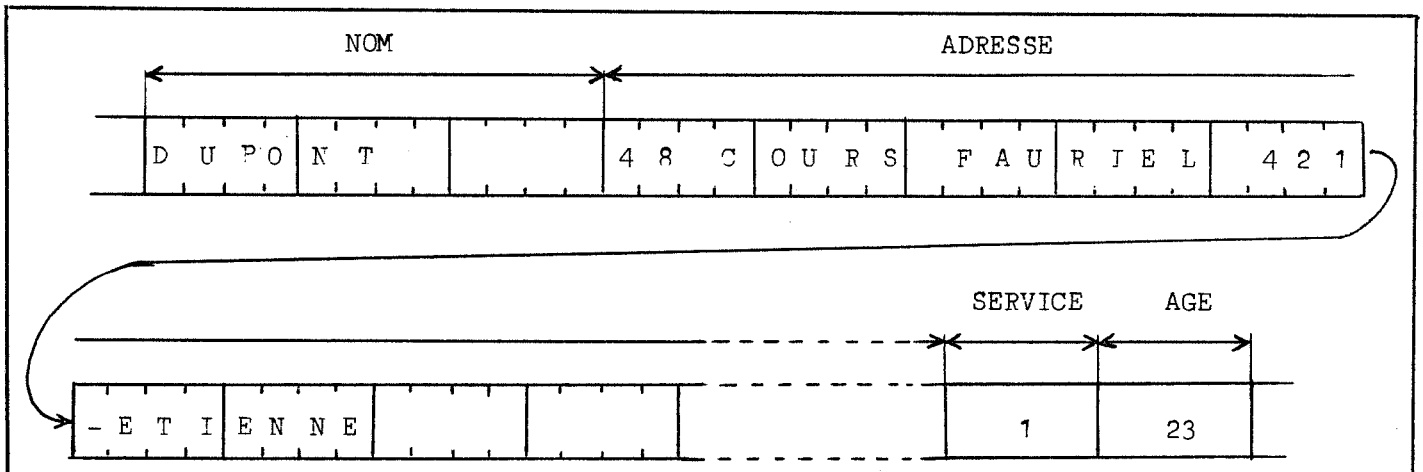


Fig.III.15 - représentation d'un bloc dans le fichier-données

Remarque :

Le système travaillant sur des frontières de mots la place occupée par la caractéristique NOM est un nombre entier de mots soit 3.

3.3.3. - IDEM :

La caractéristique IDEM ne nécessite que la réservation de la place correspondant aux informations liées à la caractéristique citée.

3.3.4. - ENTITE . ENTITE-CHOIX

exemple :

```
ENTITE 20 REPARATIONS
    DEBUT
      NATURE MOT 16
      PRIX NUMERIQUE R
    FIN
```

(cf.fig.III.16 p.46)

Au début de la représentation d'une entité dans le fichier-données figure toujours une en-tête comprenant :

- un mot contenant le nombre de réalisations existantes
- un nombre de mots correspondant à la taille de la chaîne des bits de présence. Ce nombre est égal à :

$$(MAX - 1) / 32 + 1$$

La chaîne des bits de présence permet d'obtenir rapidement une image de l'état des réalisations

Chaque réalisation commence par un mot destiné à contenir un pointeur vers la chaîne des références (cf. 3.3.5.).

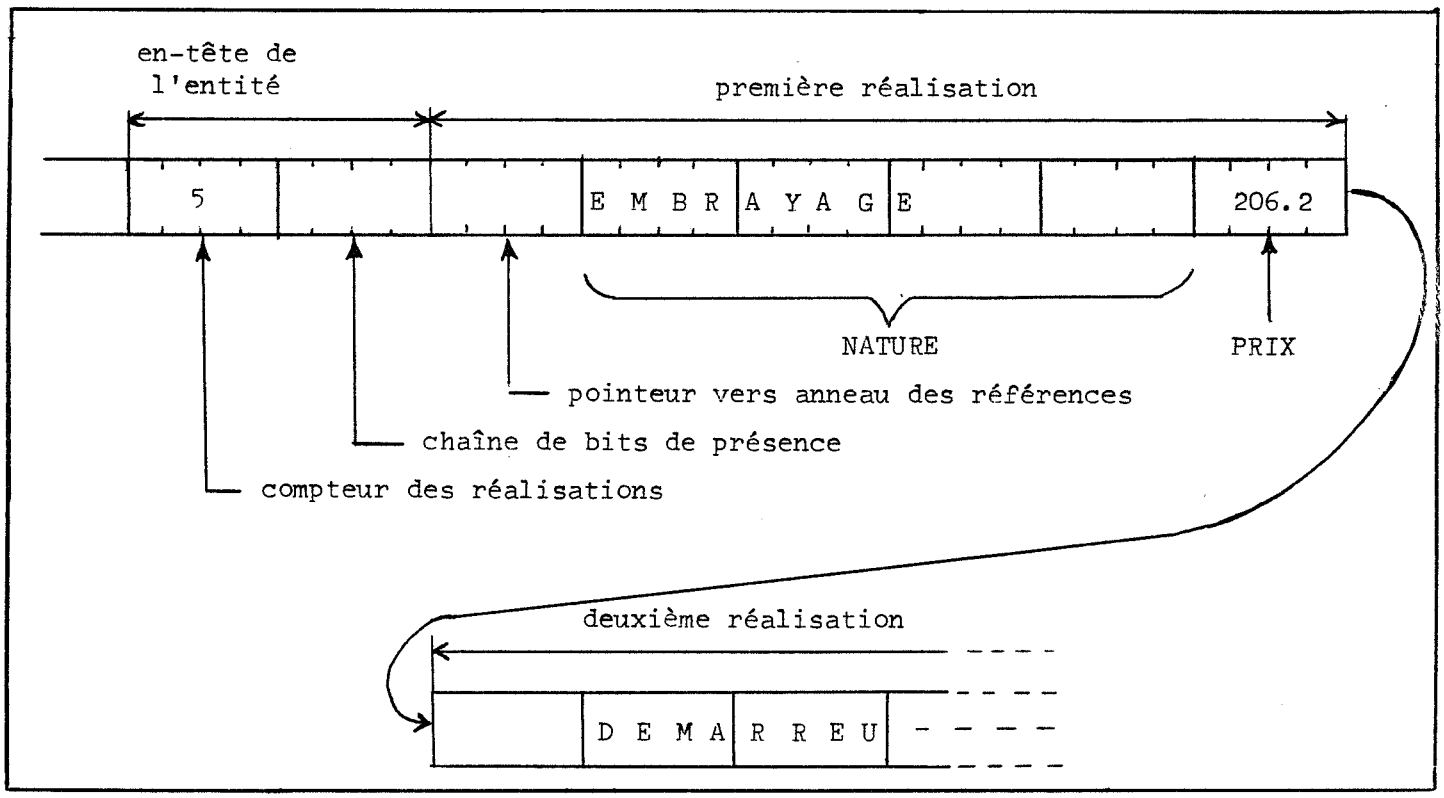


Fig.III.16 - représentation d'une entité dans le fichier-données

3.3.5. - REFERENCE :

3.3.5.1. - Chaîne des références :

Une caractéristique référence utilise 2 mots du fichier-données. Ces mots contiennent respectivement :

- un pointeur vers la réalisation référencée
- un pointeur vers une autre caractéristique référençant le même objet ou le pointeur "vide" (c'est-à-dire - 1).

exemple :

```
ENTITE 100 PERSONNE......
.........................
ENTITE 15 VOITURE  DEBUT
    PROPRIETAIRE REFERENCE UNE PERSONNE
    ...................................
```

.../...

Si la personne correspondant à la $n^{ième}$ réalisation de l'entité PERSONNE possède trois voitures, le chaînage des caractéristiques références est effectué conformément au schéma suivant (fig.III.17)

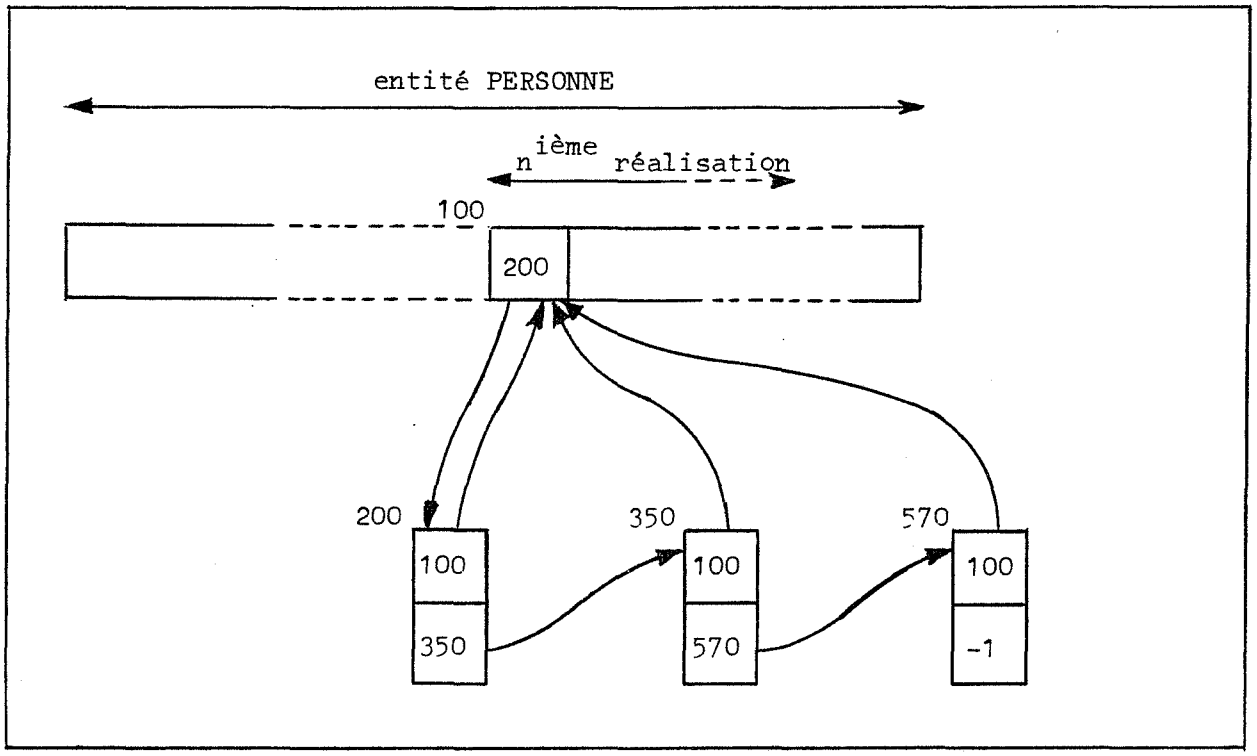


Fig.III.17 -chaînage des références

Ce chaînage permet :

* d'une part, de connaitre le propriétaire d'une voiture donnée. (remarquons que la relation est fonctionnelle et qu'une voiture ne peut avoir qu'un seul propriétaire)

* d'autre part, étant donnée une personne de retrouver toutes les voitures dont elle est propriétaire.

.../...

3.3.5.2. - Mise à jour de la chaîne des références :

La mise à jour d'une caractéristique référence consiste en l'introduction ou la suppression d'un maillon de la chaîne des références relative à une réalisation d'une entité.

L'introduction d'un nouveau maillon se fait toujours en début de chaîne. Ainsi, dans l'exemple précédent, si une nouvelle caractéristique REFERENCE a pour adresse absolue 600, la chaîne sera modifiée comme ci-dessous (les autres maillons de la chaîne restent inchangés) (fig. III.18).

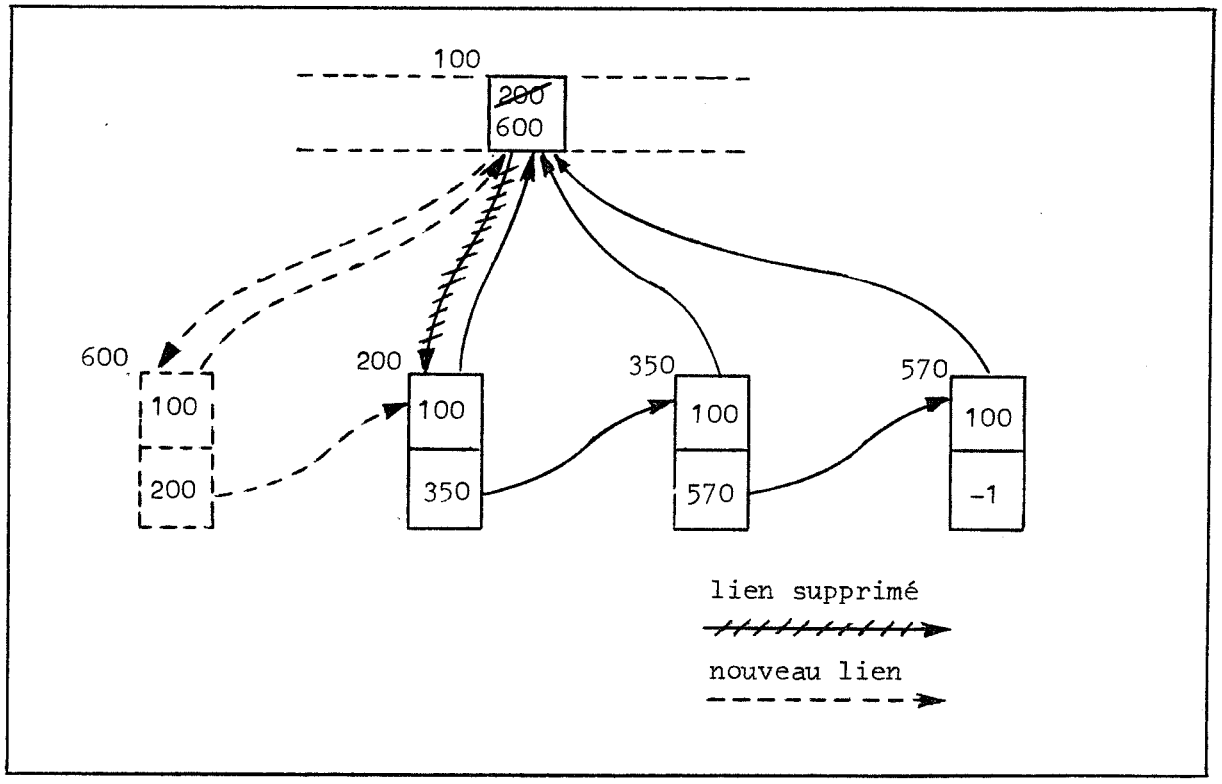


Fig. III.18 - introduction d'un maillon

La suppression du Ième maillon de la chaîne se fait simplement en remplaçant dans le (i-1)ème maillon le pointeur vers le ième par le pointeur vers le (i + 1) ème.

Pour l'exemple précédent le schéma de suppression du maillon situé à l'adresse 350 est le suivant (fig.III.19).

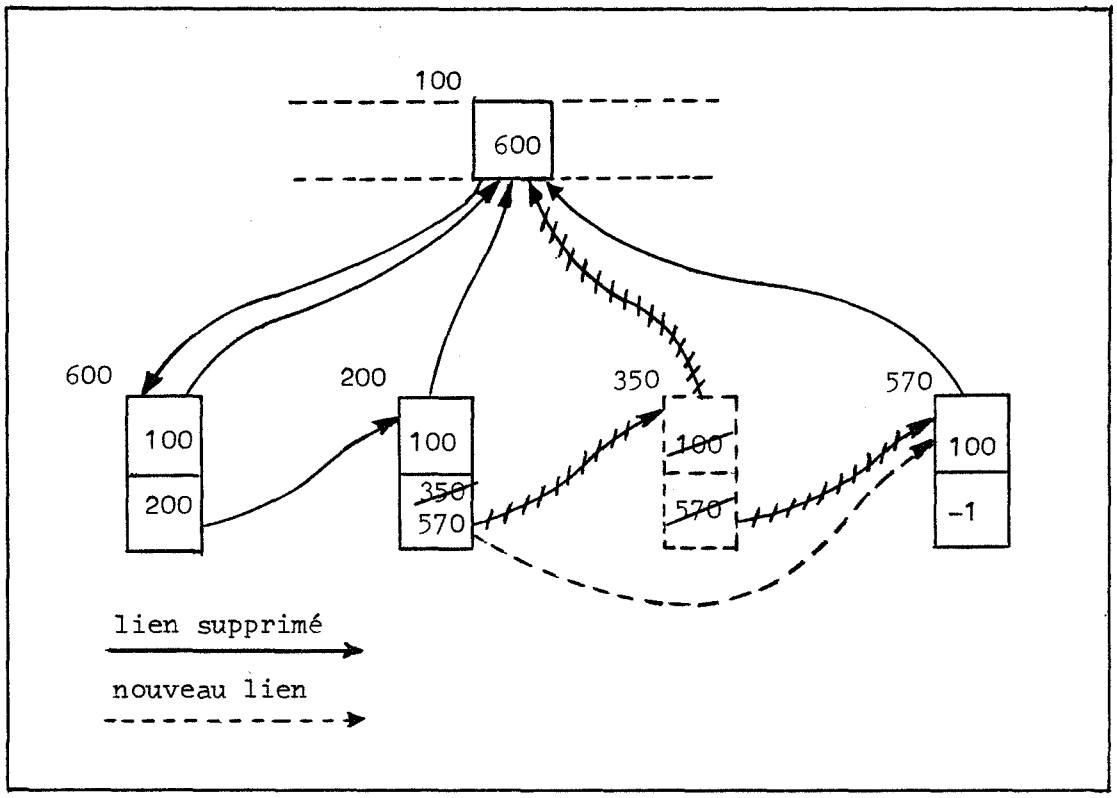


Fig.III.19 - suppression d'un maillon

.../...

3.3.6. - Inverse :

L'image de la caractéristique inverse dans le fichier-données est constituée d'un mot contenant un compteur du nombre d'objets de l'entité citée possédant la propriété inverse, et d'un nombre de mots correspondant à la taille de la chaîne de bits de l'entité citée.

exemple :

La caractéristique ENFANT définie dans 2.2.2 (fig.III.6) par

ENFANT INVERSE UNE PERSONNE

a la représentation suivante dans le fichier-donnée (l'entité PERSONNE ayant 100 réalisations) (fig.III.20).

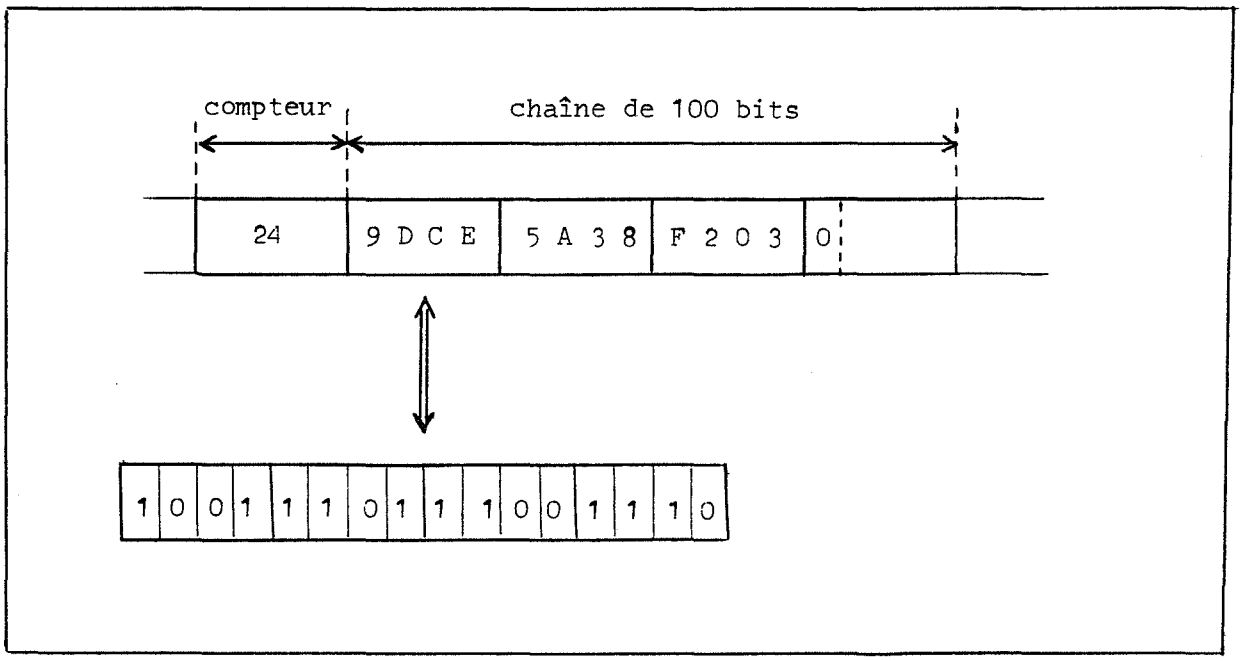


Fig.III.20 - représentation d'une caractéristique inverse dans le fichier-données

4 - IMPLEMENTATION DE LA STRUCTURE

Dans ce paragraphe nous décrivons les différentes opérations qui permettent de passer de la structure logique telle qu'elle est décrite par l'utilisateur (cf § 2) à la structure physique des informations. (cf. § 3).

4. 1. - Analyse syntaxique :

L'analyseur syntaxique décode une instruction de LDS grâce à des automates d'états finis. Une instruction de LDS peut être introduite soit à partir d'un terminal, soit par appel de sous-programme.

Cet analyseur a deux fonctions essentielles :

a) il recherche les erreurs syntaxiques. Dans le cas d'une erreur, celle-ci doit être corrigée pour que l'analyse puisse continuer (exception faite pour quelques erreurs sans conséquence pour la suite). Des messages d'erreurs sont envoyés, ils permettent de caractériser et de localiser l'erreur. Après correction de l'erreur il est nécessaire de recommencer l'analyse au début.

b) Il crée un fichier séquentiel dans lequel est rangée la structure une fois compactée et codée. (phase d'analyse lexicologique). Pour chacun des états des automates, il doit être procédé (souvent de façon exclusive) à l'analyse

* d'un mot-clé du langage
* ou d'un identificateur
* ou d'un nombre.

Les procédures mises en oeuvre dans les sous-programmes qui permettent ces différentes analyses peuvent être résumées ainsi :

4.1.1. - Analyse d'un mot-clé du langage :

Les caractères qui composent les mots-clés sont analysés deux par deux (sous programme TEST) ; parfois l'analyse est limitée aux deux premiers caractères si aucune ambiguité n'est possible. Les mots sont ensuite codés par un nombre et rangés sur le fichier séquentiel.

C'est l'analyse du mot-clé : *** qui termine l'analyse syntaxique.

4.1.2. - Analyse d'un identificateur :

L'identificateur est tronqué s'il dépasse 16 caractères. Le sous-programme IDENT recherche si le premier caractère de la chaîne est une lettre. Après analyse l'identificateur est écrit dans le fichier.

4.1.3. - Analyse d'un nombre :

Il y a lecture du nombre jusqu'au premier espace. Ensuite, le sous-programme NB vérifie que tous les caractères de la chaîne sont bien des chiffres, si oui il est écrit sur le fichier.

Lorsque la structure est déclarée syntaxiquement correcte il y a passage à l'étape suivante de l'implémentation, c'est à dire à la construction en plusieurs passes de la structure de mémorisation interne.

4. 2. - CONSTRUCTION DU PLEXE :

4.2.1. - Du résultat de l'analyse lexicologique au plexe :

Après la phase d'analyse lexicologique effectuée en même temps que l'analyse syntaxique, le premier travail consiste à créer les différents vecteurs du plexe représentant la structure en remplissant les huit premiers éléments (c'est-à-dire, IDENT, CODE, MAX, PERE, OCC) et aussi, lorsque cela est nécessaire, AD (fig. III.21).

