MASTER EN SCIENCES INFORMATIQUES

Des outils informatiques d'aide au diagnostic en médecine homéopathique

Bouchat, Pierre

Award date:
1985

Awarding institution:
Universite de Namur

FACULTES UNIVERSITAIRES
NOTRE-DAME DE LA PAIX
NAMUR.

INSTITUT D'INFORMATIQUE

DES OUTILS INFORMATIQUES D'AIDE AU DIAGNOSTIC EN MEDECINE HOMEOPATHIQUE

Pierre BOUCHAT

Mémoire présenté en vue de
l'obtention du titre de Licencié
et Maître en Informatique

Promoteur : J. FICHEFET
Année académique 1984-1985.

P R E S E N T A T I O N

I N T R O D U C T I O N

Le mémoire réalisé entre dans le cadre d'une étroite collaboration de médecins homéopathes avec des informaticiens.

Les médecins sont réunis dans une a.s.b.l.:l'U.H.I.N. (Universitas Internationalis Hominum Novorum). Cette association a plusieurs buts :

- former de bons médecins homéopathes
- venir en aide aux médecins déjà formés
- promouvoir la recherche homéopathique.

Les informaticiens sont regroupés dans l'a.s.b.l.ARCHIMEDE (Association pour la Recherche en Informatique Médicale). Celle-ci a pour motivation de proposer des solutions informatisées face à des besoins médicaux.

La collaboration entre ces deux associations a déjà permis de mener plusieurs projets à bien.

Ce mémoire est le reflet de l'état actuel des choses:

- il présente tout d'abord le contexte dans lequel il évolue. Cette première partie permet de clarifier la signification du titre : "Des outils informatiques d'aide au diagnostic en médecine homéopathique.".
- Ensuite, vient une observation des outils disponibles au terme du travail :
  - -) les programmes réalisés par d'autres personnes
  - -) les outils qui ont abouti durant mon stage chez ARCHIMEDE.
- Enfin, j'évoque un certain nombre de propositions d'outils. Ceux-ci pourront être concrétisés à plus ou moins long terme. Ils ne seront cependant pas révolutionnaires mais permettront

plutôt une évolution progressive vers une assistance informatisée du médecin homéopathe.

Ce travail n'aurait pu être mené à bien sans le soutien de personnes compétentes tant en informatique qu'en médecine homéopathique. Je tiens donc à exprimer ma reconnaissance

à Monsieur le Professeur J.FICHEFET, promoteur de ce mémoire et président de l'a.s.b.l. ARCHIMEDE,

è Monsieur le Docteur A.JACQUES, homéopathe et président de l'a.s.b.l. U.I.H.N.,

à Monsieur P. GARDIN, informaticien à l'a.s.b.l. ARCHIMEDE,

à Monsieur J. PARIS, assistant de Monsieur J. FICHEFET.

PARTIE I. :

PRESENTATION DU CONTEXTE

INTRODUCTION A L'HOMEOPATHIE

Références : de (1) à (9)

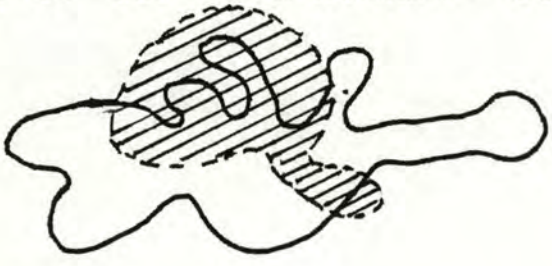
Dans ce chapitre, je décris d'abord une idée intuitive du travail du médecin homéopathe. Ensuite, je présente les sept principes fondamentaux sur lesquels repose toute l'homéopathie. Enfin, je décris brièvement la méthodologie adoptée dans le processus du traitement d'un patient.

I. UNE IDEE GENERALE.

Pour donner une première idée de l'homéopathie, je vais schématiser une maladie quelconque par la forme complexe que voici :

Le but de tout médecin, homéopathe ou non, est de supprimer la maladie. Pour ce faire, le médecin homéopathe va utiliser une sorte de "Clé" (ici, le remède) qui épousera au mieux la forme ci-dessus et qui sera suffisamment forte pour l'expulser.

Si le médecin choisissait n'importe quel remède, il risquerait de causer une aggravation (il suffit de constater les dégâts que provoque la "Clé" - en hachuré - du schéma suivant):



La maladie schématisée par le premier dessin s'extériorise par un ensemble de symptômes. Il convient donc que le remède couvre au mieux tous ces symptômes pour pouvoir les annihiler.

Le remède ci-dessous semble donc avoir beaucoup plus de chance de succès que le précédent et présente moins de risques d'aggravation.

Un tel remède est celui qui provoquera les symptômes du patient malade sur un individu en bonne santé.

II. LES SEPT PRINCIPES DE BASE.

Certains disent que l'homéopathie trouve ses racines dans l'Antiquité, période durant laquelle des mages de Babylone auraient dégagé un principe d'analogie et de similitude. Mais c'est grâce à Samuel HAHNEMANN (1755-1843) qu'un système cohérent a pu être établi. C'est d'ailleurs à partir de son ouvrage "Organon der Heilkunst" que furent extraits, en 1940, par la Commission de Doctrine du Deuxième Congrès National Mexicain de Médecine Homéopathique, les sept principes qui suivent. Ces principes constituent la base de la méthodologie homéopatique.

A) La loi des semblables.

" Une maladie dynamique dans l'organisme vivant est vaincue et détruite de façon durable par une autre maladie plus forte si celle-ci, sans être de même nature, lui ressemble cependant beaucoup dans sa manifestation."

B) Principe vital, dynamisme vital.

L'état de santé correspond à un état harmonieux et équilibré de l'énergie de l'organisme. L'individu participe totalement à cet état. Quand survient la maladie, l'équilibre énergétique est rompu. L'énergie vitale de l'homme malade est devenue différente dans sa totalité, elle a changé d'état.

Ce changement d'état de l'énergie se manifeste par des symptômes extérieurs, signes des efforts de rétablissement de l'équilibre de santé par l'organisme.

C) Natura morborum medicatrix.

" La maladie primitive est curable d'emblée et spontanément par l'action de la seule force de l'organisme."

Donc : " face à l'agression, lorsque la force vitale est suffisante, c'est-à-dire lorsque l'état d'énergie est suffisamment équilibré, l'organisme se répare de lui-même et récupère son équilibre sans aucune aide extérieure."

Mais, " lorsque la force vitale n'arrive pas à être victorieuse par elle-même, à ce moment la médecine doit agir avec intelligence et opportunité. La meilleure voie, c'est d'imiter la nature qui est "médicatrice", l'imiter c'est donner au malade une substance capable de produire les mêmes symptômes que ceux présentés par le patient, c'est rejoindre le principe de similitude, c'est rechercher le similimum."

D) Dose minimale, infinitésimale.

HAHNEMANN a découvert qu'en diminuant les doses par dilution, il arrivait à des résultats plus rapides et plus profonds. Il était donc parvenu à développer les propriétés pharmacodynamiques des substances utilisées en manipulant des doses infinitésimales.

E) Expérimentation pure.

L'homéopathe doit trouver un médicament capable de reproduire les symptômes de la maladie à soigner (par la loi des semblables). Il faut donc pouvoir connaître la symptômatologie de chaque remède. Pour cela, il convient de les expérimenter sur des individus sains pour ne pas risquer un mélange de symptômes et il est de la plus grande importance de les tester un à un pour ne pas obtenir un complexe de symptômes dont on ne pourrait rien tirer.

F) Les miasmes.

Le miasme désigne l'état morbide constitutionnel; c'est la constatation des altérations structurelles, fonctionnelles et psychiques que se transmettent les générations.

Il existe trois miasmes fondamentaux qui imposent au malade trois modes de réaction différents, à savoir :

-1. la psore : la carence, l'inhibition (ex.: anémie)
-2. la sycose : l'excès, l'exagération (ex.: indigestion)
-3. la syphilis : la réaction destructive (ex.: ulcère)

De plus, tous les terrains des individus sont constitués des trois miasmes en proportions différentes. Ceci explique la particularisation du malade face à la maladie : chacun d'eux fait sa maladie différemment.

G) Individualisation du malade et du remède.

Chaque malade réagit à une maladie en fonction de son propre terrain. L'effort de l'organisme pour rétablir l'équilibre vital est total, donc on retrouvera des symptômes partout sur l'organisme. Cet ensemble de symptômes fournira une image de la maladie. Cette image sera propre au malade. Le médecin homéopathe tentera d'adapter un remède qui produit la meilleure image possible en vertu de celle qu'il a observée par les symptômes. On obtient donc, pour une même maladie, un remède particulier à chaque malade.

III. APPROCHE METHODOLOGIQUE.

Ce paragraphe n'a pas la prétention dé se substituer à un cours d'homéopathie ni d'en résumer le contenu, mais bien de donner une approche intuitive de la difficulté d'une telle démarche.

A) Relevé des symptômes.

Le médecin procède d'abord à un relevé, le plus exhaustif possible, des symptômes du patient. Ces symptômes sont obtenus dans un certain ordre et suivant une technique particulière d'interrogatoire qu'il me semble inadéquat d'expliquer présentement.

Pour plus d'informations à ce sujet, il vous sera aisé de compulser le Précis de Technique Répertoriale Homéopathique de KENT.

B) Consultation des répertoires.

Une fois les symptômes relevés, le médecin consultera un dictionnaire des symptômes offrant pour chaque symptôme une liste de remèdes susceptibles de guérir. Le dictionnaire le plus répandu est celui de KENT.

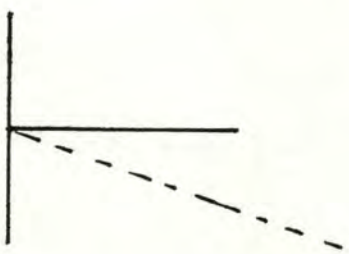
C) Choix du remède.

Sur base de l'étape précédente, le médecin obtient une première idée du remède à prescrire. S'il a un doute, il consultera la Matière Médicale pour affiner son jugement. Cette Matière Médicale reprend pour chaque remède (il y en a plus de 1500) les conclusions apportées par l'expérimentation pure ainsi que les symptômes observés. La plus répandue est celle de ALLEN.

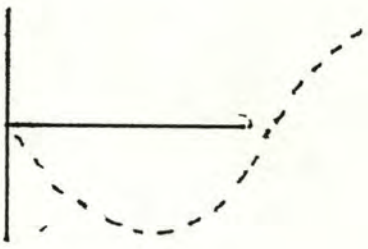
D) Evolution de la maladie.

La loi de HERING nous apprend comment une maladie peut aller en s'améliorant: " les symptômes disparaissent de haut en bas, de dedans en dehors, dans l'ordre inverse de leur apparition."

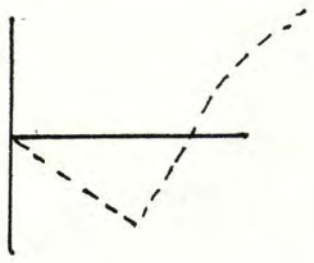
Dès lors, une fois le remède pris par le malade, douze éventualités peuvent se présenter :

1. 

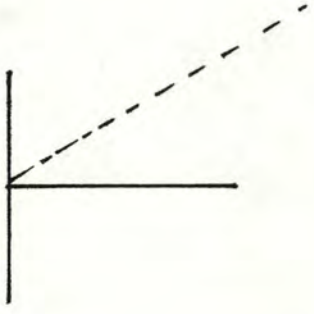
Aggravation prolongée avec déclin final du malade. INCURABILITE.

2. 

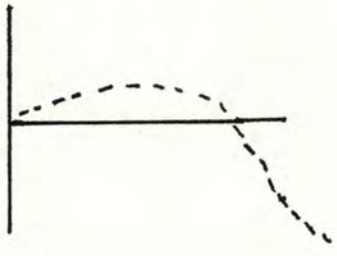
Longue aggravation, mais finalement suivie d'une lente amélioration.

3. 

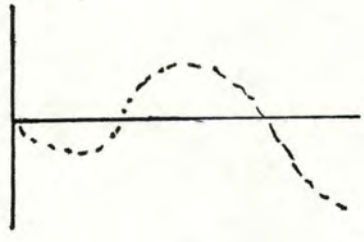
Aggravation rapide, brève, bien marquée, suivie d'une amélioration prompte et durable.

4. 

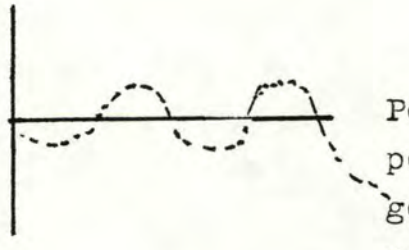
Retour à la santé, sans aucune aggravation quelle qu'elle soit.

5. 

L'amélioration précède l'aggravation, pronostic défavorable.

6. 

Trop courte durée du soulagement des symptômes.

7. 

Période d'amélioration de la maladie pendant 24 heures, cependant sans soulagement réel auquel on serait en droit de s'attendre pour le malade.PALLIATION.

8. Certains malades hypersensibles expérimentent tous les remèdes qu'ils prennent. INCURABILITE.

9. Action des médicaments sur des expérimentateurs sains PATHOGENESIES.

10. Apparition de nouveaux symptômes après l'administration du remède - mauvaise prescription
- pronostic défavorable

11. Retour d'anciens symptômes - pronostic favorable.

12. Mauvaise direction des symptômes - à l'encontre de la loi de HERING - pronostic défavorable.

En fonction de ces douze éventualités, le médecin continuera le traitement ou s'orientera vers une prescription différente de la première.

OUTILS D'AIDE AU DIAGNOSTIC HOMEOPATHIQUE

I. DEFINITION.

J'ai tenté de dégager ici quelques caractéristiques spécifiques du concept d'aide au diagnostic.

Un système d'aide au diagnostic :

-) concerne un processus non entièrement automatisable : cela veut dire qu'un ordinateur ne peut pas mener seul le processus de bout en bout.
-) ne remplace pas le médecin, mais contribue à améliorer son jugement (ou à l'accélérer).
-) nécessite une très bonne connaissance du domaine de la part de l'utilisateur (afin d'utiliser le système à bon escient).
-) peut conduire à une conclusion fausse qui forcera l'utilisateur à comprendre le cheminement de l'erreur (cela lui donnera une maîtrise supplémentaire).

On fournira donc un ensemble d'outils qui traiteront les parties automatisables et laborieuses du processus de diagnostic. L'utilisateur emploiera ces outils pour aboutir à un diagnostic ou pour améliorer son comportement dans l'approche du diagnostic.

II. POURQUOI EN HOMEOPATHIE.

Le processus de décision pour le choix d'un remède est long et complexe. Il fait intervenir des facteurs qui font qu'un ordinateur seul ne peut suffire. En voici quelques uns :

-) la subjectivité du médecin et son degré de connaissance des matières médicales orientent l'interrogatoire lors de la consultation.
-) la quantité énorme d'informations utilisables(60.000 symptômes et 1500 remèdes) font qu'arrivé à un certain niveau, un homme doit décider.

-) une vie humaine est en jeu (celle du patient); il faut donc un homme pour supporter les responsabilités et pouvoir réfuter l'ordinateur.
-) un principe de l'homéopathie veut que chaque maladie soit individualisée. Aucune règle de traitement globale n'a donc pu être définie à l'heure actuelle. Chaque cas est particulier (on ne peut pas dire : pour telle maladie, prescrire tel remède). Il existe seulement des règles méthodologiques insuffisantes pour le traitement total.
-) une certaine psychologie humaine est nécessaire au médecin pour pouvoir déterminer des symptômes mentaux délicats. Cette psychologie est acquise par l'expérience humaine, inaccessible à l'ordinateur.

Un système d'aide au diagnostic homéopathique va tenter d'allier la mémoire et la capacité de calcul de l'ordinateur à l'intelligence et à l'expérience de l'homme pour aboutir à une amélioration du diagnostic.

Il est à noter que des recherches en intelligence artificielle tendent à donner de plus en plus d'intelligence et d'expérience aux ordinateurs ("Systèmes experts" et "Machine learning").

III. UNE CLASSIFICATION DES OUTILS D'AIDE AU DIAGNOSTIC.

Je distingue ici deux types d'outils d'aide au diagnostic:

A) Les outils méthodologiques.

Si je les ai appelés ainsi, c'est parce qu'ils sont utilisés dans la phase de diagnostic proprement dit. On retrouvera dans cette catégorie des programmes qui traitent des parties automatisables d'une consultation.

exemple : la répertorisation fastidieuse et souvent incomplète devient une tâche routinière une fois programmmée: le repérage d'un symptôme et des remèdes associés est quasi instantané.

B) Les outils a posteriori.

Ce type d'outils est utilisé en dehors du déroulement d'une consultation. Il sert à apporter une connaissance supplémentaire au médecin ou à lui faciliter le travail dans ses recherches. Ces outils servent donc à affiner la théorie et la méthodologie homéopathiques afin de les rendre de plus en plus formalisables.

exemple : la consultation d'une banque de données cliniques apporte au médecin de nombreux enseignements sur ce qui a déjà été prescrit antérieurement.

Ce type d'outils fait progresser le domaine concerné.

Il n'y a aucune restriction au fait qu'un outil appartienne aux deux types d'outils à deux moments différents (ou pour deux utilisateurs différents). Mais, à un instant donné, l'outil, de par la façon dont il sera utilisé, appartiendra à une seule des classes définies ci-dessus. Il est à noter aussi qu'un outil peut être utilisé efficacement pour un tout autre usage que celui initialement prévu. Ceci rejoint l'idée des systèmes informatiques d'aide à la décision, avec la nuance que certains outils fournis ici imposent certaines contraintes par convention d'utilisation, par exemple, l'utilisation de la banque de données cliniques oblige le médecin à exécuter les programmes qui remplissent celle-ci.

P A R T I E II. :

O B S E R V A T I O N

D E

L' E X I S T A N T

OUTILS EXISTANTS

Trois outils sont déjà utilisables par les médecins :

1) un navigateur dans le KENT
2) un utilitaire d'extraction
3) un programme de consultation d'extraction

Le premier a été conçu pour intervenir lors d'une consultation tandis que les deux autres ont un but pédagogique et scientifique.

Je vais décrire rapidement en quoi consiste ces trois outils.

Il est cependant possible que la description qui suit devienne rapidement incomplète car des modifications sont à l'étude continuellement.

I. LE NAVIGATEUR.

On peut distinguer deux parties à ce programme :

A) L'accès au symptôme.

Le KENT entier est encodé sur le VAX/VMS. L'utilisateur peut donc accéder à un symptôme particulier ainsi qu'à la liste des remèdes associés. Pour accéder à un symptôme, il dispose de deux méthodes: 1) soit il introduit tout ou partie du symptôme désiré

2) soit, à partir d'une position donnée, il lit les symptômes séquentiellement comme s'il tournait les pages du KENT.

B) La sélection d'un remède.

Sur base des symptômes que le médecin a retenus, le programme offre deux possibilités de sélection d'un remède :

1) soit par la fréquence : le remède choisi est celui qui couvre le plus de symptômes.
2) soit par le degré : le remède choisi est celui dont la somme des degrés est la plus élevée (le degré d'un remède pour un symptôme est le potentiel du remède pour guérir ce symptôme).

Pour le navigateur, plusieurs extensions sont à l'étude actuellement.

On pourrait :

1) accéder au premier symptôme du KENT qui contiendrait un mot particulier (ex.: abcès),
2) obtenir une traduction des symptômes,
3) sélectionner un remède avec une méthode du type "Electre",
4) fournir un tableau qui, pour chaque remède, donnerait l'ensemble des symptômes retenus par l'utilisateur,
5) appliquer une pondération sur les symptômes (certains symptômes sont jugés prédominants par l'utilisateur).

II. L'UTILITAIRE D'EXTRACTION.

Ce programme permet de réaliser un grand nombre de choses différentes:

1) Pour un remède choisi, il fournit :
   toutes les rubriques où il apparaît à un degré 2 ou 3
   et/ou
   toutes les rubriques où il apparaît seul ou avec peu d'autres remèdes (le but est d'obtenir les traits particuliers du remède).

2) il produit les rubriques et/ou les remèdes avec leur fréquence conditionnellement à un mot choisi.

3) il permet une étude comparative sur tout le KENT d'un ensemble de dix remèdes au maximum (intersection et union).

III. UN PROGRAMME DE CONSULTATION D'EXTRACTION.

Ce programme est la suite logique du précédent. Il permet, une fois l'extraction réalisée, d'éditer le contenu à l'écran. Cette édition permet de parcourir ce texte avec les outils habituels d'un éditeur classique :

-) avancement avant et arrière
-) avancement ligne par ligne
-) avancement page par page
-) recherche sur base d'une séquence de caractères
-) etc.....

Mais il est impossible de modifier quoi que ce soit.

L'avantage de ce programme est d'éviter de sortir le résultat d'une extraction sur listing lorsque le besoin ne s'en fait pas sentir.

REALISATION DE LA BANQUE DE DONNEES CLINIQUES.

I. PRESENTATION.

Le but de cette banque de données est double :

1) permettre à chaque médecin de gérer sa propre clientèle
2) obtenir un certain nombre de cas remarquables qui serviront de cas d'école.

Mais en quoi ces deux faits peuvent-ils intervenir dans l'aide au diagnostic homéopathique? Chaque point intervient à des moments différents.

La gestion informatisée de la clientèle oblige le médecin à formuler toutes les informations avec précision et rigueur. Ceci lui impose de clarifier au mieux tous les problèmes rencontrés. Le cas pathologique sera mieux compris et lui permettra un meilleur diagnostic. Cette partie de la réalisation a donc un effet à court terme.

La deuxième partie - la banque de données centralisée - apporte une aide à plus long terme. En effet, quand la plupart des médecins auront garni leur propre banque de données, chacun proposera à l'U.I.H.N. une série de cas remarquables. Ceux-ci pourront être centralisés et donc mis à la disposition de tous. Chaque médecin pourra consulter ces cas particuliers, en tirer des statistiques ou les éditer sur papier. Le groupe profitera donc de l'expérience de chacun.

II. L'ETUDE DU PROJET.

Ce projet a été commencé l'an dernier et l'analyse préparatoire à la réalisation a été effectuée par Mesdemoiselles CH.CLARENNE et C. SONET. Cette analyse a été rédigée dans le mémoire intitulé " REALISATION ET EXPLOITATION D'UNE BANQUE DE DONNEES CLINIQUES HOMEOPATHIQUES." 1984-1985. C'est donc sur base des informations contenues dans ce mémoire que j'ai commencé la réalisation effective de la banque de données.

J'ai cependant dû prendre certaines décisions qui m'ont paru arbitraires mais nécessaires. En effet, certains points de l'analyse comportaient des incertitudes. D'autre part, certains changements ont été demandés et ont nécessité de légères modifications de l' analyse.

A) Points d'incertitude.

Voici d'abord quelques points d'incertitude :

1) Le dictionnaire des données est parfois imprécis quant à la signification de la donnée. La définition n'apporte rien comme information supplémentaire par rapport au nom de la donnée.
exemple : no-version-OMS: numéro de version du code OMS.

2) Certaines fonctions m'ont semblé répétitives. Je les ai donc supprimées, notamment toutes les fonctions de la gestion patient-groupe-patient.

3) D'autres fonctions ont été difficiles à comprendre :
exemple : le suivi d'un patient.

4) L'utilisation de TDMS pour la gestion d'écran était incompatible avec les spécifications.

5) L'utilisation de "Datatrieve" pour la gestion des fichiers n'avait aucune justification. Aucune analyse de faisabilité n'a été rédigée. J'ai cependant cru bon de suivre ce qui avait été écrit. Pour des raisons de facilité, j'ai modifié la spécification de la gestion des fichiers (les nouvelles spécifications se trouvent dans les programmes).

6) Les cinq points évoqués ci-dessus m'ont poussé à partir de l'analyse fonctionnelle à considérer les algorithmes comme faux ou inadéquats (parfois ils étaient même absents - ex. les statistiques). Ils ne sont pas totalement faux, mais j'ai

préféré recommencer ces algorithmes plutôt que de chercher les erreurs ou refaire les parties fausses.

7) Un manque de spécification au niveau des fonctions a compliqué la situation.

8) Aucune description d'écran n'était fournie.

Les problèmes soulevés en 1), 2) et 3) n'ont pratiquement pas occasionné de retards, mais les points 4), 5), 6), 7) et 8) ont fait que je ne pouvais plus me limiter au rôle de programmeur. J'ai donc dû refaire toute la démarche d'analyse pour comprendre ce que chaque fonction était censée faire et par là créer les algorithmes et les écrans qui me semblaient adéquats.

B) Modifications requises.

Voici enfin quelques changements demandés. Comme, de toute façon, je devais refaire tous les algorithmes, ces changements n'ont pas prêté à conséquence.

1) la localisation de la banque de données centralisée.

Suivant le mémoire, la banque de données était confondue avec la banque de données des médecins. Or le besoin s'est fait sentir de les séparer pour des raisons de protection de l'information : un médecin ne peut plus modifier un dossier qu'il a fait centraliser.

2) l'organisation de la banque de données de chaque médecin.

Un dossier complet d'un patient doit pouvoir être traité sur le "Rainbow" du médecin ou sur le VAX/VMS.
Ces deux nouvelles contraintes ont fait que la banque de données se présente sous la forme suivante :

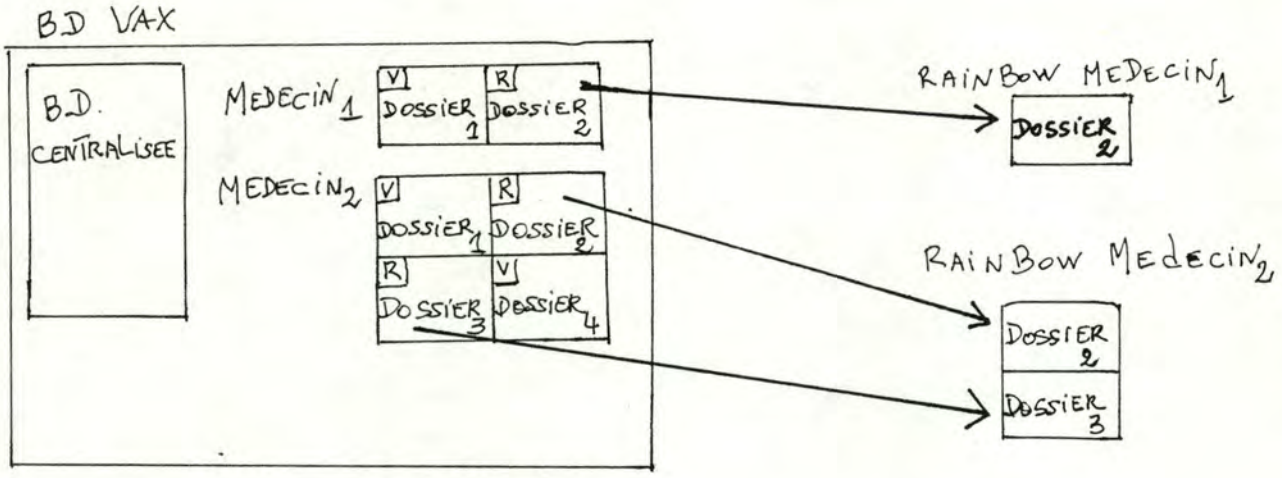


fig. II. 1

C) Constitution de la banque de données.

