Contribution des systèmes a bases de connaissances en sciences de l'eau ; promise : un simulateur de projet ; moise : un système de diagnostic en assainissement autonome

François-Noël Cres

ECOLE NATIONALE SUPERIEURE
DES MINES DE SAINT ETIENNE

N° D'ORDRE : *35 HD*

THESE
Présentée par

François-Noël CRES

Ingénieur ISIM

pour obtenir le titre de

DOCTEUR

de l'ECOLE NATIONALE SUPERIEURE des MINES de PARIS

et de l'ECOLE NATIONALE SUPERIEURE des MINES de SAINT-ETIENNE

(Spécialité : Hydrologie et Hydrogéologie Quantitatives)

Contribution des systèmes à bases de connaissances en sciences de l'eau;
Promise : un simulateur de projet;
Moïse : un système de diagnostic en assainissement autonome.

soutenue à Saint-Etienne le 27 novembre 1989

COMPOSITION du JURY :

M.	**P. HUBERT**	*Président*
MM.	**P. DAVOINE**	
	J. QUINQUETON	*Rapporteurs*
MM.	**D. GRAILLOT**	
	D. PETER	
	J.-P. RIDEAU	*Examinateurs*

ECOLE NATIONALE SUPERIEURE
DES MINES DE SAINT ETIENNE

N° D'ORDRE : *35 HD*

THESE

Présentée par

François-Noël CRES

Ingénieur ISIM

pour obtenir le titre de

DOCTEUR

de l'ECOLE NATIONALE SUPERIEURE des MINES de PARIS

et de l'ECOLE NATIONALE SUPERIEURE des MINES de SAINT-ETIENNE

(Spécialité : Hydrologie et Hydrogéologie Quantitatives)

Contribution des systèmes à bases de connaissances en sciences de l'eau; Promise : un simulateur de projet; Moïse : un système de diagnostic en assainissement autonome.

soutenue à Saint-Etienne le 27 novembre 1989

COMPOSITION du JURY :

M.	**P. HUBERT**	*Président*
MM.	**P. DAVOINE**	
	J. QUINQUETON	*Rapporteurs*
MM.	**D. GRAILLOT**	
	D. PETER	
	J.-P. RIDEAU	*Examinateurs*

REMERCIEMENTS

Je remercie ici tout particulièrement les membres du jury :

Monsieur P. HUBERT, professeur à l'école des Mines de Paris, d'avoir accepté de présider ce jury et de m'avoir accueilli dans la formation doctorale qu'il dirige;

Monsieur P. DAVOINE, professeur des sciences de l'eau à l'école des Mines de Saint-Etienne, qui a tout particulièrement suivi le développement du simulateur de projet "Promise" et qui a bien voulu le tester en concevant une base de connaissances;

Monsieur J. QUINQUETON, directeur de recherche au centre de recherche en informatique de Montpellier, qui a accepté d'examiner ce travail quelque peu éloigné de ses activités habituelles;

Monsieur D. PETER, chargé de mission au ministère de l'environnement, qui nous a, dès le début, encouragés à développer les applications télématiques de "Moïse";

Monsieur J.-P. RIDEAU, directeur de l'antenne de Clermont-Ferrand de l'agence financière de bassin Loire-Bretagne, qui a testé les règles de "Moïse" et nous a conforté dans notre approche du problème;

enfin, Monsieur D. GRAILLOT, directeur de recherche à l'école des Mines de Saint-Etienne, ingénieur à la société RHEA, qui m'a fait confiance pour mener à bien ces travaux, et qui a su les orienter, les contrôler et les discuter; il est l'initiateur de nombre d'idées et de thèmes développés dans ce mémoire.

J'exprime toute ma reconnaissance à Monsieur A. MATHON, directeur du département "Stratégie du Développement" et professeur à l'école des Mines de Saint-Etienne, pour m'avoir accueilli au sein de son équipe et pour m'avoir fourni le cadre et les moyens propices pour mener à bien ces travaux.

Monsieur G. de Marsily, professeur à l'Ecole des Mines de Paris et à l'université Paris VI, et Monsieur E. LEDOUX, professeur à l'école des Mines de Paris m'ont

régulièrement reçu afin de discuter des orientations prises, et d'apporter leur expérience à la cohérence des options retenues; je les en remercie vivement.

Je remercie toutes les personnes qui ont croisé ma route durant ces deux années, pour l'aide qu'elles m'ont apportée; entre autres :

Monsieur FERRAND, directeur de la Direction Départementale de l'Action Sanitaire et Sociale de la Loire, a fourni les cas concrets pour tester les règles de "Moïse";

Monsieur ANICET, ingénieur au Service Régional d'Aménagement des Eaux Rhône-Alpes, a fourni les données de terrains pour le simulateur "Promise",

Monsieur NICAISE, président du Syndicat d'Aménagement des Eaux de la Varèze, nous a convié aux réunions du syndicat et a autorisé la diffusion des études réalisées;

Monsieur NARDIN, maire de Vernioz, a guidé nos pas sur sa commune;

Monsieur DI BENEDETTO, professeur à l'école des Mines de Saint-Etienne, a fourni la matière à une extrapolation de la méthodologie de "Moïse" vers les guides de choix en instrumentation chimique.

J'exprime toute ma sympathie à tous les membres du département "Stratégie du Développement" de l'école des Mines de Saint-Etienne, et plus particulièrement :

Marie-Agnès GIRARD qui a acceptée d'être un "utilisateur-cobaye" du simulateur "Promise" et Philippe BEAUNE, avec qui j'ai partagé un bureau et de longues heures devant un ordinateur, notamment pour la mise au point de la version télématique de "Moïse".

Agnès L'AOUR-DUFOUR, ingénieur hydrologue, analyste au Groupe Français d'Informatique et -surtout- mère de mon fils, a relu tant sur la forme que sur le fond ce mémoire et a grandement contribué à son élaboration. Elle a toujours soutenu cette entreprise qu'est la conception et la rédaction d'une thèse; Pour tout cela, est-il vraiment utile de dire que je la remercie ?

SOMMAIRE

Annexes

Introduction

« Il venait de poser là "LA GRANDE QUESTION" : l'auto-organisation est-elle le début de l'intelligence ?

Si vous admettez l'idée selon laquelle l'organisation spontanée des tornades ou des tourbillons de BENARD sont "un peu d'intelligence", vous aurez "compris" mon livre.

Si vous pensez, comme moi, que "la prise de décision" par un être humain est plus proche du mécanisme de formation d'une tornade que du test "si ..alors ..sinon .." des ordinateurs, alors cet ouvrage aura dépassé son objectif. »

J. Cl. PEREZ [PEREZ 1988]

J. Cl. PEREZ est-il un provocateur ?

Mettre de l'intelligence dans une tornade et reléguer le raisonnement "si ..alors ..sinon .." au stade du "B, A, BA" de l'analphabète qui veut apprendre à lire, il y a de quoi surprendre plus d'une personne.

Cette entrée en matière, sorte de traitement de choc, est destinée à mettre en évidence combien la volonté de comprendre les mécanismes du raisonnement (humain à priori) est devenue un problème d'actualité.

Mieux, il semble que pour la première fois, on ait un outil d'investigation permettant de mettre en œuvre les modèles de représentation de la connaissance : l'ordinateur.

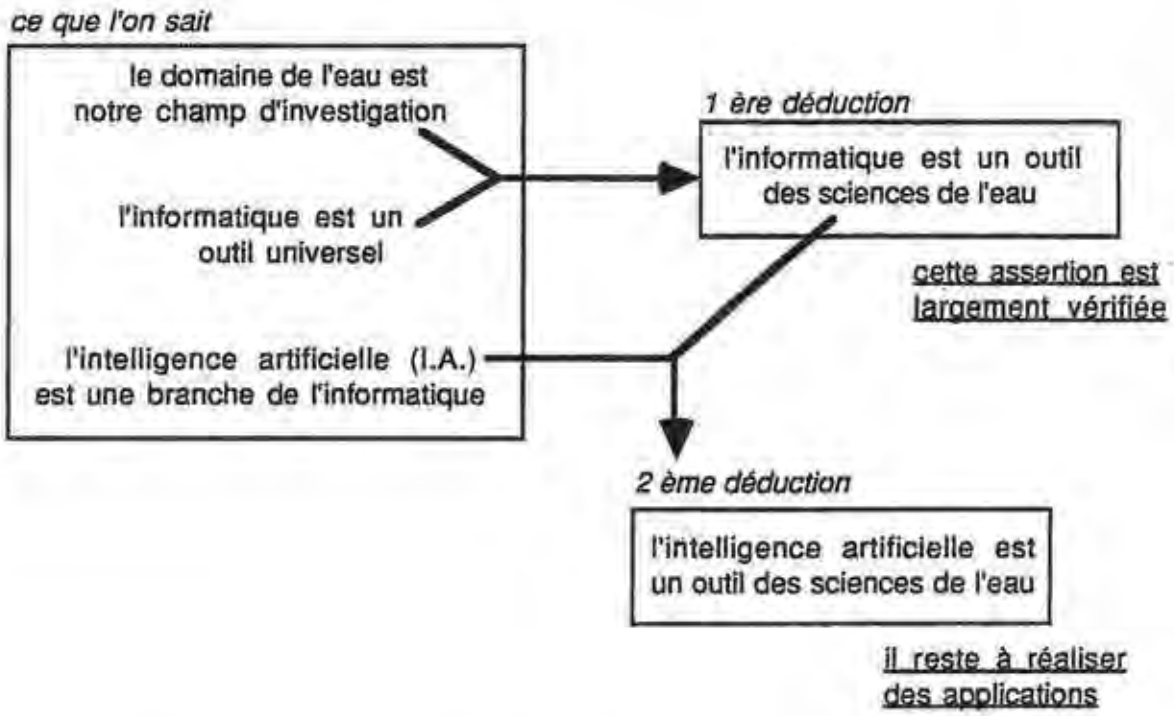
Ce dernier deviendra-t-il notre alter ego, voire, nous dépassera-t-il dans ses capacités de raisonnement ?

En d'autres termes, est-il possible de faire mieux en matière d'intelligence que notre cerveau ?

Bref, que le lecteur désabusé se rassure, c'était là la dernière touche de rêve et de provocation, et ce mémoire de thèse ne prétend, ni même n'envisage de répondre à ces questions.

Notre ambition est bien plus modeste.

Puisque l'on va beaucoup parler de raisonnement, de règles de décisions dans les pages qui suivent, nous pourrions mettre sous forme d'un schéma de déduction l'origine profonde de nos travaux :



Pour simpliste que soit ce schéma, il n'en reste pas moins vrai et pertinent :

au même titre que l'outil mathématique et ses sous-ensembles (analyse, algèbre, statistiques ...) équipent indifféremment de nombreuses structures scientifiques (on entend par "structures scientifiques" des domaines de la science au sens large), l'outil informatique, qui a fait son apparition il y a une trentaine d'années, devient l'instrument permettant l'analyse, la prévision et la compréhension de nombreux phénomènes. Bref, l'outil 'informatique' a acquis un caractère d'universalité.

L'intelligence artificielle

" *l'I.A. cherche à comprendre les mécanismes de compréhension* " [FARRENY, GHALLAB 1988].

L'intelligence artificielle est une branche de l'informatique; mais c'est aussi une branche de la psychologie, la linguistique, la neurologie ...

Loin de vouloir aborder simultanément toute les facettes de l'I.A., la branche qui va nous intéresser dans ce mémoire est la branche informatique.

Et nous allons même aller encore plus loin dans la décomposition des applications possibles de l'I.A. en nous intéressant aux *outils informatiques* utilisés en I.A.

Autrement dit, à partir de la question:" je pense, donc comment est-ce que je fais ?" des outils spécifiques (Prolog, systèmes experts) ont été mis au point pour aborder la réponse. Nous nous proposons de les réutiliser afin de développer des *applications dans le domaine de l'eau.*

Deux aspects sont évoqués dans l'alinéa précédent :

Δ L'innovation : bien que les termes "Prolog" et "systèmes experts" soient désormais entrés dans notre vocabulaire courant, leur utilisation pratique est encore loin d'être quotidienne. Cela peut tenir à plusieurs raisons :

- manque de formation des utilisateurs;
- moyens techniques limités : la diffusion de l'I.A. va de pair avec l'explosion technologique des ordinateurs; cette explosion ne fait que commencer par la montée en puissance des micro-ordinateurs;
- manque de besoins : les entreprises, pour lesquelles l'informatisation est une nécessité vitale, ont, pour la grande majorité d'entre elles, d'autres priorités que la mise en œuvre de tels outils.

Δ L'adaptation : c'est ce que nous appelons "applications". Cet aspect peut parfois conduire à une dégradation des outils à la manière de la formule populaire " prendre un marteau pilon pour écraser une mouche". Nous ne pensons pas être tombés dans ce travers. Inversement, le recours à Prolog (langage dit de 4ème

génération) nous a permis d'atteindre efficacement les objectifs fixés, sans pour autant apparaître comme un langage à la puissance démesurée par rapport au problème.

Au contraire, l'utilisation d'un langage moins puissant se serait soldé par un délai de réalisation beaucoup plus long, et une souplesse d'utilisation réduite. Certes, nous n'avons pas prouvé expérimentalement cette dernière assertion; c'est cependant une "heuristique" à laquelle nous accordons une très grande confiance.

En résumé ...

- des hommes cherchent à comprendre comment ils pensent et inventent un nouveau domaine de la science : l'intelligence artificielle.

- les informaticiens s'intéressent à ce nouveau domaine et produisent de nouveaux langages et de nouvelles techniques pour modéliser le raisonnement.

- nous utilisons ces nouveaux langages et nouvelles techniques pour 2 applications dans le domaine de l'eau. Le but originel (comprendre le raisonnement) est relégué au second plan.

La mise au point des applications n'en demeure pas moins intéressante et importante, et permet une valorisation, au moins à court terme, des recherches en I.A.

Les 2 applications évoquées, développées en Prolog et basées sur la structure des systèmes à base de connaissances (que nous développerons plus loin) sont un simulateur de projet (baptisé "Promise") et un système expert en assainissement individuel (baptisé "Moïse"). La suite de ce mémoire en décrit la structure et le fonctionnement.

Auparavant, une analyse bibliographique récapitule les buts et les moyens de l'Intelligence Artificielle, et tente de restituer l'apport possible de l'I.A. dans les sciences de l'eau.

Intelligence artificielle et systèmes experts

Analyse bibliographique

Applications aux sciences de l'eau

1. HISTOIRE DE L'INTELLIGENCE ARTIFICIELLE

1.1 CURRICULUM VITAE

Nom	:	**Intelligence Artificielle**
Age	:	plus de 30 ans
Née à	:	conférence de DARMOUTH College (HANOVER, NEW HAMPSHIRE)
Père	:	J. McCARTHY
Parrains	:	M. MINSKY, A. NEWEL, H. SIMON, C. SHANNON ...

Cette présentation pourrait faire croire à une sorte de génération spontanée. Bien évidemment, il n'en est rien.

Mise à part la naissance de l'univers (pour laquelle on a bien du mal à imaginer un "avant"), on sait que depuis ce fameux "Big-Bang", tout est évolution : « rien ne se perd, rien ne se crée, tout se transforme » a remarqué LAVOISIER.

L'I.A. obéit à ce principe d'évolution et constitue la résultante d'une quête incessante. C'est en fait cette association, attirante et provocante, des 2 mots "intelligence" et "artificielle" qui attire l'œil et retient l'esprit. Ce nom de baptême est déjà tout un programme.

Le but que McCARTHY et ses collaborateurs fixaient en 1956 peut se résumer ainsi :

" *reproduire et initier un comportement intelligent sur une machine*".

Ce comportement intelligent serait à l'image de l'homme (c'est sous-entendu) et certains l'affirmèrent, peut-être plus puissant. SIMON ne prévoyait-il pas une dizaine d'années avant que la machine ne dépasse l'homme ? ...

A quand remonte cette idée ? Elle n'est concrètement pas très vieille et correspond à l'avènement de l'ordinateur. Remarquons tout de même que l'homme a souvent identifié le fonctionnement de son cerveau avec les principes technologiques à sa disposition. Ainsi les romains imaginaient une sorte de réseau d'égouts, puis à l'époque des standards téléphoniques, la comparaison fut aisée : une image et un nom, 2 idées, une

couleur et une impression seraient reliés à la manière de la connexion qu'établit la standardiste entre 2 abonnés.

Dans ce type d'analogie, l'ordinateur a détrôné ses prédécesseurs. Ce coup d'état imparable provient des concepts alors associés au raisonnement; c'est ce qu'écrivait HOBES en 1958 : « ... lorsqu'un individu raisonne, il ne fait rien d'autre que de concevoir la résultante d'opérations parcellaires car le raisonnement n'est rien d'autre qu'un calcul ...» [dans DREYFUS 1984].

Qu'est-il advenu des développements espérés à cette époque ?

A la manière de 3 expériences professionnelles (style curriculum vitae), nous distinguerons 3 phases [PEREZ 1988] :

1956 - 1965 : l'illusion

Cette illusion repose sur des faits et des enchaînements simples :

- le raisonnement est comparable à un calcul
- l'ordinateur sait calculer très vite
- l'ordinateur peut manipuler des symboles aussi rapidement que des nombres donc l'ordinateur doit pouvoir raisonner.

1965 - 1980 : la traversée du désert

Le problème apparaît beaucoup plus complexe que prévu. C'est probablement aussi l'époque d'une désillusion pour beaucoup de chercheurs. Ces 15 années furent cependant l'occasion d'un mûrissement des idées et du développement d'un certains nombres d'applications : systèmes experts, perception-vision par ordinateurs, robotique, interfaces homme-machine, compréhension de la parole et des langues naturelles en contexte restreint, environnement de programmation et premières machines de traitement symbolique [FARRENY, GHALLAB 1988].

depuis 1980 : l'éveil et le décollage de l'I.A.

La meilleure illustration en est le défi japonais des ordinateurs de 5ème génération, projet planifié sur 10 ans. Les systèmes experts (on retrouve dans ces termes le même aspect "magique" et "mystérieux" que dans "intelligence artificielle") représentent assurément un acquis. Encore que l'on verra plus loin les limites de tels systèmes. Ce succès va de pair avec de grands progrès technologiques permettant une diffusion grand public des logiciels issus de l'I.A.

1.2 BILAN

1.2.1 Qu'est-ce que l'intelligence ?

Sur ce point, il semble que les progrès soient très maigres. Nous ne nous lancerons pas dans une tentative de définition. Chacun garde sa vision du phénomène et la traduit en des termes différents à l'image de ces 3 citations :

« c'est une faculté intérieure comme la vue et l'ouïe sont des facultés tournées vers l'extérieur »

Swami RITAJANONDA

« l'intelligence, c'est l'agilité avec laquelle on se déplace dans le champ de ses propres connaissances; tandis que la créativité consiste à s'élever au-dessus »

Daniel MORENO

« l'intelligence, ça n'existe pas. C'est un faux mot qui sert à gratifier des gens capables de répondre à des tests de QI imaginés par des personnes qui se croient intelligentes »

Henri LABORIT

Aucune de ces définitions n'est complètement fausse. Toutes sont probablement incomplètes. Cette remarque suffit pour mettre le doigt sur la gageure qu'ont entrepris les chercheurs en I.A. : comment inculquer à une machine une notion que l'on ne sait même pas définir ?.

En ce qui concerne l'apparition de l'intelligence, 2 écoles s'affrontent : une vieille école qui suppose que l'intelligence est implantée sur un support (à la manière d'un programme sur une machine) et une école plus récente qui envisage plutôt "l'émergence" de l'intelligence à travers une structure organisée, la cellule de base pouvant être quelque chose de très simple.

Trop osé, trop philosophique, penseront certains. C'est pourtant ce qui apparaît en toile de fond des chercheurs en I.A. avec l'approche "d'en bas" ("bottom-up") à

comparer avec l'approche plus classique "d'en haut" ("top-down"). C'est ce que nous allons voir dans le paragraphe suivant.

1.2.2 l'intelligence artificielle

1.2.2.1 l'approche "d'en bas" ("bottom-up")

Cette approche est plus connue sous d'autres noms : réseaux neuronaux, connexionisme, neuromimétisme ...

Nous ne nous étendrons pas sur ces différents termes. Elle repose sur :

- une philosophie résumée par Bernard ANGENIOL, responsable d'une unité de recherche sur les réseaux neuronaux chez THOMSON-CSF [d'après 'le monde informatique' du 12/12/1988] : « les réseaux neuronaux proposent de copier le comportement inconscient en partant du bas niveau, c'est-à-dire des données sensorielles »;
- une technologie encore bafouillante à base d'une architecture dite "parallèle de masse" qui permettrait de reconstituer l'organisation des neurones de notre cerveau.

Contrairement aux ordinateurs séquentiels actuels, de tels réseaux ne se programment pas par des instructions, mais par apprentissage au moyen d'exemples. Nous renvoyons le lecteur à la bibliographie pour plus d'informations (notamment [LA RECHERCHE 1988]).

1.2.2.2 l'approche "d'en haut" ("top-down")

Cette approche est la plus avancée actuellement. Elle agit à un niveau symbolique en « s'attaquant aux problèmes à partir d'un haut niveau, en copiant la partie consciente du raisonnement humain » (Bernard ANGENIOL).

La partie apparente, la plus connue, de cet iceberg est constituée par les systèmes experts. De nombreux domaines d'applications sont abordés par cette voie : traitement du langage naturel, recherche intelligente dans des bases de données, démonstration de théorèmes, résolution de problèmes gourmands en temps ...

1.2.2.3 comparaison des 2 approches

Il s'agit en fait de 2 approches très classiques que l'on peut retrouver dans de nombreux domaines de la science. Elles correspondent à 2 visions possibles :

- l'esprit de géométrie qui cherche à comprendre le niveau microscopique pour expliquer et avancer dans le niveau macroscopique;
- l'esprit d'analyse qui ne considère que la résultante des niveaux microscopiques pour ne prendre en compte que le niveau macroscopique.

La chimie fournit un excellent exemple de ces deux approches : L'esprit d'analyse permit de la hisser, au dix-neuvième siècle, au rang de science, sans que fut encore découvert le modèle atomique (esprit de géométrie). En ce qui concerne les processus décisionnels - donc le raisonnement - SIMON postule qu'il est possible de les étudier et d'en obtenir une représentation informatique à " un niveau d'agrégation qui est celui du traitement de l'information, c'est-à-dire celui de la manipulation des signes " [dans LEVINE, POMEROL 1989].

En rapport avec les capacités techniques actuelles, presque toute la recherche en I.A. s'est jusqu'à maintenant focalisée sur l'approche analytique, l'approche géométrique étant une voie de recherche à l'état embryonnaire mais peut-être plus prometteuse car plus représentative des phénomènes physiologiques régissant la manière de raisonner de l'homme.

D'un point de vue informatique, le tableau 1-1 résume les différences et les possibilités des machines issues des 2 approches.

1.3 LES SYSTEMES EXPERTS

Après avoir passé rapidement en revue d'une façon globale l'Intelligence Artificielle, ses tenants et ses aboutissants, nous allons maintenant cerner de plus près un des aspects les plus développés : les systèmes experts.

Pour reclasser cet aspect parmi toutes les notions développées précédemment, on peut noter que les systèmes experts constituent un type d'approche analytique.

Ordinateurs conventionnels	Neuro-ordinateurs
NUMERIQUE/TEMPS DISCRET traite des informations codées entre 0 et 1 pour la précision par commutation de portes logiques synchronisées par les pulsations d'une horloge.	ANALOGIQUE/TEMPS CONTINU Traite des informations codées par des signaux analogiques continus de basse précision, par transmission dans un réseau de processeurs en temps réel.
CALCUL SEQUENTIEL Un seul processeur traite séquentiellement quelques bits de données de la zone mémoire.	CALCUL MASSIVEMENT PARALLELE Les unités de traitement interconnectées traitent toutes les données en même temps.
MEMOIRE LOCALISEE Enregistre l'information dans une zone dédiée à la mémoire. L'adresse physique permet de retrouver facilement chaque donnée.	MEMOIRE ASSOCIATIVE DISTRIBUEE DANS LE RESEAU Enregistre l'information de manière répartie, par la modification des poids des connexions du réseau. Chaque donnée rappelle automatiquement les informations qui lui sont reliées.
LOGIQUE BOOLEENNE Prend des décisions OUI/NON basées sur des fonctions logiques.	LOGIQUE FLOUE Prend des décisions pondérées à partir de données floues incomplètes ou contradictoires.
RESULTAT EXACT Trouve des réponses précises à un problème dans des délais parfois prohibitifs.	RESULTAT APPROCHE Trouve rapidement de bonnes solutions approchées pour des problèmes très complexes.
PROGRAMMABLE PAR INSTRUCTIONS Manipule des données de manière structurée. Les opérations sont toujours sous contrôle et les résultats prévisibles. Adapté à l'exécution de taches séquentielles et dur à programmer par expériences.	PROGRAMMABLE PAR EXPERIENCE Formule de manière spontanée ses propres méthodes de traitement de l'information par auto-organisation lors de l'adaptation des connexions. Mal adapté à la programmation séquentielle car les récursions et les boucles sont dures à implémenter en termes de réseaux.
SENSIBLE AUX PANNES MATERIELLES La défaillance d'un seul composant de la machine peut avoir des conséquences catastrophiques.	TOLERANT VIS-A-VIS DES PANNES MATERIELLES Les performances se dégradent graduellement en fonction des défaillances des composants car l'information et le traitement sont distribués sur plusieurs unités.

[Source Ambassade de France aux USA
d'après Le Monde Informatique du 12 décembre 1988]

Tableau 1-1 : Comparaison entre ordinateurs
conventionnels et neuro-ordinateurs

1.3.1 définition

Ils sont à l'origine du succès médiatique et industriel de l'I.A. Il existe de nombreuses définitions dans une littérature désormais abondante sur ce sujet. Nous retiendrons celle de H. FARRENY [FARRENY 1985] :

> les systèmes experts sont des logiciels peut-être bientôt aussi des matériels
> destinés à remplacer ou assister l'homme dans des domaines
> où est reconnue une expertise humaine :
> - insuffisamment structurée pour constituer une méthode de travail précise, sûre, complète, directement transposable sur ordinateur;
> - sujette à révisions ou compléments (selon l'expérience accumulée).

Une telle définition fait apparaître au moins deux conditions pour implanter un système expert (S.E.):

- l'absence d'algorithme facilement programmable pour résoudre le problème
- la nécessité de prévoir une maintenance facile.

Ce sont à notre avis ces 2 critères qui constituent l'amorce de l'implantation des S.E. On verra dans la suite de ce mémoire que les 2 applications que nous avons développées répondent à ces 2 conditions.

D'autres auteurs ajoutent également comme conditions :

* l'existence d'un domaine bien délimité et bien défini. Cette condition traduit la difficulté que peut éprouver le logiciel à prendre en compte un environnement non prévu, non programmé. Elle assimile ainsi l'expérience déçue des premiers temps de l'I.A. où l'on pensait pouvoir faire des systèmes généraux. De plus, elle est en accord avec les termes du théorème de GODEL qui stipule, en l'appliquant au domaine de l'expertise, que tout système formel qui veut représenter la connaissance dans sa globalité (donc dans sa complexité) se heurtera soit à un problème d'incomplétude, soit à un problème d'inconsistance [DELAHAYE 1989].

* la collaboration d'un expert résolvant les problèmes de ce domaine. Le problème d'extraction de la connaissance d'un expert est encore entier. La

méthodologie la plus en vogue jusqu'à maintenant consiste à faire venir un 'cogniticien', se doublant d'un psychologue, qui va par un jeu de questions savamment distillées "tirer les vers du nez" de l'expert et pouvoir ainsi programmer la machine. Nous ne pensons pas que ce soit la meilleure façon d'aborder une expertise, car elle ne met pas l'expert en situation, mais c'est la seule actuellement opérationnelle. D'autres méthodes à partir de simulation ou d'apprentissage seraient plus appropriées, mais elles n'en sont qu'à un stade expérimental.

* un intérêt économique directement lié à la résolution du problème par un S.E. Ce n'est pas une Lapalissade, mais presque. On pourra aussi évoquer des intérêts autres qu'économiques, pédagogiques par exemple, comme nous le développerons dans un de nos exemples.

1.3.2 Structure et typologie des systèmes experts

Les terminologies employées pour décrire la structure d'un système expert sont nombreuses. Nous distinguerons deux entités :

- le moteur (ou interpréteur de règles) capable d'inférer (de déduire logiquement) à partir des règles (ou instructions). Par extension, on pourra parler de "générateurs de systèmes experts" (G.S.E) , c'est-à-dire des logiciels offrant un environnement pour développer des systèmes experts (éditeur de règles, visualisation des chemins empruntés dans la logique ...);
- les connaissances que l'on décompose en :
  - base de règles : ce sont les connaissances accumulées concernant un certain domaine;
  - base de faits : elle accumule les résultats et acquis temporaires permettant au couple moteur-base de règles de résoudre un problème (il s'agira par exemple de la valeur acquise par différents paramètres, ou du stockage des déductions déjà réalisées ...).

D'un point de vue conceptuel, cette organisation est somme toute classique et on retrouve le canevas habituel (programme ou moteur - données ou base de connaissances). Cependant, il y a une différence qui fait la force des systèmes experts,

et plus généralement des langages de l'I.A. par rapport aux programmes habituels tels qu'on les rencontrait jusqu'à présent. Le programme habituel décrit le "comment résoudre un problème" et les données apportent les bases réelles et concrètes du sujet.

En I.A., le "comment" va être implicitement contenu dans la base de règles; « toute la spécificité du problème à traiter est transférée vers l'organisation des données (base de règles et base de faits)» [PEREZ 1988]. Le programme perd quant à lui son caractère de maître des opérations pour devenir une sorte de procédure permettant à la base de règles de déduire : c'est devenu le "moteur" dont la complexité est complètement transparente à l'utilisateur.