Sachant que la structure est décrite en employant la méthode du préordre et connaissant les différentes informations ci-dessus, nous avons déjà une représentation complète de la structure en mémoire.

Le reste du travail va donc consister à partir de ces informations, à affecter des valeurs aux autres pointeurs ou fonctions composant le plexe et dont l'utilité a été précisée dans 3.2.

Trois passes à travers la structure sont alors nécessaires :

* détermination des pointeurs vers les autres éléments de la structure.
* Calcul de la taille des réalisations d'entité et des blocs.
* Calcul des adresses relatives.

.../...

DESIGN	CODE	MAX	PERE	OCC	TAILLE	FRERE	FILS	AR			AD
EXEMPLE	1	0	0	0	-1	-1	-1	-1	0	-1	-1
DATE	2	0	0	0	-1	-1	-1	-1	0	-1	-1
JOUR	7	0	15	0	-1	-1	-1	-1	0	-1	-1
MOIS	9	10	15	0	-1	-1	-1	-1	0	-1	-1
AN	7	0	15	0	-1	-1	-1	-1	0	-1	-1
PERSONNE	10	100	0	0	-1	-1	-1	-1	0	-1	-1
SEXE	14	2	75	0	-1	-1	-1	-1	0	-1	-1
MASCULIN	120	****	****	****	********	195	-1	-1	-1	-1	-1
NOM	9	10	75	1	-1	-1	-1	-1	0	-1	-1
ADRESSE	6	2	75	1	-1	-1	-1	-1	0	-1	-1
NAISSANCE	11	0	75	1	-1	-1	-1	-1	0	-1	15
SERVICE-MILITAIR	8	2	75	1	-1	-1	-1	-1	0	-1	-1
OUI	-1	****	****	****	********	-1	-1	-1	-1	-1	-1
NOM	9	10	75	2	-1	-1	-1	-1	0	-1	-1
NOM-DE-JEUNE-FIL	11	0	75	2	-1	-1	-1	-1	0	-1	120
ENFANT	5	100	0	0	-1	-1	-1	-1	0	-1	75
VOITURE	3	15	0	0	-1	-1	-1	-1	0	-1	-1
PROPRIETAIRE	4	0	240	0	-1	-1	-1	-1	0	-1	75
MARQUE	9	12	240	0	-1	-1	-1	-1	0	-1	-1
REPARATIONS	3	25	240	0	-1	-1	-1	-1	0	-1	-1
NATURE	9	16	285	0	-1	-1	-1	-1	0	-1	-1
PRIX	12	0	285	0	-1	-1	-1	-1	0	-1	-1
COULEUR	8	11	240	0	-1	-1	-1	-1	0	-1	-1
BLEU	-1	****	****	****	********	-1	****	********	****	****	-1
NOIR	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
**********	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
**********	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
***	0	-2	-2	-2	-1	-1	-1	-1	0	-1	-1

Fig.III.21 - les vecteurs du plexe après l'analyse lexicologique.

.../...

4.2.2. - Détermination des pointeurs vers les autres éléments de la structure (fig.III.22).

La fonction PERE étant connue, pour déterminer le FRERE et le FILS d'une caractéristique, il suffit de parcourir le plexe de la gauche vers la droite en faisant des comparaisons entre les valeurs de PERE.

Pour un noeud i donné, la première caractéristique rencontrée ayant même PERE aura son adresse structurelle affectée à FRERE (i), et la première caractéristique ayant pour valeur de PERE l'adresse structurelle du noeud i verra son adresse structurelle affectée à FILS (i).

La représentation adoptée pour le plexe permet d'avoir un pas d'incrémentation constant (égal à 15 mots) pour passer d'une caractéristique à une autre. Une exception est retenue cependant pour la caractéristique " liste de valeurs" (ou "liste de valeurs choix") car il faut sauter les grains dans lesquels sont rangés les termes de la liste.

DESIGN	CODE	MAX	PERE	OCC	TAILLE	FRERE	FILS	AR			AD
EXEMPLE	1	0	0	0	-1	0	15	-1	0	-1	-1
DATE	2	0	0	0	-1	75	30	-1	0	-1	-1
JOUR	7	0	15	0	1	45	0	-1	0	-1	-1
MOIS	9	10	15	0	3	60	0	-1	0	-1	-1
AN	7	0	15	0	1	0	0	-1	0	-1	-1
PERSONNE	10	100	0	0	-1	225	90	-1	0	-1	-1
SEXE	14	2	75	0	-1	120	105	-1	0	-1	-1
MASCULIN	120	****	****	****	********	195	-1	-1	-1	-1	-1
NOM	9	10	75	1	3	135	0	-1	0	-1	-1
ADRESSE	6	2	75	1	30	150	0	-1	0	-1	-1
NAISSANCE	11	0	75	1	-1	165	30	-1	0	-1	15
SERVICE-MILITAIR	8	2	75	1	1	195	180	-1	0	-1	-1
OUI	-1	****	****	****	********	-1	-1	-1	-1	-1	-1
NOM	9	10	75	2	3	210	0	-1	0	-1	-1
NOM-DE-JEUNE-FIL	11	0	75	2	-1	0	0	-1	0	-1	120
ENFANT	5	100	0	0	5	240	0	-1	0	-1	75
VOITURE	3	15	0	0	-1	0	255	-1	0	-1	-1
PROPRIETAIRE	4	0	240	0	2	270	90	-1	0	-1	75
MARQUE	9	12	240	0	3	285	0	-1	0	-1	-1
REPARATIONS	3	25	240	0	-1	330	300	-1	0	-1	-1
NATURE	9	16	285	0	4	315	0	-1	0	-1	-1
PRIX	12	0	285	0	1	0	0	-1	0	-1	-1
COULEUR	8	11	240	0	1	0	345	-1	0	-1	-1
BLEU	-1	****	****	****	********	-1	****	********	****	****	-1
NOIR	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
**********	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
**********	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
***	0	-2	-2	-2	-1	-1	-1	-1	0	-1	-1

Fig. III.22 - Le plexe après la détermination des pointeurs vers les autres éléments de la structure et de la taille des caractéristiques élémentaires.

4.2.3. - Calcul de la taille des blocs et des entités (fig.III.23).

Les tailles des caractéristiques élémentaires (MOT, TEXTE, NUMERIQUE....) sont calculées et inscrites au cours de la passe précédente. Il ne reste donc à calculer que les tailles des BLOCS et des ENTITES.
Cette opération se fait en explorant le plexe de la gauche vers la droite. Les calculs sont effectués dans l'ordre dans lequel BLOCS et ENTITES sont rencontrés ; toutefois, dans le cas des BLOCS COMPOSES et des ENTITES IMBRIQUEES ce sont les tailles des BLOCS ou des ENTITES les plus internes qui sont calculées en premier (ce qui nécessite la gestion d'une pile).

* taille du bloc :

Elle est égale à la somme des tailles de ses caractéristiques.

exemple : BLOC DATE (fig.III.23)

taille de JOUR = 1
taille de MOIS = 3 ⟹ taille de DATE = 5
taille de AN = 1

* la fonction TAILLE du grain du plexe correspondant à une entite (cf. 3.2.2.4.) a pour valeur la taille d'une réalisation de l'entité. Celle-ci est égale à la somme des tailles des caractéristiques qui la composent augmentée de 1 (pointeur vers l'anneau des références).

exemple : ENTITE REPARATIONS (fig.III.23)

pointeur REFERENCE = 1
taille NATURE = 4 ⟹ taille REPARATIONS = 6
taille PRIX = 1 (1 réalisation)

* Pour une ENTITE-CHOIX, les tailles des différentes réalisations possibles sont calculées, mais seule la plus grande est retenue.

.../...

DESIGN	CODE	MAX	PERE	OCC	TAILLE	FRERE	FILS	AR			AD
EXEMPLE	1	0	0	0	6502	0	15	-1	0	-1	-1
DATE	2	0	0	0	5	75	30	-1	0	-1	-1
JOUR	7	0	15	0	1	45	0	-1	0	-1	-1
MOIS	9	10	15	0	3	60	0	-1	0	-1	-1
AN	7	0	15	0	1	0	0	-1	0	-1	-1
PERSONNE	10	100	0	0	41	225	90	-1	0	-1	-1
SEXE	14	2	75	0	1	120	105	-1	0	-1	-1
MASCULIN	120	****	****	****	********	195	-1	-1	-1	-1	-1
NOM	9	10	75	1	3	135	0	-1	0	-1	-1
ADRESSE	6	2	75	1	30	150	0	-1	0	-1	-1
NAISSANCE	11	0	75	1	5	165	30	-1	0	-1	15
SERVICE-MILITAIR	8	2	75	1	1	195	180	-1	0	-1	-1
OUI	-1	****	****	****	********	-1	-1	-1	-1	-1	-1
NOM	9	10	75	2	3	210	0	-1	0	-1	-1
NOM-DE-JEUNE-FIL	11	0	75	2	3	0	0	-1	0	-1	120
ENFANT	5	100	0	0	5	240	0	-1	0	-1	75
VOITURE	3	15	0	0	159	0	255	-1	0	-1	-1
PROPRIETAIRE	4	0	240	0	2	270	90	-1	0	-1	75
MARQUE	9	12	240	0	3	285	0	-1	0	-1	-1
REPARATIONS	3	25	240	0	6	330	300	-1	0	-1	-1
NATURE	9	16	285	0	4	315	0	-1	0	-1	-1
PRIX	12	0	285	0	1	0	0	-1	0	-1	-1
COULEUR	8	11	240	0	1	0	345	-1	0	-1	-1
BLEU	-1	****	****	****	********	-1	****	********	****	****	-1
NOIR	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
**********	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
**********	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
***	0	-2	-2	-2	-1	-1	-1	-1	0	-1	-1

Fig. III. 23 - le plexe après le calcul de la taille des réalisations d'entité et des blocs.

.../...

exemple : ENTITE-CHOIX PERSONNE (fig. III.23)

taille d'une réalisation correspondant à l'occurence 1 :

REFERENCE		SEXE		NOM		ADRESSE		NAISSANCE		SERVICE-MILITAIRE	
1	+	1	+	3	+	30	+	5	+	1	= 41

Taille d'une réalisation correspondant à l'occurence 2 :

REFERENCE		SEXE		NOM		NOM-DE-JEUNE-FILLE		
1	+	1	+	3	+	3	=	8

la fonction TAILLE pour la caractéristique PERSONNE prend la valeur 41

* Dans le cas des entités imbriquées il est nécessaire de connaître la taille des entités internes.

exemple : le calcul de la taille d'une réalisation de l'entité VOITURE nécessite la connaissance de la taille de l'entité REPARATIONS
La taille d'une entité s'obtient en additionnant :

- la dimension de l'en-tête de l'entité (cf. 3.3.4.)
- le produit de la taille d'une réalisation par le nombre maximal de réalisations.

Ainsi, pour l'entité REPARATIONS, nous avons :

$$\text{taille de l'EN-TETE} = 1 + \left(\frac{MAX - 1}{32} + 1\right) = 2$$

taille d'une réalisation = 6
nombre de réalisations = 25
taille de l'entité = 2 + 6 x 25 = 152

.../...

La taille d'une réalisation de l'entité VOITURE se calcule donc ainsi :

pointeur REFERENCE	= 1	taille d'une réalisation = 159
taille PROPRIETAIRE	= 2	
taille MARQUE	= 3	
taille REPARATIONS	= 152	
taille COULEUR	= 1	

4.2.4 - Calcul des adresses relatives (fig.III.24)

Une dernière passe à travers la structure permet de déterminer les adresses relatives (AR) dans le fichier-données des différentes caractéristiques par rapport aux caractéristiques mères. Comme dans les étapes précédentes les grains du plexe sont analysés successivement.

dans un bloc, le premier fils a comme adresse relative 0 (exemple : JOUR) ; l'adresse relative du ième fils est égale à l'adresse relative du (i - 1) ème fils augmentée de la taille de ce dernier .

exemple :

pour MOIS : AR = 1
TAILLE = 3

pour AN : AR = 1 + 3 = 4

Dans une entité, les adresses rèlatives sont données par rapport au début des réalisations. Les calculs sont les mêmes que pour un bloc mais il faut rajouter 1 à l'AR du premier fils à cause du mot réservé pour le pointeur vers l'anneau des références.

exemple :

AR (PROPRIETAIRE) = 1 (1er fils dans l'entité VOITURE)

AR (COULEUR) = AR (REPARATIONS) + TAILLE DE L'ENTITE REPARATIONS
= 6 + 152 (cf 4.2.3.)
= 158

.../...

DESIGN	CODE	MAX	PERE	OCC	TAILLE	FRERE	FILS	AR			AD
EXEMPLE	1	0	0	0	6502	0	15	0	0	-1	-1
DATE	2	0	0	0	5	75	30	0	0	-1	-1
JOUR	7	0	15	0	1	45	0	0	0	-1	-1
MOIS	9	10	15	0	3	60	0	1	0	-1	-1
AN	7	0	15	0	1	0	0	4	0	-1	-1
PERSONNE	10	100	0	0	41	225	90	5	0	-1	-1
SEXE	14	2	75	0	1	120	105	1	0	-1	-1
MASCULIN	120	****	****	****	********	195	-1	-1	-1	-1	-1
NOM	9	10	75	1	3	135	0	2	0	-1	-1
ADRESSE	6	2	75	1	30	150	0	5	0	-1	-1
NAISSANCE	11	0	75	1	5	165	30	35	0	-1	15
SERVICE-MILITAIR	8	2	75	1	1	195	180	40	0	-1	-1
OUI	-1	****	****	****	********	-1	-1	-1	-1	-1	-1
NOM	9	10	75	2	3	210	0	2	0	-1	-1
NOM-DE-JEUNE-FIL	11	0	75	2	3	0	0	5	0	-1	120
ENFANT	5	100	0	0	5	240	0	4110	0	-1	75
VOITURE	3	15	0	0	159	0	255	4115	0	-1	-1
PROPRIETAIRE	4	0	240	0	2	270	90	1	0	-1	75
MARQUE	9	12	240	0	3	285	0	3	0	-1	-1
REPARATIONS	3	25	240	0	6	330	300	6	0	-1	-1
NATURE	9	16	285	0	4	315	0	1	0	-1	-1
PRIX	12	0	285	0	1	0	0	5	0	-1	-1
COULEUR	8	11	240	0	1	0	345	158	0	-1	-1
BLEU	-1	****	****	****	********	-1	****	********	****	****	-1
NOIR	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
**********	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
**********	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
***	0	-2	-2	-2	-1	-1	-1	-1	0	-1	-1

Fig.III.24 - le plexe sous sa forme finale (après le calcul des adresses relatives)

.../...

5 - CONCLUSION

L'objet de cette partie a été de décrire le modèle de structuration des informations adopté pour le projet SOMINE. Nous avons étudié successivement :

* le langage de définition de structure qui permet au concepteur d'une base de données SOMINE de décrire la structure logique de ses informations .

* la représentation de cette structure dans le fichier-structure et l'organisation du fichier-données,

* les différentes phases de travail effectuées par le programme d'implémentation d'une structure.

La structure SOMINE, de type réseau, permet une mémorisation simple d'informations variées et autorise des accès diversifiés. Sa représentation sous forme de plexe dans le fichier-structure rend possible d'éventuelles modifications du logiciel :

* par l'introduction de nouveaux liens, pointeurs ou compteurs (les éléments des grains du plexe ne sont pas tous utilisés). Cela nous servira dans l'étude des modifications de structures que nous envisageons dans une prochaine version de SOMINE.

* par la définition de nouvelles caractéristiques. A la suite de l'application réalisée en C.A.O. (cf /8/ partie IV), par exemple, nous avons ajouté à l'éventail des caractéristiques utilisables,

la caractéristique PROGRAMME. Le vecteur définissant une telle caractéristique est le suivant :

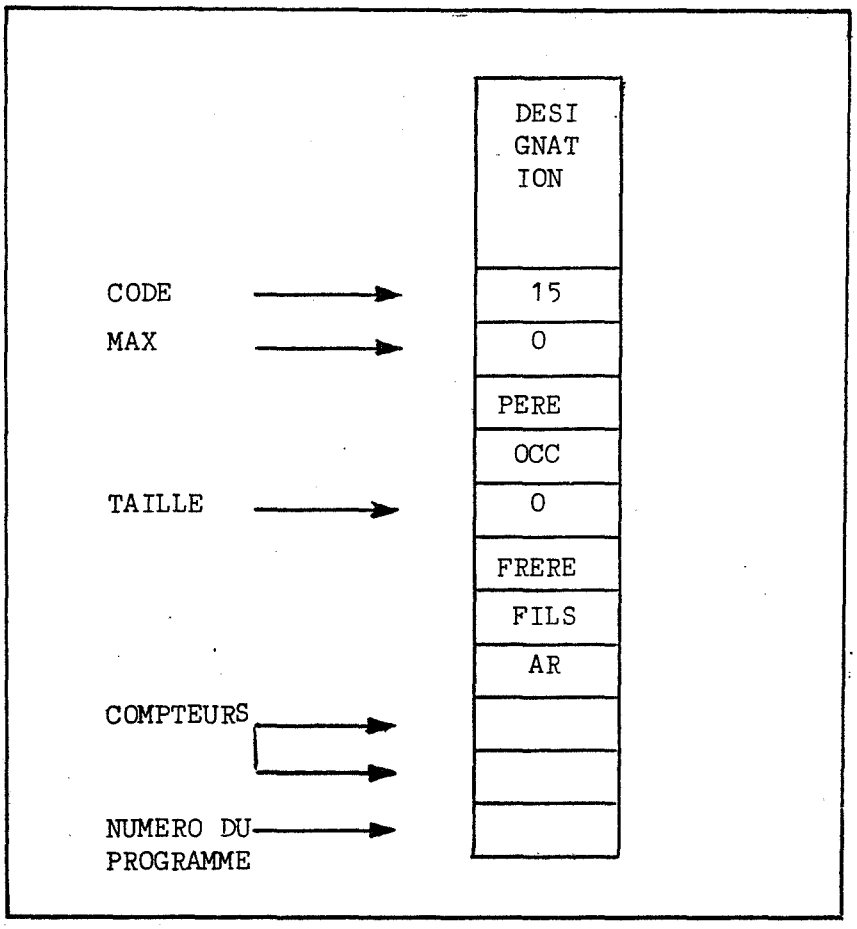


Fig. III. 25

Le code qui lui a été attribué est 15.
Le dernier mot de la sous-page contient un numéro qui sera celui du ou des programmes dont on désire l'activation au moment de l'exécution de la requête. Les autres mots du vecteur de définition (pointeurs, OCC, compteurs) ont la même destination que pour les autres caractéristiques.

La vitesse d'implémentation, de l'ordre de la minute pour des structures importantes, nous semble correcte d'autant qu'il s'agit d'une opération qui ne doit intervenir qu'une seule fois au cours de la vie d'une base de données.

.../...

Nous pensons qu'il suffit de peu de temps à un utilisateur pour apprendre le L.D.S. et utiliser SOMINE ; par contre, la phase d'analyse préalable est plus délicate et pose de nombreux problèmes, c'est cet aspect de la conception d'une base que nous allons aborder dans la partie suivante.

6 - BIBLIOGRAPHIE

/1/ R. MAHL
"Algorithme et structures de données "
Ecole Nationale Supérieure des Mines de St Etienne - 1973

/2/ F.R.A. HOPGOOD
"Techniques de compilation"
Dunod - 1970

/3/ J.R. ABRIAL
"Projet SOCRATE - Spécifications générales "
Université de Grenoble - Août 1970

/4/ T.D.M.S.
"Time shared data management system . A new approach to data management"
SDC, Santa Monica SP.2747. - février 1967

/5/ C. PAIR
" Structures de données et algorithmes fondamentaux "
Ecole Nationale Supérieure des Mines et des Industries Métallurgiques de Nancy - 1974

/6/ C. DELOBEL
" Contributions théoriques à la conception et à l'évaluation d'un système d'informations appliqué à la gestion ".
THèse d'Etat . Université de grenoble - octobre 1973

.../...

/7/ R. MAHL

"Organigrammes - FORTRAN "

Ecole Nationale Supérieure des Mines de St.Etienne - juin 1973

/8/ C. SAYETTAT

" Contribution à la conception, la réalisation et l'utilisation du système de gestion de bases de données SOMINE : les accès aux bases - aide à la conception assistée par ordinateur "

Thèse de Doctorat de Spécialité . Ecole Nationale Supérieure des Mines de St. Etienne - janvier 1976

/9/ M. GAILLARD

"Contribution à la conception,la réalisation et l'utilisation du système de gestion de bases de données SOMINE : gestion des mémoires - enseignement assisté par ordinateur ".

Thèse de Doctorat de Spécialité . Ecole Nationale Supérieure des Mines de St.Etienne - janvier 1976

/10/ F. BODART, C.DEHENEFFE, J.L. HAINAUT, H. HENNEBERT, B. LE CHARLIER, W. PAULUS

" Système de conception et d'exploitation d'une base de données "

Institut d'informatique. Facultés universitaires Notre Dame de la Paix.Namur - décembre 1974

/11/ G. LOUIS-GAVET

"Etude de synthèse sur la sémantique des bases de données "

I.U.T. - Université de Lyon 1

/12/ J.R. ABRIAL

"Data semantics "

Université de Grenoble - novembre 1973

.../...

/13/ I.M.S. (Information Management System)
"I.M.S. / 360 Applications Description Manual " IBM
White Plains, N.Y. , GH.20.0765

/14/ ADABAS
"Adascript User's Manual ", Software AG, Reston , Va., 1974

/15/ I.D.S. (Integrated Data Store)
"Application manual"
General Electric Corporation Phoenix, Arizona - juillet 1969

/16/ CODASYL
"Data Base Task Group Report"
A.C.M., New York - 1971

/17/ E.F. CODD
" A relational model of data for large shared data banks "
C.A.C.M., vol 13, n° 6 - juin 1970

/18/ C. DELOBEL
" Les systèmes de bases de données "
Université de Grenoble - juin 1975

/19/ C. DELOBEL
"Le concept de base de données "
Journées d'étude sur les systèmes généraux de bases de données
Université des Sciences et Techniques du Languedoc - 15 et 16 mai 1975

PARTIE IV

DES INFORMATIONS A UNE STRUCTURE SOMINE

"Toute acquisition véritable de connaissance repose en effet sur une étude des relations une recherche de structure"

KORSYBKI - Introduction à la sémantique générale

SOMMAIRE

1 - INTRODUCTION 5

2 - QUELQUES REMARQUES SUR LES STRUCTURES SOMINE UTILISEES DANS LES PREMIERES APPLICATIONS DU PROJET 7

2.1. - Un exemple de structure pour un problème d'aide à la C.A.O. 7

2.2. - Une structure SOMINE dans une application en E.A.O. 9

2.2.1. - Le problème posé 9

2.2.2. - La structure adoptée 11

2.3. - Un exemple de structure SOMINE en gestion 14

2.4. - Conclusion 14

3 - INFORMATIONS ET STRUCTURES 17

3.1. - Les étapes de la construction d'une base de données 17

3.2. - Trois niveaux d'études possibles 17

3.3. - Les relations 19

3.3.1. - Champ élémentaire 20

3.3.2. - Les relations $\overleftarrow{R}$ et $\overrightarrow{R}$ 20

3.3.2.1. relations sémantiques 20

3.3.2.2. relations logiques 20

3.3.2.3. exemple 21

3.4. - Choix d'un modèle de chemins d'accès 21

3.5. - Un exemple 22

.../...