Il y a deux types de banque de données distincts :
- 1) la banque de données centralisée sur le VAX/VMS
- 2) l'ensemble des dossiers des médecins du système.

Voyons en détail chacune d'entre elles.

- 1) la banque de données centralisée.

Elle contient, comme il a toujours été convenu, des cas particuliers qui ont été introduits sous l'approbation de l'U.I.H.N. Elle est accessible à tous en consultation. Lors de la centralisation d'une maladie, on ne fait que déplacer le dossier de la banque de données du médecin concerné vers la banque de données centrale.

- 2) les dossiers des médecins.

L'ensemble des dossiers d'un médecin se trouve réparti sur le VAX/VMS et sur le "Rainbow"; ceci afin de limiter le nombre de disquettes nécessaires chez le médecin. Les disquettes contiendront les dossiers les plus fréquemment utilisés.
Le VAX/VMS contiendra les dossiers moins importants ou plus anciens. Il ne faut pas perdre de vue qu'un dossier devra être

retiré du VAX/VMS pour être disponible sur disquette (et inversément), mais il faut savoir également qu'un médecin pourra aussi bien traiter un dossier sur disquette que sur le VAX/VMS. Le transfert sur disquette n'est donc pas obligatoire mais est moins onéreux.

Prenons l'exemple :

Un médecin a une clientèle de 100 patients. Il sait que 10 patients sont en cours de traitement chez lui. Il aura donc sur disquette les 10 dossiers correspondants. Un de ces patients guérit. Il n'a donc plus besoin de son dossier jusqu'à sa prochaine maladie. Il peut donc compléter ce dossier et le transférer sur le VAX/VMS.

Par la suite, un patient enregistré sur le VAX/VMS revient pour une autre maladie : le médecin peut transférer son dossier sur disquette pour le mettre à jour, mais il peut également travailler sur ce dossier directement sur le VAX/VMS.

Comment peut-on, pour la banque de données d'un médecin particulier , assurer une sécurité qui se pose à deux niveaux?

-) garantir la cohérence de chaque dossier : il faut que chaque accès à un dossier fournisse une version complète et correcte de ce dossier.
-) garantir une possibilité de reprise en cas de panne : lorsqu'un dossier se perd, il faut pouvoir récupérer une version antérieure pour limiter la perte de temps.

La solution apportée est la suivante :

-) le médecin créera lui-même son dossier sur une disquette qui sera la seule utilisée à ce moment.
-) les dossiers, inutiles sur la disquette, pourront être transférés sur le VAX/VMS. Ils disparaîtront donc de celle-ci.
-) les dossiers situés sur le VAX/VMS pourront soit être modifiés sur place, soit être transférés sur disquette avant d'y être traités.

-) les dossiers qui auront été transférés sur disquette ne seront pas effacés du VAX/VMS, ils seront seulement marqués comme étant indisponibles (cf. fig.II.1). Il sera cependant possible de les récupérer en cas de problème.

La cohérence est respectée car un dossier ne peut pas être modifié à la fois sur le VAX et sur le "Rainbow" : il est considéré comme localisé à un seul endroit. La possibilité de reprise est permise par le fait que, si le dossier a été transféré au moins une fois sur le VAX, il existe une version inaccessible sur celui-ci par le médecin, mais récupérable par l'opérateur.

Remarque : il y a tout intérêt à transférer régulièrement un dossier sur le VAX pour être sûr de disposer d'une version de secours en cas de panne.

Cette nouvelle façon de voir la banque de données m'a contraint à préciser le lieu d'un dossier : sur le "Rainbow" ou sur le VAX/VMS. Pour accéder à un dossier sur ce dernier, il faut vérifier sa présence et son accessibilité. Ceci ne prend pas trop de temps car la décision est simple:

- ou bien le dossier est présent et on peut agir
- ou bien il est présent mais inaccessible et on ne peut autoriser aucune action.

III. LA REALISATION.

Il est peut-être bon de rappeler quelles étaient les phases décrites dans le mémoire de référence.

1. Création d'une fiche-médecin
2. Mise à jour d'une fiche-médecin
3. Création d'une fiche-secrétaire
4. Mise à jour d'une fiche-secrétaire
5. Edition de rappel-courrier
6. Enregistrement des analyses

7. Transfert des questionnaires enregistrés
8. Contrôle des transactions par médecin
9. Centralisation d'un dossier médical
10. Gestion de la liste des identifications
11. Suivi d'un dossier groupe-patient
12. Suivi d'un dossier-patient
13. Edition de dossiers
14. Edition/Affichage de statistiques
15. Création/Maj dossier patient-groupe-patient
16. Encodage de questionnaires patient et individu

Toute la réalisation est basée sur cette découpe en phases.

A) Ce qui a été fait.

Vous trouverez, en annexe, tous les programmes qui ont été réalisés. Pour chaque programme il y aura :
-) la source "COBOL" du programme
-) un exemple de fichier de résultats s'il existe
Joints à ces programmes, vous trouverez également en annexe :
-) la description des écrans utilisés
-) les requêtes, c'est-à-dire les demandes de saisie et d'affichage des zones d'un écran
-) les librairies de requêtes
-) la description des records, des domaines et des ports utilisés dans "DATATRIEVE".

Chaque programme correspond, le plus souvent, à une phase de l'analyse fonctionnelle telle qu'elle fut réalisée dans le mémoire qui me sert de référence.

Il y aura également en annexe :
-) un manuel d'utilisation général qui reprendra les caractéristiques communes d'utilisation des programmes.
-) un manuel d'initialisation globale
-) trois manuels d'utilisation particuliers (un pour chaque médecin, un pour l'U.I.H.N., un pour les opérateurs).

B) Ce qui n'a pas été fait.

1) La validation des codes (OMS, profession, activité, analyses..)

La validation des codes semble être une fonction importante pour la centralisation. En effet, toutes les statistiques sur les dossiers centralisés vont se faire sur base de ces codes. Il convient donc qu'ils soient utilisés à bon escient. Ils devaient être validés à chaque introduction. La validation prévue n'était qu'une vérification de l'existence du code dans un fichier de référence inexistant.

Il m'apparut alors que cette validation coûtait beaucoup de temps pour très peu de profit. Il me semblait plus utile de vérifier la concordance entre le code et le libellé et de laisser la possibilité au médecin d'effectuer la validation de ses codes quand il le désirait (en Batch si nécessaire).

En résumé :

Problèmes : 1) les fichiers de référence pour les codes sont indisponibles.
2) la validation proposée est trop faible et donc inefficace.
3) la validation est indispensable mais uniquement pour une maladie à centraliser.

Solution: 1) dans un premier temps, validation manuelle par une personne compétente avant la centralisation.
2) dans un deuxième temps:
a) élaboration d'un programme plus performant de comparaison du code et du libellé.
b) programme fonctionnant non pas au coup à coup mais d'un bloc pour une maladie à centraliser.
c) programme lancé à la demande et non pas exécuté d'office.

La solution proposée permet au médecin de ne pas devoir valider les codes des dossiers qui ne seront jamais centralisés (gain de temps).

2) Le suivi d'un dossier-groupe-patient.

Je n'ai pas jugé nécessaire de programmer cette phase car elle me semblait tout à fait redondante avec la phase intitulée "Création/Maj. dossier-groupe-patient". Si tel n'est pas le cas, il y a un gros problème au niveau de l'analyse fonctionnelle.

La phase 15 précitée telle qu'elle est définie dans le mémoire qui me sert de référence, recouvre le suivi d'un dossier-groupe-patient car :

1) la création et/ou la mise à jour d'un dossier individu sont des actions possibles au niveau de la phase 15.
2) le type d'information et la façon de saisir cette information sont similaires.
3) la phase 15 est plus complète que la phase 11("Suivi d'un dossier-groupe-patient").
4) le fait que ces deux phases peuvent s'adresser à deux types de personnes différentes (médecin ou secrétaire) ne pose pas de problème en ce qui concerne la programmation.

Ces quatre raisons font que j'ai jugé inutile de réaliser la phase 11 du suivi d'un dossier-groupe-patient.

3) La gestion de la liste des identifications et l'encodage de questionnaires.

Ces deux phases avaient déjà été codées lors du mémoire précédent. J'ai cependant dû modifier la façon de considérer un numéro de dossier.

Au départ, le numéro d'un nouveau dossier était attribué par le programme séquentiellement et par ordre croissant. Il s'est avéré nécessaire de laisser le choix de ce numéro au médecin utilisateur. De plus, la numérotation était numérique et le médecin pouvait introduire de l'alphanumérique.

Ces modifications ne sont pas disponibles en annexe car la plus grande partie de la programmation a été réalisée par d'autres.

Pour consulter ces programmes, il faudra s'adresser aux personnes concernées de l'a.s.b.l. ARCHIMEDE.

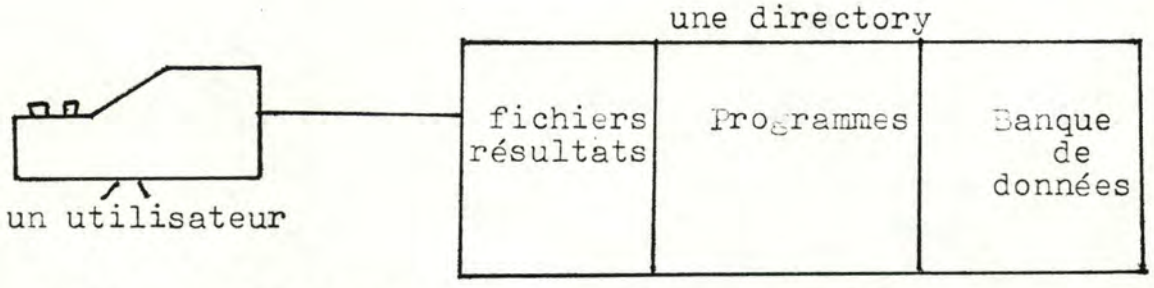
4) L'implémentation finale.

Un problème reste en suspend à ce niveau-là. Bien que tous les programmes fonctionnent, ils ne pourront pas être effectifs pour les médecins. En effet, toute la conception et la mise au point ont été réalisées dans la directory qui m'a été attribuée sur le VAX/VMS. Tous les résultats apparaissent donc dans la même directory.

Le problème suivant est double :

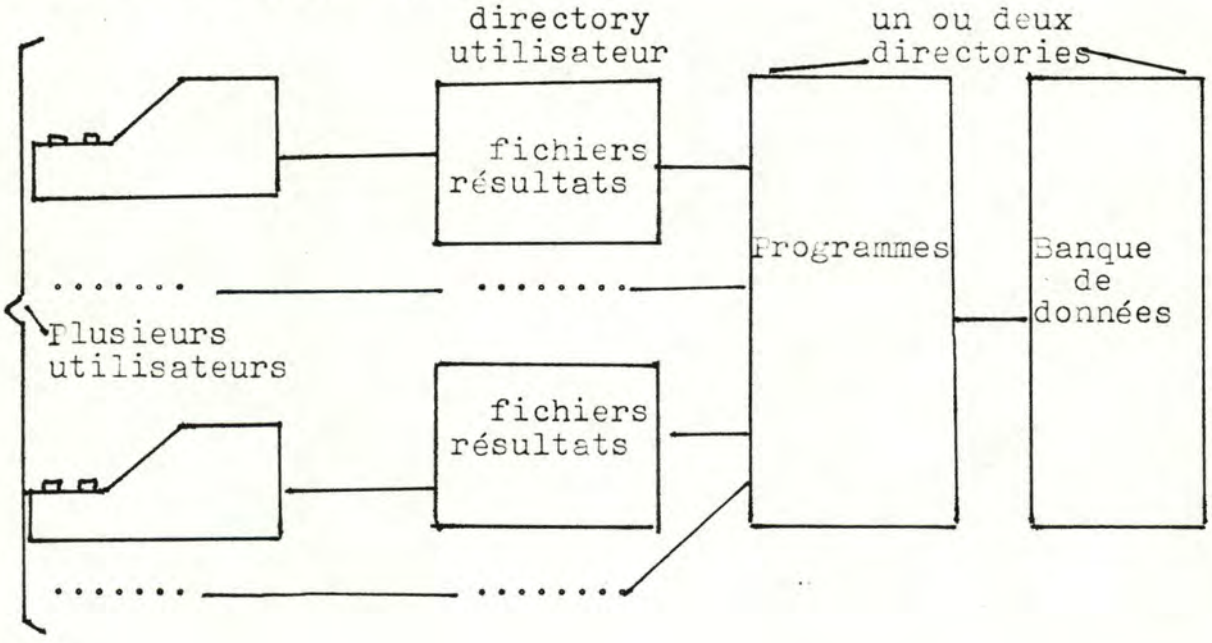
1) la gestion de la concurrence
2) la personnalisation des fichiers résultats

La situation telle qu'elle était pour la mise au point est la suivante :



Comme le montre la figure ci-dessus, tout est localisé à la même place et pour un seul utilisateur.

Mais la situation telle qu'elle devrait être lors de l'utilisation est la suivante :



Comme le montre la figure ci-avant, deux utilisateurs différents peuvent exécuter un même programme sur la banque de données tout en produisant des résultats différents. Il faut donc que chaque utilisateur ait accès uniquement à ses propres résultats.

exemple : deux médecins veulent éditer certains dossiers.
Ils exécutent donc ensemble le programme EDITER,
l'un pour le patient "A310" du médecin "02"
l'autre " " " "B5A3" " " "07"
A la fin de l'exécution, chaque médecin va faire imprimer son propre fichier résultat. Il faut donc qu'un même programme écrive à deux endroits différents mais lise dans la même banque de données.

Le principe de solution de ce problème est le suivant:

1) pour les fichiers résultats, le programme les écrira d'office chez l'utilisateur : c'est la façon de procéder par défaut sur le VAX/VMS.
2) pour la banque de données, il faudra signifier explicitement dans le programme qu'il faut écrire dans une directory spécifique.
3) pour les programmes, une fois placés dans la directory de travail, il faudra :
   a) changer la protection du fichier exécutable et des fichiers de la banque de données pour que tout le monde puisse l'utiliser.
   b) définir pour les médecins des noms logiques pour ces programmes pour qu'ils n'aient pas à se soucier de l'endroit où ils se trouvent.

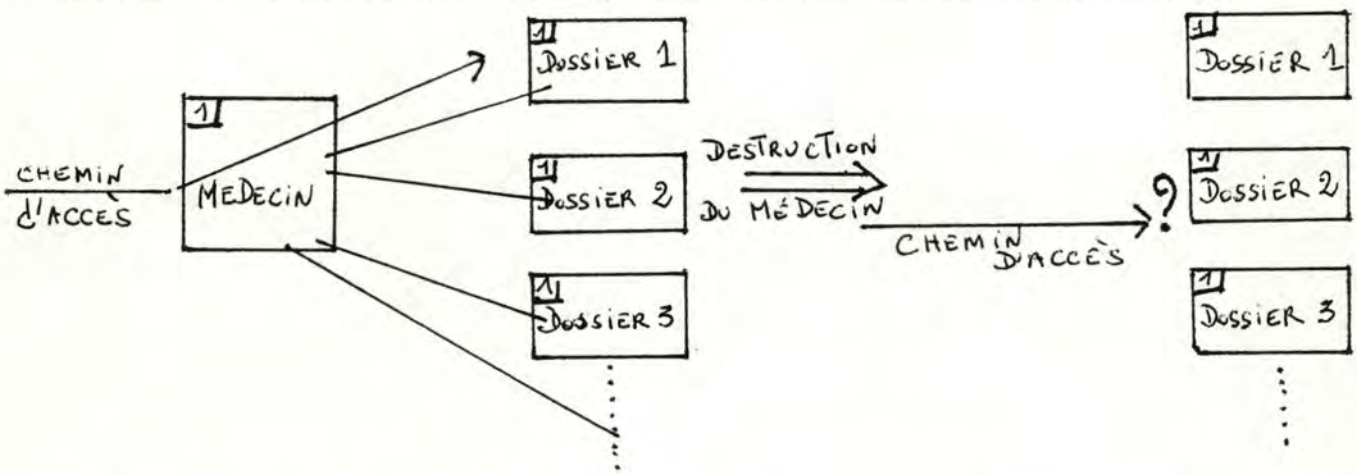
C) Ce qui a été ajouté.

Je me suis permis d'ajouter deux fonctions supplémentaires au niveau de la gestion des médecins et de leurs secrétaires : -) la destruction d'un médecin
-) la destruction d'un(e) secrétaire

La destruction me semblait surtout importante au niveau d'un(e) secrétaire. Il est possible qu'un(e) secrétaire parte.

Il faut donc pouvoir l'enlever du système. Cette opération se fera sans risque de fausser la banque de données.

Il en va de même pour un médecin qui viendrait à décéder ou à qui on interdirait l'utilisation du VAX/VMS. La fonction de destruction d'un médecin du système est cependant à manipuler avec grande précaution. Les dossiers de ses patients lui sont attachés par un lien implicite (le numéro interne du médecin). Si on détruit le médecin, on perd toute possibilité d'accès à ses dossiers bien qu'ils soient toujours présents.



Recréer le médecin ne servirait absolument à rien vu que le programme lui attribuerait un nouveau numéro interne et ne pourrait pas ainsi recréer le lien rompu.

D) Les choix au niveau de l'utilisation.

J'ai classé les programmes en deux parties :

-) les interactifs
-) les non interactifs ou ceux qui ont une courte partie interactive

Pour les programmes non interactifs (constitution de fichiers d'édition,...), il me semble plus adéquat de les faire traiter en Batch pour ne pas monopoliser une ligne inutilement.

Passons en revue les différentes phases :

1) la gestion des médecins et des secrétaires.

Cette phase est purement interactive pour l'introduction des

informations et se termine par une constitution d'un fichier d'édition. Ce fichier reprend tous les médecins et tou(te)s les secrétaires du système. Il serait peut-être utile de le construire en Batch lorsque le nombre des médecins deviendra conséquent (plus de 30 médecins).

2) L'édition de rappels-courrier et le contrôle des transactions par médecin.

Ces phases ne nécessitent pas d'intervention lors de leur déroulement. On peut donc les faire exécuter en Batch. Il conviendra cependant de vérifier si le(s) fichier(s) constitué(s) répond(ent) bien à l'attente.

3) L'enregistrement des analyses et le transfert des questionnaires enregistrés.

Ces deux phases agissent en création ou modification sur la banque de données du VAX/VMS. Il me semble donc relativement nécessaire de surveiller le bon fonctionnement de ces programmes en permanence. C'est pourquoi, j'ai disposé tout au long de chaque processus des instructions d'affichage de résultats intermédiaires. Ceci ne peut donc pas être réalisé en Batch. Toutefois, lorsque le risque d'erreur pourra être négligé, on pourra ôter les instructions d'affichage et exécuter le programme en Batch. Pour ma part, le risque est encore suffisamment élevé que pour confier la charge de ces deux opérations aux seuls programmes.

4) Centralisation d'un dossier médical et édition de dossiers.

Ces deux phases ont un intérêt particulier du fait qu'elles sont interactives au début puis deviennent fortement non interactives. Toutes deux ont une phase de sélection totalement dépendante des désirs de l'utilisateur. Cette première phase est suivie d'un processus autonome de traitement correspondant à la sélection préalable. Ce dernier peut être lancé en Batch. Ainsi, on ne mobilisera la ligne que le temps d'opérer la sélection voulue.

5) Affichage de statistiques - Suivi du dossier d'un patient - Création/Maj. d'un dossier patient-groupe-patient.

Ces trois phases constituent l'essentiel de l'acquisition et de la consultation d'informations pour la banque de données. Il est donc évident que tout est interactif.

IV. L'UTILISATION.

On ne peut pas commencer à utiliser les programmes dans n'importe quel ordre. Il est impensable de créer une nouvelle maladie pour un patient qui n'existe pas encore. Ce paragraphe explique brièvement quelles phases doivent être exécutées en priorité.

Passons en revue les phases programmées dans l'ordre de leur priorité :

1) La gestion des médecins et des secrétaires.

Le préalable à toute utilisation du système par un médecin est que ce médecin soit repris comme utilisateur de ce système. Pour cela, il faut introduire certaines informations le concernant et lui fournir un numéro d'utilisateur.

2) Le transfert des questionnaires enregistrés et la Création/Maj. du dossier patient-groupe-patient.

Une fois le médecin accepté dans le système, il devra d'abord introduire les informations sur le patient et ses proches avant de pouvoir traiter ses maladies et ses consultations.

3) Le Suivi d'un dossier-patient.

Enfin le médecin pourra gérer les maladies des patients encodés.

4) Toutes les autres phases sauf l'affichage des statistiques.

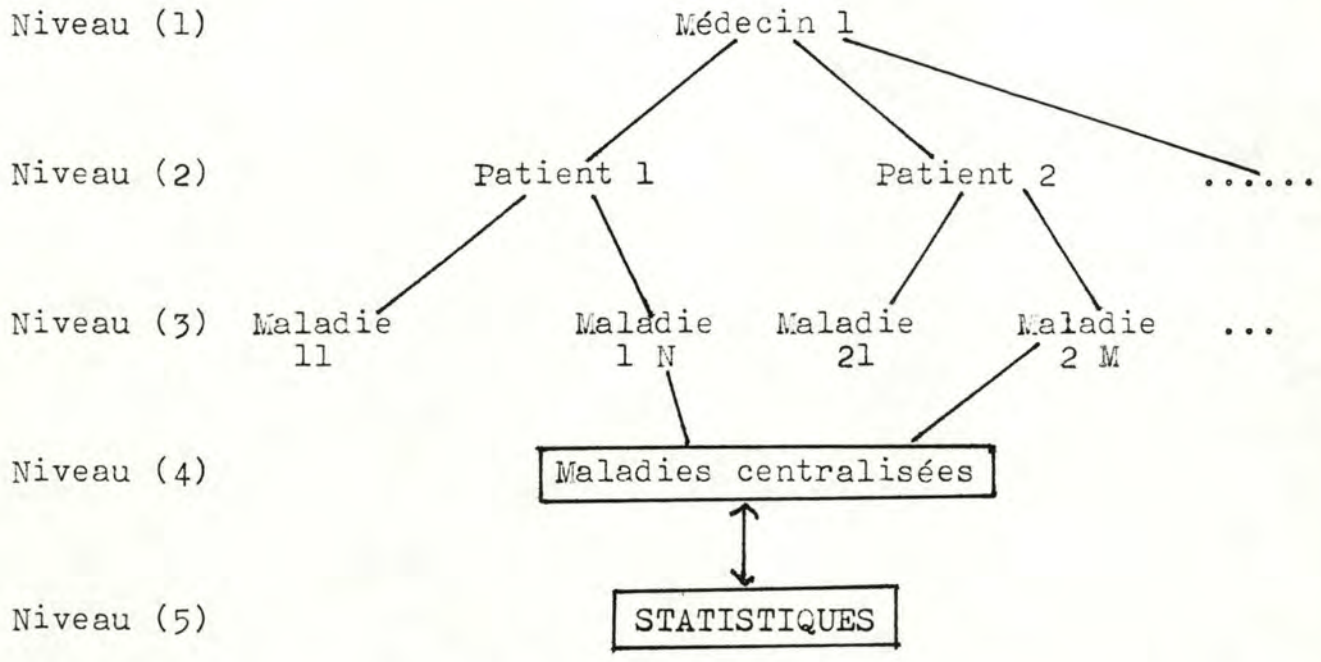
Toutes ces phases peuvent être exécutées lorsqu'un minimum d'informations a été encodé.

5) Affichage des statistiques.

Cette phase ne peut être exécutée que lorsqu'un nombre suf-

fisamment grand de dossiers de médecins aura été centralisé.

En résumé, on obtient une hiérarchisation comme suit :



Il est donc très important de respecter cette hiérarchie dans les étapes de fonctionnement.

Pour de plus amples informations sur l'utilisation stricte des programmes, il est préférable de s'en référer aux annexes qui contiennent les manuels d'utilisation adéquats.

V. CRITIQUE DE LA REALISATION.

Cette critique se constitue de quatre phases :

-) la gestion d'écran
-) la gestion de fichier
-) la programmation générale
-) le style de programmation

Elle a pour but de considérer, avec le plus d'objectivité possible, les avantages et les défauts de la réalisation.

A) La gestion d'écran.

A) La gestion d'écran.

De par mon expérience personnelle en gestion d'écran, je peux affirmer que TDMS est un outil performant pour l'aide à la conception d'écrans. Seuls quelques défauts dus à la généralisation nécessaire à un tel outil sont à remarquer. Il est cependant relativement facile de concevoir un écran et il est aisé d'utilisation.

Voici quelques avantages et inconvénients :

1) Avantages :

-) TDMS offre une rapidité d'action suffisamment acceptable (surtout au niveau du champ).
-) simplification très grande de la programmation(request).
-) l'utilitaire d'aide à la conception des écrans est très facile et permet de multiples possibilités.

2) Inconvénients :

-) en utilisation, le mode d'insertion n'existe pas au niveau du champ - ex. DUONT
   on ne sait pas insérer un P entre U et O,
   il faut remplacer chaque lettre par celle qui convient (important pour les champs très longs).
-) la superposition d'écrans est interdite : un seul écran à la fois.
-) le constructeur présente TDMS comme avantageux par le fait qu'il évite de devoir recompiler le programme après la modification d'un écran. Cet avantage est réduit à néant par le fait qu'il faille reconstruire la librairie des écrans. Par curiosité, j'ai comparé la vitesse de création d'une librairie et la vitesse de compilation et de linkage du programme qui utilise cette librairie (en l'occurence: OPERATEUR_LIB et PROGR.COB).