La technologie des S.E. peut s'appliquer partout où la formalisation de la connaissance peut difficilement être mise sous forme procédurale : cette caractéristique ouvre de nombreux domaines d'application que nous résumons dans la figure 1-1 suivante [CARSALADE 1988; RETOUR 1984] :

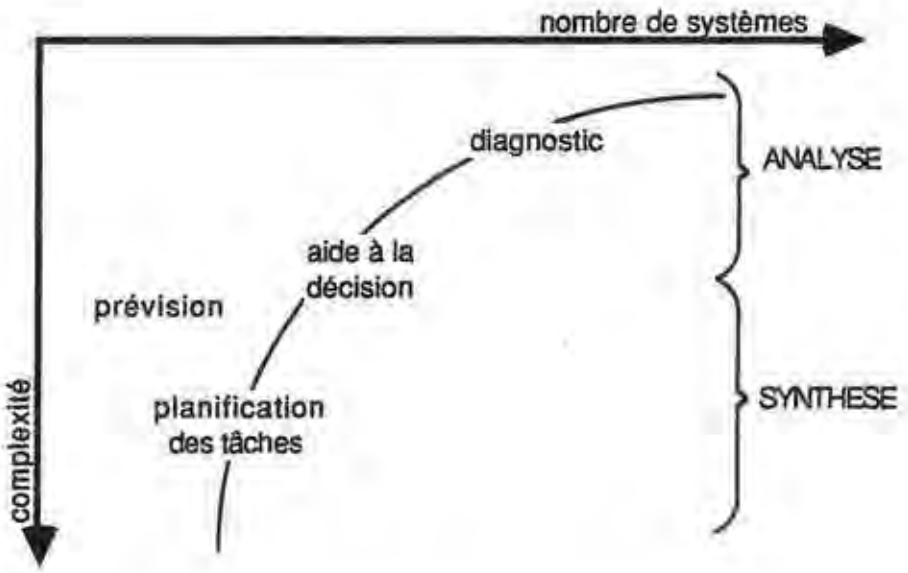


figure 1-1 : typologie des systèmes experts

1.4 L'ANTITHESE DE L'I.A.

Cette étude bibliographique serait incomplète sans un développement des travaux de Hubert DREYFUS [DREYFUS 1984] qui fixe les limites de l'I.A.

Face à un profane qui lui demanderait : « est-ce qu'un système expert peut remplacer un homme ? », H. DREYFUS répond non, en tout cas, pas avec les méthodes analytiques actuelles, et pas avant très longtemps [d'après BOUVERESSE 1985].

En d'autres termes, pour DREYFUS, un expert ne raisonne pas avec un ensemble de tests du style " Si ..Alors .. Sinon ..", mais plutôt à partir d'une globalisation du contexte, une image mentale en quelque sorte. Nous avons pris deux exemples afin d'illustrer la pensée de H. DREYFUS, qui classe les capacités des individus selon leur expertise (tableau 1-2).

Ainsi, lorsque le cogniticien questionne l'expert afin de programmer ses connaissances et son raisonnement, il lui demande de traduire en termes de débutant (à partir de tests élémentaires) une réflexion de synthèse élaborée à la suite d'une longue expérience dont on ignore finalement le mécanisme de génèse. Autrement dit, l'expert ne raisonnerait pas à partir d'une analyse, mais plutôt à partir d'un apprentissage. *L'expérience ne peut pas se réduire à des règles.*

H. DREYFUS constitue un précieux contradicteur en I.A. qui ramène un peu cette nouvelle science aux limites de la réalité, qu'elles soient technologiques, philosophiques ou "de bon sens".

Notre point de vue se rapproche beaucoup de celui de H. DREYFUS. La définition qu'il avance de l'expertise nous parait satisfaisante.

Il n'en demeure pas moins que de nombreux problèmes peuvent être abordés par une voie analytique, même si cela n'est plus de l'expertise au sens de H. DREYFUS, donc même si les solutions prônées par un système dit "expert" ne sont pas absolument parfaites. En allant plus loin, un S.E. qui ne résoud correctement que 80% des cas qui lui sont soumis, pourvu qu'il sache se taire dans les 20% restants, peut être considéré comme un bon outil d'aide à la décision par exemple, si rien de mieux n'avait été fait auparavant.

La question qui transparaît dans l'alinéa précédent est en fait de savoir si le but d'un S.E. est de remplacer un expert, ou bien d'intervenir comme un outil informatique d'aide à la décision ou autre. Un chercheur en Intelligence Artificielle visera la première option. Un décideur, dans le domaine de l'eau par exemple, sera ravi d'avoir un potentiel de la deuxième option.

NIVEAU	JEU D'ECHECS	AMENAGEMENT en EAU
Débutant	Affecte des poids standards aux pièces indépendamment de leur position	Il conçoit une pluie centenale comme une pluie capable de se produire une fois tous les 100 ans. C'est le stade "déchiffrement" d'un tableau de valeurs sans capacité d'interprétation.
Débutant avancé	Il reconnaît certains aspects ou positions à un niveau plus global et relationnel entre les pièces.	Il se réfère à des indices régionaux pour pondérer ses résultats : il peut se hisser à un niveau d'abstraction lui permettant d'effectuer des comparaisons de cas.
Compétent	"Joueur de classe A", il choisit son but et focalise son attention sur tout ce qui peut servir ce but, prend d'éventuels risques, isolant arrière-plans et avant-plans, et donc, n'analysant pas certaines situations parce qu'elles ne servent pas son but.	Il choisit ses méthodes d'investigation en fonction de la qualité des données disponibles et des buts à atteindre. Il anticipe les réponses aux éventuelles critiques.
Efficace	"Maître d'échecs", reconnaît un large répertoire de types de positions. Apprécie inconsciemment le sens d'une position. Il sent qu'il doit plutôt attaquer ou plutôt se défendre, mais il ne peut pas dire avec précision pourquoi ...	Il intègre globalement de multiples paramètres sociaux, économiques et hydrologiques pour mener à bien ses projets.
Expert	Il agit par expérience; on a avancé qu'un maître d'échecs pouvait mémoriser et reconnaître 50000 situations types. Il agit par intuitions. Il n'utilise pas la partie analytique de son cerveau, ce qui a été vérifié avec l'exemple suivant : on a occupé le grand maître KAPLAN à additionner sans arrêt des nombres ... en simultanéité avec une partie d'échecs !	Il envisage une stratégie d'aménagement sachant que les niveaux tactiques et techniques se mettent inconsciemment en place, à la suite d'une longue expérience professionnelle.

(exemple du jeu d'échecs d'après [PEREZ 1988])

tableau 1-2 : exemples d'échelle de connaissances dans deux domaines

En résumé, il paraît douteux que, dans leur forme actuelle, les systèmes experts puissent égaler un expert humain, sauf dans certains cas particuliers et restreints. Il n'en est pas moins vrai qu'ils représentent une avancée au sens informatique, par la mise en service de nouveaux langages et d'une technique appropriée. C'est ce que nous tâcherons de mettre en évidence par la suite.

2. SYSTEMES EXPERTS ET HYDROLOGIE

2.1 DEVELOPPER UN SYSTEME EXPERT

2.1.1 Vous avez dit "systèmes experts" ?

A la lumière des propos précédents, nous nous proposons de tirer un certain nombre de conclusions relatives aux applications de l'I.A. aux sciences de l'eau. Tout d'abord, comme l'indique le titre de ce chapitre, nous nous limiterons aux systèmes experts. En effet, si l'on veut rester réaliste, il apparait que les machines et logiciels accessibles à l'hydrologue commun le limite au développement de systèmes experts (on ne connaît pas à l'heure actuelle d'applications de réseaux neuronaux dans les sciences de l'eau).

Ces développements doivent respecter 2 principes :

1er principe : *justifier l'utilisation d'un S.E.*

c'est-à-dire répondre à l'un ou aux 2 critères évoqués auparavant

- pas d'algorithme pour résoudre le problème;
- nécessité d'une maintenance aisée.

On peut se poser la question de savoir ce que signifient concrètement ces 2 critères. On pourra d'abord remarquer que le second découle souvent du premier : là où la technique de résolution n'est pas figée par un algorithme (une équation par exemple), il est fréquent que cette technique de résolution puisse évoluer, d'où la nécessité d'une maintenance aisée.

Prenons un exemple pour illustrer le premier critère : l'assainissement individuel que nous traiterons complètement plus loin. Le choix d'une filière d'assainissement va se faire en fonction de combinaisons de paramètres. On verra que ce nombre de

paramètres peut être variable, que chaque spécialiste y incorpore des notions plus ou moins précises, avec parfois certaines restrictions. Toutes ces imprécisions ouvrent la voie à une explosion combinatoire ainsi qu'à une série de cas particuliers non maîtrisables dans un algorithme pour aboutir à une solution.

Autre exemple que nous traiterons : un simulateur de projet; la notion d'algorithme s'éloigne encore plus puisqu'un projet est fait d'une suite de décisions, d'évènements plus ou moins aléatoires, d'interférences entre des objectifs... Le simulateur doit pouvoir restituer cet environnement avec une approche globalisante et réagir à toutes ces situations. Dans ce cas, un S.E. s'avère plus efficace pour modéliser le problème.

Dans les deux cas, la "maintenance aisée" correspond à une contrainte primordiale qui va influencer directement la structure de la base de connaissances. Peut-être certains s'interrogeront-t-ils : pourquoi la maintenance est-elle plus aisée avec un système expert ? (par le terme maintenance, nous entendons la mise à jour et l'actualisation des solutions qu'offre le S.E.). Cette particularité provient du transfert de compétences dont nous parlions plus haut entre les 2 entités { programme ou moteur } et { données ou base de faits/règles }. Par ce transfert, le moteur acquiert une certaine neutralité vis à vis de la base de connaissances qui contient finalement la logique de déduction. L'organisation de cette base fera que la maintenance sera plus facile que celle d'un programme pour lequel il faut posséder une maîtrise du langage et de sa programmation.

2ème principe : adapter l'outil au problème

Le marché Français actuel offre plus de 50 générateurs de systèmes experts [Le Monde Informatique de 30 mai 1988]. Leur nombre ne cesse d'augmenter. Les possibilités, les prix, les compatibilités aux systèmes d'exploitations, aux machines et de nombreux autres paramètres forment des combinaisons complexes. L'humour de certains leur fait déjà dire qu'il faudra bientôt un système expert pour s'y retrouver. Par ailleurs, on n'est pas obligé de choisir un générateur de systèmes experts (G.S.E.) pour développer un système expert. On peut repartir d'un langage (Lisp ou Prolog de préférence) pour développer un moteur adapté à son problème.

Face à ces choix et stratégies possibles, 3 critères peuvent être retenus :

• la stratégie et l'environnement du moteur : ce sont les capacités du moteur, en termes d'exploration et de mise au point de la base de connaissances. Ce critère très technique n'est interprétable que pour un bon spécialiste. Il traduit le fait que les possibilités demandées à un système expert ne sont pas les mêmes s'il s'agit d'un problème de diagnostic ou de planification. Cependant, il nous paraît important qu'un moteur puisse prendre en compte des variables dans ses règles (moteur d'ordre 0^+ ou 1). La présence du chaînage avant, arrière ou mixte est optionnelle suivant le problème à traiter. Les possibilités de visualisation graphique, d'interrogation sur le "pourquoi" d'une question du S.E. sont plus ou moins appréciées par les développeurs et les utilisateurs et dépendantes de la structure de la base de connaissances. Ces 2 possibilités sont à envisager pour une base de connaissances très arborescente; elles se sont avérées inutiles pour nos deux applications, c'est pour cela qu'elles n'ont pas été développées.

• l'intégration : c'est une question importante de savoir si des procédures, écrites dans des langages variés, pourront être incluses au système expert, et vice versa, si le S.E. pourra être facilement intégré à d'autres logiciels. On peut illustrer ces propos à partir des 2 applications que nous avons développées :

- le simulateur de projet fait appel à de nombreuses procédures : simulation des écoulements dans les réseaux, simulation de sondages pédologiques ... Dans la mesure où ces logiciels existent, il est impensable de les réécrire dans le langage du moteur pour assurer leur intégration;
- il a été envisagé, dès la conception du système expert en assainissement individuel, de pouvoir connecter ce logiciel au réseau MINITEL. Comment envisager de coupler un serveur avec un G.S.E. ou un langage totalement fermé ? Aussi, notre choix fut très orienté par cette contrainte.

• le prix : il peut varier, pour un G.S.E. de 1 à 400 (1.500 F pour VP-Expert à 650.000 F pour ADS) avec une médiane autour de 50.000 F [au printemps 1988]. A titre de comparaison, l'achat du compilateur de langage Turbo-Prolog (pour créer son propre moteur) revient à 1.500 F avec un environnement d'aide au développement (fenêtrage, trace ...). Ce critère "prix", très accessible même au profane, dépend des capacités financières de chaque utilisateur. Cependant, il doit quand même être ajusté au profit escompté de l'utilisation du système expert.

Un autre aspect que nous ne classons pas parmi les critères intègre les concepts de formation et de maintenance de G.S.E. ou de langages. De nombreux vendeurs de logiciels proposent des stages de formation pour l'apprentissage de leurs produits. Cette politique est quasiment commune à tous les bons commerciaux.

2.1.2 Générateurs ou langages ?

Compte tenu des propos précédents, c'est en effet la première question que l'on peut se poser : vaut-il mieux développer son système expert à partir d'un générateur ou bien mettre au point son moteur à partir d'un langage de l'I.A. ou autre. Répétons que cela dépend beaucoup du problème à traiter. Nous avons choisi de développer notre moteur à partir d'un langage : Prolog.

Ce choix a été dicté par des contraintes d'intégration et par la nécessité d'avoir une bonne maîtrise de la structure de la base de connaissances pour des problèmes de maintenance : le simulateur de projet doit être en perpétuelle évolution au fur et à mesure des sessions, l'assainissement individuel n'est pas une technique figée et les critères de choix sont divergents selon les experts.

Cette stratégie permet aussi de parfaitement caler le moteur à ses besoins, et fournit des possibilités de modifications pour l'adapter à un autre problème, voire de développer un autre moteur complètement différent.

Nous avons par ailleurs rencontré plusieurs personnes travaillant avec Nexpert, générateur de système expert relativement répandu. Deux phrases reviennent fréquemment dans leurs propos au sujet de Nexpert :

- long apprentissage : il faut plusieurs mois pour se familiariser avec les options, fonctions et autres possibilités avant de pouvoir se consacrer à son problème;
- avenir restreint : ces personnes n'ont pas envie de recommencer cette phase d'apprentissage avec un autre générateur, ce qui limite la réalisation de leurs recherches et applications à Nexpert pour plusieurs années.

Il n'est pas ici question de réfuter les qualités de Nexpert (ne l'ayant pas testé faute de temps) ou des G.S.E. en général. Ces logiciels nous semblent relever d'une mise en

œuvre plus lourde tant du point de vue financier que technique par rapport au développement à partir d'un langage. Cependant, par l'environnement et l'aide au développement que les G.S.E. peuvent offrir, ils apparaissent comme incontournables pour mettre en œuvre des applications complexes : celle qui se rapprocheraient le plus d'une expertise humaine, au sens d'H. DREYFUS.

2.1.3 caractère opérationnel et degré d'expertise d'un système expert

Dans quelles mesures un système expert reflète-t-il une réelle expertise et est-il opérationnel ?

C'est un débat que certains ramènent à l'éternelle opposition entre théoriciens et praticiens. C'est à notre avis un peu le cas.

le théoricien : il n'existe pas de système expert vrai, ou alors cette expertise n'est valable que pour un domaine d'application très réduit. Ces systèmes, de toutes façons, n'auront jamais les capacités d'un expert qui peut modifier en temps réel ses modes de raisonnement et intégrer des notions non répertoriées à l'origine. 99% des S.E. ne s'attaquent pas à des problèmes très complexes et automatisent des manuels de procédures ou des règlementations.

le praticien : appelons les systèmes experts des systèmes à base de connaissances ou un autre nom, cela n'a pas d'importance. Mais il ne faut pas jeter le bébé avec l'eau du bain ! La technologie des systèmes experts rend de grands services en assurant une ergonomie de développement et permet donc la mise au point de logiciels plus confortablement et plus rapidement. Elle amène un progrès, c'est ce qui est notable et à retenir.

Ces 2 monologues, complètement imaginés, sont représentatifs des 2 états d'esprit. Chacun des protagonistes a raison tant qu'il borne ses réflexions aux mêmes frontières que les champs d'applications dans lesquels il œuvre.

Loin de mépriser l'avis du "théoricien", nous nous sentons plus proche du praticien et c'est dans cette perspective que nos travaux ont été menés. *Les vrais systèmes experts posent aujourd'hui un problème de représentation de connaissances face à l'étendue des notions mises en avant par un expert dans son travail.* La nécessaire

réduction de cette étendue et le codage des connaissances pour les besoins de la machine entraînent une dégradation de la pensée experte. C'est pour l'instant un passage obligé. Cela n'en demeure pas moins un premier pas qu'il convient de franchir et de valoriser à bon escient.

En adoptant le statut du "praticien", le caractère opérationnel d'un système expert se rapproche de celui de nombreux logiciels :

• robustesse : qui concerne à la fois le moteur et la base de connaissance. On oublie en effet trop souvent que le moteur est un programme et qu'à ce titre, il peut connaître des problèmes de conception et d'utilisation qui se traduisent par des erreurs dans son fonctionnement. On parlera là d'erreurs de 'bas niveau'. La robustesse d'une base de connaissances s'évaluera par une série de tests visant à comparer les déductions du système expert avec celle d'un ou plusieurs experts; Les divergences qui apparaissent sont le fait d'erreurs de 'haut niveau' qui peuvent être :

- syntaxiques : des règles dont l'enchaînement est mal contrôlé et qui conduisent à des erreurs de déductions;
- analytiques : la non prise en compte de paramètres ou de notions qui entraînent des différences de raisonnement entre l'expert et la machine.

• maintenance : on ne parlera pas du moteur (langage ou G.S.E.),en face duquel l'utilisateur commun est généralement désarmé. Par contre, la base de connaissances sera plus ou moins maintenable selon l'habileté du concepteur et la nature de la connaissance mise en jeu. Plus cette nature sera digne du qualificatif d'experte, et plus cette maintenance s'avèrera difficile. En effet, la formulation analytique des raisonnements du spécialiste se heurtera de plus en plus aux approximations que ce dernier réalise de façon automatique et 'inconsciente'. Le nombre de règles de la base de connaissances risque de s'accroître au point de générer des incohérences.

• diffusion : à partir du moment où langages et G.S.E. sont accessibles sur des micro-ordinateurs du marché, le premier pas vers une bonne diffusion est fait. Reste le côté logiciel où les contextes sont variables : seuls les S.E. compilés permettent une diffusion sans contrainte (mis à part les systèmes d'exploitation

et les capacités mémoires des machines); beaucoup de G.S.E. nécessitent un "run-time" pour fonctionner, et les langages interprétés nécessitent l'achat de l'interpréteur et pénalisent en cela l'utilisateur.

2.2 QUELQUES SYSTEMES EXPERTS RECENTS DANS LES SCIENCES DE L'EAU EN FRANCE

Les systèmes experts que nous présentons dans les lignes qui suivent sont des exemples récents de travaux alliant systèmes experts et sciences de l'eau, présentés par ordre alphabétique [COUSIN, BOURBIGOT 1989], [DENOEUX 1988], [DETAY, POYET 1988], [SIBONY, BEUTLER, LEGRAND 1989]. Ils représentent une liste non exhaustive et illustrent les domaines qui peuvent être abordés. Notre objectif dans ce paragraphe est moins de décortiquer le fonctionnement des logiciels que de faire ressortir les moyens utilisés et les justifications avancées pour leur mise en œuvre.

Apogés

réalisation : CERGRENE (Centre d'Enseignement et de Recherche; Gestion des Ressources Naturelles et de l'Environnement).

matériel : Vax (compatible AT à terme), langage Prolog II (Arity Prolog à terme).
temps de développement : plusieurs années-hommes.

but : rationaliser la gestion des crédits d'entretien et de réparation d'un important réseau d'assainissement en fournissant une aide à la décision.

justification : les décisions dans ce domaine impliquent la prise en compte d'une grande quantité de connaissances (géologie, chimie, mécanique, génie civil) utiles au diagnostic de dégradation et au choix des méthodes de réhabilitation. Ces connaissances ne sont pas modélisables sous forme d'algorithme. L'étendue du domaine abordé implique une programmation à partir d'heuristiques permettant d'isoler rapidement des cas standards. De plus, l'aspect maintenance est important étant donnée l'évolution des connaissances, des pratiques et des objectifs du service d'assainissement.

remarque : ce projet est extrêmement ambitieux. Sa réalisation complète risque de ne voir le jour que très tardivement. Cependant, il a permis un début de formalisation de la démarche des décideurs et des experts qui s'est notamment traduite par une révision des circuits de circulation de l'information et l'amélioration de la connaissance du réseau. La stratégie de développement du système expert à partir d'un langage (et non d'un générateur de systèmes experts) a probablement permis de mieux caler les capacités du moteur et de la base de connaissances en fonction des contraintes du problème.

Hydroexpert
réalisation : M. DETAY, M. POYET.

matériel : compatible AT , langage Turbo-Prolog.
temps de développement : 2 à 3 années-hommes.

but : aide à la décision en matière d'implantation de forages (hydraulique villageoise) en Afrique de l'Ouest.

justification : la démarche de l'hydrogéologue qui veut implanter un forage avec succès doit prendre en compte un grand nombre d'informations hydrologiques, géologiques, météorologiques, sociales. Cette démarche est typiquement celle d'un expert qui doit relier et corréler ces informations afin d'en tirer des conclusions sur les capacités du forage. De plus, les auteurs ont voulu faire d'Hydroexpert un outil de formation.

remarque : la volonté des auteurs de faire de ce logiciel un outil pratique de terrain contraint son développement sur micro-ordinateur. Il devenait indispensable de réécrire un moteur étant donné les fonctions qui voulaient être implémentées pour l'aspect formation de l'outil (retour en arrière, interpréteur sémantique ...). Les auteurs ont largement utilisé les possibilités du Turbo-Prolog pour arriver à leurs fins, grâce à leur grande connaissance de ce langage.

Optimisation du fonctionnement d'une station d'épuration
réalisation : OTV (Omnium de Traitement et de Valorisation).

matériel : IBM PS2, générateur de systèmes experts NEXPERT.

temps de développement : environ 1 année-homme pour la maquette.

but : optimiser le fonctionnement de la station de MEYZIEU (69) à biofiltres (4 biocarbones). A terme, le système expert serait relié à un ensemble de capteurs permettant une gestion en temps réel.

justification : la conduite d'une station d'épuration requiert la prise en compte de beaucoup de données et informations : météorologiques, hydrologiques, taux et origine de la pollution, configuration de la station ... Le traitement de ces informations doit intégrer globalement tous ces paramètres rapidement; c'est ce qui est tenté dans le cadre de ce système expert.

remarque : les possibilités actuelles du système expert sont limitées et consistent en une aide à la décision du lavage des biofiltres. Il est cependant intéressant de constater la mise en parallèle de modules de calcul (prévision des débits entrants par exemple) avec un système expert. On notera aussi qu'un S.E. ne se justifie guère dans ce cas limité pour l'instant; il s'agit d'une anticipation des besoins à venir. De plus, les capteurs sensés informer ultérieurement le S.E. posent le problème de leur coût et de leur fiabilité étant donné le milieu très agressif des réseaux d'assainissement. Les conclusions et directives du S.E. seront directement dépendantes des mesures.

Pilote

réalisation : CERGRENE (Centre d'Enseignement et de Recherche; Gestion des Ressources Naturelles et de l'Environnement).

matériel : compatible AT , générateur de systèmes experts NEXPERT OBJECT.
temps de développement : 30 mois-hommes pour une maquette.

but : optimiser le fonctionnement d'un groupe de pompage dans la MARNE pour l'alimentation en eau potable.

justification : ce groupe de pompage est constitué de

i=2 pompes d'un débit unitaire de 1250 m^3/s
j=6 pompes " " " " 4166 m^3/s
k=2 pompes " " " " 11250 m^3/s.

La stratégie de pompage sur 24 heures par pas de temps de 1 heure peut se représenter par un ensemble de triplets (i,j,k) à chaque heure. Sur 24 heures, cela conduit à 63 puissance 24 configurations possibles. Il est donc impensable de simuler toutes les configurations pour choisir la meilleure solution afin :

- d'assurer une protection suffisante avec une bonne marge de sécurité (pression, niveau dans les réservoirs),
- de respecter les normes de qualité,
- de minimiser les coûts (électricité, arrêt et redémarrage des pompes).

En fait, le chef de poste chargé de choisir une configuration raisonne par des règles, issues de son expérience, qui lui permettent d'éliminer en bloc un grand nombre de solutions. C'est la mise au point de ces règles qui constitue la partie système expert de Pilote.

remarque : comme le système expert précédent, Pilote utilise en parallèle des modules de calcul (modèle statistique de consommation, modèle de simulation des écoulements dans les tuyaux ...) et un module expert. Ce module n'intervient que comme un maillon d'un ensemble qui constitue l'outil d'aide à la décision.

Suivi de la qualité du traitement d'eau potable
réalisation : CGE (Compagnie Générale des Eaux).

matériel : compatible AT, générateur de systèmes experts GURU.
temps de développement : 1 année-homme pour une maquette concernant une partie du traitement.

but : suivre en temps réel la qualité du traitement de l'eau potable et déclencher une alerte en cas d'anomalie; diagnostiquer alors la cause la plus probable de mauvais fonctionnement.

justification : la raison invoquée par les auteurs est la facilité de développement et de mise à jour du logiciel. En fait, il est probable que le diagnostic de la cause de mauvais fonctionnement du traitement fournisse un bon support à la mise en œuvre de la technique des systèmes experts.

remarque : la justification qu'invoquent les auteurs pourra surprendre. L'emploi de GURU peut paraître abusif dans ce cas, mais il peut se comprendre si l'un des auteurs connaissait déjà ce générateur. Dans le cas contraire, l'investissement semble surestimé par rapport au but car on peut penser qu'un bon résultat aurait pu être obtenu avec un langage classique (mis à part l'aspect facilité de développement et de mise à jour).

Enfin, nous mettons nos 2 applications à la suite de ces exemples, avec la même présentation, afin de mieux comparer ces différentes réalisations.

Promise

réalisation : EMSE (Ecole des Mines de Saint - Etienne)

matériel : compatible AT, langage Turbo-Prolog
temps de développement : 1 année-homme.

but : simuler la réalisation d'un projet dans le domaine de l'eau (assainissement, adduction en eau potable, irrigation) dans un but pédagogique.

justification : un projet, du domaine de l'eau ou autre, constitue un ensemble complexe fait de décisions, d'évènements, de disponibilité d'outils. L'imbrication de ces éléments et leurs inter-dépendances rend opportune l'utilisation de la technologie des systèmes experts. De plus, en tirant profit des simulations qui seront effectuées, il s'agit d'offrir la possibilité d'une mise à jour aisée.

remarque : le système expert n'est que la partie "intelligente" du simulateur permettant par exemple de rendre pertinents les évènements en fonction du passé du projet. Il permet aussi l'appel de procédures de calcul ou autres, l'ensemble constituant le simulateur. C'est un nouvel exemple de l'intégration d'un système expert dans un outil informatique.

Moïse

réalisation : EMSE (Ecole des Mines de Saint - Etienne)

matériel : compatible AT, langage Turbo-Prolog

temps de développement : 2 à 3 années-hommes.

but : dans le cadre de l'assainissement en milieu rural, Moïse aide au choix d'un assainissement individuel.

justification : l'assainissement individuel rassemble un certain nombre de techniques, parmi lesquelles le choix s'opère à partir de valeurs de paramètres ou de notions qualitatives. La gestion des critères de choix et des filières d'assainissement conduit à une difficulté d'ordre combinatoire. De plus, il n'existe pas d'unanimité parmi les spécialistes conduisant à des règles admises par tous. *Moïse doit donc permettre à chaque spécialiste de créer sa base de connaissances.*

remarque : au niveau matériel, les choix ont été fait en fonction des objectifs définis ayant trait à la diffusion du logiciel : Turbo-Prolog a été choisi pour ses possibilités d'interfaçage avec des logiciels existant (serveurs pour Minitel), et le micro-ordinateur pour une diffusion éventuelle vers des bureaux d'études généralement équipés avec ce type de matériel.

L'analyse de ces exemples permet de tirer les enseignements suivants :

matériel : une forte tendance à l'utilisation de micro-ordinateurs qui peut tenir à plusieurs faits :

- la puissance qu'atteint ce type de machines à l'heure actuelle permet un développement dans de bonnes conditions pour un coût raisonnable;
- les laboratoires œuvrant dans les sciences de l'eau n'ont pas pour vocation de détenir un matériel informatique lourd et dédié.

Beaucoup d'équipes se tournent vers le développement de leur propre moteur à partir de Prolog essentiellement. Après être longtemps apparus comme un problème de spécialistes informatiques, les langages de 4ème génération entrent désormais dans la panoplie des outils informatiques à la disposition des hydrologues et des scientifiques de façon plus générale.

temps de développement : il est relativement long pour chacune des applications. Il semble que ce temps inclue en fait une bonne part de formation aux langages ou aux

générateurs de systèmes experts, quand ce type d'application constitue une première dans les exemples donnés. A l'avenir, ce temps de développement devrait être dégressif car plutôt consacré à la mise en forme des connaissances. Sinon, l'investissement en temps peut aussi s'expliquer par la relative difficulté et nouveauté des problèmes abordés dans la mesure ou des algorithmes ne peuvent pas être dégagés.

justification : elle est en majorité issue de la difficulté à résoudre le problème par une technique classique de programmation (algorithmique et structurée). Mais l'aspect maintenance est souvent évoqué pour le choix de la technique des systèmes experts. C'est en effet un problème souvent mal résolu dans des logiciels classiques et il semble que les S.E. apportent une meilleure solution.

Enfin, preuve de la "vulgarisation" des systèmes experts, ceux-ci interviennent de plus en plus comme *procédure* dans un outil abordant diverses facettes d'un problème. Cette intégration vient comme une pierre supplémentaire pour bâtir une structure mieux apte à fournir une aide à la décision ou à un diagnostic dans la majorité des cas.

3. CONCLUSION DE L'ANALYSE BIBLIOGRAPHIQUE

L'hydrologie n'est pas une science fondamentale, mais une science pluridisciplinaire. Elle fait appel à des notions de mathématique, de physique, de chimie, de géologie, de météorologie ... et ... d'informatique. Ce dernier élément est devenu essentiel : quel hydrologue ne s'est pas un jour tourné vers l'informatique pour l'aider à résoudre son problème ?

De par ses préoccupations "tous azimuts", l'hydrologue est un "fouineur"; il fait partie de ceux qui ne vivent pas sur des acquis, mais qui doivent renouveler et améliorer sans cesse leurs techniques. C'est par cet aspect que l'intelligence artificielle s'ouvre un droit d'existence dans les sciences de l'eau.

Que retenir de l'I.A. ?

A l'heure actuelle, les systèmes experts émergent et, grâce à leur adaptabilité sur de nombreuses machines accessibles, ils sont une source d'exploitation pour l'hydrologue. Les S.E. s'appuient sur une représentation. à partir d'une approche

analytique, de la connaissance et ouvrent la voie vers la programmation de problèmes non algorithmiques. Ceci est la première propriété des S.E. La deuxième est que le formalisme de représentation de la connaissance ne modélise pas seulement de la connaissance, mais aussi la manière de l'utiliser : c'est un mélange d'information et de mode d'utilisation. Contenu dans la base de connaissances, ce formalisme est plus accessible que noyé dans un programme classique : la maintenance, problème souvent mal résolu, est plus facile.

