Une synthese des methodes et des outils d'aide a la conception de systemes d'information

M. Bouzeghoub

▶ **To cite this version:**

M. Bouzeghoub. Une synthese des methodes et des outils d'aide a la conception de systemes d'information. RR-0258, INRIA. 1983. inria-00076300

HAL Id: inria-00076300
https://inria.hal.science/inria-00076300

Submitted on 24 May 2006

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.

IRIA

CENTRE DE ROCQUENCOURT

Rapports de Recherche

original.

N° 258

UNE SYNTHÈSE DES MÉTHODES ET DES OUTILS D'AIDE A LA CONCEPTION DE SYSTÈMES D'INFORMATION

Mokrane BOUZEGHOUB

Décembre 1983

Institut National de Recherche en Informatique et en Automatique

Domaine de Voluceau
Rocquencourt
BP 105
78153 Le Chesnay Cedex
France
Tél. (3) 954 90 20

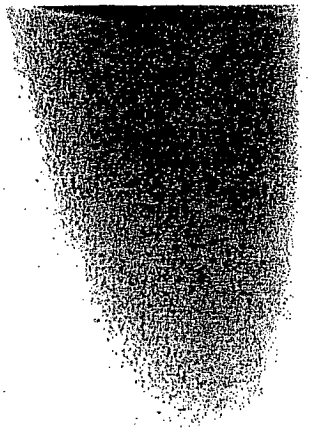
UNE SYNTHESE DES METHODES ET DES OUTILS

D'AIDE A LA CONCEPTION DE SYSTEMES D'INFORMATION

Mokrane Bouzeghoub

Projet SABRE INRIA Rocquencourt,
Institut de Programmation PARIS VI

PAPIER RÉCUPÉRÉ ET RECY



SOMMAIRE

RESUME

Ce rapport présente une synthèse critique des méthodes et des outils d'aide à la conception de systèmes d'information. La plupart des méthodes distinguent l'aspect statique représenté par une base de données et l'aspect dynamique représenté par une base de programmes. Chacun de ces aspects est décrit par les 3 schémas communément admis : conceptuel, externe, interne.

Au niveau conceptuel et externe, nous discuterons des modèles relationnels et sémantiques de données et des modèles de spécification des transactions inspirés des réseaux de Petri ou des types abstraits.

Au niveau interne, nous résumerons les paramètres essentiels à une bonne implantation d'une base de données. Ces méthodes seront illustrées par quelques exemples d'outils formels, automatiques ou manuels, d'aide à la conception.

MOTS-CLES

Modeles de donnés, conception logique, conception physique, systèmes d'information, spécification de transactions, outils d'aide à la conception.

ABSTRACT

This report presents a critical synthesis of the different methods and tool for information system design. All these methods distinguish between static aspects (data) and dynamic aspects (processing) which are both described by the well-known three schemas (conceptual, external and internal). At the conceptual and external levels, we especially focus on the relational and semantic data models and on the dynamic models inspired by Petri-nets or abstract data types.

At the internal level, we will summarize the main parameters for a good physical database design. Both the conceptual and internal levels will be illustrated by some formal, automatic or manual design tools.

KEY-WORDS

Data models, logical design, physical design, information systems, transaction specification, design aid tools.

1 - INTRODUCTION

Les modèles relationnels connaissent actuellement des applications de plus en plus nombreuses et un interêt de plus en plus croissant. Plusieurs prototypes de SGBD relationnels ont été développés sur toutes les gammes de materiel (du gros ordinateur au microprocesseur) [ASTR76, STON76, DEWI78, ...]. Certains produits connaissent même un débouché commercial. Les constructeurs et les societés de service n'hésitent plus à investir dans ce domaine.

La raison essentielle de cet essors tient au fait que les modèles relationnels sont bien les modèles qui assurent la plus grande indépendance des données par rapport aux applications tant recherchée depuis une douzaine d'années. Avec de tels modèles l'utilisateur peut desormais écrire ses applications sur des structures logiques sans se préoccuper de la réalisation concrète de ces structures [CODD70]. Cependant à chaque fois que ces produits sont évoqués on ne peut s'empêcher de penser au coût que cela occasionne. L'"overhead" subi par les utilisateurs est très important. Le choix des relations à stocker et l'optimisation des opérateurs relationnels semblent être les préoccupations essentielles des concepteurs de BD et de SGBD.

Pour réduire ces coûts, beaucoup de solutions sont proposées à different niveaux : algorithmes de placement des informations en mémoire secondaire, methodes d'accès sophistiquées, choix des chemins d'accès, optimisation des

requêtes, parallèlisation des opérations à un degré de plus en plus élevé, mémoires associatives, filtrage des données, etc...Le projet SABRE explore un large éventail de solutions dans ce domaine.

Parallèlement à l'evolution des logiciels, on assiste également à un foisonnement de methodes de conception de Bases de Données. Ceci se justifie au moins par deux raisons:

- La richesse des structures relationnelles conduit à une intégration de plus en plus grande des Systèmes d'Information dans l'organisation. De là, la conception de la BD qui represente ce SI devient une tâche de plus en plus longue et de plus en plus complexe. Sans outil methodologique, l'entreprise d'une telle tâche serait une gageure.
- Tenant compte des problèmes de coûts soulevés dans le paragraphe précédent, on se dit souvent que si on avait une conception "saine" de la BD, on reduirait sans doute de facon appréciable ces coûts.

Les recherches dans ce domaine se consacrent souvent a differents niveaux de perception (ou d'abstraction), à construire des modèles se voulant tantôt une representation fidèle de l'ensemble des informations circulant dans l'entreprise ainsi que leurs interactions, tantôt une representation fidèle du comportement de ces informations face aux modifications qu'elles subissent, ou enfin une synthèse des deux. Ces methodes s'accordent toutes actuellement à distinguer l'aspect statique (les données) et l'aspect dynamique (les traitements). Chacun de

ces aspects est decrit par les trois schémas communément admis: conceptuel, externe, interne [ANSI75]. Ces trois schemas se situent à deux niveaux d'abstraction differents:

- Au niveau conceptuel, on trouve :
  * le schéma global (ou schéma conceptuel),
  * les shémas partiels (ou schémas externes ou vues des utilisateurs).
- Au niveau interne, on trouve :
  *le schéma logique qui est une transformation des schémas conceptuel en tenant compte d'une part des orientations organisationnelles dictées par l'entreprise (tâches manuelles, tâches automatiques, batch, temps reel, repartition...), d'autre part des contraintes technologiques générales (utilisation de SGF, SGBD de type Codasyl ou Relationnel...).
  *le schéma physique qui précise l'organisation des ressources utilisées, le stockage physique des données, les méthodes d'accès, les transactions...

Enfin l'ensemble des méthodes offrent une série d'étapes et de repères materialisant le processus de conception, et l'idée d'outil automatique d'aide à la conception commence à se généraliser. La plupart des méthodes tendent vers le schéma de conception represenré fig.1.

Dans le deuxième chapitre de ce rapport, nous passerons en revue les differents modèles de description du niveau conceptuel tant en ce qui concerne les données qu'en ce qui concerne les traitements. Seront également résumés dans ce

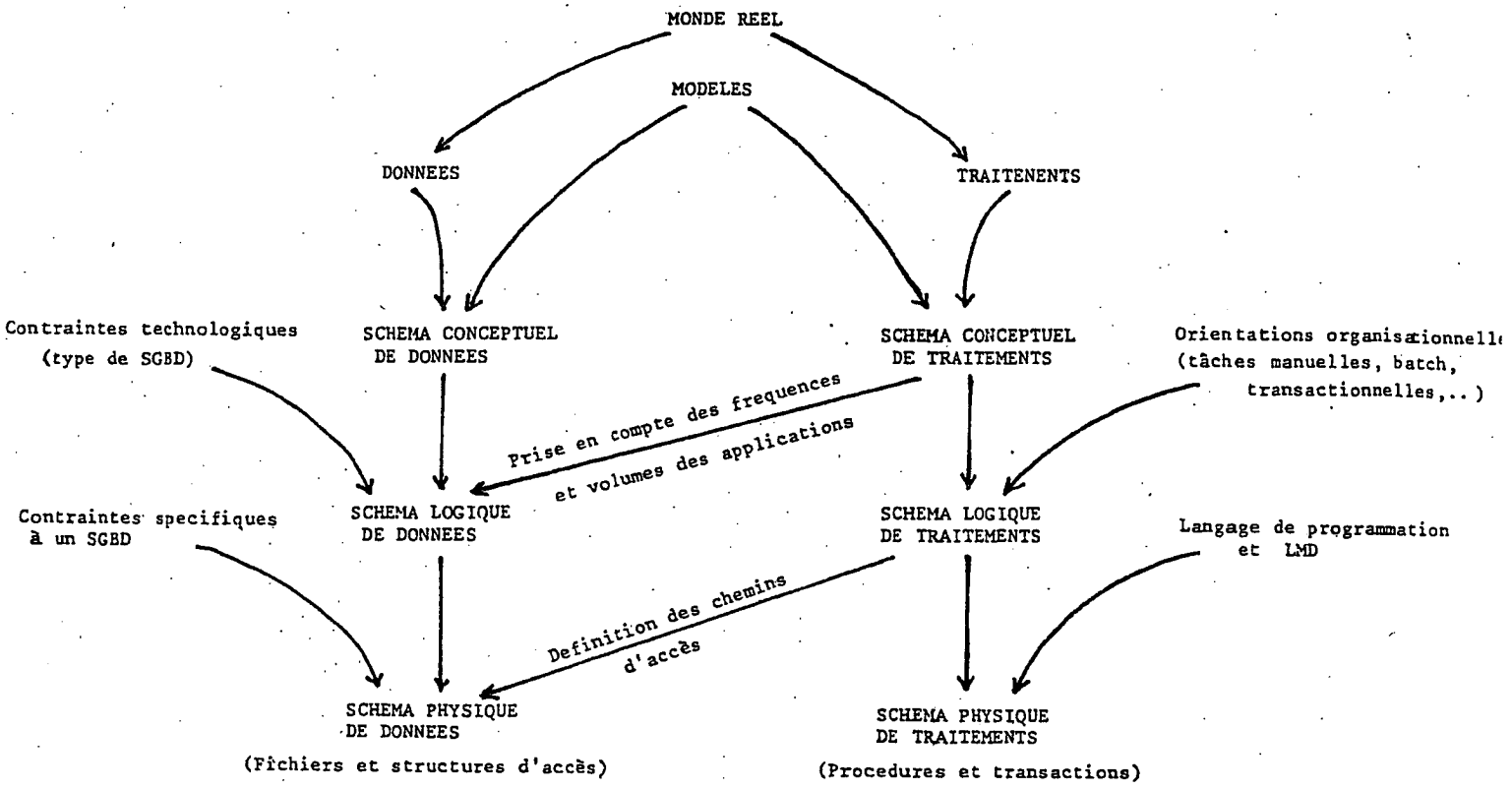


Figure 1 : Schema de synthèse des méthodes de conception

chapitre quelques outils d'aide à la conception.

Le chapitre 3 est un résumé des differents modèles d'évaluation des structures de données interne. L'accent sera mis sur le placement des données et le choix des chemins d'accès.

Enfin, en conclusion, nous résumerons les problèmes soulevés par la conception des SI et les tendances actuelles pour les résoudre.

2- LE NIVEAU CONCEPTUEL

2-1 DESCRIPTION DES DONNEES

2-1-1- PRINCIPAUX MODELES

Plusieurs formalismes sont proposés pour décrire les données, nous résumons ci-dessous leurs caractéristiques.

a) Le modèle relationnel de Codd

Avec la possibilité d'exprimer des relations n-aires et la puissance de son langage de manipulation, le modèle de CODD est plus proche de l'utilisateur bien qu'il souffre de l'absence d'une représentation graphique adéquate. Il faut noter aussi que la recherche des dépendances fonctionnelles et le processus de normalisation sont de complexité exponentielle à la quantité d'informations à structurer. Un des problèmes posés aussi par ce modèle est la prise en compte des valeurs nulles. Ce problème est ressenti de facon differente au niveau conceptuel et au niveau interne. Au niveau interne il est perçu comme un problème de décidabilité (on ne sait pas faire la jointure de deux valeurs nulles [CODD79]) ou de risque d'anomalies (en suppression par exemple). Au niveau conceptuel il apparait

- D'une part comme un problème engendré par la classification des objets en types: à vouloir construire des types trop "larges" il y a risque de regrouper dans une même classe des objets ne se decrivant pas exactement de la même facon (les détails les distingant sont ignorés par un niveau d'abstraction trop élevé).
- D'autre part comme un problème inhérent au traitement de

l'information: la connaissance et l'introduction des valeurs de certains attributs est differée (par exemple on ne connait pas l'adresse de tel employé, on l'introduira plus tard).

Le premier cas est résolu dans certaines méthodes par la définition même des types d'entités et de relations ("un type d'entité ou de relation est doté des mêmes proprietés sur toute l'étendue du reel" [TARD76], ce qui signifie que tous les attributs décrivant un objet doivent avoir un sens pour toutes les occurrences de cet objet.). Ceci introduit au niveau de la méthode une étape de verification s'assurant que pour toutes les occurrences du type il existe une valeur connue ou inconnue mais pas impossible pour chaque attribut.

Le second cas semble plus difficile à traiter et cependant fort important pour l'utilisateur. Des méthodes sont proposées pour le prendre en charge au niveau interne [CODD79, GOLD81, LEON81].