1) @ RDU
   RDU > BUILD LIBRARY OPERATEUR_LIB OPLIB.RLB
   RDU > EXIT
   @

2) @ COB PROGR
   @ LINK PROGR, DTR/OPT
   @

La première série d'instructions prend près du double de temps par rapport à la seconde.
Or, pour une gestion d'écran classique par fichiers de descripteurs, aucune compilation ni construction de librairie n'est requise. La conception des fichiers de descripteurs nécessite cependant une très grande rigueur dans l'élaboration des écrans.

En conclusion, TDMS est un outil qui offre des performances plus qu'acceptables pour l'utilisateur et permet une très grande compacité de programmation (donc de rapidité) pour le concepteur.

B) La gestion des fichiers.

DATATRIEVE ⟷ COBOL

En voyant la lenteur d'exécution des routines de gestion des fichiers utilisant DATATRIEVE, il m'est venu à l'idée de comparer ce qui a été réalisé avec un programme utilisant une gestion de fichiers classique du style COBOL. J'ai effectué cette comparaison sur le programme PROGR. La version COBOL de ce programme est disponible en annexe sous le nom "DOUBLE.COB".

De cette étude comparative, j'ai déduit un certain nombre de points positifs et négatifs pour chaque type :

1) La rapidité.

On obtient un rapport de 1 à 5 en faveur de COBOL. L'accès à un fichier par DATATRIEVE est quasi inacceptable. Cette différence est encore accentuée par le système de mémoire virtuelle du VAX/VMS.

2) La gestion de la concurrence.

COBOL offre une gestion de la concurrence à deux niveaux: le fichier et le record. Pour ce qui est de DATATRIEVE, je serais nettement plus réservé. La gestion de la concurrence en lecture ne pose aucun problème, mais si la concurrence se

porte sur des fichiers ouverts en écriture, je ne peux rien garantir. J'obtiens à chaque fois un message non pas de DATATRIEVE mais de RMS, et ce message est impossible à gérer. Tandis qu'avec COBOL aucun problème n'est survenu. Il est à noter que la vitesse d'exécution en COBOL aide grandement ; on tombera rarement au moment où une des routines des fichiers est exécutée tant elles sont rapides.

3) La maîtrise du programme.

Le nombre d'états particuliers dans lequel peut se trouver DATATRIEVE, le nombre d'options possibles et le nombre de routines possibles pour un même état font que DATATRIEVE est difficile à maîtriser par programme. Les routines sont considérées comme exactes tant qu'elles se terminent correctement. Le problème réside dans le fait qu'il y a plusieurs façons de réaliser la même chose (mais ces méthodes sont-elles équivalentes dans tous les cas?).
En COBOL tout est clair et précis. Il permet de gérer soi-même les erreurs encourues.
Ce problème s'observe également dans le point suivant.

4) La longueur des routines.

Alors que DATATRIEVE est d'un niveau plus élevé que COBOL, c'est COBOL qui a les routines les plus courtes (car les plus spécifiques).

5) Les types d'accès à un fichier.

Ici apparaissent un avantage de DATATRIEVE et un inconvénient de COBOL. On peut accéder par DATATRIEVE à un ensemble de records du fichier sur base de n'importe quelle clé (facile pour les statistiques) même si elle n'a pas été déclarée dans la définition du fichier. Par contre, en COBOL, il faut déclarer toute clé d'accès. Ceci oblige donc un parcours séquentiel de tout le fichier si on veut accéder sur base d'une valeur non déclarée comme clé d'accès. De plus, deux

clés d'accès différentes ne peuvent commencer sur le même Byte (très inconfortable).

En conclusion, "DATATRIEVE" semble être un outil remarquable en usage interactif mais il est préférable d'utiliser la gestion de fichier disponible dans le langage utilisé pour la programmation. "DATATRIEVE" est d'un niveau trop élevé et donc entouré d'une trop grosse quantité de softwares inutiles et lourds à manipuler. Un avantage considérable de "DATATRIEVE" est la possibilité de l'utiliser en interactif ou en programmation sans aucun changement. L'interactif permet alors l'utilisation de certains utilitaires tels que DECGRAPH pour la représentation graphique de statistiques.

C) La programmation générale.

Au départ, je ne pouvais compter que sur l'analyse fonctionnelle : aucune description d'écrans n'existait et la presque totalité des algorithmes était à remettre en question. J'ai donc commencé à programmer la phase de gestion des informations sur un médecin et un(e) secrétaire. Ce premier programme fût réalisé alors que je ne maîtrisais pas encore complètement le COBOL. De plus, je n'avais pas encore exploité toutes les possibilités offertes par la gestion d'écran (TDMS) et la gestion des fichiers (DATATRIEVE). Ces possibilités, je les ai découvertes au fur et à mesure de la programmation. Il est donc évident que je suis plus satisfait du dernier programme que du premier.

Il me semble important à ce niveau d'indiquer l'ordre dans lequel j'ai réalisé la programmation :

1) "PROGR" : pour la gestion des médecins et des secrétaires
2) "VERIF" : pour la vérification des transactions par médecin
3) "TRANSFERT" : pour le transfert des questionnaires patients
4) "TRIND" : pour le transfert des questionnaires individus
5) "SUIVIP" : pour le suivi d'un dossier-patient
6) "RAPPELS" : pour l'édition de rappels-courrier
7) "ANAL" : pour l'enregistrement des analyses
8) "QUEST" : pour la création/Maj. du dossier patient-groupe-patient

9)"CENTR" : pour la centralisation d'un dossier médical
10)"EDITER" : pour l'édition de dossiers
11)"MENU" : pour affichage de statistiques

Ensuite, comme je n'étais pas satisfait, j'ai entrepris de reprendre tous les programmes et de les améliorer. Faute de temps, je n'ai pu améliorer que le premier "PROGR"; pour les autres, je me suis contenté de les mettre au point définitivement.

Mon souci premier était donc de produire un ensemble de programmes exécutables répondant à l'analyse fonctionnelle. Je pense y être arrivé en offrant parfois plus que ce qui était demandé (ex.: l'édition de dossiers, l'affichage de statistiques). Pour s'en convaincre, il convient de regarder les manuels d'utilisation en annexe.

Pour l'affichage de statistiques, j'ai construit un langage d'interrogation de la banque de données. Ce langage est constitué d'opérateurs logiques ($\wedge$, $\vee$) et de noms. Je suis parvenu à obtenir une personnalisation suivant l'utilisateur.
Exemple : pour un code de remède, un utilisateur peut le définir comme "CODE" et un autre comme "CODER".
Ceci afin de faciliter l'utilisation pour chacun.

Mon seul regret est de ne pas avoir réalisé un ensemble de programmes suffisamment présentables. De gros défauts sont apparents au niveau de l'aide interactive à l'utilisateur. Il en va de même pour la disposition des écrans et l'agencement des fonctions réalisées par le programme. Je ne suis pas pleinement satisfait de ce qui a été réalisé à ce niveau. J'attribue cet état de fait à la quantité de programmes à concevoir et au peu de temps qui m'était imparti.

D) Le style de programmation.

Il est clair que ma façon de programmer a évolué sur les cinq mois qu'a duré la partie programmation. Pour s'en convaincre, il suffit de regarder le programme "PROGR" pour la gestion des informations sur un médecin et/ou un(e) secrétaire et de le comparer au programme "STAT" d'affichage de statistiques. (tous deux se trouvent en annexe)

Je vais comparer ces deux programmes suivant deux critères différents :

1) Présentation des possibilités d'action.

Par possibilités d'action, j'entends les fonctions mises à la disposition de l'utilisateur.

Le premier programme fonctionne par une succession d'écrans présentant les différentes options. Le second présente ses possibilités sous forme d'un langage de commandes.

Une fois la période d'apprentissage terminée, la succession de menus devient laborieuse à utiliser. Au contraire, un langage de commandes permet de visualiser directement un nouvel état du système. L'utilisation d'un tel procédé donne un avantage psychologique à l'utilisateur en lui donnant l'impression de diriger l'action.

2) Messages particuliers.

Ceci concerne les messages d'erreurs et les questions supplémentaires du genre : "Confirmez la demande (Oui-Non)?".

Le programme "PROGR" use (et abuse) de ce type de messages tandis que le second n'affiche que quelques messages d'erreurs relatifs à une syntaxe de phrase incorrecte (pour le langage d'interrogation de la banque de données).

Voici quelques caractéristiques de l'utilisation de tels messages :

a) de tels messages permettent aux concepteurs du programme d'avoir une plus grande maîtrise de ce qui est exécuté. Les risques d'erreur de manipulation peuvent être réduits fortement.
   exemple: lors de la destruction d'un médecin, un message de confirmation est envoyé. En cas de doute, cela permet d'éviter de sérieux problèmes.

b) l'utilisation de tels messages rend l'exécution du programme plus lente et plus lourde. De tels messages coupent la continuité du traitement en cours.

c) un tel système nécessite moins de responsabilités de la part de l'utilisateur.
"Si le programme se trompe, ce n'est pas de ma faute, j'ai fait ce qu'il m'a demandé.".

Il est évident que le fait de ne pas utiliser un tel système de messages produit les caractéristiques inverses.

Le choix d'utilisation d'une telle méthode peut-être motivé par le type d'utilisateur du programme et par la fréquence d'utilisation. Il est clair qu'un programme (comme "STAT") qui sera utilisé fréquemment par les médecins nécessitera nettement moins de messages.

Une autre motivation est le fait que le programme modifie la banque de données ou non. Si c'est le cas, il vaut mieux vérifier deux fois plutôt qu'une que l'opération est celle désirée. Par contre, lorsque le programme n'altère pas la banque de données (ex.: le programme "STAT"), toute erreur de la part de l'utilisateur n'est préjudiciable qu'à lui-même. Il devra seulement recommencer ce qu'il a mal exécuté.

VI. CONCLUSION.

Je pense que vu les critères émis, il est préférable de considérer le travail réalisé comme une maquette plus ou moins élaborée et non comme un logiciel prêt à l'emploi. Les programmes créés serviront de base de discussion avec les utilisateurs en vue d'une amélioration certaine du produit fini.

Voici les quatre faits qui m'ont amené à conclure cela :

-) un choix des écrans sans aucune concertation avec les utilisateurs
-) une lenteur d'exécution au niveau de la banque de données
-) une relative incomplétion de certaines phases (ex.validation)
-) un choix délibéré de ma part de supprimer certaines fonctions

Cependant, comme observé dans le point IV ci-dessus, on constate qu'il y a une certaine hiérarchisation dans le début des phases. Tous les programmes ne doivent pas commencer en même temps. On peut donc utiliser les premiers tels quels avant d'apporter des modifications aux autres. Les améliorations potentielles peuvent être développées au fur et à mesure de l'évolution du système tout entier.

P A R T I E III. :

P R O P O S I T I O N S

D' E X T E N S I O N S

LES OUTILS POSSIBLES.

Après avoir passé en revue les outils qui sont utilisables immédiatement ou à court terme, il est bon de penser à en proposer de nouveaux. Il ne s'agit pas de repartir à zéro en se lançant dans une autre voie, mais plutôt de faire évoluer le contexte actuel dans deux domaines distincts : l'étude des cas cliniques et la manipulation des symptômes et des remèdes. Le premier domaine a trait à la banque de données centralisée qui vient d'être terminée et le second touche plus particulièrement le navigateur qui fonctionne déjà depuis quelque temps.

I. EVOLUTION EN FONCTION DE LA BANQUE DE DONNEES.

L'évolution en fonction de la banque de données cliniques pourra s'effectuer en deux étapes successives :

-) la constitution d'une banque de schémas diagnostiques

-) la construction d'une famille de systèmes experts

De la réussite de la première étape dépendra la réalisation de la seconde.

Je vais donc d' abord développer quelques idées sur ces schémas diagnostiques.

A) La banque de schémas diagnostiques.

Ce paragraphe dégagera un ensemble de conditions initiales pour pouvoir appliquer la méthode décrite dans la référence (10). Ensuite viendront des indications quant à la réalisation et à l'évaluation d'une telle méthode.

- 1) définition d'un schéma diagnostique.

Etant donné une maladie, un schéma diagnostique est un ensemble de règles qui permettent d'établir une correspondance entre des symptômes observés et les remèdes qui peuvent être appliqués au patient pour cette maladie (avec un certain degré de confiance):

exemple : pour une maladie M1, la pratique a dégagé le schéma diagnostique suivant :

$S_1 \wedge S_2 \wedge S_3 \wedge S_4 \Rightarrow R_1(0,6)$	où S_1, S_2, S_3, S_4 sont des
$S_1 \wedge S_2 \wedge S_3 \wedge S_4 \Rightarrow R_2(0,2)$	symptômes observés et
$S_1 \wedge S_2 \wedge S_3 \wedge S_4 \Rightarrow R_3(0,2)$	R_1 , R_2 , R_3 des remèdes
$S_1 \wedge S_2 \wedge S_3 \wedge \neg S_4 \Rightarrow R_2(1)$	homéopatiques

Les règles du schéma diagnostique sont déduites de la banque de données cliniques sur base d'estimations statistiques.

- 2) les préalables à la constitution d'un schéma diagnostique.

Comme il est décrit ci-dessus, les règles que l'on pourrait déduire proviendraient d'inférences statistiques sur les cas pratiques contenus dans la banque de données cliniques. Il faut donc impérativement que cette banque de données soit garnie de façon conséquente. Mais que signifie conséquente? Ceci veut dire que l'échantillon que constitue la banque de données doit être suffisamment grand que pour être représentatif de la population avec une marge d'erreur minime. On serait donc enclin à attendre que la banque de données soit de taille gigantesque et donc à avoir un échantillon de taille imposante (ex. un million de cas). Cependant, le gain d'informations obtenu avec un tel échantillon par rapport à un autre de taille dix fois moindre est presque négligeable. Ceci veut dire qu'avec un échantillon de taille moyenne, on arrive à obtenir les informations espérées avec un risque d'erreur acceptable.

Par exemple : les sondages d'opinion s'effectuent en général sur un millier de personnes et donnent environ 5% d'erreur sur l'estimation.

C'est ici que trois facteurs interviennent : le temps, la place mémoire, le coût. Plus l'échantillon est grand et plus la quantité des trois facteurs précités est importante. Il convient donc de trouver un compromis : un échantillon assez grand pour avoir un pourcentage d'erreur acceptable mais suffisamment petit pour réduire les coûts d'évaluation.

Cependant, il n'est pas suffisant de se limiter à la

taille de la banque de données cliniques. En effet, il faut garder à l'esprit que c'est sur une seule maladie à la fois que l'on va se concentrer pour établir son schéma diagnostique. Donc, n'importe quelle banque de données de taille respectable ne conviendra pas nécessairement.

exemple : prenons deux banques de données cliniques de même taille : 10.000 dossiers - l'une BD_1 contient 2.000 cas distincts - l'autre BD_2 en contient 100 - BD_1 contiendra donc en moyenne 5 dossiers par cas alors que BD_2 en contiendra 100. Il est clair que BD_2 pourra être exploité statistiquement alors que BD_1 sera quasiment inutile - On ne peut rien affirmer de sérieux quand seulement cinq occurences d'un même cas se sont produites.

Il est donc important de déterminer une stratégie efficace de remplissage de la banque de données cliniques afin d'éviter du travail inutile.

De plus, il est bon de ne pas oublier que la banque de données cliniques est évolutive et qu'il convient donc de remettre en question régulièrement les schémas diagnostiques déjà construits. Ceci permettra de les conserver comme une image la plus proche de l'état courant de la banque de données.

En conclusion, il est possible de travailler statistiquement sur base d'une banque de données relativement restreinte moyennant une stratégie de remplissage efficace et la prise en compte de l'erreur d'estimation propre à l'état de la banque de données cliniques.

- 3) le choix d'une maladie.

Voici quelques critères qui m'ont paru importants dans le choix d'une maladie dont on voudrait construire le schéma diagnostique :

-) il faut que la maladie soit bien connue et qu'elle fasse l'objet de nombreuses occurences dans la banque de données cliniques. Il est inutile de commencer des études statistiques

sur une maladie dont seulement une vingtaine de cas ont été décelés et enregistrés.

-) Il faut que tous les remèdes possibles pour cette maladie puissent être relevés de façon la plus exhaustive qui soit à partir de la banque de données cliniques. Il est inutile d'étudier une maladie dont on aurait laissé de côté un ensemble de remèdes potentiels. Pour cela, il suffit de relever les remèdes disponibles dans le KENT et ceux observés dans la banque de données. La comparaison permettrait de voir si la banque de données est complète.

-) Il faut éviter de choisir des maladies particulières qui pourraient entraîner des biais dans les inférences statistiques telles que les épidémies (pour la grippe).

Je n'ai pas dressé la liste complète des critères de choix pour une maladie. J'ai simplement montré quelques contraintes d'ordre technique inhérentes aux modélisations statistiques. A ces contraintes, viennent s'en ajouter d'autres plus subjectives telles que la préférence, le choix de la stratégie, etc...

- 4) la réalisation d'un schéma diagnostique.

Une fois la maladie choisie, il faut construire les règles d'inférences qui constituent le schéma diagnostique de cette maladie. Pour cela, je propose une solution en trois parties :

- extraire les cas concernés par cette maladie de la banque de données cliniques.
- dégrossir les données brutes de la banque de données cliniques en choisissant les symptômes pertinents pour le schéma diagnostique.
- construire le schéma diagnostique.

La partie la plus délicate me semble être la seconde car elle peut faire intervenir des critères de choix, pour un symptôme pertinent, qui sont tout à fait subjectifs (tel mot frappe plus que tel autre).

- 4) - 1 - Extraction des cas concernés.

On retire systématiquement de la banque de données cliniques tous les cas de la maladie. Cette opération est très facile et autorise déjà certains traitements tels que comptage du nombre total de cas, comptage du nombre d'occurences de remèdes ou de symptômes particuliers.

- 4) - 2 - Dégrossissage et choix des symptômes pertinents.

Pourquoi effectuer un dégrossissage?

Je vois à cela deux raisons majeures :

-) les médecins jugent souvent la situation sur un ensemble d'environ dix symptômes prépondérants au maximum. Comme la recherche d'un schéma diagnostique est basée sur l'observation de la banque de données, il est préférable d'adopter une attitude similaire puisque la banque de données aura été remplie par ces médecins.

-) pour permettre une classification assez rapide, il convient de l'effectuer sur un ensemble de symptômes et de remèdes pas trop important.

A ceci, je dois ajouter deux remarques :

-) Après dégrossissage, il subsistera encore une quantité suffisante de combinaisons de symptômes et de remèdes.

-) Les règles proches pourront être dissociées grâce à une observation plus fine de ce qui a été éliminé dans un premier temps.

Le processus de dégrossissage se déroule par affinements successifs des cas bruts extraits de la banque de données cliniques. Tout d'abord, on a retiré les symptômes propres à la maladie en cours. Il ne faut pas les oublier, mais ils ne seront pas déterminants dans le choix final du remède. On les place donc à l'écart pour former en quelque sorte le titre de la maladie. Ensuite, on se limite aux symptômes repris dans la banque de données cliniques sous la rubrique " Symptômes Actuels". On néglige donc dans l'immédiat les prescriptions antérieures, les analyses et les antécédents du patient. Enfin , on se limite à un éventail de 50 à 80

remèdes les plus courants. Cette limite est implicite car on ne peut pas exiger d'un médecin homéopathe de connaître tous les remèdes : il prescrira toujours en fonction des remèdes qu'il maîtrise. Ceci ne veut pas dire qu'on laisse de côté les autres remèdes définitivement; ils seront pris en compte au niveau d'un système expert. Mais au niveau du schéma diagnostique d'une maladie, seuls les remèdes prescrits par les médecins sont retenus (ce sont les seuls dans la banque de données cliniques).

Pour ce qui est du choix des symptômes pertinents, il faudrait pouvoir tenir compte du poids d'un symptôme afin de pouvoir juger s'il a été primordial dans le choix de la prescription. Il faudrait alors se fixer une borne inférieure sur le poids des symptômes au-dessus de laquelle un symptôme est considéré par la majorité comme pertinent et déterminant. La difficulté réside dans le fait que chaque médecin a une façon personnelle d'attribuer un poids à un symptôme. La borne inférieure semble pratiquement impossible à établir de façon fixe et définitive. Il faut donc trouver une méthode qui permette de définir une borne inférieure particulière à chaque cas rencontré. Il faut pouvoir calculer une borne inférieure relative à chaque cas extrait de la banque de données (b_{rel}).

Voici donc une façon de procéder pour calculer cette borne relative :

1.- se fixer une borne inférieure absolue en pourcents valable pour tous les cas.
exemple : $b_{abs} = 30\%$

2.- pour chaque cas extrait de la banque de données :
- retenir la différence entre le poids maximal et le minimal des symptômes du cas
- multiplier cette différence par la borne absolue
- ajouter ce dernier résultat au poids minimal pour obtenir la borne inférieure relative.

exemple :

Symptômes	poids du symptôme
S_1	1
S_2	1
S_3	5
S_4	10
S_5	1
S_6	6
S_7	8

soit la borne inférieure absolue de 30%, alors :
le poids minimal est de 1,
le poids maximal est de 10,
et la différence vaut 9.

$b_{rel} = (9.30\%) + 1 \simeq 4$

seuls les symptômes 3, 4, 6 et 7 seront retenus car leur poids est supérieur à 4.

- 4) - 3 - Construction du schéma diagnostique.

Références : (10)

Une fois réalisé le dégrossissage de chaque cas, on construit une liste recouvrant tous les cas avec tous les symptômes restants $\{s_1, s_2, \ldots, s_n\}$ tel que $\forall i,j : i \neq j \Rightarrow s_i \neq s_j$ $1 \leq i, j \leq n$. On procède de même pour les remèdes et on obtient la liste $\{r_1, r_2, \ldots, r_m\}$ tel que $\forall i,j : i \neq j \Rightarrow r_i \neq r_j$ $1 \leq i, j \leq m$. Un cas particulier dégrossi pour cette maladie peut-être représenté par le couple $((s_{i_1}, s_{i_2}, \ldots, s_{i_k}), (r_j))$ avec $1 \leq k \leq n$; $\forall j : 1 \leq i_j \leq n$; $r_j \in \{r_1, r_2, \ldots, r_m\}$. Ce couple est interprété comme suit : " le patient présentant les symptômes pertinents $s_{i_1}, s_{i_2}, \ldots, s_{i_k}$ a été guéri par le remède r_j ". $(s_{i_1}, s_{i_2}, \ldots, s_{i_k})$ est appelé le complexe de symptômes et noté S_k (c'est-à-dire une combinaisons des symptômes relevés). (r_j) est appelé le complexe des remèdes et noté R_k (c'est-à-dire une combinaisons des remèdes recueillis). Un complexe de remèdes dans ce cas-ci ne contiendra qu'un et un seul remède. C'est une question de

choix homéopathique. L'homéopathe pluraliste aura un complexe de remèdes de la forme (r_{i1} ... r_{ik}). L'U.I.H.N. formant des homéopathes uniscistes, je me limiterai donc à la forme (r_{i1}). Cette restriction limite le nombre de combinaisons possibles mais ne change en rien le procédé. On peut donc adapter la méthode développée ci-après pour le pluraliste comme pour l'unisciste.

Voici, en quelques lignes, la description du processus de construction du schéma diagnostique. Pour plus de précisions, il conviendra de consulter la référence (10):

(1) relever toutes les combinaisons possibles de complexes de symptômes S_k.

(2) relever toutes les combinaisons possibles de remèdes R_k. (dans le cas qui nous occupe, il y en a m :
$R_1, R_2, \ldots R_m$ avec $R_i = (r_i)\ \forall i$

(3) construire tous les couples(S_k, R_t) possibles.

(4) pour chaque couple construit, compter ses occurences dans la banque de données dégrossie.

(5) construire le tableau correspondant :

	S_1	S_2	S_3	TOTAL
R_1	100	150	300	550
R_2	0	250	0	250
R_3	500	200	100	800
TOTAL	600	600	400	1600 ← TOTAL GLOBAL

(S_1, R_1) = 100 est interprété comme suit " 100 cas ont été trouvés dans la banque de données cliniques tels que le complexe de symptômes S_1 ait été guéri par le complexe de remèdes R_1 ".

(6) supprimer les colonnes et les lignes qui ne contiennent que des zéros.

(7) calculer le tableau des probabilités suivant :

$P(R_j$ et $S_i)$	S_1	S_2	S_3	$P(R_j)$
R_1	1/16	3/32	3/16	11/32
R_2	0	5/32	0	5/32
R_3	5/16	1/8	1/16	1/2
$P(S_i)$	3/8	3/8	1/4	1

(8) suivant la formule

$$P(R_j / S_i) = \frac{P(R_j \text{ et } S_i)}{P(S_i)}$$

calculer le tableau suivant à l'aide du point (7)

$P(R_j/S_i)$	S_1	S_2	S_3
R_1	1/6	3/12	3/4 *
R_2	0	5/12 *	0
R_3	5/6 *	1/3	1/4

(9) De (8), pour un S_i donné, le R_j conseillé (le plus probable) est celui tel que

$P(R_j/S_i) = \max_{k \in \{1,\ldots,m\}} \{P(R_k/S_i)\}$

On déduit donc les règles d'inférences suivantes pour le schéma diagnostique :

$S_2 = (s_{2_1}, s_{2_2}, \ldots, s_{2_n}) \Rightarrow R_2$ (5/12); R_3(1/3); R_1(3/12)

$S_1 = (s_{1_1}, s_{1_2}, \ldots, s_{1_b}) \Rightarrow R_3$ (5/6)

$S_3 = (s_{3_1}, s_{3_2}, \ldots, s_{3_t}) \Rightarrow R_1$ (3/4)

Pour l'exemple ci-dessus, il conviendrait d'affiner le jugement pour le complexe de symptômes S_2. Pour cela, il faut faire intervenir les symptômes mis de côté, de même que les divers antécédents du patient pour tenter une classification plus fine. En effet, le résultat obtenu tel quel signifie que le complexe de symptômes S_2 n'est pas suffisamment discriminant. Il faut donc trouver des symptômes ou des caractéristiques spécifiques à la prescription de R_1, R_2 ou R_3 .

- 5) Tests de validité d'un tel modèle.

Références : (11)

Il est important de vérifier si, malgré les précautions prises, les données sont utilisables statistiquement. Pour cela, je vois trois types de tests à effectuer:

1) Un test d'hypothèse sur l'indépendance des complexes de symptômes par rapport aux complexes de remèdes. Si l'hypothèse d'indépendance ne peut pas être rejetée, cela signifie qu'on se trouve dans le cas S_2 de l'exemple du paragraphe précédent pour tous (ou presque tous) les S_i. A ce moment, il est plus facile de tirer un remède au sort plutôt que d'effectuer tous les calculs précédents.