Ce sont ces deux propriétés, dont l'hydrologue peut tirer avantage, qui vont orienter une partie des applications dans les sciences de l'eau. Les moyens à utiliser peuvent être variables. Nous avons choisi de développer sur micro-ordinateur pour des raisons de coût et de diffusion et de réécrire des moteurs en Prolog afin d'aboutir à une meilleure maîtrise des logiciels tout en leur assurant une adaptation aux problèmes posés. Dans les deux cas, le coût d'investissement est minimisé et il ne semble pas que la formation à un langage comme Prolog soit plus longue que la formation à un générateur de systèmes experts. Nous préconisons donc ces choix. Ils correspondent à un "profil bas" qui nous paraît convenir à l'hydrologue qui doit pouvoir réutiliser facilement ses acquis (matériels et intellectuels) dans des applications différentes.

Les chapitres qui suivent décrivent deux applications réalisées à partir des principes évoqués. Elle montrent que la mise en œuvre des structures légères "Micro-ordinateur + Prolog" permet d'obtenir des résultats intéressants.

Un simulateur de projet "intelligent" : Promise

RESUME DE L'APPLICATION

Depuis plusieurs années, sous l'impulsion de Ph. Davoine, enseignant des sciences de l'eau, l'Ecole des Mines de St-Etienne développe un programme d'E.A.O. (Enseignement Assisté par Ordinateur).

Ce programme a fait l'objet de travaux de thèse de D. Graillot [GRAILLOT 83], [GRAILLOT 86] et consiste en un outil, baptisé 'MISE' (Modèle Intégré en Stratégie de l'Eau), de simulation de projets.

Les sessions de formations ('sessions Mise') durent environ 1 semaine pendant laquelle les apprenants doivent réaliser un projet du domaine de l'eau. L'ordinateur fournit les outils de calcul et génère des incidents qui vont perturber les prévisions émises par les apprenants.

Deux types de projets ont été développés jusqu'à présent :

- un projet d'adduction en eau potable,
- un projet d'irrigation.

L'analyse des sessions Mise auxquelles nous avons participé nous a permis de concevoir un outil de simulation beaucoup plus évolué, permettant une simulation dite "intelligente".

Ce simulateur reprend la technologie des systèmes à base de connaissances ('systèmes experts').

Voici, en résumé, les particularités de ce simulateur baptisé 'Promise' :

- prise de décision selon le formalisme : action - objet - précisions; (exemple : contacter - organisme régional - la Direction Départementale de l'Agriculture).

- simulation intelligente par la prise en compte de l'historique du projet que crée l'apprenant. Ainsi Promise génère des incidents "pertinents", puisque corrélés à l'historique, et conditionne l'emploi des outils de calculs.

- intégration poussée du système : l'appel de toute procédure (sous-système expert ou programme de calcul) est automatisé et piloté par Promise. Ainsi,

l'interlocuteur privilégié de l'apprenant est le simulateur. La base de faits permet le stockage de l'historique et la circulation des informations (données et/ou résultats) entre les logiciels.

- mémorisation et enrichissement (capitalisation) des sessions : la base de connaissances (ou base de règles) se présente sous forme d'un fichier aisément transformable. Ainsi, à volonté, le concepteur peut augmenter les scénarios possibles du projet.

- possibilité d'intervenir 'en ligne' pour l'apprenant à tout moment pour :
  * consulter des informations sur le projet,
  * laisser des remarques,
  * contrôler le bilan de son projet,
  * visualiser l'historique du projet.

- le simulateur aborde l'aspect "extraction de connaissances" sous 3 axes :
  * l'enregistrement des remarques de l'utilisateur,
  * le simulateur peut poser des questions,
  * le simulateur peut provoquer un contexte ou le simuler.

Une base de connaissances, développée à partir d'un projet d'assainissement en milieu rural (projet d'assainissement de la Varèze; bourgs de Vernioz et de St-Alban-de-Varèze dans l'Isère), et les outils qui s'y rattachent, ont permis le test du simulateur. Orientée vers un objectif pédagogique, cette base de connaissances a montré l'adaptabilité (créée en quelques jours pour la phase pré-étude : contacts et recueil de données), l'évolutivité (les règles peuvent être aisément complétées et/ou modifiées) et la fiabilité du simulateur (une simulation a été réalisée).

Plus tard, des projets A.E.P. (Adduction en Eau Potable) et irrigation pourront être incorporés à la panoplie de Promise; il suffit pour cela de changer la base de connaissances, les données et éventuellement les outils, le "moteur" du logiciel pouvant intégrer des projets d'un autre domaine que l'assainissement et restant inchangé.

Pour assurer sa portabilité, le simulateur est développé sur micro-ordinateur type IBM PC.

PROLOGUE

Supposons que l'on veuille accumuler, sans se préoccuper de la forme, les paramètres qui interviennent dans un projet du domaine de l'eau.

Le travail est vaste, tant qualitativement (données économiques, politiques, scientifiques ...) que quantitativement (séries de mesures météorologiques, agro-économiques ...). On aurait même tendance à dire que cela n'est pas faisable : il faut simplifier.

On sait assez bien le faire avec des valeurs numériques par l'intermédiaire de l'analyse de données qui permet de représenter un échantillon de valeurs par quelques paramètres (statistique descriptive), ou de comparer 2 échantillons (inférence statistique). Cependant, dans le cadre d'un projet et des actions stratégiques que l'on va initier, cela ne suffit pas toujours. Certes, on va comparer éventuellement 2 valeurs, mais notre jugement sera pondéré par de multiples facteurs car nous sommes toujours influencés, concernés ou tributaires d'un contexte.

Que dire alors quand on aborde des notions non numérisables ? Chacun a ses propres références qui lui permettent de graduer si nécessaire cette notion en classes. Vouloir justifier de cette graduation n'est pas toujours une entreprise aisée car, plus que pour des nombres, les relations de cause à effet sont plus difficilement discernables.

Bref, à partir d'un certain niveau que l'on qualifiera de stratégique, il devient délicat, voire impossible d'identifier clairement les paramètres d'un projet, et donc de le piloter efficacement.

Là où des méthodes de type recherche opérationnelle permettent d'optimiser un ou plusieurs critères correspondant à une phase tactique ou technique du projet, ces méthodes vont se heurter au manque de formalisation des variables régissant un niveau plus élevé de décision, de par la variété et la complexité des logiques d'acteurs appelés à jouer un rôle dans le processus par lequel s'élabore une décision stratégique.

Comment alors modéliser sur une machine ces niveaux de décisions pour lesquels on maîtrise mal les variables?

Précisons ici l'utilité d'une telle modélisation : elle se place dans un contexte d'aide à la décision et au déroulement d'un projet. Ces notions sont différentes de celle de

recherche d'une solution optimale pour laquelle il faudrait pouvoir modéliser le problème de telle sorte que :

"• chaque solution envisagée soit exclusive de toutes les autres;

• l'ensemble des solutions considérées soit fixé une fois pour toutes;

• les solutions puissent être ordonnées de façon incontestable de la plus mauvaise à la meilleure "[ROY, BOUYSSOU 1988].

Ici, le but de la modélisation est de gérer les connexions entre les différentes phases d'un projet et de représenter le savoir-faire d'un spécialiste à travers une formalisation de son expérience. Le résultat recherché est à terme le développement d'un outil de pilotage de projet.

C'est bien un problème que se propose d'aborder l'Intelligence Artificielle avec le traitement des connaissances. Encore faut-il :

• avoir des données pour nourrir ces systèmes à bases de connaissances, c'est-à-dire que quelqu'un (un expert) puisse décrire, volontairement ou non, comment il raisonne;

• construire une structure informatique d'accueil engrangeant ces connaissances.

Pour y répondre, D. GRAILLOT proposait d'utiliser le système d'enseignement assisté par ordinateur Mise, simulateur de projet, comme "piège à connaissances" et les systèmes experts comme "support informatique de la connaissance" [GRAILLOT 1986].

Cependant, si le système Mise est effectivement un bon simulateur de projet, il apparaît qu'il est "inerte" face aux bénéfices qu'il est sensé apporter dans la maîtrise stratégique et technique d'un projet. On montrera par la suite que Mise, de par sa structure informatique, ne peut guère évoluer qualitativement et constitue un facteur limitant face à une fonction "piège à connaissances". Nous essaierons donc de refondre complètement le simulateur, étape nécessaire avant d'entrevoir les possibilités d'application d'un simulateur de projet en tant que station de travail pour le responsable d'un projet d'aménagement.

Quant au "support de connaissances", le problème reste entier face aux difficultés de formalisation d'expérience (au sens du savoir-faire) et face à l'évolution des règles de conduite d'un projet.

C'est très globalement le problème que nous abordons dans ce mémoire, et plus particulièrement dans cette deuxième partie : l'écriture d'un logiciel (en Prolog) qui essaie de prendre en compte des décisions et évènements d'ordre stratégique, tout en étant capable d'activer des outils à vocation plutôt tactique, et ce à travers la simulation de projet. C'est faire "d'une pierre deux coups" puisqu'à ce moment là, la simulation peut devenir "intelligente", et constituer un piège à connaissances bien plus subtil.

Ce logiciel, Promise, n'a pas pour ambition d'être une panacée, mais plutôt une exploration de cette voie. Les difficultés sont énormes et nous ne prétendons pas en avoir levé plus d'une infime partie.

Promise n'est qu'une tentative pour modéliser le déroulement d'un projet en gérant les liens et les interférences entre ses différentes composantes.

La mise en place informatique des ces liens oblige le concepteur à une réflexion poussée, une mise à plat de ses connaissances et de son savoir-faire en matière de conduite de projet. Le schéma fonctionnel obtenu (adapté à la simulation), allié au piège à connaissances que peut être Promise, constitue un autre pas vers une meilleure maîtrise des facteurs d'un projet.

1. PROMISE ET LA SIMULATION SUR ORDINATEUR

1.1 EVOLUTION DE LA SIMULATION SUR ORDINATEUR

Afin de bien situer la place de la simulation de projets parmi les applications des sciences de l'eau liées à l'ordinateur, nous distinguons quatre étapes illustrant l'évolution dans la complexité des phénomènes modélisés sur une machine. Il s'agit bien d'une illustration dont le but est de bien saisir le positionnement de la simulation de projets parmi l'ensemble des applications.

1.1.1 LA MODELISATION MATHEMATIQUE

La première des propriétés d'un ordinateur est sa vitesse de calcul, de traitement de l'information. Cette propriété peut être utilisée pour résoudre une équation qui représente un phénomène physique; Il s'agira, par exemple, de la modélisation d'une

onde de crue sur une rivière, ou de l'étude de l'écoulement des eaux dans un réseau d'alimentation en eau potable (A.E.P.). On peut considérer qu'il s'agit là de la première étape qui fut franchie dans la simulation assistée par ordinateur.

Dans ce cas, l'objectif qui peut être poursuivi est la prévision des conséquences d'un phénomène naturel (une crue), ou l'aide au calcul pour la mise en œuvre d'une intervention humaine (un réseau A.E.P.).

1.1.2 L'AIDE A LA DECISION

Une deuxième étape correspond à l'utilisation répétée d'un même outil informatique de modélisation dans un but d'aide à la décision.

Dans le cas de la modélisation d'une onde de crue, on pourra adapter le logiciel de façon à tester différents aménagements possibles de lutte contre les crues, à étudier l'interférence entre ces aménagements afin de choisir le plus efficace. Quant à l'étude du réseau A.E.P., on pourra rechercher quelles améliorations on peut mettre en œuvre, ou comment minimiser le coût de raccordement d'une nouvelle zone d'aménagement.

Dans tous les cas, cette aide à la décision s'effectue au mieux à partir d'une analyse monocritère ou multicritère qui évalue, à l'aide de fonctions, les conséquences des différentes actions possibles. Ces critères, obligatoirement numériques, ne s'intéressent souvent qu'à un aspect ou une phase du projet étudié et souffrent, en général, d'un manque d'intégration au niveau global du projet (optimisation partielle).

1.1.3 LA SIMULATION DE PROJET

C'est l'approche systémique d'un projet qui va constituer la troisième étape. C'est-à-dire que l'on va simuler les différentes composantes d'un projet par l'intermédiaire d'un ensemble d'outils informatiques. La grande difficulté dans cette entreprise est la représentation de l'interférence de phénomènes très différents, sociaux ou physiques, aléatoires ou déterminés par le passé et le contexte du projet.

En reprenant les deux outils précédents (prévision des crues et étude de réseau A.E.P.), peut-être se rendra-t-on compte que la protection de la zone à équiper pour l'A.E.P. est d'un coût trop important; inutile donc de chercher à optimiser le réseau. Ou bien la zone de pompage pour l'eau potable est en zone inondable auquel cas il faudra prévoir des surcoûts d'équipements des puits et/ou une protection rapprochée contre les crues...

1.1.4 LE PILOTAGE DE PROJET

Cette quatrième étape est en fait un sujet d'avenir. Elle incorpore la simulation de projets en tant qu'outil pour intégrer tout l'environnement d'un projet. Son objectif est de proposer un guide dans le déroulement du projet, les outils disponibles en fonction des données collectées, les solutions possibles face à un évènement non prévu... Ceci est possible dans la mesure où la cohérence entre les différents éléments du projet est respectée et où, à des niveaux partiels, un ou plusieurs critères sont optimisés : c'est le but ultime d'une station de travail. Cette station de travail est nécessairement un outil évolutif, capable d'incorporer au fur et à mesure de nouveaux éléments et de nouvelles connaissances.

1.1.5 ANALYSE DE CETTE EVOLUTION

Cette classification nécessairement schématique, que nous venons d'exposer, et que nous résumons dans le tableau 2-1, traduit une évolution technique et intellectuelle. En effet, plus on veut se rapprocher de la réalité de la conduite d'un projet et de son déroulement, plus il faut prendre en compte d'éléments, et plus les interférences augmentent. On atteint rapidement une explosion combinatoire des stratégies et des évènements possibles.

Il est également intéressant de noter que la nature des connaissances le long de cette évolution est de moins en moins mathématique et algorithmique. Bien entendu, il n'y a pas abandon des procédures de calculs, mais celles ci sont de plus en plus reléguées au rang de simples outils et activées quand le besoin s'en manifeste.

1.2 POSITIONNEMENT DU LOGICIEL PROMISE

Dans cette évolution, Promise est un logiciel qui se positionne au niveau de la troisième étape : c'est un simulateur de projets. Ce logiciel sera donc capable de représenter quelques évolutions plausibles d'un projet tout en contrôlant l'agencement des différentes étapes. Dans la mesure où Promise ne prend pas en compte toutes les évolutions possibles, tous les éléments que l'on peut recenser, tous les évènements qui peuvent survenir et n'intègre pas d'outils d'optimisation tenant compte d'un environnement global, ce n'est pas un outil d'optimisation de projet du type station de

ETAPE	CARACTERISTIQUE	EXEMPLES
modélisation mathématique	résolution d'une équation	propagation d'une crue; modèle "boîte noire"
aide à la décision	prise en compte d'un environnement restreint	connaissant la forme des berges et des zones à protéger, où est il le plus judicieux d'installer des digues pour un moindre coût ?
simulation de projet	représentation d'un ou plusieurs scénarios d'un projet	si on envisage un barrage, comment la construction de celui-ci s'intégre-ra-t-elle dans l'économie régionale ? Quelle pourra être la réaction des agriculteurs face à l'engloutissement des terres ?
pilotage de projet	prise en compte d'un environnement global dans le déroule-ment d'un projet	Face à la montée en puissance d'un parti écologique, vaut-il mieux un barrage, des digues ou tout laisser tomber ?

tableau 2-1 : Evolution de la complexité de la simulation sur un ordinateur

travail, même rudimentaire. On verra qu'il peut en constituer simplement une ébauche logique et informatique.

A quoi sert alors la simulation de projet ?

En effet, pourquoi faire un énorme programme, qui a besoin d'un grand nombre de procédures et de données, si c'est pour étudier toujours le ou les mêmes scénarios d'évolution ?

Il y a deux domaines où cette démarche même répétitive peut-être rentable : le premier est *la formation* , depuis l'enseignement des jeunes ingénieurs ou techniciens jusqu'aux stages de personnels qualifiés dans le cadre des formations continues. Tout d'abord, l'apprenant (élève, stagiaire) échappe à l'aspect répétitif du scénario du

simulateur s'il n'en a qu'un, à moins d'avoir à faire à un redoublant dans le cas d'un élève ou d'un récidiviste dans le cas d'un stagiaire. Mais, ce qui est plus important, le simulateur leur permet de se mettre en situation de projet par la prise en compte d'éléments divers et de leurs interactions complexes. Il mobilise chez l'apprenant ses qualités d'imagination, son intuition, sa créativité, des éléments épars de connaissances... C'est une démarche dynamique comparée à un système classique d'enseignement (cours magistraux, conférences...).

On verra dans le fonctionnement de Promise comment ce rendu de la réalité d'un projet peut être obtenu.

Le milieu industriel n'a pas tardé à se rendre compte des avantages de tels systèmes pour la formation de leurs personnels :

Loin du domaine de l'eau, FUJITSU, entreprise informatique japonaise, emploie 20 personnes pour concevoir des simulateurs de projets. Une des conclusions du travail réalisé par les logiciels sous forme de jeu de rôles vient des participants eux-mêmes, dont 75% pensent que le jeu remplace efficacement 6 mois d'apprentissage sur le terrain et cela en trois jours [le Monde Informatique du 8 décembre 1988]. Même en pondérant ces chiffres (les participants étant eux-mêmes employés de FUJITSU ou ses filiales), le gain de temps - donc d'argent - réalisé reste appréciable.

Trois aspects du simulateur sont évoqués dans sa mise en œuvre pour un enseignement assisté par ordinateur :

- les "travaux pratiques", car les apprenants ont la possibilité de comprendre l'usage des formules par une expérimentation fictive. En effet, les simulations sont conduites à l'initiative de l'apprenant, elles sont menées à divers niveaux d'abstraction (du purement qualitatif au complètement abstractif), elles incluent tout un spectre de cas particuliers, enfin, elles peuvent être conçues pour obliger l'apprenant à confronter ses "théories naïves" (celles qu'il élabore lui-même) aux théories scientifiques [HEBENSTREIT 1988].

- le don d'ubiquité du logiciel, qui peut être opérationnel simultanément sur plusieurs machines. Chaque apprenant pourra s'attarder, indépendamment de l'état d'avancement de ses collègues, sur les points qu'il désire pour surmonter

une difficulté théorique ou réfléchir sur certains aspects difficiles du projet : ainsi s'accroît la flexibilité de l'enseignement.

- la fidélité du logiciel qui est un gage pour l'enseignant, seul maître du contenu du simulateur. Ce dernier réagira toujours comme l'enseignant l'a programmé : il n'y a pas de déformation du message enseignant.

Le deuxième domaine où la simulation de projet et son aspect répétitif peut apporter un plus consiste en *un ajustement itératif des connaissances* et de leur mise en forme. En effet, un simulateur nécessite une mise au point, une mise à jour régulières de ce que nous appelerons plus loin "des règles de simulation". Celles-ci gèrent les scénarios d'évolution du projet, les procédures de calcul numérique ou autres, les aléas qui peuvent se produire et modélisent l'interférence entre les différentes connaissances mises en jeu. Elles sont le fruit de l'expérience du concepteur de la base de règles et à ce titre, leur formalisation est une œuvre de longue haleine.

On remarquera la complémentarité entre les deux domaines où la simulation s'applique, puisque sa mise en œuvre dans le cadre de l'enseignement permet un ajustement itératif des règles, et en retour, cet ajustement rend mieux compte des différents aspects du projet et améliore par là même la qualité de l'enseignement.

Promise est un simulateur conçu pour des projets du domaine de l'eau. Cependant, sa structure lui permet d'être extrapolable vers d'autres champs d'applications.

Avant d'aborder le fonctionnement de Promise, nous allons d'abord analyser le système Mise qui peut être considéré comme l'ancêtre de Promise dans la mesure où il fut une première dans la mise en œuvre d'un simulateur de projets dans le domaine de l'eau.

2. MISE : MODELE INTEGRE EN STRATEGIE DE L'EAU

2.1 HISTORIQUE

C'est depuis 1980, sous l'impulsion de Ph. DAVOINE, chargé de l'enseignement des sciences de l'eau à l'Ecole Nationale Supérieure des Mines de Saint-ETIENNE, qu'a été développé un programme d'enseignement assisté par ordinateur (E.A.O.).

En effet, la vocation de l'école des Mines est la formation d'ingénieurs généralistes. Aussi, le volume d'heures consacré à l'enseignement des sciences de l'eau s'avère faible (150 heures pour l'année scolaire 1988-1989) devant l'ampleur de la tâche. Face à ce dilemme, Ph. DAVOINE imagina de traiter cet enseignement sous forme d'une simulation de projet du domaine de l'eau. Ce choix implique que l'apprenant n'aborde pas toutes les facettes qui constituent les sciences de l'eau, mais se contente de certains aspects.

Le projet est pris sous l'angle "système complexe", c'est-à-dire un système dont on ne domine pas tous les paramètres, où ces derniers sont nombreux, et dans lequel une approche à la fois globale et planifiée est indispensable.

Les travaux réalisés ont conduit au Mise avec une version Adduction en Eau Potable (A.E.P.) et une version irrigation. Ces deux versions s'appuient sur des projets de référence réels :

- alimentation en eau potable de 3 communes rurales en cours de développement, pour l'horizon 2000, du nord du département de la LOIRE : Saint-JUST / Saint-RAMBERT sur LOIRE, BONSON, Saint-CYPRIEN.
- projet d'irrigation du périmètre de PAJAY au sein de la plaine de BIEVRE-VALLOIRE, à cheval sur les départements de l'ISERE et de la DROME [DAVOINE, GRAILLOT 1986].

Par ailleurs, dans sa thèse, D. GRAILLOT avait jeté les bases d'un Mise "assainissement" dont l'organisation s'inspirait directement des 2 expériences précédentes [GRAILLOT 1986].

Une longue période de tests à l'Ecole des Mines de Saint-ETIENNE a prouvé que Mise était un système viable et opérationnel, il a été exporté et est utilisé aujourd'hui par divers organismes de formation ($2^{ème}$ cycle, $3^{ème}$ cycle ou formation permanente) Français ou étrangers :

- l'université de PARIS VI;
- l'Ecole des Mines de PARIS;
- l'université du MINNESOTA (USA) en version Anglaise;
- la Compagnie Générale des Eaux.

2.2 ORGANISATION D'UNE SESSION MISE

Nous prenons comme exemple le Mise A.E.P., sachant que l'organisation du Mise Irrigation est semblable.

Une session dure idéalement 5 jours [DAVOINE, GRAILLOT 1984] :

1er et 2ème jour	acquisition des données Elaboration d'une stratégie (visite de terrain si possible)
3ème jour	prospection des ressources Exploitation
4ème jour	Réseau de distribution
5ème jour	Gestion de la société de distribution et évaluation du projet devant un jury

Les apprenants sont répartis en équipes. Quand cela est possible, des décideurs, des élus locaux, des responsables administratifs, des experts sont invités et intégrés aux équipes. Ceux-ci participent alors activement et de façon naturelle - inconsciemment - à la formation des apprenants par l'expérience qu'ils peuvent faire partager. Inversement, ces personnels qualifiés dans leur domaine peuvent découvrir des aspects de la conduite de projets qu'ils n'avaient jamais abordés jusque là.

En effet, le projet est mené depuis la recherche de données et d'informations (les participants ne savent rien du projet en arrivant), jusqu'à l'élaboration d'une société de distribution d'eau en passant par de nombreux aspects techniques (mise en place de forages, optimisation du réseau de distribution), l'aspect financier (émission de prêts pour financer les travaux et le démarrage de la société), sans oublier les aspects sociaux, administratifs et politiques (élections proches, mouvements de grève ...).

Dans cet ensemble, l'enseignant - dit coordinateur ou superviseur - joue un rôle primordial du fait qu'il représente différents personnages :

- animateur : il sera amené à jouer le rôle du maire grincheux qui refuse obstinément une possibilité, ou qui aimerait bien que les forages soient implantés à tel endroit.

• théoricien : c'est son rôle originel, en explicitant les équations qui interviennent dans un modèle.

• conseiller : il pourra être un bon conseiller en remettant sur une bonne voie une équipe perdue, ou un mauvais conseiller en piégeant un peu une équipe qui avance vite.

• vérificateur : en contrôlant l'emploi des logiciels et les résultats qu'obtiennent les différentes équipes.

Le tableau 2-2 reproduit le partage du temps, en moyenne, du coordinateur durant une session.

ROLE	EXEMPLES D'ACTIONS	TEMPS EN %
animateur	coordinateur simulation de personnages palliatif aux lacunes du MISE	20
théoricien	explication théorique des méthodes employées, liens entre les différents outils	60
conseiller	simulation d'un expert intervenant à la demande d'une équipe (rattachée à un coût)	10
vérificateur	jury, contrôle en cours de session	10

tableau 2-2 : répartition en temps des activités de l'enseignant durant une session MISE

2.3 ELEMENTS DU SYSTEME MISE [DAVOINE 1988]

2.3.1 LE SYSTEME DECISIONNEL

C'est un inventaire hiérarchisé suggérant les moyens, méthodes et techniques utilisés à l'heure actuelle pour réaliser des projets d'alimentation en eau potable. Cet inventaire se présente sous forme d'une liste (tableau 2-3) dont chaque élément se repère par un numéro qui se décompose en étape et sous-étape.

2.3.2 LA BANQUE DE DONNEES

Données à accès libre : elles comportent des informations de base : cartes topographiques, plan d'occupation des sols ... et certaines données spécifiques au projet considéré (mode de financement, abaque de coûts ...).

Données à accès contrôlé : elles sont associées à un coût uniquement temporel (études bibliographiques, enquêtes auprès des administrations ...) ou bien à un coût temporel et financier (prospection géophysique, construction du réseau ...). Tous ces coûts sont imputables au budget temporel et financier initial, imparti à l'utilisateur en début de session.

2.3.3 EXEMPLES D'OUTILS INFORMATIQUES A DISPOSITION

Simulation de la prospection des ressources et de la réalisation des forages d'exploitation

Divers fichiers et logiciels permettent aux utilisateurs d'évaluer les ressources. Ils simulent :

- l'exécution de sondages (reconnaissance ou exploitation) aux nœuds du maillage de la zone préalablement discrétisée.
- la mise en œuvre de campagnes géophysiques.
- la saisie d'informations diverses (piézométrie, pluviométrie, débits des eaux de surface, qualité des eaux pompées ...).

Un modèle hydrodynamique de nappe aquifère indique les réactions de la nappe en fonction des pompages que l'utilisateur programme, en débit et en durée.

1... PRISE DE CONTACT
- 101 élus locaux
- 102 administrations
- 103 distributeurs d'eau

2...RECENSEMENT DES DONNEES EXISTANTES
- 201 pompages
- 202 piézométrie
- 203 études générales
- 204 facteurs de risques

3...COMPLEMENTS D'INFORMATION
- 301 sourcier
- 302 consultations : experts, bureau d'études
- 303 pluviométrie
- 304 eaux de surface, débits
- 305 géophysique
- 306 relevés piézométriques
- 307 carottages et diagraphies
- 308 et 309 perméabilités et débits d'échanges (modèle hydrodynamique)
- 310 qualité des eaux de surface

4...POMPAGES D'ESSAIS
- 401 essais de débits ponctuels
- 402 analyses de qualité
- 403 simulation hydrodynamique complète de la nappe

5...FORAGES D'EXPLOITATION
- 501 aménagements électriques périmètres de protection
- 502 dimensionnement
- 503 exécution technique
- 504 traitements éventuels

6...IMPLANTATION DU RESEAU
- 601 localisation des usagers
- 602 schéma de distribution
- 603 réalisation
- 604 optimisation technique
- 605 optimisation économique

7...EXPLOITATION DU RESEAU
- 701 entretien et surveillance
- 702 extension du réseau
- 703 modernisation du réseau

8...GESTION DE LA DISTRIBUTION
- 801 type de gestion, financement des investissements
- 802 investissements initiaux et en cours d'exploitation
- 803 entretien et surveillance (aspects économiques)
- 804 provisions
- 805 salaires et charges sociales
- 806 charges financières d'investissements
- 807 amortissement des investissements
- 808 impots, taxes et redevances
- 809 frais généraux

tableau 2-3 : éléments du système décisionnel du Mise A.E.P.

Simulation de la distribution des ressources en eau

Un modèle mathématique de réseau simulant les écoulements dans les canalisations à installer permet de dimensionner de façon optimale les structures de distribution.

Gestion en continu du projet

Un logiciel comptable, destiné à établir un compte général d'exploitation, ainsi qu'un historique comptable, enregistre les coûts et les temps de réalisation du projet et permet une comparaison entre les programmes prévisionnels et réalisés.

2.3.4 RYTHME D'UNE SESSION MISE

A l'intérieur du cycle de 5 jours depuis la découverte du projet à réaliser jusqu'à l'exposé final face à un jury, chaque étape et sous-étape, agencée dans un ordre décidé par chaque équipe, est contenue dans un cycle :

choix d'une étape et sous-étape	---->	activation d'un logiciel *[éventuel]*	---->	réalisation et perturbation

Le passage éventuel par l'activation d'un logiciel correspond à certaines étapes (par exemple n° 401 : essais de débits ponctuels) pour lesquelles l'apprenant va utiliser un logiciel de calcul dont le nom lui est communiqué par le coordinateur (simulation hydrodynamique de la nappe pour l'étape 401). Pour l'étape n°101 : prise des contacts avec les élus locaux, il n'y a pas besoin d'activer la moindre procédure et on passera directement à la fin du cycle.

Dans tous les cas, la phase réalisation/perturbation va provoquer des évènements qui seront avantageux, neutres ou désavantageux vis-à-vis de l'étape programmée. Cette phase, très importante, pourra amener de nouvelles informations ou constater un échec.

exemple : étape 101 prise des contacts avec les élus locaux

succès de la réalisation : intéressés par votre campagne de recherche, les élus locaux vous fournissent des informations concernant les pompages actuels. (le

coordinateur, au vu de ce message, fourni un document ou le nom d'un fichier à consulter).

échec de la réalisation : l'administration locale vous accueille fraîchement et ne voit aucune raison de collaborer avec des intérêts privés.

Parfois, si l'objectif programmé s'y prête, on aura une fluctuation des résultats par rapport à ce qui pouvait être espéré.

exemple : étape 503 forages d'exploitation, exécution technique

évènement : vous avez crépiné plus de 30% de la zone saturée. Les conditions de nappe libre ne vous y obligeaient pas. Cela entraîne une augmentation de 10% sur le coût de la colonne de captage, et une baisse de 5% sur le débit.

Les réalisations/perturbations sont obtenues par un tirage aléatoire dans un fichier de réalisations possibles auquel l'apprenant communique le numéro d'étape/sous-étape concerné.