3.5.1. - Définition des relations 22

3.5.1.1. - les champs 22

3.5.1.2. - les relations 24

3.5.2. - Les questions posées 27

3.5.3. - Matrice " questions-relations" 27

3.5.4. - Vers la structure logique 32

3.5.5. - Une étude partielle 38

3.6. - Nécessité d'une expérimentation 38

3.7. - Conclusion 38

4 - MESURES 43

4.1. - Pourquoi mesurer ? 43

4.2. - Quoi mesurer ? 43

4.3. - Comment mesurer ? 44

4.4. - Nos options 45

4.5. - Pour une meilleure structuration des informations 45

4.5.1. - Détermination des coûts d'accès 45

4.5.1.1. - Définitions et rappels 46

4.5.1.2. - Exemples 49

4.5.1.3. - Calcul de l'adresse structurelle 50

4.5.1.4. - Calcul de l'adresse absolue 52

4.5.1.5. - Action provoquée par la requête 54

4.5.1.6. - Remarques 54

4.5.1.7. - Conclusion 56

.../...

4.5.2. - Comptage des accès aux informations 56

4.6. - Premières mesures effectuées sur le prototype 57

4.6.1. - Analyse syntaxique et requête 57

4.6.2. - Les types de caractéristiques 58

4.6.3. - Structure large - structure haute 58

4.6.4. - Référence 59

4.6.5. - Devant l'écran 59

4.7. - Conclusion 60

5 - CONCLUSION 63

6 - BIBLIOGRAPHIE 65

ANNEXE - Une structure utilisée en gestion 67

1 - INTRODUCTION

Etudiant des applications de SOMINE dans diverses directions (E.A.O., C.A.O., gestion), nous nous sommes vite rendus compte qu'il existait une phase de travail très importante : celle qui consiste à bâtir, en tenant compte de la nature des informations, de leur propriétés, de l'usage que veut en faire l'utilisateur , la structure logique de la base. Lorsque l'utilisateur désire effectuer des requêtes très variées, le choix d'une bonne organisation est complexe.

En même temps qu'une notice technique /13/, un complément utile à fournir à un utilisateur de SOMINE serait un court rapport dans lequel on pourrait lui indiquer le moyen de construire la structure logique optimale d'informations qui lui permette de répondre aux questions qu'il se pose.

Malheureusement, les recherches entreprises n'ont pas permis, pour l'instant, de donner une méthode automatique pour passer d'un ensemble d'informations et de questions à une organisation structurelle du type de celle que nous présentons avec SOMINE.

Nous envisagerons seulement, à propos d'un exemple, de montrer la complexité du problème à résoudre et de développer quelques éléments et pistes de recherche . Nous espérons, ainsi, inciter l'utilisateur de SOMINE à une réflexion approfondie avant la définition d'une structure, ce qui lui permettra, peut être, d'éviter de commettre certaines erreurs.

.../...

2 - QUELQUES REMARQUES SUR LES STRUCTURES SOMINE UTILISEES DANS LES PREMIERES APPLICATIONS DU PROJET

Nous avons cherché à utiliser la version actuellement opérationnelle de SOMINE dans des directions différentes

* d'une part pour tester techniquement le système,

* d'autre part pour vérifier les hypothèses avancées quand à sa capacité d'apporter une aide importante dans des domaines aussi divers que l'enseignement assisté par ordinateur, la conception assistée par ordinateur de projets scientifiques ou techniques ou la gestion.

Nous allons essayer d'évoquer brièvement quels sont les problèmes qui se sont posés lors de l'étude de ces applications dans la phase qui a consisté à faire coïncider l'ensemble des informations et de leurs liens avec un modèle de structure formelle susceptible d'être défini par SOMINE.

2.1. - UN EXEMPLE DE STRUCTURE POUR UN PROBLEME D'AIDE A LA CONCEPTION ASSISTEE PAR ORDINATEUR :

Dans cette étude qui a été plus spécialement développée par C. SAYETTAT /8/, l'objectif était de montrer l'utilité d'une base de données telle que SOMINE dans un processus de conception assistée par ordinateur. Nous avons donc cherché un problème dont la solution puisse servir d'exemple ou inspirer d'autres applications dans les domaines scientifiques ou techniques. Le problème retenu a été l'étude des caractéristiques mécaniques des aciers de construction dans le but de concevoir des structures métalliques ayant une tenue en service convenable.

.../...

Après avoir recherché la méthode de résolution de ce problème il a été nécessaire de rassembler toutes les données pouvant intervenir dans la résolution, de déterminer leur nombre, leur nature et de mettre en évidence les liens entre ces données.

Entrant dans le cadre d'une application de SOMINE, cette dernière phase d'étude a été menée avec la perspective d'organiser les données en structure de type hiérarchique ou réseau. Un premier arbre structurel a été conçu ainsi d'une manière très empirique. Cet arbre s'est trouvé ensuite remanié au fur et à mesure que le problème était précisé. Enfin c'est l'énoncé des requêtes nécessaires à l'interrogation et à la mise à jour de la base qui nous a permis d'arriver à la structure actuellement utilisée pour gérer les données employées dans la résolution du problème. (figure IV. 1)

Prenant un peu de recul, nous sommes amenés à nous poser quelques questions concernant cette démarche :

* la structure proposée est-elle la meilleure possible pour la résolution du problème donné ? rend-elle compte de tous les liens entre les données ? est-elle la mieux adaptée aux types d'accès envisagés ? autrement dit a-t-on construit un bon modèle ?

* Pouvait-on obtenir cette structure (ou une plus performante) par une autre méthode ? Ne faut-il pas analyser plus à fond la matière (données, liens, accès éventuels .....) et proposer des critères permettant une organisation basée autrement que sur une construction empirique rectifiée par l'expérience ?

.../...

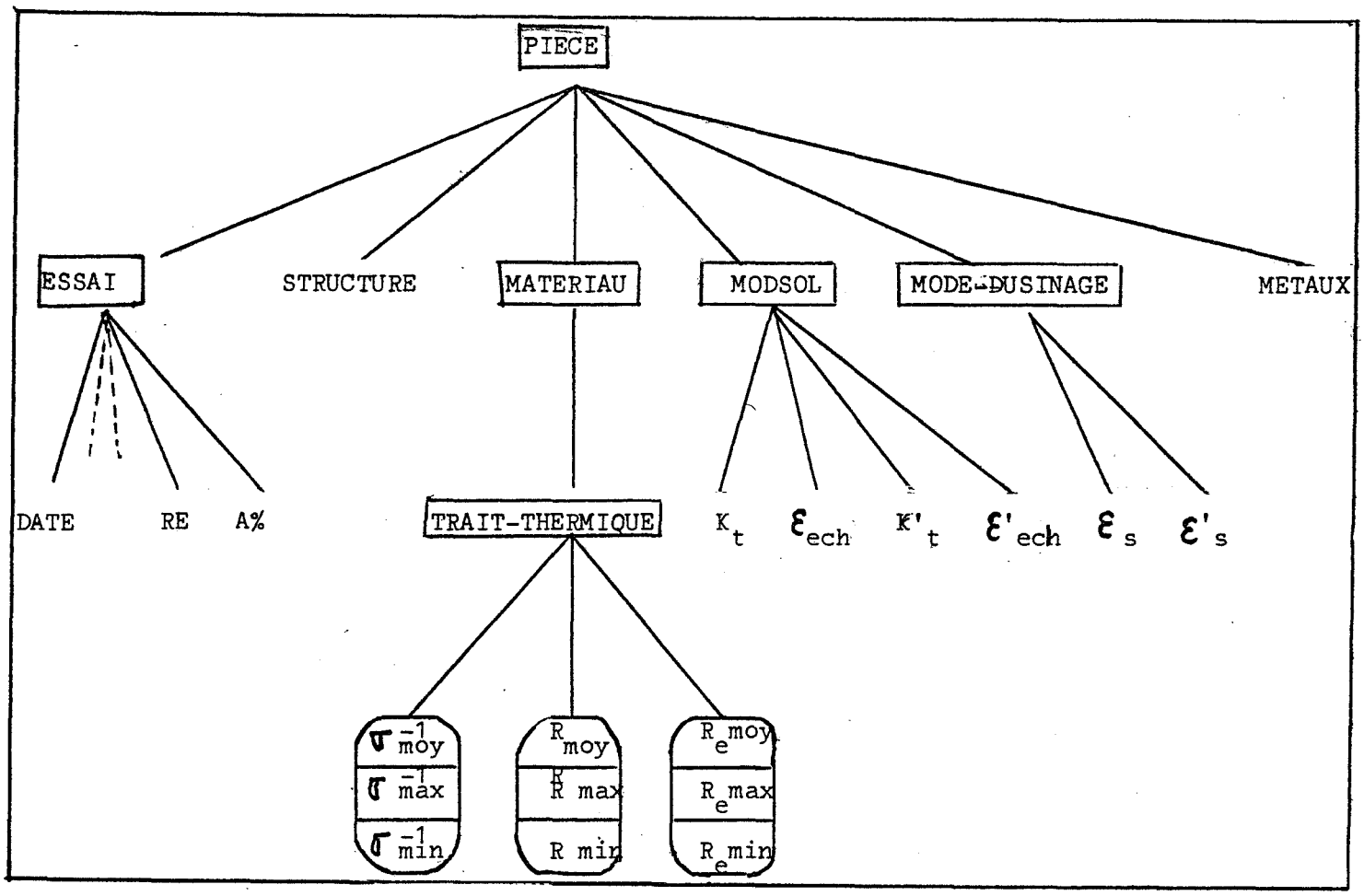


Fig.IV.1 - Une partie de la structure de l'exemple de C.A.O.

2. 2. - UNE STRUCTURE SOMINE UTILISEE DANS UNE APPLICATION EN ENSEIGNEMENT ASSISTE PAR ORDINATEUR :

2.2.1. - Le problème posé :

Dans une application de SOMINE à l'enseignement assisté par ordinateur décrite en détail par M. GAILLARD /9/, les problèmes posés par la structuration des informations ont été de deux ordres.
Il a fallu en effet distinguer :

* l'organisation de la matière à enseigner
* la méthode de présentation de cette matière (choix d'une stratégie pédagogique).

.../...

a) Organisation de la matière :

Le premier travail à effectuer lorsque l'on aborde un problème d'enseignement est de faire une étude approfondie de la matière à enseigner en vue de l'apprentissage. Cette étude doit déboucher sur une décomposition de la matière en éléments (items) et un schéma d'organisation de ces éléments. Les phases principales de ce travail ont été décrites notamment par Régnier et Montmollin /6/ :

* dresser la liste des éléments à enseigner (concepts, règles)

* rechercher entre ces éléments des relations

* organiser ces éléments en tenant compte de leurs relations et de l'expérience de l'équipe enseignante.

b) Présentation de la matière :

L'analyse précédente conduit à une forme d'organisation qui n'est pas à confondre avec les structures à adopter dans le cadre d'une stratégie pédagogique bien définie et de l'utilisation de l'outil SOMINE. Il est en effet nécessaire dans ce cas de tenir compte non seulement des éléments et de leurs relations, mais aussi,

- des méthodes possibles d'exploration de ces éléments ("complète et détaillée" ou "partielle et synthétique") qui peuvent conduire à des accès distincts ("textes" ou "résumés").

- des modes possibles d'apprentissage :

Apprentissage élément par élément, comparaison entre éléments, recherche de lois,.....

Ces différents aspects, envisagés en détail par GAILLARD /9/, permettent de déduire un certain nombre de critères pour classer les éléments d'un domaine d'enseignement, et ces critères conduisent à la définition des fonctions d'enseignement qui vont intervenir dans l'élaboration de la structure formelle qui nous utiliserons dans cet

exemple d' E.A.O.

exemples de fonctions d'enseignement :

fonction " sommaire" : elle donne la liste des éléments

fonction " recherche de similitude " : elle met en évidence les points communs entre deux éléments quelconques du domaine d'enseignement.

2.2.2. - La structure adoptée :

Une structure arborescente peut permettre une bonne représentation de la matière à enseigner. Elle fait apparaître aisément des relations d'ordre (nécessaires notamment pour l'apprentissage élément par élément), ou des relations d'appartenance ..... et peut servir de support pour la mise en place des relations évoquées au paragraphe précédent.

Toutefois, une arborescence d'analyse de la matière à enseigner peut comporter un nombre très grand de niveaux.(fig.IV.2). L'utiliser telle qu'elle est et la traduire à l'aide du L.D.S. conduirait à des requêtes complexes comme celle-ci :

I TEXTE DU A DU C DU F DU H

ce qui ne manquerait pas d'entraîner une lourdeur de fonctionnement :

requêtes longues à formuler, temps de réponse moins bons que pour des requêtes plus brèves.

.../...

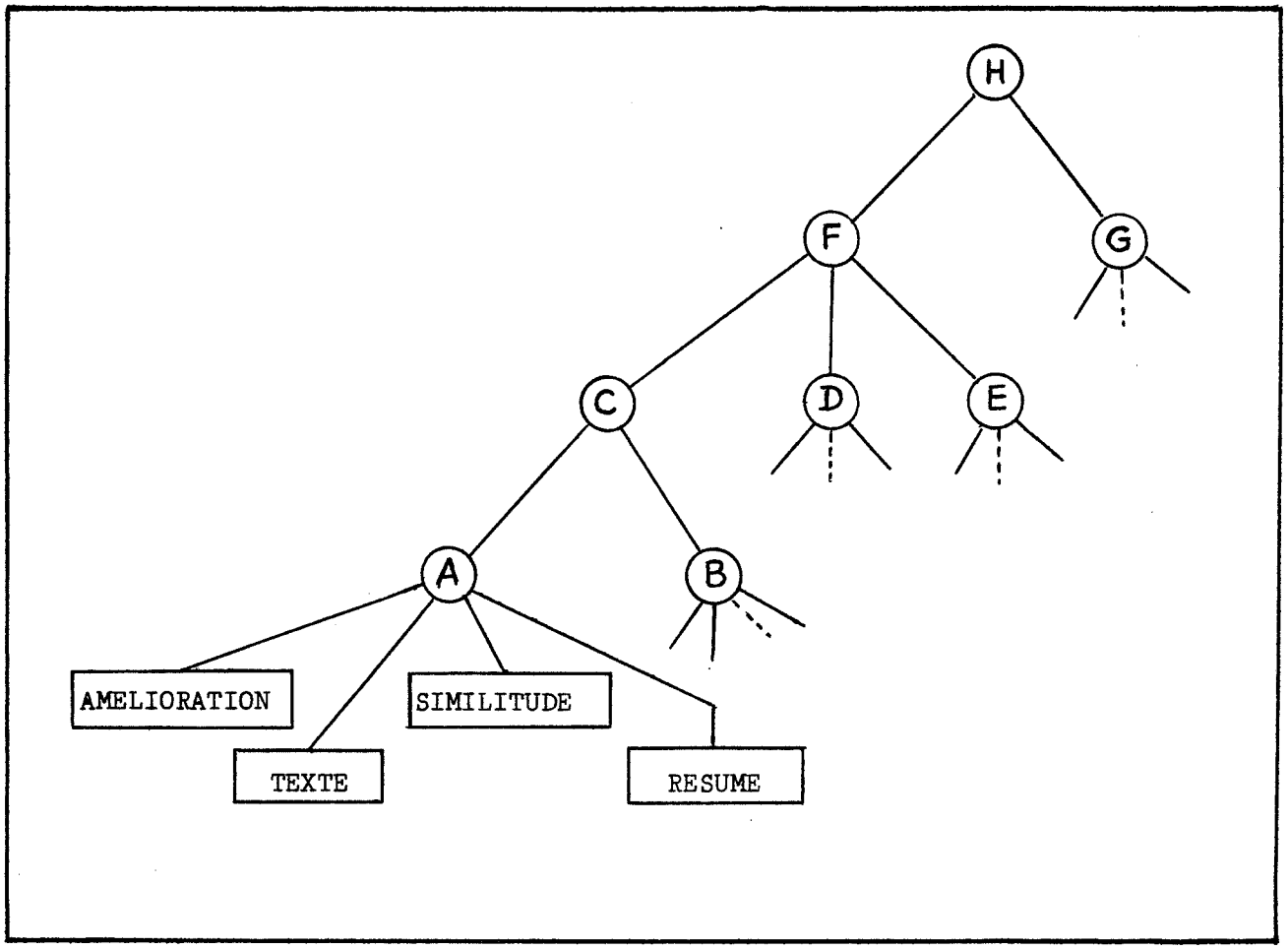


Fig.IV.2 - Arborescence d'analyse de la matière

Il nous est ainsi apparu rapidement qu'il fallait tenir compte des critères importants suivants dans le cadre d'un travail en mode conversationnel :

* les ordres de mise à jour et surtout d'interrogation doivent être simples et condensés,

* les temps de réponse doivent être les plus courts possibles.

A partir de ces considérations et de la volonté de donner à l'équipe enseignante un outil souple, adaptable à chaque domaine d'enseignement, nous avons adopté une structure plate (fig. IV.3.).

.../...

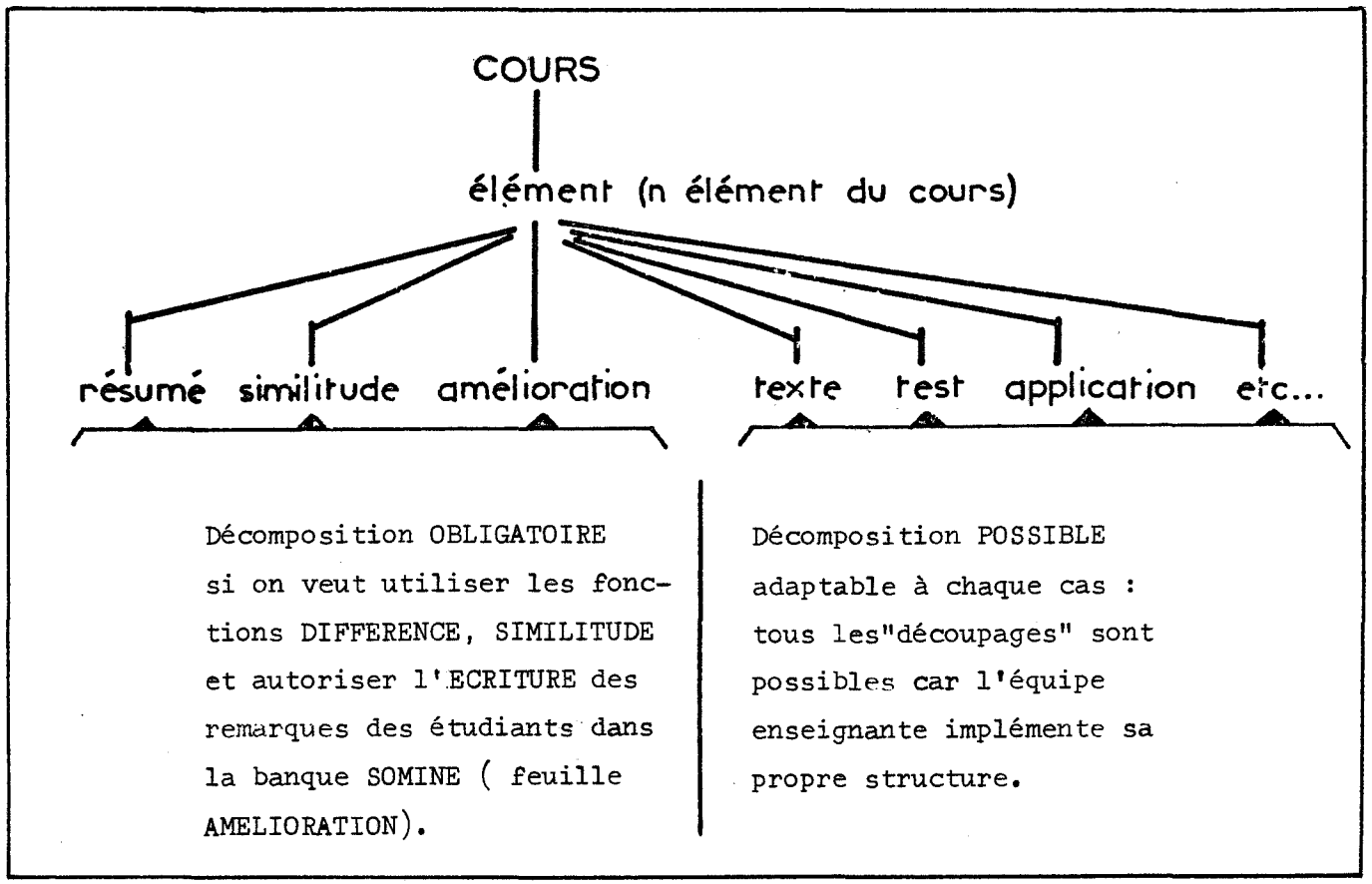


Fig.IV.3 - Structure formelle d'enseignement

Nous voyons donc sur cet exemple comment la seule considération des informations et de leurs liens pourrait conduire à des erreurs. Il faut savoir reconnaitre, cerner, définir des critères permettant de créer la meilleure structure possible en réponse à des objectifs précis.

2. 3. - UN EXEMPLE DE STRUCTURE SOMINE EN GESTION (cf. annexe partie IV)

De nombreuses applications de S.G.B.D. ont été réalisées dans la gestion des entreprises et nous ne pensons pas que SOMINE soit d'un apport original dans ce domaine. Il est à noter cependant que le système SOMINE, transportable,peu encombrant,est simple à utiliser et qu'il permet de rédiger des programmes d'application en langage évolué et en particulier en COBOL. Il peut donc répondre à certains besoins notamment dans la gestion de moyennes entreprises. Nous avons donc été conduits à nous intéresser à une application concernant la gestion d'une entreprise de la région stéphanoise.

Comme pour les exemples précédents, dans une première phase nous avons :

* rassemblé des informations
* étudié les relations possibles entre ces informations .

Nous avons également consulté les utilisateurs pour connaître les types de requêtes qu'ils désiraient effectuer sur la base de données.

Avec ces renseignements, nous avons construit une structure, puis une autre qui nous paraissait être plus satisfaisante que la précédente, puis une autre..... Le fichier devait-il avoir comme sommet l'entité CLIENT ou l'entité PRODUIT ? Devait-on classer les usines par SECTEUR GEOGRAPHIQUE ou par PRODUIT UTILISE ?..... Nous avons choisi finalement d'une manière assez subjective, et nous pouvons nous poser les mêmes questions qu'à la fin de 2.1.

2. 4. - CONCLUSION :

L'étude de ces exemples nous a montré combien il est important de pouvoir guider l'utilisateur dans la struturation des informations en lui fournissant une méthode d'approche pour l'obtention d'une structuration logique correcte de ses informations et des critères d'appréciation. Suivant la manière de procéder on peut faire de SOMINE

.../...

un outil lourd et inefficace ou bien un outil simple, maniable et efficace.

Dans le chapitre suivant, nous allons donc essayer de dégager une méthodologie pour définir une structure logique à partir des relations qui lient chaque type de données. Nous verrons ensuite comment cette structure peut être éventuellement améliorée par une discussion sur le coût des requêtes. Enfin, un espionnage systématique du fonctionnement de la base à l'aide de compteurs réels doit permettre d'affiner encore la structure.

3 - INFORMATIONS ET STRUCTURES

3. 1. - LES ETAPES DE LA CONSTRUCTION D'UNE BASE DE DONNEES :

Le processus de conception d'un système d'informations peut être divisé en plusieurs phases organisées suivant le schéma de la figure IV. 4 avec à la périphérie l'application et son environnement et au centre la mémoire (cf. DELOBEL /1/). Le problème posé est : Comment peut-on passer de l'application à la "mémoire " ?

Dans SOMINE seules les étapes qui permettent le passage de l'organisation logique des données à l'organisation physique des données puis à la mémoire sont automatisées. Il importe donc d'essayer de voir comment obtenir, à partir des informations telles qu'elles sont connues au niveau de l'application, une organisation logique des données qui soit conforme à celle que permet de traiter SOMINE et qui reflète le plus fidèlement possible l'application.