2) Un test d'hypothèse sur la distribution de la probabilité des complexes de symptômes par rapport à chaque complexe de remèdes. Si tous les remèdes sont identiquement distribués, cela revient à dire que tous les complexes de remèdes sont équivalents et qu'on peut se limiter à un seul complexe de remèdes pour n'importe quel complexe de symptômes.

3) Ce dernier test est, à mon avis, le plus important. Il provient du fait que la banque de données cliniques est évolutive. Il faut donc remettre régulièrement en question les schémas diagnostiques d'une maladie. A un schéma diagnostique correspond une distribution de probabilités (cf. tableau (7) Paragr.précédent). Il faut tester si les nouveaux cas introduits depuis la création du schéma diagnostique n'ont pas perturbé de façon significative cette distribution de probabilités. Il faudra donc tester la nouvelle grille obtenue pour vérifier si elle suit la loi de probabilité précédemment établie. Si tel est le cas, le schéma diagnostique peut être considéré comme adéquat pour les nouvelles données. Sinon, il faut reconstruire une nouvelle table des probabilités ((7) Paragr. précédent) qui tiendra compte de l'évolution. De là, on pourra constituer les nouvelles règles d'inférence du schéma diagnostique.

Ces trois types de tests sont de grands classiques des statistiques et sont appelés les "tests d'hypothèse". Le principe est le suivant :

" On émet une hypothèse sur des observations et on la compare conformément à une loi χ^2. Le résultat dit si l'hypothèse peut être rejetée ou non en fonction d'une marge d'erreur possible."

Ces types de tests sont fréquemment utilisés et interviennent dans tout cours de statistiques. J'ai donc jugé inutile de développer ce sujet.

- 6) Interprétation des résultats des tests.

Dans le plus mauvais des cas, le test 2) du paragraphe précédent ne rejette pas l'hypothèse des distributions identiques. C'est pratiquement impossible du point de vue homéopathique : un remède ne peut pas suffire pour tous les patients. Il y a donc un biais dans l'échantillon des cas. On ne peut pas travailler sur base de ces chiffres.

Si le premier test ne rejette pas l'hypothèse d'indépendance, il faut prendre en considération les symptômes éliminés ainsi que les antécédents du patient. Le dégrossissage a peut être été trop important et on a perdu beaucoup de relations nécessaires. Il faut donc revoir la stratégie du paragraphe -4) - 2 - de façon à être plus souple et permettre par là un accroissement du nombre de complexes de symptômes.

Remarque : Cet accroissement ne doit pas être trop important, il faut rester dans les limites de faisabilité de l'ordinateur.

Le troisième test, quant à lui, ne pourra être réalisé que si un schéma diagnostique a déjà pu être établi et que la banque de données cliniques continue à évoluer pour la maladie en question. Ce dernier test peut contraindre à modifier les règles d'inférence du schéma diagnostique dans certaines proportions. Je pense qu'un certain effet de stabilisation ne tardera pas à

être observé au fur et à mesure que la quantité de données collectées sera grande. La nécessité de ce test deviendra moins évidente et le dernier schéma diagnostique établi pourra être considéré comme le meilleur possible en fonction d'une certaine marge d'erreur. S'il n'y a pas d'effet de stabilisation, il faut voir si la stratégie de remplissage de la banque de données cliniques est adéquate. Par exemple, si on rentre toute une série de cas exceptionnels et, ensuite, si on insère les cas les plus courants, cette irrégularité risque de se trouver répercutée au niveau des règles d'inférence si on a effectué le processus de construction des règles entre ces deux opérations.

- 7) Conclusion.

Si, pour une maladie, tous les tests sont réussis et si le schéma diagnostique est stabilisé, on peut alors envisager de construire un système expert concernant cette maladie. Une grande partie de la connaissance sur celle-ci sera constituée par les règles d'inférence du schéma diagnostique.

En résumé, il faut, en préalable à l'élaboration d'un système expert pour une maladie :

- récolter suffisamment de cas sur cette maladie
- construire un modèle qui permette de déduire des règles d'inférence
- tester ce modèle pour valider ces règles.

Il me semblerait un peu hâtif de commencer la réalisation d'un système expert sans avoir tout d'abord considéré la faisabilité d'un tel système : c'est le but principal de la constitution d'une banque de schémas diagnostiques.

En attendant que le système expert relatif à la maladie soit terminé, l'ensemble des règles du schéma diagnostique peut déjà être utilisé par les médecins comme support à leur décision. Il suffirait de construire un traducteur qui permettrait d'obtenir les règles en Français, par exemple.

B) Un Système expert.

Réf. : de (12) à (28)

A ce stade, on suppose que tout a marché correctement au niveau du schéma diagnostique d'une maladie particulière. On peut donc commencer une première approche pour un système expert pour cette maladie. Mais n'est-ce pas un peu trop tôt pour aborder ce problème? La banque de données cliniques ne contient encore aucun cas à l'heure où sont rédigées ces lignes. On ne sait donc pas s'il existera une maladie à partir de laquelle on pourra déduire des règles d'inférence.

Je pense que ce qui suit n'est pas inutile. En effet, il me semble intéressant de préparer le terrain pour un tel projet. La réalisation d'un système expert est une tâche de longue haleine qui nécessite de nombreux efforts. Ce qui suit devrait permettre de se rendre compte de la difficulté d'une telle entreprise.

Je n'ai pas la prétention d'apporter une proposition de solution pour un tel système ni l'architecture de sa réalisation. Je me bornerai à en dégager des caractéristiques importantes et à en proposer des lieux de choix. Je montrerai un ensemble de modules indispensables selon ce que j'ai pu observer sur des systèmes existants.

- 1) Définition d'un système expert.

Un expert est une personne consultée pour sa connaissance approfondie d'un problème qu'elle est à même de résoudre et d'expliquer. Donc, un système expert est un programme qui simule le comportement d'un expert humain dans le domaine concerné et qui possède donc les quatre propriétés suivantes :

-) il est capable de manipuler des connaissances
-) il sait expliquer le raisonnement qu'il a suivi
-) ilpeut étendre et adapter ses connaissances
-) il est apte à communiquer avec l'utilisateur dans un sous-ensemble de sa langue naturelle.

Ces quatre propriétés seront développées ultérieurement au niveau des qualités d'un bon système expert.

Le système expert remplace donc l'expert humain dans sa fonction de conseiller.

- 2) Les constructeurs d'un système expert.

N'importe qui ne peut pas, du jour au lendemain, décider de construire un système expert en homéopathie. Quel que soit le domaine d'application, la construction d'un tel système nécessite la collaboration étroite d'une équipe d'informaticiens et d'un ou plusieurs experts du domaine concerné. En effet, il faut une compréhension très profonde à la fois du domaine mais aussi de l'informatique. Dans certains cas, des experts seuls suffiront si toutefois ils sont d'excellents informaticiens, mais jamais un seul groupe d'informaticiens ne suffira sauf dans le cas particulier où le domaine d'expertise est l'informatique. (ex. R_1 système conçu par Digital Equipment pour l'aide à la configuration d'architectures de machines). Il est à remarquer que, par leur étroite collaboration, les experts deviendront de bons informaticiens et les informaticiens des experts du domaine en question.

La collaboration est nécessitée par la bonne compréhension que chacun doit avoir des problèmes et des solutions que l'expert peut rencontrer ainsi que des contraintes techniques auxquelles l'informaticien est soumis.

- 3) Les qualités d'un système expert.

Voici quelques unes des qualités qu'un bon système expert doit posséder dans une certaine mesure.

- 3) - 1 - Il est capable de manipuler ses connaissances.

Il ne doit pas se limiter à utiliser ses connaissances telles quelles comme ferait un programme habituel avec une banque de données. Le système doit pouvoir inférer de nouvelles connaissances à partir de ce qu'il sait déjà.

exemple : connaissance initiale: -) une pièce a 4 murs
-) ma chambre est une pièce
connaissance inférée : -) ma chambre a 4 murs

Donner au système des possibilités d'inférence permet de réduire la taille des connaissances initiales indispensables. En effet, il peut retrouver facilement des règles particulières

en déduisant à partir de ses bases de départ.

- 3) - 2 - Il est utile.

Il me semble ridicule de dépenser une grosse quantité d'efforts à mettre au point un système qui serait sous-employé ou dont l'utilité serait à remettre en question. Il faut qu'il existe un besoin par rapport à ce système et il faut que celui-ci réponde au besoin de la façon la plus satisfaisante possible, c'est-à-dire que l'utilisateur potentiel a besoin d'une aide experte dans un certain domaine et il faut que l'aide fournie soit adéquate et suffisante.

- 3) - 3 - Il procure un enseignement approprié.

Le besoin de consulter un système expert provient du fait que l'utilisateur ne maîtrise pas suffisamment le domaine pour prendre une décision seul. En plus de donner un conseil sur la décision à prendre, le système expert doit également pouvoir apprendre à l'utilisateur ce qu'il ne savait pas. Le système doit cependant garder une certaine flexibilité à ce niveau. L'utilisateur doit apprendre à son propre rythme. C'est donc lui qui décidera quand et ce qu'il veut apprendre.

- 3) - 4 - Il fournit une explication adéquate de son raisonnement.

Un système expert n'est pas un programme classique (ce dernier est le résultat de la maîtrise d'un processus dont on a voulu accélérer le déroulement). Pour le cas qui nous occupe, l'utilisateur ne maîtrise pas du tout le processus. Il faut donc que le programme soit capable d'expliquer son propre déroulement. Il faudrait également qu'il l'explique comme un expert humain le ferait. Donc, le "raisonnement" du programme doit être l'image du raisonnement de l'expert.

- 3) - 5 - Il communique en langage naturel.

La communication se fera plutôt dans un sous-ensemble du langage naturel, sur base de mots-clés spécifiques au domaine.

On ne peut pas exiger que le système entretienne n'importe quelle conversation avec l'utilisateur, ni qu'il puisse construire une syntaxe complexe. On lui demande seulement de pouvoir articuler une phrase simple autour d'un mot significatif. Il doit pouvoir également traiter certaines fautes d'orthographe (pour pouvoir gagner du temps) :

exemple : l'utilisateur entre le mot "ARSNICUS"
le programme doit directement comprendre qu'il s'agit de "ARSENICUM"

- 3) - 6 - Il acquiert de nouvelles connaissances.

L'expert humain sait faire évoluer sa base de connaissances: il réfléchit et il apprend par d'autres experts. Un système expert doit pouvoir agir de façon plus ou moins similaire. Il doit être capable de produire de lui-même de nouvelles règles sur base de ses connaissances actuelles et des utilisations qui en sont faites. Mais il doit aussi être capable de recevoir de nouvelles connaissances provenant d'experts humains autorisés. L'acquisition de ces nouvelles connaissances ne doit pas se limiter à un rangement adéquat des nouvelles règles; le système doit aussi vérifier si elles sont cohérentes dans le contexte actuel de travail.

exemple : connaissance actuelle:
"Tout ce qui est rare est cher"
nouvelle connaissance :
" un appartement bon marché est rare"
incohérence car le système (par sa qualité - 1 -)déduit:
" un appartement bon marché est cher"

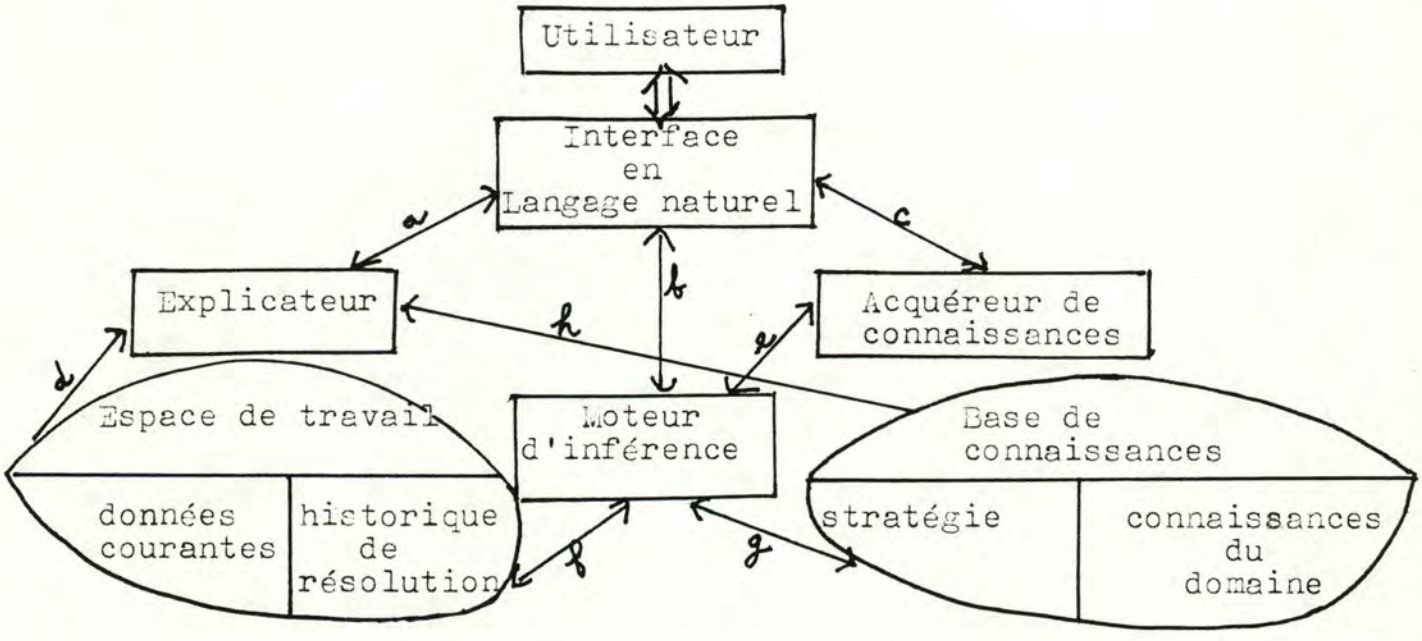
- 3) - 7 - Il est facile à modifier.

En vue d'améliorer ses performances facilement, les fonctions du système expert sont fortement modularisées. On peut ainsi modifier telle ou telle partie sans devoir reconsidérer l'ensemble.

Les SEPT qualités précitées se retrouvent dans tout système expert à des degrés divers. Tel système favorisera l'acquisition de nouvelles connaissances alors que tel autre mettra l'accent sur une bonne communication. A partir de ces sept qualités, je dégagerai l'architecture générale qui suit.

- 4) Une architecture générale.

L'architecture qui suit n'est pas spécifique au système expert qu'on pourrait concevoir en homéopathie. C'est plutôt un agencement des modules que de tels systèmes possèdent et qui peuvent être déduits des qualités décrites précédemment. Cette architecture sera approfondie au fil des paragraphes qui suivront. C'est dans ces paragraphes que je montrerai quelques choix et certaines difficultés inhérentes à chaque module.



Voici, en quelques mots, une explication des différents modules :

1° l'interface en langage naturel:
c'est le passage obligatoire pour toute communication entre l'utilisateur et le système

2° l'explicateur :
est le module qui permet au système d'expliquer son raisonnement et d'apprendre de nouvelles choses à l'utilisateur

3° l'acquéreur de connaissances :
reçoit de nouvelles règles d'un utilisateur privilégié ou les construit lui-même

4° le moteur d'inférence :
constitue le coeur du système. C'est lui qui opère les

déductions à partir de la base des connaissances. Il est également la plaque tournante du système. Il importe donc qu'il soit réalisé de façon efficace.

5° L'espace de travail :
contient les données qui ont été récoltées via l'utilisateur ainsi que l'historique de la résolution du problème (les étapes déjà réussies dans l'approche de la solution finale).

6° La base de connaissances:
est formée des connaissances pures sur le domaine concerné et de la stratégie de résolution qui est greffée sur ce domaine.

Je dois maintenant expliquer les relations qui lient les différents modules. Les flèches sur ces relations indiquent la direction que peut prendre le flux d'informations entre deux modules :

-) les relations a, b, c fournissent les informations brutes à l'interface en langage naturel pour que celle-ci les traduise à l'utilisateur. La réciproque est aussi valable.
-) les relations d et h montrent que l'explicateur tire ses sources à la fois de la base des connaissances et de l'espace de travail pour répondre à l'attente de l'utilisateur.
-) la relation e montre que l'acquéreur de connaissances utilise le moteur d'inférence pour vérifier la cohérence des nouvelles règles.
-) les relations f et g montrent que le moteur d'inférence est le seul module qui puisse à la fois consulter et mettre à jour la base de connaissances et l'espace de travail.

- 5) Des choix au niveau d'un module.

Maintenant que l'architecture générale est mise en place, je peux apporter plus de détails au niveau de chaque module qui la compose.

- 5) - 1 - La base des connaissances.

Je vais, dans ce paragraphe-ci :

- indiquer un ensemble de représentations de connaissances les plus couramment employées dans les systèmes experts,
- soulever un problème inhérent aux représentations,
- tenter de montrer une voie à suivre que W. CLANCEY a utilisée dans "NEOMYCIN" pour résoudre ce problème.

a) les représentations disponibles

Voici donc pour commencer quelques modes de représentation de connaissances. Il est conseillé, pour plus de précisions, de consulter à ce sujet les références (12) à (21).

a-1) les règles de production.

-) présentation : les connaissances du système expert sont représentées par un ensemble de règles de la forme

si P_1 et P_2 et ..... et P_n
alors A_1 et A_2 et .... et A_m
sinon A_{m+1} et .... A_p

où les P_i sont les prémices et les A_i sont les conclusions ou les actions à effectuer en fonction des prémices.

-) interprétation : si tous les P_i sont vrais, alors exécuter $A_1 \longrightarrow A_m$ sinon $A_{m+1} \longrightarrow A_p$

-) réalisation des inférences pour un tel système :

Les règles qui forment la base de connaissances du système expert sont ordonnées suivant leur importance.

Le système se trouvant dans un état intermédiaire, comment peut-il faire pour progresser vers l'état final?

1) il va regarder si l'environnement de travail lui permet d'appliquer certaines règles (en vérifiant si les prémices sont ou non établies)
2) il va choisir, suivant des critères préétablis (ordre,..) une des règles qu'il a sélectionnée ci-dessus
3) il va appliquer la règle choisie pour modifier l'état intermédiaire de son espace de travail.

-) remarque :

Ce sytème de représentation des connaissances est celui qui est le plus ancien et le plus répandu. Ceci est dû au fait qu'il répond à la façon habituelle de décider au niveau d'un programme :

Condition $\Longrightarrow$ Action

a-2) le calcul des prédicats.

-) présentation : le calcul des prédicats ajoute les notions d'arguments et de quantificateurs au calcul propositionnel habituel. Les expressions de la forme :
règle : $\forall$ x : P(x) $\Longleftarrow$ y: Q (x,y) et R (y)
ou
fait: P (x_o) c'est-à-dire P est vrai pour la valeur particulière x_o

-) interprétation : pour vérifier si P est vrai avec la valeur particulière "a", il y a deux possibilités :
soit x_o = a et alors P(x_o)
soit il faut vérifier si $\exists$y: Q (a,y,) et R (y)
Un prédicat est un ensemble d'expressions de cette forme qui contiennent toutes P (x) dans le membre de gauche.
exemple a-2 :
1) voiture (x) $\Longleftarrow$ a-quatre-roues (x)
2) voiture (x) $\Longleftarrow$ a-un-coffre (x) et a-un-moteur (x)
3) voiture (Fiat)
4) voiture (Renault)
ce prédicat signifie que , pour vérifier si un élément est une voiture, il suffit de regarder si c'est soit une Fiat, soit une Renault
ou si l'élément a quatre roues
ou si l'élément a un coffre et un moteur
Ceci suppose qu'on sache déterminer si un élément a quatre roues ou un coffre ou un moteur

-) réalisation des inférences :
le but à tout moment est de vérifier si un prédicat donné avec une valeur particulière est vrai ou non dans le

contexte de travail.

exemple : est-ce que voiture (Peugeot) sachant que a-quatre-roues (Peugeot)?

Pour cela, il faut :

-) regarder si la valeur permet d'obtenir un fait qui est toujours vrai, ce qui n'est pas le cas avec l'exemple

-) si aucun fait n'est vérifié, il faut considérer les règles de résolution, pour cela :

1° on choisit une règle suivant une stratégie préétablie

2° si tous les prédicats du membre de droite par rapport à $\Leftarrow$ sont vérifiés en remplaçant les arguments par leurs valeurs trouvées, alors le prédicat initial est vrai

exemple: - on choisit la règle n°1 de l'exemple a-2) précédent
- on remplace x par Peugeot
- on a donc que Peugeot est une voiture si Peugeot a-quatre-roues. Ceci étant vrai, Peugeot est une voiture.

-) remarque :

Ce mode de représentation de connaissances devient de plus en plus courant grâce à l'extension du langage PROLOG basé sur le calcul des prédicats.

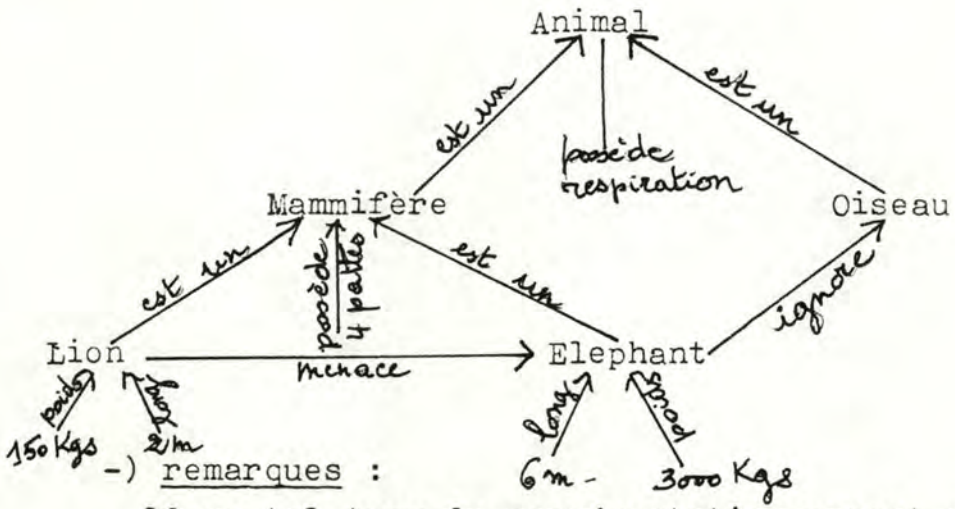
a-3) Les objets structurés.

Le paragraphe sur les objets structurés se limitera à une présentation du principe général, suivie de quelques remarques. En effet, sous le vocable "Objets structurés" se regroupent plusieurs types de représentations : les graphes, les réseaux sémantiques, les frames, etc...

-) Principe général.

On construit une sorte de toile d'araignée composée des éléments à retenir sur le domaine. Chaque élément est uni à un ou plusieurs autres par une relation utile au système.

Cette façon de procéder amène un schéma général du même genre que celui-ci pour un réseau sémantique.



-) remarques :

1° un tel type de représentation permet une description très précise du domaine concerné grâce à la quantité importante de relations qui peuvent exister entre chaque objet. Ce type de représentation est donc très utile pour un domaine simple mais qui demande une précision élevée : par exemple, la reconnaissance des formes dans un environnement restreint.

2° la précision mentionnée ci-dessus entraîne un niveau de complications élevé lorsqu'il s'agit d'ajouter un élément au système et de le relier aux précédents.

3° la propriété d'héritage est souvent très utile pour obtenir rapidement des informations pertinentes sur un élément. Cette propriété permet à un élément d'avoir toutes les caractéristiques de son supérieur hiérarchique. Il en va également de même pour ce dernier. On a donc, par exemple : le lion est un mammifère, il possède quatre pattes et peut respirer.

a-4) Les logiques déviantes et les logiques étendues.

Ces deux types de logiques sont plus récentes donc nettement moins utilisées.

Les logiques déviantes ont pour principe qu'il existe autre chose que le vrai et le faux. Ces nuances apportent des complications pour la formalisation et la construction de telles logiques.

Les logiques étendues apportent une dimension supplémentaire aux logiques précédentes. Elles permettent de tenir compte de ce que le système "croit savoir" ou qu' "il est possible qu'il sache" alors que les logiques classiques imposaient de savoir ou de ne pas savoir quelque chose. La nuance est sur la façon plus ou moins bonne de savoir quelque chose (même si quelque chose n'est pas bien connu) et non pas de savoir quelque chose plus ou moins bien.

Ces logiques semblent assez rébarbatives et je ne me sens pas capable d'aller plus loin dans une explication. Certains éléments supplémentaires sont disponibles dans les références: (13) et (21)

b) Conclusion.

L'éventail de choix dans la représentation des connaissances est suffisamment large que pour permettre la construction de bon nombre de systèmes experts. Il ne faut cependant pas perdre de vue que la réalisation préalable d'une banque de schémas diagnostiques a fourni un ensemble de règles qui se rapproche fortement d'un système de règles de production. Il faut donc penser à s'économiser du temps de travail inutile.

De plus, le choix de plusieurs modes de représentation dans un même système peut garantir une certaine efficacité.

c) Problème inhérent à la représentation.

Une qualité d'un bon système expert est sa capacité à expliquer son raisonnement pour pouvoir apprendre de nouvelles choses à l'utilisateur. Quand l'utilisateur demande pourquoi le système a agi ainsi, ce dernier doit pouvoir fournir une réponse satisfaisante tant au point de vue opérationnel qu'au niveau de l'intention générale de l'action.

Le problème provient du fait que, dans tout mode de représentation, on trouve des connaissances implicites. Elles proviennent de certains choix qui ont dû être effectués.

exemple :

Prenons un ensemble de règles de production :

1) si P_1 et P_2 alors A_1
2) si P_1 et P_2 et P_3 alors A_2 et A_3
3) si P_1 et $\neg P_2$ alors A_2 et A_4

La façon dont est organisé cet ensemble impose une stratégie d'action implicite :

1) la règle n°1 sera considérée en premier
   L'ordre des règles est préétabli dans une intention précise mais implicite : la règle n°1 est la plus importante de cet ensemble
2) dans chaque règle, on vérifiera P_1 avant les autres prémices. la condition P_1 est plus restrictive que les autres et permet un gain de temps si elle est fausse d'emblée : l'ordre des prémices est imposé par une stratégie particulière mais inexplicable telle quelle.