ETUDIANT(NUM_ETUD, NOM_ETUD, PREN_ETUD, ADR_ETUD)
ENSEIGNANT(NUM_ENSEIGN, NOM_ENSEIGN, PREN_ENSEIGN, ADR_ENSEIGN)
COURS(CODE_UV, NUM_TD, HEURE, SALLE, NUM_ENSEIGN)
INSCRIPTION(NUM_ETUD, CODE_UV, DATE_INSCRIPT, NUM_TD)
MODULE(CODE_UV, NOM_UV)

Figure 2: Un exemple de schéma relationnel.

b) Les modèles Entité/Association (E/R)

Les modèles Entité/Associations (E/R) sont sans doute ceux qui sont les plus proches de l'utilisateur. Avec des concepts simples, facilement assimilables, ils ont pour souci de

décrire aussi fidèlement que possible les objets manipulés dans l'organisation. Leur représentation graphique est d'une aide inestimable à la reflexion.

Ce type de modèles se propose de decrire le monde reel en faisant trois types de classifications des objets de ce reel: type d'attribut, type d'entité et type d'association. Les objets d'une même classe se caracterisent par le fait qu'ils se décrivent tous de la même facon (ou ce qui revient au même, ils prennent tous leurs valeurs dans le même domaine). Mais les notions d'objet-type et d'occurrence de type sont toutes relatives, et selon le niveau où l' on se place un type peut devenir occurrence et une occurrence peut devenir type [KENT81]. Par exemple, les occurrences de l'objet type OUVRAGE peuvent être des titres d'ouvrage (auquel cas il n'y a qu'un seul exemplaire d'un titre) ou des côtes d'ouvrage (auquel cas il peut y avoir plusieurs exemplaires d'un même titre) ou une confusion des deux (cas ou l'on veut à la fois les titres et les côtes). Reste donc à savoir s'il est interessant et possible de définir une hierarchie de types. Par exemple OUVRAGE dont les occurrences sont des titres qui sont à leur tour des types dont les occurrences sont les documents physiques identifies par leurs côtes.

Les mêmes remarques sont à faire lorsque dans une même classe d'objets, on distingue des partitions. Par exemple dans la classe d'objets OUVRAGE , on peut distinguer des ouvrages de mathématiques, de statistiques, de Bases de Données, de théories

des langages ; etc... L'interêt d'un tel partitionnement est souligné chez plusieurs auteurs et notamment dans [SMIT78] et [BROD81] qui proposent pour le décrire les concepts de Généralisation et de Spécialisation, agrégation et association (cf. paragraphe sur les modèles sémantiques).

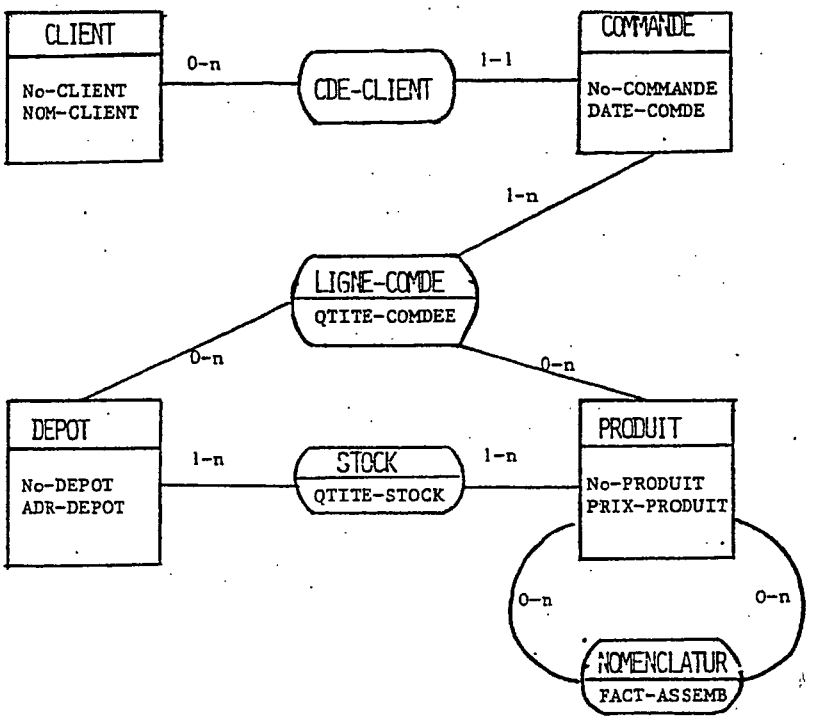


Figure 3: Un exemple de schéma entité/association

Un point noir dans le modèle E/R, c'est l'absence de langage de manipulation bien formalisé et adapté au modèle. Autant au niveau des concepts, le modèle a reçu le consensus de la plupart des utilisateurs, autant au niveau du langage les quelques propositions qui ont été faites [CHEN77, SHOS77, ATZE81] font encore l'objet de reflexion. La plupart s'accommodent de

cette lacune en utilisant les langages relationnels, la transformation d'un modèle E/R en modèle de CODD étant simple (les entités et associations du modèle E/R sont des relations 3FN ou 2FN du modèle relationnel). On considère en fait que le modèle E/R et le schéma relationnel de CODD sont hierarchiquement l'un au dessus de l'autre dans les niveaux d'abstraction; on construit un schéma E/R qu'on transforme en schéma relationnel sur lequel on exprime les requêtes. On profite ainsi de la simplicite descriptive de l'un et de la puissance du langage de l'autre.

c) Les modèles binaires

Ces modèles, comme le modèle relationnel de Codd, se proposent de décrire le monde reel avec le concept de relation. Mais Coid s'appuie sur le concept de table pour illustrer ses relations; représentation élégante mais sémantiquement pauvre. Les modèles binaires viennent combler cette lacune en définissant une relation par deux fonctions inverses l'une de l'autre. Nous allons présenter ici le modèle binaire d'Abrial. Ce modèle est basé sur deux concepts:

- Le concept de catégorie qui fait une classification d'objets en types.
- Le concept de relation binaire entre deux catégories et définie par deux fonctions inverses l'une de l'autre qui précisent la sémantique du lien qui relie les deux catégories. Ces fonctions ne sont pas des fonctions au sens mathématique du terme car elles peuvent produire plusieurs valeurs, on les

appelle des fonctions d'accès. Chaque fonction est caractérisée (comme l'association dans le modèle E/R) par un couple de valeurs qui sont les cardinalités minimum et maximum de la fonction. Ce couple de valeurs précise pour une occurrence de catégorie le nombre minimum et le nombre maximum de valeurs que peut produire la fonction d'accès définissant la relation à laquelle participe cette categorie. Un modèle binaire est représenté par un graphe dont les sommets sont des catégories et les arcs des relations entre catégories (voir fig.4).

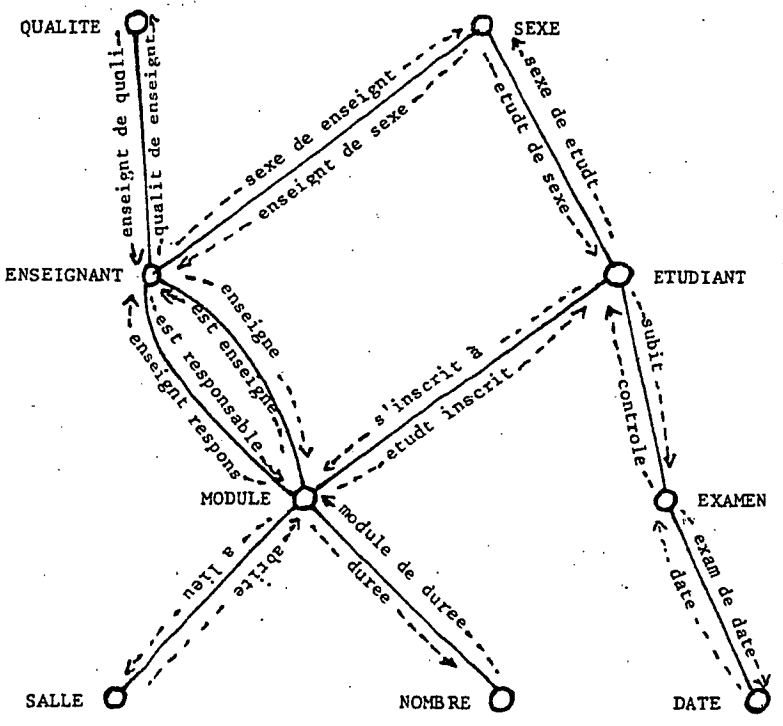


Figure 4 : Un exemple de schema binaire

Il faut noter qu'avec les deux fonctions d'accès qui définissent une relation, ce modèle précise exactement le sens d'une relation. Du point de vue interne, chaque fonction sera representée par une procédure de contrôle incluant toutes les contraintes d'integrités que doit verifier la relation lors de

chacune des manipulations dont elle fait l'objet. Le langage de manipulation associé à ce modèle est un langage procédural disposant d'instructions de création et d'affectation de noms a des catégories et à des occurrences de catégories, de connexion et de deconnexion de deux occurrences de catégories au travers d'une fonction d'accès, de test d'appartenance d'un objet à une catégorie ou à une relation...

d) Les modèles sémantiques

Ces modèles sont des extensions des modèles binaires enrichis par certains concepts de telle sorte qu'ils puissent représenter certains aspects de la connaissance que les modèles vus précédamment ne savent pas représenter. Ces aspects regroupent essentiellement les points suivants:

- Le même modèle doit pouvoir représenter des types et des occurrences de types sur un même schéma.
- On doit pouvoir distinguer plusieurs partitionnements dans un même type (cf Généralisation Spécialisation chez [SMIT77]). Ce partitionnement va permettre des déclarations de vues non seulement au niveau des types mais aussi au niveau des valeurs; ce qui est realisé traditionnellement en deux temps dans les modèles CODASYL: définition du sous-ensemble de schema desiré et selection de valeurs dans ce sous-ensemble.
- La notion de type est relative au niveau d'abstraction auquel on se refère. Selon ce niveau, ce qui est considéré comme type peut devenir occurrences et inversement. Le modèle sémantique

doit donc représenter cette hierarchie de types en distingant entre classifications qui sont des constructions de types de premier niveau (obtenus à partir des occurrences) et généralisations qui sont des abstractions de types à plusieurs niveaux [SMIT77].

Exemple:

Classifications: ETUDIANT

ENSEIGNANT

Généralisation : PERSONNE

L'entite PERSONNE est caracterisée alors par toutes les proprietés communes a ETUDIANT et ENSEIGNANT.

- Enfin le modèle doit pouvoir spécifier pour chaque objet manipulé (entité, relation, attribut) le rôle qu'il joue en tant que acteur dans le modèle, et ceci à tous les niveaux d'abstraction.

On peut définir la structure d'un modèle sémantique comme un graphe de type réseau dont les noeuds et les arcs traduisent une certaine sémantique. Les differents modèles sémantiques se ditinguent entre eux d'une part par la richesse de cette sémantique d'autre part par la facon dont elle est exprimée. Les noeuds de ces graphes sont consideres comme des objets (valeur attribut occurrence entité) et les arcs comme des assersions entre ces objets (relation prédicat). Cependant il n'est pas toujours aisé de décider si une information doit être representée par un noeud (objet) ou un arc (assertion). La seule idée directrice reste, comme dans le modèle E/R, à considerer

comme noeud toute information susceptible d'être reliée à d'autres informations.

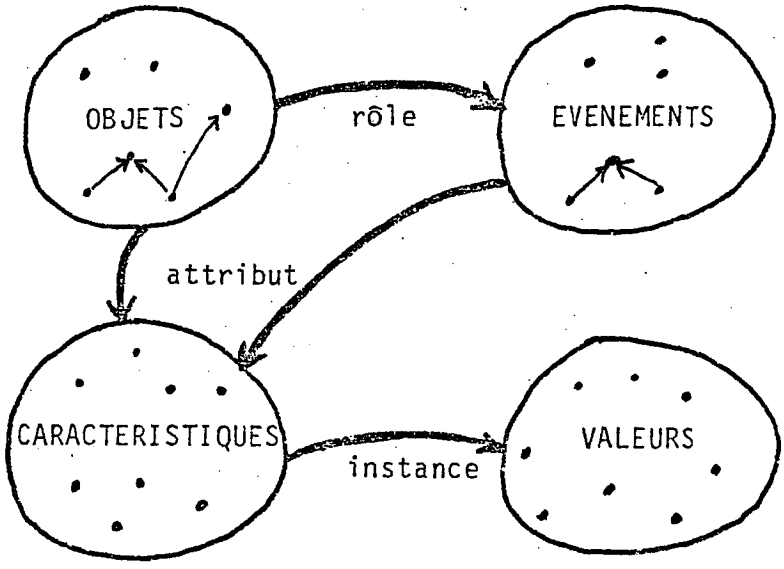


Figure 5: Categories de noeuds et d'arcs

[MIL076] et [LEV79] identifient 4 catégories de noeuds:

- Les concepts ou objets,
- Les événements,
- Les caracteristiques ou attributs,
- Les valeurs,

et deux catégories d'arcs:

- Ceux reliant les catégories de noeuds,
- Ceux reliant les membres d'une même catégorie de noeuds

La fig.5 illustre les catégories de noeuds et d'arcs.