3. 2. - TROIS NIVEAUX D'ETUDE POSSIBLES :

Dans l'étude d'un système de gestion de bases de données nous pouvons distinguer plusieurs niveaux possibles suivant que nous nous intéressons :

(a) aux informations et aux relations qui existent entre elles,

(b) aux moyens d'accès aux informations (cheminement dans la structure),

.../...

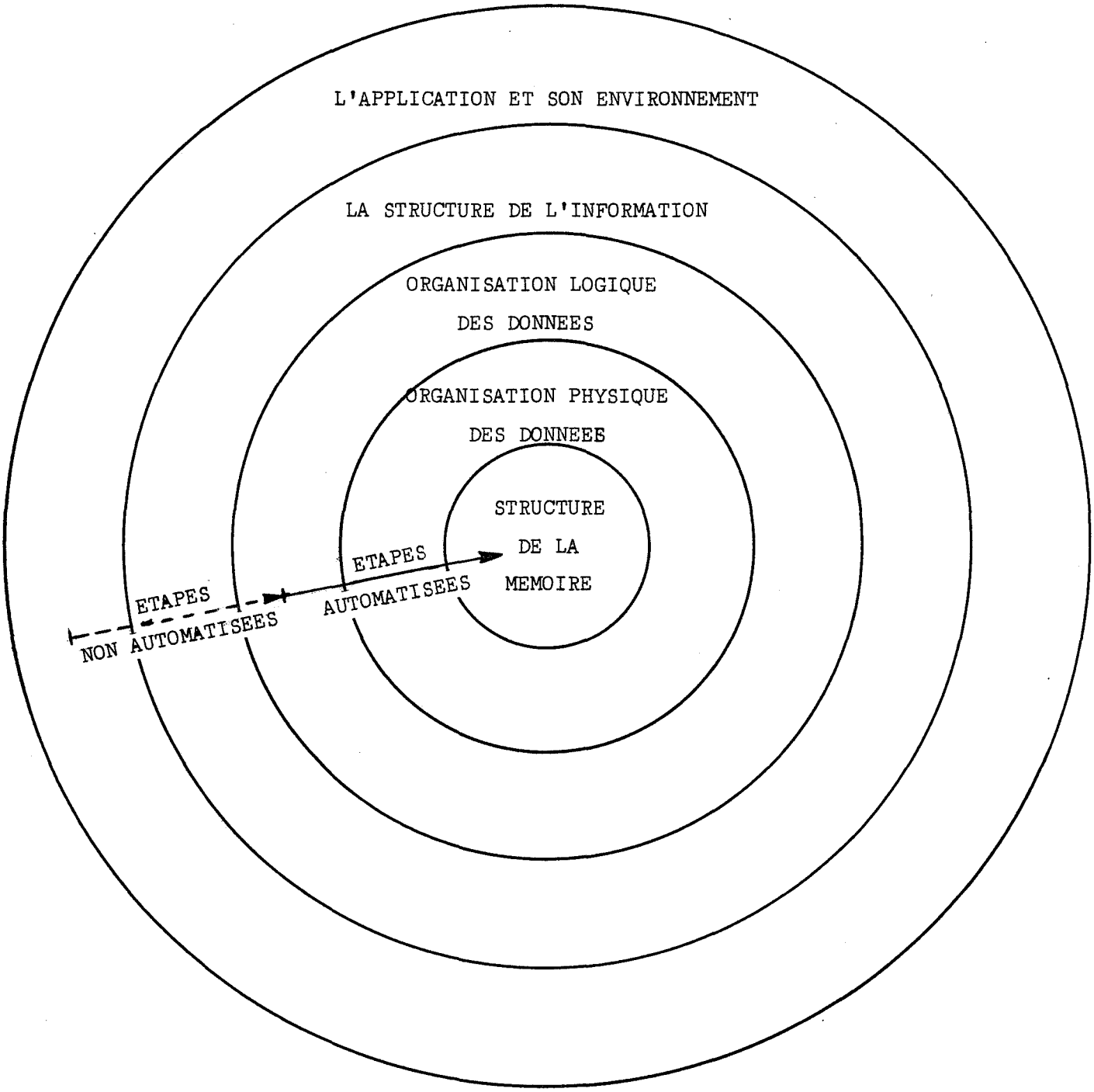


Fig.IV.4 - Les différentes étapes dans la conception d'un système d'information (d'après une figure extraite de /1/).

(c) à la configuration des données sur le support physique.

Le fait de privilégier la réflexion à l'un ou l'autre de ces niveaux conduit à la construction de bases de données qui peuvent avoir des caractères très différents.

La plupart du temps, notamment dans l'emploi de modèles de types hiérarchiques ou dérivés de types hiérarchiques, l'utilisateur insiste sur l'étude des deux derniers niveaux qu'il perçoit comme importants et n'aborde que très sommairement (ou néglige) l'étude des relations.

Dans /2/ (SENKO....), les trois niveaux sont mis en évidence ; les paramètres de chacun d'eux sont décrits mais pas la manière de passer de l'un à l'autre.

Une tendance de recherche actuelle est de construire des modèles au niveau (a) c'est-à-dire, essentiellement, de définir et d'étudier des relations sans se préoccuper des accès. Mais, comment, à partir de ces modèles, construire des modèles de chemins d'accès ? Le problème reste encore pratiquement non résolu.

3. 3. - LES RELATIONS

Nous allons essayer d'étudier le problème des relations en évitant toute formulation mathématique compliquée. Le lecteur intéressé par des études plus théoriques pourra se reférer à des publications comme /1/, /2/, /3/, /5/, /7/.

Nous définirons d'abord des relations binaires $\overline{R}$ et $\overrightarrow{R}$ dans l'ensemble des champs élémentaires puis (fin de 3.5.1.) des relations binaires Ri, à partir du produit cartésien de deux champs.

.../...

3.3.1. - Champ élémentaire :

" Nous appellerons champ élémentaire X une caractéristique d'un fait ou d'un objet suivant lequel on désire enregistrer des informations. Un champ élémentaire est un ensemble de valeurs appelées des données,". (cf. DELOBEL /1/).

3.3.2. - Les relations $\overleftrightarrow{R}$ et $\overrightarrow{R}$

Dans l'ensemble des champs élémentaires nous distinguerons deux types de relations :

3.3.2.1. - Les relations que nous qualifions de sémantiques :

Soient A et B deux champs élémentaires. Dire que A et B vérifient la relation sémantique $\overleftrightarrow{R}$ indique seulement que l'association de ces champs a un sens du point de vue de l'application considérée :

notation : $A \overset{R}{\text{———}} B$ ou $A \overleftrightarrow{R} B$

Une telle relation est symétrique :

$$\forall A,B \quad ; A \overleftrightarrow{R} B \Longrightarrow B \overleftrightarrow{R} A$$

3.3.2.2. - Les relations définies à un niveau logique :

Dire que A et B pris dans cet ordre vérifient la relation logique $\overrightarrow{R}$ signifie que B est un champ descendant de A

notation : $A \xrightarrow{R} B$ ou $A \overrightarrow{R} B$

A noter que :

$$\left[A \overrightarrow{R} B\right] \Longrightarrow \left[\text{non} \quad B \overrightarrow{R} A\right]$$

La définition d'une telle relation fige une structure hiérarchique .

3.3.2.3. - Exemple :

Si A et B désignent les champs PRODUIT et CLIENT,

$A \overset{R}{\text{———}} B$ signifie que les informations PRODUIT et CLIENT ont un lien entre elles (et devront donc être reliées dans la structure à construire).

$A \overset{R}{\longrightarrow} B$ signifie que le champ PRODUIT est d'un niveau hiérarchique plus élevé que CLIENT c'est-à-dire que l'accès à CLIENT passe obligatoirement par PRODUIT dans le cas d'une structure SOMINE . (avec une citation du type "client de produit ").

3. 4. - CHOIX D'UN MODELE DE CHEMINS D'ACCES :

Cela nous montre que, si nous raisonnons en définissant dès le départ des relations au niveau logique, nous sommes déjà figés dans une structure alors que, si nous nous plaçons au niveau des relations sémantiques, nous obtenons une représentation idéale avec tous les degrés de liberté possibles.

Il faut bien avoir conscience cependant que la construction d'un modèle doit conduire en fin de compte à la définition de chemins d'accès et qu'il faudra toujours choisir entre

$[A \longrightarrow B]$ et $[B \longrightarrow A]$

("CLIENT DE PRODUIT" et " PRODUIT DE CLIENT ")

même si un artifice permet dans certains cas de rendre les deux compatibles ; une structure hiérarchique ne rend en effet pas compte des relations symétriques.

.../...

Toute la difficulté de l'optimalisation de la recherche d'un modèle de chemin d'accès provient du fait que les critères de choix ou d'évaluation sont complexes, multiples, pas toujours bien définis (coût d'un accès, fréquence des accès aux informations suivant les modes,.....).

Pour une application nous proposons le schéma d'étude suivant :

- construction d'un modèle au niveau des relations
- construction d'un modèle de chemins d'accès à partir du précédent
- évaluation du modèle ainsi construit

3. 5. - UN EXEMPLE :

Pour illustrer les remarques faites ci-dessus, nous avons extrait quelques éléments d'un problème qui nous a été posé dans le cadre d'une application à la gestion d'une entreprise (cf. annexe à la fin de la partie IV).

3.5.1. - Définition des relations :

3.5.1.1. - Les champs :

Leur liste avec les symboles qui seront utilisés est donnée dans le tableau de la figure IV. 5.

Le choix est arbitraire mais nous avons voulu limiter le nombre de données pour que l'exemple reste simple et démonstratif.

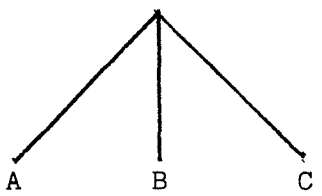
CHAMP	SYMBOLE
ADRESSE	AD
CHIFFRE D'AFFAIRE	CH
CLIENT	CL
PRIX DE VENTE	PV
PRODUIT	PR
REPRESENTANT	RE

Fig. IV. 5 - Les champs et leurs symboles

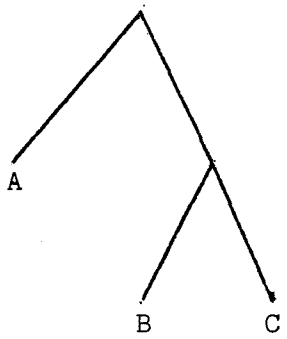
3.5.1.2. - Les relations :

Toujours par souci de simplicité nous limiterons l'essentiel de l'exposé qui suit aux relations binaires car elles permettent des représentations schématiques aisées. D'autre part , toute relation n-aire peut être représentée par un ensemble de relations binaires :

Ainsi, par exemple, la relation ternaire (A, B, C) peut-être considérée comme un ensemble de deux relations binaires et remplacée par (A, (B,C)) (cf.PAIR /5/) conformément au schéma ci-dessous :



(A,B,C)



(A, (B,C))

Plus généralement (A, B,..... N) peut-être remplacée par

(A, (B,C....., N)).

Nous définissons une relation binaire R dans l'ensemble des champs par la donnée du tableau matriciel de la figure IV. 6.

.../...

	AD	CA	CL	PV	PR	RE
AD			X			X
CA			X		X	X
CL	X	X			X	X
PV					X	
PR		X	X	X		X
RE	X	X	X		X	

Fig.IV.6 - Définition de la relation $\overline{R}$

Dans la définition de cette relation, nous n'avons associé que les champs qui présentent un lien au sens de l'application considérée. En effet, après une première analyse, le concepteur de l'application doit être à même de définir les liens entre les différents champs qu'il a répertoriés et, donc, de construire un tableau comme ci-dessus (fig. IV.6). Un tel tableau est ainsi une représentation de la sémantique de l'application avant toute tentative de structuration plus fine (en particulier hiérarchique).

Dans l'exemple que nous étudions, nous considérons qu'il existe un lien entre CLIENT et ADRESSE et nous écrivons " CL $\overline{R}$ AD" (ou " AD $\overline{R}$ CL") ; par contre, un lien entre PRIX-DE-VENTE et ADRESSE ne nous semble pas devoir être défini : " non [PV $\overline{R}$ AD]".

.../...

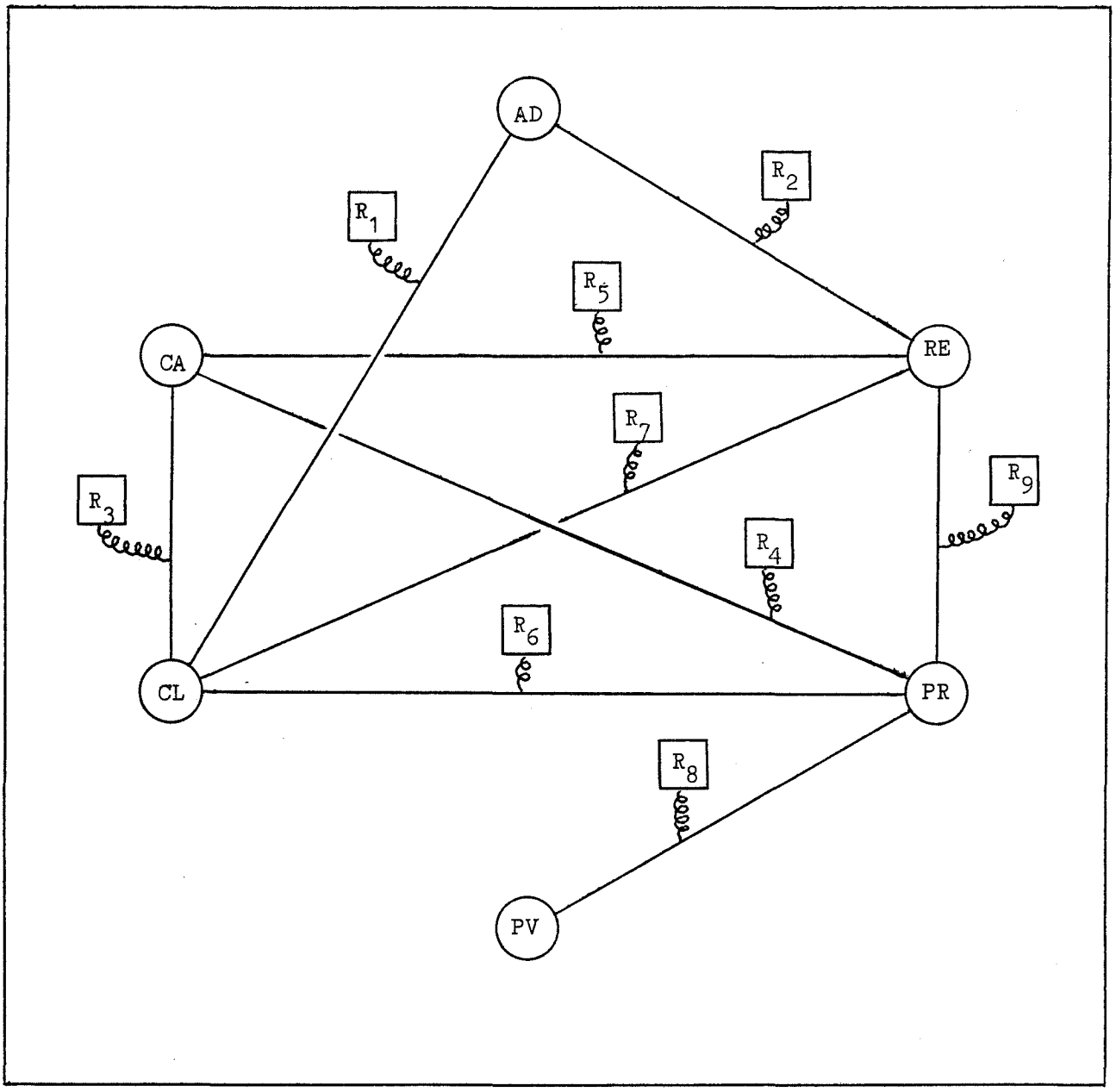


Fig.IV.7 - Liens sémantiques entre les champs.

Le tableau IV . 6 est caractéristique d'une relation symétrique. Ainsi, à ce niveau, nous ne définissons par un ordre et donc nous n'émettons aucune restriction sur la structure à élaborer. Une autre représentation est donnée figure IV. 7.

Dans cette représentation nous avons attribué une étiquette à chaque doublet dont les éléments vérifient $\overline{R}$. Cela revient à considérer 9 relations binaires R_i définies à partir du produit cartésien de 2 champs.

3. 5. 2. - Les questions posées :

Les champs utilisés étant choisis, nous avons extrait d'une liste de questions fournies par l'utilisateur toutes celles qui les font intervenir. Cette liste avec les symboles utilisés est donnée figure IV.8 p.28.

Chacune de ces questions utilise un ou plusieurs doublets définis dans 2.6.1. La question Q_4 par exemple utilise le doublet {PR , CL} noté R_6 ; la question Q_8 utilise les doublets {CA , PR} et {PR, CL} notés respectivement R_4 et R_6.

3. 5. 3. - Matrice " Questions - relations "

Nous pouvons alors construire une matrice qui donne une représentation des relations utilisées dans les différentes questions proposées (figure IV.9 p.29).

L'observation de cette matrice appelle quelques remarques :

a) Aucune colonne du tableau n'est vide, ce qui signifie que toutes les relations envisagées dans ce cas servent.

.../...

symbole	QUESTION
Q_1	Quels sont les clients affectés à un représentant donné ?
Q_2	Quel est le chiffre d'affaire du client X ?
Q_3	Quels sont les clients qui achètent le produit X ?
Q_4	Quels sont les produits achetés par le client X ?
Q_5	Quelles sont les adresses des représentants affectés au client X ?
Q_6	Quel est le prix de vente du produit X acheté par le client Y ?
Q_7	Quel est le chiffre d'affaire du représentant X ?
Q_8	Quel est le chiffre d'affaire pour le produit X du client Y ?
Q_9	Quel est le chiffre d'affaire pour le produit X du représentant Y
Q_{10}	Quelle est l'adresse du client X ?
Q_{11}	Quelle est l'adresse du représentant X ?
Q_{12}	Quels sont les représentants auprès du client X ?

Fig. IV.8 - Les questions avec leurs symboles

	R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8	R_9
Q_1							X		
Q_2			X						
Q_3						X			
Q_4						X			
Q_5		X					X		
Q_6						X		X	
Q_7					X				
Q_8				X		X			
Q_9				X					X
Q_{10}	X								
Q_{11}		X							
Q_{12}							X		

Fig. IV.9 - Matrice questions-relations

.../...

b) Les colonnes qui contiennent le plus de croix sont celles qui correspondent à R_6 (4 croix) et R_7 (3 croix). Ces deux relations sont donc celles qui sont citées le plus souvent (du moins si nous négligeons de tenir compte de la fréquence des questions).

Or, ces relations portent respectivement sur les doublets {CL, PR} et {CL, RE} . Il est intéressant de noter que le champ élémentaire CL intervient dans chacun de ces doublets, il joue donc un rôle important et sera un des éléments clés de la structure à construire (sommet de l'arbre par exemple).

c) Certaines lignes comportent plusieurs croix. Elles correspondent à des questions qui font appel à plusieurs relations. Par exemple :

Q_5 fait intervenir R_2 et R_7

Q_9 fait intervenir R_4 et R_9

d) La relation R_9 n'est utilisée que dans la question Q_9 et en même temps que R_4 . On peut se demander alors s'il ne serait pas plus simple d'envisager le groupement de ces deux relations binaires pour obtenir une relation ternaire définie sur le triplet {CA, PR, RE} Le temps de réponse à la question Q_9 diminuerait certainement. (Voir paragraphe 3.6.1.).

Le fait que R_4 intervienne aussi dans Q_8 et que, par défaut, les questions se trouvent avoir la même importance (lè même poids) nous conduit à conserver à R_4 et R_9 une existence propre.

e) Une autre représentation des rapports entre questions et relations est donnée figure IV.10.

.../...

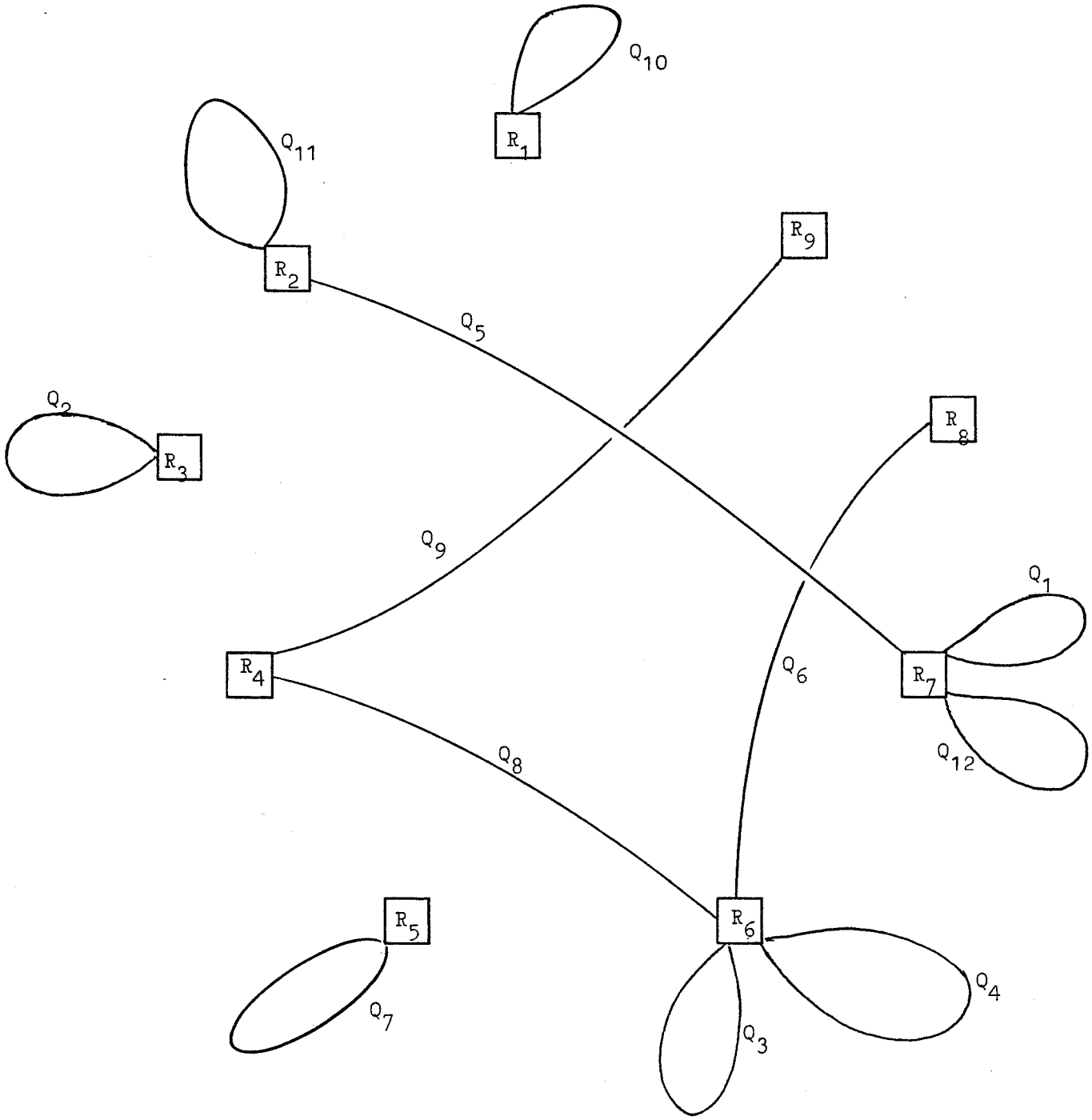


Fig.IV.10 - Questions-relations.

.../...