Le système expert ne sait pas expliquer pourquoi une règle se trouve avant une autre. Il ne peut pas découvrir une connaissance qui est contenue dans son mode de représentation. Donc, un système expert se limitera à n'expliquer que ce qui est représenté. L'utilisateur doit admettre tel quel tout ce qui est implicite pour le système expert.

d) Solution possible.

Références : (22) à (26)

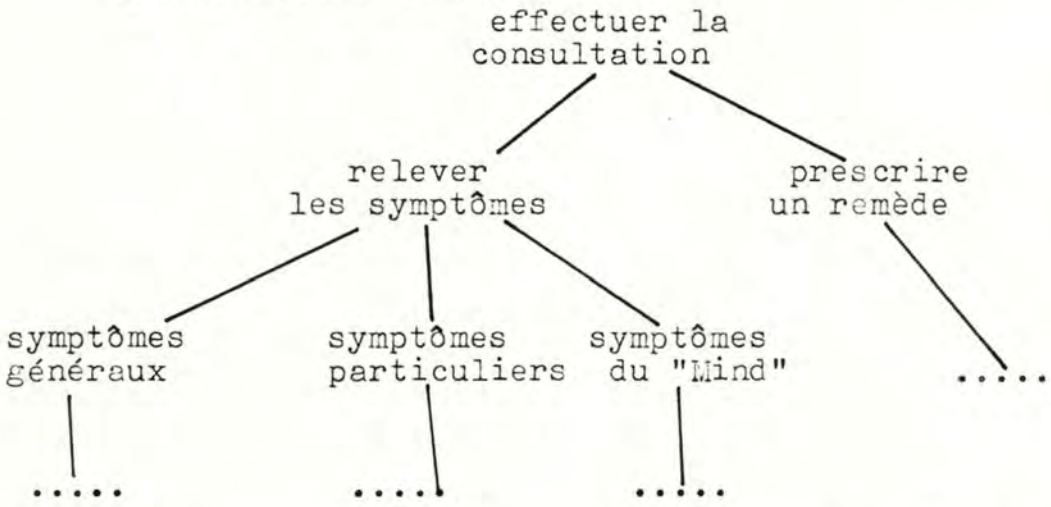
Il faut permettre au système expert de pouvoir expliquer le plus possible. L'idée de base de la solution telle qu'elle fut élaborée par W.CLANCEY pour NEOMYCIN est la suivante : ajouter un niveau d'abstraction supérieur pour les connaissances dans lequel on rejettera tout ce qui est implicite du niveau inférieur. La base de connaissances sera dissociée en deux parties :

- la première contiendra toujours les connaissances propres au domaine
- la seconde sera plus générale et contiendra les connaissances sur la stratégie générale de résolution. Elle contiendra, au minimum, tout ce qui était implicite auparavant.

Le niveau stratégique contiendra également des connaissances implicites mais elles seront admises et connues par tous les utilisateurs. Il sera donc inutile de réitérer le procédé. Un niveau d'abstraction semble suffisant.

Dans le cas qui nous concerne, la connaissance pure proviendra en grande partie du schéma diagnostique de la maladie en question auquel on pourra ajouter certaines connaissances provenant d'un expert homéopathe. Pour ce qui est de la stratégie, il faudrait considérer les choses comme suit :

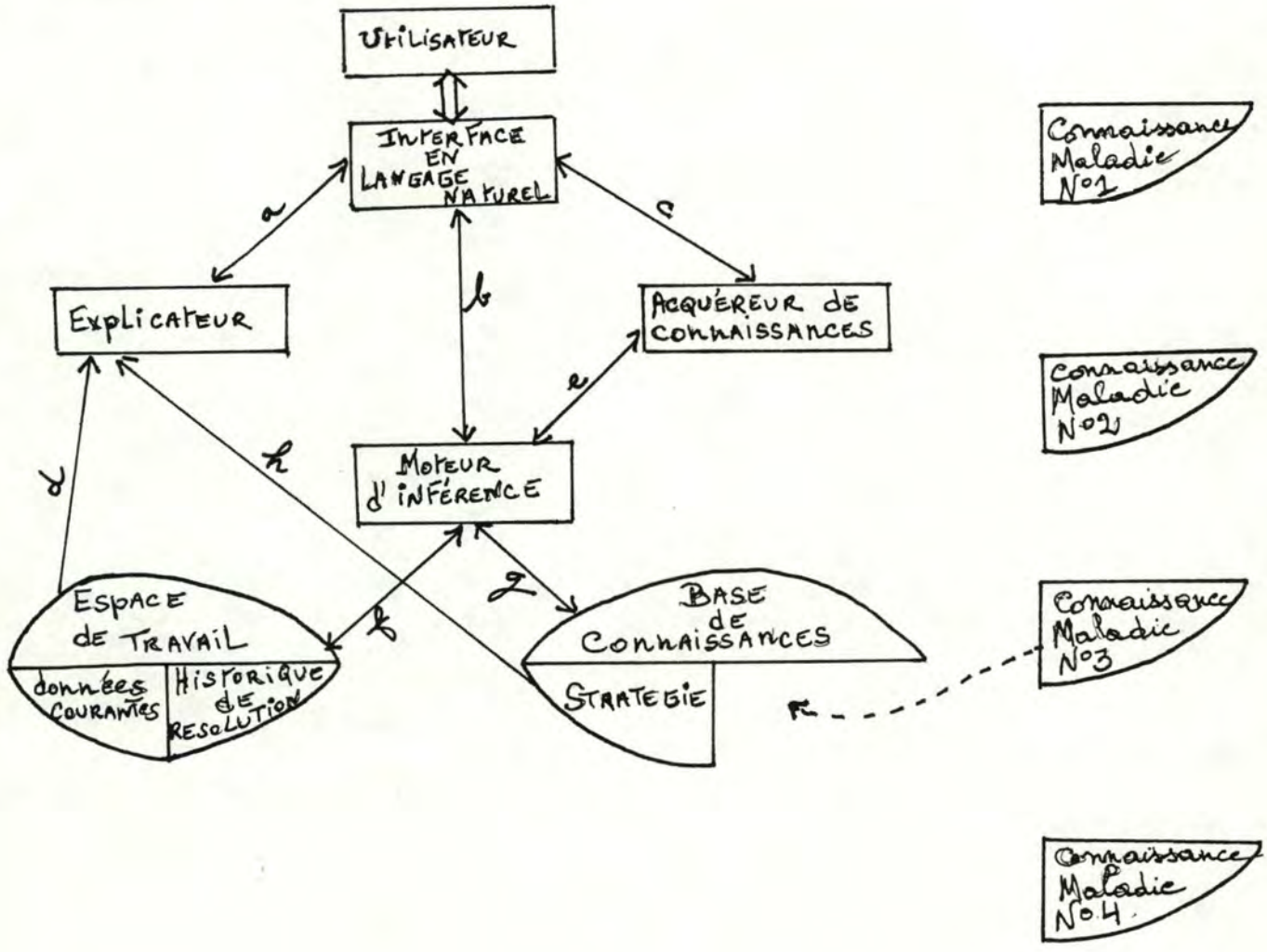
-) constituer une hiérarchisation des tâches stratégiques à effectuer. Ces tâches devront être dégagées par l'expert et constitueront la partie implicite du système.
exemple de hiérarchie :



-) pour chaque tâche de la hiérarchie, constituer des métaconnaissances. Celles-ci permettent de déterminer quel sous-ensemble des connaissances du domaine est utile à un instant donné.
exemple de métaconnaissance pour un système de règles de productions: si l'ensemble des symptômes a été récolté, alors appliquer les règles qui permettent de déduire un remède possible.

Cette façon de procéder, permet, dans ce cas-ci, de construire un système adaptable. Une fois que l'expert homéopathe aura établi la stratégie de résolution, elle sera bonne pour

toutes les maladies considérées. On aura donc un système général auquel on adaptera, en fonction des besoins du moment, les connaissances spécifiques à une maladie. Cette situation peut être exprimée par le schéma suivant :



Ce schéma indique que l'utilisateur a choisi de consulter le système expert pour la maladie n°3. Il faut donc charger dans le sytème les connaissances sur la maladie choisie.

- 5) - 2 - L'espace de travail.

L'espace de travail d'un système expert comprend deux catégories d'informations :

-) les données courantes

-) l'historique de résolution

Cet espace de travail permet au système de retenir à la fois les informations qui lui seront indispensables pour progresser ainsi que l'enchaînement de ses propres actions pour pouvoir les expliquer à l'utilisateur.

a) Les données courantes.

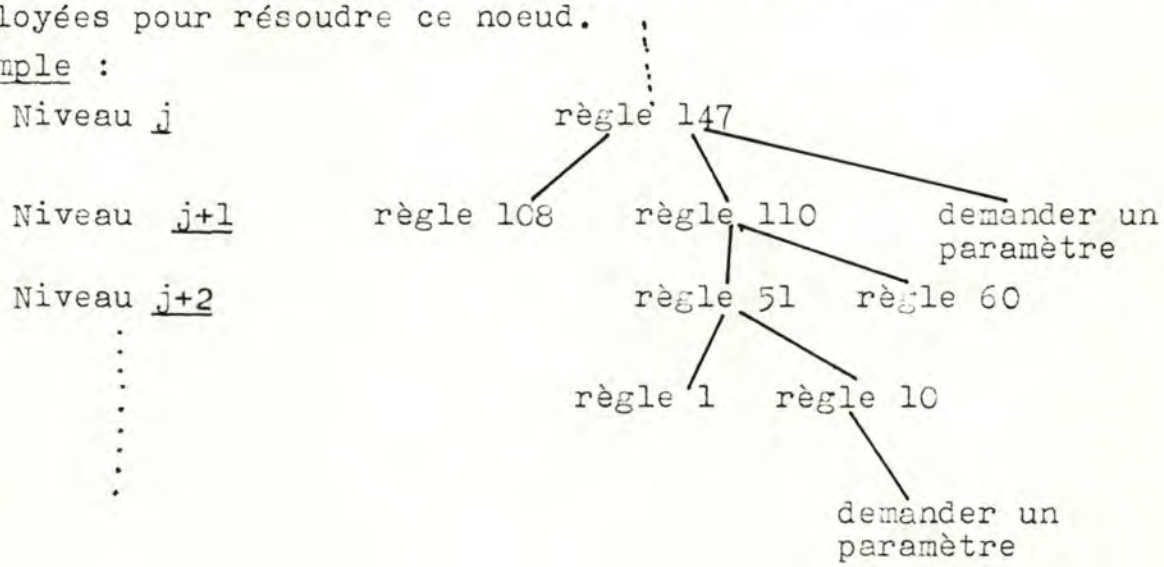
Il est important de déterminer les objets sur lesquels va travailler le système expert. Ces objets doivent être établis sur base des concepts utilisés dans la connaissance. Le nom de l'objet et l'objet lui-même sont peu importants. Ce qui prime, ce sont les propriétés qui décrivent ces objets. Ce sont principalement ces dernières qui interviendront dans l'évolution du problème. Un bon modèle pour établir l'espace des données courantes semble être le schéma classique "Entité/Association" expliqué par Mr. Fr.BODART dans son cours d' "Analyse fonctionnelle" (réf.12). Les entités représenteront les objets, et les attributs, les propriétés descriptives. Je ne m'attarderai pas plus longtemps sur ce point vu que ce n'est qu'une proposition de choix et non une analyse détaillée de la représentation de l'espace de travail. Le seul point obscur restera la façon de représenter physiquement ces données. Tout dépendra du choix du langage pour le moteur d'inférence comme on le verra plus loin.

b) L'historique de résolution.

Le système doit pouvoir retrouver les connaissances qu'il a utilisées au cours de sa résolution, mais aussi la façon dont ces connaissances étaient liées entre elles ainsi que leur enchaînement. Il faut donc enregistrer dans l'historique, d'une part, des marques particulières (nom de règle, de prédicat,...) qui permettent de retrouver les connaissances

et d'autre part, des relations entre ces connaissances. On pourrait donc avoir, pour chaque tâche de la stratégie, un arbre de contextes. Les noeuds représenteraient les connaissances; les arcs lieraient chaque noeud aux connaissances employées pour résoudre ce noeud.

exemple :



Cet exemple a rapport à un ensemble de règles de production. La réalisation du sous-objectif j-1 d'une tâche a nécessité l'utilisation de la règle 147. Cette règle 147 est devenue un sous-objectif du niveau j qui a été réalisé par les règles 108 et 110 et par la demande à l'utilisateur d'un paramètre particulier. L'objectif exprimé par la règle 108 n'a pas nécessité le raffinement en sous-objectifs du niveau j+2 tandis que la règle 110 a fait intervenir d'autres règles.

- 5) - 3 - Le moteur d'inférence.

Le moteur d'inférence est le programme qui permet de réaliser un objectif en découvrant et en déclenchant des sous-objectifs adéquats. Ce programme réalisera les inférences nécessaires pour passer d'un état à un autre en manipulant la base des connaissances. Le programme devra avant tout être le plus performant en demandant le moins possible à l'utilisateur. Il faut donc que, se trouvant devant un paramètre inconnu, il essaie, d'après ce qu'il sait, d'en estimer la valeur. S'il se trouve dans l'incapacité de déduire lui-même une telle valeur ou si celle-ci est trop imprécise, il pourra

alors demander plus de précision à l'utilisateur.

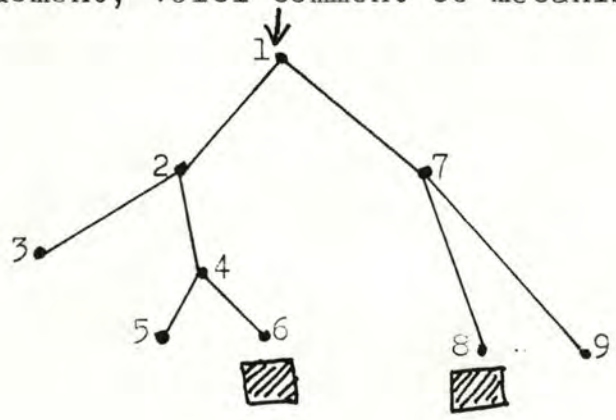
Pour le fonctionnement proprement dit, les inférences peuvent s'effectuer de trois façons différentes :

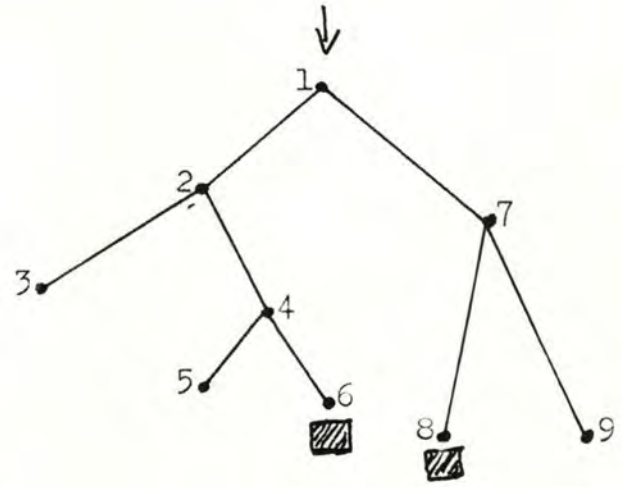
-) le chaînage arrière : partant de l'hypothèse initiale, on découvre au fur et à mesure des sous-objectifs, récurcivement, jusqu'à obtenir des faits qui confirment ou infirment l'hypothèse de départ.
-) le chaînage avant : partant des faits, on les généralise en sous-objectifs jusqu'à obtenir une hypothèse finale cohérente.
-) le mélange des deux : on récolte des informations initiales générales. De là, on formule une hypothèse. Ensuite, on teste cette hypothèse en récoltant de nouvelles données qui permettent de la raffiner, et ainsi de suite jusqu'à l'établissement d'une hypothèse définitive.

Le choix d'un type particulier est fortement lié à la façon dont la base de connaissances a été conçue. Le mélange de chaînages avant et arrière apparaît surtout dans les systèmes experts où stratégie et connaissances du domaine sont dissociés. Ceci est dû au fait que la stratégie est une suite de tâches à réaliser et présente une certaine discontinuité propre à ce type d'inférence.

On peut également disposer d'un mécanisme particulier appelé "Backtracking". Ce mécanisme permet d'envisager toutes les solutions possibles d'un problème et de revenir en arrière si le test d'une solution s'avère négatif.

Schématiquement, voici comment ce mécanisme fonctionne :





Partant de 1., le système doit aller chercher les solutions en 6. et 8. Pour cela, il va descendre en 3. en passant par 2. Se rendant compte que 3. ne contient pas de solution, il effectue alors le "backtracking" :

-) il revient au noeud précédent (2)
-) il envisage les alternatives à partir de ce noeud
-) s'il y en a, il s'y engage
-) s'il n'y en a pas, il remonte au noeud précédent.

Le système effectuera donc le parcours d'exploration suivant :

→ 1.- 2.- 3.- 2.- 4.-5.-6.-4.-2.-1.-7.-8.-7.-9.-7.-1. → fin
solution solution

Ce mécanisme est surtout utilisé par l'interpréteur PROLOG. Il a le défaut d'être extrêmement lent.

Pour ce qui est du choix d'un langage de programmation d'un moteur d'inférence, tous les choix sont permis mais liés au choix effectué pour la représentation des connaissances.

- 5) - 4 - L'Explicateur.

Ce module ne doit pas agir de lui-même : il ne peut pas se permettre de tout expliquer au fur et à mesure à l'utilisateur au risque de faire durer le programme trop longtemps. De plus, certaines explications pourraient être redondantes avec ce que l'utilisateur sait déjà. Il faut donc laisser l'initiative à ce dernier : il activera le module explicateur

lorsqu'il en aura besoin ou lorsqu'il en aura envie.
Plusieurs stratégies d'utilisation d'un tel module sont possibles et il faudra en tenir compte lors de la conception :

- l'utilisateur appelle le module au moment même où il a besoin de se faire expliquer quelque chose. Cette stratégie permet de limiter la taille de l'historique de résolution et de constater immédiatement si l'utilisateur est dans le bon.
- l'utilisateur appelle le module en fin de session du système expert. Il faut donc retenir toutes les opérations réalisées lors de cette session pour pouvoir en expliquer l'une ou l'autre. Il faut même prévoir de les enregistrer pour permettre une explication à plus long terme. Cette stratégie est surtout efficace lorsque le système expert doit être consulté pour prendre une décision rapide et correcte. L'explication du raisonnement n'est pas requise immédiatement, elle peut donc être différée. L'utilisateur peut, par après, revenir sur le cas et demander des renseignements.
- cette dernière stratégie est la conjonction des deux précédentes : elle permet une explication immédiate et/ou différée. L'espace de travail sera le même que pour la stratégie précédente mais le module explicateur sera plus complet : il devra intervenir à deux types d'instants différents bien que ce soit le même genre d'explication. Il faudra, par exemple, manipuler un pointeur sur la situation courante.

Quelle que soit la stratégie choisie, l'utilisateur devra pouvoir poser deux types de questions :

- sur la connaissance : il pourra demander au système de lui expliquer certaines connaissances qu'il manipule, leur contexte de travail, leur utilisation, etc...
- sur la consultation en cours : l'utilisateur pourra demander pourquoi le système exécute telle action ou comment compte-t-il arriver à tel sous-objectif.

Ces différentes questions tournent donc autour des connaissances utilisées ou non lors de la consultation. Il faut veiller à ne pas se limiter à la traduction des connaissances mais également

à procéder à une certaine généralisation nécessaire pour établir le contexte dans lequel elles évoluent. Sans ce contexte, l'explication est désuette et pratiquement inutile. Il faut donc, dans toute explication, fournir la connaissance utilisée sur le moment mais aussi la stratégie de résolution pour laquelle elle a été utilisée.

- 5) - 5 - L'Acquéreur de connaissances.

Ce module est assez délicat. N'importe qui ne peut pas se permettre de venir ajouter de nouvelles connaissances. Il faut que ce processus soit strictement contrôlé. Alors, soit ce module est inclus dans le système expert, soit il peut être ajouté et retiré par une personne autorisée. Ce dernier cas n'est possible, sans nécessiter trop de travail, que dans certains langages interprétés. La première solution demandera quant à elle une gestion très stricte des utilisateurs du système expert. Un système de mots de passe sera indispensable pour l'accès à l'acquéreur de connaissances. La solution la plus sûre semble être celle qui consiste à supprimer physiquement le module d'acquisition et de ne le confier qu'à une ou plusieurs personnes autorisées. Comme l'acquisition de nouvelles connaissances ne se fait pas fréquemment, on peut se permettre une certaine perte de temps pour intégrer le module au système.

L'acquisition de nouvelles connaissances doit pouvoir se faire par l'intermédiaire d'une personne ou par le système lui-même. Dans le premier cas, le module devra effectuer des inférences pour vérifier :

- la correction de la connaissance en elle-même (sur des objets existants, etc...)
- la cohérence avec les connaissances actuelles (cf.paragr. B - 3)-6-)

La nouvelle connaissance ne s'ajoutera pas nécessairement à l'actuelle : le module devra être capable d'adapter ses connaissances en fonction du nouveau fait. Si toutefois la nouvelle connaissance doit être ajoutée, le système doit être

capable de la placer au bon endroit pour le fonctionnement du moteur d'inférence : une connaissance stratégique ne peut pas se retrouver avec des connaissances spécifiques. Toutes ces actions se feront suivant un dialogue avec l'utilisateur:

- demandes de confirmations
- demandes de corrections d'erreurs
- reformulation de la connaissance telle que le système expert la comprend
- etc.....

Dans le second cas, le module doit découvrir et apprendre de lui-même avec l'assistance d'un expert. Cette partie du module semble nettement plus compliquée. Mais surtout, il semble difficile de se fixer une limite sur la façon dont le système doit fonctionner : jusqu'où doit-il aller dans ses inférences pour que ça soit utile? Cette partie du module peut arriver à ressembler à un système expert où les connaissances seraient constituées d'heuristiques de raisonnement. MICHALSKI parle même de "Machine learning" (réf.(27)).
J'ai cependant deux critiques à formuler :

1) la machine ne peut pas apprendre seule sur base de ses propres observations; il faut lui fournir des critères de travail fort proches de la solution à acquérir.
2) de par la première remarque, j'ai constaté qu'on pouvait donc arriver à n'importe quelle conclusion du moment où on a fourni à la machine les critères qui permettent d'y arriver.

Il serait plus raisonnable, à mon avis, de ne réaliser que quelques inférences en fonction des observations réalisées lors des consultations du système. Un processus de généralisation des observations suffirait à vérifier si les connaissances actuelles les recouvrent.

Je pense donc que la fonction principale de l'acquéreur de connaissances est de garantir la cohérence et la correction des nouvelles informations introduites par un expert. L'auto-acquisition est un luxe inutile dans un premier temps.

- 5) - 6 - L'Interface en langage naturel.

L'interface doit permettre à l'utilisateur de penser qu'il entame un dialogue face à une personne intelligente, ceci afin de ne pas rendre le système vulnérable à l'interface alors que les inférences ont été réalisées correctement. L'interface doit donc permettre une communication rapide, efficace et suffisamment bonne techniquement pour le niveau du système. Elle doit fournir des phrases courtes et précises sur base d'un vocabulaire restreint et doit pouvoir fonctionner alternativement en mode déclaratif et en mode interrogatif.

Voici quelques fonctions qui me semblent requises pour ce module :

- une analyse syntaxique élémentaire permettant de construire une phrase interrogative ou déclarative avec un sujet, un verbe conjugué et un ou plusieurs compléments.
- une gestion d'un dictionnaire de mots-clés contenant des mots techniques, des mots classiques (pourquoi, comment) et un ensemble de verbes courants (être, faire, avoir,...) avec une conjugaison.
- un ensemble de schémas de phrases préconstruites ou de phrases types pour chaque objet et chaque propriété d'objet manipulé dans l'espace de travail.
- une gestion des erreurs et des fautes de frappe de l'utilisateur : ce dernier doit être mis en confiance devant un tel système. Cette gestion d'erreurs se fera avec le dictionnaire ci-dessus sur base d'une pondération et d'un degré de similitude entre deux mots.

On peut rêver à une interface fonctionnant par reconnaissance de la parole, mais ici, il est question d'efficacité donc de simplicité.

- 6) Comment construire un système expert.

Ce paragraphe-ci se limitera à présenter l'ordre dans lequel les choix énoncés doivent être effectués.

- tout d'abord, il s'agit de cerner correctement le contexte de travail : il faut énumérer complètement les objets à manipuler.

- ensuite, les experts établissent la base de connaissances et la stratégie de résolution dans un système de représentation des connaissances.
- enfin, on construit le moteur d'inférence qui utilisera ces deux premiers points.

Le choix d'un langage sera influent sur ces trois actions fort dépendantes.

Unefois qu'elles auront été réalisées, on peut dire que la charpente du système expert sera bâtie et pourra déjà être testée sérieusement. Pour tester un tel système, il faut prendre deux cas pratiques fort proches donnant des solutions très différentes. On les teste chacun séparément et s'ils donnent les résultats escomptés, avec le même raisonnement que l'expert, alors tout va bien.

Après cette première phase vient ce que j'appellerai le raffinement. On ajoute des modules qui ne sont pas immédiatement indispensables :

- on construit l'explicateur
- on met au point l'acquéreur de connaissances
- on adapte une interface en langage naturel.

Ce dernier module se comportera de façon similaire à la base de connaissances : dans le cas de l'homéopathie, il comprendra une base commune à toutes les maladies qui peuvent faire l'objet d'un système expert à laquelle on aura ajouté une partie propre à la maladie du moment. Cette séparation se retrouvera principalement au niveau du dictionnaire de mots-clés et du schéma de phrases préconstruites. Les deux autres parties (syntaxe et gestion d'erreurs) sont communes à toutes les maladies.

- 7) Conclusion.

J'ai résumé, durant ces quelques pages, un ensemble de points où certains choix délicats doivent être effectués lors de la construction d'un système expert. J'ai délibérément oublié le choix d'un langage pour l'implémentation : il ne sert à rien de vouloir implémenter un tel système sans une étude approfondie d'opportunité et de faisabilité. De plus, le but de ceci était

de donner une idée générale sur la difficulté de concevoir un système expert. Pour ce qui est de l'homéopathie, il semblerait qu'on puisse construire un système général commun à chaque maladie. A cette partie commune, on ajouterait les connaissances plus spécifiques à une maladie choisie dans un éventail possible. Ces connaissances spécifiques proviendraient :

- du schéma diagnostique de la maladie tel qu'il a été décrit auparavant,
- d'un ensemble de connaissances provenant de l'expert homéopathe.