(1) Les catégories de noeuds

- Les concepts: Ce sont des paramètres constants de l'application qu'on veut représenter. Ces paramètres sont classés en types à differrents niveaux d'abstraction: au premier niveau

on trouve les jetons (ou concepts de base), au second niveau on trouve les types classe, au troisième niveau et aux niveaux supérieurs les métaclasses (type de type).

- Les évenements: Ce sont les actions de l'application a représenter, c'est ce qui constitue la dynamique de l'objet a représenter. Cette catégorie de noeuds peut être classée en types comme pour les concepts.

- Les caracteristiques: Ce sont des propriétés decrivant des concepts ou des évenements.

- Les valeurs: Ce sont les domaines de valeurs des caracteristiques précédentes.

(2) Les catégories d'arcs

- Arcs reliant deux noeuds de deux categories differentes:

* Arc reliant un évenement et un concept: exprime le rôle joue par le concept invoqué par l'évenement.

*Arc reliant une caracteristique et un concept: exprime qu'une caracteristique décrit un concept.

* Arc reliant une caracteristique et une valeur: pointe la valeur d'un attribut.

- Arcs reliant deux membres d'un même catégorie:

*Arc reliant deux jetons d'une categorie (concepts ou événements): exprime une assertion entre ces jetons.

* Arc reliant deux classes ou métaclasses d'une même catégorie de noeuds (concepts ou événements): traduit une relation binaire entre ces classes.

* Arc reliant un jeton à une classe: exprime que le jeton est une réalisation de la classe

* Arc reliant une classe à une metaclasse: traduit le fait que la classe résulte du partitionnement de la métaclasse.

Dans [SMIT77] les jetons sont appelés des instances, les classes sont appelées des agrégats et les métaclasses des generiques. En ce qui concerne les arcs, [SMIT77], ne prenant en compte qu'une seule categorie d'objets (les concepts), il ne distingue que les trois derniers.

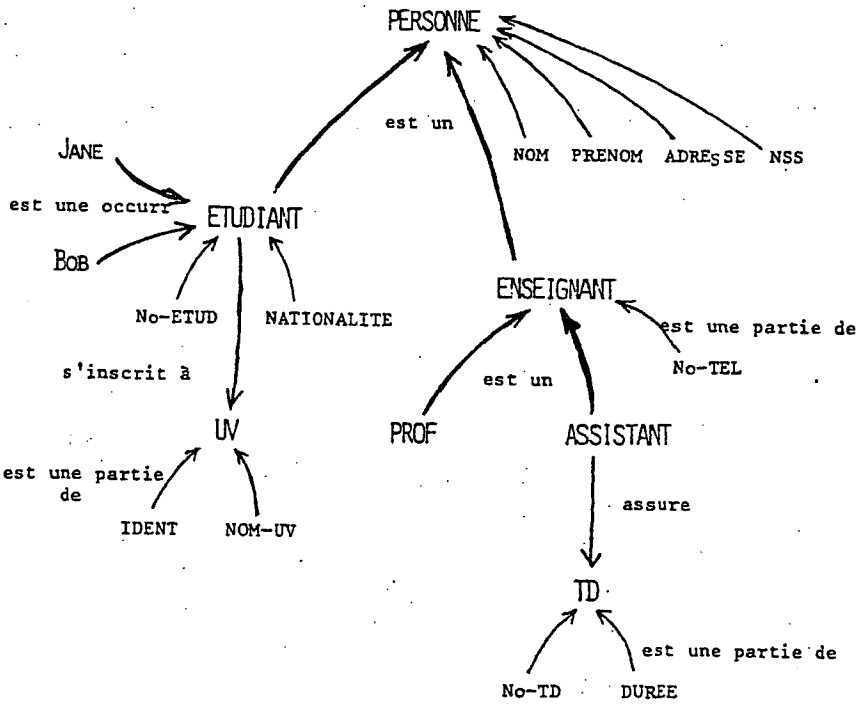


Figure 6: Un exemple de reseau semantique

e) Les modèles fonctionnels.

Ces modèles sont inspirés de la programmation fonctionnelle proposée par BACKUS [BACK78#] ainsi que par les concepts des langages applicatifs tels que LISP et APL. Plusieurs chercheurs en Bases de Données ont alors proposé des modèles dits fonctionnels pour représenter et manipuler des bases de données. Contrairement aux autres modèles où les concepts de description et ceux de manipulation sont distincts, langage et concepts sont intimement imbriqués dans les modèles fonctionnels. Nous nous refererons, dans la suite, aux travaux de BUNEMAN et SHIPMAN sur FQL [BUNE82] et DAPLEX [SHIP81], deux modèles fonctionnels.

Ces modèles suggèrent que la plupart des objets et des relations entre ces objets peuvent être representés comme des fonctions mathématiques définies sur des domaines de base ou des domaines construits. Ainsi, les objets PEERSONNE et NOM, au lieu d'être définis comme entité et attribut de cette entité, réspectivement, seront declarés comme des fonctions. Dans DAPLEX [SHIP81], ces déclarations sont de la forme:

DECLARE PERSONNE() =====>> ENTITY

DECLARE NOM(PERSONNE) =====> STRING.

ENTITY et STRING sont des domaines de base, PERSONNE et NOM sont des fonctions qui, à leur tour, peuvent être considérées comme des domaines construits sur lesquels d'autres fonctions peuvent être définies, etc... Par exemple:

DECLARE ETUDIANT() =====>> PERSONNE.

PERSONNE et ETUDIANT sont des fonctions zéro-adic alors que NOM est une fonction mono-adic. PERSONNE represente un ensemble d'entités (double fleche), NOM represente une chaine de caractères.

Dans FQL [BUNF82], les déclarations précédentes s'écrivent à peu de choses près de facon similaire:

PERSONNE: -------> !ENTITY

NOM: PERSONNE -------> STRING

ETUDIANT: -------> PERSONNE.

!ENTITY désigne un ensemble d'entités. Les fonctions zero-adic n'ont pas de domaine de départ. Une requête FQL peut avoir la forme suivante:

!PERSONNE.*NOM;

où *NOM désigne une extension des noms de personnes. Le point est mis pour une composition de fonctions. La requête suivante

!PERSONNE.|MARIE.*NOM

donne l'ensemble des noms de personnes mariées, en supposant que MARIE avait été précédemment défini comme suit (fonction boolèenne):

MARIE: PERSONNE --------> BOOL

Dans DAPLEX, la première requête aurait la forme suivante:

For each PERSONNE print NOM(PERSONNE)

et la seconde:

```
For each PERSONNE
    such that MARIE(PERSONNE) = True
    print NOM(PERSOONE).
```

Ces deux instructions montrent que DAPLEX a imbrique ses fonctions dans un langage procédural alors que FQL reste déclaratif.

Ces deux applications de la programmation fonctionnelle aux bases de données permettent à leurs auteurs de redéfinir une base de données comme une collection de fonctions primitives et de formes fonctionnelles (composition de fonctions) [BUNF82]. Il faut également remarquer que ces fonctions ne sont pas standards, elles sont définies par l'utilisateur (administrateur) et dépendent de chaque application.

f) Le modèle infologique

Cette section est un resumé du chapitre 11 de [TSL081] consacré au modèle infologique et de l'article de LANGEFORS sur ce modèle [LANG80]. Ce modèle, bien que peu répandu, contient des idées et des concepts originaux sur le niveau conceptuel. C'est pour ces raisons que nous le rappelons ici.

Cette approche préconise deux niveaux d'abstraction dans la description de systèmes d'information:

- le niveau infologique et
- le niveau datalogique.

Tous les modèles de description existant actuellement décrivent, pour la plupart, le second niveau. Quelque soit le degré d'abstraction atteint par ces modèles et malgré les bonnes intentions de leurs promotteurs de leur conferer des concepts et des structures indépendantes de la technologie, ces modèles sont

tous plus ou moins lies à la structure de la mémoire de l'ordinateur.

Dans le monde reel, il y a des objets qui sont des entités ou des phénomènes sur lesquels nous voulons avoir de l'information. Cette information correspond à la perception que l'on a du monde reel considéré. Pour représenter cette perception, on suppose que les "faits élémentaires" suffisent. Le problème d'abstraction revient donc à representer ces faits élémentaires. Pour mieux décrire ces faits, on s'inspire de la manière dont l'homme se les représente dans son environnement. Les gens décrivent les faits élémentaires dans leur langage naturel en termes d'objets (ou de types d'objets), de proprietés de ces objets (ou des relations liant ces objets) et le temps d'apparition de ces objets.

Un objet est défini comme quelque chose d'interessant à representer. Un objet ne peut être précisement défini tant qu'on ne connait pas ce qui le constitue. De plus, il faut tenir compte de la relativité de la perception, ce qui peut paraitre un objet pour une personne ne l'est pas pour une autre. Cette relativité dépend de l'interêt qu'on porte à l'objet en tant que tel ou en tant que structure. L'existence d'un objet est liée à sa naissance, sa mort et ses diverses modifications dans le temps. Les objets naissent losqu'on s'y interessent, meurent lorsqu'on se desinteressent et se modifient lorsque leur état precedent est suffisamment different pour qu'on les voit differents.

Dans le modèle infologique, un objet peut exister

indépendamment de ses proprietés ou des relations qui le relient aux autres objets (contrairement aux autres modèles où l'existence d'un objet dépend de l'existence d'une valeur pour sa clé). Les seules proprietés liées à l'existence d'un objet sont sa date de naissance et sa date de disparition.

Un fait élémentaire est défini par le triplet (x, y, z) où x represente un tuple d'objets, y represente la propriete (relation) de cet objet et z le temps (sa naissance et sa disparition). Un tel triplet est appelé "constellation élémentaire".

Si x est un objet élémentaire et y sa propriete, ce triplet est appelé "constellation élémentaire d'une proprieté-type". Par exemple, une certaine personne (x) est malade (y) à un certain temps (z). Si x est un tuple d'objets (o1, o2, ...on) et y une relation, ce triplet est appelé "constellation élémentaire d'une relation-type". Les objets peuvent être composés de tuples d'objets, d'ensembles d'objets ou de constellations.

Un groupe d'objets O(p) lié à une propriete p est défini comme tous les objets ayant, en negligeant le temps, la propriete p. Ceux qui satisfont la propriete p à un instant t constituent un sous-ensemble du groupe d'objets appelé "tranche de temps du groupe d'objets".

Dans les autres modèles, un attribut est défini de facon générique (information de base ne pouvant être définie que par rapport à elle-même). Dans le modèle infologique, un attribut

est défini comme un ensemble de proprietés A={pi} d'un groupe d'objets O(p) tel que à chaque instant t, chaque objet x de Ot(p) soit contenu au moins dans un Ot(pi) pour un pi donné. Les proprietés pi sont des valeurs de l'attribut A du groupe d'objets O(p).

Soit un groupe d'objets O dont les éléments ont une certaine propriété p à iifferents instants, et un attribut A, le couple (O, A) est appelé un "attribut de type constellation élémentaire". Cet attribut correspond à la définition classique d'un attribut dans les autres modèles, les proprietés pi formant le domaine de A.

Une occurrence d'une constellation élémentaire (o, p, t) d'une propriété-type correspond à un attribut de type constellation élémentaire (O, A) si o appartient a O et p appartient à A.

Une "relation de type constellation élémentaire" est un couple ((O1, ..., On), R) où les Oi sont des groupes d'objets et R une relation n-aire entre ces objets. Cette relation ainsi définie correspond tout à fait à la relation definie par CODD dans le modèle relationnel. Une occurrence d'une constellation élémentaire de relation-type ((o1, ..., om), r) correpond à une relation de type constellation élémentaire si oj appartient à Oi et r=R.

L'identification interne des objets se fait grâce à une propriété interne appelée clé dans la plupart des modèles. L'existence de l'objet est alors conditionné à l'existence d'une

valeur pour cette clé. Mais cette solution possède au moins deux inconvénients:

- Le changement d'une valeur de cette clé correspond à la création d'un nouvel objet et à la suppression de l'ancien. Un historique sur ces valeurs est difficile à maintenir.
- On ne peut pas faire référence à un objet par plusieurs propriétés differentes.

Dans le modèle infologique, on s'interesse à l'identification des constellations plutôt que des objets. L'identification d'une constellation se fait par le triplet (x, y, z) où x represente les tuples de référence unique à des objets, y les relations entre références et z la date de référence. Cette constellation qui permet l'identification d'autres constellations est appelée "message élémentaire". Chaque identifiant peut être considéré comme une requête: 'quels sont les objets x ayant la propriété y à l'instant z'. Un message élémentaire est dit incomplet si l'une de ces references n'est pas unique. Ces messages peuvent être regroupés en classes de la même façon que les constellations.

2-1-2- DEMARCHE POUR LA STRUCTURATION DES DONNEES

Dans les approches relationnelles on distingue globalement deux demarches : celle qui part des informations élémentaires et des dépendances fonctionnelles pour deriver le schéma conceptuel et celle qui part des objets plus ou moins

structurés existant dans le monde reel pour déduire ce schéma.

La première plus algorithmique est fastidieuse. La seconde plus pragmatique est basée sur des choix subjectifs. La première tend vers une unicité de la perception, elle est basée sur la mise en évidence (par les DF) d'une structure préexistante et une fois les DF retrouvées tout le modèle peut se décrire par une grammaire. La seconde pose le problème de l'indeterminisme dans la modèlisation [KENT81], plusieurs facons de représenter les mêmes objets impliqués dans la même réalité (choix des entités, des relations...), mais les relations décrites sont plus explicites et d'une richesse sémantique superieure à celle décrite par les DF.