2.4 ANALYSE DU SYSTEME MISE

2.4.1 INTERET PEDAGOGIQUE

Cet intérêt est largement démontré par la motivation et les conclusions des apprenants. Mise se démarque de l'énoncé d'un problème classique et d'un enseignement habituel par plusieurs caractéristiques :

- toutes les données du problème ne sont pas forcément entre les mains des apprenants : soit la recherche de documents se traduit par un échec, soit les informations récoltées sont erronées (c'est alors à l'apprenant de le découvrir). Ce dernier cas est souvent l'occasion d'une prise de conscience de l'apprenant face à la relativité des chiffres lorsqu'il doit brusquement changer de données en cours de projet.

- le travail en équipe implique une organisation des apprenants indispensable devant la complexité et l'ampleur de la tâche. Les choix de stratégies donnent lieu à des discussions (émulation de groupes) et les prises de décisions entraînent la

responsabilité individuelle des coéquipiers : gare aux accusations vengeresses en cas d'échec.

• la planification du projet implique un aperçu d'ensemble des moyens et des techniques disponibles. Cela n'empêche pas l'apprenant de descendre jusqu'au stade du dimensionnement de chaque tuyau. Ainsi s'entrelacent généralités et spécificités d'un projet.

• les liens enseignants/enseignés sont largement ressérés car l'ambiance de la simulation est favorable aux échanges qui peuvent se produire. Face aux cas pratiques auxquels ils sont confrontés, les apprenants hésitent moins à poser des questions, à invoquer des cas limites, pour être certains d'avoir bien assimilé certaines notions.

Les apprenants ne repartent pas au bout de 5 jours avec *x* pages de notes sur les modèles hydrauliques ou autres (qu'ils pourront de toutes façons se procurer dans la littérature), mais avec une expérience, certes fictive, mais suffisamment réaliste pour constituer un germe stable qui pourra plus tard servir de référence.

2.4.2 HOMMAGE AU MISE

Nous avons assisté à plusieurs sessions Mise dans le but avoué d'en extraire des idées d'amélioration. Ce sont ces observations, énumérées plus loin, qui nous ont permis de dresser un cahier des charges pour le logiciel Promise.

En effet, il est apparu ce que nous appelons des 'failles', et non pas des 'erreurs', que nous attribuons à la difficulté de mise au point d'un simulateur, quand il s'agit ainsi d'une première. Ces failles ne peuvent être mise en évidence que par l'existence du Mise qui est un premier pas dans le domaine. Le souci de ses auteurs a bien plus été de prouver le caractère opérationnel de ce système d'E.A.O. que de répondre par avance aux critiques de conception ou de fonctionnement. Ils ont pleinement réussi en la matière et ouvert la voie vers des produits plus élaborés.

2.4.3 FAILLES DU SYSTEME MISE

2.4.3.1 FAILLE CONCEPTUELLE

Le système Mise n'a pas été conçu pour être un interlocuteur à part entière; c'est-à-dire qu'il a besoin pour fonctionner d'interventions incessantes de l'utilisateur : *l'intégration* et la *connection* entre les différents élements n'est pas assez poussée.

Certes, vu avec un certain recul, on trouve dans Mise tous les ingrédients pour mener à bien le projet. De plus près, on s'aperçoit que chaque procédure est indépendante du contexte, chaque étape est isolée dans sa réalisation, et que le système dans son ensemble est incapable de contrôler ce qui se passe durant le projet.

Cet inconvénient tient à l'absence d'un *programme maître* qui pourrait servir de liant à tous les éléments du système Mise. Ceci entraîne un certain nombre de défauts de fonctionnement que nous détaillons ci-après.

De plus, face aux besoins d'évolution d'un tel produit, Mise n'offre que des possibilités parcellaires dans la mesure où chaque fonction est dissociée de l'ensemble : on perd la richesse d'information contenue dans les liens entre les éléments d'un projet. Ce sont ces liens qui vont constituer les règles de réalisation d'une simulation, ou les règles de décision d'une station de travail.

En ce sens, Mise présente une limite d'évolution des scénarios de la simulation, et de la fonction "piège à connaissances".

2.4.3.2 FAILLES FONCTIONNELLES

- *manque d'interactivité*

C'est une conséquence directe de l'absence du programme maître : l'ordinateur est passif et ne prend pas à son compte la gestion des coûts, des décisions et des appels de procédures utiles à la simulation du projet. Ainsi, les réalisations/perturbations doivent être "activées" par l'apprenant qui frappe au clavier le nom d'un programme. Cette action n'a strictement rien à voir avec les notions intervenant dans la conduite d'un projet du domaine de l'eau et vient surcharger et encombrer la démarche de l'apprenant. Il faut donc rendre transparent ce qui dans la réalité apparaît de façon naturelle : la réalisation d'une action par exemple.

• *appels de procédures*

Dans le même ordre d'idées, l'apprenant doit gérer une série de procédures de calcul spécifiques au projet (simulation hydrodynamique de la nappe ou autres). Nous pensons que c'est au simulateur de gérer ces procédures. En conséquence, il ne faut parfois pas hésiter à rajouter à ces procédures un module "projet", voire dans les cas extrêmes à faire des versions "projet" assurant une parfaite intégration dans le système de simulation, bien que l'on perde alors l'aspect universel de la procédure. On gagnera sur l'aspect convivial et opérationnel du simulateur.

• *indépendance des aléas*

Les réalisations/perturbations ne sont déterminées qu'en fonction d'un numéro d'étape/sous-étape et d'un tirage aléatoire parmi une description d'évènements possibles rattachés à une probabilité d'apparition. Il n'y a aucune interaction avec le passé du projet et les décisions qui ont été prises auparavant. Ceci a pour conséquences:

* une certaine linéarité dans le déroulement du projet en ce qui concerne certaines phases; par exemple la recherche de documents et d'informations est une étape qui n'apparaît généralement qu'une fois dans le projet simulé par l'apprenant, souvent au début de la session. En effet, si on veut à nouveau rechercher des informations au milieu de la simulation, le simulateur donnera une réponse indépendante de la première, éventuellement contradictoire. L'apprenant se rend rapidement compte des limites de la simulation et de l'impossibilité de retour vers certaines actions, retour possible dans la réalité.

* des aléas non pertinents car indépendants du contexte. Il n'y a pas de prise en compte du passé du projet, l'alinéa précédent en étant l'exemple le plus simple. Dans sa génération d'aléas, le simulateur doit aussi être sensible à l'enchaînement des décisions, c'est-à-dire que le système doit par exemple refuser la réalisation d'un objectif (ou masquer cette réalisation sous forme d'un évènement équivalent à un échec complet) sachant que l'apprenant n'a pas auparavant franchi une étape indispensable. C'est le cas du financement des travaux qui doit être impossible si l'on a a pas d'abord estimé le coût de ces travaux par une étude.

• *pas de cycles courts prévision-simulation-évaluation*

C'est un cas particulier de ce nous venons d'évoquer. Le système doit être capable d'être un assistant du coordinateur en établissant régulièrement des contrôles soit automatiques (cas du financement des travaux qui doit être précédé d'une étude), soit provoquant l'intervention du coordinateur qui, après avoir pris connaissance de l'état du projet, pourra autoriser la suite des opérations. Nous pensons notamment à l'étape du dimensionnement des conduites qui dans le Mise actuel peut conduire certaines équipes à confectionner des réseaux aberrants soit parce que les procédures sont mal utilisées, soit parce qu'elles sont carrément court-circuitées par manque de temps ou de "conscience professionnelle". Ces grossières erreurs n'apparaîtront qu'au cours de la séance finale d'évaluation.

• *extraction d'expérience, voire de connaissances, délicate*

Le logiciel comptable, destiné à établir le compte général d'exploitation, ainsi que l' historique comptable, est le seul outil qui permet au coordinateur de Mise d'avoir un historique de la session, à condition que les différentes équipes aient bien activé au fur et à mesure ce logiciel et l'aient correctement utilisé. De plus, mis à part les échanges oraux, les apprenants n'ont aucun moyen de laisser des remarques sur le fonctionnement du logiciel, tant du point de vue informatique qu'interne à la logique spécifique du projet. Ceci nuit à l'évolution du système d'E.A.O. car l'enseignant n'a matériellement pas le temps pendant une session de noter des remarques et de réfléchir sur des modifications. Une structure de recueil de réflexions en temps réel permettrait peut-être d'amorcer un débat, même si celui-ci s'établit après la simulation.

Les inconvénients que nous venons d'énumérer n'empêchent pas Mise d'être un système opérationnel, donnant des résultats encourageants. Mais ils mettent en évidence qu'un gros travail doit être accompli pour faire évoluer ce principe d'E.A.O. La structure actuelle n'est pas apte à assurer certaines fonctions représentatives de la réalité, ou facilitant l'évolution de la simulation.

3. LE LOGICIEL PROMISE

3.1 APPROCHE STRUCTURELLE

Compte tenu des remarques précédentes, concernant en particulier les critères d'intégration et d'interactivité du Mise, il faut que le nouvel outil à développer soit un seul logiciel, et qu'il soit le seul interlocuteur de l'apprenant, bref que la machine et le logiciel ne fassent qu'un.

Ainsi, du système Mise, on évolue vers le programme Promise capable de gérer des logiciels et d'assurer les fonctions autrefois contenues dans un système fait d'éléments épars. La nuance entre "système" et "programme" est d'importance car on franchit un degré dans l'approche intégrée d'un projet. En d'autres termes, les différents éléments du système étant considérés comme acquis et maîtrisés, on va s'attacher maintenant à fabriquer la structure pouvant *les accueillir et les agencer*.

Il y a deux types principaux de phénomènes à modéliser et à faire cohabiter dans le simulateur :

- un mécanisme bien identifié, représenté par l'enchaînement répétitif de :

choix d'un objectif (ou prise de décision)	- -->	réalisation de l'objectif (ou conséquence de la décision)

Ce cycle est un algorithme extrêmement simple à écrire et n'offre pas de difficulté particulière. Il imprimera le rythme de la simulation.

Avec cet algorithme, le simulateur est toujours actif puisque dans le pire des cas, il est en attente au début du cycle, prêt à enregistrer une nouvelle décision de l'utilisateur.

- l'écriture de chaque élément du cycle est par contre plus délicate. Le choix d'un objectif peut à la rigueur se ramener à une sorte de menu (style tableau 2-3) où l'utilisateur sélectionne une action. Nous verrons que nous avons essayé d'approfondir ce concept.

La réalisation de l'objectif peut correspondre à l'activation d'un logiciel de calcul, et/ou à l'apparition d'un aléa pertinent, c'est-à-dire relié au passé du

projet. Nous avons tenté de mettre en place une structure modulaire réalisant les fonctions ci-dessus.

La structure de Promise doit également laisser accessible la logique du projet, c'est-à-dire que doivent être paramétrées :

- les décisions;
- les réalisations (procédures ou aléas);

Ceci afin que l'enseignant puisse intervenir facilement pour compléter ou modifier les scénarios de la simulation.

Enfin, Promise doit pallier aux points faibles de l'enseignement au cours d'une simulation, c'est-à-dire, en se référant au tableau 2-2, il doit assurer des fonctions de vérification de la logique de l'enchaînement des décisions de l'apprenant, de la façon dont ce dernier apprend, assurer une fonction de conseil en tenant par exemple à chaque instant des informations générales ou particulières à la disposition des utilisateurs, tout ceci en respectant et en essayant de renforcer l'authenticité de la simulation.

3.2 CHOIX TECHNIQUES

3.2.1 TECHNIQUE DE DEVELOPPEMENT

Un aléa pertinent (ou le déclenchement conditionnel d'une procédure) s'écrit de façon simplifiée :

soit l'objectif (k) à réaliser :
la réalisation sera (r_1) si le passé (p_1) est observé;
la réalisation sera (r_2) si le passé (p_2) est observé;
.....
la réalisation sera (r_n) si le passé (p_n) est observé;

L'objectif (obj_k) s'entend comme la volonté, de la part de l'apprenant, d'entreprendre une action pour faire avancer le projet.

La réalisation (r_n) correspond à ce qui va effectivement se produire suite au lancement de l'objectif; cela peut être le déclenchement d'un évènement et/ou l'activation d'une procédure.

Le passé (p_i) est une suite de conditions ($Cond_i$) représentant ce qui s'est passé pendant la réalisation des objectifs antérieurs.

On a donc une suite de règles du type

SI le passé est (p_j) ***ALORS*** la réalisation est (r_j)

permettant d'identifier un contexte, et d'enclencher une réalisation.

On verra par la suite qu'il ne s'agit surtout pas d'écrire tous les contextes passés possibles, mais de sélectionner les objectifs passés qui vont influencer ou imposer la réalisation de l'objectif en cours.

Les systèmes experts, qui offrent une technique de programmation sous forme de règles et permettent une manipulation aisée des symboles, présentent ici un avantage. Un deuxième avantage apparaît, lié à la structure des S.E. (figure 2-1), dans laquelle la répartition des rôles pourra ainsi s'opérer :

- le moteur prend en charge la partie algorithmique (le cycle de base) et toutes les fonctionnalités (appels de procédures, affichages de messages ...) qui s'y rattachent;
- la base de règles contient la partie paramétrée (la logique) du simulateur; c'est-à-dire les décisions possibles et leurs réalisations associées au contexte. Isolée dans la base de règles, cette logique est plus accessible;
- la base de faits stocke l'historique du projet et devient le centre d'échanges d'informations avec le monde extérieur.

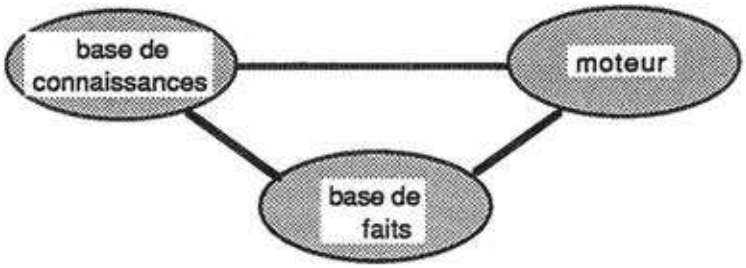


figure 2-1 : structure tripartite d'un S.E.

Toutes ces raisons justifient l'emploi de la technique des systèmes experts pour développer Promise.

3.2.2 LANGAGES ET MACHINES

Turbo-Prolog a été sélectionné. L'annexe A1 développe les caractéristiques de ce langage que l'on récapitule brièvement ici :

- langage de 4ème génération (traitement des listes, récursivité, remontée, traitement des symboles, programmation déclarative ...)
- environnement de développement (fenêtrage, mode trace ...)
- rapidité d'exécution
- faible coût à l'achat
- possibilité de compiler les programmes
- compatible avec le système d'exploitation MS-DOS et les micro-ordinateurs type IBM PC 286 ou 386. Ces machines sont désormais suffisamment puissantes pour assurer les fonctionnalités requises avec une bonne efficacité et des temps de réponse intéressants.

[Il est recommandé au lecteur profane en Prolog de consulter l'annexe relative à Turbo-Prolog pour une bonne compréhension du fonctionnement des logiciels qui suivent. Bien entendu, cette précaution est inutile pour l'utilisateur du simulateur pour lequel aucune connaissance en informatique n'est requise].

3.3 FONCTIONNEMENT DE PROMISE

Ce chapitre traite du fonctionnement du logiciel. Il décrit notamment comment nous avons imaginé l'interface avec l'utilisateur (l'arbre décisionnel), la mise en place des liens entre les éléments du projet (la base de règles), le stockage d'une session (la base de faits) et la structure du moteur favorisant l'émergence d'un certain nombre de fonctionnalités (extraction de connaissances entres autres).

Les solutions apportées essaient d'optimiser l'aspect maintenance et clarté de la base de règles, ainsi que l'aspect convivial du simulateur pour une meilleure intégration de l'utilisateur qui constitue après tout, lui aussi, un élément de la simulation de projet.

3.3.1 LA BASE DE REGLES

FONCTION

La base de règles contient le scénario de la simulation, c'est-à-dire d'une part qu'elle décrit l'ensemble des décisions qui peuvent être prises par l'utilisateur ("l'arbre décisionnel") et d'autre part, qu'elle contrôle le déroulement du projet en générant par exemple les évènements qui s'y produisent ("les réalisations").

3.3.1.1 LES REGLES SONT ECRITES SOUS FORME DE FAITS

Il s'agit ici d'un aspect informatique particulier, mais dont les conséquences sont importantes.

Il peut paraître étrange et inutile d'écrire des règles sous forme de faits, mais cela procure différents avantages :

- on peut ne pas compiler les règles avec le moteur, ce qui va leur procurer une indépendance et une souplesse d'utilisation.

- les règles peuvent être stockées dans un fichier DOS classique, voire dans plusieurs fichiers, ce qui permet de scinder la base de connaissances en unités se référant à telle ou telle partie du projet. Le moteur, selon les besoins, pourra intégrer au fur et à mesure ces unités et/ou en ignorer certaines. Ainsi, l'organisation que l'on peut concevoir de la connaissance sous forme de règles se trouve concrétisée sous forme de différents fichiers : on va pouvoir structurer la base de connaissances en fonction des domaines de compétence intervenant dans un projet et on s'achemine vers une amélioration de l'adéquation entre "solution technique informatique" et "simulation de la réalité".

- la manipulation des règles par le moteur est possible : il peut les manier comme il manie des faits : effacements, ajouts, modifications. C'est une possibilité que nous n'utilisons pas mais qui laisse libre la voie aux interventions dynamiques sur la base de règles.

Ne pas compiler la base de règles va compliquer le travail - donc l'écriture - du moteur. La base de règles est donc "chargée" - on pourrait dire "portée à la connaissance du moteur" - par un ordre '*consult*' qui a pour conséquence de considérer les règles comme des faits.

exemple :

soit la règle: **SI** (A est vrai *et* B est vrai) **ALORS** (C est vrai)

elle pourra être enregistrée dans la base de règles sous la forme du fait :

TEST_REGLE (A,B,C); **test_regle** étant un prédicat

Le travail du moteur consistera alors à interpréter le prédicat *test_regle* comme la règle évoquée plus haut.

Deux prédicats essentiels sont représentés dans la base de règles; ils décrivent :

* l'arbre décisionnel
* les réalisations possibles.

3.3.1.2 L'ARBRE DECISIONNEL

PRINCIPE

Les objectifs - ou décisions - à déclencher au cours de la simulation d'un projet sont modélisés par un formalisme :

Action - *Objet* (de l'action) - *précision(s)*

Action est généralement un verbe,l'*Objet* s'entendant au sens grammatical de 'complément d'objet'; *précision(s)* sert à décrire de façon exhaustive la décision et peut être complété selon un mode récursif.

exemple :

Action	*objet*	*précision*
contacter	élus ou responsables	1) au niveau local 2) le maire
dimensionner	un réseau	AEP

On obtient un arbre décionnel à *n* niveaux:

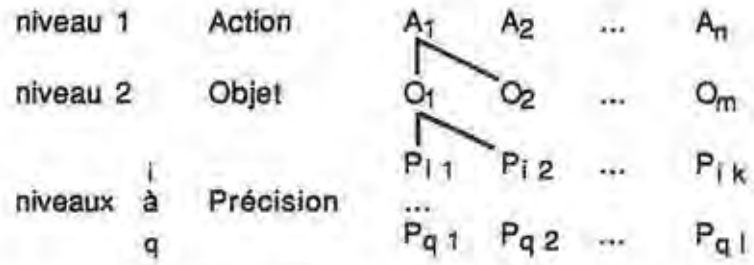


figure 2-2 : schéma de l'arbre décisionnel.

Il se peut qu'un objectif ne possède que 2 niveaux. Inversement, il n'y a pas de limite théorique au nombre de niveaux de précisions. Pratiquement, il ne semble pas souhaitable d'aller au-delà de 4 à 5 niveaux pour une bonne compréhension de l'utilisateur.

Ce formalisme est relativement simple, ce qui est un avantage pour sa maintenance. C'est aussi un inconvénient car il fige la description d'une décision à travers un langage peu évolué. Cependant, il paraît envisageable d'améliorer, en direction de l'utilisateur, cet aspect en lui permettant de s'exprimer en 'français'; le problème est alors de tronçonner une phrase en éléments *Action - Objet - précision(s)* afin de retomber dans le formalisme précédent, compréhensible par le simulateur.

MISE EN ŒUVRE INFORMATIQUE

Cet arbre est décrit par une série de prédicats '*liste*' qui comprennent 2 listes de chaînes de caractères comme arguments :

liste ([L_1] , [L_2]).

Deux de ces prédicats auront obligatoirement comme premier argument L_1 la valeur suivante :

liste ([action] , [L_i]).
liste ([objet] , [L_k]).

Le moteur y reconnaitra la description des niveaux 1 et 2 de l'arbre décisionnel. On aura par exemple (en reprenant le symbolisme du schéma 2-2) :

liste ([action] , [A_1, A_2, ..., A_n]).
liste ([objet] , [O_1, O_2, ..., O_m]).

La description des niveaux de précisions se fait en écrivant en L_1 le chemin emprunté pour arriver jusqu'au niveau i à préciser. L_2 prendra alors pour valeur la précision en question.

exemple : liste ([A_1, O_1] , [$P_{i\,1}$, $P_{i\,2}$]).
liste ([A_1, O_1, $P_{i\,1}$], [P_{n1}, P_{n4}, P_{n7} ,...]).
liste ([A_1, O_1, $P_{i\,2}$], [...]).

Un objectif décrit par simplement 2 niveaux (action - objet) n'aura pas de prédicat '*liste*' de précisions.

On trouvera des exemples complets d'arbres décisionnels dans l'application du chapitre suivant.

3.3.1.3 LES REALISATIONS

PRINCIPE

qualifier une réalisation

Un objectif qui a été lancé - ou une décision qui a été prise - va se traduire par une réalisation. Celle-ci sera plus ou moins favorable. Il s'agit de mémoriser, de qualifier cette réalisation afin d'en garder la trace.

Regarder dans le passé du projet reviendra ensuite à scruter les objectifs réalisés - donc les décisions prises - et leurs qualificatifs. On peut adopter le symbolisme :

un objectif Obj = [A_1, O_1, $P_{i\,1}$,... , P_{n1}] et $r_{i,\ i=1,m}$ ses réalisations

exemple : obj = [contacter, élus ou responsables, au niveau local, le maire]

r_1 = absent, r_2 = en vacances, r_3 = entrevue réussie , ...

Les r_i que l'on décrit sont des mots ou des phrases clés qui résument une réalisation, et qui seront facilement "repérables". A l'utilisateur, il sera par contre proposé des messages très explicites décrivant précisément la réalisation. Nous traiterons des exemples plus loin.

coût d'une réalisation

Nous distinguons 3 coûts temporels ou financiers :

- le délai de réalisation : c'est le temps écoulé depuis le lancement de l'objectif jusqu'à la fin de sa réalisation.

- le temps d'imputation : c'est le temps réellement passé pour la réalisation. Le temps d'imputation est inférieur ou égal au délai de réalisation.

- le coût financier : il s'agit de la somme déboursée (éventuellement) pour la réalisation.

remarque : les coûts salariaux d'un objectif peuvent être soit calculés d'après le temps d'imputation, soit intégrés au fur et à mesure dans les coûts financiers.

exemple :

obj = [effectuer, sondages de terrain, à la pelle mécanique]

* délai = 3 jours (on a rarement une pelleteuse sous la main);
* imputation = 1 journée (1 coup de téléphone pour réserver la pelleteuse + 1 journée sur le terrain pour le creusement des trous);
* coût financier = 1200 F (location d'une pelleteuse + frais de déplacement).

Il est évident que l'on donne là des chiffres moyens et que la réalisation pourra modifier ces valeurs selon les précautions qui auront ou qui n'auront pas été prises, et selon le degré de chance lors du creusement du trou (percement d'une conduite devenant possible si les plans du réseau AEP ou EDF n'ont pas été collectés).

hiérarchie des réalisations possibles d'un objectif

Supposons pour commencer un objectif totalement indépendant des autres, donc indépendant du passé du projet (on pourrait penser que la première décision du projet est indépendante du passé puisque celui-ci n'existe pas. En fait, un passé vide n'est pas innocent; par exemple, si "financer les travaux" est la 1ère décision, la réalisation sera forcément un échec puisqu'on ne connaît rien du projet).

A la manière de Mise, on va décrire un certain nombre de réalisations possibles (r_1, r_2, ..., r_n) et y associer une probabilité ($Prob_1$, $Prob_2$, ..., $Prob_n$). Tout cela peut être représenté le long d'un axe gradué de 0 à 100 (en exprimant les probabilités en pourcentage) :

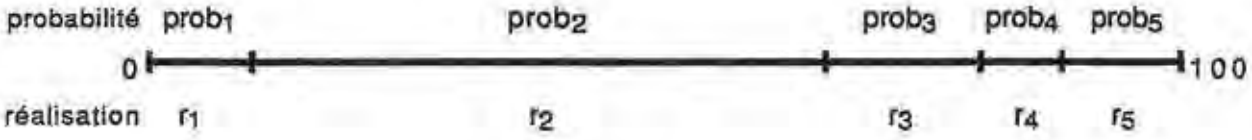


figure 2-3 : axe des réalisations

Un tirage aléatoire dans une loi uniforme entre 0 et 100 permettra de sélectionner une réalisation. On remarquera que, quel que soit l'ordre d'agencement des (r_i) le long de cet axe, le résultat global est le même : au bout de (n) tirages aléatoires, on retrouvera bien que la réalisation (r_i) a une fréquence d'apparition ($prob_i$).

Nous allons utiliser cette propriété pour choisir et imposer un ordre de classement des réalisations le long de l'axe. Ce classement se fera selon l'aspect défavorable, neutre ou favorable de la réalisation vis à vis de l'objectif tenté. On aura par exemple :

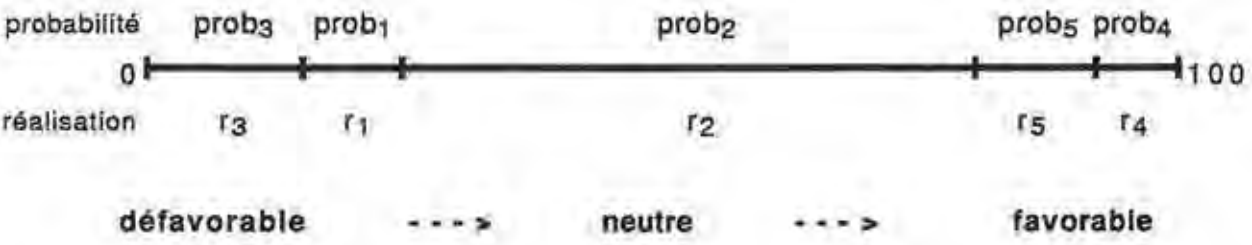


figure 2-4 : axe des réalisations classées

Il s'agit bien d'un classement relatif : toutes les réalisations peuvent présenter un caractère défavorable, celui-ci le sera plus ou moins, ce qui permettra d'établir la hiérarchie.

Bien sur, cette hiérarchie n'est pas toujours facile à mettre en place, les nuances entre "plutôt favorable" et "assez favorable" pouvant être minimes.

Mettre en place ce classement ne relève pas d'un algorithme qui permettrait de quantifier un indice "favorable" ou "défavorable" associé à une réalisation. Le concepteur du classement devra décider de ce classement en fonction de sa propre expérience, et d'un ensemble de critères qu'il serait trop compliqué, voire impossible à décomposer. C'est le domaine de l'heuristique. Nous abordons clairement ici le traitement des connaissances, en abandonnant l'aspect binaire de l'informatique classique.

Il en va de même pour les probabilités associées aux réalisations dont les valeurs ne prétendent pas être exactes - au sens mathématique du terme - ou le fruit d'une longue analyse décomposant cas par cas tout ce qui intervient, mais plutôt "l'idée", issue de son expérience, qu'aura un spécialiste sur les probabilités d'apparition d'un évènement.

influence du passé sur une réalisation

Nous traitions dans l'alinéa précédent un objectif indépendant des autres. Supposons que l'on veuille écrire maintenant que la réalisation d'un objectif (obj_i) - ou d'une décision i - est influencée par la réalisation qu'a pu avoir auparavant un objectif (obj_k) . En supposant cette influence favorable (ce qui est envisageable dans la mesure où les réalisations sont *classées*, comme nous l'avons vu au paragraphe précédent), on pourra représenter cette influence par la règle (ici simplifiée) :

SI l'objectif (k) a eu auparavant la réalisation (r_k)
ALORS c'est plutôt mieux pour la réalisation de l'objectif (i)

{ on verra plus loin comment, à partir des mêmes principes, on écrit la règle
SI l'objectif (k) a eu auparavant la réalisation (r_k)
ALORS la réalisation de l'objectif (i) est *obligatoirement* (r_i) }

Cela revient à provoquer un décalage à droite (valeurs plus proche de 100) sur l'axe des réalisations, donc concrètement à ajouter un bonus au tirage aléatoire dans la loi uniforme. Soit la notation :

description Obj(i) : je décris l'objectif i=[A_1, O_1, $P_{i\,1}$,... , P_{n1}]
Si Obj(k), réal(r_k) : si l'objectif (k) a eu la réalisation r_k
Alors p(a) : alors j'ai une pondération (a) sur le tirage aléatoire de la réalisation de Obj(i)

L'inverse de la règle précédente :

SI l'objectif (k) a eu auparavant la réalisation (r_k)
ALORS c'est plutôt néfaste pour la réalisation de l'objectif (i) {pond. -b}

pourra se noter :

description Obj(i), **Si** Obj(k) réal(r_k) **Alors** p(-b)

Il va de soi que, dans les deux cas, il faut borner l'opération entre 0 et 100. Si le tirage aléatoire pondéré sort des limites 0 / 100, le résultat est ramené à la limite la plus proche.

En complétant ce formalisme, on pourra écrire des règles un peu plus compliquée comme dans l'exemple suivant :

exemple :
SI l'objectif (k) a eu auparavant la réalisation (r_k)
ALORS c'est plutôt mieux pour la réalisation de l'objectif (i) {pond. a}
SINON c'est plutôt néfaste pour la réalisation de l'objectif (i) {pond. -b}

description Obj(i), **Si** Obj(k) réal(r_k) **Alors** p(a),p(-b)

(le premier terme de pondération est appliqué si la condition est vraie, le deuxième si elle est fausse).