II. EVOLUTION INDEPENDAMMENT DE LA BANQUE DE DONNEES CLINIQUES.

Le chapitre qui précède fait mention des évolutions possibles en fonction du développement de la banque de données cliniques. Cette évolution ne pourra être faite qu'à long terme. Il faut donc penser aussi à ce qui pourrait être amélioré dans l'immédiat. Pour cela, je me suis penché sur les outils existants ainsi que sur certaines réalisations qui ne donnèrent pas de conclusions significatives. Les améliorations que je suggère ici semblent a priori réalisables mais ne donneront pas nécessairement des résultats satisfaisants.

Je propose donc deux améliorations :

- la résolution du conflit entre deux ou plusieurs remèdes pour le navigateur.
- une tentative de classification des remèdes.

Cette dernière interviendra, comme je l'expliquerai plus tard, pour la résolution du conflit entre remèdes.

A) Evolution pour le navigateur.

Dans ce paragraphe, je vais expliquer le problème de conflit qui peut survenir entre des remèdes lors de l'utilisation du navigateur. Par après, j'exposerai une solution qui aidera le médecin à résoudre ce problème et à choisir un seul remède final.

- 1) Problème rencontré.

Une fois qu'un médecin a fourni au navigateur les symptômes qu'il a observé sur le patient, il peut demander au programme les meilleurs remèdes qui ont guéri ces symptômes. A l'heure où ces lignes sont écrites, deux politiques de sélection de remède sont disponibles :

- sur base de la fréquence du remède sur l'ensemble des symptômes observés,
- sur base de la somme des degrés du remède pour tous ces symptômes.

Le problème rencontré est indépendant de la stratégie choisie.

Le navigateur fournit au médecin une liste de remèdes avec les fréquences (ou degrés cumulés). Il peut arriver que la liste se présente comme suit :

	fréquence
Arsenicum album	26
Lychopodium	25
Aurum	16
etc..	..

Il est alors impossible pour le médecin de décider objectivement s'il doit choisir "arsenicum" ou "lychopodium". Il sait cependant qu'il ne devra pas choisir "aurum". Le navigateur devrait, à mon avis, pouvoir l'aider à trancher.

- 2) Solution possible.

Avant de chercher une solution, il me semble indispensable de trouver la cause du conflit pouvant exister entre deux ou trois remèdes. Pour cela, je vais introduire quelques conventions de représentation utilisées pour modéliser le processus.

- 2) - 1 - Notations.

-) r_1, r_2, ... représenteront les remèdes

-) $S(r_1)$ sera l'ensemble des symptômes du KENT qui sont recouverts par le remède r_1

-) $S(r_1)_{ch_i}$ sera l'ensemble des symptômes qui sont recouverts par le remède r_1 dans le chapitre "i".

-) S (obs) sera l'ensemble des symptômes du KENT qui ont été observés sur le patient

-) l'ensemble $S(r_1)$ sera représenté par le diagramme suivant

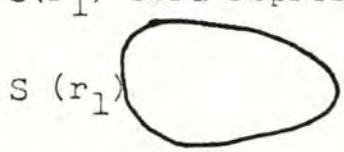


fig. 1.1

-) lorsque trois remèdes seront analysés ensemble, je combinerai $S(r_1)$; $S(r_2)$ et $S(r_3)$ comme suit :

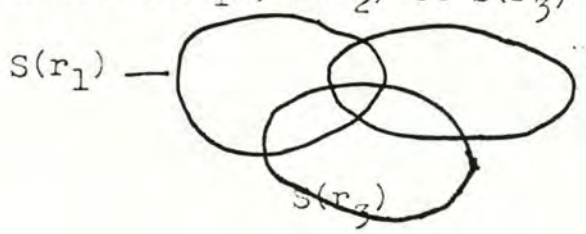


fig. 1.2

afin de permettre les possibilités d'intersection entre les ensembles de symptômes pour ces trois remèdes.

-) je me bornerai dans ce qui suit à régler le conflit entre trois remèdes au maximum. Plus serait inutile à mon sens si le médecin a fait un relevé correct des symptômes observés sur le patient.

-) les autres notations sont celles utilisées habituellement dans toute théorie ensembliste : intersection ($\cap$), union ($\cup$), différence ($\setminus$).

- 2) - 2 - Cause de conflit.

Pour moi, la cause du conflit entre deux ou trois remèdes possibles provient du choix des symptômes. Les symptômes entrés dans le navigateur ne sont pas suffisamment discriminants au niveau des remèdes concernés.

Je représenterai ce phénomène par le diagramme suivant:

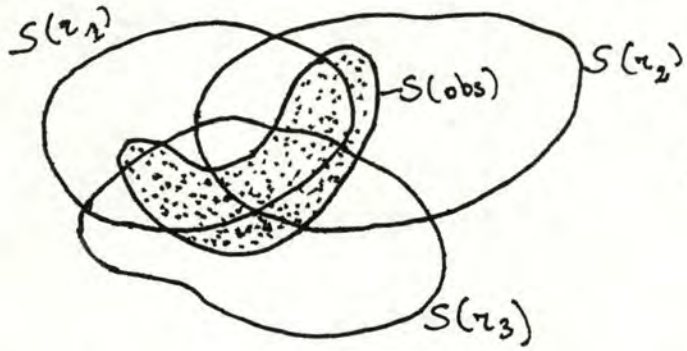


fig. 2.1

L'ensemble S (obs) se trouve réparti sur les intersections des remèdes. Les symptômes spécifiques à un remède sont en trop petite quantité. Le choix d'un remède ne peut se faire que s'il y a suffisamment de symptômes propres à un seul remède. Le diagramme idéal serait le suivant :

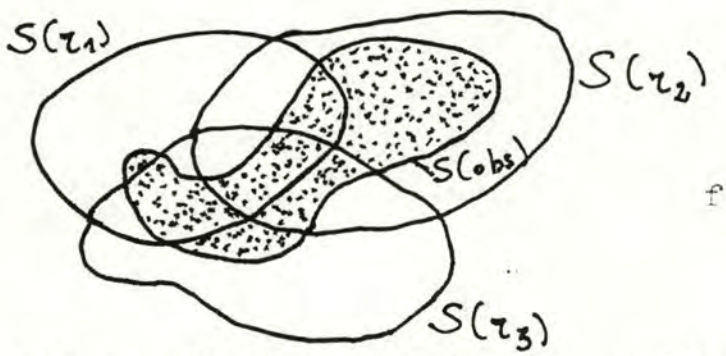


fig. 2.2

et il permettrait au médecin de choisir le remède r_2 comme prescription.

Pour régler le conflit, il faut donc trouver le moyen de passer d'une figure 2.1 à une figure 2.2. Il faudrait pouvoir orienter le médecin vers les symptômes discriminants du remède approprié.

- 2) - 3 - Exposé de la solution.

En fait, deux questions majeures viennent à l'esprit pour résoudre ce problème :

- quel remède éliminer ou conserver?
- de quelle façon orienter les débats vers ce remède?

L'idée générale de la solution est de poser des questions utiles à l'utilisateur sur les symptômes non observés, c'est-à-dire les questions qui ont le plus de chance de dissocier les remèdes en conflit. Elles seront d'abord générales pour devenir de plus en plus précises si besoin est.

a) Le Choix d'un chapitre discriminant.

Pour pouvoir poser la première question, il faut d'abord trouver un chapitre du KENT sur lequel sera orienté le débat. Ce chapitre devra être discriminant pour les remèdes en conflit : les symptômes guéris par tous ces remèdes sont peu nombreux par rapport à ceux guéris par un seul à la fois.

La façon de trouver un tel chapitre :

- soit on dispose pour chaque chapitre du KENT d'une classification des remèdes
- soit on utilise une méthode plus systèmatique mais plus lente.

a)-1- Par l'utilisation d'une classification.

L'exposé de la classification sera fait dans un paragraphe ultérieur. Il est cependant nécessaire d'expliquer en quoi cette classification sera utile ici :
Les remèdes ont été regroupés, pour chaque chapitre, suivant leurs affinités. Un chapitre sera appelé discriminant si chaque remède en conflit se trouve dans autant de groupes différents. Le chapitre à choisir sera un de ces chapitres discriminants.

a)- 2 - Par une méthode plus générale.-

Comme la classification précédente n'est pas disponible, il convient de prévoir une méthode de remplacement :

- récolter pour chaque chapitre tous les symptômes des remèdes concernés. On obtiendra donc le diagramme suivant pour le chapitre "i":

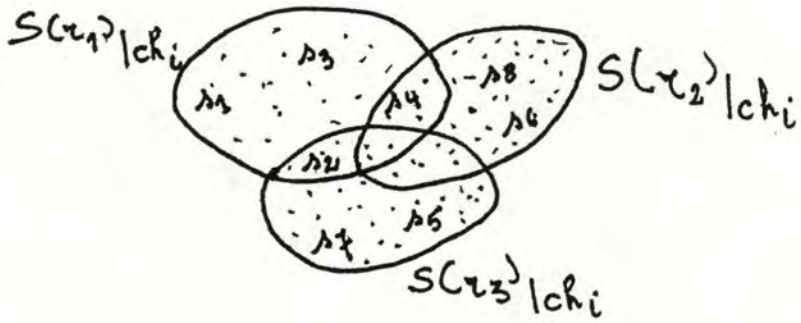


fig. 3.1

- observer le résultat (ceci peut se faire en parallèle avec le 1° point).

Si un chapitre n'a des symptômes que pour un seul des remèdes en question

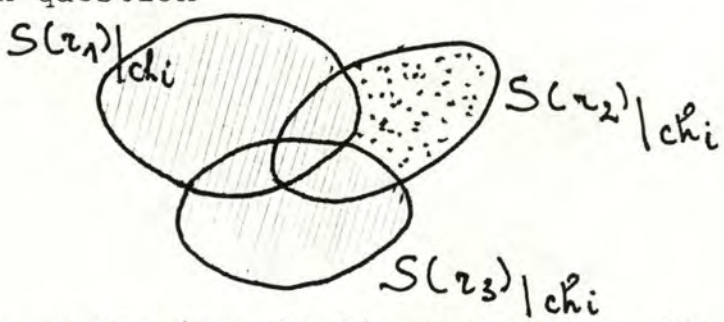


fig. 3.2.

OU si pour un chapitre le diagramme de la figure 3.1 devient le suivant :

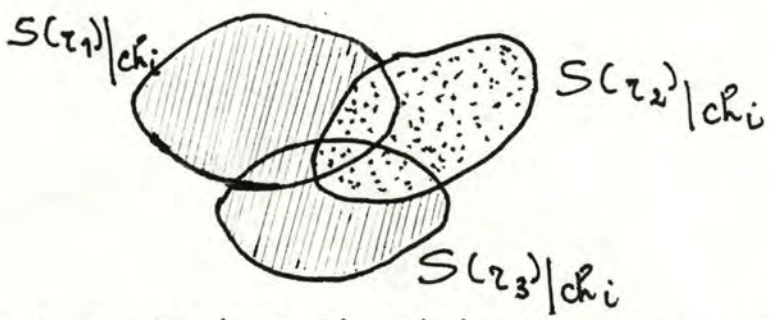


fig. 3.3.

Alors, on a un chapitre discriminant permettant d'éliminer ou de choisir r_2 comme remède final.

Si tel n'est pas le cas, il faut continuer.

- pour chaque chapitre, on effectue l'opération suivante:
  a) considérer séparément:
     - les symptômes propres à chaque remède (symptômes discriminants du remède)
     - les symptômes communs à deux ou trois remèdes (symptômes non discriminants)

  la figure 3.1 devient :

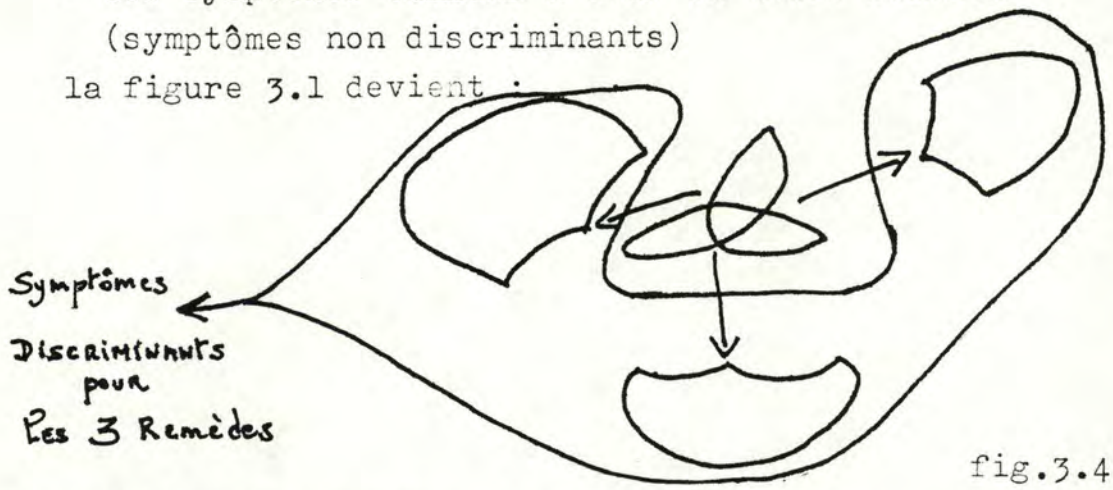


fig.3.4

  b) calculer le rapport

$$\frac{\#\text{ symptômes discriminants pour les remèdes}}{\#\text{ symptômes totaux}}$$

  avec $\#$ symptômes totaux $= \#\left(S(r_1)_{|ch_i} \cup S(r_2)_{|ch_i} \cup S(r_3)_{|ch_i}\right)$

  ce rapport est le degré suivant lequel le chapitre "i" est discriminant pour les remèdes en conflit.
  Je noterai ce degré : $d_i^{r_1,r_2,r_3}$
  et je l'appellerai degré de discrimination du chapitre "i" pour les remèdes r_1, r_2, r_3.

- On choisira comme chapitre le plus discriminant celui qui a le plus grand degré de dicrimination pour r_1, r_2, r_3.

$\Longrightarrow$ le numéro du chapitre choisi $= j$ tel que $d_j^{r_1,r_2,r_3} = \max_{1 \leq i \leq 31} \{d_i^{r_1,r_2,r_3}\} = d^*(r_1,r_2,r_3)$

$=$ degré maximum de discrimination pour r_1, r_2, r_3.

remarque : il me semble nécessaire de se fixer une borne inférieure sur le degré maximum de discrimination. En effet, si ce degré est de 0,3, cela voudrait dire qu'il n'y a pratiquement pas moyen de dissocier les remèdes en conflit. Tous sont donc possibles pour une prescription.

La méthode que je viens d'exposer est très lente au niveau du premier point: il faut récolter tous les symptômes pour chaque chapitre. Je pense qu'elle peut donc être améliorée en plusieurs endroits :

- les degrés de discrimination pourraient être construits à l'avance pour des combinaisons particulières de remèdes (ex. les plus fréquentes,...)
- le programme pourrait retenir les degrés de discrimination qu'il a déjà calculés auparavant (ceci lui éviterait de recommencer un long travail qu'il a déjà réalisé).

b) Poser la première question.

A ce niveau, je pense qu'il convient de poser une question générale ayant trait au titre du chapitre choisi ci-dessus.

exemple : "Avez-vous relevé des symptômes particuliers concernant les yeux?"
ou encore
"N'avez-vous rien à ajouter en ce qui concerne les symptômes des yeux?"

Cette première question doit être modulée en fonction des symptômes déjà observés. Dans l'exemple ci-dessus, il sera préférable de poser la deuxième question si le médecin a déjà relevé des symptômes sur les yeux.

La question générale est posée au médecin pour voir s'il n'a pas oublié un élément capital dans le relevé des symptômes.

Si le médecin répond par un ensemble de symptômes supplémentaires, il orientera automatiquement le système vers un remède.

c) Poser des questions sur la tendance des symptômes.

Il se peut que le médecin ait déjà observé des symptômes relatifs au chapitre concerné. Mais ce relevé peut être incomplet et ne pas contenir de symptômes pertinents. Avant de poser une question, il faudra donc résumer les symptômes discriminants pour chaque remède. On pourra alors poser une ou plusieurs questions suivant les tendances observées et les symptômes déjà retenus.

Cette partie de la solution est impossible à réaliser dans l'immédiat : le résumé des symptômes nécessite de manipuler la sémantique de ces symptômes. Actuellement, la structure du symptôme encodé ne le permet pas. Il faudra donc se limiter à des questions générales.

Il faut éviter de poser des questions directes au sujet d'un symptôme particulier : il ne faut pas suggérer le symptôme au médecin. Il faut que le médecin garde toute son objectivité par rapport au cas à traiter. C'est pour cela qu'il faut se limiter à proposer des orientations.

- 2) - 4 - Conclusion.

Une telle solution peut être appliquée à plusieurs reprises en ôtant le chapitre choisi après chaque ensemble de questions le concernant. On est cependant limité :

- soit par la borne inférieure du degré (maximum) de discrimination
- soit par le nombre de chapitres discriminants de la classification.

Le problème exposé au paragraphe - 1) sera surtout rencontré par les débutants. Ceux-ci n'auront pas relevé suffisamment de symptômes adéquats pour prendre une décision. Mais certains remèdes (les remèdes polychrestes) seront, à mon avis, délicats à départager même pour un médecin confirmé. La solution proposée pourra alors leur venir en aide malgré sa lenteur.

B) Une Classification des remèdes.

Une précédente tentative de classer les remèdes à échoué. Je vais donc proposer un autre type de classification qui, si elle réussit, pourra être exploitée pour aider :

- le navigateur dans l'évolution proposée précédemment
  et
- le médecin dans une meilleure maîtrise des remèdes (indépendamment du navigateur).

Le type de classification suggéré pourra être réalisé en utilisant deux méthodes différentes. Cependant, il est possible qu'aucune méthode ne donne des résultats satisfaisants.

- 1) Une précédente tentative.

Sur base d'une théorie de hiérarchisation élaborée par Mr. J.FICHEFET (réf.(29), Mrs. GARDIN et PARIS ont tenté de réaliser une classification générale. Celle-ci devait utiliser tout le KENT et permettre de regrouper les remèdes suivant leurs affinités. Le degré de similitude entre deux remèdes était fonction du nombre de symptômes qu'ils guérissaient l'un et l'autre. Ils n'ont cependant pu aboutir à aucune conclusion : les résultats fournis par les programmes étaient inexploitables. Aucune classification valable n'a pu être construite. Pensant avoir pris un trop grand ensemble de symptômes, ils se sont limités aux symptômes cardio-vasculaires. Là aussi, aucune conclusion n'a pu être réalisée. Ils se sont également limités à retenir les remèdes du KENT apparaissant au troisième degré sans plus de succès.

Il est donc possible que le type de classification que je vais proposer n'aboutisse pas non plus. Cependant, j'espère que, par son aspect plus restrictif, elle permette de dégager des affinités plus sélectives entre les remèdes.

- 2) Type de classification proposé.

Comme la tentative précédente n'a pas été couronnée de succès, j'ai regardé ce qui pouvait en être la cause. Il me

semble que le fait de classer les remèdes à travers tout le KENT était fort contraignant. Il m'est venu à l'idée qu'une classification localisée à un chapitre pouvait mieux réussir. D'un chapitre à l'autre, cette classification pourrait être fort différente. Ce qui expliquerait le fait qu'une classification globale soit impossible.

exemple :

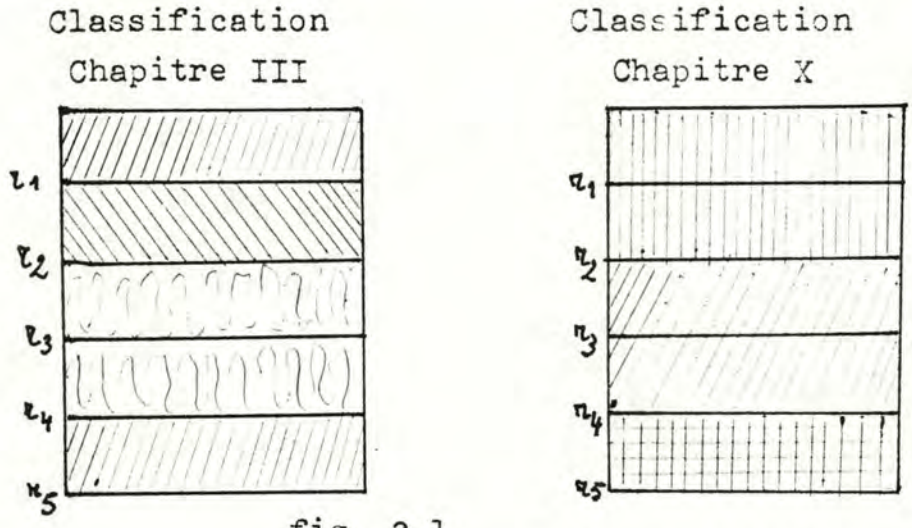


fig. 2.1

La classification pour le chapitre III est $\{\{r_1, r_5\}, \{r_2\}, \{r_3, r_4\}\}$
La classification pour le chapitre X est $\{\{r_1, r_2\}, \{r_3, r_4\}, \{r_5\}\}$
Il est pratiquement impossible d'établir une classification valable pour ces deux chapitres.

Je propose donc de réaliser une classification pour chaque chapitre du KENT. Il se peut qu'aucun chapitre n'admette de classification. Mais pourquoi n'y en aurait-il pas au moins un?

- 3) Utilité d'une telle classification.

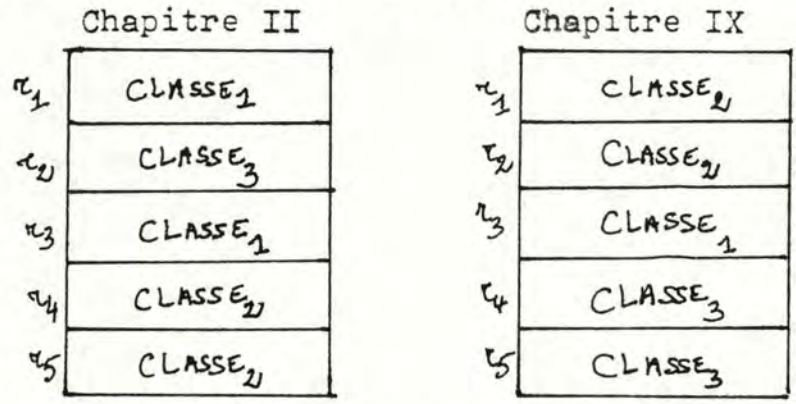
Si la classification est réussie pour chaque chapitre, cela veut dire qu'il y aura suffisamment de différences entre les remèdes de classes distinctes. On pourra donc régler un conflit entre deux ou trois remèdes sur base d'un chapitre où ils apparaissent dans des classes différentes. La constitution d'une classification permettra de gagner du temps pour apporter une solution au problème du conflit évoqué dans les paragraphes antérieurs: il ne faudra plus relever tous les symptômes du KENT pour les remèdes concernés.

De plus, les classifications de chaque chapitre pourront être utilisées telles quelles afin de permettre aux médecins de découvrir de nouveaux concepts tels que la tendance d'une classe de remèdes dans un chapitre du KENT. Cette tendance peut être expliquée comme étant l'idée maîtresse des symptômes spécifiques à la classe. Elle n'est cependant pas décidable par ordinateur comme je l'ai déjà évoqué précédemment. Sur base des classifications, on peut chercher une autre façon de cerner un remède pour un patient. On ne peut plus alors parler de l'évolution, mais plutôt de révolution. Cette dernière nécessiterait de reconstruire une nouvelle méthodologie homéopathique.

Voici donc les premières bases d'une nouvelle proposition:
Etant donné:

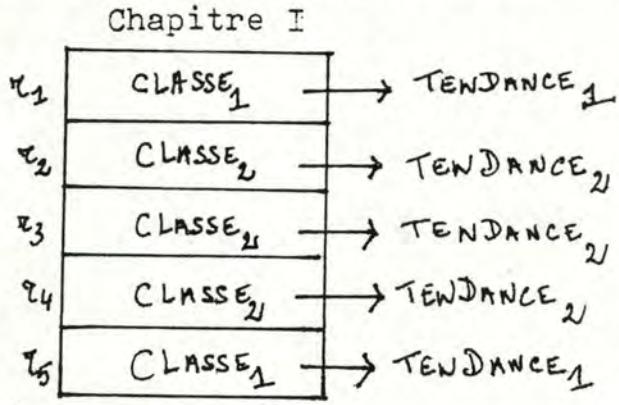
-) une classification des remèdes pour chaque chapitre du KENT

exemple:



-) une tendance explicite pour chaque classe d'un chapitre

exemple:



On peut dire que :

- un remède peut être décrit dans un chapitre par la tendance qui le concerne
- un remède est défini par l'ensemble des tendances qu'il a dans chaque chapitre.

On obtiendrait donc la façon suivante de trouver un remède:

-) relever pour chaque chapitre la tendance observée sur le patient
-) identifier le remède à prescrire avec le patient ainsi décrit.

On procèderait donc sur base de grandes lignes ou de traits généraux et non plus sur des symptômes spécifiques et limités.

Ce qui vient d'être dit n'est qu'une suggestion basée sur un résultat possible pour permettre un développement de l'homéopathie. Cela pourrait être utilisé comme argument pour justifier des recherches plus poussées à ce sujet.

- 4) Deux méthodes de classification.

Comme de nombreuses références sont disponibles sur les deux méthodes à venir, je me bornerai à expliquer le contexte d'utilisation de chacune. Des algorithmes existent pour toutes deux; il suffit de les adapter pour l'implémentation. Pour utiliser ces deux méthodes, il faut pouvoir d'abord déterminer une distance entre deux remèdes quelconques.

- 4) - 1 - Distance entre deux remèdes.

Références : (29)

La distance entre deux remèdes ne peut se calculer que sur base des symptômes observés dans le KENT. De plus, il faut toujours garder à l'esprit que la classification est spécifique à chaque chapitre. On n'utilisera donc que les symptômes du chapitre concerné.

Pour définir une distance, donc une différence, on peut d'abord considérer ce qui rapproche les deux objets à comparer. On utilisera donc un indice de similarité entre deux remèdes noté $I(r_i, r_j)$.