Dans la première approche on ne considère un schéma conceptuel comme propre que lorsqu'il est normalisé 3FN au moins [CODD70, BERN76, FAGI77]. Dans la seconde approche, pour choisir une representation, on utilise souvent des critères de performance qui ne relèvent pas du niveau conceptuel. Ce soucis de cohérence et d'optimisation ne devrait pas en effet relever du niveau conceptuel mais du niveau interne. Au niveau conceptuel on devrait mettre en evidence plus la sémantique du modèle que ses performances.

2-1-3- UN REGARD SUR LES VUES

L'interêt des vues par rapport au shéma global est multiple. D'une part, les vues permettent aux utilisateurs de ne

considerer que la partie du modèle qui les interesse les débarassant ainsi de tout ce qui peut les encombrer inutilement. Elles lui permettent aussi d'exprimer sa propre perception des objets qu'il manipule. D'autre part les vues assurent une partie de la confidentialite de la BD.

Il existe plusieurs méthodes d'elaboration des vues et de leur intégration avec le schema global:

- On construit d'abord toutes les vues connues, on fait ensuite une intégration pour en déduire le schéma global.
- On construit d'abord le schéma global puis on derive des vues au fur et à mesure des besoins de ce schema global.
- On fait une conception parallèle des vues et du schéma global et on s'assure que l'ensemble s'intégre.

Une discussion des deux premières approches est faite dans [ATZE81]. Il faut ajouter seulement les remarques suivantes:

(1) La première approche a l'avantage d'être simple et l'inconvénient d'être fermée. En effet une fois l'integration faite et le système en marche, il serait très difficile d'intégrer d'autres vues; c'est alors une perpetuelle remise en cause et restructuration du schéma.

(2) La deuxième approche a l'inconvenient d'obliger les utilisateurs à avoir la même perception du monde reel, c'est a dire à percevoir les mêmes objets que ceux définis dans le modèle global. Ce qui peut avoir comme conséquence une certaine distorsion de la réalité (parce que on n'aurait pas trés bien saisi la signification de chaque objet).

(3) La troisième approche a l 'avantage de supprimer les problèmes précédents et l'inconvénient de la complexite des règles d'intégration. Mais il nous semble plus interessant d'orienter les recherches dans cette direction, en s'aidant éventuellement d'outils informatiques pour réduire la complexite du problème. De plus l'intégration de schémas conçus de manière complétement autonome est une bonne validation de la pertinence de la perception qu'on a de l'organisation [TARD76].

2-2- DESCRIPTION DES TRAITEMENTS

De nombreux formalismes sont proposés actuellement pour décrire les traitements (ou ce qu'on appelle aussi la dynamique des SI). Certains sont des extensions des formalismes de données [ATZE81, BROD81], d'autres proposent de nouveaux concepts [HECK80, ROLL81]. Les réseaux de Petri attirent aussi une attention toute particulière, mais les idées ne se sont pas encore stabilisées dans ce domaine dont l'interêt n'a attiré l'attention des chercheurs que depuis trois ou quatre ans. Les approches et les formalismes diffèrent par les objectifs que se proposent d'atteindre les concepteurs : observer les états successifs du système, décrire les transitions elles-mêmes ou les opérations, etc... Nous allons présenter trois approches differentes pour modèliser la dynamique :

- l'adaptation des réseaux de Petri dans MERISE [HECK80] ;
- l'expression de la dynamique dans le projet REMORA [ROLL81] ;

- l'expression de la dynamique par les types abstraits [BROD81].

2-2-1 L'ADAPTATION DES RESEAUX DE PETRI DANS MERISE

Pour décrire l'activite d'un ou plusieurs processus, on utilisait déjà depuis plusieurs années les Réseaux de Petri, mais cette utilisation s'est orientée plus particulièrement vers certains algorithmes systèmes de partage de ressources, de parallèlisme et de synchonisation. De ce fait, les Reseaux de Petri sont restés l'outil de reflexion et de démonstration de quelques spécialistes uniquement. Cependant leur base théorique et leur commodité de représentation graphique devaient permettre leur utilisation au niveau conceptuel. Ainsi, il s'agissait de rapprocher les concepts utilises dans les RdP (place, transition, jetons, declenchement...) des objets habituellement manipulés dans l'organisation (message, évenement, opération, résultat...). C'est l'orientation prise dans la méthode MERISE [HECK80].

Pour modèliser la dynamique des SI, MERISE utilise une approche permettant de décrire à la fois les états successifs du SI et le sequencement des transitions permettant le passage d'un état à un autre. Le formalisme utilisé est le suivant:

- Les concepts de base:

EVENEMENT: C'est la traduction du fait que quelque chose est arrivée soit de l'exterieur vers le SI (evenement externe), soit d'une partie du SI vers une autre partie (événement interne).

RESULTAT: ensemble d'informations déduites en réaction à un

évènement. Ce résultat peut agir à son tour comme évènement interne.

OPERATION: action ou ensemble d'actions effectuées par le SI en réaction à un évenement. Elle produit un ou plusieurs résultats selon une condition R.

SYNCHRONISATION: c'est une pré-condition pour le déclenchement d'une opération. Elle est exprimée par une expression booleène S dont les variables sont des évènements nécessaires au déclenchement d'une opération.

Par analogie avec les Réseaux de Petri, les évènements et résultats correspondent à des places, les opérations à des transitions et les synchronisations à des conditions de déclenchement. Les occurrences d'évènement sont des jetons.

- Les concepts associés: Pour mieux préciser la sémantique du modèle dynamique, on ajoute aux concepts de base précédants les contraintes suivantes:

CARDINALITE d'un résultat: c'est le nombre d'occurrences identiques produites.

Nombre de PARTICIPATIONS d'un évènement à une synchronisation: c'est le nombre d'occurrences distinctes du même évènement participant à une synchronisation.

CAPACITE d'un évènement/résultat: c'est le nombre maximum d'occurrences distinctes.

DUREE de CONTRIBUTION: c'est le délai au delà duquel l'évènement attendu ne presente plus aucun interêt s'il n'est déjà arrivé.

DUREE d'une OPERATION: durée comprise entre le déclenchement de l'opération et la production de résultats.

- Les règles de vérification concernent:

. La syntaxe du modèle,

. La coherence avec le modèle de données: toute information du modèle de traitement est une information des modèles externes de données.

. Le fonctionnement du modèle de traitements: ce sont toutes les règles de verification du fonctionnement d'un réseau de Petri (conflit, cycle, atteignabilite...).

On dit qu'un modèle dynamique est globalement verifié s'il est

(1) syntaxiquement correcte,

(2) cohérent avec les modèles de données,

(3) bien formé pour tous les états initiaux prévisibles.

Remarque : L'exemple traité figure 7 modélise les demandes de réservation et annulation de chambres d'hôtel. Il est inspiré des exemples traités figures 8 et 10 qui sont tirés de [ROLL81] et [BROD81].

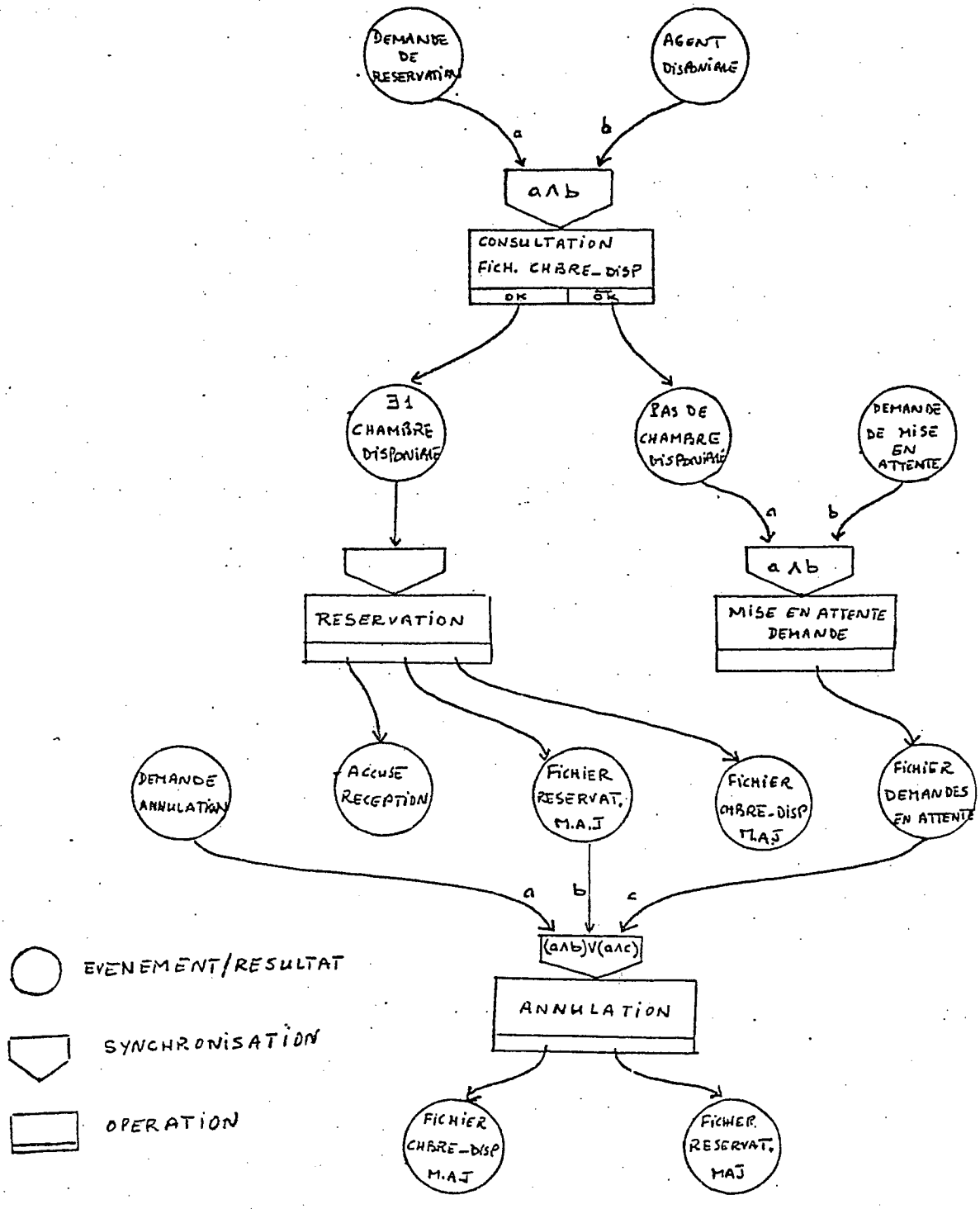


Figure 7: Un exemple de schema dynamique MERISE

2-2-2- L'EXPRESSION DE LA DYNAMIQUE DANS LE PROJET REMORA

Le projet REMORA [ROLL81] adopte une approche structurelle offrant un schema incluant à la fois l'aspect statique et l'aspect dynamique des SI. L'originalitè de la mèthode consiste en la classification des informations en donnèes permanentes et donnèes temporelles. Une date est associèe à ces dernières permettant ainsi de conserver dans la base d'informations tous les états successifs du SI. Cette approche s'appuie sur les trois concepts d'objet, d'évènement et d'opèration por dècrire le SI.

OBJET: c'est un composant concret ou abstrait de l'organisation, qui est durable et qui peutêtre particularisè.

EVENEMENT: c'est tout ce qui peut survenir à un instant donnè. C'est la constatation d'un changement d'état d'un ou plusieurs objets par l'éxecution d'une opération.

OPERATION: c'est l'expression de la plus petite transformation qui peut arriver à un objet. Elle peut modifier un ou plusieurs objets.

Le niveau conceptuel est décrit en termes de catégories d'objets (C-Objet), de catégories d'évènements (C-Evènement) et de catégories d'opérations (C-Operation).