3.5.4. - Vers la structure logique :

Essayons à présent, par une analyse plus précise des relations et des questions dans lesquelles elles interviennent, de montrer comment peuvent s'opérer les choix conduisant au modèle de chemins d'accès, à la structure logique.

Il apparait nettement sur la figure IV. 9 et surtout sur la figure IV. IO qu'aux relations R_1, R_3, R_5 , R_8 et R_9 qui ne sont utilisées chacune que dans une seule question vont correspondre des accès dont les sens sont parfaitement définis.

Reprenons, par exemple, la question Q_{10} qui porte sur R_1 :

" Quelle est l'adresse du client x ? "

Cette question étant la seule faisant intervenir le doublet {AD , CL} nous pouvons respecter l'ordre hiérarchique imposé par les termes mêmes de la question et définir ainsi sans ambiguité le lien logique :

$$\begin{array}{c} CL \\ \downarrow \\ AD \end{array}$$

la citation correspondante étant "ADRESSE DU CLIENT".

Remarquons que cette option exprime le choix d'un chemin d'accès privilégié mais n'exclut pas la possibilité d'une requête correspondant à la situation inverse comme la recherche d'un client habitant à une adresse donnée (AD $\longrightarrow$ CL). Dans ce cas, la mise en oeuvre, par exemple, d'une procédure de recherche séquentielle permet d'apporter une réponse à la requête. Ce dernier accès étant certainement plus couteux que l'autre mais possible, nous voyons ainsi

qu'il s'agit plus de sélectionner les chemins d'accès principaux afin de les rendre plus rapides que d'éliminer les autres.

Par des raisonnements analogues portant sur les relations R_3, R_5, R_8, R_9 et se référant respectivement aux questions Q_2, Q_7, Q_6 Q_9 nous construisons les segments de structure logique suivants :

$$\begin{array}{ccccccc} CL & & RE & & PR & & PR \\ \downarrow & & \downarrow & & \downarrow & & \downarrow \\ CA & & CA & & PV & & RE \end{array}$$

Les autres relations sont citées dans plusieurs questions et nous devons les étudier séparément :

Les questions Q_8 et Q_9 portant sur R_4 impliquent toutes les deux le même chemin d'accès et déterminent le schéma :

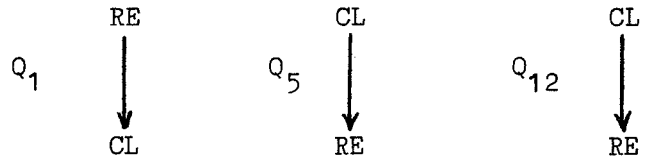
$$\begin{array}{c} PR \\ \downarrow \\ CA \end{array}$$

La situation est identique pour R_2 avec Q_5 et Q_{11} et nous obtenons :

$$\begin{array}{c} RE \\ \downarrow \\ AD \end{array}$$

Trois questions Q_1, Q_5 et Q_{12} portant sur R_7 conduisent séparément aux liens logiques respectifs suivants :

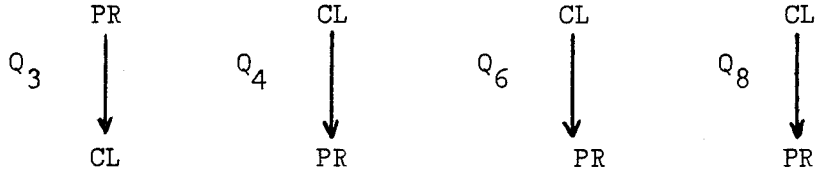
.../...



soit

Avec les hypothèses simplificatrices que nous avons faites et en supposant notamment que les poids des questions sont identiques, l'ordre CL $\longrightarrow$ RE peut être choisi mais dans la réalité il faut tenir compte d'autres éléments.

R_6 apparait dans les quatre questions Q_3 , Q_4, Q_6 et Q_8 et permet d'écrire :



soit

De ce schéma, avec les mêmes restrictions que dans le cas précédent, nous déduisons le chemin

Pour résumer, nous pouvons reprendre le schéma de la figure IV.7 en représentant des liens logiques (fig IV.11).

.../...

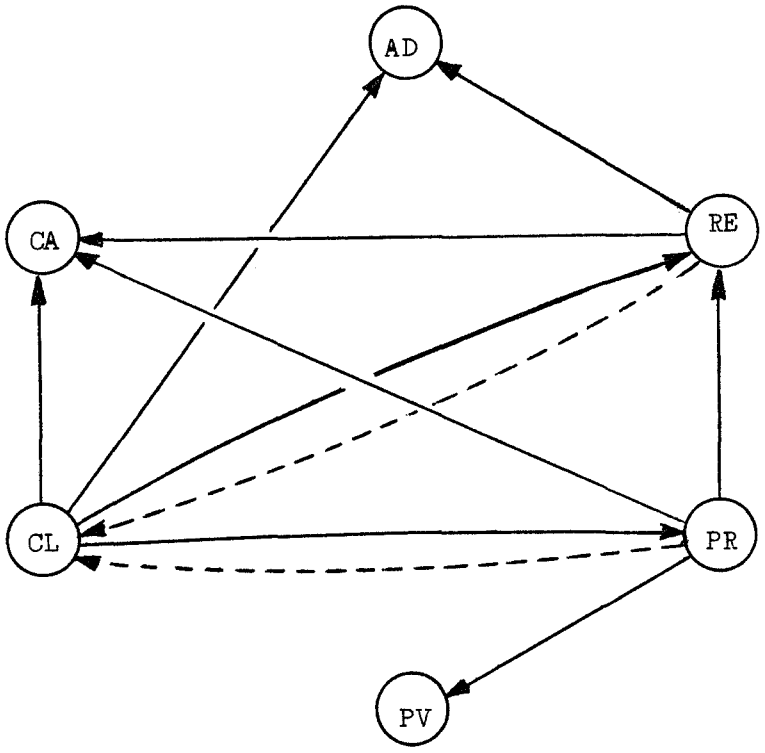


Fig.IV.11 - Liens logiques entre les champs.

Les flèches en pointillé signalent les liens logiques considérés dans l'étude sommaire précédente comme moins importants que les liens de sens opposés.

Nous obtenons ainsi une structure fonctionnelle que nous pouvons schématiser légèrement différemment (fig. IV. 12) et qui confirme le rôle privilégié du champ " CL".

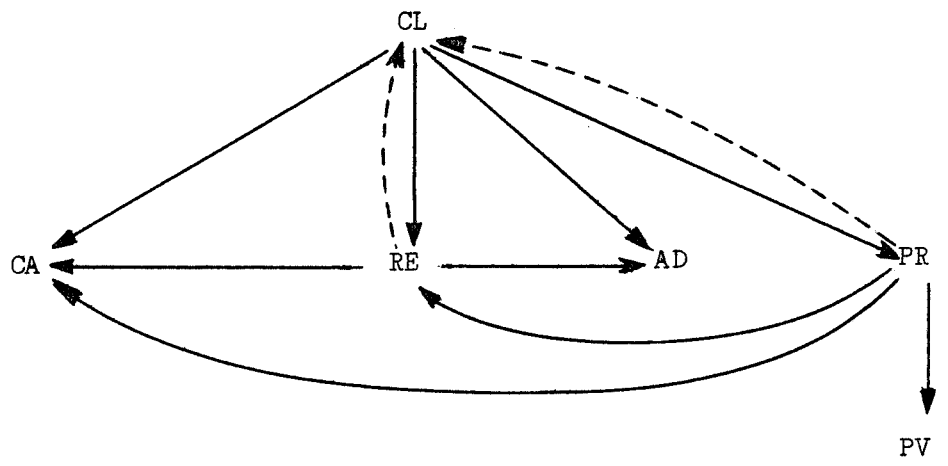


Fig.IV.12 - Structure fonctionnelle.

Une structure logique déductible de ce schéma est donnée dans la fig.IV.13.

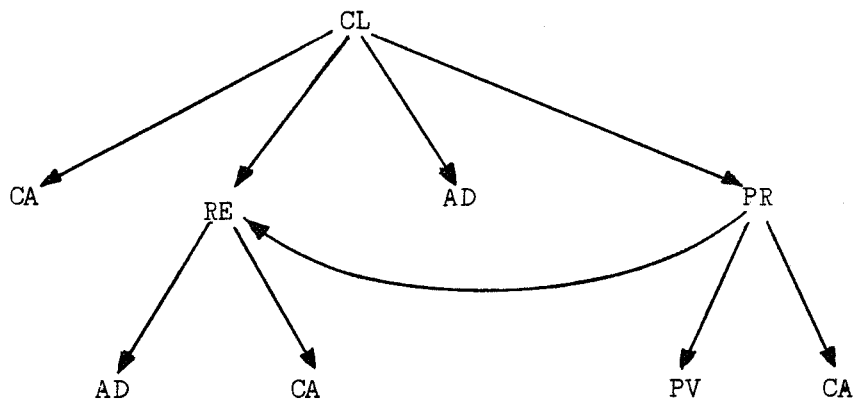


Fig.IV.13 - Structure logique.

Le fait que AD et CA apparaissent plusieurs fois dans ce graphe ne signifie pas qu'il y a redondance des informations mais simplement qu'il n'est ni indispensable ni économique de stocker par exemple les adresses des clients et celles des représentants dans la même caractéristique.

Les champs CLIENT (CL), REPRESENTANT (RE) et PRODUIT (PR) correspondant à des entités, la description du graphe de la figure IV.13 en langage de définition de structure SOMINE se fait ainsi :

```
ENTITE 2000 CLIENT
    DEBUT CHIFFRE-AFFAIRE NUMERIQUE  R
        ENTITE 20 REPRESENTANT
            DEBUT     ADRESSE TEXTE  2
                      CHIFFRE-AFFAIRE NUMERIQUE R
            FIN
        ENTITE 20 PRODUIT
            DEBUT     REPRESENTANT REFERENCE REPRESENTANT
                      PRIX-DE-VENTE NUMERIQUE R
                      CHIFFRE-AFFAIRE NUMERIQUE  R
            FIN
    FIN
```

3.5.5. - Une étude partielle :

Cet exemple montre la difficulté que représente l'élaboration d'une structure logique alors que, pourtant, l'on ne considère qu'un nombre restreint d'informations : listes réduites de champs élémentaires, de relations et de questions. Or, dans un problème réel, beaucoup d'autres éléments sont impliqués et les choix deviennent bien plus compliqués .

3.6. - NECESSITE D'UNE EXPERIMENTATION :

Compte-tenu de cette étude, nous ne croyons pas qu'il soit possible d'obtenir, même après une réflexion approfondie, pour un problème donné une structure logique d'information qui soit optimale . Une expérimentation assez longue et minutieuse doit suivre l'implémentation d'une première structure ; les observations faites concernant son utilisation doivent conduire progressivement à une structure rendant de meilleurs services.

Pour faciliter cette observation nous avons introduit dans SOMINE au niveau de l'implémentation et pour chaque caractéristique :

* un compteur permettant de connaître le nombre d'accès effectués dans le mode interrogation,
* un compteur renseignant sur le nombre d'accès dans le mode mise à jour (partie 3 paragraphe 2.2.)

3. 7. - CONCLUSION :

Dans ce chapitre nous avons posé beaucoup de questions qui peuvent se résumer ainsi :

* Comment construire un modèle au niveau des relations ?

* Comment ensuite définir un modèle de chemins d'accès ?

Au cours de leur étude, nous avons essayé de dégager une méthodologie qui permette, à partir d'un ensemble de questions à poser à la base, de définir une structure SOMINE donnant une bonne représentation de l'application étudiée .

Cette réflexion, partielle et sommaire, doit être précisée, approfondie et prolongée notamment afin de cerner ce qui, dans la démarche envisagée, peut se prêter à une automatisation. A titre d'exemple, nous citerons, deux pistes de recherche qui nous semblent intéressantes :

* L'étude linguistique des questions à poser à la base. Ces questions constituent un sous-ensemble du langage naturel qui pourrait être analysé par une méthode automatique analogue à celle qui est décrite par A. MESGUICH et B. NORMIER /10/. Il devrait être ainsi possible de générer la matrice questions-relations (§3.5.3.) et aussi peut-être d'étudier les liens logiques entre les champs (cf. 3.5.4.)

* Ces liens logiques étant obtenus, on peut les représenter à l'aide d'une matrice susceptible de permettre la production automatique d'une structure logique. A titre indicatif, les liens spécifiés dans le schéma de la fig. IV .11 (§ 3.5.4.) sont représentés à l'aide de la matrice fig. IV. 14 (p.suivante).

Dans cette matrice :

- une ligne horizontale peut avoir plusieurs croix : on peut en effet, pour un client donné parler de son adresse, de son chiffre d'affaire, du produit qu'il achète, de son représentant.

- une ligne verticale ne peut, par contre avoir plus d'une croix. Cela conduit à multiplier le nombre de colonnes ; ainsi, par exemple,

	AD		CA			CL		PV	PR	RE	
AD											
CA											
CL	X		X						X	X	
PV											
PR				X		⤫		X			X
RE		X			X		⤫				

Fig. IV.14 - Représentation matricielle de la figure IV.11
(les croix ⤫ correspondent aux flèches en pointillé)
(elles ne sont pas prises en compte dans la fig. IV.15)

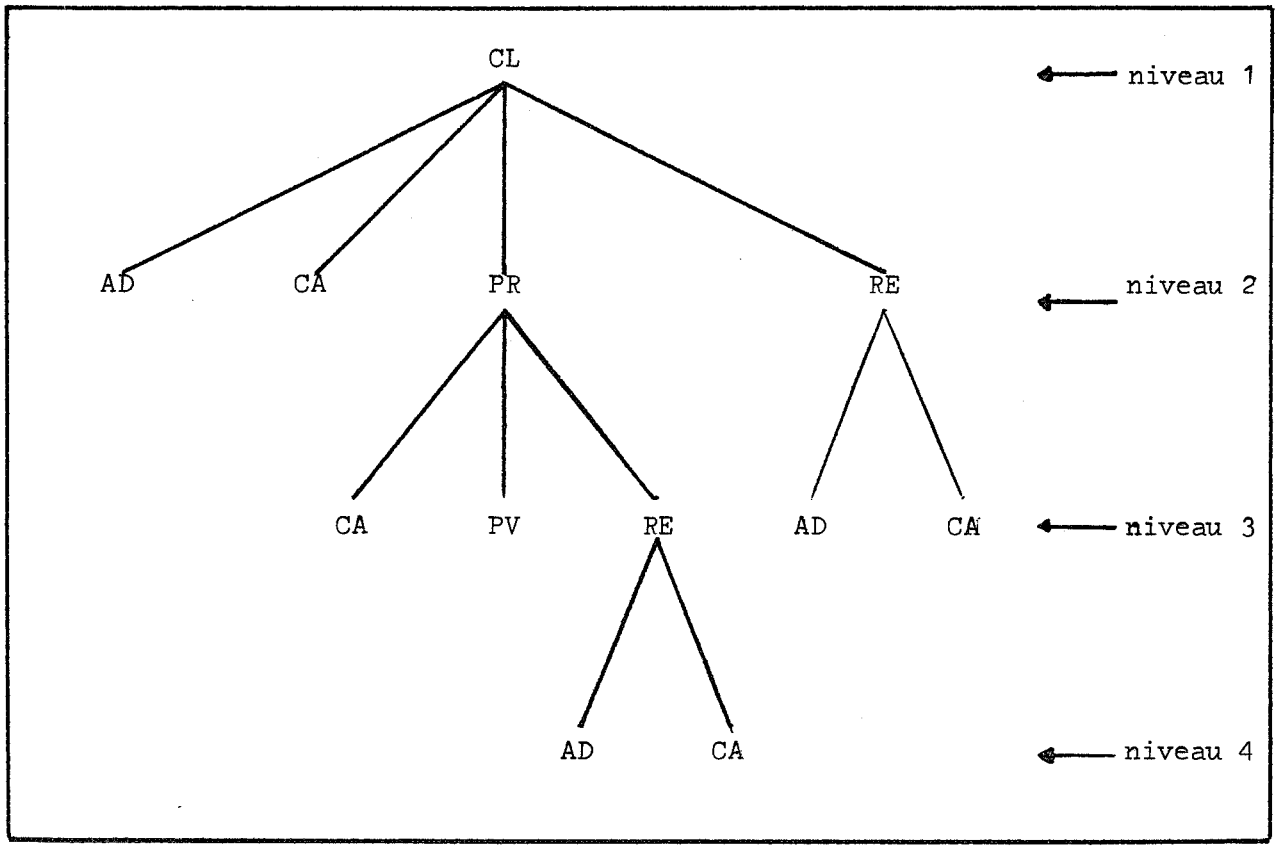


Fig. IV.15 - Structure déduite de la matrice IV.14

.../...

les notions d'adresse pour un client ou un représentant sont considérées comme deux notions distinctes.

L'exploitation de cette matrice se fait en examinant alternativement lignes et colonnes. Dans l'exemple choisi, la colonne CL qui ne comporte pas de croix est éliminée en premier, le champ CL est placé au niveau 1 de la structure (fig. IV. 15). Les croix qui figurent dans la ligne CL mentionnent des caractéristiques de niveau inférieur (niveau 2) que nous devons étudier successivement. Si les lignes qui leurs correspondent ne comportent pas de croix, ce sont des feuilles de la structure ; dans le cas contraire un niveau supplémentaire est défini.....

Le processus itératif simplifié que nous venons de décrire sommairement ne permet pas pour l'instant de résoudre tous les problèmes et des difficultés restent notamment pour les réfé - rences. Son avantage est qu'il peut conduire à la génération automatique d'une structure.

Cette étude et l'étude linguistique déjà évoquée peuvent, si elles sont développées suffisamment, apporter une contribution importante dans la recherche méthodologique que nous avons entreprise.

4 - MESURES

4. 1. - Pourquoi mesurer ?

Lorsque nous avons réalisé le système SOMINE nous nous sommes naturellement demandés quelles étaient ses performances . Cela nous a conduits à nous poser des questions telles que :

* les temps d'exécution des requêtes sont-ils bons ?

* Comment varient les performances suivant le type des structures utilisées (plate, haute, avec des références ,....) ?

Dans le paragraphe précédent, notamment dans 3.6.1. nous évoquons la notion de coût des accès. Comment évaluer le coût d'un accès à la base ?

4. 2. - Quoi mesurer ?

Certaines mesures peuvent intéresser les concepteurs du système afin qu'ils puissent connaître et améliorer le prototype. D'autres mesures (mais aussi éventuellement les mêmes) peuvent intéresser un utilisateur et lui permettre d'exploiter au mieux les possibilités du système.

Les mesures envisageables,qu'elles intéressent plus spécialement les concepteurs ou les utilisateurs, peuvent être classées en deux niveaux suivant qu'elles sont liées à la réalisation effective du prototype et donc, en particulier, au matériel utilisé ou suivant qu'elles sont liées à la logique même du S.G. B.D. :

.../...

* mesures au niveau réalisation :

- temps d'utilisation de l'unité centrale
- temps de réponse aux écrans en mode conversationnel
- etc....

* mesures au niveau logique :

- mesure du nombre des accès aux fichiers dans l'exécution d'une requête
- évaluation du nombre et de la nature des accès aux données lors d'une application
- mesures portant sur le système de gestion de mémoire
- etc ....

4. 3. - COMMENT MESURER ?

* Il est possible d'utiliser des instruments de mesure indépendants du S.G.B.D., par exemple :

- routines du constructeur pour les temps d'utilisation de l'unité centrale
- chronomètre manuel pour les temps de réponse aux écrans.

* Il est possible de prévoir dans le système des compteurs permanents ou provisoires.

- compteurs mesurant les accès aux données

* Faut-il créer un outil indépendant qui puisse relever les compteurs après une session de travail (ce qui peut être intéressant dans le cas d'une interruption accidentelle de session pour retrouver les modifications intervenues) ou stocker les résultats des mesures dans une partie de la structure SOMINE en créant une caractéristique "MESURE" mise à jour spontanément ?

4. 4. - NOS OPTIONS :

Restant dans l'optique d'aide à l'utilisateur indiquée dans l'introduction de ce chapitre, nous nous sommes orientés essentiellement vers l'étude des mesures susceptibles d'apporter à l'utilisateur des informations lui permettant d'utiliser au mieux le système SOMINE.

Une caractéristique importante de SOMINE étant la transportabilité, nous avons envisagé principalement des mesures non liées à la réalisation effective du prototype, mais relevant de la logique de conception du système. Toutefois, à titre indicatif, nous donnerons quelques résultats concernant la réalisation effectuée à l'Ecole Nationale Supérieure des Mines de St Etienne (calculateur P 1175 de la Société PHILIPS).

4. 5. - POUR UNE MEILLEURE STRUCTURATION DES INFORMATIONS :

Deux objectifs peuvent être poursuivis à ce niveau :

i. permettre une meilleure connaissance du système afin d'orienter l'utilisateur vers une première structuration de ses informations qui soit la meilleure possible. Pour cela nous essaierons d'évaluer des coûts d'utilisation des requêtes en fonction des structures utilisées.

ii. fournir à l'utilisateur les outils qui lui permettront d'analyser le fonctionnement effectif de sa base afin d'en tirer des conclusions en vue de modifications ultérieures.

4.5.1. - Détermination des coûts des accès :

Comme nous l'avons évoqué précédemment (paragraphe 3 de ce chapitre) lors de la phase de création d'une base qui consiste à définir une structure logique des informations, l'utilisateur est constamment obligé de faire des choix. Pour l'aider nous avons cherché à introduire une notion de coût d'exécution pour les différentes requêtes afin de pouvoir les comparer suivant :

.../...

- leur mode (création, suppression, interrogation, mise à jour)
- le type de caractéristique qu'elles mettent en jeu (entité, entité-choix, mot, référence, ...)
- le niveau d'accès dans la structure.

Pour définir cette notion de coût nous avons retenu le critère qui nous semble déterminant du nombre des accès aux fichiers virtuels. Nous avons distingué les accès au fichier-structure et les accès au fichier-données (cf. partie 3) . Pour évaluer ces nombres d'accès nous pouvions :

- soit provoquer l'incrémentation de compteurs (internes aux programmes) à l'occasion des accès à ces fichiers (qui se font toujours par appel des sous-programmes " ECRIRE " ou "LECTURE" ; cf. GAILLARD /9/).

- soit simplement effectuer ce comptage en étudiant les organigrammes ou programmes correspondants.

Nous avons choisi cette deuxième solution car elle nous permet une analyse précise des conditions dans lesquelles les accès interviennent et donc la possibilité d'établir des règles permettant le calcul du nombre des accès aux fichiers nécessités par n'importe quelle requête.

4.5.1.1. - Définitions et rappels :

Etant donné une structure arborescente, nous définissons et numérotons des niveaux conformément à la figure IV. 16 . Nous numérotons également les fils d'un même père structurel de la gauche vers la droite (numéro d'ordre).