Voici cinq indices de similarité possible dans le cas qui nous occupe :

1) Russel et Rao : $I(r_i, r_j) = \frac{s_{ij}}{n}$

2) Kendall : $I(r_i, r_j) : 1 - \frac{u_{ij} + v_{ij}}{n}$

3) Jaccard : $I(r_i, r_j) = \frac{s_{ij}}{s_{ij} + u_{ij} + v_{ij}}$

4) Roger et Tanimoto:
$$I(r_i, r_j) = \frac{n - (u_{ij} + v_{ij})}{n + (u_{ij} + v_{ij})}$$

5) Ochiai : $I(r_i, r_i) = \frac{s_{ij}}{\sqrt{(s_{ij} + u_{ij})(s_{ij} + v_{ij})}}$

avec $\underline{n}$ = le nombre de symptômes du chapitre
$\underline{s_{ij}}$ = le nombre de symptômes communs aux remèdes r_i et r_j
$\underline{u_{ij}}$ = le nombre de symptômes propres à r_i
$\underline{v_{ij}}$ = le nombre de symptômes propres à r_j
$\underline{r_i}$ = le remède "i" de l'ensemble des remèdes ($\mathcal{R}$)

Ces indices de similarité ont été recueillis dans la référence (29). Pour plus de précision à ce sujet, il convient de consulter la littérature.

Je vais pouvoir construire une distance entre deux remèdes à partir d'un indice choisi parmi les cinq précités. Mais il me faut tout d'abord définir la notion de distance.

<u>définition</u> : on appelle distance dans l'ensemble $\mathcal{R}$ des remèdes une application $d : \mathcal{R} \times \mathcal{R} \to \mathbb{R}_+ : (r_i, r_j) \to d(r_i, r_j)$ qui vérifie les conditions suivantes :

(1) $\forall r_i \in \mathcal{R} : d(r_i, r_j) = 0$

(2) $\forall r_i, r_j \in \mathcal{R} : d(r_i, r_j) = d(r_j, r_i)$

(3) $\forall r_i, r_j, r_k \in \mathcal{R} : d(r_i, r_k) \leq d(r_i, r_j) + d(r_j, r_k)$

A partir d'un indice de similarité $I(r_i, r_j)$, on définira la distance comme suit ;

$$d(r_i, r_j) = 1 - I(r_i, r_j)$$

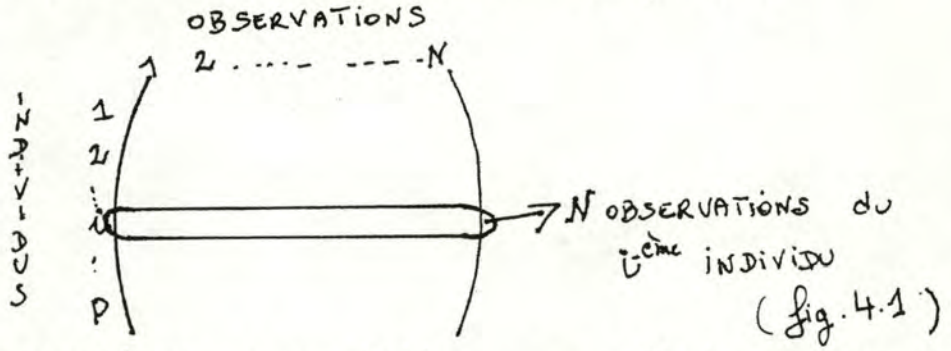
Il faudra contrôler alors si la distance ainsi définie vérifie les trois propriétés précédentes. Si tel est le cas, on peut calculer les distances entre remèdes.

- 4) - 2 - L'Analyse en composantes principales.

Références : (30) et (31)

a) types de problème résolu.

Etant donnée une matrice de départ constituée de n observations ou variables sur p individus (fig. 4.1)



(fig. 4.1)

exemple : observations → population, superficie, nombre de grandes villes

individus ↓

	population	superficie	nombre de grandes villes
Gde-Bretagne	50	50	100
France	50	100	100
Belgique	10	30	20

on peut :

-) interpréter d'énormes matrices de données en diminuant le nombre de variables pertinentes
-) analyser la proximité des points
-) réduire la taille de la matrice
-) transformer les variables en variables orthogonales.

b) La matrice de départ.

Dans le cas qui nous occupe, les individus seront les remèdes et la i-ème observation sera la distance séparant

un remède du i-ème remède.
On obtiendra donc la matrice suivante :

	Distance au remède 1	Distance au remède 2	....	Distance au remède n
Remède 1				
Remède 2				
.....				
Remède n				

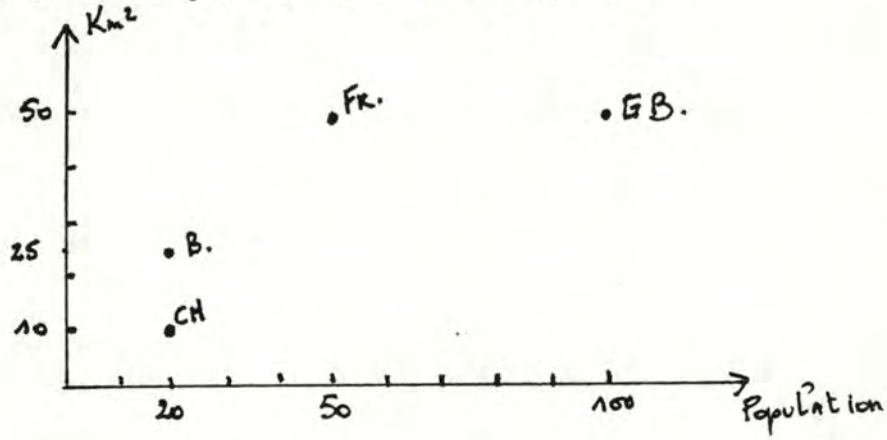
c) Principe de la méthode.

Telles qu'elles sont, les variables forment un référentiel de dimension "n". Comme "n" est largement supérieur à 2, elles sont inexploitables géométriquement. On va donc chercher deux variables dont les axes formeront un référentiel de dimension 2. Ces deux variables doivent permettre de décrire au mieux les observations initiales en perdant le moins d'information possible. Mais elles ne seront pas nécessairement des variables initiales de la matrice. Ces variables seront appelées les composantes principales.

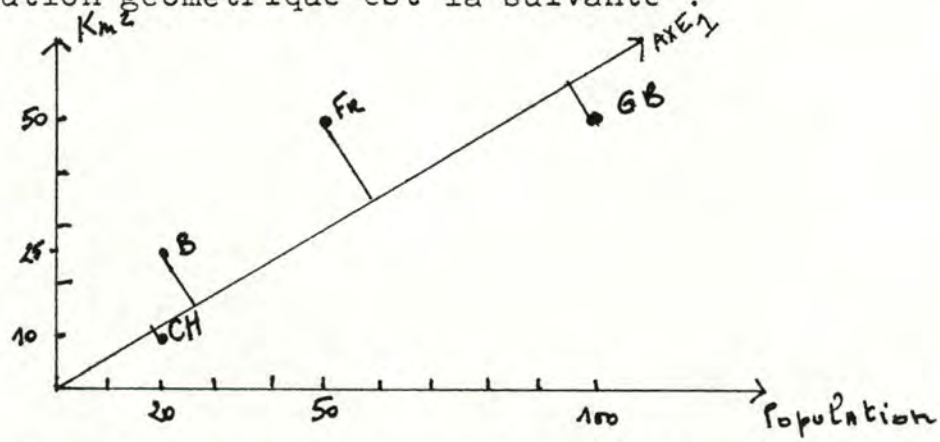
Prenons un exemple pour expliquer géométriquement la recherche de la première composante principale.
La matrice de départ sera la suivante :

	Population	Km^2
Gde-Bretagne	100	50
France	50	50
Suisse	20	10
Belgique	20	25

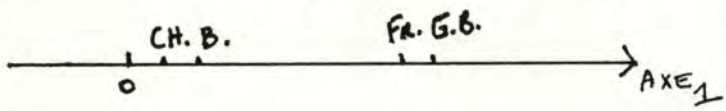
On pourra la représenter par le schéma suivant :



La matrice étant jugée trop compliquée à exploiter (pour l'exemple), on décide de se ramener à une seule composante principale. Elle devra garder suffisamment d'informations permettant d'observer la dispersion des points initiaux. Cette composante principale fournira un axe sur lequel seront projetés les points initiaux. Le problème est donc d'obtenir un axe tel que les projections soient les plus représentatives de la situation observée. Un tel axe peut être réalisé par calcul matriciel comme étant celui qui minimisera l'inertie des points (au sens physique du terme) et qui passera bien sûr par l'origine.
La solution géométrique est la suivante :



On aura donc le schéma des projections suivant :



Dans le cas général, une fois obtenue la première composante principale, on peut chercher la seconde. L'axe de celle-ci a les mêmes contraintes que celui de la première :

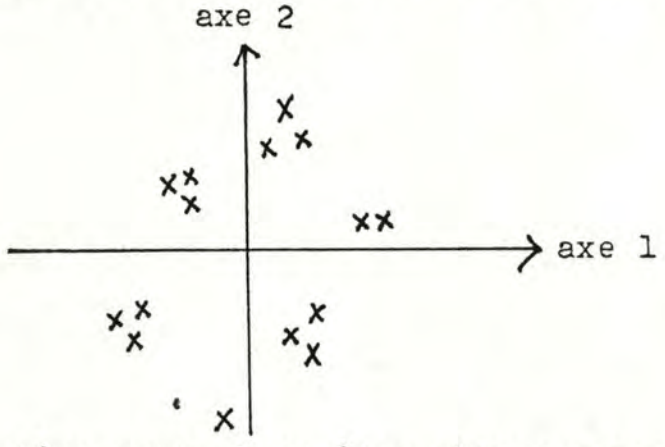
- passer par l'origine
- minimiser l'inertie des points initiaux.

Mais, en plus, il doit être perpendiculaire à l'espace engendré par la première composante.

Le procédé peut être poursuivi du moment que le nouvel axe cherché soit orthogonal au référentiel constitué par les précédentes composantes. On aura donc au maximum "n" composantes principales.

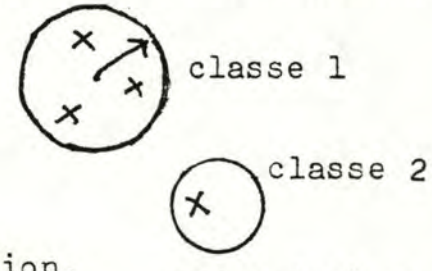
d) Résultats.

Suite à l'application d'une telle méthode (limitée en général à la recherche de deux composantes principales), on obtient une représentation géométrique de la forme suivante :



La classification peut alors être obtenue en observant le schéma ci-dessus : il suffit de se fixer un voisinage pour établir une classe.

exemple :



- 4) - 3 - La Hiérarchisation. Réf. : (32) et (33)

a) Contrainte sur la distance.

La distance choisie lors du paragraphe -4)-1- doit en plus être ultramétrique, c'est-à-dire posséder la propriété suivante :

$$\forall r_i, r_j, r_k \in \mathcal{R} : d(r_i, r_k) \leq \sup\left(d(r_i, r_j),\ d(r_j, r_k)\right)$$

b) La Matrice de départ.

Elle sera identique à celle du paragraphe -4)-2-.

c) Principe de la méthode.

1° au départ, chaque point constitue une classe

2° on cherche alors un seuil le plus petit possible à partir de zéro

3° on regroupe deux par deux les classes qui sont l'une de l'autre à une distance inférieure ou égale au seuil donné pour former de nouvelles classes.

4° on calcule toutes les distances entre ces nouvelles classes (par exemple, la distance entre deux classes peut être définie comme la distance entre les centres de gravité de ces classes)

5° on réitère le procédé à partir de 3° avec un seuil plus élevé.

La classification obtenue dépendra du seuil auquel on s'est arrêté.

La méthode sera illustrée par le schéma suivant :

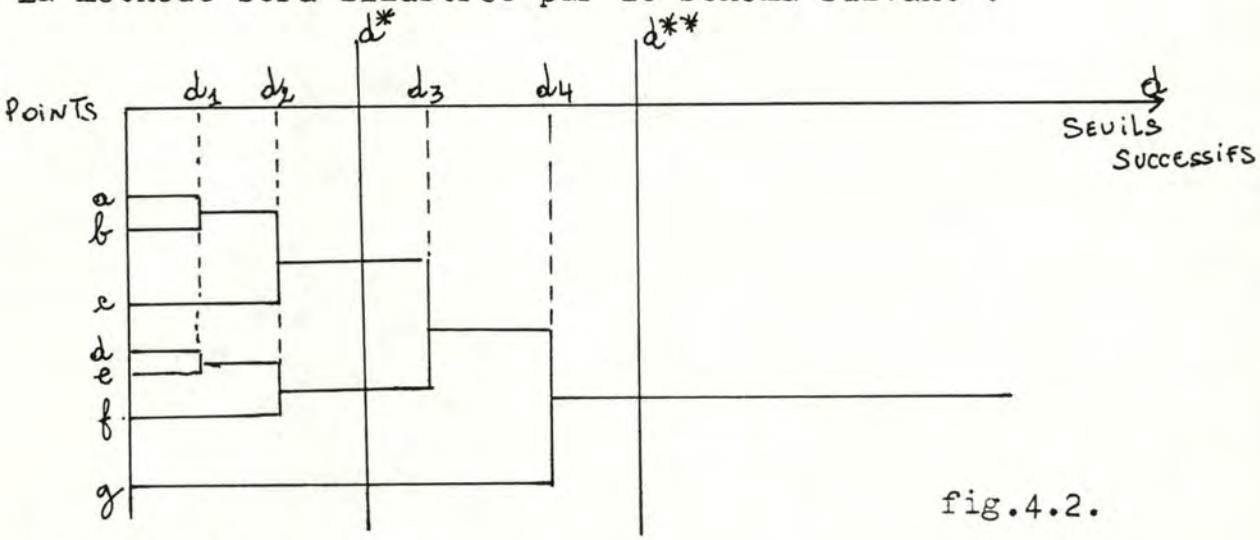


fig.4.2.

d) Résultats.

Pour un seuil $\underline{d}^*$ fixé (fig.4.2), on obtiendra la classification suivante :

$\{\{a,b,c\},\{d,e,f\},\{g\}\}$ - - - - - on a trois classes

Pour un seuil $\underline{d}^{**}$ fixé (fig.4.2), on aura une autre classification :

$\{\{a,b,c,d,e,f,g\}\}$ - - - - - on a une seule classe

- 4) - 4 - Conclusion.

Ces méthodes, ainsi que des algorithmes détaillés, peuvent être consultés dans de nombreuses publications. J'ai donc jugé inutile de détailler la théorie utilisée au profit d'une explication plus générale du fonctionnement.

- 5) Choix.

Avant de poser un choix, il convient de voir les avantages et les inconvénients de chaque méthode :

	Avantages	Inconvénients
Analyse en composantes principales	-) programme disponible aux Facultés	-) taille limitée de la matrice (n=50) -) interprétation pas facile pour la classification
Hiérarchisation	-) méthode déjà utilisée pour la précédente tentative paragr.-1) -) semble bien adaptée au problème de classification	-) restriction sur la distance utilisée

En conclusion, comme chaque méthode a des défauts, compensés par des qualités, pourquoi ne pas essayer les deux et comparer les résultats obtenus pour chacune? L'enjeu d'une classification me semble suffisamment important que pour justifier une telle proposition.

C O N C L U S I O N

L'impression d'ensemble que je garde au terme du travail est assez bonne. J'ai pu en effet apprendre une foule de choses tant au niveau de la programmation qu'au niveau des propositions de solution réalisables.

Sur le plan de la programmation, j'ai pu aborder des problèmes aussi divers que la gestion de fichiers, la gestion de listes d'impressions, la composition d'une syntaxe pour la consultation statistique de la banque de données centralisée, etc...

Du point de vue des propositions de solutions, j'ai dû beaucoup me documenter sur les systèmes experts ainsi que sur certaines méthodes statistiques d'analyse de données.

Comme je l'ai dit lors d'un chapitre, ce que j'ai programmé n'est certainement pas parfait. De plus, les médecins de l'U.I.H.N. n'ont pas encore donné leur avis à ce sujet. Ces programmes devront donc être considérés comme une base de discussion en vue d'améliorer l'environnement de travail.

Pour ce qui est des extensions possibles, elles sont nombreuses et suffisamment importantes que pour motiver une collaboration profitable entre médecins et informaticiens. Ces extensions proposées ne sont pas exhaustives et peuvent facilement laisser la place à d'autres objectifs prioritaires si nécessaire. Elles ont également un caractère plus évolutif que révolutionnaire : il ne s'agit pas de proposer une nouvelle méthodologie homéopathique.

Si la collaboration se poursuit entre l' U.I.H.N. et

ARCHIMEDE, beaucoup d'outils d'aide au diagnostic homéopathique peuvent encore être réalisés ou améliorés. Il suffit pour cela d'imagination, de compétence et de bonnes volontés.

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

(1) "Introduction à l'Homéopathie Hahnemannienne"
A. Jacques
U.I.H.N. Namur, août 1983.

(2) "Les principes de bases de l'Homéopathie"
A. Jacques
U.I.H.N. 4-11-1980

(3) "Organon of Medecine"
S.C.F. Hahnemann
B. Jain Publishers, New Delhi - 110016 - 1978

(4) "La Consultation Homéopathique - l'art d'interroger"
Pierre Schmidt, M.D.

(5) "Première Prescription - Deuxième Prescription - Pronostic - Lois de Hering"
A. Jacques
U.I.H.N. 2-12-1980

(6) "Technique Homéopathique- Répertoire et Pratique de l'Homéopathie"
Dr. Jacques Baur (Lyon)

(7) "Précis de Technique Répertoriale Homéopathique de Kent"
Dr. J. Hui Bon Hoa
Editions Coquemard, Angoulême, 1963.

(8) "The Encyclopedia of Pure Materia Medica" (12 vol.)
T.F.A. Allen
B. Jain Publishers, New Delhi- 110016- 1977.

(9) "Repertory of the Homeopathic Materia Medica with word index"
J.T. Kent
Ed. World Homeopathic Links, New Delhi - 10015.

(10) "Une banque de schémas diagnostiques en homéopathie?"
J. Fichefet
Namur, 2 mai 1983.

(11) "Cours d'introduction à la statistique"
J.P. Rasson
F.U.N.D.P.

(12) "Cours d'analyse fonctionnelle"
F. Bodart
F.U.N.D.P.

(13) "Knowledge representation"
P. Jackson, Department of Artificial Intelligence, Edinburgh University.
Cours de la Chair I.B.M. 1985.

(14) "A Frame for Frames : representing Knowledge for Recognition"
Benjamin J. Kuipers.

(15) "Frame representations and the Declarative/procedural controversy"
Terry Winograd

(16) "Logic programming for expert systems"
P. Hammond, F. Kriwaczek, M. Sergot
Cours de la Chair I.B.M. 1985.

(17) "Logic for Problem solving"
Robert Kowalski
Artificial intelligence series (vol.7)
North Holland

(18) "Représentation des connaissances dans les systèmes experts"
Suzanne Pinson
AFCET Bordas Dunod
R.A.I.R.O. Informatique/Computer Science
vol.15 n°4, 1981, p. 343 à 367.

(19) "Représentation et utilisation des connaissances"
première partie: "Les systèmes experts"
J.L. Laurière
T.S.I.

(20) "Représentation et utilisation des connaissances"
deuxième partie: "Représentation des connaissances"
J.L. Laurière
T.S.I.

(21) "Reasoning about belief in the context of advice-giving systems"
P. Jackson

(22) "NEOMYCIN : reconfiguring a rule—based expert system for application to teaching"
William J. Clancey and Reed Letsinger
Proceedings of the Seventh International Conference on Artificial Intelligence.
August 1981.

(23) "The advantages of abstract control knowledge in expert system design"
William J.Clancey
November 1983.

(24) "Strategic Explanations for a diagnostic Consultation System"
Diane Warner Hasling, William J.Clancey and Glenn Rennels
November 1983.

(25) "Classification Problem Solving"
W.J. Clancey
July 1984.

(26) "Acquiring, Representing and Evaluating a Competence Model of Diagnostic Strategy"
William J. Clancey
February 1984.

(27) "Machine Learning : An Artificial Intelligence Approach"
Ryszard S. Michalski, Jaine G. Carbonell and Tom M. Mitchell
Springer -Verlag, 1984.

(28) "Computer-based medical consultations, MYCIN"
Edward H. Shortliffe
American Elsevier Publishing Co.,Inc., 1976.

(29) Notes manuscrites de Mr. J.Fichefet :
"Chapitre II: Classification hiérarchique des symptômes"

(30) "Analyse de données multi-fonctionnelles"
P.Bertier et J.M. Bouroche
Presse Universitaire de France, 1975.

(31) "Cours de Statistique appliquée à la gestion"
M. Noirhomme
F.U.N.D.P.

(32) "Statistical methods for digital computers " Volume III
Enslein K., Ralston A., Wilf H.S.
Wiley Interscience publication.

(33) "Images, Principes, Traitements, Applications"
Gabriel E. Lowitz, Georges Stamon
Douzième Ecole internationale d'informatique
F.U.N.D.P., Juillet 1982. AFCET

P R E S E N T A T I O N D E S A N N E X E S

- Manuel d'initialisation générale
- Manuel d'utilisation général
- Manuel d'utilisation pour l'opérateur
- Manuel d'utilisation pour l'U.I.H.N.
- Manuel d'utilisation pour le Médecin

DANS LA FARDE LISTING :

- Les librairies de requêtes
- Les requêtes utilisées dans les programmes
- Les écrans utilisés par les requêtes
- Les records utilisés par les fichiers
- Les domaines utilisés par DATATRIEVE
- Les ports définis pour l'utilisation programmée de DATATRIEVE
- La source COBOL des programmes et les fichiers de résultats

MANUEL D'INITIALISATION GENERALE

Avant de pouvoir utiliser le système, il faut procéder à certaines initialisations, notamment sur les fichiers de "DATATRIEVE".

Voici la séquence d'opérations à réaliser dans la directory qui contiendra ces fichiers

```
$ DTR
DTR > DEFINE FILE DERNIERS
 "       "    "   SECRETAIRE KEY = NO_INT_MED_S(NO CHANGE,DUP)
 "       "    "   MEDECIN KEY = NO_INT_MED (NO CHANGE, NO DUP)
 "       "    "   DOSSIER_INDIVIDU KEY = NO_DOSSIER-I (NO CHANGE,
                                                    NO DUP)
 "       "    "   DOSSIER_SUITE KEY = NO_DOSSIER_S (NO CHANGE,
                                                    NO DUP)
 "       "    "   SUITE_DEUX KEY = NO_DOSSIER_SD (NO CHANGE,
                                                    NO DUP)
 "       "    "   SUITE_TROIS KEY = NO_DOSSIER_ST (NO CHANGE,DUP)
 "       "    "   IDX_ANTEC_IND KEY = NO_DOSSIER_IAI (NO CHANGE,DUP)
 "       "    "   DECES KEY = NO_DOSSIER_D (NO CHANGE,DUP)
 "       "    "   RENS_MALADIE KEY = NO_DOSSIER_MALADIE
                                        (NO CHANGE, NO DUP)
 "       "    "   REM_ANT_D KEY = IDENT_REM_ANT (NO CHANGE, DUP)
 "       "    "   MEDIC_ANT KEY = IDENT_MEDIC_ANT (NO CHANGE,DUP)
 "       "    "   IDX_ANTEC_MALADIE KEY = NO_DOSSIER_MALADIE_I
                                          (NO CHANGE,DUP)
 "       "    "   CONSULTATION KEY = NO_DOSSIER_MALADIE-C
                                          (NO CHANGE, DUP)
 "       "    "   SYMPTOMES_ACTUELS KEY = IDENT_SYMPTOMES_ACTUELS
                                          (NO CHANGE, DUP)
 "       "    "   REM_ACTUELS KEY = IDENT_REM_ACTUELS
                                          (NO CHANGE, DUP)
 "       "    "   MEDIC_ACTUELS KEY = IDENT_MEDIC_ACTUELS
                                          (NO CHANGE, DUP)
 "       "    "   DOSSIER_ANALYSES KEY = IDENT_DOSSIER_ANAL
                                          (NO CHANGE, DUP)
 "       "    "   IDX_RELATION KEY = NO_INT_MED_REL
                                          (NO CHANGE, DUP)
 "       "    "   DECES_C
 "       "    "   RENS_MALADIE_C
 "       "    "   MEDIC_ANT_C
 "       "    "   REM_ANT_D_C
```

```
DTR > DEFINE FILE IDX_ANTEC_MALADIE_C
 "        "      "  CONSULTATION_C
 "        "      "  SYMPTOMES_ACTUELS_C
 "        "      "  REM_ACTUELS_C
 "        "      "  MEDIC_ACTUELS_C
 "        "      "  DOSSIER_ANALYSES_C
DTR > READY DERNIERS WRITE
DTR > STORE DERNIERS
DERN_NO_CENTR_ATTR : Ø
DERN_NO_MEDIC_ATTR : Ø
DTR > EXIT
$
```

En ce qui concerne les fichiers qui contiennent les écrans, il faut les transférer dans la directory définitive et veiller à ce que les programmes aillent lire dans cette directory.

Les fichiers d'écrans et les fichiers de la banque de données doivent recevoir une protection particulière. Pour cela, il faut exécuter l'opération suivante :

```
$ SET PROTECT = (W:R) *.DAT
$ SET PROTECT = (W:R) *.RLB
```

MANUEL D'UTILISATION GENERAL

Tous les programmes qui ont été conçus dans le cadre de ce mémoire dialoguent avec l'utilisateur suivant le même principe : une succession d'écrans (avec, en option, une succession de messages sur la dernière ligne de l'écran).

1) Action sur l'écran :

Lorsque l'écran est rempli correctement (suivant l'utilisateur), il suffit d'appuyer sur la touche de retour (< ret> ou ↵) pour passer à l'écran suivant.

remarques : - on ne peut pas revenir à l'écran précédent.
- deux écrans successifs peuvent être identiques

Entre deux écrans peut apparaître sur la dernière ligne un message demandant une information supplémentaire. Il faut alors introduire la réponse suggérée (souvent O ou N) suivie du retour.

2) Action sur une zone à remplir de l'écran.

Le programme a sélectionné un ensemble de zones de l'écran qu'il souhaite voir remplir. Le curseur est positionné automatiquement sur la première zone.