- Les C-Objets: ce sont des relations obtenues à partir des relations normalisées 3FN de CODD, en introduisant la contrainte supplémentaire de dépendance permanente (DP) entre les attributs (deux attributs sont en DP s'ils ont la même durée de vie). Par

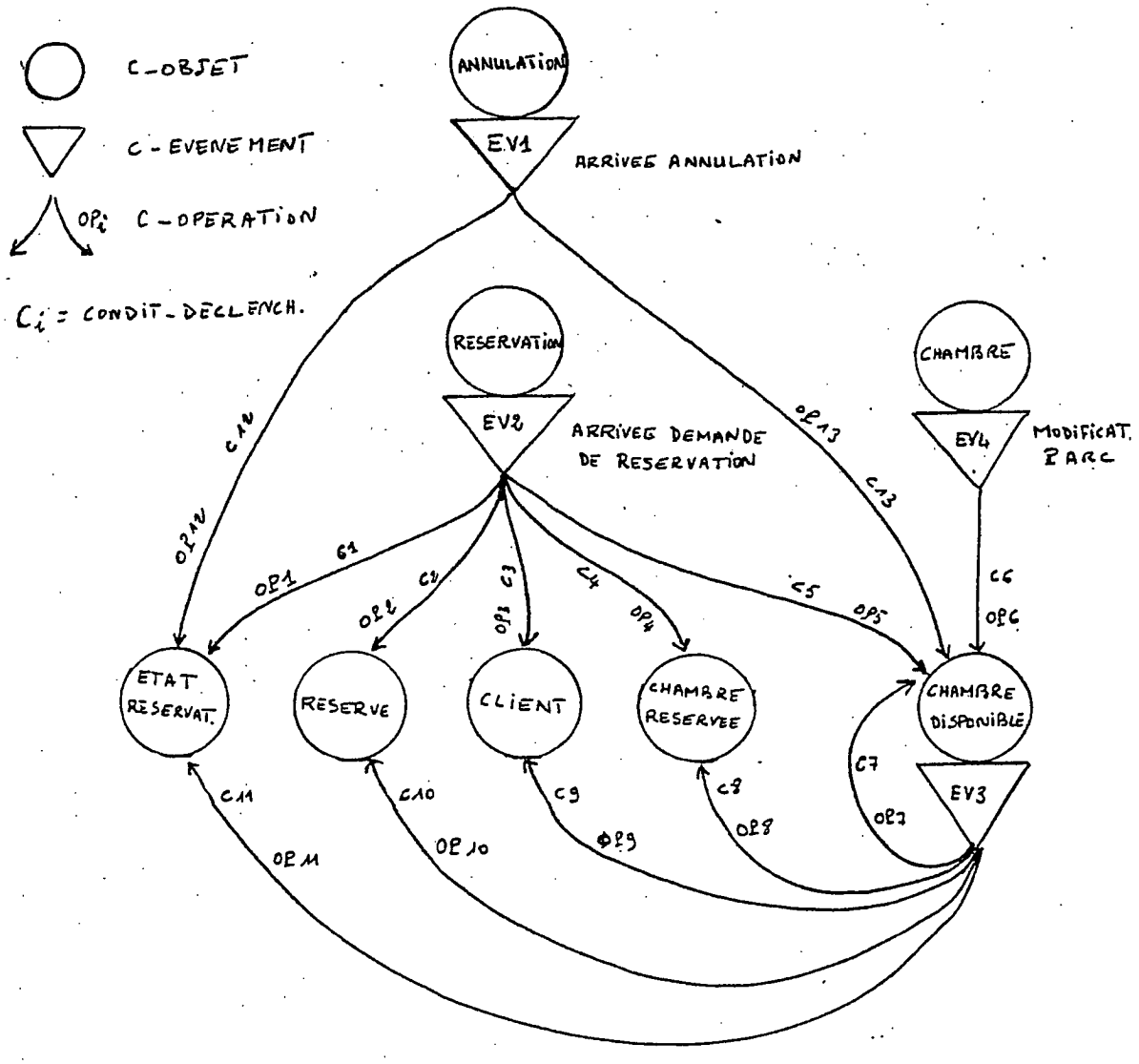


Figure 8: Un exemple de schéma dynamique REMORA

exemple, la relation 3FN suivante :

CLIENT(NoCli, Nom, Prenom, Adresse, Chiffre-affaire)

se décompose en trois C-Objets en utilisant les DP :

CLIENT_ID(NoCli, Nom, Prénom)
CLIENT_AD(NoCli, Date-adr, Adresse)
CLIENT_CA(NoCli, Date-ca, Chiffre-affaire)

- Les C-Evènements: ce sont des relations permanentes satisfaisant aux contraintes suivantes exprimées en termes de

dépendances fonctionnelles:

(1) C-eve -----> C-obj
(2) C-eve -----> Type modif (creation, suppression, mise à jour)
(3) C-eve -----> Predicat (lié au changement d'état du C-objet)
(4) C-eve -----> C-Operation

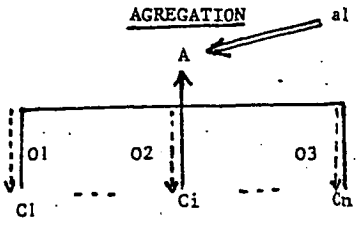
- Les C-Operations: ce sont des relations permanentes satisfaisant aux contraintes suivantes exprimées en termes de dépendances fonctionnelles.

(1) C-oper -----> C-objet
(2) C-oper -----> type modif (creat., suppress., màj)
(3) C-oper -----> texte operation

La notation C-eve -----> C-objet précise qu'un événement n'evoque qu'un seul objet. De même que C-oper----->type-modif signifie qu'une opération n'implique qu'une seule action: création, suppression ou mise à jour.

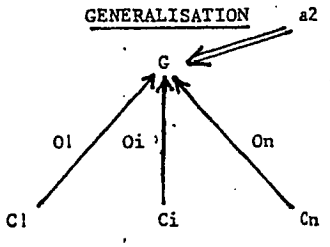
2-2-3- L'EXPRESSION DE LA DYNAMIQUE PAR LES TYPES ABSTRAITS

L'approche par les types abstraits se propose de faire une spécification précise et explicite de la sémantique d'un SI. Cette approche est initialisée par les travaux de [SMIT77] qui introduisent les concepts d'agrégation et de généralisation pour décrire l'aspect statique d'un SI. Dans l'approche types abstraits, on considère qu'un système est complètement défini si on a decrit tous ses objets et les opérations qui opèrent sur ces objets. Partant de cette hypothèse, BRODIE ajoute aux deux concepts d'agrégation et de généralisation celui d'ensemble (set) et propose de mettre face aux trois concepts de structuration de données les trois structures de programme traditionnellement connues: la séquence, l'alternative et l'iteration [BROD81].



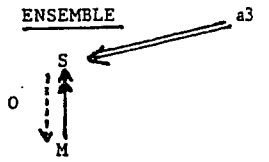
SEQUENCE

a1 est defini comme:
O1 sur C1
⋮
Oi sur Ci
⋮
On sur Cn;



ALTERNATIVE

a2 est defini comme:
a2 G
Case G of C1: O1 sur C1
⋮
Ci: Oi sur Ci
⋮
Cn: On sur Cn;



ITERATION

a3 est defini comme:

For each m in M
O sur m;

Figure 9: Structures de donnees et structures de controle correspondantes

A chaque classe d'objets (agrégat, ensemble, generique) est associée une action permettant, à l'exclusion de tout autre moyen, de la manipuler tout en conservant son integrité et celle des objets qui lui sont reliés.

Une action est un enchainement d'opérations élémentaires (primitives de consultation et de mise à jour) integrées dans l'une des structures de programme précédentes. Comme dans MERISE, chaque action est précédée d'une pré-condition et d'une post-condition permettant de contrôler le déclenchement et la terminaison. Un ensemble d'actions forme une transaction qui est le seul moyen offert à l'utilisateur pour interroger ou modifier le SI [Brod 81].

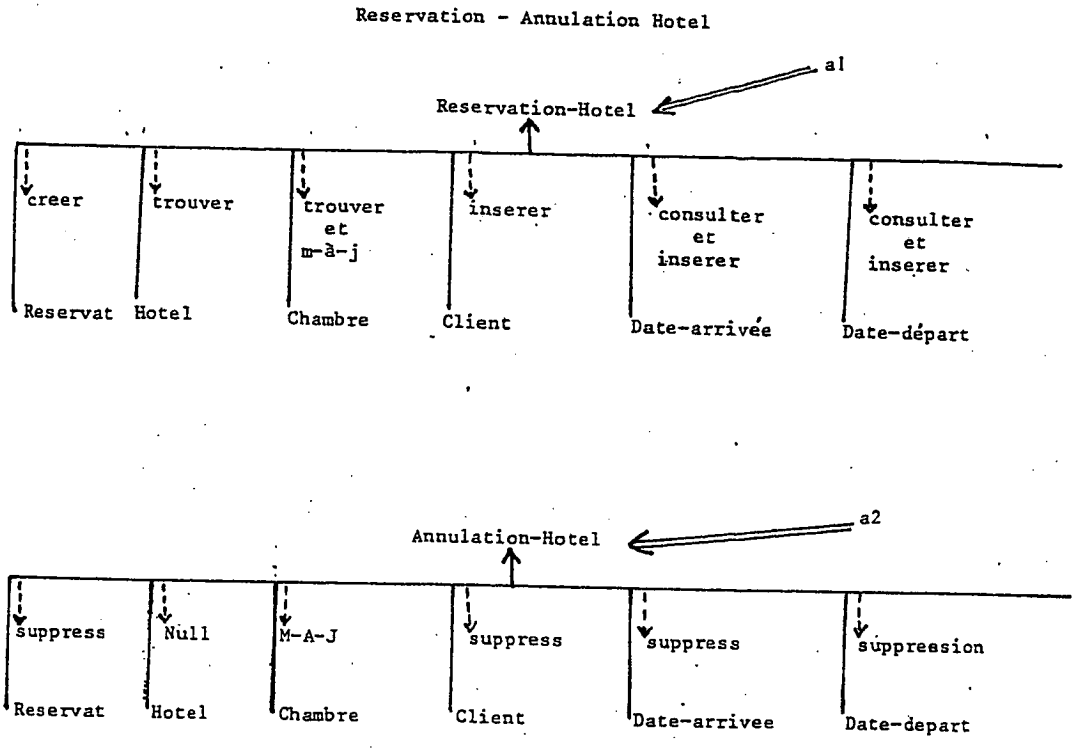


Figure 10: Un exemple de schema abstrait

L'action a1 peut être décrite comme suit:

```
Action  Reserv-hotel(h, p)
        if hotel existe(h) and personne-legale(p)
        then begin
                consulter-date-arrivée(a);
                consulter-date-départ(d);
                créer-num-reserv(n);
                reserver-chambre(n, h, r, a, d);
             end;
         if chambre-reservée(n, h, r, a, d)
         then reservat-faite = + (n, h, p, r, a, d);
end action
```

avec n = numero de reservation, p = client h = hotel

a = date arrivée, d = date depart, r = chambre.

2-3- OUTILS D'AIDE A LA CONCEPTION

L'idée séduisante d'une concéption automatique d'une BD est soulignée dans plusieurs rapports [LUM79]. En effet structurer une liste d'informations devient vite un processus complexe dès que l'on dépasse une centaine d'informations. Dans un premier temps, l'outil qui s'imposait de lui même était un catalogue de ces informations avec leurs significations, leurs contraintes d'integrité, leurs règles de gestion, etc...Ces catalogues ou dictionnaires de données font actuellement l'objet d'un interêt de plus en plus croissant [TARD75, LEFC77, CURT81]. Certains auteurs considèrent même que le DD est partie intégrante et indispensable à la BD et de ce fait doit attirer l'attention des chercheurs autant que les Bases de Données. La plupart des outils d'aide à la conception supposent l'existence de DD. Leur conception dépend des deux demarches citées en [2-1-2], nous nous interesserons ici à l'outil lui-même.

2-3-1- PRINCIPES GENERAUX

L'outil est généralement conversationnel et directif. Les réponses du concepteur sont introduites soit par clavier [TARD76], soit par écran graphique [CHAN80]. Ceci est dû au fait qu'il existe des choix que seul le concepteur peut faire (choix d'une entité, d'une relation, d'une DF...). L'outil est là uniquement comme aide memoire rappelant au concepteur certains

repères en lui posant un ensemble de questions pertinentes. Dans tous les cas ces outils ne font qu'une verification syntaxique du schéma, ils ne peuvent en aucun cas dire si le schema est cohérent ou non, ou encore moins s'il correspond à la réalite que l'on veut décrire.

Ils sont basés essentiellement sur un ensemble de scénarios définis à l'avance selon les caracteristiques et les règles de construction du modèle que l'on utilise. Par exemple, pour le modèle E/R , la définition d'une association ne peut se faire que si préalablement on a défini les entités participant a cette association. Ou encore un attribut non clé ne peut pas être affecté à la fois à deux objets differents, de même que si deux entités/associations ont un certain nombre d'attributs identiques alors il peut s'agir de la même entité/association [TARD76, ATZE81]. Pour mieux faire préciser au concepteur la définition de certains objets (entité, relation), l'outil s'aide souvent de quelques occurrences de ces objets (voir plus loin). Un autre moyen aussi de raffiner la structure d'un schéma est d'utiliser le processus d'intégration des vues externes a ce schéma. La confrontation de deux conceptions parallèles peut être d'un enrichissement important. Cependant cette confrontation peut être très complexe, ce qui reni encore plus accentué le besoin d'outil informatique.

Ce qui précède est brièvement illustre ci-dessous par deux approches distinctes mais complémentaires: CHAN et LOCKHOVSKY [CHAN80] pour l'interaction graphique entre le système

et le concepteur, et TARDIEU [TARD76] et ATZENI [ATZE81] pour l'aide proprement dite à la conception d'un schéma E/R.

2-3-2- APPROCHE GRAPHIQUE [CHAN80]

La première approche propose un écran graphique découpé en trois zones:

- une zone schéma recevant les éléments grahiques du modèle,
- une zone menu proposant un ensemble de primitives de manipulation de la zone schéma,
- et une zone message servant soit à la reception de messages provenant du systeme soit à l'edition d'informations décrivant le modèle.

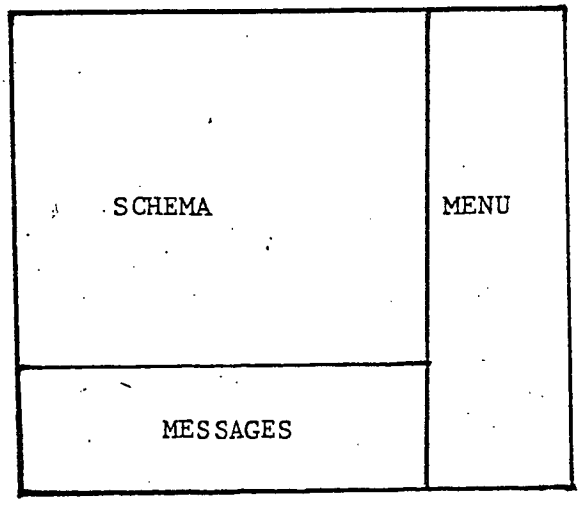


Figure 11: Découpage d'un écran graphique.