Pour comprendre ce qui va suivre il est nécessaire de se rappeler qu'une requête SOMINE comprend la définition de l'action à entreprendre (création, mise à jour, ...) et une citation qui relie la

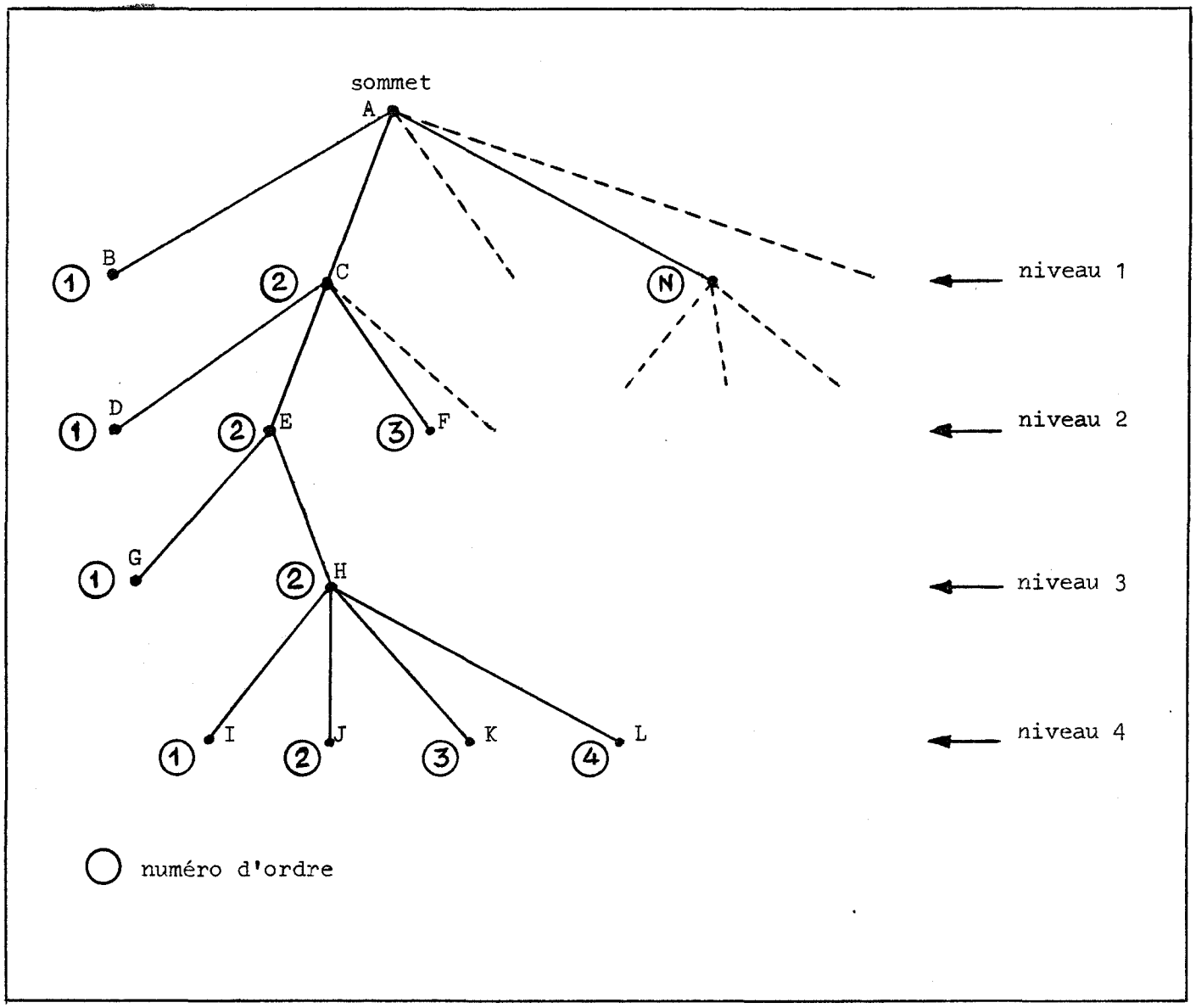


Fig. IV.16 - Niveaux et numéros d'ordre.

.../...

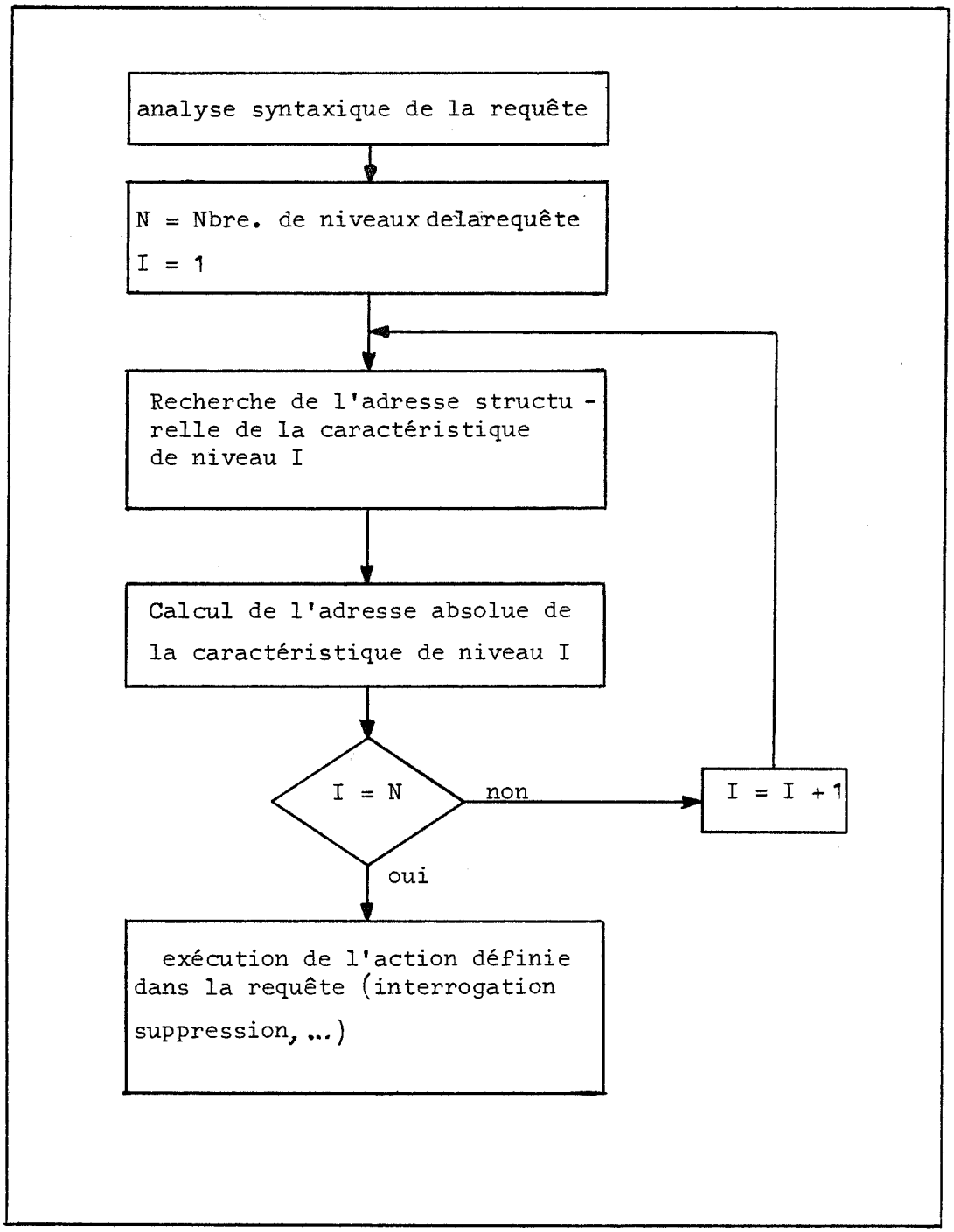


Fig. IV.17 - Programme d'exécution d'une requête

.../...

feuille sur laquelle porte la requête au sommet de l'arbre en désignant tous les niveaux intermédiaires :

exemple : M J DE H DE E DE C

mise à jour niveau 4 niveau 3 niveau 2 niveau 1

Les étapes principales du programme d'exécution d'une requête sont résumées figure IV. 17

Les notions d'ADRESSE STRUCTURELLE et d'ADRESSE ABSOLUE ont été définies dans la partie 3, elles correspondent respectivement à des adresses virtuelles dans le fichier-structure et dans le fichier-données.

4.5.1.2. - Exemples :

Nous donnons dans un tableau (fig. IV. 19) les nombres d'accès aux fichiers correspondant à différentes requêtes portant sur une base dont la structure est décrite fig. IV 18. Nous notons respectivement S et D les accès au fichier-structure et au fichier données.

L'analyse syntaxique d'une requête ne nécessitant aucun accès aux fichiers virtuels, le tableau ne comporte que des colonnes relatives aux 3 autres étapes du programme d'exécution d'une requête.

.../...

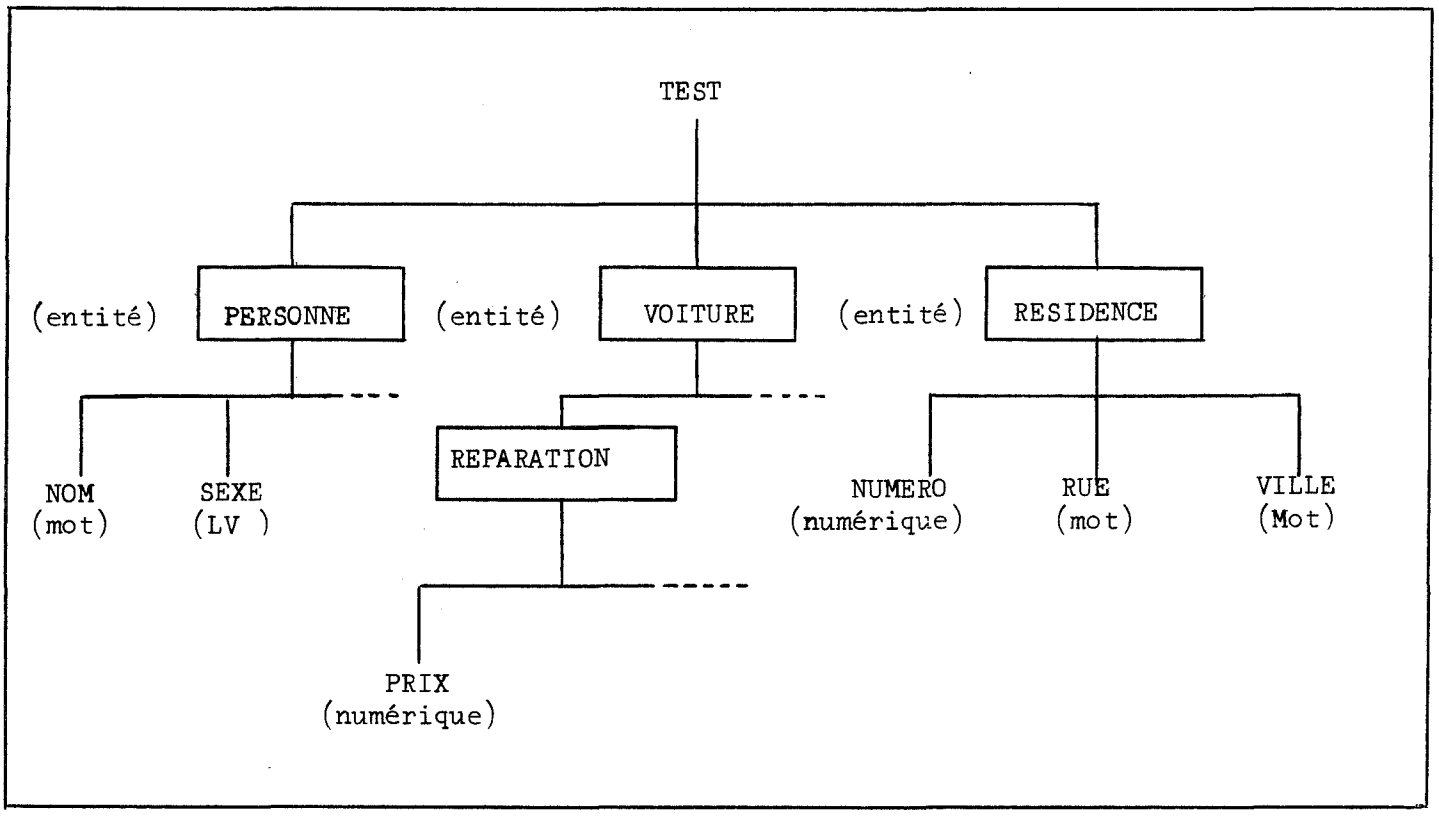


Fig. IV.18 - Structure "TEST"

4.5.1.3. - Calcul de l'adresse structurelle :

Dans la recherche de l'adresse structurelle d'une caractéristique :

* seul le fichier structure est utilisé
* le nombre d'accès dépend du niveau de la caractéristique et de son numéro d'ordre.

Si N est le nombre de niveaux intervenant dans la requête et si K (I) désigne le numéro d'ordre de la caractéristique de niveau I considérée

.../...

REQUETE	calcul adr. struct.	calcul adr. abs.	action
C PERSONNE 1	2S	1S	$2S + 4D + \left(\frac{MAX - 1}{32} + 1\right) D$ MAX = nbre max. de réali.
C PERSONNE 4	2S	1S	2S + 4 D
S PERSONNE 1	2S	1S	3S + 4D + ITAIL x D * + REF x 3D
C VOITURE 1	4S	1S	$2S + 4D + \left(\frac{MAX - 1}{32} + 1\right) D$
I NOM DE LA PERSONNE 2	5S	5S + D	2S + D
M SEXE DE LA PERSONNE 2	7S	5S + D	2S + MAX x S + 1D MAX = nbre d'éléments de la liste de valeurs
M PRIX DE LA REPARATION 1 DE LA VOITURE 1	10S	9S + 2D	S + D
M VILLE DE LA RESIDENCE 4	13S	5S + D	2S + D

* ITAIL = taille en mots de la réalisation
REF = nombre de maillons de la chaîne de références
S = accès au fichier-structure
D = accès au fichier-données

Fig. IV.19 - Coût de requêtes relatives à la structure TEST

.../...

comme fille de celle de niveau I-1, le nombre d'accès AS nécessité par le calcul d'adresse structurelle est :

$$AS = \left(\sum_{I=1}^{N} (2.K(I) + 1) - 1 \right) S$$

4.5.1.4. - Calcul de l'adresse absolue :

La recherche de l'adresse dans le fichier-données de la caractéristique étudiée va nécessiter la recherche successive des adresses absolues des caractéristiques intervenant dans la citation en commençant par le niveau 1. Le coût en accès de ces calculs dépend de la nature des caractéristiques considérées (BLOC ou ENTITE ou ENTITE-CHOIX...)

L'adresse absolue pour le niveau I étant connue, le calcul pour le niveau I + 1 implique :

4S + D	s'il s'agit d'une entité
4S + 2S	s'il s'agit d'une entité-choix
2S	s'il s'agit d'un Bloc
2S + D	s'il s'agit d'une référence
1S	s'il s'agit du dernier niveau

Si nous avons, par exemple, une citation à N niveaux qui comporte M idenfificateurs d'entités, L identificateurs de blocs, alors le nombre d'accès ABS nécessité par le calcul de l'adresse absolue est :

$$ABS = (M-1) (4S + D) + 2L \times S + S$$

soit $$ABS = (4M + 2L - 3) S + (M - 1) D$$

remarques :
- Cette formule est établie en considérant que le 1er niveau est 1 entité (d'où la présence de M-1)
- L + M = N

.../...

Création d'une réalisation d'entité		$2S + 4D + \underbrace{\left(\frac{MAX - 1}{32} + 1\right) D}_{\text{(seulement pour la 1ère création)}}$ MAX = Nbre maximal de réalisations de l'entité
Suppression d'une réalisation d'entité		3S +(4 + ITAIL + REF x 3) D (ITAIL = taille en mots de la réalisation) (REF= Nbre. de maillons de la chaîne réf.)
MOT TEXTE	(Interrog. ou mise à jour)	2S + D
NUMERIQUE INVERSE	(Interrog. ou mise à jour)	S + D
Liste de valeurs	Interrogation	2S + D (MAX = Nombre max. d'éléments de la liste)
	Mise à jour	2S + D + MAX x S
REFERENCE	SUPPRESSION	4S + Calcul de ABS de l'entité citée en référence + J x D + 4D (J = numéro d'ordre dans la chaîne des références)
	CREATION	4S + calcul de ABS de l'entité citée + 3 D + 1 D (seulement à partir du 2ème maillon de la chaîne)
IDEM		1S + même coût que la caractéristique citée

Fig. IV.20 - Coût des actions provoquées par les requêtes

.../...

4.5.1.5. - Action provoquée par la requête :

Le coût de cette phase d'exécution d'une requête dépend du mode de la requête et du type de la caractéristique sur laquelle elle porte.

Nous avons résumé les différentes possibilités dans le tableau IV.20. Il faut ajouter pour chacune d'entre-elles 1 accès au fichier-structure correspondant à la mise à jour des compteurs dont nous verrons l'utilité dans 4.5.2.

4.5.1.6. - Remarques :

1) Le système de gestion de mémoire (cf GAILLARD /9/) prévoit que 4 pages (257 mots) appartenant à l'un ou l'autre fichier résident en mémoire centrale. Lorsqu'il est nécessaire d'appeler une nouvelle page en mémoire centrale c'est la page résidente qui n'a pas été utilisée depuis le plus longtemps qui cède sa place.

Le fichier-structure dépasse rarement 2 pages ce qui correspond à environ 32 caractéristiques ; d'autre part, les informations qu'il contient sont constamment utilisées comme nous l'avons indiqué dans les paragraphes précédents. Nous pouvons donc espérer que les pages contenant la structure résident en permanence en mémoire centrale. Nous l'avons vérifié sur tous les exemples que nous avons traités en faisant éditer pour chaque opération effectuée les numéros des pages résidant en mémoire centrale et les anciennetés d'utilisation de ces pages.

Le fichier-données par contre est en général très vaste et l'exécution des requêtes entraînera de fréquents transferts de pages entre la mémoire centrale et la mémoire auxiliaire.

Ces considérations nous conduisent à accorder aux accès D un poids plus fort qu'aux accès S. Pour évaluer précisément ces poids il faudrait être en possession de suffisamment de résultats pour qu'un

Création d'une réalisation d'entité	$2S + 4D + \underbrace{\left(\frac{MAX - 1}{32} + 1\right) D}_{\text{(seulement pour la 1ère création)}}$ MAX = Nbre maximal de réalisations de l'entité
Suppression d'une réalisation d'entité	3S +(4 + ITAIL + REF x 3) D (ITAIL = taille en mots de la réalisation) (REF= Nbre. de maillons de la chaîne réf.)
MOT TEXTE (Interrog. ou mise à jour)	2S + D
NUMERIQUE INVERSE (Interrog. ou mise à jour)	S + D
Liste de valeurs — Interrogation	2S + D (MAX = Nombre max. d'éléments de la liste)
Liste de valeurs — Mise à jour	2S + D + MAX x S
REFERENCE — SUPPRESSION	4S + Calcul de ABS de l'entité citée en référence + J x D + 4D (J = numéro d'ordre dans la chaîne des références)
REFERENCE — CREATION	4S + calcul de ABS de l'entité citée + 3 D + 1 D (seulement à partir du 2ème maillon de la chaîne)
IDEM	1S + même coût que la caractéristique citée

Fig. IV.20 - Coût des actions provoquées par les requêtes

.../...

4.5.1.5. - Action provoquée par la requête :

Le coût de cette phase d'exécution d'une requête dépend du mode de la requête et du type de la caractéristique sur laquelle elle porte.

Nous avons résumé les différentes possibilités dans le tableau IV.20. Il faut ajouter pour chacune d'entre-elles 1 accès au fichier-structure correspondant à la mise à jour des compteurs dont nous verrons l'utilité dans 4.5.2.

4.5.1.6. - Remarques :

1) Le système de gestion de mémoire (cf GAILLARD /9/) prévoit que 4 pages (257 mots) appartenant à l'un ou l'autre fichier résident en mémoire centrale. Lorsqu'il est nécessaire d'appeler une nouvelle page en mémoire centrale c'est la page résidente qui n'a pas été utilisée depuis le plus longtemps qui cède sa place.

Le fichier-structure dépasse rarement 2 pages ce qui correspond à environ 32 caractéristiques ; d'autre part, les informations qu'il contient sont constamment utilisées comme nous l'avons indiqué dans les paragraphes précédents. Nous pouvons donc espérer que les pages contenant la structure résident en permanence en mémoire centrale. Nous l'avons vérifié sur tous les exemples que nous avons traités en faisant éditer pour chaque opération effectuée les numéros des pages résidant en mémoire centrale et les anciennetés d'utilisation de ces pages.

Le fichier-données par contre est en général très vaste et l'exécution des requêtes entraînera de fréquents transferts de pages entre la mémoire centrale et la mémoire auxiliaire.

Ces considérations nous conduisent à accorder aux accès D un poids plus fort qu'aux accès S. Pour évaluer précisément ces poids il faudrait être en possession de suffisamment de résultats pour qu'un

.../...

traitement statistique soit significatif. Nous essaierons de le faire lorsque l'expérimentation sera plus avancée. Nous tiendrons cependant compte de cette remarque dans les suivantes.

2) L'examen des formules qui permettent les calculs de l'adresse structurelle et de l'adresse absolue montre que :

* les numéros d'ordre des caractéristiques pour chaque niveau de la citation interviennent dans l'évaluation du coût du calcul de l'adresse structurelle mais seulement en accès de type S. Compte-tenu de la 1ère remarque leur incidence sur le coût n'est pas très importante

* Le nombre de niveaux de la citation intervient non seulement pour le calcul de l'adresse structurelle mais aussi pour le calcul de l'adresse absolue. Dans ce dernier cas la multiplication du nombre des niveaux entraîne un accroissement du nombre des accès D.

Ainsi, dans la mesure où la nature et la structure de ses informations le permettent, le créateur d'une base doit éviter de multiplier les niveaux et donc préférer une structure "large" à une structure "haute".

3) Le tableau donné en 4.5.1.5. permet de prévoir des coûts d'exécution sensiblement identiques quels que soient les types de caractéristiques élémentaires sur lesquels les requêtes porteront. Rappelons que nous ne tenons pas compte de tous les facteurs et que si, en particulier, nous considérons les problèmes d'édition, les interrogations d'un MOT ou d'un TEXTE de 50 lignes ne seront pas aussi rapides l'une que l'autre.

4) Les opérations de création ou de suppression de réalisations d'entités ou de références sont plus coûteuses que les autres requêtes ; toutefois, on peut raisonnablement penser que l'utilisateur d'une base manipulera moins ces types de requêtes que les autres, surtout après un premier "remplissage" de la base.

.../...

Notons également que la création d'une première réalisation d'entité est beaucoup plus coûteuse que la création des autres réalisations surtout si le nombre maximal des réalisations possibles est élevé ; elle n'a lieu qu'une fois par entité pour la durée de vie de la base.

4.5.1.7. - Conclusion :

Dans les paragraphes précédents, nous n'avons retenu qu'un critère pour l'évaluation des coûts d'accès aux informations contenues dans une base. Ce critère, s'il n'est pas le seul à pouvoir être retenu, est cependant significatif et nous verrons en étudiant les résultats de mesures effectuées sur le prototype que les conclusions développées ci-dessus sont confirmées (cf. 4.6.).

4. 5. 2. - Comptage des accès aux informations :

Ce n'est qu'en observant son fonctionnement qu'un utilisateur de base de données pourra vérifier si la base qu'il a créée répond au mieux à ses besoins.

Pour aider à l'analyse du fonctionnement d'une base nous avons incorporé au système deux compteurs qui permettent de connaître le nombre et le mode des manipulations effectives sur les caractéristiques.

Ces compteurs sont placés dans le fichier-structure (cf. partie 3) et ne perturbent pas sensiblement la durée d'exécution des requêtes. Ils présentent toutefois l'inconvénient de compter les manipulations à un niveau où il n'est pas possible de faire un discernement par réalisation. Leur relevé se fait par une requête spéciale (cf. SAYETTAT /8/).

Créer des compteurs au niveau des réalisations d'entité permet une connaissance plus fine du fonctionnement de la base (quelles sont par réalisation les caractéristiques les plus souvent modifiées, interrogées ?) et aussi une remise en état de la base après une

.../...

interruption accidentelle d'un programme de mise à jour. Les relevés de ces compteurs donnent en effet la possibilité de connaître d'une manière précise les modifications réalisées effectivement au cours d'une session. Toutefois, ces compteurs ne peuvent être implémentés que dans le fichier-données ; nombreux, ils risquent de coûter de la place et leur gestion du temps à l'exécution . Ces derniers inconvénients semblent moins importants que les avantages développés plus haut et nous envisageons :

- d'introduire ces compteurs au niveau des réalisations
- de créer une requête permettant leur relevé.