Voici les touches qui peuvent être utilisées au niveau des zones :

- TAB : pour passer à la zone suivante
- BS : pour revenir à la zone précédente
- →: pour avancer d'un caractère dans la zone
- ←: pour reculer d'un caractère dans la zone
- PF2 : pour demander de l'aide sur la zone à remplir
- LF : pour vider la zone de son contenu
- ⇐: pour effacer le caractère précédent le curseur

remarques : on ne sait pas insérer un caractère au milieu d'une zone, on ne peut que remplacer un caractère par un autre.

ex. : DPONT on ne peut insérer U entre D et P
il faut réaliser la séquence suivante:

DUONT
DUPNT
DUPOT
DUPON
DUPONT

-) Des messages accompagnés d'un signal peuvent apparaître si le champ n'est pas rempli de la façon souhaitée.
Voici une liste non exhaustive des messages en anglais avec leur interprétation :
Bad choice : la donnée introduite ne correspond pas à une des valeurs attendues
Input required : la zone est à remplir obligatoirement
Must fill : la zone est à remplir complètement
Value out of range : la donnée dépasse des bornes qui ont été fixées.

-) Un signal seul est entendu si :
° soit on essaie d'accéder à une zone suivante alors que c'est la dernière,
° soit on essaie de remplir une zone de plus de caractères qu'elle ne peut en contenir.

3) Messages particuliers.

-) Des messages tels que "$$$ Planté..." ou "*** Planté..." peuvent apparaître.
Il faut stopper le programme le plus rapidement possible et expliquer les circonstances le plus précisément possible à la personne responsable. Ces messages sont dus à une erreur dans le programme.

-) Il peut arriver également que l'écran soit perturbé par des mots (en anglais le plus souvent) qui ne constituent pas les messages habituels de la dernière ligne. Il faut alors cesser toute activité au terminal et suivre la même procédure que précédemment.
ex. : - DTR> peut apparaître au milieu de l'écran
- Illegal ASCII numeric...

4) Remarques générales.

-) Les caractères nationaux seront refusés systématiquement.
ex. ü, é, è, ê, à, ç, ...

-) Certains caractères seront acceptés mais créeront des problèmes par après. Il faut donc les éviter impérativement. Ce sont :
(;, /, ¦, ?, ¦, _).
Il est à noter que (,,.,-) fonctionnent sans problèmes.

5) Comment faire imprimer un fichier.

a) Si le texte doit être imprimé proprement (format A4), il convient d'introduire l'instruction suivante :

PRINT LA100 : nom du fichier

b) Si le texte doit être imprimé rapidement sans fioriture (grand format, papier vert), il convient d'introduire l'instruction suivante :

GPRINT nom du fichier — si le texte ne comporte pas d'accent ou autre caractère national.

PRINT nom du fichier — dans le cas contraire.

Pour voir si la demande d'impression a été prise en considération, il suffit de faire :

LPQ

remarque : il convient d'attendre un certain temps avant de réaliser cette opération, si l'opération "PRINT nom du fichier" a été réalisée. La saisie de la demande d'impression est conditionnée par un traitement antérieur des caractères nationaux.

MANUEL D'UTILISATION POUR L'OPERATEUR

Les possibilité d'action de l'opérateur se regroupent en deux catégories :

-) la gestion des médecins dans le système
-) l'exécution de travaux pour les médecins.

I. LA GESTION DES MEDECINS DANS LE SYSTEME.

A) Le programme "progr" ("run prog" pour le lancer).

Le programme "progr" permet de créer ou de mettre à jour des informations concernant un médecin et/ou les secrétaires d'un médecin. Il permet également de "détruire" un médecin ou un(e) secrétaire qui n'aurait plus de raison d'être dans le système. Cette fonction est cependant fort délicate en ce qui concerne un médecin. En effet, "détruire" un médecin signifie, en plus, "détruire" tous les dossiers de ce médecin.

remarque: cette destruction n'est pas physique, elle n'est que logique. Les dossiers du médecin sont toujours présents mais inaccessibles : ils sont perdus dans le système.

Le programme "progr" donne également un rappel succint de
1) la façon d'utiliser les écrans
2) la façon de lancer les rappels courriers
3) la façon de lancer la vérification des transactions
4) la façon de lancer le transfert des questionnaires patients et individus
5) la façon de lancer l'enregistrement des analyses.

Pour choisir l'action à exécuter, il suffit d'introduire le numéro correspondant dans le menu de l'opérateur.
Le programme présentera le menu au début et après chaque fin d'action tant qu'on ne veut pas arrêter le programme. Si on veut arrêter, alors il faudra introduire "00" dans la zone correspondant au choix. En général la touche PF2 donnera suffisamment d'informations pour remplir la zone de façon adéquate.

Le programme "Progr" fournit, si au moins un médecin a été ajouté ou modifié dans le système, un fichier résultat contenant la liste de tous les médecins pour l'U.I.H.N.
Le nom du fichier résultat est "fichier TXT". Il est accessible via l'éditeur et il peut être imprimé à l'écran (Type fichier TXT) ou sur imprimante (voir manuel d'utilisation général).

En ce qui concerne les messages qui apparaissent au bas de l'écran ils sont de trois types :

1) message qui demande une réponse
   exemple : a) médecin inexistant. <ret> pour continuer
             b) confirmez la demande (Oui-Non)?
   Ces messages bloquent le programme tant que l'utilisateur n'y a pas répondu.

2) message d'attente :
   exemple : veuillez patienter quelques instants
   Ces messages signalent que l'ordinateur travaille seul pour certaines actions. Il faut donc attendre qu'il envoit un autre message pour pouvoir continuer.

3) les messages du type :
   "$$$ Planté....."
   ou
   "*** Planté....."
   Pour ceux-là, voyez le manuel d'utilisation général.

B) La vérification des transactions des médecins du système.

Ce programme peut fonctionner seul. L'opérateur va donc le lancer et, dès qu'il sera lancé, le système permettra à l'opérateur d'exécuter un autre travail pendant ce temps (le prompt habituel "$" apparaîtra normalement).

Pour lancer ce programme, il suffit d'introduire :

$ @ VERIFICATION

Le système enverra un message signifiant qu'il a pris la demande en considération et qu'il va exécuter seul la vérification des transactions.
L'opérateur peut voir à tout moment si la machine a terminé le programme en introduisant:

$ SHB

Si l'ordinateur renvoit un message contenant : "....VERIF....", c'est qu'il n'a pas fini le traitement. Sinon, la vérification est terminée.
A ce moment-là, de deux choses l'une, soit le programme est correct et le résultat en est fourni dans le fichier "VERIFT.TXT", soit le programme s'est mal terminé et il convient de regarder les fichiers "VERIFT.TXT" et "ERREURV.TXT". En cas de mauvais fonctionnement, il faut avertir au plus vite la personne responsable qui prendra une décision en fonction des erreurs signalées dans le fichier "ERREURV.TXT".

II. L'EXECUTION DE TRAVAUX POUR LE MEDECIN.

Ces différents travaux sont :

-) l'enregistrement des résultats d'analyses provenant du laboratoire,
-) l'édition de rappels-courriers,
-) le transfert des disquettes contenant les questionnaires patients et individus.

A) L'enregistrement des analyses.

Pour effectuer cette fonction, il faut que la bande magnétique contenant les analyses soit montée sur le VAX. Ensuite, le contenu doit être transféré tel quel dans le fichier "ANALYSES.DAT" de la directory de travail.

On peut alors exécuter le programme ("RUN ANAL") pour transférer les analyses à leur place dans la banque de données.

B) L'édition de rappels-courriers.

Pour obtenir le fichier "RAPPELS.TXT" contenant les lettres de rappels pour chaque médecin, il suffit d'introduire l'instruction

$ @ COURRIER

Pour voir si le programme est terminé, il suffit de faire:

$ SHB

Si la machine répond "....RAPPELS....", alors ce n'est pas encore fini.

Pour imprimer le fichier de résultats, il faut s'en référer au manuel d'utilisation général.

C) Le transfert des questionnaires.

Il faut procéder fichier par fichier.

-) Le fichier à traiter sera transféré de la disquette vers le VAX grâce à POLY/XFR et à un programme mis au point par Mr. P.GARDIN.
-) Si les questionnaires portent sur des patients, il faudra :
   - ° transférer le fichier du VAX dans le fichier "QUESTPAT.DAT" de la directory de travail,
   - ° exécuter l'opération "RUN TRANSFERT" pour transférer les questionnaires dans la banque de données cliniques.
-) Si les questionnaires portent sur des individus, il faudra :
   - ° transférer le fichier du VAX dans le fichier "QUESTIND.DAT" de la directory de travail,
   - ° exécuter l'opération "RUN TRIND" pour transférer les questionnaires dans la banque de données cliniques.

MANUEL D'UTILISATION U.I.H.N.

La seule fonction automatisée disponible pour l'U.I.H.N. est la centralisation des maladies.

Cette fonction se déroule en deux parties:

-) la saisie des numéros de maladie à centraliser
-) la centralisation de ces numéros.

I. LA SAISIE.

Pour effectuer la saisie des numéros, il faut lancer l'instruction suivante :

$ RUN SAISIE

Ce programme permet d'enregistrer au maximum cent numéros de maladies à centraliser. Ce numéro est constitué :

-) du numéro interne du médecin
-) du numéro de dossier du patient
-) du numéro de la maladie à centraliser.

II. LA CENTRALISATION.

Pour effectuer la centralisation, il faut lancer l'instruction suivantes :

$ @ CENTRALISER

Lorsque le programme est terminé, il a centralisé toutes les maladies désignées, mais il a aussi produit un ensemble de fichiers de résultats :

-) NEXIST.TXT : contient les numéros des maladies inexistantes,
-) NVAL.TXT : contient les numéros des maladies dont les codes n'ont pas été validés,
-) DEXIST.TXT : contient les numéros des maladies centralisées antérieurement,
-) DOSNEX.TXT : contient les numéros des dossiers des patients inexistants,
-) MEDNEX.TXT : contient les numéros des médecins inexistants,
-) DOSRAI.TXT : contient les numéros des dossiers qui se trouvent sur le "RAINBOW".

Pour voir si le programme est terminé, il faut introduire l'instruction suivante :

```
$ SHB
```

Si l'ordinateur envoit un message contenant le mot "CENTR", ce n'est pas fini.

MANUEL D'UTILISATION POUR LE MEDECIN

Chaque médecin du système disposera de cinq possibilités d'action :

1) l'encodage de questionnaires
2) le suivi du dossier d'un patient
3) l'édition de dossiers
4) la consultation de dossiers centralisés
5) l'affichage de statistiques sur les dossiers centralisés

I. L'ENCODAGE DE QUESTIONNAIRES : ("RUN QUEST")

Le médecin introduit :

-) son numéro interne
-) le numéro de dossier (qu'il veut créer ou modifier)
-) soit C pour indiquer qu'il veut créer un nouveau dossier
   ou
   soit M pour indiquer qu'il veut modifier un dossier existant
-) soit I pour indiquer qu'il traite un questionnaire individu
   ou
   soit P pour indiquer qu'il traite un questionnaire patient.

Si aucune erreur de compatibilité n'a été commise (ex.vouloir créer un dossier existant), le médecin peut introduire ses modifications en fonction du manuel général d'utilisation.

Il peut : - modifier les renseignements généraux
- ajouter des antécédents
- ajouter des professions

au patient ou à l'individu.

II. LE SUIVI DU DOSSIER D'UN PATIENT ("RUN SUIVIP")

Le médecin introduit: - son numéro interne
- le numéro du dossier à traiter.

Si le dossier existe, il peut alors choisir un ensemble de traitements sur celui-ci :

-) traiter une maladie existante
-) créer une nouvelle maladie
-) modifier les renseignements généraux du patient
-) traiter une prescription en dehors d'une maladie
-) traiter les analyses
-) traiter un décès
-) valider les codes (OMS et autres) d'un dossier.

remarques:-la validation des codes n'existe pas encore
-le traitement des prescriptions se limite à en créer. On ne peut pas encore les consulter.
-pour pouvoir traiter un dossier, il doit exister et être accessible sur le VAX : le médecin ne peut pas le garder sur disquette.

A) Traiter une maladie existante.

Le médecin peut demander à obtenir un résumé des maladies et consultations du patient concerné. Il peut ensuite soit :
-) modifier les éléments d'une maladie existante,
-) ajouter une consultation à une maladie en cours,
-) consulter une maladie existante.

Ces trois actions pourront se faire en répondant à un ensemble de questions apparaissant sur la dernière ligne de l'écran. En général, il suffit d'y répondre par "O" ou par "N".

remarques: - la modification d'une maladie et l'ajout d'une consultation peuvent être interrompus. Pendant l'interruption, le programme propose l'exécution de certaines fonctions dites "ininterruptibles". Quand ces dernières sont terminées, le programme revient à l'endroit où on l'avait interrompu.

B) Créer une nouvelle maladie.

Le médecin crée une nouvelle maladie et la première consultation de cette maladie (s'il le veut). Ce traitement est également interruptible.

C) Traiter les renseignements généraux.

Cette fonction se limite à permettre de modifier les renseignements généraux sur un patient, ses professions successives et ses antécédents (rougeole, ...).

D) Traiter les analyses.

Cette fonction possède trois parties différentes :

1) le médecin peut créer un ensemble d'analyses pour une consultation. Comme ces analyses portent sur une même consultation, elles recevront un numéro de référence identique. Ce numéro de référence devra être fourni au laboratoire pour les demandes d'analyses de la consultation.

   exemple: un patient nécessite cinq analyses
   un autre sept

   ⇒ on aura deux références:
   - une concernant les cinq premières analyses
   - une concernant les sept dernières.

   ⇒ il faut effectuer deux demandes différentes (une pour chaque groupe d'analyses)

2) le médecin peut détruire les analyses correspondant à une référence s'il a annulé la demande.

3) le médecin peut retrouver un numéro de référence qu'il a oublié : il introduit la date de la consultation et le programme fournit la référence concernée.

Remarque : la modification ou la consultation d'analyses complètes se fera au niveau du traitement d'une maladie existante.

E) Traiter un décès.

Si un patient vient à décéder, le médecin peut introduire les causes de décès. Le médecin peut également modifier ces causes de décès. Il peut aussi consulter en entier le dossier d'un patient décédé (c'est d'ailleurs la seule façon d'accéder à un patient décédé).

III. L'EDITION DE DOSSIERS.

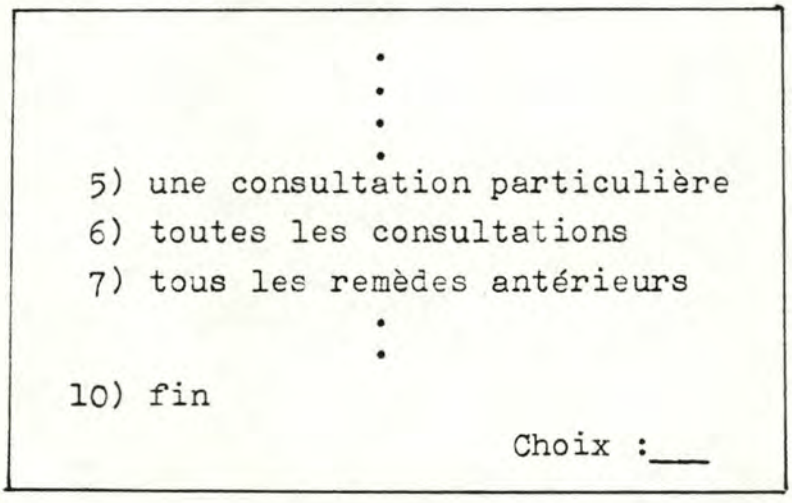
Pour éditer des dossiers, le médecin doit procéder en deux étapes successives :

1) l'acquisition des parties à éditer
2) le traitement effectif de l'édition

A) L'acquisition.

A) L'acquisition.

Pour commencer l'acquisition, il faut exécuter "RUN ACQUER" Le médecin introduit alors son numéro interne. Puis, il sélectionnera dans un ensemble de menus les parties à éditer.
Pour expliquer la façon de procéder, je prendrai l'exemple suivant: Une partie de l'écran qui se présente est la suivante :



Supposons que le médecin ait déjà choisi une maladie. Il veut maintenant toutes les consultations de cette maladie et tous les remèdes antérieurs. Il va donc introduire le numéro 6 à l'endroit du choix. Il pourra alors traiter la consultation (par une suite de menus du même genre). Lorsqu'il aura fini la consultation, il introduira 7 pour ajouter les remèdes. Enfin, il introduira 10 pour terminer la maladie désirée.

B) L'édition.

Il suffit d'introduire "@ EDIT" pour lancer le programme d'édition. Lorsqu'il sera terminé, le fichier résultat à imprimer sera "EDITION.TXT". Pour voir si le programme est terminé, il faut introduire $ SHB. Si la réponse contient "EDITER", alors le programme n'est pas terminé.

IV. LA CONSULTATION DE DOSSIERS CENTRALISES ("RUN CONSUL").

Le médecin introduit le numéro de centralisation attribué à la maladie. Il peut alors consulter :

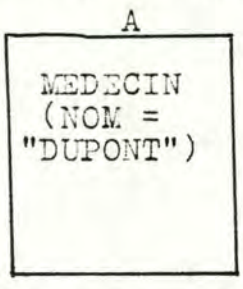
-) les renseignements généraux sur cette maladie
-) les antécédents du patient
-) les causes du décès (s'il y en a)
-) les remèdes antérieurs
-) les médicaments antérieurs
-) les maladies antérieures
-) les renseignements généraux d'une consultation
-) les symptômes, remèdes, médicaments et analyses d'une consultation.

Avant d'accéder à un point particulier, il peut demander un résumé des consultations qui ont eu lieu pour cette maladie.

V. L'AFFICHAGE DE STATISTIQUES.

Le principe du programme est basé sur la manipulation d'ensembles. Ces ensembles sont représentés par une boîte avec un nom (A,B,C,D,E,F). Cette boîte contient des lignes qui peuvent être remplies. Ce qui sera écrit dans ces lignes sera la définition de l'ensemble manipulé.

exemple :



ensemble de nom "A" et qui contient tous les médecins dont le nom est "DUPONT".

Au départ, toutes les boîtes sont vides.
La première ligne de l'écran apparaît comme suit :

C)réer D)étruire F)in R)ésultat T)raiter V)isualiser

et le curseur est en bas de l'écran près du prompt " @ " :
il attend une réponse (exemple: F)

La réponse à fournir est une action qui peut être réalisée avec ces ensembles :

F -----> fin du programme
C -----> créer un nouvel ensemble
D -----> détruire un ensemble existant
T -----> traiter deux ensembles (les combiner, ...)
V -----> visualiser le contenu d'un ensemble
R -----> exprimer la forme du résultat à visualiser

Je vais donc expliquer chaque action en particulier.

A) La fin du programme.

En introduisant F, on peut sortir du programme et retourner aux commandes principales du VAX.

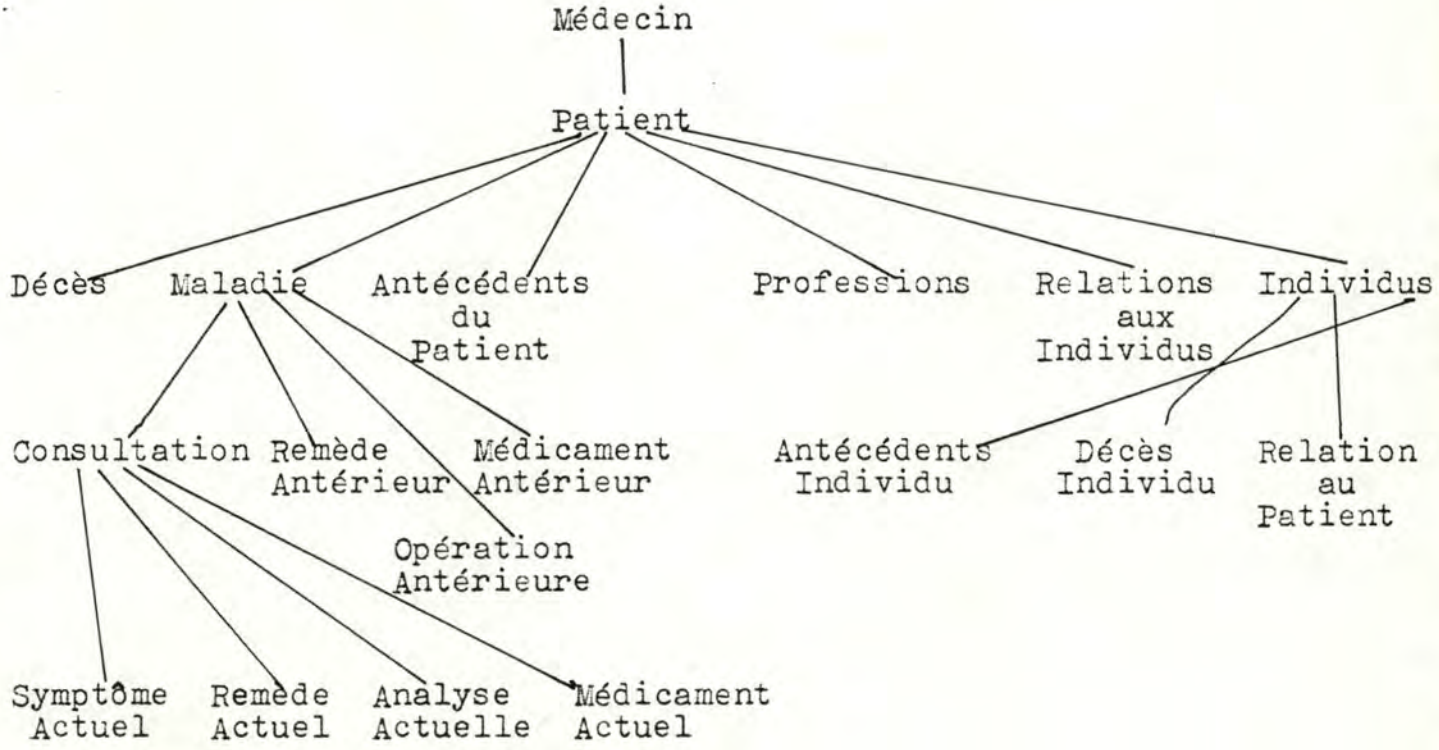
B) Créer un nouvel ensemble.

En introduisant C, on demande au programme de prévoir la place pour la définition d'un nouvel ensemble. S'il reste une place, le prompt de la dernière ligne deviendra "#".
A ce moment-là, il faudra introduire une phrase définissant l'ensemble et respectant une certaine syntaxe (l'exposé de la syntaxe suivra). Si la phrase est correcte, elle apparaîtra dans une boîte qui était vide et le programme reviendra aux commandes d'une action à réaliser:

exemple : @ C
MEDECIN (NOM = "DUPONT")
@

Explication de la syntaxe :

La syntaxe repose sur la hiérarchie des objets qui suit :



Remarques :

-) pour plus de facilité, des noms plus courts devront être donnés aux noeuds de la hiérarchie :

ex. : MEDECIN ------> MED

-) les caractéristiques de chaque noeud devront posséder également un nom :

ex. : pour le noeud "MEDECIN" le nom du médecin sera désigné par "NOMMED".

-) il faudra pour ces deux remarques consulter des listes établies par les personnes qui initialiseront le programme.

Pour établir une phrase correcte, il faudra donc suivre un chemin dans la hiérarchie.

Exemple$_1$:"le médecin qui a prescrit le remède actuel "ARSENICUM"" sera:

```
MEDECIN(PATIENT(MALADIE(CONSULTATION(REMEDE ACTUEL(NOM =
                                                "ARSENICUM")))))
```

Exemple$_2$: "L'ensemble des remèdes actuels dont le nom est "ARSENICUM". REMEDE ACTUEL (NOM = "ARSENIVUM")

Exemple 3 : "Le médecin qui a plus de dix patients".

MEDECIN (# PATIENT > = "10")

Caractéristiques :

-) à toute parenthèe ouvrante doit correspondre une parenthèse fermante.

-) la valeur de comparaison doit être entourée de guillements ex. "ARSENICUM".

-) les opérateurs de comparaison admis seront les suivants :

= ou ==	égalité
>	strictement plus grand
<	strictement plus petit
<> ou ><	différence
=> ou >=	plus grand ou égal
=< ou <=	plus petit ou égal

-) on dispose d'un opérateur de cardinalité : "#"
Celui-ci permettra d'utiliser la comparaison à un nombre d'occurences. ex. le nombre de patients

PATIENT

Cet opérateur porte sur un noeud de la hiérarchie.

C) Détruire un ensemble.

La lettre D permet de vider une boîte de son contenu. Pour cela, il faut introduire D puis la lettre qui identifie la boîte à vider.

exemple : @ D / # A ⇒ videra la boîte "A" de son contenu.

D) Traiter deux ensembles.

La lettre T permet d'effectuer des opérations ensemblistes.

exemple : @ T / # A = B∪A ⇒ on met dans la boîte "A" à la fois le contenu de "B" et celui de "A".

exemple : @ T / # C = B\A ⇒ on met dans la boîte "C" les éléments de "B" auxquels on a enlevé les éléments de "A".

exemple : @ T
#F = D^E ⇒ on met dans la boîte F les éléments qui sont à la fois dans la boîte D et la boîte E.

Syntaxe :

[nom d'une boîte] = [nom d'une boîte] { U ∩ \ } [nom d'une boîte]

E) Visualiser un ensemble.

En introduisant la lettre V et le nom de la boîte à visualiser, on obtient un nouvel écran. Cet écran contient une zone particulière de cinq lignes visibles. En fait, cette zone contient dix lignes. Pour voir les lignes cachées, il suffit de faire descendre le curseur (avec la touche "TAB") ou de le faire remonter (avec la touche "BS").
Lorsqu'il y a plus de dix éléments dans une boîte, la touche de retour permet de continuer la visualisation.

F) Exprimer la forme du résultat.

Cette fonction permet de ne retenir que les caractéristiques importantes et permises des éléments d'une boîte pour la visualisation.

exemple : on a créé la boîte "A" : MEDECIN (NOM = "DUPONT")
avant de dire qu'on veut visualiser, il faut dire le résultat qu'on veut visualiser

@ R
# LOCALITE^NOMBRE HABITANTS LOCALITE : MEDECIN

ceci veut dire que la visualisation affichera pour chaque médecin retenu dans la boîte "A" le nom de sa localité et le nombre d'habitants de cette localité.

Remarques : -) LOCALITE et NOMBRE HABITANTS LOCALITE seront des mots définis pour MEDECIN

-) syntaxe :
nom_1^ nom_2^ nom_3^....^nom_6 : nom du noeud concerné.
(maximum)