On conserve la possibilité d'écrire l'indépendance de la réalisation de l'objectif (obj_i) par rapport à l'objectif (obj_k) en introduisant une *pondération nulle.*

En mixant toute les cas de figures précédents, on pourra par exemple écrire la règle suivante :

SI l'objectif (k) a eu auparavant la réalisation (r_k)
ALORS c'est plutôt néfaste pour l'objectif (i) en cours {pond. -a}
SINON c'est sans importance pour l'objectif (i) en cours {pond. 0}

description Obj(i), **Si** Obj(k) réal(r_k) **Alors** p(-a),p(0)

combinaison de l'influence de plusieurs objectifs passés

En réalité, il va falloir tenir compte d'un ensemble de réalisations passées (ou d'absence de réalisations passées) pour pondérer les réalisations de l'objectif en cours. Deux fonctions vont prendre en charge l'analyse du passé, en combinant de façon logique les réalisations passées avec des "et" et des "ou" :

fonction l_et_influ

(*l_et_influ* pour liste de conditions *influ*entes avec des *et*)
Elle traduit une influence sur la réalisation si *toutes* les conditions qu'on déclare sont remplies.

exemple :

description de l'objectif (i)
SI l'objectif (k_1) a eu auparavant la réalisation (r_1)
et l'objectif (k_2) a eu auparavant la réalisation (r_2)
...
et l'objectif (k_n) a eu auparavant la réalisation (r_n)
ALORS c'est plutôt mieux pour la réalisation de l'objectif (i) {pond. a}
SINON c'est sans importance pour la réalisation de l'objectif (i) {pond. 0}

description Obj(i), **Si** *l_et_influ* { Obj(k_1) réal(r_1)
Obj(k_2) réal(r_2)
...
Obj(k_n) réal(r_n) } **Alors** p(a),p(0)

fonction l_ou_influ

(*l_ou_influ* pour liste de conditions *influ*entes avec des *ou*)

Elle traduit une influence sur la réalisation si *au moins une* des conditions qu'on déclare sont remplies.

exemple :

description de l'objectif (i)

SI l'objectif (k_1) a eu auparavant la réalisation (r_1)

ou l'objectif (k_2) a eu auparavant la réalisation (r_2)

...

ou l'objectif (k_n) a eu auparavant la réalisation (r_n)

ALORS c'est sans importance pour la réalisation de l'objectif (i) {pond. 0}

SINON c'est plutôt néfaste pour la réalisation de l'objectif (i) {pond. -a}

description Obj(i), **Si** *l_ou_influ* { Obj(k_1) réal(r_1)
Obj(k_2) réal(r_2)
...
Obj(k_n) réal(r_n) } **Alors** p(0),p(-a)

Nous avons maintenant un formalisme de règles nous permettant d'associer un coefficient d'incertitude à une réalisation (la probabilité d'apparition) et de pondérer ce coefficient en fonction du passé.

Tout se traite à un niveau extrêmement global, la pondération notamment qui se contente d'indiquer une tendance plus ou moins favorable. Ce système permet de biaiser le tirage aléatoire en "rognant" sur une des deux extrémités de l'axe des réalisations. C'est l'équivalent d'une troncature. Une des conséquences de cette méthode est d'augmenter la fréquence d'apparition des évènements extrêmes au détriment des évènements dits neutres puisque les pondérations vont constamment provoquer un décalage des tirages vers la droite ou la gauche de l'axe des réalisations.

L'influence combinée de plusieurs réalisations (ou groupes de réalisations) passées est décomposée en tachant d'identifier la part d'influence de chacun. Globalement, les actions passées favorables et défavorables peuvent se compenser.

Nous avons envisagé 2 autres techniques de prise en compte du passé qui n'ont finalement pas été retenues. La première consiste à écrire pour chaque contexte passé

les réalisations possibles (toujours rattachées à une probabilité). L'inconvénient est évident et de taille puisqu'il faut décrire tous les contextes. On peut à la rigueur se contenter de décrire les contextes parmi lesquels on a identifié des objectifs passés influents, sachant que dans tous les autres cas, l'axe des réalisations est le même. Mais on se heurte rapidement à une explosion combinatoire; par exemple, 3 objectifs comportant chacun 2 réalisations conduisent déjà à écrire 8 contextes. De plus, dans chaque cas, il faut faire un axe des réalisations !.

La deuxième technique se rapproche de la technique retenue dans la mesure où on va isoler des objectifs (ou groupes) passés qui vont chacun engendrer un axe des réalisations. L'influence globale peut alors s'estimer en calculant un axe moyen des axes de réalisations (moyenne arithmétique de chacune des probabilités) pour lesquels le contexte est vrai. L'inconvénient de cette méthode est qu'elle conduit à une surcharge de la base car il faut décrire beaucoup d'axes de réalisations.

imposer une réalisation

Dans ce système, il faut aussi pouvoir imposer une réalisation, sans chercher la moindre pondération. Cet effet est obtenu en affectant des coefficients de pondération très élevés : on va aller d'un seul coup très loin des bornes habituelles 0/100 de l'axe défini précédemment. Le moteur est conçu pour interpréter cette brusque variation comme une imposition dans la réalisation, et ne prendra plus en compte d'autres influences, ni le tirage aléatoire.

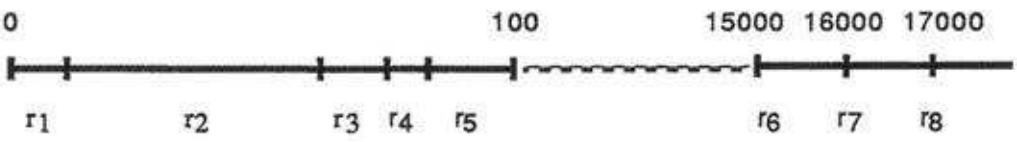


figure 2-5 : extrapolation de l'axe des réalisations

Ces coefficients auront pour valeur minimale 15000.

On peut garder le même formalisme que dans les exemples précédents.

exemple :

description de l'objectif (i)

SI l'objectif (k) a eu auparavant la réalisation (r_k)

ALORS la réalisation de l'objectif (i) est obligatoirement (r_i)

description Obj(i), **Si** Obj(k) réal(r_k) **Alors** p(c),p(0)
avec c ≥ 15000

Comme avec l'influence sur une réalisation, on pourra combiner une série de réalisations passées avec des "et" et des "ou" pour imposer éventuellement une réalisation. C'est l'objet des deux fonctions suivantes :

fonction l_et_obli

(*obli* étant l'abréviation pour obligé)

Elle traduit une obligation sur la réalisation si *toutes* les conditions qu'on déclare sont remplies.

exemple :

description de l'objectif (i)

SI l'objectif (k_1) a eu auparavant la réalisation (r_1)

et l'objectif (k_2) a eu auparavant la réalisation (r_2)

...

et l'objectif (k_n) a eu auparavant la réalisation (r_n)

ALORS la réalisation de l'objectif (i) est obligatoirement (r_i)

description Obj(i), **Si** *l_et_obli*{ Obj(k_1) réal(r_1)
Obj(k_2) réal(r_2)
...
Obj(k_n) réal(r_n) } **Alors** p(c),p(0)

fonction l_ou_obli

Cette fonction est en fait implicitement contenue dans la précédente dans la mesure où le succès d'une obligation de réalisation va provoquer le court-circuit de toutes autres conditions. Elle n'a donc pas été développée.

exemple :

description de l'objectif (i)

SI l'objectif (k_1) a eu auparavant la réalisation (r_1)
ou l'objectif (k_2) a eu auparavant la réalisation (r_2)
...
ou l'objectif (k_n) a eu auparavant la réalisation (r_n)
ALORS la réalisation de l'objectif (i) est obligatoirement (r_i)

revient à écrire une série de fonctions I_et_obli du type :

SI l'objectif (k_1) a eu auparavant la réalisation (r_1)
ALORS la réalisation de l'objectif (i) est obligatoirement (r_i)
SI l'objectif (k_2) a eu auparavant la réalisation (r_2)
ALORS la réalisation de l'objectif (i) est obligatoirement (r_i)
...
SI l'objectif (k_n) a eu auparavant la réalisation (r_n)
ALORS la réalisation de l'objectif (i) est obligatoirement (r_i)

Remarque :

Dans le cas d'une réalisation imposée, la deuxième pondération est toujours nulle (en fait, le moteur l'ignore). En effet, sachant :

- qu'une fonction I_et_obli, si elle est appliquée, va court-circuiter toutes autres influences, ainsi que le tirage aléatoire;
- que bien qu'on essaie de sortir de l'aspect binaire de l'informatique, il n'en reste pas moins vrai que la proposition " Obj(k_i) réal(r_i) " est soit vraie, soit fausse (et obligatoirement une des deux);

si on applique dans les deux cas une pondération, la fonction I_et_obli réussira toujours , et le simulateur stoppera l'exploration du passé dès qu'il rencontrera une fonction I_et_obli.

On échappe à cet effet de bord en n'appliquant la fonction I_et_obli que si " Obj(k_i) réal(r_i) " est vraie (on verra plus loin qu'on peut tester si la proposition est fausse en considérant l'absence d'une réalisation).

test d'une réalisation quelconque d'un objectif

Il s'agit de tester si un objectif a été tenté par le passé, quelle que soit la réalisation qu'il ait pu avoir. On utilisera le qualificatif réservé "*réal*" pour "*réalisé*".

exemple :

description de l'objectif (i)

SI l'objectif (k_1) a déjà été tenté

et l'objectif (k_2) a eu auparavant la réalisation (r_2)

...

et l'objectif (k_n) a eu auparavant la réalisation (r_n)

ALORS c'est plutôt mieux pour la réalisation de l'objectif (i) {pond. a}

SINON c'est sans importance pour la réalisation de l'objectif (i) {pond. 0}

description Obj(i), **Si** *l_et_influ* { Obj(k_1) réal(*réal*)
Obj(k_2) réal(r_2)
...
Obj(k_n) réal(r_n) } **Alors** p(a),p(0)

test de l'absence d'une réalisation passée

Il s'agit de prendre en compte la "non-réalisation" r_i d'un objectif; il suffira de faire précéder r_i par "*n_*" pour tester l'absence d'une réalisation.

exemple :

description de l'objectif (i)

SI l'objectif (k_1) n'a pas eu la réalisation (r_1)

et l'objectif (k_2) a eu auparavant la réalisation (r_2)

...

et l'objectif (k_n) a déjà été tenté

ALORS c'est plutôt mieux pour la réalisation de l'objectif (i) {pond. a}

SINON c'est sans importance pour la réalisation de l'objectif (i) {pond. 0}

description Obj(i), **Si** *l_et_influ* { Obj(k_1) réal(*n_r1*)
Obj(k_2) réal(r_2)
...
Obj(k_n) réal(réal) } **Alors** p(a),p(0)

test de l'absence totale de toute réalisation d'un objectif

C'est un cas particulier des 2 alinéas précédents : pour prendre en compte un objectif qui n'apparaît pas du tout dans le passé du projet, on utilisera le qualificatif "*n_réal*" pour "non réalisé".

exemple :

description de l'objectif (i)
SI l'objectif (k_1) n'a pas eu la réalisation (r_1)
et l'objectif (k_2) n'a pas eu auparavant de réalisation
...
et l'objectif (k_n) a déjà été tenté
ALORS c'est plutôt mieux pour l'objectif (i) en cours {pondération a}
SINON c'est sans importance pour l'objectif (i) en cours {pondération 0}

description Obj(i), **Si** *l_et_influ* { Obj(k_1) réal(n_r_1)
Obj(k_2) réal(*n_réal*)
...
Obj(k_n) réal(réal) } **Alors** p(a),p(0)

l'influence sur les coûts de réalisation

Chaque type de réalisation peut avoir une influence sur les 3 coûts prévus. Ceux-ci seront augmentés si la réalisation est défavorable et vice-versa.

soit ∂d est la modification sur le délai de prévision,
∂i " " " " le temps d'imputation,
∂f " " " " le coût financier,
le formalisme de règle peut ainsi être complété :

description Obj(i), **Si** *l_et_influ* { Obj(k_1) réal(*n_réal*)
Obj(k_2) réal(r_2)
...
Obj(k_n) réal(r_n) }
Alors p(a),p(0), c(∂d), c(∂i), c(∂f)

Les coûts réels après réalisation s'écriront (j + ∂j).

l'agencement général d'une règle

En reprenant tous les éléments évoqués, nous allons pouvoir fabriquer une *règle* régissant la réaction du simulateur face au lancement d'un objectif. Cette règle est organisée en 3 parties :

• identification de l'objectif
on y décrit quel objectif est en cause, par reconnaissance avec un chemin de l'arbre décisionnel. On y trouve aussi les coûts moyens prévus et une information sur l'objectif (contenue dans un fichier, à l'usage de l'utilisateur).

• analyse du passé
c'est une suite de fonctions l_et_influ, l_ou_influ, l_et_obli qui, en scrutant le passé du projet, vont influencer ou imposer une réalisation. Il est préférable de décrire d'abord les fonctions l_et_obli puisque si l'une d'entre elles s'avère vraie, le simulateur fera abstraction des autres conditions.

• la réalisation
c'est la description de l'axe des réalisations et des influences sur les coûts prévus.

MISE EN ŒUVRE INFORMATIQUE

L'ensemble d'une règle est inscrite dans le prédicat "description" qui a 7 arguments:
- 5 pour la partie identification de l'objectif,
- 2 pour les deux autres parties.

• les 5 premiers arguments sont :
- une liste du type [A_1, O_1, $P_{i\,1}$,... , P_{n1}] décrivant l'objectif en question
- un nom de fichier qui contient diverses informations sur l'objectif à l'usage de l'utilisateur (ex : qu'est-ce qu'un sondage à la pelle mécanique, qu'est-ce qu'on peut en attendre ...)
- les 3 coûts moyens prévus, temporels et financier.

• le deuxième argument est une liste constituée des fonctions d'analyse du passé, avec les incidences sur le tirage aléatoire.

• le troisième argument décrit, tronçon par tronçon et en définissant les bornes de chaque réalisation, l'axe des réalisations et la modification des coûts dans chaque zone.

Une règle générique s'écrit :
(où *ch* est une fonction introduisant une chaîne de caractères)
l_ch " " " une liste de chaînes de caractères,
p " " " un réel)

```
                                        1ère partie
description(                        |   nom du prédicat
   [ obj_i ],                       |   du type [ A1, O1, Pi 1 ,... , Pn1 ]
   nom1,                            |   nom du fichier DOS d'information
   Cd, Ci, Cf,                      |   les 3 coûts prévus
                                        2ème partie
   [fonction ( [                    |   fonction=l_et_influ par exemple
         l_ch( [ obj_k ] ), ch( r_k ),  |   si (obj_k) a eu (r_k) comme réalisation
         l_ch( [ obj_m ] ), ch( r_m ),  |      et / ou selon la fonction
         ...                        |      ...
         l_ch( [ obj_n ] ), ch( r_n ),  |      ...
         p(a), p(b) ] ),            |   pondération appliquée sur le tirage
                                    |
   ... ],                           |   nouvelle fonction
                                        3ème partie
   [ p( b1 ), p( b2 ),              |   entre les bornes b1 et b2
     p( ∂d ), p( ∂i ), p( ∂f ),     |   influence sur les coûts
     ch(nom2),                      |   nom du fichier DOS de la réalisation
     ch( rj),                       |   qualificatif de la réalisation
                                    |
   ... ]                            |   nouvelles bornes de l'axe
).                                  |   fin de la règle
```

On trouvera des exemples de règles dans l'application du chapitre suivant.

3.3.1.4 PARAMETRAGE DE PROCEDURES

Nous venons de présenter les 2 principaux prédicats

-liste et

-description

qui interviennent dans la base de règles et qui constituent le "combustible" du moteur.

Cette base se présentant sous forme d'un fichier, donc d'accès relativement facile, on peut avoir avantage à y regrouper certaines fonctions de paramétrage de procédures, surtout si celles-ci sont écrites en Turbo-Prolog. Ceci ne constituant nullement une obligation, chaque procédure pouvant posséder son propre fichier de paramétrage.

3.3.1.5 CONCLUSION SUR LA BASE DE REGLES

La base de règles, logique du simulateur, contient (mises à part d'éventuelles fonctions de paramétrage de procédures) :

- un prédicat de description de tous les objectifs possibles (c'est l'arbre décisionnel);
- un prédicat permettant d'identifier chaque objectif et explicitant ses réalisations conditionnelles; cette deuxième partie est constituée à partir d'une cellule de base telle que sur la figure 2-6.

Il y a différents avantages à construire la base de règles à partir d'une "brique" élémentaire :

- en rajoutant des briques, ou en étoffant le contenu d'une brique, on enrichit petit à petit les capacités de la base;
- on peut incorporer à la base un nouveau type de cellules, pourvu qu'elles soient compatibles avec le type que nous venons de décrire. On peut imaginer des cellules prenant en compte des fonctions numériques par exemple;
- la maintenance est améliorée par la grande structuration de la base de connaissances.

La construction d'une cellule se fait par étapes :

1) étude des objectifs passés influents
2) sélection des réalisations possibles et classement

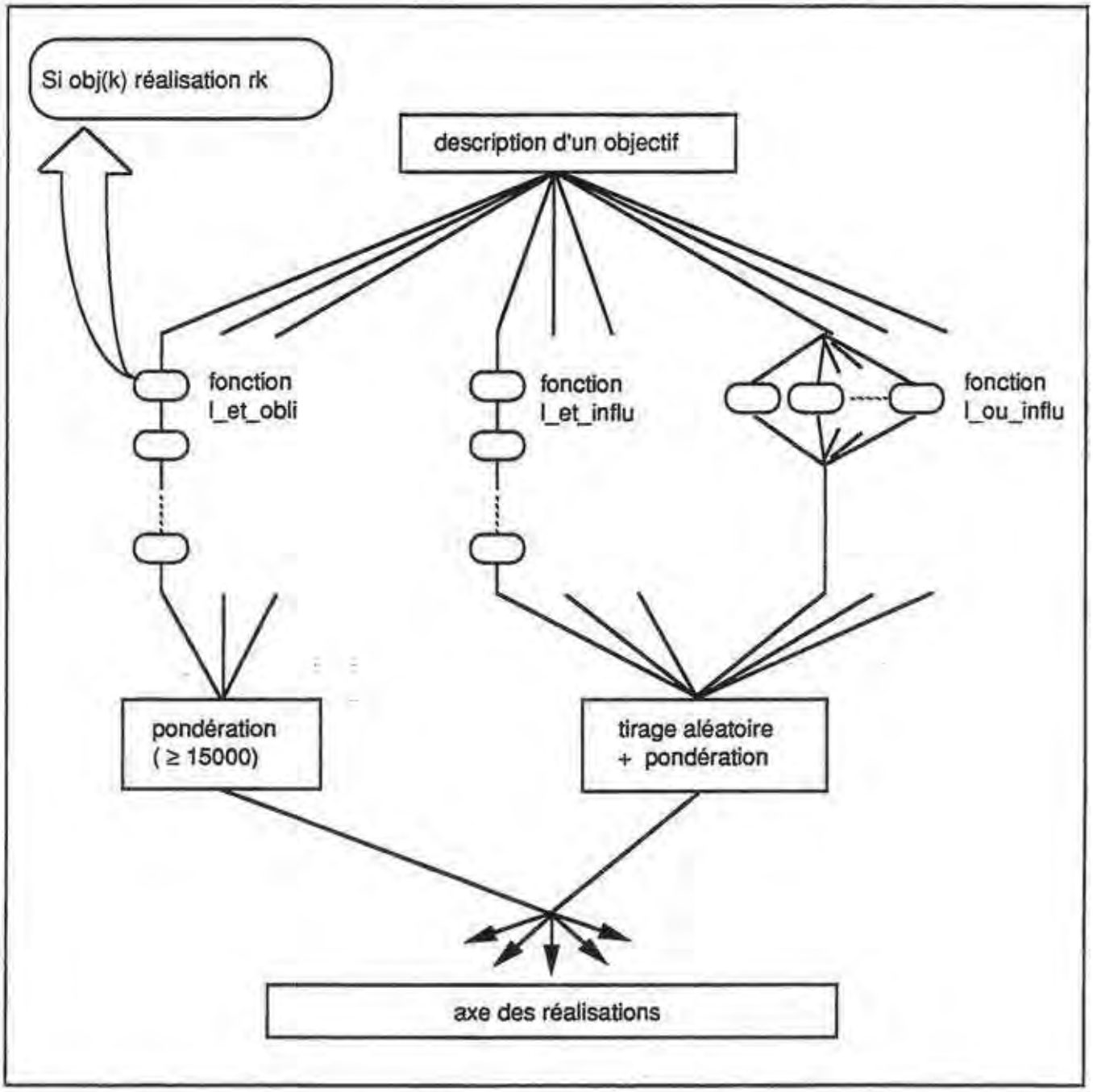


figure 2-6 : cellule élémentaire de la base de règles

3) choix des probabilités d'apparition
4) étude des pondérations en fonction du passé

On remarquera cependant qu'il n'y a pas indépendance complète entre les cellules. Les jonctions sont assurées par les qualificatifs de chaque réalisation. En effet, quand on écrit une condition (Obj(i) réal(r_j)), il faut bien que la réalisation (r_j) apparaisse parmi les réalisations possibles d'un objectif, sinon la condition ne sera jamais vraie (à moins que (r_j) apparaisse dans une procédure).

Comme on le verra sur les exemples d'applications, la base de règles est, d'un point de vue syntaxique, difficile à manipuler: elle doit respecter la syntaxe Prolog. De plus, la description des objectifs dans les différentes parties de la base doit être rigoureusement identique; une faute d'orthographe nuira à la bonne reconnaissance d'un objectif. Enfin, chaque règle de simulation est associée à un ou plusieurs fichiers (fichiers d'informations, de réalisations ...); l'absence d'un fichier pourrait conduire à une erreur informatique en cours d'exécution.

Toutes ces contraintes sont prises en charge par un programme (logiciel STB - voir annexe A2) dont l'objectif est de signaler toutes les imperfections de la base de règles afin de délivrer au moteur une base de connaissances saine, d'un point de vue orthographique, syntaxique et informatique.

3.3.2 LA BASE DE FAITS

FONCTION

La base de faits concerne 1 session de simulation. Elle en est la mémoire en stockant les différents évènements qui s'y produisent : c'est dans la base de faits que se conserve l'histoire du projet simulé.

3.3.2.1 C'EST UN FICHIER DYNAMIQUE

Par construction en Turbo-Prolog, la base de faits est un fichier dynamique en mémoire vive (ou RAM). On peut donc intervenir sur ce fichier avec des fonctions d'ajout ("*asserta*" ou "*assertz*"), de retrait ("*retract*") ou de modification (en passant par les étapes de retrait et d'ajout après modification) de faits. De plus, ce fichier peut

être sauvegardé sur un support physique (fonction "*save*") tel que disque dur ou disquette.

Cette dernière caractéristique permet d'interrompre la simulation à n'importe quel moment, d'utiliser la machine pour d'autres besoins, et de relancer la simulation là où on s'était arrêté. Il suffit de sauvegarder la base de faits et de la relire (fonction "*consult*") au moment du redémarrage.

3.3.2.2 STOCKAGE DU PASSE

Comparativement aux développements précédents sur la base de règles, ceux qui interviennent dans la base de faits sont très simples. Au fur et à mesure que les réalisations se font, elles sont enregistrées sous la forme : obj_i, r_j .

D'un point de vue informatique, il s'agit du prédicat "*fait*" avec 2 arguments représentant une liste de chaînes de caractères du type [A_1, O_1, $P_{i\,1}$,... , P_{n1}] et une chaîne de caractères (le qualificatif de la réalisation).

exemple : fait ([obj_i], r_j).

3.3.2.3 UN CENTRE D'ECHANGE D'INFORMATIONS

La base de faits, de par sa grande maniabilité et sa structure simple, peut constituer un moyen d'échange d'informations entre le moteur et des procédures, ou entre procédures. Ces dernières peuvent être des sous systèmes experts qui analysent aussi le passé du projet, ou bien des procédures de calcul qui laissent une trace - valeurs numériques issues d'un modèle mathématique par exemple - à l'intention d'une autre procédure si elle est déclenchée. Dans ce cas, de nouveaux types de prédicats doivent être introduits, qui sont capables de manipuler ces valeurs numériques.

3.3.3 LE MOTEUR

FONCTION

Le moteur assure d'une part l'interface avec l'utilisateur, et d'autre part, la communication entre les différents éléments du simulateur (base de règles, base de faits, procédures ...) et leur mise en œuvre.

Il serait bien trop technique et fastidieux d'expliciter en détail le fonctionnement du moteur. Nous nous contenterons d'indiquer son organisation très générale et les principales fonctions qui y sont traitées (on trouvera en annexe A2 des spécifications techniques).

algorithme général

C'est le cycle de base évoqué plus haut, avec quelques compléments :

(a) ouvrir la base de connaissances
(b) ouvrir la base de faits
(c) choisir un objectif <----------
(d) récapituler la situation |
(e) réaliser en actualisant la base de faits | oui
|
on continue la simulation - - - - - - - - -
| non
v
(f) sauvegarde de la base de faits
(g) arrêt

(a) et (b) : ces deux ordres permettent de porter à la connaissance du moteur
- le ou les scénarios de la simulation (base de règles)
- l'état du projet (base de faits).

(f) : lorsqu'on sort de l'algorithme général, on sauvegarde toujours le passé du projet.

(g) : cet arrêt de Promise a lieu dans deux cas :
- arrêt de la simulation (fin du projet ou utilisation de la machine à d'autres fins);
- activation automatique d'une procédure par Promise qui "laisse donc la main". Il la reprendra à la fin de l'exécution de cette procédure.

Les autres lignes de l'algorithme sont détaillée ci-après.

(c) : choisir un objectif

Il s'agit de définir une séquence Action - Objet - Précision(s) choisie dans l'arbre décisionnel. Les différents niveaux de cet arbre sont représentés dans des fenêtres comme sur la figure 2-7 où l'action "contacter" et l'objet "élus ou responsables" ont déjà été sélectionnés, le curseur étant sur "au niveau régional" de la fenêtre "précision".

L'utilisation du curseur permet de se déplacer dans une fenêtre en activant la vidéo inverse sur la ligne en cours de sélection. La touche "retour chariot" ("CR") permet de confirmer la sélection. La fenêtre "précisions" apparaît tant qu'on a pas atteint le dernier niveau de l'arbre de décision. La fenêtre "sélection" récapitule au fur et à mesure les sélections opérées.

Quand un chemin complet de l'arbre a été décrit, une confirmation de la sélection globale réalisée est demandée; on peut à ce niveau annuler un objectif sélectionné.

touche F1 : infos

Cette touche est accessible à n'importe quel moment. Elle permet d'accéder au fichier d'informations concernant un objectif, ainsi qu'aux coûts temporels (en jours) et financiers (en kilo-francs) prévus (voir figure 2-8).

3 types de réponses peuvent être fournies par le simulateur :

- une information comme sur la figure 2-8;
- la réponse "veuillez préciser votre objectif" si l'objectif n'est pas complètement décrit et que l'ordinateur reconnaît le début d'un chemin de l'arbre décisionnel;
- la réponse "pas d'information sur cet objectif" s'il n'y a effectivement pas d'intérêt à fournir une information (objectif très explicite) ou si le moteur ne reconnaît ni un début, ni l'intégralité d'un chemin de l'arbre décisionnel.

touche F3 : remarque

Un éditeur de texte est activé parallèlement à l'ouverture d'une fenêtre dans laquelle l'utilisateur inscrit le texte de son choix. Cette option permet en temps réel la saisie des remarques de l'utilisateur. Promise stocke dans un fichier le texte entré ainsi que la situation du simulateur au moment de la remarque (quel objectif était en cours).

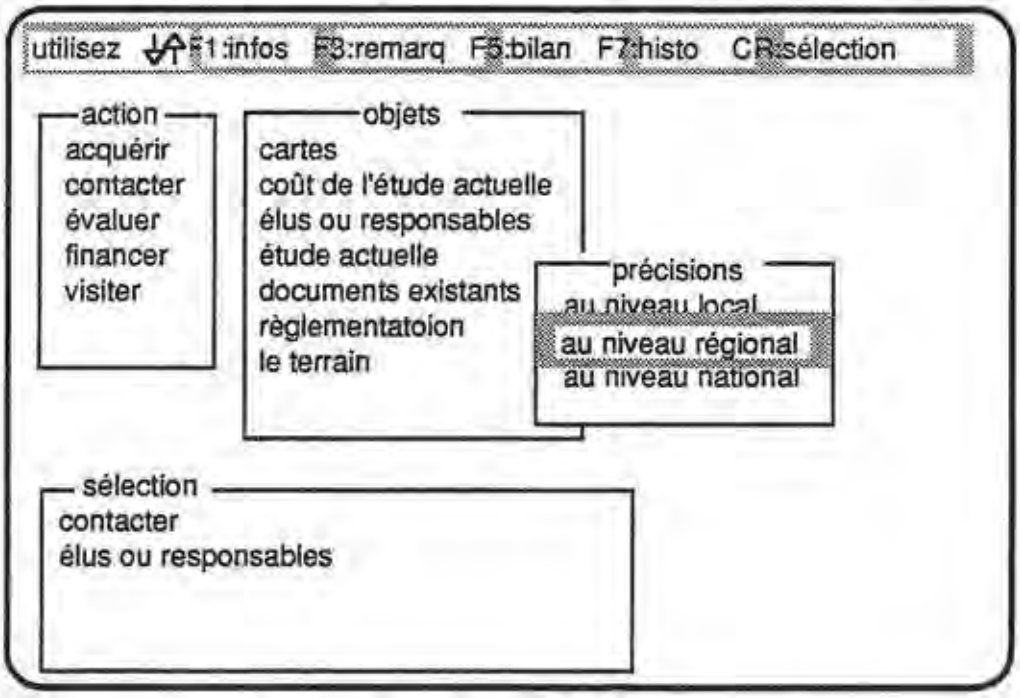


figure 2-7 : exemple de choix d'un objectif; présentation de l'écran

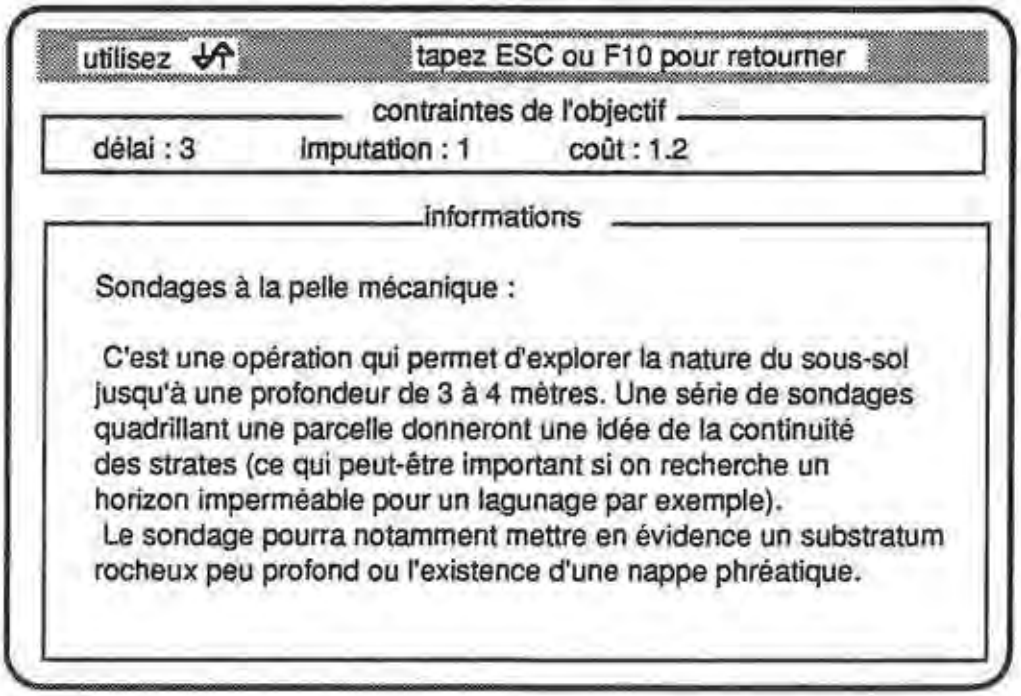


figure 2-8 : exemple d'écran d'informations

touche F5 : bilan

En frappant cette touche, l'utilisateur voit apparaître le bilan de son projet en terme de coûts temporels et financier. Il s'agit d'un cumul de ces coûts au fur et à mesure que des réalisations se font. Ce cumul est réalisé automatiquement par le moteur et l'utilisateur n'y a pas accès.

touche F7 : historique

Le moteur stocke dans un fichier tout ce qui se passe dans le projet au niveau des réalisations. Ce stockage est sommaire, identique à l'information contenue dans la base de faits récapitulant le passé du projet. C'est ce fichier qui peut être consulté par la touche F7.