4. 6. - PREMIERES MESURES EFFECTUEES SUR LE PROTOTYPE :

Nous avons crée plusieurs bases de données d'essai sur le S.G.B.D. SOMINE actuellement opérationnel au Centre de Calcul de L'Ecole Nationale Supérieure des Mines de St.Etienne. Le calculateur utilisé est un P 1175 de la Société PHILIPS. Nous avons effectué des mesures sur ces bases en utilisant les routines du constructeur permettant de connaître les temps d'utilisation de l'unité centrale. Les principaux résultats de ces premières mesures sont commentés dans les paragraphes qui suivent .

4.6.1. - Analyse syntaxique et requête :

Ayant mesuré les temps d'exécution (temps d'utilisation de l'U.C.) de l'analyse syntaxique pour 50 requêtes de natures diverses portant sur 4 structures différentes nous constatons :

* La durée d'exécution du programme d'analyse syntaxique est pratiquement constantes quelle que soit la requête considérée. Nous trouvons seulement une différence de 4 % entre les temps les plus courts et les temps les plus longs (requêtes comportant le plus grand nombre de niveaux).

.../...

* le pourcentage de la durée de l'analyse syntaxique par rapport à la durée totale de la requête est très élevé. Le pourcentage moyen est de 48 %, l'étendue étant de 24 % (écart entre le plus faible et le plus élevé de ces pourcentages).

Rappelons que dans cette phase d'exécution d'une requête il n'y a aucun accès aux fichiers virtuels et que le critère de coût adopté précédemment n'intervient pas. La première constatation est donc cohérente avec notre hypothèse sur les coûts.

La deuxième constatation doit nous conduire lors de l'écriture d'une 2ème version de SOMINE à faire porter un effort particulier sur la programmation de l'analyse syntaxique.

4. 6. 2. - Les types de caractéristiques :

Sur une structure d'essai comportant des caractéristiques élémentaires (mot, texte, liste de valeurs, numérique (E, R), idem) filles d'une entité , nous avons créé, mis à jour complètement et interrogé 50 réalisations de l'entité.

Les différences entre les temps de mise à jour ou d'interrogation (temps U.C.) des différentes caractéristiques sont faibles (au maximum 6 %) . Aucune différence n'existe entre mot, LV et numérique par contre une augmentation de la durée est sensible pour la caractéristique texte (problème d'édition).

L'augmentation de la durée d'exécution d'une requête portant sur une caractéristique de type idem par rapport à la caractéristique citée est en moyenne de 3 %.

4. 6. 3. - Structure large - structure haute :

Nous avons constaté en comparant des requêtes portant sur des frères structurels du mêmettype (caractéristique MOT) que les différences entre les durées (temps U.C.) de requêtes impliquant deux frères

consécutifs étaient de l'ordre de 1 à 2 %.

Par contre, nous avons obtenu des différences plus importantes en effectuant des requêtes sur une structure à quatre niveaux . La durée (toujours en temps U.C.) augmente en effet d'environ 15 % à chaque fois que la citation dans la requête s'enrichit d'un niveau.

Ces résultats confirment les conclusions que nous avions formulées en 4.5.1.6. et qui informent le créateur d'une base SOMINE que les accès seront plus coûteux dans le cas d'une structuration "haute" que dans le cas d'une structuration "large".

4. 6. 4. - Référence :

La création d'un lien du type référence dure (temps U.C.) 15 % plus longtemps qu'une mise à jour d'une caractéristique élémentaire.

L'interrogation ou la mise à jour d'une requête comportant une référence à sensiblement même durée qu'une requête sans référence et ayant le même nombre de niveaux. (environ 2 % de différence).

4. 6. 5. - Devant l'écran :

Le calculateur utilisé travaillant en multiprogrammation, les conditions d'expérimentation sont soumises à des variations constantes et nous n'avons pas procédé encore à des relevés systématiques de temps de réponse aux écrans le système fonctionnant en mode conversationnel. Ces relevés ne seront précis que si les expériences ont lieu dans des conditions identiques le seul programme actif étant SOMINE. Toutefois, les indications approximatives obtenues aux écrans semblent confirmer avec une certaine amplification les résultats commentés dans les paragraphes précédents (écarts de 20 à 30 % par niveau, par exemple).

.../...

4. 7. - CONCLUSION :

* L'examen des premières mesures effectuées dans le cadre d'une réalisation précise permet de vérifier que le critère admis pour l'évaluation des coûts des accès aux informations contenues dans la base est significatif.

* L'estimation des coûts qui a été faite conduit aux principales conclusions suivantes formulées sous forme de conseils à l'utilisateur :

- dans la mesure du possible éviter les structures comportant trop de niveaux.
- ne pas hésiter à employer une structure large, la dégradation des performances en fonction de la largeur étant faible.
- utiliser indifféremment les caractéristiques élémentaires car elles entraînent des performances sensiblement identiques sauf peut-être pour la caractéristique TEXTE si le nombre de lignes qu'elle comporte est grand.

* Nous pensons que l'étude sur la détermination des coûts d'accès (§ 4.5.1.) peut, après quelques perfectionnements nécessaires (coût des requêtes booléennes, des requêtes groupées.....), servir systématiquement d'outil d'analyse pour toute structure logique avant même son utilisation effective dans la création d'une base de données.

En effet, il est possible et relativement facile d'ajouter au logiciel d'accès aux bases un module permettant, d'une manière optionnelle, de déterminer le coût d'une requête sans pour cela en provoquer nécessairement l'exécution. Nous avons établi, dans les paragraphes qui précèdent, des formules qui permettent le calcul des différents coûts d'accès à partir de variables qui sont toutes connues dès la fin de l'analyse syntaxique des requêtes (nombre de niveaux, numéros d'ordre, code des caractéristiques.....).

.../...

L'apport que nous proposons pour compléter la démarche méthodologique décrite dans le chapitre 3, est donc un calcul automatique du coût des requêtes après l'implémentation de la structure sur le support physique mais avant la création effective de la base de données. Cette manière de procéder permet de réajuster, si cela est nécessaire, la structure en tenant compte des temps d'accès des principales requêtes envisagées.

5 - CONCLUSION

L'utilisation d'une base de données passe par l'étude des informations et de leur structuration. De cette étude dépendent en grande partie les performances réalisées et la qualité des services rendus. La construction d'une structure logique à partir de "l'application et de son environnement" est une opération difficile. Un effort de recherche important reste à faire afin d'être en mesure de proposer une méthode précise pour générer une structure logique. Nous avons, pour notre part, tenté de montrer la complexité du problème à résoudre et de le présenter en des termes accessibles à tout utilisateur.

La méthodologie que nous avons essayée de dégager dans cette partie se décompose en trois phases :

1) Définir une structure à partir d'un ensemble de questions à poser à la base (chapitre 3) ;

2) Tester cette première structure (avant de créer la base) en évaluant le coût des principales requêtes devant être utilisées ; construire ainsi une version mieux adaptée de la structure (chapitre 4 , § 4.5.1) ;

3) Analyser le fonctionnement de la base en temps réel (après sa création) à l'aide des compteurs d'accès, et effectuer, si cela est possible, les modifications de structure qui en résulteraient (chapitre 4 , § 4.5.2.).

.../...

Ce dernier point et le fait que les conditions d'une application peuvent évoluer au cours de la vie d'une base, font qu'il est nécessaire d'envisager les différentes possibilités de modification de structure et d'aménager SOMINE de manière à ce que le maximum d'entre elles puissent être proposées à l'utilisateur.

L'étude des modifications de structure et l'approfondissement de la méthodologie décrite ci-dessus peuvent constituer des prolongements intéressants à nos travaux.

6 - BIBLIOGRAPHIE

/1/ C. DELOBEL
"Contributions théoriques à la conception et l'évaluation d'un système d'informations appliqué à la gestion "
Thèse d'Etat - Université de Grenoble - octobre 1973

/2/ M.E. SENKO, E.B. ATLMAN, M. ASTRAHAM, P.L. FEDHER
"Data Structure and accessing in data systems "
IBM - Systems Journal - Nov. 1973

/3/ C. DELOBEL
"Aspects théoriques sur la structure de l'information dans une base de données"
R.I.R.D. n° B 3 - 1971

/4/ C.DELOBEL
"Les systemes de bases de données "
Université de grenoble - juin 1975

/5/ C.PAIR
"Structures de données et algorithmes fondamentaux"
Ecole Nationale Supérieure des Mines et des Industries Métallurgiques de Nanvy

/6/ J. REGNIER et M. MONTMOLLIN
"Reconnaissance de l'organisation, recherche de l'ordonnancement et choix du mode d'enseignement de la matière"
Actes du colloque de l'O.T.A.N. de Nice - mai 1968

.../...

/7/ J.R. ABRIAL
"Data Semantics"
Université de Grenoble - novembre 1973

/8/ C. SAYETTAT
"Contribution à la conception, la réalisation et l'utilisation du système de gestion de bases de données SOMINE : les accès aux bases - aide à la conception assistée par ordinateur "
Thèse de Doctorat de Spécialité. Ecole Nationale Supérieure des Mines de St. Etienne - janvier 1976

/9/ M. GAILLARD
"Contribution à la conception, la réalisation et l'utilisation du système de gestion de bases de données SOMINE : gestion des mémoires-enseignement assisté par ordinateur ".
Thèse de Doctorat de spécialité. Ecole Nationale Supérieure des Mines de St.Etienne - janvier 1976

/10/ A. MESGUICH, B. NORMIER
" Une expérience d'interrogation de base de données en un sous-ensemble du langage naturel"
R.A.I.R.O. n° juillet 1975 B-2

.../...

ANNEXE

UNE STRUCTURE UTILISEE EN GESTION

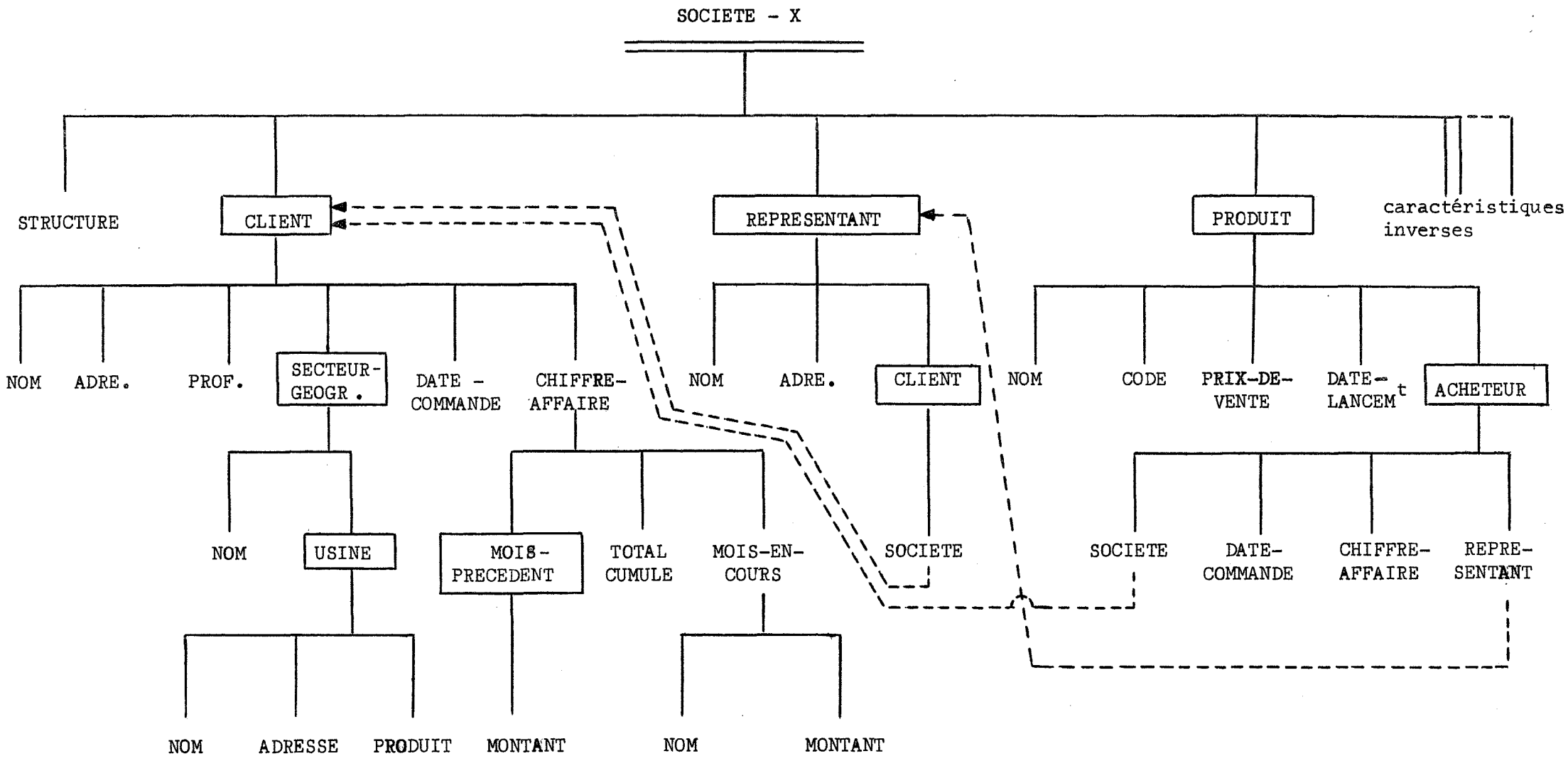
UN EXEMPLE DE STRUCTURE UTILISEE DANS UNE APPLICATION A LA GESTION

"X" est une entreprise qui fabrique des outils coupants de précision et des machines à affûter soit une vingtaine de catégories de produits (ébauches, cloches,....).

Les clients, environ 5000, sont caractérisés par leur catégorie professionnelle (revendeurs, automobiles, aviation, confrères,....) et leur secteur géographique (25 secteurs en France et à l'étranger). Des représentants exercent dans un ou plusieurs secteurs géographiques, avec certains clients pour une ou plusieurs catégories de produits.

Nous donnons dans les pages qui suivent l'arbre structurel (et sa traduction en LDS) qui correspond à cet exemple.

.../...



Les mots encadrés sont des identificateurs d'entités

Les flèches en traits discontinus représentent des références

Les caractéristiques inverses ne sont pas détaillées sur ce schéma

```
SOCIETE-X
DEBUT
   STRUCTURE TEXTE 50
   ENTITE 2000 CLIENT
       DEBUT  NOM MOT 28
              ADRESSE TEXTE 2
              PROFESSION (REVENDEUR AVIATION AUTOMOBILE CONFRERE
                          PARTICULIER AUTRE) 8
              ENTITE 25 SECTEUR-GEOGRAPHIQUE
                   DEBUT  NOM  MOT 12
                          ENTITE 10 USINE
                              DEBUT  NOM MOT 28
                                     ADRESSE TEXTE 2
                                     PRODUIT MOT 16
                              FIN
                   FIN
              DATE-COMMANDE  MOT 2
              CHIFFRE-AFFAIRE
                   DEBUT ENTITE 12 MOIS-PRECEDENT
                           DEBUT MONTANT NUMERIQUE R
                           FIN
                     TOTAL-CUMULE NUMERIQUE R
                     MOIS-EN-COURS
                           DEBUT NOM NUMERIQUE E
                                 MONTANT NUMERIQUE R
                           FIN
                    FIN
       FIN
   ENTITE 20 REPRESENTANT
       DEBUT  NOM MOT 16
              ADRESSE TEXTE 2
              ENTITE 300 CLIENT
                     DEBUT SOCIETE REFERENCE UN CLIENT
                     FIN
        FIN
```

.../...

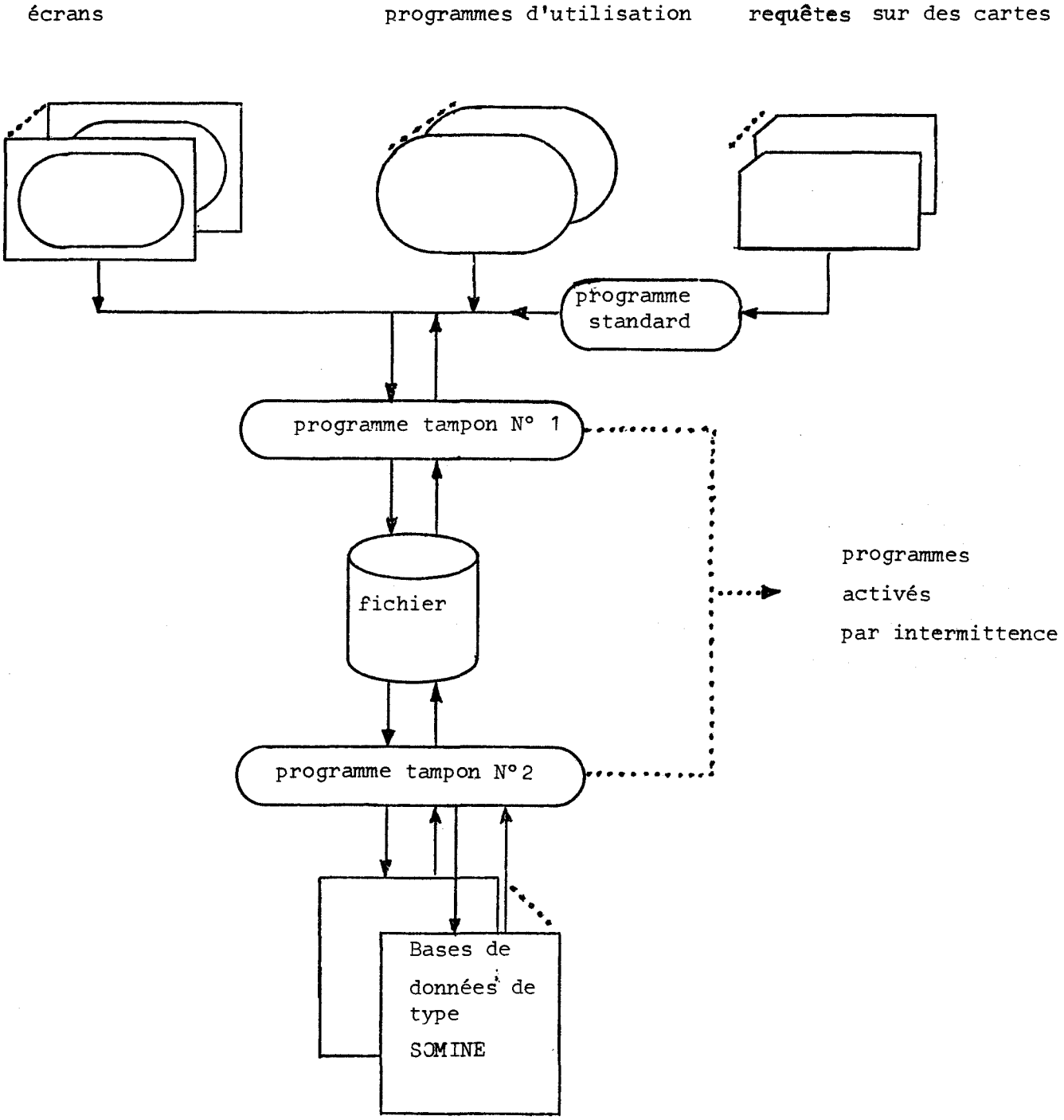
```
ENTITE  20  PRODUIT

        DEBUT  NOM MOT 16
        CODE NUMERIQUE  E
        PRIX-DE-VENTE  NUMERIQUE R
        DATE-DE-LANCEMENT MOT 2
        ENTITE 2000 ACHETEUR

                    DEBUT SOCIETE REFERENCE UN CLIENT
                          DATE-COMMANDE  MOT 2
                          CHIFFRE-AFFAIRE IDEM CHIFFRE-AFFAIRE
                          REPRESENTANT REFERENCE UN REPRESENTANT

                    FIN

        FIN
REVENDEUR INVERSE UN CLIENT
AVIATION INVERSE UN CLIENT
.........................
AUTRE INVERSE UN CLIENT
PRODUIT 1 INVERSE UN CLIENT
.........................
PRODUIT 20 INVERSE UN CLIENT

FIN ***
```

PARTIE V

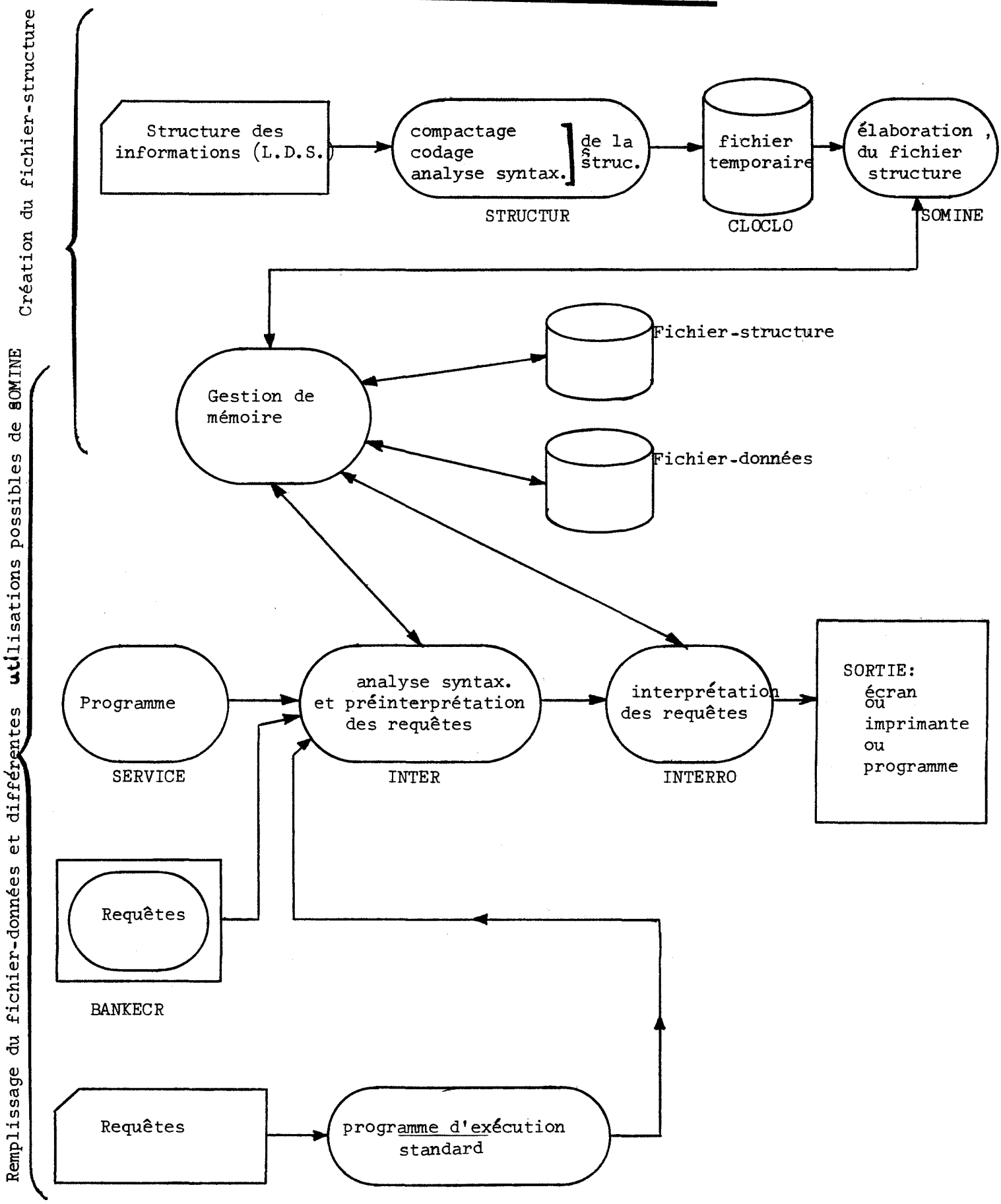
CONCLUSION

Voici venu le moment de faire la synthèse et le bilan des travaux que nous avons effectués et dont une partie fait l'objet de cette thèse. C'est le système de bases de données SOMINE qui constitue l'essentiel de nos travaux, nous allons le schématiser et le concrétiser avec les figures des deux pages suivantes. La première représente l'utilisation d'un système de bases de données de type SOMINE, la deuxième illustre le fonctionnement d'une de ces bases depuis l'étape de création jusqu'aux différentes possibilités d'utilisation.