Le menu proposé comporte trois volets:

. le premier est une entrée dans le systeme par les differentes transactions que l'on veut réaliser (creation/modification d'un

schéma, visualisation d'un ou d'une partie de schéma...),

. le second est un ensemble de primitives de construction de schéma: ajout, suppression et modification d'entité, de relation ou d'attributs.

. le troisième est un ensemble de primitives de contrôle d'écran: grossir ou retressir une partie du schéma, visualisation des caracteristiques d'un élément particulier du schéma, impression du schéma.

Ces trois volets sont applicables soit au schéma global soit aux vues partielles. Derrière cet interface, l'outil réalise trois types de fonctions:

- Contrôle des règles de construction du schéma:

. on ne peut pas ajouter des associations si les entités participantes ne sont pas déjà introduites,

. la suppression d'une entité n'est possible que si elle ne participe pas à une association,

. une entité faible ne peut être introduite avant son ascendant, etc...

- Contrôle des règles de construction des vues. La construction des vues est une selection des objets du schéma global. Une vue correspond à un sous-ensemble stricte du schéma global. Une modification du schéma global doit se répercuter sur toutes les vues.

- Contrôle de l'écran: l'outil est implémenté sur un système graphique appele GPAC tournant sur PDP11/45. Le système graphique fournit toutes les primitives de manipulation de ségments

d'écran, de curseur... Ces primitives sont appelées par des CALL à partir du langage de programmation C. Les figures (rectangle, losange, traits) sont générées automatiquement à l'endroit que l'utilisateur aura précisé dans son écran à l'aide d'une tablette traçante.

2-3-3- APPROCHE DE TYPE CAO [TARD76] ET [ATZE81]

La deuxième approche s'interesse d'avantage aux règles de constuction et de vérification du modèle qu'à la forme de la conversation entre le système et le concepteur.

a) CATI-CAOMI

Chez [TARD76] tout le système est construit autour d'un dictionnaire de données servant tout le long du processus de conception. La structure Codasyl de ce DD (CATI) est representée fig.8. et commentée ci-dessous.

- INFORMATION : tout message décrivant un attribut, une contrainte d'integrité, une règle de gestion, un évenement ...
- MOT-CLE est l'ensemble des termes aidant à la définition d'une information,
- HISTO-INF est l'historique de l'information (differentes modifications subies au cours du temps),
- VALEUR est l'ensemble des valeurs que peut prendre l'information,
- COMPOSANT: ce sont les composants de l'information si elle

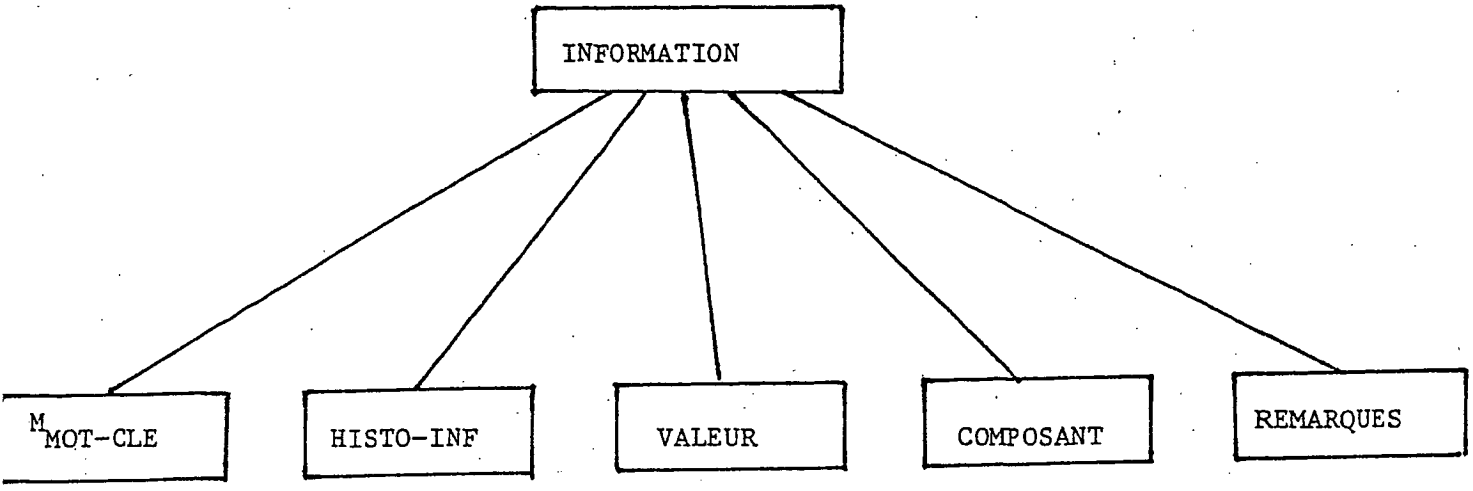


Figure 12 : Dictionnaire de donnees CATI.

n'est pas élémentaire, chaque composant devenant à son tour information,

- REMARQUE est tout commentaire fait sur une information.

Le dictionnaire de données permet d'enregistrer les informations en vrac afin de faire par la suite une etude lexicale permettant de préciser la signification de certaines informations, supprimer les synonymes et les homonymes et tirer une liste d'informations épurée à partir de laquelle se fera le travail de structuration proprement dit. Ce travail de structuration est fait par un module conversationnel à part, indépendant du DD. L'outil demande au concepteur de proposer ses entités et ses associations et de leur rattacher des attributs pris dans la liste précédente. Pour chaque entité introduite, l'outil demande trois exemples d'occurrences. Se servant de ces

exemples, il va poser une série de questions au concepteur concernant les valeurs prises par les attributs de chaque entité afin de verifier que la définition de ces dernières est correcte (l'ensemble des attributs qui les décrivent possèdent une valeur possible pour toutes les occurrences d'entités, que ces occurrences sont identifiables...) L'outil vérifie également la cohérence entre cardinalités et dépendances fonctionnelles entre entités et recherche des cycles de dépendances fonctionnelles pouvant conduire à des abhérations dans la définition de certaines entités. Enfin l'outil permet d'éditer sous forme textuelle et/ou graphique la structure du schema.

b) INCOD -

INCOD [ATZE81] ne précise pas la structure du dictionnaire de données mais donne l'architecture suivante de l'outil de conception:

. Le module "contrôle" aide le concepteur à se brancher sur l'un quelconque des autres modules.

. Le module de conception permet de mettre au point un schéma E/R par une serie de raffinages successifs.

. Le module d'intégration permet d'integrer plusieurs morceaux de schémas conçus séparément.

. Le module de dérivation permet d'extraire les vues des usagers du schéma global.

. Le module de documentation informe le concepteur des caracteristiques de chaque schéma et des transactions qui

l'utilisent.

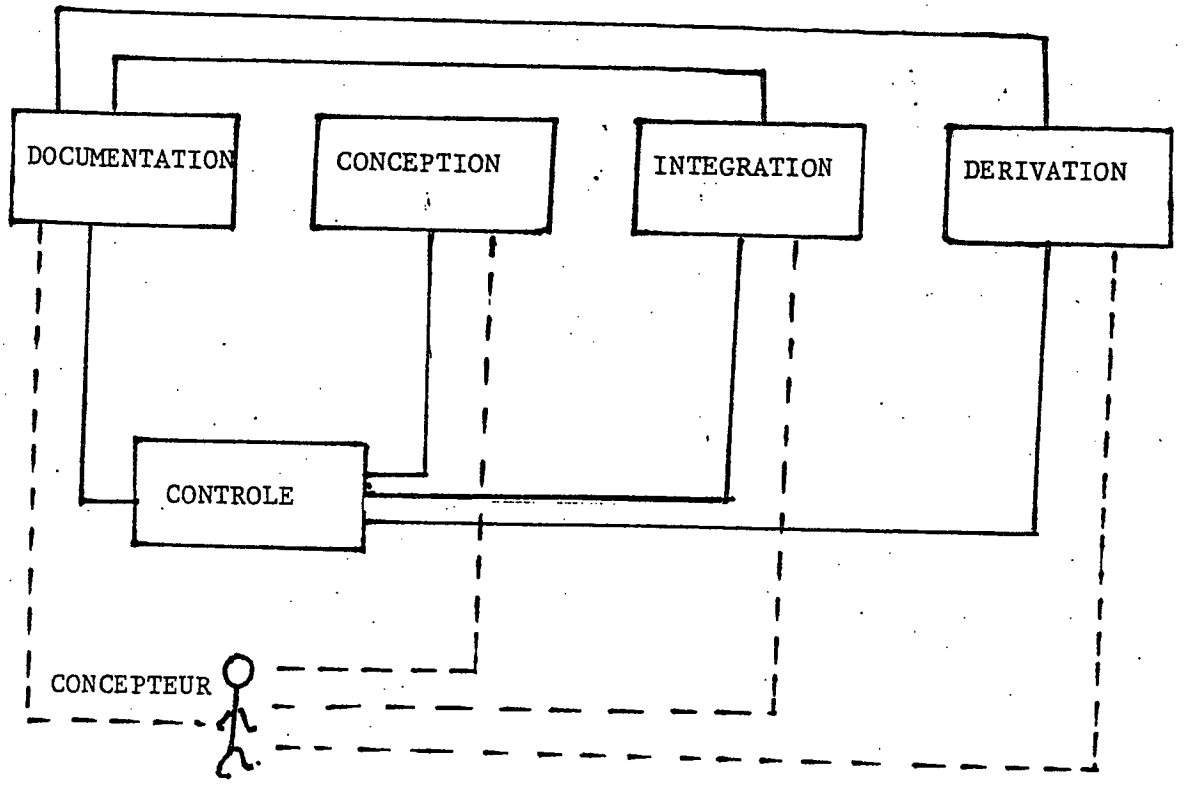


Figure13: Architecture du systeme INCOD.

Les differences essentielles entre [TARD76] et [ATZE81] résident dans les points suivants:

. le premier fait une conception globale du schéma, le second fait une conception parcellaire puis une intégration des parcelles,

. la conception des vues chez le premier est totalement indépendante du schema global, c'est l'étape de validation qui réalise le mapping entre les deux. Les vues sont simplement derivées du schema global chez le second.

2-4- CONCLUSION SUR LE NIVEAU CONCEPTUEL

L'objectif premier des BD etait de rendre independante la programmation des applications des structures de donnees manipulees. Les modèles relationnels et les niveaux de description conceptuel et externe d'ANSI/SPARC permettent d'atteindre cet objectif. Cependant la conception des deux modèles de donnees et de traitement n'est pas totalement indépendante l'une de l'autre. La structure même du modèle de donnees est influencee par les traitements qu'on veut y effectuer. De même l'expression des requêtes utilisateurs n'est pas toujours aussi indépendante des structures qu'on le desire (dans la plupart des LMD, le rapprochement de deux relations dans une selection se fait toujours au moyen d'attributs communs aux deux relations (jointures)).

En conséquence, l'utilisateur doit avoir une connaissance parfaite, directement ou par l'intermediaire de l'administrateur, de la structure conceptuelle globale de la base.

Un second point important à noter concerne l'évolutivite des schemas. L'organisation évolue, se transforme, se modifie ; les modèles de donnees sont ils capables de prendre en charge cette évolution? Il apparait déjà assez clairement que les schémas de données sont plus stables que les schémas de traitements. Ceci provient du fait que le schéma de donnees est une synthèse de l'ensemble des informations concernant le reel étudie. Il est donc rare que la quantite d'informations soit

remise en cause. Par contre cet ensemble d'informations peut évoluer quelque fois par sa structure qui doit alors s'adapter aux nouveaux besoins non prévus (par exemple repenser la définition de certaines entités ou relations, ajouter de nouvelles relations, de nouvelles contraintes d'integrité, etc...). En ce qui concerne les traitements il est plus difficile de recenser les besoins divers et variés que peuvent exprimer les utilisateurs. Certains traitements ont un cycle de vie égal à celui de la base (ils naissent avec elle et se repètent pendant toute la vie de la base), ce sont en général les applications potentielles pour lesquelles la base a été mise en place. Dautes apparaissent au cours de la vie de la base et/ou disparaissent après une période déterminée.

En conclusion, les méthodes de conception qui ne prènnent pas en compte les traitements ont échoué car les structures de données proposées ne minimisent pas l'activité de la base. A l'oppose celles qui utilisent comme préalable la connaissance de l'ensemble des traitements ne sont pas réalistes car elles ne tiennent pas compte de l'évolution de l'organisation ou tout simplement des difficultes qu'on rencontre à faire exprimer aux utilisateurs tous les besoins en une seule fois. Une solution intermédiaire consisterait à considerer la connaissance des traitements comme un parametre privilégié à la modèlisation des donneés au niveau conceptuel servant à mettre en évidence certaines rélations et à mieux préciser leur sémantique.

Nous verrons dans le paragraphe suivant comment on peut utiliser cette connaissance qu'on a des applications pour trouver la structure interne la plus optimale.

3- LE NIVEAU INTERNE

Par opposition au niveau conceptuel dont l'objectif est la description fidèle du monde reel, le niveau interne adapte cette description à l'outil informatique et plus spécifiquement a un SGBD. On distinge à ce niveau deux sous-niveaux [TARD79] :

- le niveau d'imlémentation logique et
- le niveau d'implémentation physique.