(d) récapitulation d'un objectif

Cette étape récapitule l'objectif qui a été sélectionné, ses coûts associés et le bilan du projet; les touches (f1:infos), (F3:remarq) et (F7:histo) sont toujours accessibles (voir figure 2-9).

Cet écran est très utile dans la phase d'apprentissage du simulateur par l'apprenant car il marque bien la frontière entre les étapes "sélection d'un objectif" et "réalisation de l'objectif"; par la suite, son apparition systématique peut paraître rébarbative. Cependant, c'est une dernière chance offerte à l'utilisateur pour annuler son objectif (touche Esc).

(e) réalisation de l'objectif sélectionné

C'est à cette étape que le moteur travaille le plus et de façon totalement transparente à l'utilisateur. Une fois l'objectif acquis, la règle correspondante est recherchée. Si elle n'est pas trouvée (objectif non répertorié ou aberrant dans la mesure où on peut combiner à loisir les actions et les objets), le message suivant apparaît :

"je ne comprend pas ...
veuillez formuler autrement votre objectif"

Quand la bonne règle est isolée, le moteur en étudie les conditions (à la manière du schéma 2-6) pour en déduire une réalisation, et gère les incidences sur les différents coûts (figure 2-10).

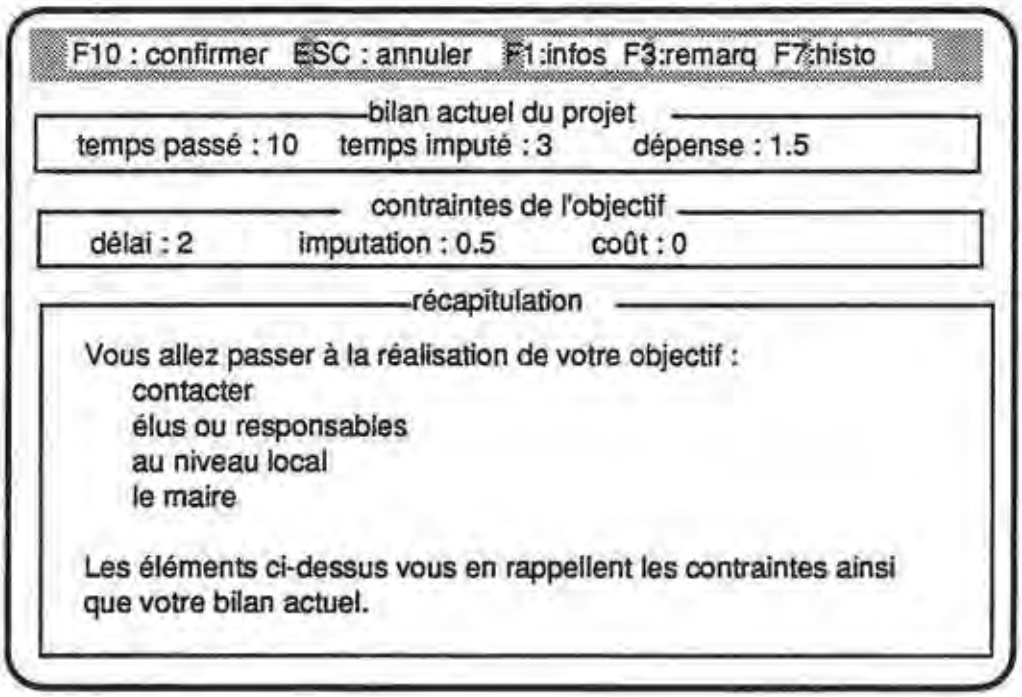


figure 2-9 : exemple d'écran à l'étape "récapitulation"

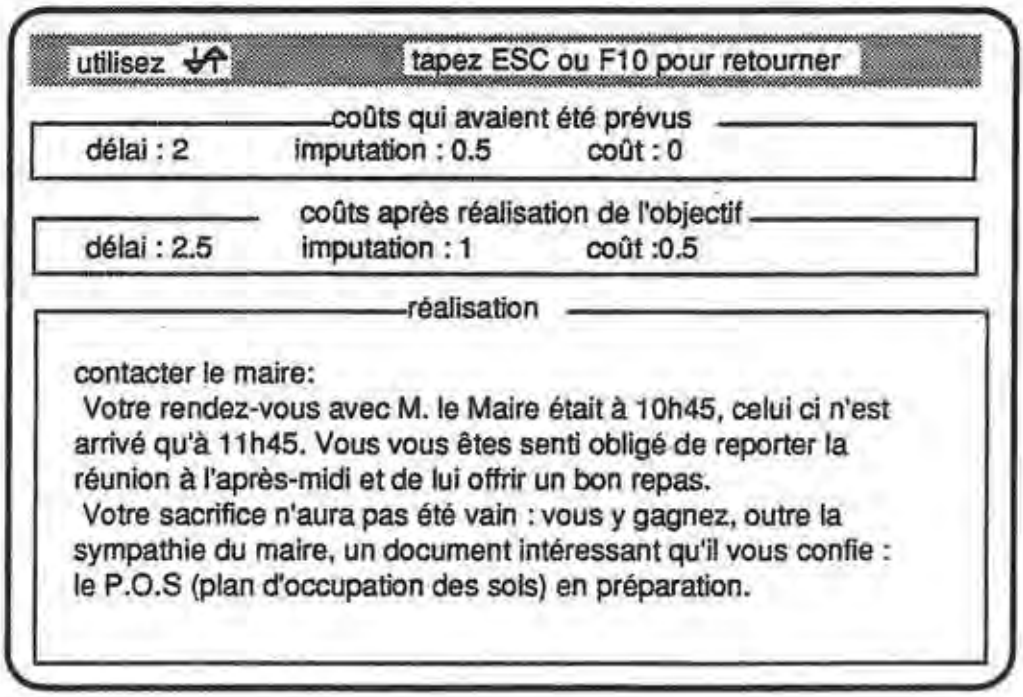


figure 2-10 : exemple d'un écran de l'étape "réalisation".

activation d'une procédure

La réalisation peut aussi correspondre à l'activation d'une procédure. La syntaxe de la règle reste identique. Le simulateur saura qu'il faut activer une procédure si le nom du fichier de réalisation commence par la lettre "*p*" (comme procédure), la suite de ce nom étant le nom de la procédure (ex: "phydro" pour activer la procédure "hydro").

Le simulateur laissera alors la main le temps de l'exécution de la procédure et la reprendra automatiquement le pilotage de la session dès la fin de la procédure.

Il va de soit que les procédures sont totalement indépendantes du simulateur, c'est-à-dire qu'elles sont compilées séparément et écrites dans n'importe quel langage.

activation d'une question

Avant d'afficher une réalisation ou de lancer une procédure, le simulateur peut poser une question.Tout comme pour la fonction (F3:remarque), l'éditeur de texte est activé et l'utilisateur peut alors répondre. Le simulateur saura qu'il faut activer une question si le nom du fichier de réalisation commence par la lettre "q" (comme question), la suite de ce nom étant le nom du fichier où est contenue la question. La réponse est récupérée dans un fichier spécifique (voir annexe A2).

Cette question, que le concepteur de la base de règles a prévu de poser dans un certain contexte, peut concerner un point technique ou stratégique. Il peut s'agir d'un moyen de contrôle de l'apprenant ou une source d'extraction de connaissances quand un spécialiste est présent pour l'amélioration de la base de connaissance. C'est le point que nous abordons dans le chapitre suivant.

3.4 ASPECT "EXTRACTION DE CONNAISSANCES"

Toute action pédagogique provoque une circulation d'informations dans le sens "enseignant --> enseigné". C'est ce qui se passe dans Promise qui est d'abord un outil d'E.A.O. Cependant, il nous a paru intéressant d'ouvrir quelques passages par lesquels l'enseignant pourrait avoir un retour lui permettant d'améliorer les performances du simulateur, de compléter la base de règles ou d'éclaircir un point technique particulier (surtout quand des spécialistes participent à une session). Trois moyens peuvent être employés :

- l'enregistrement des remarques de l'utilisateur : c'est la touche F3, qui ouvre une fenêtre et active le simulateur. Ici, on fait appel à une démarche volontaire

de l'utilisateur qui, de sa propre initiative, va dire ce qu'il pense sur un aspect quelconque du logiciel ou sur la base de règles.

• le simulateur peut poser des questions qui sont programmées pour être activées dans un certain contexte du projet (exploration du passé) : c'est une démarche qui va provoquer l'utilisateur, l'obligeant à se justifier et donc à réfléchir. Eventuellement, cette question servira à remettre sur une bonne voie l'apprenant.

• le simulateur peut provoquer un contexte : soit par l'intermédiaire des réalisations (imposer un évènement pour un passé donné du projet), soit par l'intermédiaire des procédures où il pourra s'agir d'un aspect plus technique. En assainissement individuel par exemple, en présence d'un spécialiste, les paramètres d'une parcelle pourront être choisis pour représenter un cas limite entre deux filières.

De plus, nous pensons qu'une importante source d'informations est contenue implicitement dans la base de connaissances en ce qui concerne les règles de conduite d'un projet. Il s'agit alors plus d'auto-extraction de connaissances de la part de l'enseignant. La constitution d'un scénario de simulation oblige à une réflexion approfondie sur l'interférence des différentes actions possibles, sur l'opportunité d'y inclure certains outils ... La formalisation de ces connaissances et la mise en place des contraintes entre les éléments du projet sont un premier pas pour éclaircir la démarche du spécialiste.

3.5 CONCLUSION

A partir d'une idée originale, le Mise, nous avons tenté de construire un logiciel capable d'assurer un certain nombre de fonctions nouvelles, et de représenter les interactions entre les différentes composantes de la conduite d'un projet.

Ce logiciel s'appuie sur la technique des systèmes experts avec une structure tri-partite "base de règles - moteur - base de faits" (figure 2-11). Dans ce cas, le langage Prolog amène une grande souplesse de programmation, notamment par le traitement des listes. Cependant, c'est au niveau fonctionnel, directement lié à l'aspect

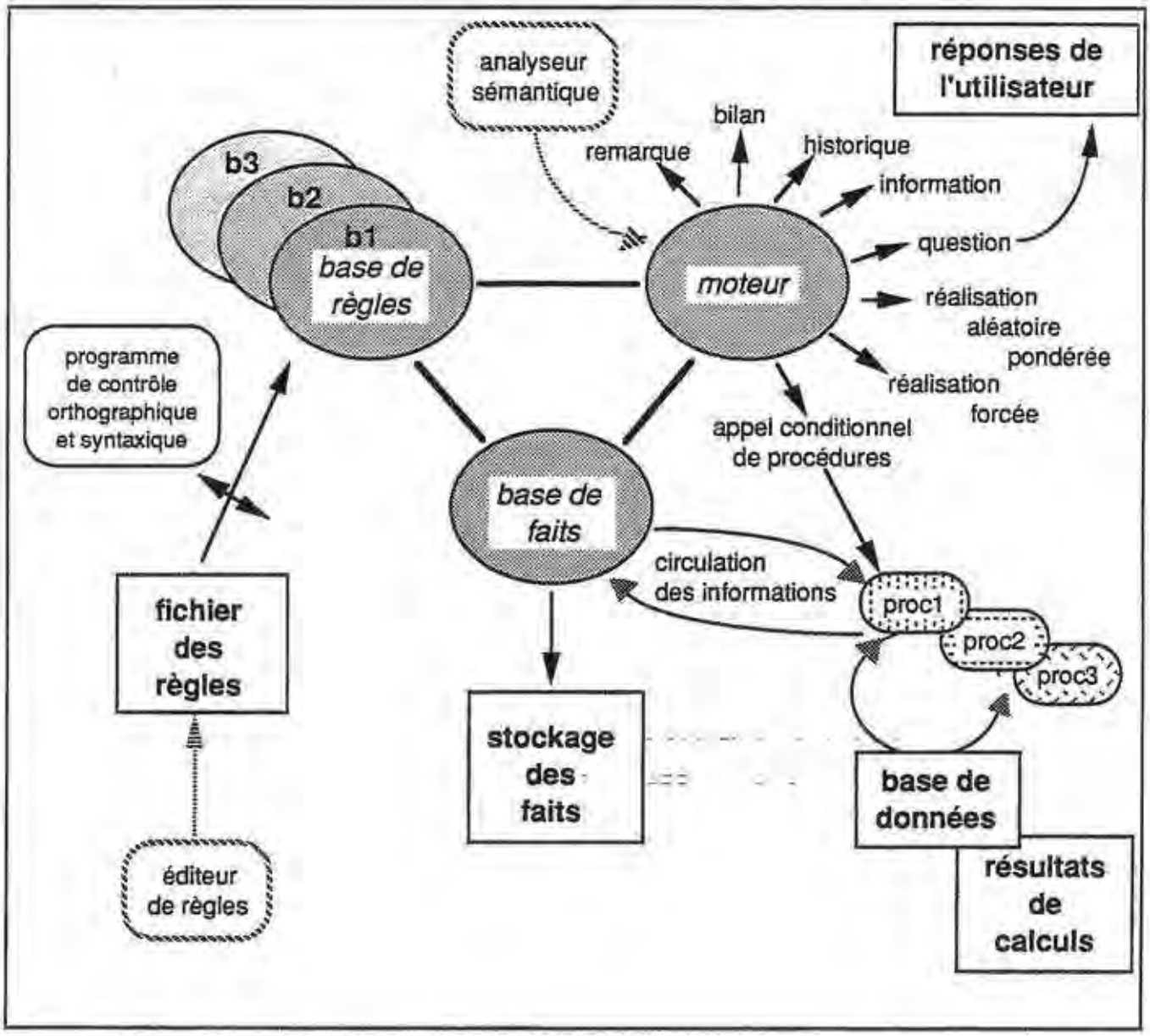


figure 2-11 : schéma informatique du simulateur

maintenance, que la séparation des tâches de Promise est intéressante : chaque entité peut-être travaillée séparément pour améliorer les capacités de l'ensemble :

• la base de faits est la structure la plus simple; elle est la mémoire d'une session de simulation.

• le moteur assure toutes les fonctions d'affichage, de gestion de coûts, d'appel de procédures et d'exploitation de la base de connaissances : interprétation des règles et présentation des objectifs possibles. Ce dernier problème, résolu à partir d'un formalisme "action - objet - précisions", pourrait être amélioré par l'introduction d'un interpréteur sémantique qui permettrait à l'utilisateur de s'exprimer librement.

• la base de règles, adaptée à la simulation de projet dans un but pédagogique, est constituée à partir de cellules pseudo-indépendantes (figure 2-6). La mise au point d'une base de règles représente un travail à part entière, long et difficile. Promise ne prétend pas régler ce problème mais propose une structure d'accueil de la connaissance. Cette tache pourrait être facilitée par la mise au point d'un éditeur de règles permettant un affichage agréable et des fonctions d'ajout ou de modification conviviales. En effet, la base de règles doit respecter la syntaxe Prolog ce qui nuit actuellement à sa lisibilité. De plus, à partir d'une certaine taille, la manipulation des fichiers n'est pas aisée. Tous ces inconvénients peuvent être pris en charge par l'éditeur.

Nous décrivons dans le chapitre suivant des exemples d'arbre décisionnel et de règles de simulation. Il nous paru également important de confier la mise au point d'une base de règles à une personne non avertie en Prolog afin de vérifier la maniabilité de sa structure, et de confirmer la robustesse informatique de Promise par un test de simulation avec un utilisateur.

4. EXEMPLE D'APPLICATION

La mise au point de Promise a été réalisée à partir d'exemples pris dans un projet d'assainissement en milieu rural. L'objectif de ce chapitre est de montrer comment on va concrètement réaliser les arbres décisionnels et les règles de simulation en s'appuyant sur le projet en question.

Les tests que nous présentons ensuite n'ont pas eu pour but de valider une base de connaissances, mais plutôt de mettre à l'epreuve le simulateur face à un enseignant non initié à Prolog qui doit mettre au point un scénario de simulation, et face à un utilisateur non averti qui manipule les fonctionnalités de Promise.

Il va de soi que la suite de ces travaux doit aller vers une utilisation "de masse" de ce logiciel, sachant que la base de connaissances doit au fur et à mesure s'améliorer à l'aide de tests complémentaires.

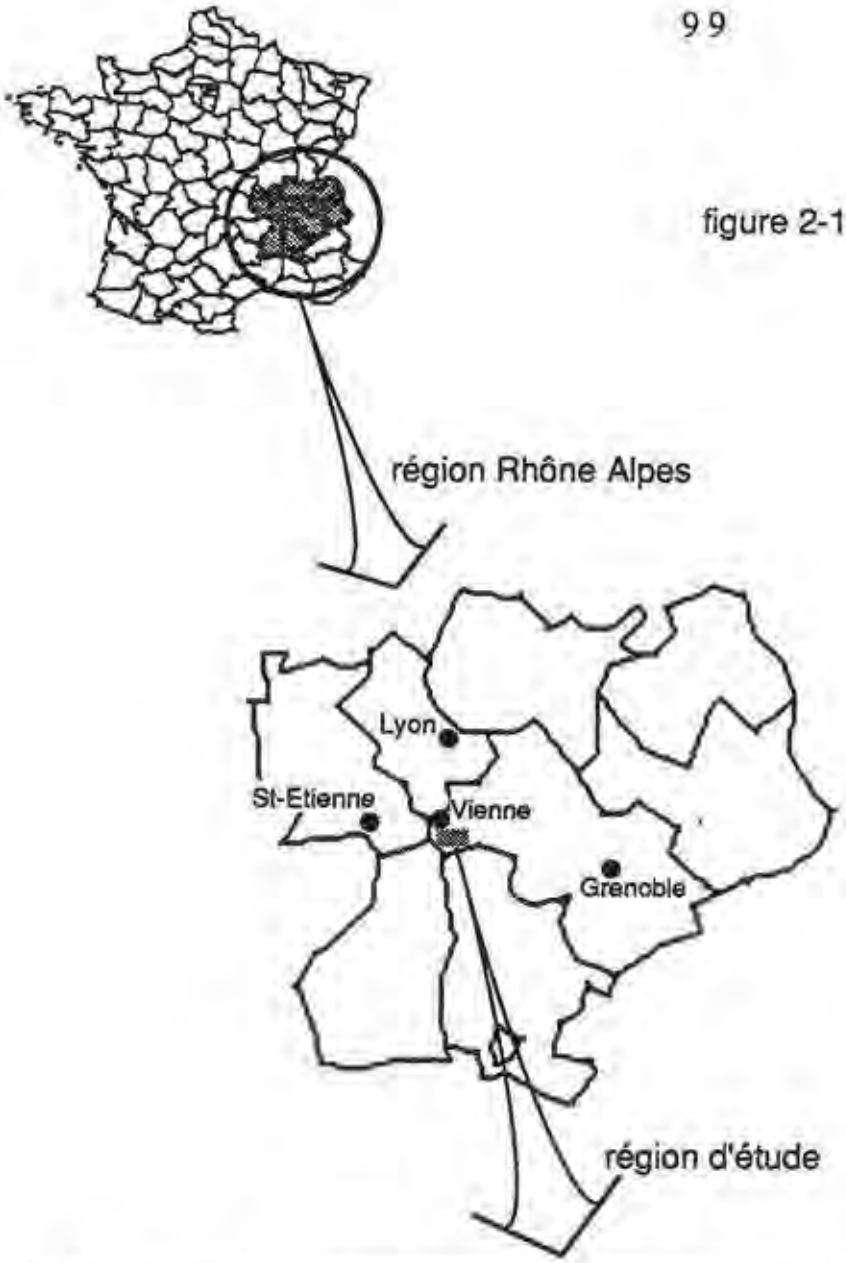
De plus, nous n'aborderons pas l'aspect du caractère opérationnel et de la validité de la simulation de projets dans le cadre de la formation, celui-ci ayant été largement développé et démontré par ailleurs [GRAILLOT 1983], [GRAILLOT 1986].

4.1 UN PROJET D'ASSAINISSEMENT EN MILIEU RURAL

Les deux projets qui ont auparavant été développés dans le cadre de Mise, traitent de l'adduction en eau potable et de l'irrigation (paragraphe 2.1). Aussi, choisir un projet d'assainissement vient compléter la panoplie des domaines abordés. L'assainissement en milieu rural, outre qu'il s'agit d'un problème d'actualité, offre l'avantage d'une bonne diversité de contextes (géologique, pédologique, social), de solutions possibles (assainissement collectif, autonome, individuel - voir la troisième partie de ce mémoire -) et des évènements possibles, favorisant de nombreux scénarios de simulation.

Le projet de référence retenu est celui de l'assainissement de la commune de VERNIOZ (1132 hectares) regroupant les bourgs de VERNIOZ et de Saint-ALBAN-de-VAREZE, dans le cadre de l'assainissement général de la vallée de la VAREZE dans l'ISERE (38) (figure 2-12). Cette étude est pilotée par le Service Régional d'Aménagement des Eaux (S.R.A.E.) RHONE-ALPES.

figure 2-12 : plan de situation de
la zone d'étude (Vernioz)



échelle = 1/18500

Extrait de la carte IGN 3033 EST (Vienne) au 1/25000

Notons que dans ce cas, il s'agit bien d'un projet de référence, c'est-à-dire dont on s'inspire pour élaborer une simulation à des fins pédagogiques. Aussi, la réalité pourra être modifiée pour répondre à certaines contraintes, pourvu qu'elle reste plausible. Cette référence fixe un noyau de données physiques et sociales. Libre ensuite au pédagogue de les adapter pour optimiser son action. Ceci est également vrai dans le cadre du piège à connaissances que peut constituer Promise : la sagacité du concepteur de la base de règles fera que les règles de conduite de projets à déduire seront plus ou moins généralisables selon qu'elles ont été expérimentées dans un plus ou moins grand nombre de situations, réelles ou fictives.

De plus, VERNIOZ n'est qu'une partie de l'aménagement régional que constitue celui de la VAREZE, qui comprend en tout une dizaine de bourgs. Il serait donc extrèmement intéressant d'observer dans quelle mesure la base de règles adoptée à VERNIOZ s'adapte aux autres localités. Les compléments ou modifications à apporter seraient révélateurs des points communs et des divergences entre plusieurs projets de même nature, et bénéfiques pour l'apprentissage de la conduite d'un projet d'assainissement en milieu rural.

4.2 CADRE DU PROJET

4.2.1 GEOLOGIE [CPGF 1988]

Le secteur d'étude, au sud-est de VIENNE, correspond aux contreforts les plus méridionaux des collines du BAS-DAUPHINE. Il s'agit du plateau de BONNEVAUX constitué essentiellement de terrains tertiaires : miocène et pliocène. Une grande partie du plateau est masquée par des formations quaternaires, pour la plupart d'origine glaciaire. Cet ensemble de collines, qui a subi un aplanissement généralisé, est fortement réentaillé par les vallées de la SANNE et de la VAREZE, de direction générale est-ouest.

On distingue dans les fonds de vallées des alluvions actuelles et des alluvions fluviatiles würmiennes, constituées par de petits galets de roches calcaires métamorphiques éruptives, emballés dans une matrice sableuse.

Les unités géologiques peuvent se diviser en 3 catégories :

- plateaux et pentes essentiellement argileux;
- formation à dominante gravelleuse dans la vallée et à proximité des rivières;

• dans les positions intermédiaires, moyenne vallée et pied de pente, formation sablo-argileuse représentées par les molasses ou les colluvions .

4.2.2 PEDOLOGIE

Directement liée à la géologie, le cadre pédologique du secteur correspond au schéma d'ensemble suivant :

• des sols acides, limoneux, hydromorphes sur les plateaux;

• des sols bruns plus ou moins lessivés sur les pentes et sur les alluvions quaternaires anciennes;

• des sols d'érosion à texture grossière dans les combes;

• des sols alluviaux hydromorphes dans les vallées.

4.2.3 HABITAT ET ECONOMIE

La population totale est de 700 habitants environ (avec un taux de croissance annuel moyen de 5.5% entre 1975 et 1982), dont 500 habitants et jusqu'à 550 à 600 habitants en période de pointe (maisons secondaires - hors camping -) dans les 2 bourgs de VERNIOZ et de Saint-ALBAN-de-VAREZE qui sont bien individualisés et distants d'à peu près 1.5 km . Ces 2 bourgs se caractérisent par un centre ville assez vieux et une expansion rapide de maisons individuelles et de lotissements en périphérie. Un camping de 600 places est ouvert l'été sur les bords de la VAREZE. Il n'y a pas d'industrie, l'activité professionnelle des habitants se répartissant grossièrement entre l'agriculture (15% de la population active) et diverses activités (les 2/3 de la population active travaille dans les principales villes environnantes, surtout VIENNE).

4.2.4 EAU ET ASSAINISSEMENT

L'alimentation en eau potable de la population se fait à partir du captage d'une source (source de NASSIN : 72 m^3/j) et de 2 puits de forages (Le CORTET : 49 m^3/j et Le MOULIN : 48 m^3/j). Le réseau de distribution totalise 23 km environ en PVC et en fonte, et 5 réservoirs. Le mode de facturation est de type binôme avec un forfait de 258F par an et 4.90F par m^3 consommé (en 1987).

400 habitants au total sont raccordés à un réseau d'assainissement sommaire, de type unitaire, dont les rejets s'effectuent directement dans des fossés rejoignant la VAREZE. Il reste environ 300 habitants qui doivent prendre en charge leur assainissement par des techniques d'épandage individuel ou autonome. Cependant, il semble que les quelques installations qui sont implantées ne donnent pas satisfaction. Cela pourrait s'expliquer par un mauvais choix de filière étant donné la faible perméabilité des terrains sur certaines parcelles.

Le camping possède sa propre station d'épuration, mais qui ne semble pas fonctionner normalement.

4.2.5 INTERET DE CE TYPE DE PROJET

On voit, à travers cette rapide description, que de nombreuses questions se posent pour s'attaquer au problème de l'assainissement :

- collectif en ville, mais faut-il regrouper les 2 bourgs ou bien traiter leurs eaux usées séparément ?
- comment définir la limite d'extension du réseau collectif, limite à partir de laquelle l'épuration se fera par des techniques individuelles et autonomes ?
- comment homogénéiser les tarifications entre les habitants des bourgs reliés au réseau et les autres ? La commune doit-elle endosser l'entretien des assainissements individuels ?
- pourquoi l'assainissement individuel ne convient-il pas bien sur les intallations existantes ?
- que faire de la station d'épuration du camping ?

Il y a donc un grand nombre de choix possibles, plus ou moins liés entre eux et dépendants de conditions physiques, sociales ou politiques.

On s'aperçoit, à travers ces quelques questions, de la difficulté d'optimiser une solution selon un critère qui permette de quantifier le projet d'aménagement dans sa globalité.

La seule façon de procéder est bien ici de travailler sur des scénarios d'aménagement qui vont constituer des hypothèses qui, même sans vouloir toutes les explorer, sont déjà nombreuses.

Un certain nombre de données physiques, hydrologiques et sociales ont été réunies pour alimenter ces scénarios.

4.3 OUTILS DE LA SIMULATION

Dans le cadre de la simulation test qui a été réalisée, un certains nombre d'outils informatiques ont été mis en œuvre. Ces outils ont été soit spécifiquement développés, soit importés d'autres applications.

Ces outils sont de différents types dont nous donnons ici quelques exemples :

OUTIL DE CALCUL

Simulation des écoulements dans les canalisations

Il s'agit du logiciel EAUSER développé par P. VOIGNIER [VOIGNIER 1989] à l'Ecole des Mines de Saint-ETIENNE. Ce logiciel, très complet, intègre des modules de saisie des réseaux, des pluies de projet et de l'ensemble des paramètres nécessaires à la simulation des écoulements (caractéristiques des bassins versants, équipements hydrauliques ...) et une batterie de modèles de propagation hydraulique plus ou moins complexes et rapides à l'exécution (régime permanent ou transitoire).

Aide au calcul des profils en long des canalisations d'assainissement

A partir de la topographie naturelle le long de la conduite à poser (que l'on doit communiquer point par point), de la profondeur minimale d'enfouissement et de la pente minimale désirée, PROFIL propose un profil en long de la conduite. Ce logiciel permet de visualiser ce profil grâce à une interface graphique et un filtre minimise les ruptures de pentes.

AIDE A LA DECISION

Aide au choix d'un assainissement individuel

Le fonctionnement de ce logiciel, Moïse, est développé dans la troisième partie de ce mémoire. A partir d'un certain nombre de caractéristiques physiques de la parcelle à équiper, Moïse propose une filière d'assainissement individuel, avec les précautions éventuelles qui accompagnent sa mise en œuvre. Un logiciel d'accompagnement, DIM, permet de dimensionner le dispositif.

Ces deux logiciels peuvent facilement être adaptés pour la constitution de deux nouveaux outils de choix et de dimensionnement d'un assainissement autonome (lagunage, épandage collectif ...).

BASE DE DONNEES

Simulation de sondages

A partir d'un maillage de la zone d'étude, sur laquelle on reconstitue en chaque point la nature du sous-sol (soit d'après les données disponibles, soit en introduisant des cas aptes à valoriser et nourir la simulation - diversité des scénarios d'un point de vue formation ou extraction de connaissances -), le programme PELLE restitue la nature des strates que l'on rencontre sur le premier mètre de sol, ou sur les 3 ou 4 premiers mètres selon qu'il s'agit d'un sondage manuel ou à la pelle mécanique.

Simulation des mesures de perméabilité

Comme pour la procédure précédente, le maillage de la carte pédologique dont les unités de sol sont corrélées avec un coefficient de perméabilité, permet une simulation de ces mesures (programme MUNTZ), avec l'introduction d'un aléa reproduisant les écarts importants pouvant apparaître pour 2 mesures géographiquement proches.