SYSTEME DE BASE DE DONNEES SOMINE



UNE BASE DE DONNEES SOMINE



En réalisant le système de bases de données SOMINE, nous avons essayé de résoudre le problème de l'optimisation de l'organisation et de la recherche des informations appartenant à une collection d'informations.

Par certains côtés ce souci de discipline dans l'organisation et la structuration des informations en vue d'approcher d'une connaissance n'est pas sans rappeler la structure même du cerveau humain. On peut aussi penser aux philosophies orientales qui par des postures physiques et mentales essayent d'atteindre la connaissance et le bonheur.

Nous nous sommes donc attachés à trouver une philosophie applicable aux bases de données et assurant la vie de la collection d'informations qu'elles gèrent. Cette vie devra être conforme aux objectifs que nous nous étions fixés dans l'introduction de ce rapport, la nécessité de quelques autres critères s'est aussi dégagée au cours de notre travail.

Le premier objectif retenu est essentiel puisque c'est de lui que découle la notion même de base de données. Il s'agit d'assurer un traitement varié et efficace des informations contenues dans la base. Nous pouvons développer un peu les principales actions qui devront être réalisables pour que le traitement des informations soit satisfaisant. La base devra assurer :

- le placement des informations dans les mémoires auxiliaires réservées à cet effet.

- la recherche et la découverte rapide d'une information sur ces mêmes supports.

- enfin, et c'est ce point qui permet de juger de l'efficacité d'une base, les informations doivent pouvoir être manipulées dans le but d'en créer de nouvelles. Ces créations peuvent être faites de plusieurs façons, nous pouvons citer les calculs sur les éléments de la base ainsi que les informations composées par coordination ou subordination des éléments de la base. Enfin des applications telles que la concaténation d'informations ou les filtres sur les informations permettent de juger de l'aptitude à donner naissance à de nouvelles informations obtenues à partir de celles qui sont dans la base.

.../...

Si le traitement des informations est assuré par une base, on peut se fixer d'autres objectifs permettant d'en faciliter l'utilisation ou d'en augmenter les qualités. les critères suivants nous ont semblés intéressants à retenir :

- indépendance du système par rapport au matériel utilisé ceci afin d'en assurer la transportabilité.

-indépendance de la collection d'informations par rapport aux utilisateurs (personnes ou programmes)

- adaptabilité à de nouvelles exigences .

- fiabilité et protection de la base .

- mesurabilité du système en vue d'améliorer ses performances ou de modifier la structure des informations.

Ces critères que nous avons retenus sont le fruit de notre réflexion et de notre expérimentation . Nous avons également consulté " l'étude sur les softwares de gestion et d'interrogation des fichiers " publication éditée par la délégation à l'informatique groupe " systèmes d'informations " (avril 1973).

Ce document constitue une très bonne étude comparative des différentes bases de données. Il se termine en donnant pour chacun des systèmes de bases de données étudié une fiche signalétique. Nous avons retenu ce modèle de fiche signalétique pour SOMINE, nous l'avons complété et placé en annexe de cette conclusion.

Le lecteur pourra s'y reporter pour connaître des détails techniques mais nous allons faire ici un bref rappel des originalités ou des qualités du produit SOMINE; dans cette énumération nous aurons présent à l'esprit les objectifs que nous nous étions fixés.

.../...

- Naturellement les bases SOMINE peuvent effectuer les actions de rangement et de recherche des informations qui leur sont confiées. Cela est possible grâce aux requêtes de base dont la forme est simple et logique et dont l'apprentissage est rapide.

- La mise en oeuvre des filtres (caractéristique inverse) et des requêtes booléennes assurent une bonne faculté de création d'informations non contenues explicitement dans la base. Toute requête contient une citation hiérarchique elle assure donc la naissance d'informations composées par subordination, quant aux informations composées par coordination elles peuvent être obtenues par les requêtes groupées ou celles contenant le séparateur "tout". Une assez grande possibilité de manipulation des informations est laissée à l'utilisateur qui pourra les mettre en oeuvre dans ses programmes d'utilisation.

- Le système peut être amélioré assez facilement car la programmation des différentes parties est modulaire. De plus les automates de définition sont simples et une ou plusieurs transitions entre deux états sont toujours possibles à programmer. En somme nous pouvons dire que SOMINE est un produit évolutif, c'est -à- dire que le rajout d'une nouvelle fonction de manipulation, d'un nouveau type de caractéristique ou d'un nouveau mode de recherche ne brise en rien l'unité du système.

- Puisque ce logiciel est adaptatif il est perfectible. Naturellement la programmation n'en est pas parfaite et une optimisation peut toujours être envisagée, mais le plus intéressant est la possibilité d'effectuer une programmation plus fine (en assembleur par exemple) de certaines parties du système. Ceci ne peut être envisagé nous semble -t-il que dans le cadre d'une application donnée et pour un matériel donné. En effet généraliser cette remarque irait à l'encontre de l'objectif suivant qui est :

- l'indépendance vis à vis du matériel utilisé. Pour Somine celle-ci est assurée par la programmation effectuée presque totalement en Fortran d'où la transportabilité du produit.

- rappelons que les programmes d'implémentation de la structure et les programmes d'utilisation et d'analyse des requêtes sont totalement indépendants, ceci évite d'avoir à recréer le fichier des informations si les traitements qu'elles subissent varient.

.../...

- SOMINE est un système mesurable sous deux aspects. D'une part l'évaluation du temps d'exécution de l'implémentation ou de l'utilisation d'une base peut être faite facilement. D'autre part des compteurs situés dans la structure permettent d'espionner et de compter le nombre d'accès à chaque caractéristique. Une simple lecture de ces compteurs permet des évaluations statistiques et peut-être une modification de la structure des informations s'en déduira ce qui pourra contribuer à l'optimiser.

- du point de vue de l'utilisateur nous remarquons que les modes d'accès aux bases de type SOMINE ont été diversifiés : batch, écran, programme. Ce dernier point, l'appel par programme, est dû au fait que les requêtes sont traitées comme des chaînes alphanumériques et peuvent donc faire partie d'un programme écrit dans un langage qui permet leur traitement. Ces chaînes alphanumériques sont des phrases syntaxiquement correcte du langage de requête dont l'apprentissage est particulièrement rapide et logique.

- la protection peut être assurée facilement par un mot (ou un code) de passe contenu dans la requête ou la précédant.

- l'utilisation simultanée d'une base par plusieurs utilisateurs est réalisée par deux programmes tampons activés à des intervalles de temps réguliers et qui ont accès à un fichier qui contient les requêtes ainsi que le nom de l'utilisateur, ce fichier sert aussi de véhicule aux informations qui transitent entre la base et les utilisateurs. Ces programmes et ce fichier assurent le partage du temps entre les différents utilisateurs, les accès à la base sont déclenchés dans l'ordre dans lequel ils ont été demandés.

Ces différents rappels ont permis de montrer que SOMINE a atteint les objectifs que nous avions retenus dans l'introduction.

On peut se demander si SOMINE n'est pas un langage. Nous pensons que oui et qu'il est complémentaire des langages algorithmiques comme Fortran, Algol, PL/1 puisque les requêtes de SOMINE peuvent être traitées par ces langages. Nous ne pensons pas que SOMINE fasse double emploi avec PL/1 car la définition des structures et les différentes utilisations possibles sont dynamiques dans SOMINE. Comme atout pour notre produit nous pouvons ajouter le fait que la gestion de mémoire est tout à fait invisible à l'utilisateur. SOMINE a aussi l'avantage d'être efficace dans la recherche des données manipulées.

.../...

En effet, la structuration des informations sous forme de graphe ou d'arbre est plus riche que la plupart des structurations trouvées dans les langages de programmations usuels. Rappelons que :

- Fortran travaille sur des tableaux de taille fixe.

- Algol travaille sur des tableaux de tailles variables (la mémoire libre est gérée par le système).

- Dans PL/1 il existe des structures imbriquées mais l'allocation et la désallocation de la mémoire doivent être demandées par l'utilisateur.

- Dans le langage Pascal il existe également des structures mais elles sont statiques.

Nous espérons que ces rappels et ces comparaisons ont mis en évidence les originalités de SOMINE. La souplesse et la simplicité des accès à la base favoriseront, nous l'espérons, son utilisation dans divers domaines.

En dehors de la conception et de la réalisation de SOMINE nos recherches se sont orientées vers l'aspect utilisation et cela dans trois directions principales :

* La Conception Assistée par Ordinateur (cf. thèse de C. SAYETTAT). Cette application a permis :

- de montrer l'aide que peut apporter l'utilisation de bases de données SOMINE dans la réalisation de problèmes scientifiques ou techniques,

- de vérifier le caractère évolutif de SOMINE en enrichissant la structure de caractéristiques supplémentaires ("PROGRAMME" et "VALEUR + PROGRAMME").

.../...

* L'Enseignement Assisté par Ordinateur (cf. thèse de M.GAILLARD). Le rôle efficace de SOMINE comme pivot d'un logiciel servant de support à un enseignement de type non directif est décrit dans cette application.

* L'aide à la conception d'une base de données SOMINE. Cette étude est exposée dans la partie IV de ce mémoire. Nous avons recherché une méthodologie permettant, à partir d'un problème donné, de définir la structure SOMINE la mieux adaptée, c'est-à-dire, permettant de répondre essentiellement aux deux questions suivantes :

- Est-elle une bonne représentation de l'application mise en oeuvre ?
- Est-elle optimale au niveau des temps d'accès ?

La méthode que nous développons permet d'approcher de la solution optimale en trois temps successifs :

1 - Une structure peut être définie à partir des informations et d'un ensemble de questions à poser à la base.

2 - Cette première structure peut être testée par évaluation du coût des principales requêtes devant être utilisées. Cette évaluation, en cours d'automatisation, peut conduire à une deuxième version, mieux adaptée, de la structure.

3 - Enfin, la structure précédente étant implantée, l'utilisation de la base en temps réel dans l'application concernée peut être analysée à l'aide des compteurs d'accès. Les modifications éventuelles qui en résultent doivent permettre encore de perfectionner la structure à condition, naturellement, que les problèmes de modification de structure soient résolus. Cela constitue un prolongement de notre travail dans son aspect conception et élaboration du système SOMINE.

.../...

Nous venons d'exposer une partie du travail qu'a réalisé notre équipe depuis deux ans que l'idée du produit SOMINE est née. Tant que les travaux continuent, le système évolue et nous ne manquerons pas de publier ses perfectionnements ainsi que l'état d'avencement des études dans chacun des domaines explorés. Nous espérons que notre équipe a la souplesse et l'adaptabilité de notre produit mais nous souhaitons également posséder l'imagination et la créativité qui seules peuvent rendre notre système de bases de données utile et utilisé.

Nous pensons continuer de rechercher ou d'exploiter les voies d'utilisation ouvertes; n'oublions pas cependant qu' apprendre à créer et à utiliser une base SOMINE ne demande pas beaucoup de temps ni de peine. Nous espérons que cette facilité encouragera les utilisateurs éventuels que nous sommes prêts à renseigner et à former.

A N N E X E

F I C H E S I G N A L E T I Q U E D E S O M I N E

extraite de "l'étude sur les SOFTWARES
de gestion et d'interrogation de Fichiers"
Délégation à l'informatique, groupe
"Systèmes d'information" -

LES CRITERES DES FICHES SIGNALETIQUES

Elles peuvent constituer un outil de travail important pour l'utilisateur. Elles sont présentées de manière à pouvoir comparer caractéristique par caractéristique en suivant la numérotation. Au total soixante six paramètres caractérisent le S.G.B.D.

Sept parties définissent la fiche :

1 - Les renseignements généraux (Cinq caractéristiques)

On trouvera là en particulier le principe de commercialisation et les installations existantes à la date d'édition de ce rapport.

2 - Les caractéristiques hardwares et softwares (Onze caractéristiques)

Elles définissent la configuration, les terminaux et le système d'exploitation nécessaires à la mise en oeuvre du S.G.B.D.

3 - La souplesse du S.G.B.D. (Sept caractéristiques)

Elle précise :

- L'existence d'un macro-langage de génération du système
- La facilité d'utilisation
- La portabilité du sytème au sens changement de configuration et/ou changement d'ordinateur
- La compatibilité fichiers-langage de haut niveau

4 - L'organisation des fichiers (Onze caractéristiques)

Elle précise :

- La structure logique : le type, l'organisation logique, les correlations possibles et le langage de description
- La structure physique

5 - Les fonctions remplies par le système (Vingtcaractéristiques)

Elles précisent :

- L'interface de création
- Les possibilités de consultations
- La mise à jour
- Les possibilités de traitement
- Les possibilités d'édition

6 - Maintenance - Sécurité - Protection - Intégrité (Huit caractéristiques)

7 - Renseignements divers (quatre caractéristiques)

Nous donnons en annexe un modèle de fiche signalétique avec un commentaire si nécessaire des critères retenus.

(extrait de l'"étude comparative de classes de softwares de gestion et d'interrogation de fichiers" - CETE d'AIX.EN.PROVENCE - 1973)

.../...

I - RENSEIGNEMENTS GENERAUX :

1.1. - Nom du système

SOMINE

1.2. - Réalisateurs :

Marcel GAILLARD
Pierre MARTY
Claudette SAYETTAT

Ecole Nationale Supérieure des Mines
SAINT - ETIENNE

1.3. - Personnes responsables:

les réalisateurs

II - CARACTERISTIQUES HARDWARES ET SOFTWARES :

2. 1. - Unité centrale

SOMINE est implémentable sur tout matériel.

2.2. - Taille mémoire centrale nécessaire

80 K octets

2.3. - Unités périphériques nécessaires :

. lecteur de cartes
. disques
. imprimante
. écrans alphanumériques ou graphiques

2.4. - Types de terminaux supportés

. toute imprimante ou lecteur de cartes
. tous écrans graphiques ou alphanumériques

2.5. - Nombre de terminaux supportés

limité par le temps de réponse seulement

2.6. - Taille mémoire centrale supplémentaire par terminal

non chiffrée actuellement

.../...

2.7. - Système d'exploitation utilisé

Système PHILIPS P 1000

2.8. - Langage de programmation du S.G.B.D.

SOMINE est programmé en FORTRAN (+ 50 instructions en assembleur PHILIPS)

2.9. - Taille du système

limitée par les matériels utilisés

2.10. - Processeurs nécessaires pour l'utilisation du système

- Compilateur Fortran
- éditeur de liens
- système de gestion des terminaux

2.11. - Type du système

réseau

III - SOUPLESSE DU SYSTEME :

3.1. - Macrolangage de génération du système

inexistant

3.2. - Facilité d'utilisation du système

3.2.1. - Niveau requis par les utilisateurs du système

aucune compétence particulière

3.2.2. - Temps de formation de l'utilisateur

très rapide (2 ou 3 heures)

3.2.3. - Niveau requis par le responsable du système

niveau ingénieur

3.2.4. - Temps pour réaliser une application

Très variable suivant l'application. Il y a lieu de faire une analyse approfondie de la structuration des informations cela peut demander une dizaine de jours.

.../...

3.3. - Portabilité du système :

Peut être implémenté sur tout matériel ayant une taille mémoire suffisante.

3.4. - Comptabilité fichiers - langage de haut niveau

Accessibilité aux fichiers par langages de tous niveaux

IV - ORGANISATION DES FICHIERS :

4.1. - Structure logique :

4.1.1. - Types de structures logiques admis

- structures arborescentes à plusieurs niveaux
- structures de graphes (caractéristiques références)

4.1.2. - Organisation des fichiers :

4.1.2.1. - Dictionnaire des fichiers :

Les fichiers n'ont plus de signification logique, seules les adresses de mots sont prises en compte. Elle sont calculées par la gestion de mémoire.

4.1.2.2. - Descriptif d'un fichier :

Le fichier qui contient la structure est formé d'une suite de vecteurs de 15 mots mémoire contenant les renseignements nécessaires au calcul des adresses virtuelles des caractéristiques.

4.1.2.3. - Organisation des données :

A cause de la gestion de mémoire (transparente à l'utilisateur) la notion de fichier disparait dans SOMINE. Les données sont caractérisées par leurs adresses virtuelles (dans l'espace virtuel) et c'est la gestion de mémoire qui calcule les adresses réelles des données sur le disque.

4.1.3. - Corrélations

4.1.3.1 - Au niveau des fichiers :

Les adresses dans le fichier des données sont calculées à partir des

.../...

éléments contenus dans le fichier structure.

4.1.3.2. - Au niveau enregistrement :

Dans SOMINE, la notion d'enregistrement n'a pas la signification habituelle. Il faudrait plutôt parler de réalisations d'entités définies par la structure des informations.

4.1.4. - Langage de description

L.D.S. défini par des automates

4.2. - Structure physiques

4.2.1. - Support physique des données :

Les données sont placées sur des disques magnétiques

4.2.2. - Organisation physique des données :

Dans la mémoire virtuelle, pour chaque entité, on réserve une place correspondant au nombre maximum de réalisations. Seules les zones de mémoire virtuelle qui contiennent de l'information sont projetées sur l'espace disque. Cette projection se fait par blocs de longueur variable (de 2^3 à 2^8 mots) appelés sous-pages.

4.2.3. - Méthode d'accès :

- Accès séquentiel (limité à une seule réalisation d'entité)
- Accès par les caractéristiques inverses ou par les expressions booléennes de telles caractéristiques.
- Accès direct par le numéro de réalisation.

4.2.4. - Type d'adressage :

SOMINE calcule l'adresse réelle à partir de l'adresse virtuelle d'une caractéristique, il y a création de "squatters" (qui sont chainés entre eux) lorsque plusieurs adresses virtuelles donnent la même adresse réelle. La gestion de ces "squatters" est assurée par la gestion de mémoire.

V - FONCTIONS REMPLIES PAR LE SYSTEME :

5.1. - Interface de création

5.1.1. - Nature du processeur de création

La création se fait en "batch" il y a analyse de la structure des informations

écrite en L.D.S., cette chaine alphanumérique est ensuite transformée en vecteurs contenant les renseignements relatifs à chaque caractéristique et qui permettront de calculer son adresse virtuelle dans l'espace virtuel du fichier-données.

5.1.2. - Type de données admises :

. chaînes alphanumériques de longueur inférieure à 1280 caractères

. données numériques de type entier ou réel simple précision ou double précision.

. liste de valeurs

. chaine d'inverse

. pointeurs de référence

. programmes, valeur + programme, procédure

5.1.3. - Possibilité de codage et de décodage

Les données sont stockées dans leur état d'utilisation

5.2. - Consultation :

5.2.1. - Nature du processeur de consultation

Programmes Fortran d'analyse et d'interprétation des chaînes alphanumériques que constituent les requêtes. L'action à entreprendre est déterminée par le mode de la requête : interrogation, mise à jour, création, suppression, fréquence.

5.2.2. - Mode de consultation :

SOMINE permet de mode conversationnel (écran,) et le mode "batch".

5.2.3. - Multi-accès

possibles

5.2.4. - Consultation multi-fichiers

Possible

5.2.5. - Bibliothèque de questions

Chaque utilisateur peut créer son propre recueil de questions dans chacun de ses programmes d'utilisation.

5.2.6. - Extraction des fichiers

Très simple puisque se sont des fichiers FORTRAN à accès direct.

.../...

5. 3. - Mise à jour :

5.3.1. - Nature du processeur de mise à jour :

Tout programme en conversationnel ou en "batch" écrit dans un langage algorithmique.

5.3.2. - Mode de mise à jour :

Dans la version actuelle de SOMINE les mises à jour se font à l'aide de requêtes à l'écran ou en "batch"

5.3.3. - Mise à jour au niveau des enregistrements

Cette mise à jour se fait sans problème mais le nombre maximum de realisation d'une entité est limité à celui déclaré au moment de la définition de la structure.

5.3.4. - Mise à jour au niveau des informations élémentaires :

Se fait à l'aide de requêtes ponctuelles. Des précautions doivent être prises lorsque la base contient des caractéristiques programme (possibilité d'actions implicites)

5.3.5. - Mises à jour spéciales :

Il est possible de mettre à jour des caractéristiques filtrées par une caractéristique inverse ou une expression booléenne sur des caractéristiques inverses. Signalons également la possibilité d'effectuer des mises à jour groupées ou des mises à jour à valeur variable.

5.3.6. - Mise à jour de la structure logique des données

Semble nécessaire pour certains types d'applications des bases de données. Les travaux sont en cours.

5.4. - Possibilités de traitement :

SOMINE permet d'effectuer les requêtes de bases suivantes :

- interrogation
- mise à jour
- création
- suppression

.../...

- fréquence
- valeur

on peut facilement ajouter d'autres modes

5.5. - Editions :

5.5.1. - Nature du processeur d'édition

C'est l'utilisateur qui par une valeur donnée à une variable décide de la nature de l'édition :

- imprimante
- écran
- formatage (spécification FORTRAN)

5.5.2. - Support d'édition

- imprimante
- écran

5.5.3. - Tri

Peut être effectué par des instructions contenues dans le programme d'utilisation écrit en langage évolué.

VI - MAINTENANCE-SECURITE-PROTECTION-INTEGRITE

6.1. - Maintenance des fichiers :

6.1.1. - Réorganisation, rechargement des fichiers

Sauvegarde du système utilisé

6.1.2. - Statistiques d'occupation des fichiers

Non prévues dans le prototype

6.2. - Sécurité d'utilisation

6.2.1. - Possibilités de redémarrage du système

En cas de panne, les modifications effectuées dans les quatres pages (de 257 mots) qui se trouvent dans la mémoire centrale (plan primaire) sont perdues.

6.2.2 - Point de reprises

non

.../...

6.2.3. - Journal des munipulations effectuées sur les fichiers :

pourra être mis en oeuvre si des utilisateurs en sentent la nécessité.

6.3. - Protection :

6.3.1. - Identification de l'utilisateur :

Un code de passe peut précéder ou être contenu dans les requêtes et filtrer ainsi les entrées dans la base.

6.3.2. - Réglementation de l'accès à l'information

Un système analogue à l'identification de l'utilisateur peut être mis en oeuvre pour réglementer l'accès à l'information. SOMINE étant encore un prototype sa réalisation dépendra de la demande.

6.4. - Intégrité des fichiers :

Les références sont chaînées automatiquement mais c'est l'utilisateur qui s'assure de la cohérence de ses fichiers

VII - RENSEIGNEMENTS DIVERS :

7.1. - Encombrement des fichiers :

La notion de mémoire virtuelle entraîne un encombrement minimum des fichiers SOMINE

7.2. - Temps de réponse :

Variable suivant la nature du traitement. Dans le contexte actuel (multiprogrammation) il dépend de l'encombrement de la machine.

7.3. - Fiabilité :

Le produit est trop récent pour que nous donnions une réponse valable à cette question.

7.4. - Coût d'utilisation :

Par une bonne organisation du fichier-structure et une bonne formulation des requêtes, on doit arriver à une dépense de temps machine assez faible.