Cette distinction permet de séparer l'étape de définition de l'environnement de réalisation (choix du SGBD, évaluation des volumes de données à manipuler et des fréquences d'exécution des transactions, calcul du coût global du SI) de la réalisation proprement dite (génération des structures et programmation des transactions). Le niveau d'implémentation physique étant spécifique à chaque environnement nous ne nous interesserons qu'au niveau d'implémentation logique.

3-1- LE NIVEAU D'IMPLEMENTATION LOGIQUE

Les schémas de données élaborés à ce niveau sont des traductions de ceux du niveau conceptuel en tenant compte du fait que les structures logiques sont d'une part des structures

d'accès et d'autre part sont organisées de telle sorte à rendre optimale l'activite de la base. Ce niveau est donc basé essentiellement sur l'élaboration de modèles dactivité calculant le coût de fonctionnement du SI. Ce coût dépend du volume de données traité, de la fréquence des transactions identifiées, du choix des chemins d'accès et du placement des données.

La plupart des approches proposent des modèles évaluateurs de l'activité de la base. Ces modèles sont basés essentiellement sur la connaissance des traitements que l'on fait sur la base et les caracteristiques internes des SGBD. Beaucoup de modèles d'activité ont été développés pour les bases de données hierarchiques ou CODASYL; [CHEN77] et [LUMV79] en donnent un état de l'art; on trouve peu de choses sur le modèle relationnel; cela provient du fait que les SGBD relationnels ne sont devenus une réalité que très récemment. Cependant nous allons résumer quelques uns de ces modèles évaluateurs pour les modèles réseau et hierarchique car si les résultats obtenus ne proposent pas une structure relationnelle, la démarche utilisée et les paramètres pris en compte ne sont pas sans interêt.

3-1-1- CLASSES DE MODELES EVALUATEURS

La première classe de modèles évaluateurs regroupe ceux qui sont préoccupés uniquement par la mesure du temps de réponse. Ils opèrent en général sur des bases de données operationnelles [TEOR76] et se contentent d'observer le comportement des bases en

plaçant un certain nombre d'espions (compteurs) qui comptabilisent pour chaque transaction le nombre d'acces, le volume de donnees transfere, les quantites de ressources MC, MS, ou UC utilisees...Toutes ces informations sont regroupees dans un fichier statistique qui sera analyse par la suite pour deduire un modele d'evolution theorique. Ces evaluations ont souvent plus d'interet lorsqu'elles sont couplees avec des utilitaires de reorganisation permettant de corriger certains choix physiques (taille des blocs, indexes supplementaires, etc...) en vue d'ameliorer les temps de reponse. Au niveau de conception ou l'on se place, ces modeles auront peu d'interet.

La deuxieme classe de modeles est celle qui s'interesse essentiellement a une evaluation globale de l'activite de la base en vue de selectionner la structure de donnees adequate qui minimise cette activite. Il existe deux categories de tels modeles: les modeles simulateurs et les modeles analytiques.

- Les modeles par simulation ont l'avantage "d'observer" le comportement a priori et de quantifier certains elements difficiles a analyser (activite d'un indexe, activite d'une memoire cache, degradation provenant des debordements, etc...) et l'inconvenient d'une mise en oeuvre difficile et onereuse.
- Les modeles analytiques se presentent sous la forme d'une ou plusieurs fonctions de cout dont les facteurs representent les elements a mesurer et d'un algorithme d'optimisation dont les actions sont determinees par les valeurs prises par la fonction ou seulement par ses facteurs. L'interet de tels modeles est en

général une manipulation facile (dans le cas contraire on peut s'aider de l'ordinateur en programmant la rocédure de calcul de la fonction et/ou l'algorithme d'optimisation) et l'inconvénient est en général la difficulté de prendre en compte des comportements aléatoires (mémoire cache, filtres ...) et d'effets de bord (débordements , réorganisation...).

Dans les deux catégories de modèles, il se pose cependant les mêmes problèmes de choix des phénomènes à observer, des paramètres nécessaires à cette observation et des règles d'optimisation adéquates.

a) Parametres à observer

Qu'est ce qu'on observe? Essentiellement les accès disques car ce sont actuellement les plus pénalisants en temps de réponse. Viennent ensuite les volumes de données: un accroissement de volume peut dans un cas réduire les accès (redondances et indexes en consultation) et dans un autre cas les augmenter (en mise a jour). Enfin tres peu de modèles évaluateurs tiennent compte du temps nécessaire aux algorithmes de gestion de données (supposé négligeable devant les accès) et des coûts supplémentaires occasionnés par les blocs de débordement, les réorganisations et les contextes d'utilisation des données (partage des données, parallèlisme etc...). Les modèles simulateurs sont en général plus adaptes à intégrer ce dernier point que les modèles analytiques. Le temps de calcul nécessaire à l'application n'est pratiquement jamais considéré.

Quels sont les parametres en entrée? [CHEN79] donne une liste de paramètres généralement pris en compte. Ils se répartissent en trois groupes. Les parametres concernant les volumes de données (nombre de tuples, longueur moyenne d'un tuple, nombre de clés, sélectivité des attributs...), les paramètres d'activité ou de charge (nombre de consultations/insertions/suppressions par transaction et par période, taille moyenne en volume de données d'une réponse à une requête, ...) et des paramètres de stockage (temps d'accès moyen en direct et en séquentiel, temps moyen de transfert, longueur de bloc, taille des pointeurs, etc...). [IRAN79] ajoute les relations entre attributs et les contraintes de sécurité, la taille maximum des buffers, ... [TARD79] intégre les accroissements de volume et d'activité dûs aux indexes dans le volume et l'activité globale. [SCHK79] recherche des statistiques sur l'utilisation des attributs, les caracteristiques du SGBD et du matériel cibles.

b) Règles et algorithmes d'optimisation

Quelles sont les règles d'optimisation? La réponse a cette question dépend essentiellement de ce que savent faire les SGBD. Durant le placement des données, on utilise souvent des regroupements de tuples (clustering), les pointeurs (avant et/ou arrière), l'ordonnancement des tuples, les indexes secondaires, les fichiers inversés, etc... Durant la recherche de l'information, on opère au niveau de la requête en cherchant la

meilleure facon de l'executer (décomposition et optimisation des requêtes).

Les algorithmes d'évaluation et d'optimisation dépendent de l'approche choisie. Dans l'approche simulateur, il faut distinguer la phase évaluation et la phase optimisation. Dans la phase évaluation il y a un certain nombre de tests à faire en faisant varier à chaque fois les paramètres choisis. Ensuite vient une étape de dépouillement des résultats obtenus pour analyser l'origine de chaque coût. La phase d'optimisation consiste à changer de configuration de tests et à choisir entre les nouveaux résultats et les anciens. Dans l'approche analytique on se définit une fonction F à optimiser et l'algorithme calcule une valeur de cette fonction, opère un certain nombre d'optimisations, recalcule la valeur de la fonction F, la compare avec la précédente et décide de retenir ou de rejeter l'action d'optimisation, et recommence à nouveau. Il s'arrête lorsque par élimination, on aura rejeté toutes les actions d'optimisation. Pour calculer les valeurs de la foncton F, on se base en général sur les applications ayant une influence potentielle sur l'activité de la base. Selon les methodes, on définit pour chaque application sa fréquence, ses acces, son volume transportable etc... [SENK68] et [CARD73] donnent deux exemples de modèles simulateurs. [MILM77] et [TARD79] donnent deux exemples de modèles analytiques.

Le modèle de [CARD73] est resumé par la foction suivante:

COUT/PERIODE = COUT PISTE/PERIODE

* NOMBRE DE PISTES

+ FONCTION DE COUT DE CONSULTATION

+ FONCTION DE COUT DE MISE A JOUR

+ FONCTION DE COUT INQUANTIFIABLE.

Le modèle de [TARD79] est resumé par la fonction suivante:

F = NBRE D'ACCES + NBRE DE CARACTERES * K

avec K = PRIX DU CARACTERE/PRIX DE L'ACCES et le nombre d'accès est la somme de toutes les activites des fonctions évaluée par:

Fi = NBRE ACCES DE LA FONCT i * FREQUENCE DE LA FONCT i.

3-1-2- LE PLACEMENT DES DONNEES

On désigne par ce terme le regroupement dans une même zone physique des tuples d'une même relation ou de plusieurs relations. Ce regroupement permet dans le cas du modèle relationnel de trouver certaines sélections ou certaines jointures toutes faites. Dans le cas d'un modèle hierarchique ou CODASYL, certains chemins d'accès materialisés par une chaine de pointeurs se trouvent, le fait de les mettre dans une même zone physique, raccourcis (le pointeur ne matérialisant plus un accès disque mais un déplacement en mémoire centrale). Le principe essentiel du regroupement est de placer dans une même page les informations susceptibles de se traiter en même temps. Les limites de ce regroupement sont liées à la taille des pages

(unité de transfert).

[SCHK79] définit trois types de clustering: la ségmentation (regroupement vertical de certains attributs d'une même relation), le partitionnement (regroupement horizontal des tuples d'une même relation), et l'agrégation (regroupement des tuples de relations differentes). Les techniques de clustering partent des informations qu'on a sur l'utilisation des attributs pour les organiser en matrice et determiner les couples d'attributs ou les sous-matrices d'attributs dont les fréquences sont les plus élevées, ce qui determine les regroupements a faire.

3-1-3- LE CHOIX DES CHEMINS D'ACCES

Un des problèmes essentiels du niveau physique est le choix des index secondaires; les index primaires correspondent généralement aux identifiants des objets définis au niveau conceptuel. Ces derniers sont utilisés par les algorithmes de placement pour réaliser les chemins d'accès primaires en leur appliquant les arbres de placement, les fonctions de hachage ou toute autre transformation associant à une valeur de clé un emplacement physique sur mémoire secondaire.

Le choix et l'existence même des index secondaires doit être transparent à l'utilisateur. Il n'en est pas ainsi dans les modèles hiérarchiques ou CODASYL mais ceci est de plus en plus vrai dans les systèmes relationnels. L'existence d'index

secondaire, si elle accélère l'accès en consultation, pose un double problème en mise à jour. Dune part, pour chaque index secondaire introduit, il faut prendre en compte l'accroissement de coût occasionné en volume et en accès. D'autre part, pour l'ensemble des index, veiller à ce que la somme des accroissements des coûts consentis pour chaque index n'aille pas dans le sens inverse de l'optimisation visée. Une synthèse faite par [CHEN77] et [SCHK79] montre que les travaux les plus complets dans ce domaine ([HSIA70], [SEVE75], [YA077]...) sont loin d'être satisfaisants car ils ne traitent qu'une classe de problèmes.

Comme pour le clustering, le choix des index secondaires est basé essentiellement sur le taux d'utilisation des attributs. Par exemple si l'on recherche très souvent les employés par leur âge, il serait plus interessant de créer un index sur l'âge que de parcourir toute la base à la recherche des employés repondant à la tranche d'âge spécifiée dans la requête. Dans les modèles hiérarchiques où l'accès à un élément en feuille passe par le parcours d'une branche de l'arbre, l'index permet de raccourcir le chemin d'accès à cette feuille.

3-1-4- RESUME ET REMAQUES

En conclusion sur ce niveu d'implémentation logique, il faut remarquer les points suivants:

. Le choix d'une structure logique est base sur des critères quantitatifs, on ne sait pas prendre en compte le fait qu'une

structure est plus facilement programmable qu'une autre ou qu'elle peut exprimer plus de sémantique qu'une autre, etc...

. La complexité du modèle d'évaluation obtenu est liée au niveau de détail auquel on s'interesse et ce niveau de détail est lui même lié d'une part à la connaissance qu'on a des applications considérées, d'autre part aux hypothèses délimitant le champ d'application du modèle évaluateur.

. Les actions d'optimisation ne sont pas indépendantes les une des autres , ce qui peut rendre l'algorithme d'optimisation très complexe à réaliser et très lourd à manier.

. Le processus d'implémentation logique est iteratif mais pas entièrement automatique car il peut nécessiter des remises en cause du niveau conceptuel auquel cas seule l'intervention du concepteur est possible. Ce retour sur le niveau conceptuel est tout à fait significatif lorsqu'on se rappelle l'indeterminisme dans la conception qui caracterise ce niveau. Le niveau interne agit donc à son tour comme un outil permettant de trancher entre deux perceptions differentes du niveau conceptuel.

3-2- CONCLUSION SUR LE NIVEAU INTERNE

Ce qui caracterise ce niveau semble être essentiellement les modèles d'évaluation: évaluation de l'activite globale du SI, évaluation et choix des differents chemins d'acces, etc... Ces évaluations sont toutes basées sur une connaissance approximative ou précise de l'activite et de l'utilisation des attributs du

schema de données. En effet connaissant cette activité, certains évaluateurs proposent une procédure de clustering ou une procédure de choix des chemins d'accés à réaliser ou enfin une évaluation de l'activité globale du schéma. La pertinence des choix de réalisation interne dépend donc de toutes ces informations qu'on a sur l'utilisation des attributs. Pour obtenir ces informations beaucoup de parametres sont nécessaires: la définition des transactions à réaliser sur la base avec leurs fréquences d'exécution ; le volume de données manipulées en consultation ou en mise à jour, les attributs concernés par ces manipulations, le nombre d'accès nécéssaires à une transaction, le coût en E/S d'un accès, le coût des ressources utilisées, etc... Plus on a d'informations sur les applications qu'on veut réaliser, plus on a de précisions sur l'utilisation des attributs et mieux on peut bâtir un système optimal. Le cas le plus favorable serait alors celui ou l'on peut denombrer toutes les transactions à effectuer sur la base. Mais ce cas n'est pas réaliste, il est caracteritique des systèmes fermés, il ne tient pas compte de l'évolution de l'entreprise. Et le cas le plus souhaitable à une organisation, mais le plus défavorable à un concepteur, serait celui ou on construirait une base de données sur laquelle on peut tout faire à priori. Ces deux cas sont extrèmes et la réalite est tout autre.