OUTIL SPECIFIQUE

Simulation du financement de l'étude (ou des travaux)

L'utilisateur ayant formulé un objectif du type "demande de financement auprès de l'organisme (i)", le logiciel "FINET", au vu de l'évaluation du coût de l'étude (ou des travaux), des financements déjà obtenus, et de paramètres inhérents à l'organisme (i), génère ce que cet organisme va accorder. Les fonctions de calcul sont déterminées à partir d'une équation paramétrée (en fonction du type d'organisme) auquel vient s'ajouter un aléa; on peut imaginer à ce niveau un système prenant en compte l'histoire du projet, à la manière du simulateur.

Il faut noter que Promise, dans sa forme actuelle, ne manie pas de données numériques - à part les coûts temporels et financiers de chaque objectif -. Aussi, dès qu'une donnée numérique doit intervenir, il faut le faire par l'intermédiaire d'une procédure. C'est par exemple le cas de l'évaluation de l'étude (ou des travaux) que

l'utilisateur doit estimer (procédure EVCOU). Stockée dans la base de faits, cette valeur est ensuite réutilisée par d'autres procédures.

L'appel des procédures étant automatisé, il y a malgré tout continuité pour l'utilisateur pour lequel ces transferts de logiciels sont transparents (aux différences de "look" près : aspect visuel des écrans, absence éventuelle d'uniformité au niveau des saisies ou des affichages ...).

4.4 EXEMPLES DE BASE DE CONNAISSANCES

Notre objectif dans ce paragraphe est de donner des exemples d'instanciation d'arbre décisionnel et de règles de simulation. Ces exemples ne forment pas une base de connaissances complète. En effet, il s'agit ici de montrer comment on peut représenter les éléments d'un projet dans la structure de Promise. Ecrire une base de connaissances complète d'un projet du domaine de l'eau (depuis la pré-étude jusqu'à l'exploitation des installations) représente un important investissement intellectuel et temporel.

Le système proposé facilite la mise en place d'une telle base de connaissances car la structure cellulaire de Promise autorise une croissance graduée de cette base par le concepteur qui augmente ainsi peu à peu les capacités du logiciel.

Les exemples que nous donnons peuvent servir de germes à partir desquels pourront se construire des bases de plus en plus élaborées.

4.4.1 ORGANISATION D'UNE BASE DE CONNAISSANCES

Un projet d'assainissement peut être décomposé en 5 phases au cours d'une simulation :

- phase pré-étude pendant laquelle l'utilisateur prend connaissance du projet, recueille des documents et des informations techniques ou législatives, et découvre les moyens qui sont à sa disposition (outils de calcul);

- phase étude qui doit aboutir à un schéma d'assainissement de la zone d'étude, ainsi qu'à des propositions de coûts;

- phase pré-travaux illustrée par la rédaction d'un appel d'offres et la recherche du financement des travaux;

- phase travaux où le projet d'assainissement est mis en œuvre (pose des canalisations, construction des dispositifs d'épuration ...);

- phase d'exploitation qui conclue le projet, s'étendant sur une durée indéterminée, où les qualités techniques et juridiques du projet réalisé peuvent être confrontées à des évènements (pluie de fréquence élevée, dispositifs d'épuration défaillants ...).

Chaque phase peut faire l'objet d'une base de connaissances, chacune d'elles, pour une structure identique, pouvant prendre en compte des réalisations issues des phases précédentes ou suivantes du projet.

De plus, on notera que la structure de cette version de Promise a été conçue en pensant surtout aux deux premières phases aboutissant à un schéma d'aménagement, avec le souci de rester à un niveau stratégique. C'est une des raisons pour lesquelles les fonctions numériques sont peu représentées actuellement dans Promise.

4.4.2 EXEMPLE D'ARBRE DECISIONNEL

La figure 2-13 représente un arbre décisionnel illustrant la phase pré-étude. Il va de soi que tout n'y est pas explicitement prévu mais il peut être complété facilement par le concepteur. En fait, toute remarque ne peut que l'enrichir.

Cet arbre est sensé réunir les actions possibles au cours d'une phase d'acquisition de données et de prise de connaissance d'un projet. Il s'agit là de la première difficulté pour le concepteur de la base de règles. Le vocabulaire à employer est notamment délicat à choisir et la base que nous présentons en est un exemple.

En effet, nous avons dû, pour définir les 2 objectifs suivants :

- évaluer, coût de l'étude
- financer, étude actuelle

employer 2 objets différents alors qu'on veut parler de la même chose : l'étude en question. Dans tous les cas, les imprécisions pourront être levées grâce aux fichiers d'informations.

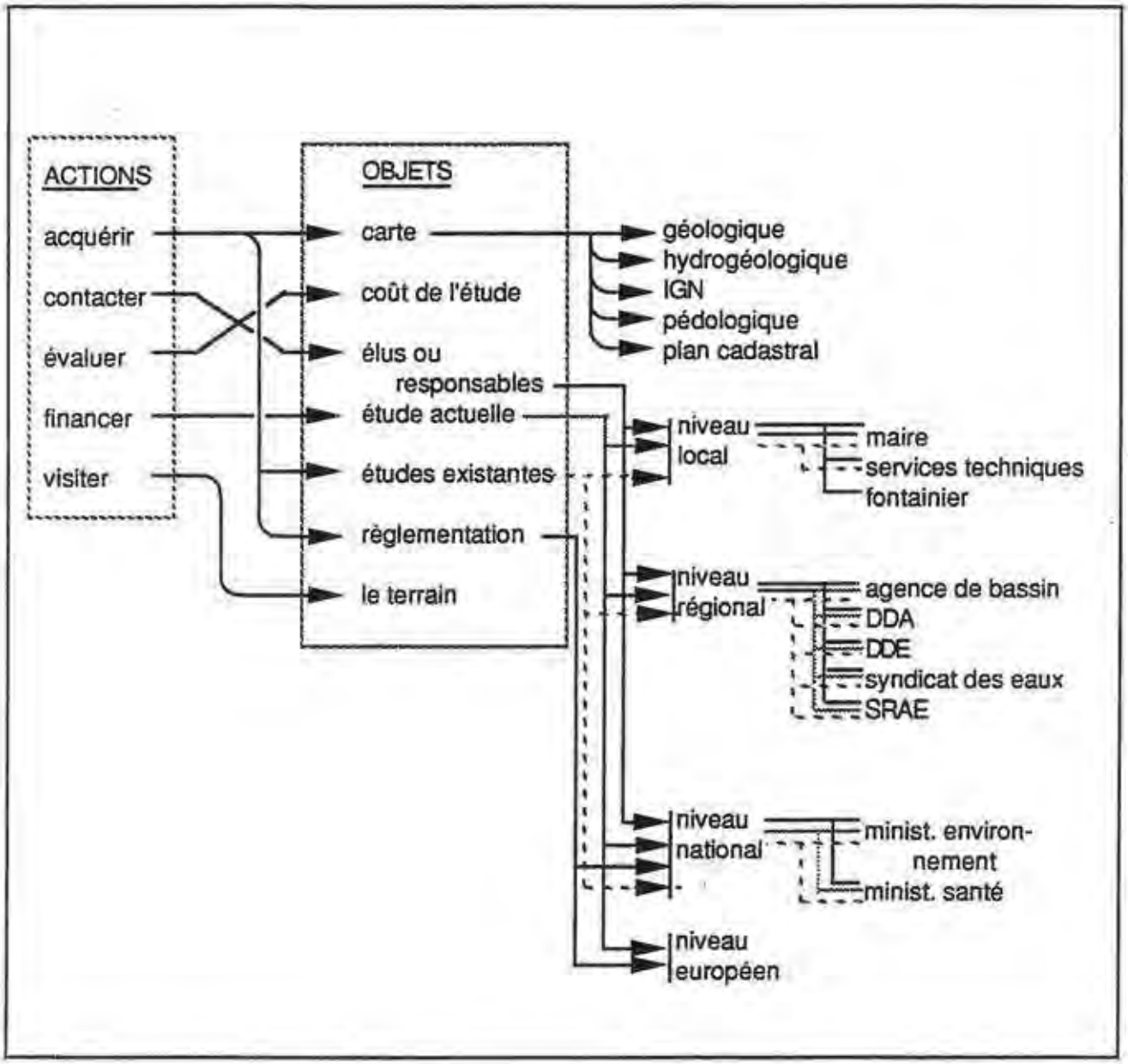


figure 2-13 : exemple d'arbre décisionnel de la phase pré-étude

Nous donnons dans le tableau 2-4 la représentation informatique de cet arbre dans Promise, sous forme de faits.

4.4.3 EXEMPLES DE REGLES DE SIMULATION

EXEMPLE 1

Ce premier exemple, même s'il ne présente pas un grand intérêt hydrologique, illustre comment on peut construire une règle, et combien ce travail est délicat. En effet, on se rend compte que chaque cellule Promise peut avoir une structure arborescente complexe.

En reprenant le schéma de l'arbre de décision 2-13, nous allons modéliser les réalisations correspondant à l'objectif : (contacter, élus ou responsables, au niveau local, le maire) qu'on simplifiera dans la suite du texte par "rencontrer le maire". De plus, on pourra supposer dans un premier temps que la réalisation de cet objectif est indépendante des autres actions possibles au cours du projet.

Soient 4 réalisations possibles et leurs probabilités associées :

- r_1 : en arrivant à votre rendez-vous, vous ne trouvez que la secrétaire du maire qui vous remet une lettre. M. le maire s'excuse de son absence en invoquant des affaires urgentes; il a essayé de vous joindre pour vous prévenir, mais sans succès. (r_1=absent, $prob_1$ = 0.20)
- r_2 : le maire est ravi de vous voir, et vous reçoit chaleureusement. Il vous communique, après une entrevue cordiale, les documents du P.O.S (Plan d'Occupation des Sols) en préparation (r_2=parfait, $prob_2$ = 0.30)
- r_3 : vous arrivez très en retard à votre rendez-vous. Votre entretien avec le maire est glacial. Pour obtenir le P.O.S (Plan d'Occupation des Sols) en préparation, vous devez formuler une demande écrite auprès de son secrétariat, ce qui rallonge d'autant les délais de votre étude (r_3=mauvais, $prob_3$ = 0.30)
- r_4 : votre rendez-vous avec le maire était à 10h. Celui-ci n'est arrivé qu'à 11h45. Vous vous êtes senti obligé de reporter la réunion à l'après-midi et de lui offrir un bon repas. Votre sacrifice n'aura pas été vain : vous y gagnez la

```
liste ([ action ], [acquérir, contacter, évaluer, financer, visiter ]).
liste ([ objet ], [carte, "coût de l'étude actuelle", "élus ou responsables",
étude actuelle" ,"documents existants" , règlementation" ,terrain ]).

liste ([ acquérir, carte], [géologique, hydrogéologique, "IGN", pédologique,
"plan cadastral" ]).

liste ([ acquérir, "documents existants" ], ["au niveau local", "au niveau
régional"]).
liste ([acquérir, "documents existants" , "au niveau local" ], ["auprès des
services techniques", "auprès du maire"]).
liste ([acquérir, "documents existants" , "au niveau régional" ], ["à l'agence
de bassin", "à la DDA", "à la DDE", "au SRAE", "au syndicat des eaux" ]).

liste ([acquérir, règlementation], ["au niveau national", "au niveau
européen"]).
liste ([acquérir, règlementation, "au niveau national"], ["ministère de
l'environnement", "ministère de la santé"]).

liste ([contacter, "élus ou responsables"], ["au niveau local", "au niveau
régional", "au niveau national"]).
liste ([contacter, "élus ou responsables", "au niveau local"], ["le chef des
services techniques", "le fontainier", "le maire"]).
liste ([contacter, "élus ou responsables", "au niveau régional], ["l'agence de
bassin", "la DDA", "la DDE", "le SRAE", "le président du syndicat des eaux"]).
liste ([contacter, "élus ou responsables", "au niveau national"], ["ministère
de l'environnement", "ministère de la santé"]).

liste ([financer, "étude actuelle"], ["au niveau local", "au niveau régional",
"au niveau national", "au niveau européen"]).
liste ([financer, "étude actuelle","au niveau régional"], ["agence de bassin",
"syndicat des eaux", "DDA", "DDE", "SRAE"]).
liste ([financer, "étude actuelle","au niveau national"], ["ministère de
l'environnement", "ministère de la santé"]).
```

tableau 2-4 : représentation informatique dans PROMISE
de l'arbre décisionnel de la figure 2-13

sympathie du maire et les documents du P.O.S. (Plan d'Occupation des Sols) en préparation (r_4=bon, $prob_3$ = 0.20)

Le classement sur l'axe des réalisations peut être :

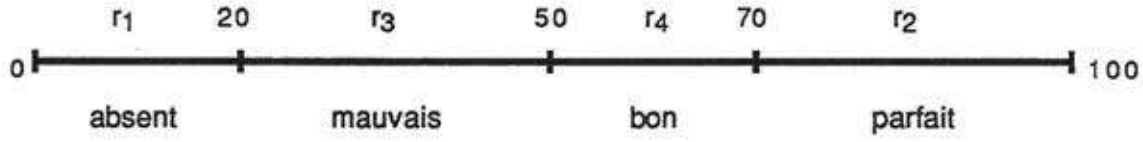


fig 2-14 : classement des réalisations de l'objectif : "rencontrer le maire"

A la première réalisation de l'objectif, on aura un tirage aléatoire permettant d'obtenir une réalisation parmi les 4 définies.

Si (r_1=absent) a été tirée, l'utilisateur va éventuellement retenter une visite au maire. En supposant que le maire prendra un peu plus de précautions pour ne pas être absent, la probabilité d'avoir à nouveau la réalisation (r_1=absent) va diminuer. Cela s'écrira :

condition 1

SI le maire était absent (r_1=absent) lors de votre dernière visite

ALORS c'est plutôt favorable car il prendra plus de précautions cette fois ci (pondération 15 par exemple)

L'utilisateur peut dans tous les cas retourner voir le maire et ce autant de fois qu'il le veut. On aura alors la suite de conditions à écrire :

condition 2

SI on a eu les réalisations (r_3=bon)
ou (r_4=parfait)

ALORS la réalisation sera : le maire vous répond au téléphone qu'il n'a pas d'autres documents à vous communiquer (r_8=rien_de_plus).

condition 3

SI on a eu la réalisation (r_2=mauvais)

ALORS la réalisation sera : le maire vous fait comprendre au téléphone que vous insistez un peu lourdement. C'est vrai qu'il ne vous porte pas dans son cœur à la suite de votre visite. Il vous répond évasivement qu'il n'a pas d'autre document à vous communiquer (r_5=encore!).

condition $_4$

SI on a eu les réalisations (r_3=mauvais)
et (r_4=encore!)
ALORS la réalisation sera : le maire vous raccroche le téléphone au nez (r_6=crac).

condition $_5$

SI on a eu la réalisation (r_3=crac)
ALORS la réalisation sera : le maire ne répond plus (r_7=plus_rien).

Il faudra classer ces conditions pour l'écriture informatique. En effet, il est préférable de faire intervenir d'abord les conditions conduisant à une réalisation obligatoire (les 4 dernières citées).

En reprenant la règle générique du paragraphe (3.3.1.3), on peut écrire une cellule complète comme sur le tableau 2-5.

Selon le même principe, on pourra faire intervenir d'autres objectifs passés qui peuvent influencer la réalisation de la rencontre avec le maire : la rencontre avec le chef des services techniques ou autre.

Aussi anodin que puisse paraître cet exemple, on notera que ses conséquences peuvent se répercuter très loin dans la simulation du projet. En effet, on pourra représenter l'enchaînement suivant :

SI la rencontre avec le maire a été mauvaise
ALORS il ne vous accompagnera pas sur le terrain et
vous passerez à coté d'informations intéressantes
ALORS vous pourrez avoir des problèmes avec les administrés au moment
de l'implantation des assainissements individuels ...

On s'aperçoit immédiatement, à travers cet exemple et son instanciation, que la mise au point logique d'une règle est un exercice difficile. Nous ne proposons qu'un outil d'aide au développement informatique (logiciel STB) mais qui permet déjà d'évacuer ce

```
                                                    1ère partie
description(                                        |
  [ renconter le maire ],                           |     objectif en question
  renc_mai,                                         |     nom du fichier DOS d'infos
  3,                                                |     délai temporel en jours
  0.5,                                              |     imputation temporelle en jours
  0,                                                |     coût financier négligeable
                                                    2ème partie
[                                                   |
 l_et_obli ( [                                      |     condition 5
  l_ch( [ rencontrer le maire ] ), ch( crac ),      |
        p(15000),p(0)]),                            |
 l_et_obli ( [                                      |     condition 4
  l_ch( [ rencontrer le maire ] ), ch(mauvais),     |
  l_ch( [ rencontrer le maire ] ), ch(encore!),     |
        p(16000),p(0)]),                            |
 l_et_obli ( [                                      |     condition 3
  l_ch( [ rencontrer le maire ] ), ch(mauvais),     |
        p(17000),p(0)]),                            |
 l_et_obli ( [                                      |     condition 2
  l_ch( [ rencontrer le maire ] ), ch( bon ),       |
        p(18000),p(0)]),                            |
 l_et_obli ( [                                      |     condition 2 (suite)
  l_ch( [ rencontrer le maire ] ), ch( parfait ),   |
        p(18000),p(0)]),                            |
 l_ou_influ ( [                                     |     condition 1
  l_ch( [ rencontrer le maire ] ), ch( absent ),    |
        p(15),p(0)]),                               |
  ],                                                |

                                                    3ème partie
[ p( 0 ), p( 20 ),                                  |     entre les bornes 0 et 20
  p( 0 ), p( 0 ), p( 0 ),                           |     influence sur les coûts
  ch(fic_r1),                                       |     nom du fichier à afficher
  ch( absent),                                      |     qualificatif de la réalisation
                                                                |
  p(20),  p(40),  p(2),p(0),p(0),  ch(fic_r3),ch(mauvais),      | autres bornes
  p(40),  p(70),  p(0),p(0.5),p(0.5),  ch(fic_r4),ch(bon),      |     "
  p(70),  p(100),  p(0),p(0),p(0),  ch(fic_r2),ch(parfait),     |     "
  p(15000),  p(15000),  p(-3),p(0),p(0),  ch(fic_r7),ch(plus_rien),
  p(16000),  p(16000),  p(-3),p(0),p(0),  ch(fic_r6),ch(crac),
  p(17000),  p(17000),  p(-3),p(0),p(0),  ch(fic_r5),ch(encore!),
  p(18000),  p(18000),  p(-3),p(0),p(0),  ch(fic_r8),ch(rien_de_plus),
                                                                |
  ]).                                                           |  fin de la règle
```

tableau 2-5 : présentation informatique
d'une règle de simulation

que l'on pourrait nommer "les problèmes d'intendance" (syntaxe Prolog, orthographe des objectifs, faisabilité informatique des règles -existence de tous les fichiers nécessaires-).

EXEMPLE 2

Cet exemple tente de montrer comment on peut contrôler l'enchaînement des décisions et bloquer l'utilisateur lorsqu'il fait fausse route.

En reprenant l'arbre élaboré pour la phase pré-étude, écrivons la condition :

SI il n'y a pas une évaluation correcte du coût de l'étude
ALORS il ne peut pas y avoir succès dans la recherche de financement de l'étude.

Etant donné que l'évaluation du coût de l'étude est une valeur numérique, celle-ci sera saisie dans une procédure qui fournira en même temps une aide pour cette estimation (accès aux objectifs possibles et à leurs fichiers d'informations de la phase "étude" par exemple). C'est à cette procédure qu'il incombera de juger l'évaluation comme correcte ou non-correcte puisque, rappelons-le, le simulateur ne sait pas manier les nombres hormis les coûts temporels et financiers. De même, le financement s'obtient par l'intermédiaire d'une procédure (paragraphe 4.3 procédure FINET).

Supposons donc que 2 réponses peuvent être obtenues de la procédure d'évaluation du coût de l'étude :

évaluation du coût de l'étude - - - > correcte
évaluation du coût de l'étude - - - > non_correcte

Dans une syntaxe proche de celle utilisée dans Promise, cela donnera :

SI obj("évaluer","coût de l'étude actuelle") réal("correcte")
ALORS activer la procédure de financement "FINET"
SINON afficher message explicatif

L'écriture informatique sera telle que dans le tableau 2-6.

Le cas précédent représente la configuration minimale. On peut l'étoffer de multiples façons. Nous en présentons une ici.

```
                                        1ère partie
description(                                |
  [ "évaluer","coût de l'étude actuelle"],  |     objectif en question
  éval_cou,                                 |     nom du fichier DOS d'infos
  5, 2, 0,                                  |     coûts prévus
                                                  2ème partie
[                                                   |
 l_ou_influ ( [
  l_ch( [ "évaluer","coût de l'étude actuelle"] ), ch( correcte ), p(50),p(50)]),
],                                                  |
                                                  3ème partie
[ p( 0 ), p( 50 ), p( -5 ), p( -2 ), p( 0 ), ch(fic_r1 ),  ch(rien),
  p(50), p(100), p(0),p(0),p(0), ch(pfinet ),ch( ok ),
  ]).                                               |  fin de la règle
```

{ fic_r_1 est le fichier contenant le message explicatif de l'échec de l'activation}

tableau 2-6 : exemple de règle de simulation contrôlant l'enchaînement des décisions de l'utilisateur

```
                                        1ère partie
description(                                |
  [ "évaluer","coût de l'étude actuelle"],  |     objectif en question
  éval_cou,                                 |     nom du fichier DOS d'infos
  5, 2, 0,                                  |     coûts prévus
                                                  2ème partie
[                                                   |
 l_et_obli ( [                                      |     cond. 4
  l_ch( [ "évaluer","coût de l'étude actuelle"] ), ch( trop_forte_1_fois ),
  l_ch( [ "évaluer","coût de l'étude actuelle"] ), ch( correcte ),
        p(15000),p(0)]),                            |
  l_et_obli ( [                                     |     cond. 3
  l_ch( [ "évaluer","coût de l'étude actuelle"] ), ch( trop_forte_1_fois ),
  l_ch( [ "évaluer","coût de l'étude actuelle"] ), ch( trop_forte ),
        p(16000),p(0)]),                            |
  l_et_obli ( [                                     |     cond. 2
  l_ch( [ "évaluer","coût de l'étude actuelle"] ), ch( trop_forte ),
        p(17000),p(0)]),                            |
l_ou_influ ( [                                      |     cond. 1
  l_ch( [ "évaluer","coût de l'étude actuelle"] ), ch( correcte ),
        p(0),p(0)]),                                |
],                                                  |
                                                  3ème partie
                                                    |
[ p( 0 ), p( 100 ), p( 0 ), p( 0), p( 0 ), ch(pfinet ), ch(ok_pour_financer),
  p(15000), p(15000), p(-3),p(-1),p(0), ch(qpfinet),ch(ok_pour_financer),
  p(16000), p(16000), p(-3),p(-1),p(0), ch(fic_r6),ch(trop_forte_2_fois),
  p(17000), p(17000), p(-3),p(-1),p(0), ch(fic_r5),ch(trop_forte_1_fois),

  ]).                                               |  fin de la règle
```

tableau 2-7 : exemple de règle de simulation contrôlant l'enchaînement des décisions de l'utilisateur avec une question

Faisons en sorte que la procédure d'évaluation du cout de l'étude puisse délivrer 3 types d'information :

évaluation du coût de l'étude - - - > trop forte (r_1)
évaluation du coût de l'étude - - - > correcte (r_2)
évaluation du coût de l'étude - - - > trop faible (r_3)

Le simulateur étudiera une première condition similaire au cas précédent :

condition 1
SI l'évaluation est correcte (r_2)
ALORS activer la procédure de financement "FINET" (r_4 = ok_pour_financer)

Dans le cas où l'estimation est trop forte, on peut proposer les conditions suivantes (sachant qu'on aura des conditions symétriques pour une estimation trop faible) :

condition 2
SI évaluation est trop forte (r_1)
ALORS message : votre évaluation est trop forte (r_5 = trop_forte_1_fois)

condition 3
SI on a déjà eu une évaluation trop forte une fois(r_5)
et l'évaluation est encore trop forte (r_1)
ALORS message : votre évaluation est trop forte ; elle doit se situer entre 100 et 200 kf pour être acceptable (r_6 = trop_forte_2_fois)

condition 4
SI on a déjà eu une évaluation trop forte une fois(r_5)
et l'évaluation est correcte (r_1)
ALORS
question : vous avez été amené à revoir votre évaluation du coût de l'étude à la baisse ... Quelle partie de l'étude doit en pâtir ? Pourquoi ? justifiez-vous.
puis activer la procédure de financement "FINET"

On a donc là un exemple de question qui peut recouvrir plusieurs objectifs :
- faire en sorte que l'utilisateur réfléchisse bien à ce qu'il entreprend;

- avoir un contrôle de la part de l'enseignant sur les raisonnements de l'apprenant;
- si l'utilisateur est un expert, essayer d'en retirer des règles de décisions.

L'écriture informatique sera telle que dans le tableau 2-7.

EXEMPLE 3

C'est plutôt un contre-exemple que nous allons développer afin de montrer une des limites actuelles du formalisme des règles de simulation, et comment on peut s'en affranchir éventuellement.

soit la règle à programmer :

SI un assainissement individuel de type épandage en tranchées a été choisi aux coordonnées (X,Y)
ET aucune mesure de perméabilité n'y a été effectuée
ET c'est une zone de faible perméabilité
ALORS c'est très défavorable pour le bon fonctionnement du dispositif

Promise ne sait pas manipuler de valeurs numériques, comme nous l'avons déjà dit, et donc encore moins les bases de données. Aussi, la seule solution actuellement envisageable consiste à reporter ce test dans une procédure (véritable sous-système Promise) capable de manipuler ce type de variables (à l'image du schéma 2-15). Ceci suppose qu'ont été mis au point les sous-programmes d'enregistrement, de stockage et d'organisation des bases de données.

La procédure -sous système Promise- prendrait donc en charge des fonctions qui incombent au simulateur, ce qui est préjudiciable au contrôle permanent que celui-ci doit exercer. A terme, une fonction de Promise peut prétendre gérer ce type de conditions, tout comme il y a les fonctions l_et_influ, l_ou_influ ...

L'écriture de ces fonctions est pour l'instant prématurée. Nous pensons que 2 étapes doivent auparavant être franchies :

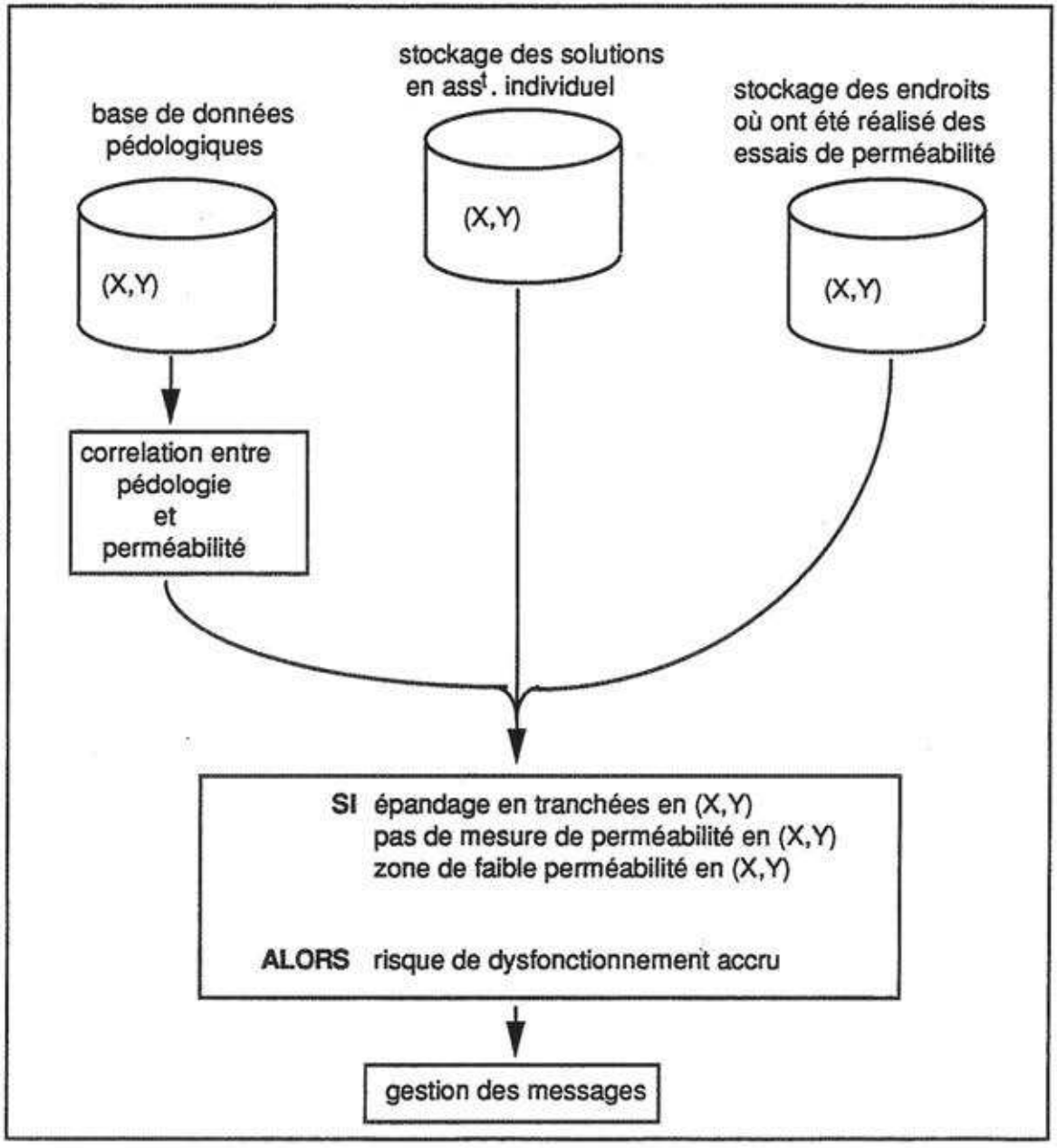


figure 2-15 : mise en œuvre d'une procédure
testant une condition

• réaliser un certain nombre de simulations avec Promise pour s'assurer de son opportunité, de sa robustesse et de son efficacité (malgré ses possibilités actuellement limitées);

• réaliser l'éditeur de règles qui doit permettre d'organiser sa base de connaissances en échappant aux problèmes de syntaxe, de compilation et autres, pour se consacrer pleinement à la logique de la simulation. On obtiendra alors ce que l'on peut appeler un G.S.S. (Générateur de Scénarios de Simulation) dont le formalisme des règles sera bien adapté au problème de la simulation.