En général, la mise en place d'une base de données est guidée par des orientations précises de ce qu'on veut en faire à long terme et des spécifications encore plus précises de ce que

l'on veut réaliser dans l'immediat. Le problème de conception se pose alors differemment. Tout d'abord, étant donné un ensemble de traitements spécifiés, trouver la structure de données optimale associée. Ensuite étudier comment varie cet optimum si on modifie certains paramètres, par exemple apparition de nouvelles fonctions (ce qui entraine un variation de la fréquence de certrains attributs), ou apparition de nouvelles données (donc peut-être de nouvelles relations, de nouveaux chemins d'accès, de nouveaux volumes etc...). Si cet optimum varie d'un facteur très faible, on peut déduire que la structure optimale proposée est stable et notre système évolutif, sinon cette structure n'est adaptée qu'à un problème particulier.

4- CONCLUSION GENERALE

Nous allons conclure cette synthèse des méthodes et des outils de conception de BD en faisant le constat suivant:

(1) La mise en place d'un système d'information automatisé passe par la connaissance de l'organisation et plus spécifiquement du domaine de l'organisation que l'on veut automatiser. La démarche pour atteindre un tel objectif relève d'une méthode générale de conduite de projets informatiques [MERI79]. La décision de représenter le système d'information par une base de données semble être actuellement un choix judicieux auquel adhère la plupart des concepteurs et utilisateurs. Le rôle du concepteur est alors plus précisement de mettre en place cette BD avec les meilleurs délais de réalisation, les meilleurs coûts d'exploitation et une grande souplesse d'évolution.

(2) La conception de la base de données n'est pas indépendante des traitements. La structure même de la base est influencée par ces applications (notamment le choix de certaines relations, le choix des chemins d'accès, etc...) Et puis on exige d'une BD de grandes pérformances, mais ces performances ne sont significatives que par rapport à un ensemble d'applications. Par conséquent, même si le concepteur de la BD n'est pas directement chargé de la réalisation de ces application, il doit en tenir compte en tant que paramètres essentiels à une élaboration judicieuse de ses structures de données.

(3) L'organisation évolue, le système de représentation de cette

organisation (BD) doit suivre cette évolution. Mais si au niveau conceptuel une retouche est toujours possible il n'en est pas toujours de même au niveau interne car l'outil informatique (SGBD) n'est pas toujours adapté pour faciliter cette évolution. Souvent une modification structurelle d'un schéma d'une BD nécessite la recréation de cette base; opération longue et coûteuse. De même que la modification des fréquences de certaines applications peut entrainer des conséquences fâcheuses sur les performances. Il revient donc au concepteur de tenir compte de cette évolution et d'élaborer des structures de données internes qui resistent à ces modifications.

(4) La conception des BD est un processus long et fastidieux et souvent très complexe car confronté à des problèmes de sémantique, de choix techniques et des coûts élevés. Par ailleurs, les méthodes de conception de SI, si elles existent et si elles sont utilisées, n'offrent que des repères permettant au concepteur d'évaluer le travail réalisé et les étapes qui restent à franchir. Mais toutes les décisions et tous les choix relèvent du concepteur. La qualité du produit obtenu dépend alors des connaissances que le concepteur a du domaine traité, de la maitrise des concepts et des outils fournis par ces méthodes et en général de l'intuition et de l'experience acquise par l'utilisateur dans le domaine de la conception.

(5) Les SGBD existant actuellement ont des performances telles que les choix faits au niveau conceptuel ne favorisent pas toujours une implémentation optimale.

En conclusion, il apparait actuellement que les méthodes et les outils de conception de SI ne peuvent avoir de succès que s'ils sont exprimés et présentés comme des systèmes informatiques puissants pouvant prendre en charge non seulement des tâches fastidieuses ou combinatoires mais aussi prendre des décisions pertinentes et faire des choix judicieux. Pour atteindre cet objectif, nous pensons qu'il est interessant de prospecter dans les domaines de l'intelligence artificielle et plus précisément des systèmes experts.

BIBLIOGRAPHIE

ABRI74 ABRIAL J.R.
Data Semantics (Data Base Management, North Holland publ. co. 1974)

ANSI75 ANSI/x3/SPARC-Study Group on Database Management Systems report
(Tsichritzis ed. feb 1975, reprinted in Information Systems vol 3, 1978)

ATZE81 ATZENI C., LENZERINI M., VILANELLI F.
INCOD: A System for Conceptual Design of Data and Transactions in the Entity-Relationship Model.
(Intle conf. on ERA, P.P. CHEN ed. ER Institut 1981)

ANDE77 ANDERSON H.D. & BERRA P.B.
Minimum Cost Selection of Secondary Indexes for Formatted Files
(ACM TODS, vol 2, No 1, March 1977)

ANTO79 ANTONELLIS v., CINDIO F., DEGLI ANTONI G., MAURI G.
Use of bipartite Graphs as a Notation for Data Bases
(Information Systems vol 4 1979)

ANTO81 ANTONELLIS V., ZONTA B.
Modelling Events in Database Applications Design
(Proceed. of 7th VLDB Conf. IEEE 1981)

BACK78 BACKUS J. "Can Programming be liberated from the Von Neumann Style? A Functional Style and its algebra of Programs" (CACM vol21,no8,1978)

BALD79 BALDISSERA C., CERI S., PELAGATTI G., BRACCHI G.
Interactive Specification and Formal Verification of user's views in Data Base Design
(Proceed. of 5th VLDB Conf. IEEE 1979)

BATI80 BATINI C. SANTUCCI G.
Top-Down Design in the Entity-Relationship Model
(Intle Conf. on ERA, P.P. CHEN ed. North Holland

BROD81 BRODIE M.L.
On Modelling Behavioural Semantics of Databases
(Proceed. of 7th VLDB Conf. IEEE 1981)

CARD73 CARDENAS A.F.
Evaluation and Selection of File Organization- A Model and System
(CACM Vol 16, No 9, Sept 1973)

CHAN80 CHAN E.P.F., LOCKHOVSKY F.H.
A Graphical Database Design Aid Using the E-R Model
(Intle Conf on ERA, P.P. CHEN ed. North Holland publ. co. 1980)

CHEN76 CHEN P.P.
The Entity-Relationship Model - Towari a Unified view of Data
(ACM Trans. on Database Systems Vol 1, No 1, March 76)

CHEN77 CHEN P.P.
Design and Performance Tools for Database Systems
(Proceed. of 3rd VLDB Conf. IEEE 1977)#

CHUN81 CHUNG I., NAKAMURA F., CHEN P.P.
A Decomposition of Relations Using the E-R Approach
(Intle Conf. on ERA, P.P. CHEN ed., ER Institut 1981)

CODD7O CODD e.F.
A Relational Model of Data fo Large Shared Data Banks
(CACM Vol 13, No 6, June 1970)

CODD79 CODD E.F.
Extending the Database Relational Model to Capture more meaning
(IBM- Sans Jose Research Report No RJ2599--1979)

COUL77 COULON D.
Les Reseaux Semantiques et les Modeles continus
(Journees de Travail sur la Comprehension
Arc et Senans, Mai 1977)

CURT81 CURTICE R.M.
Data Dictionaries: An Assesment of Current Practice and Problems
(Proceed. of 7th VLDB Conf. IEEE 1981)

FAGI77 FAGIN R.
Multivalued Dependancies and a New Normal Form for Relational Database
(ACM Trans.on Database Systems, Vol2, No 3, Sept 77)

GASC81 GASCUEL O.
Un programme general d'aide a la decision medicale structurant automatiquement ses connaissances
(Actes Congres AFCET RF-IA Nancy 81)

GRAY81 GRAY J.
The Transaction Concepts: Virtues and Limitations
(Proceed. of 7th VLDB Conf. IEEE 1981)

HAIN81 HAINAUT J.L.
Theoretical and Practical Tools fo Database Design
(Proceed. of &th VLDB Conf. IEEE 1981)

HAMM78 HAMMER M. & McLEOD D.
The Semantic Data Model: A Modelling Mechanism for Data Base Applications
(Proceed. of ACM-SIGMOD 1978)

HAMM81 HAMMER M. & McLEOD D.
Database Description with SDM :
A Semantic Data Model
(ACM Trans. on Database systems, vol 6, No 3, 1981)

HECK8O HECKENROTH H., TARDIEU H., ESPINASSE B.
Modeles et Outils pour la Conception de la Cinematique d'un System d'Information
(Publ. CETE Aix en Provence, Oct 1980)

HOUS79 HOUSSEL B.C., WADDLE V., YAO S.B.
The Functional Dependancy Model for logical DB Design
(Proceed, of 5th VLDB Conf. IEEE 1979)

IRAN79 IRANI K.B., PURKAYASTHA S., TEOREY T.J.
A Designer for DBMS_Processable Logical DB Structures
(Proceed. of 5th VLDB Conf. IEEE1979)

KARL81 KARLSON K.
Reduced Cover trees and their Application in the SABRE Acces Path Model
(Proceed. of 7th VLDB Conf. IEEE 1981)

KENT81 KENT W.
Data Model Theory Meets a Practical Application
(Proceed.of 7th VLDB Conf. IEEE 1981)

KOUL81 KOULOUMDJIAN J.
Un Modele d'Acees dans une BD Relationnelle
(ACTES Seminaire INFORSID, La Baule Oct 1981)

LANG80 LANGEFORS B.
Infological Models and Information User Views
(Information Systems, vol 5, 1980)

LAUR82 LAURIERE J.L.
Representation et utilisation des connaissances: les Systemes Experts
(TSI vol1, NO.1 et No.2, 1982)

LEON81 LEONARD M., LUONG B.T.
IInformation Systems Design Approach Integrating Data and transactions
(Proceed 7th VLDB Conf. IEEE 1981)

ROLL81 ROLLAND C., RICHARD C.
The REMORA Methodology for Information System Design and Management
(Seminaire INFORSID, la Baule Oct 1981)

SAKA78 SAKAI H.
On the Optimization of an Entity-Relationship Model
(AFIPS Conf. Proceed, 3rd USA-JAPAN Computer Conf., Oct 1978)

SAKA81a SAKAI H.
A Method for Defining Information Structures and Transactions in Conceptual Schema Design
(Proceed. of 7th VLDB Conf. IEEE 1981)

SAKA81b SAKAI H., KONDO H., KAWASAKI Z.
A Development of a Conceptual Schema Design Aid in the Entity-Relationship Model
(Intle Conf. on ERA, P.P. CHEN ed., ER Institut 1981)

SANT80 Dos SANTOS c.S., NEUHOLD E.J., FURTADO A.L.
A Data type Approach to the Entity-Relationship Model
(Intle Conf. on ERA, P.P. CHEN ed, North Holland publ. co. 1980)

SHIP81 SCHIPMAN D.W. "The Functional Data Model and the Data Language DAPLEX" (ACM TODS V6,N1,MARCH 1981)

SCHK79 SCHKOLNICK M., YAO B.
Physical Database Design
(in 1978 New Orleans Database Report No RJ2554 (33154) 1979)

SMIT77a SMITH J.M., SMITH D.C.P.
Database Abstractions: Agregation
(CACM Vol 20, No 6, June 1977)

SMIT77b SMITH J.M., SMITH D.C.P.
Database abstractions: Agregation and Generalisation
(ACM Transactions on DB Systems, vol 2, no 2, june 77)

SPAC81 SPACCAPIETRA S.
Analyse des Besoins d'Acces aux Donnees dans un Systeme d'Information
(Actes Seminaire INFORSID, la Baule oct 1981)

TARD75 TARDIEU H., HECKENROTH H., NNANCI D.
Etude d'une Methodologie d'Analyse et de Conception d'une Base de Donnees: le Schema de Reference
(publ. CETE d'Aix en Provence Mai 1975)

TARD76 TARDIEU H., HECKENROTH H., NANCI D.
Methode, Modeles et Outils pour la Conception d'une Base de Donnees d'un Systeme d'Information: le Manuel de Reference
(publ. CETE d'Aix en Provence, dec 1976)

TARD78 TARDIEU H., NANCY D.
Methode, Modeles et Outils pour la Conception d'une Base de Donnees d'un Systeme d'Information: le Passage au Modele Interne
(publ. CETE d'Aix en Provence, Nov 1978)

TEOR79 TEOREY T.J., et al.
Implementation Design
(in 1978 New Orleans Database Report No RJ2554 (33154) 1979)

WHAN81 WHANG K.Y., WIEDERHOLD G., SAGALOWCZ D.
SEPARABILITY: An Approach to Physical Database Design
(Proceed. 7th VLDB Conf. IEEE 1981)

TSIC82 TSICHRITZIS D.C. & LOCKOVSKY F.H.
Data Models (Livre, Prentice-Hall, Englewood Cliffs, New Jersey, 1982)

Imprimé en France

par

l'Institut National de Recherche en Informatique et en Automatique