5. TEST DU SIMULATEUR

Nous exposons brièvement dans ce paragraphe les 2 premiers tests effectués sur Promise concernant sa maniabilité et son caractère opérationnel. Ils ont été effectués à l'Ecole des Mines de St-ETIENNE.

5.1 ACCESSIBILITE DE LA BASE DE REGLES

Cette étape consiste à confier à une personne profane en Prolog la constitution d'une base de règles afin d'observer si sa structure actuelle, sans éditeur de règles, est suffisamment accessible pour envisager de façon simple sa mise au point et sa maintenance .

Cette mission a été confiée à P. DAVOINE, professeur des sciences de l'eau à l'école des Mines, qui est l'utilisateur final de Promise dans le cadre de son enseignement.

Après une formation rapide sur le logiciel, il a fallu environ une semaine pour mettre au point une base de règles incluant la phase pré-étude complète, et une base sommaire sur la phase étude, ce qui permet de mener un projet jusqu'au stade du schéma d'assainissement. Il va de soi que les procédures à utiliser dans le cadre de la simulation sont mises au point par ailleurs.

Ce délai a été jugé convenable étant donné qu'il s'agit d'une première prise de contact.

5.2 TEST DE SIMULATION

C'est la base de règles créée auparavant qui a été utilisée pour un test de simulation avec un apprenant. Le caractère opérationnel global du système a pu être vérifié, notamment en ce qui concerne la manipulation des différentes fonctions de Promise (informations -très utilisées-, historique ...).

Mis à part quelques défauts de fonctionnement mineurs, plusieurs aspects recherchés dans la simulation semblent atteints :

- vie du projet par des aléas pertinents;
- blocage de certaines étapes obligeant l'apprenant à rassembler tous les éléments en sa possession avant d'avancer (fonction de contrôle);
- démarche plus planifiée de l'utilisateur qui se rend vite compte que le simulateur réagit négativement à des actions désordonnées.

Les prochaines simulations qui devraient se faire avec des groupes d'apprenants permettront de mieux cerner les avantages et les défauts de Promise.

5.3 L'ASPECT BASE DE CONNAISSANCES

Répétons que ces tests avaient pour objectif de mettre à l'épreuve la structure du simulateur.

On peut cependant légitimement s'interroger sur la validité d'une base de connaissances; c'est-à-dire se demander dans quelle mesure elle reflète une réalité plausible.

Un certain nombre d'éléments nous permettent d'espérer que c'est le cas; tout au moins, nous pensons que le formalisme de règles adopté permet de restituer une partie de la réalité. Les liens que nous gérons sont relativement simples, et ce premier travail laisse entrevoir des difficultés face au nombre important de connections entre les éléments d'un projet.

Quelles pourraient être les raisons d'un échec dans la conception d'une base de connaissances ? Nous en voyons deux possibles :

- un formalisme incomplet : c'est ce que nous laissions entrevoir en parlant de compléter notre formalisme par des fonctions numériques, des fonctions d'exploration plus complexes (combinaisons de "et" et de "ou"), des fonctions mixtes ... Dans ce cas, nous pensons être sur la bonne voie et pouvoir continuer à travailler dans cette direction.

- une impossibilité physique (informatique et/ou logique) face à la complexité du phénomène "conduite de projet". Cela pourrait se traduire par des contradictions, des incohérences non maîtrisables dans la base de règles, et la non-représentativité d'interférences primordiales dans les évènements gérés par le simulateur : bref, on ne parvient pas à simuler la réalité de façon plausible.

Cette dernière constatation amènerait d'ailleurs à une réfexion plus générale : si l'on n'arrive pas à mettre au point des règles de simulation, établies dans un contexte limité et un cadre déterminé puisqu'en référence à un projet précis, on peut alors avoir des doutes fondés à propos de l'écriture de règles de décision censées piloter la conduite d'un projet. En ce sens, Promise peut être considéré comme une étape vers la formalisation des règles de décision d'une station de travail.

6. CONCLUSION

La modélisation des actions et décisions élaborées qui interviennent dans le cadre d'un projet est ardue. A l'image des 2 approches de l'intelligence artificielle (paragraphe 1.2.2), 2 stratégies sont possibles (et ne s'excluent nullement) :

- l'approche d'en haut, qui va analyser globalement un projet pour tenter d'identifier des règles de décisions, en hypothéquant sur leur formalisation;

- l'approche d'en bas, qui va consister à tenter l'écriture concrète du déroulement d'un projet, en hypothéquant sur une généralisation des règles de la simulation.

L'essentiel du problème consiste finalement "à joindre les deux bouts", à condition que ceux-ci soient suffisamment maîtrisés.

Notre contribution s'attaque à la deuxième approche à travers l'écriture d'un logiciel de simulation de projet dans lequel on va effectivement essayer de gérer les liens entre les évènements qui s'y produisent.

Une de nos préoccupations majeures dans ce travail fut de penser à la maintenance d'un tel système. Pour cela, Prolog offre une structure tri-partite extrèmement intéressante qui va permettre d'isoler la logique de la simulation dans un fichier. La structure que nous avons imprimée à ce fichier - qu'on pourrait dénommer "arbustive" - est une série de cellules élémentaires - les arbustes - qui sont reliés entre eux par la nature des évènements induits dans le projet. La formalisation des règles, adaptée à la simulation, autorise une exploration du passé qui va influencer ou imposer la réalisation des objectifs que l'on entreprend.

La construction complète d'une base de règles est un délicat et long travail, tant informatique (nous proposons pour l'instant un logiciel de contrôle syntaxique et orthographique) que logique (il serait intéressant de développer un éditeur de règles).

On remarquera cependant que la structure de Promise, et surtout de sa base de connaissances à travers la formalisation des règles qui a été retenue, est "compatible Mise". C'est-à-dire qu'une première étape dans l'élaboration de bases de règles serait peut-être de reprendre les scénarios de Mise pour les réinjecter dans Promise, la mise en place des liens entre les évènements du projet pouvant se faire dans une deuxième étape.

L'utilisation de notre travail dans un outil de conduite de projet - style station de travail - n'est pas immédiat puisque le formalisme des règles est dédié au problème de la simulation.

Il y a cependant un certains nombre d'idées à en retenir. Chaque règles de simulation est un puits de règles de décisions, car élaborée par un expert à partir de déductions du domaine des heuristiques : ce sont bien ces heuristiques qui sont intéressantes en premier lieu dans un système expert, elles qui sont le plus difficile à repérer et à écrire.

Nous ne proposons pas de modèle de formalisation des règles de décisions, mais nous avons expérimenté une structure générale du logiciel à partir de deux étages :

- 1 étage tri-partite (qui est un apport de Prolog);
- 1 étage "arbustif" de la base de règles,

qui doit pouvoir être reproduite dans un logiciel de conduite de projet, au moins pour une première version dont la maintenance doit être optimisée. Ce logiciel pourrait s'intéresser tout d'abord à l'aspect avant-projet : en effet, il existe à ce niveau un "noyau stratégique" commun à de nombreux projets du domaine de l'eau, alors qu'à un niveau plus avancé, les bases de données (notamment numériques), spécifiques à chaque projet, prennent de plus en plus d'importance et posent le problème de leur existence même et de leur manipulation.

Par contre, une phase pré-étude peut-être organisée de façon à répondre intelligemment aux question de l'utilisateur et à contrôler sa démarche : c'est déjà ce que fait Promise quand il vérifie l'enchaînement des étapes et informe (parfois "incognito" sous forme d'une réalisation) l'utilisateur sur les documents qu'il doit se procurer et les thèmes qu'il doit approfondir.

L'approche que nous avons développée est à comparer à un type de logiciels qui fleurissent actuellement sur le marché : les P.G.P. (Progiciel de Gestion de Projets) et plus précisémment, les P.S.G.P. (Petits Systèmes de gestion de Projets), ces derniers étant adaptés au micro-ordinateur.

Les P.S.G.P. sont essentiellement basés sur des gestions de plannings à partir de méthodes telles que les diagrammes de GRANTT ou le P.E.R.T. (Program Evaluation and Review Technique), avec parfois une structuration préalable comme la méthode W.B.S. (Work Breakdown Structure) . D'autres modélisations essaient d'inclure une gestion en temps réel des projets, à l'image du M.P.S.P. (Modèle pour le Pilotage Structuré de Projets) [LARDERA 1989].

Le but que poursuivent les P.S.G.P. et Promise est identique : maîtriser le déroulement d'un projet, en essayant d'y inclure toutes les phases et de les agencer. Promise, à son désavantage, ne possède pas de module de gestion de planning et ne tient donc pas compte, par exemple, des possibilités de chevauchements de tâches. A contrario, et c'est ce qui en fait son originalité, Promise s'attache à traiter de la *qualité du résultat des tâches entreprises*, et à en tirer les conséquences pour la suite du projet. Cet aspect est bien évidemment plus facile à développer pour une simulation

pour laquelle on envisage quelques scénarios, que pour un outil de pilotage qui doit être capable d'en considérer une multitude. C'est pourquoi nous avons commencé par cette étape.

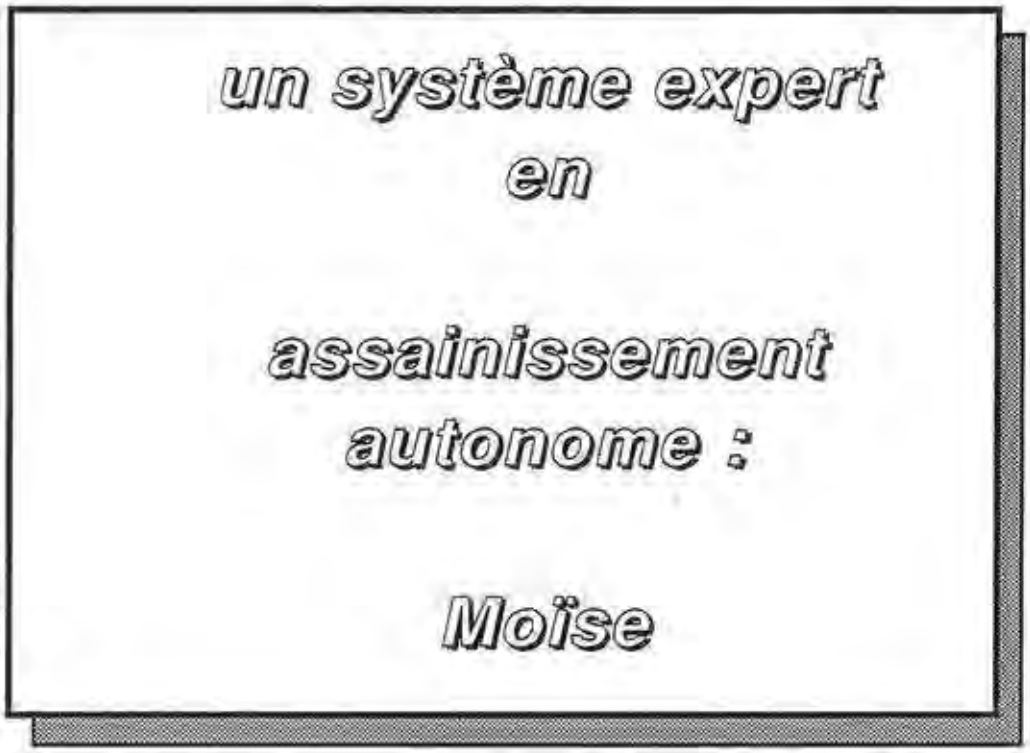
Par contre, nous pensons que les S.I.A.D. (Système Interactif d'Aide à la Décision) [LEVINE, POMEROL 1989] [MIRET 1989] montrent, dans leur philosophie et leur approche informatique, des analogies plus probantes avec Promise, la différence majeure étant que, répétons-le, Promise est dédié à la simulation.

Un S.I.A.D. offre un ensemble d'outils de visualisation graphique, de simulation, d'optimisation, d'analyse, de comparaison, de stockage, qui concourent à aider un spécialiste dans sa résolution de problèmes complexes, généralement multicritères.

Les phases décisionnelles - heuristiques - sont totalement abandonnées aux mains du pilote qui, après essais de plusieurs stratégies et estimation de leurs conséquences (estimation facilitée et accélérée par l'emploi du S.I.A.D.), devra prendre une décision.

Cependant, de plus en plus, les concepteurs de S.I.A.D. introduisent des outils intelligents (pour aboutir aux "S.I.A.D. intelligents") permettant à la machine d'analyser des situations de plus en plus complexes. Il va de soi que la décision finale appartient toujours à l'expert, mais celui-ci est mieux armé puisque disposant d'outils de plus en plus fins, capables de résoudre des problèmes de moins en moins structurés.

Outil à vocation pédagogique, Promise se rapproche d'un S.I.A.D. dans la mesure où il est un système interactif et où il sait gérer l'activation et éventuellement l'agencement entre différents outils (la gestion de l'aspect numérique, directement par le système, restant assurément à approfondir).


un système expert
en
assainissement
autonome :
Moïse

RESUME DE L'APPLICATION

Moïse est un système expert d'aide au choix d'un assainissement individuel.

A ce titre, c'est un des outils utilisés dans le simulateur de projets d'assainissement en milieu rural "Promise".

Le développement de Moïse nous a conduit à une importante réflexion sur la conception de ce type d'outils à partir de la technologie des systèmes experts.

En effet, l'expertise dans le domaine de l'assainissement individuel est très diffuse géographiquement (la règlementation officielle restant difficile à stabiliser).

Aussi, la base de connaissances du système expert doit être facilement adaptable, modifiable et compréhensible (à chaque expert son expertise et sa base de connaissances).

C'est un des buts que nous avons poursuivi en développant une structure que nous disons "en haies" à comparer à la structure classique "en arbre" d'un système expert.

La base de connaissances que nous avons créée (à partir d'une étude bibliographique et de conversations avec des spécialistes du domaine) comprend 75 règles aboutissant à 14 solutions.

Le moteur de Moïse, écrit en Prolog, fonctionne en mode déduction, ou en mode vérification. Certaines fonctionnalités permettent de modifier la base de faits et de tester la sensibilité d'une filière par rapport aux paramètres.

A partir de la technologie précédente, nous avons développé un module de dimensionnement d'un assainissement individuel, et nous montrons comment des modules de maintenance d'un dispositif (détection de la cause d'un mauvais fonctionnement) et de cohérence de la base de faits (détection de faits contradictoires communiqués par l'utilisateur) peuvent-être réalisés.

Une généralisation de la technologie et du moteur Moïse est testée dans un autre domaine : choix d'instrumentation pour analyse chimique (d'eau ou autre substance) pour les chimistes de l'école des Mines en contact avec un industriel du domaine. Les différences de fonctionnement requises pour chaque problème (choix d'un assainissement individuel et choix d'une instrumentation chimique) sont analysées.

Une version Minitel de Moïse a été développée; nous en discutons les résultats et les avantages.

PROLOGUE

Promise, simulateur de projet objet de la deuxième partie de ce mémoire, était un outil censé "ausculter" l'aspect stratégique du déroulement d'un projet. Dans cette troisième partie, on va s'intéresser à un niveau plus tactique à travers un outil de diagnostic, Moïse, aidant au choix d'un assainissement individuel. Là encore, Prolog va être utilisé et nous verrons qu'il permet à nouveau un partage des tâches algorithmiques (le moteur) et logiques (la base de règles).

Moïse est donc l'un des outils activés par Promise pour la simulation d'un projet d'assainissement en milieu rural.

Le développement de Moïse se justifie-t-il ailleurs que dans le cadre de la simulation ? La réponse sera positive si l'outil trouve son utlité auprès des acteurs de l'assainissement individuel : professionnels (entrepreneurs, lotisseurs, ingénieurs sanitaires) et/ou particuliers. Deux conditions apparaissent :

- que l'outil informatique puisse apporter une aide efficace dans la mesure où l'assainissement individuel est mal maîtrisé par une catégorie d'utilisateurs;
- que cette aide soit opérationnelle, c'est-à-dire que cet outil soit accessible à ces mêmes utilisateurs; sinon, il restera un "beau jouet".

On voit donc poindre une interférence entre la simulation, qui peut se contenter d'un aspect opérationnel limité, et la réalité, qui va imposer certaines contraintes au logiciel en ce qui concerne la façon dont l'outil est perçu (apporte-t-il vraiment un plus, une aide, voire une formation) et dont il peut être diffusé.

On verra qu'en distinguant un type de professionnels "initiés", on mettra l'accent sur la maintenance des règles de choix et sur la façon d'intégrer d'autres modules de connaissances, et un type de professionnels "non initiés", on s'attachera à l'aspect convivial et au problème de la diffusion à travers une version 'Minitel' de Moïse.

1. L'ASSAINISSEMENT DES EAUX USEES

1.1 DEFINITION ET TYPES D'ASSAINISSEMENT

On peut définir l'assainissement des eaux usées à travers ses objectifs et ses fonctions [ROUHART 1986] :

- préserver la santé des individus en collectant les eaux usées afin d'éviter la dissémination de composés nocifs, de germes dangeureux;
- sauvegarder l'équilibre écologique du milieu naturel (eaux souterraines, eaux de surface) en traitant les eaux avant leur rejet;
- éliminer les risques de nuisances en éloignant si nécessaire les eaux usées des habitations.

De ces différents aspects, sont nées des techniques selon le mode de vie des communautés :

- en milieu urbain, historiquement le premier assaini, on a construit des réseaux de collecte de plus en plus complexes et étendus, aboutissant à une station d'épuration de type biologique le plus souvent. On notera quand même qu'une ville comme Marseille rejetait ses eaux usées sans traitement en Méditerranée jusqu'à il y a encore quelques années.
- en milieu rural, qui concerne 90% des communes en France (celles dont la population est inférieure à 2000 habitants), le premier réflexe fut de transposer la technique urbaine {réseau + station}. Ainsi, sur les 15 millions d'habitants ruraux, on estime que moins du tiers sont desservis par un réseau public d'assainissement [Ministère de l'intérieur et de la décentralisation 1982]. Quant aux communes restantes, elles ne pourront jamais investir les 14 milliards de francs (valeur 1982) nécessaires à la mise en place d'un assainissement collectif de type urbain.

Aussi, ont été remises au goût du jour des techniques dites rustiques, qui n'en demeurent pas moins efficaces. Ce sont les techniques de lagunage et d'épandage dans le sol.

Le lagunage, introduit tout d'abord en France pour répondre au problème des communes soumises à une variation importante de population (tourisme), consiste à envoyer les eaux usées dans des bassins où elles sont épurées sous l'action de phénomènes biologiques naturels. A l'heure actuelle, le lagunage est de plus en plus mis en œuvre pour de petites collectivités car si les coûts d'investissement sont très légèrement inférieurs à ceux d'une station d'épuration, les dépenses d'exploitation sont nettement plus faibles.

Cependant, cette technique, par la mise en place d'un réseau de collecte et d'une "station" de lagunage se rapproche dans sa philosophie d'un assainissement de type collectif, mais d'extension moindre. Dans la majorité des cas, c'est la commune qui va prendre en charge la mise en place et la maintenance de l'ensemble en demandant une taxe d'assainissement à ses administrés.

Dans tous les cas précédents, on parle d'assainissement collectif.

Par contre, le coût de la gestion et de l'installation d'un dispositif concernant un nombre réduit d'habitations peuvent être pris en charge par un ou plusieurs particuliers : on parle alors d'assainissement autonome individuel ou groupé (encore que certains auteurs parlent d'assainissement autonome à propos du lagunage en se référant aux techniques plutôt qu'aux modes d'exploitation). On utilise généralement comme principe d'épuration les techniques d'épandage consistant en une percolation de l'eau usée à travers le sol (en place ou reconstitué), dont la flore bactérienne assure la destruction des matières organiques.

Dans la suite du texte, on parlera essentiellement d'assainissement individuel, sachant qu'on s'intéresse aux techniques d'épandage, et que la différence avec l'assainissement autonome groupé se fera surtout au niveau des dimensionnements.

Ainsi, selon les contraintes imposées par l'urbanisme et les modes communautaires, on évolue depuis l'assainissement collectif de type urbain jusqu'à l'assainissement individuel, à travers toute une série de solutions intermédiaires comme le lagunage.

Un aspect particulier va différencier l'assainissement individuel du collectif. Ce dernier correspond finalement à une concentration des eaux usées en un point (la station d'épuration). Aussi, la mise en œuvre du système d'épuration et le contrôle de son fonctionnement seront théoriquement aisés car la pollution se trouve géographiquement localisée. En ce qui concerne l'assainissement autonome et plus

particulièrement individuel, la pollution va conserver un caractère diffus, et donc plus difficile à maîtriser. Cette caractéristique de l'assainissement en milieu rural s'avère importante dans le cadre des projets d'aménagement du territoire qui devront intégrer cette contrainte vis à vis de l'aptitude des sols à l'épuration.

De plus, l'assainissement individuel pose un véritable problème face aux multiples possibilités de choix, de variantes de techniques. Or, ce qui est grave, c'est que ces variantes sont dépendantes des caractéristiques physiques de la parcelle à assainir; un mauvais choix entraînera un mauvais fonctionnement, et donc un assainissement qui ne remplira pas ses fonctions : risques sanitaires, pollution de l'environnement, nuisances du voisinage s'en trouveront accrus.

Or, il se construit actuellement en France environ 200 000 dispositifs d'assainissement individuel par an.

C'est donc avec l'objectif précis de proposer un outil d'aide au choix d'un assainissement individuel, tant dans le cadre de la simulation que dans un cadre professionnel, que nous avons mené cette partie de nos travaux.

1.2 L'ASSAINISSEMENT INDIVIDUEL

On a coutume de distinguer quatre fonctions dans un assainissement individuel (figure 3-1) :

la collecte : il s'agit de récupérer toutes les eaux usées issues d'une maison : les eaux ménagères et les eaux vannes. En ce qui concerne cette fonction, on ne rencontrera que des problèmes de tuyaux, de siphons, voire de pompes de relevage lorsqu'une alimentation d'eau est située sous le niveau de l'exutoire général des eaux usées (cave et sous-sol). Les eaux pluviales sont collectées et évacuées par ailleurs.

le pré-traitement : c'est au moins une fosse septique; hélas bien des particuliers s'en contentent en guise de système d'épuration. Pour une quantité d'eau usée comprise entre 100 et 150 l/jour/habitant, on estime qu'une fosse septique peut avoir des rendements d'élimination des matières organiques de l'ordre de 40 à 60%, de 85% environ pour les matières en suspension (M.E.S) qui atteindraient en sortie de fosse septique des concentrations de l'ordre de 60 à 70 mg/l en moyenne. La demande

figure 3-1 : principe de l'assainissement individuel

biochimique en oxygène à 5 jours ($D.B.O._5$) sortante se situerait entre 150 et 250 mg/l [GARANCHER 1986].

La véritable fonction d'une fosse septique est en fait de préparer les eaux domestiques à leur traitement ultérieur par une liquéfaction des effluents et une rétention des M.E.S.. Ainsi, en cas d'épandage, on évitera les problèmes de colmatage rapide.

Selon le cas, on pourra adjoindre des équipements complémentaires du type déshuileur ou désableur.

Les fosses septiques et autres équipements sont des éléments pré-fabriqués dont la mise en œuvre est bien maîtrisée. Leur dimensionnement est directement proportionnel à la production d'eaux usées escomptée.

le traitement : c'est la phase durant laquelle on va essayer d'obtenir des rendements tels que les M.E.S. soient ramenées à 30 mg/l et la $D.B.O._5$ à 40 mg/l au maximum [JOURNAL OFFICIEL du 9 avril 1982. Arrêté du 3 mars 1982]. Pour cela, la technique consiste à épandre l'eau dans le sol par un dispositif adéquat au sortir de la fosse septique. Les caractéristiques du sol et du sous-sol vont influer sur le dispositif d'épandage et sur son dimensionnement. Si ces caractéristiques sont très défavorables, on pourra être amené à "reconstituer" un sol (filtre à sable, tertre), voire, dans le pire des cas, à abandonner la technique d'épandage et proposer un filtre bactérien percolateur. En effet, dans la mesure où il faut essayer de proposer une solution dans tous les cas soumis, il s'agira parfois de proposer la moins mauvaise.

C'est bien dans le choix d'une solution, parmi l'ensemble des techniques possibles, que va résider la difficulté.

l'évacuation : elle consiste à envisager comment restituer au milieu naturel les effluents, après passage dans le dispositif d'épuration. Cette restitution peut s'effectuer par écoulement souterrain ou superficiel, en fonction des caractéristiques physiques et topographiques de la parcelle.

Alors qu'il n'y a pas beaucoup de problèmes en ce qui concerne la fonction collecte, peu pour la fonction pré-traitement, il n'en va pas de même pour les deux dernières qui sont dépendantes des conditions naturelles. C'est à cette partie, que nous regrouperons sous les termes "techniques d'épandage" que s'intéresse plus particulièrement Moïse.

1.3 MOISE ET LES TECHNIQUES D'EPANDAGE

Ce paragraphe a pour but de bien définir les objectifs de Moïse face aux pratiques en matière d'assainissement individuel.

Précisons tout d'abord qu'il ne s'agira pas d'innover en matière de solutions techniques, en proposant de nouveaux dispositifs d'épuration ou en essayant de comprendre les phénomènes hydrauliques et les cinétiques physico-chimiques inhérentes au processus de bio-dégradation des matières organiques. Notre objectif serait plutôt de valoriser les différentes solutions techniques en faisant en sorte qu'elles soient employées à bon escient.

En effet, il s'avère que les techniques d'épandage, que nous venons de définir sommairement, sont différentes d'une région à l'autre, liées à des pratiques et des critères régionaux, voire départementaux puisque l'état délègue ses pouvoirs en la matière aux Directions Départementales de l'Action Sanitaire et Sociale (D.D.A.S.S.).

Dans une région où la perméabilité est très faible, la limite à partir de laquelle on installe l'épandage en tranchées est moins contraignante que celle qui serait adoptée dans une région où la perméabilité serait plus forte. De même, on constate par exemple que la technique du filtre bactérien percolateur, abandonnée dans certaines régions, continue d'être appliquée dans d'autres où il n'y a peut-être pas d'autres choix possibles. On conçoit donc que les D.D.A.S.S. puissent adapter les règles de choix au contexte géo-physique, social et économique de leur département, tout en s'efforçant de respecter des rendements d'épuration convenables.

Il en découle que l'expertise en matière d'assainissement individuel est diffuse géographiquement, variable d'une région à l'autre et susceptible de compléments ou de modifications à tout moment.

Aussi, il apparaît vain de vouloir écrire un logiciel tendant à figer cette expertise dans le temps. Il est préférable de fournir un outil malléable offrant à chaque spécialiste la possibilité de communiquer à la machine ses connaissances et son savoir-faire. Cependant, le spécialiste en assainissement ne doit pas se transformer en spécialiste informaticien pour autant. Moïse doit donc proposer une structure de

formalisation des règles de l'expert suffisamment simple pour ne pas être un obstacle. On retrouve là le même type de problèmes qu'avec le logiciel précédent Promise où la principale difficulté est bien la façon dont on va aborder la formalisation des connaissances.

Il est bien clair qu'il ne s'agit pas dans ce travail de proposer des règles "universelles", mais des exemples d'instanciation de la structure que nous proposerons, tout comme nous avons proposé des exemples de bases de connaissances dans Promise.

Avant cela, observons comment se diffusent actuellement les connaissances en assainissement individuel, ce qui nous conduira à proposer une solution informatique.

2. MODES DE REPRESENTATION DES CONNAISSANCES EN ASSAINISSEMENT INDIVIDUEL

2.1 LA DISSERTATION

C'est la description dans un texte des différentes solutions possibles, et des règles pour y parvenir. C'est par exemple le texte du journal officiel du 21 septembre 1984 [JOURNAL OFFICIEL 1984]. Ce texte est forcément flou car on ne peut expliciter clairement les choix possibles sans employer de formules alambiquées. En effet, dans la mesure où il n'y a pas de règles de choix à un niveau national, le législateur est bien obligé de rester vague sur les valeurs de fourchettes numériques par exemple, en évoquant ces fourchettes par un qualificatif.

C'est le cas de tous les textes descriptifs des techniques d'épandage qui veulent avoir une portée générale. Nous citerons par exemple [GARANCHER 1986, p 64] :

> " Dans ce cas, cela implique que le sol existant, trop peu perméable sous l'ensemble du dispositif d'épandage, ne soit pas exagérément imperméable et soit d'une profondeur suffisante. Il faut également que la nappe phréatique ne remonte pas, au moins à certaines périodes de l'année, à une cote abusivement élevée."

L'extrait de cet ouvrage, par ailleurs très complet, illustre à quel point seul un spécialiste pourra interpréter les directives de ce texte, et décider de la nuance entre "trop peu perméable" et "pas exagérément imperméable".

2.2 LA REPRESENTATION GRAPHIQUE

C'est bien pour pallier à l'inconvénient décrit précédemment que de nombreux organismes ont édité des ouvrages où le graphisme est largement représenté [SNPEAI 1983], [MINISTERE DE L'ENVIRONNEMENT 1981].

La représentation graphique est un excellent moyen pour visualiser ce qu'est un épandage en tranchées, un filtre à sable ou un tertre. Accompagné d'un texte descriptif, on peut détailler comment on met en œuvre ces techniques et quelles sont les précautions à prendre. Cependant, le pourquoi du choix du dispositif (X) plutôt que (Y), là encore, se fait à travers un texte, certes plus abordable (car souvent conçu pour des non initiés), mais qui ne peut rendre compte que de cas bien typés.

2.3 TABLEAUX ET ABAQUES

On trouve un peu partout dans la bibliographie des tableaux qui éclairent les notions de perméabilité de sol ou de profondeur de nappe, ou qui fournissent des fourchettes de dimensionnement sommaires en fonction d'un ou deux paramètres. Mais, aucun ne tente de proposer des choix de filières en fonction d'un contexte.

Et pour cause, au-delà de deux paramètres, donc de deux dimensions, la représentation en plan devient un exercice délicat.

C'est pourtant un exercice que nous avons tenté et pour lequel nous avons adopté la solution qui s'est traduite par une série de tableaux. Nous verrons que l'application informatique a finalement consisté à écrire l'exploration de ces tableaux.

Pour l'expert appelé à formuler ses propres règles ou à les compléter, l'avantage d'une telle méthode réside dans la facilité de la mise au point : *il pourra visualiser ses règles et en vérifier la cohérence simplement en remplissant des tableaux. Ce n'est que lorsque les règles ainsi représentées seront validées qu'il devra se soucier de les implanter sur ordinateur pour alimenter la base de règles.*

Quant aux abaques, elles sont également difficiles à mettre au point, et nous n'en avons pas trouvé trace dans les ouvrages publiés. Nous en proposons une sur la figure 3-2, accompagnée d'un mode d'emploi (tableau 3-1). Il va de soi que les valeurs numériques, les filières conseillées sont sujettes à variation selon les régions, et selon l'avis de tel ou tel spécialiste. Ce n'est ici qu'un exemple illustrant un essai d'abaque.

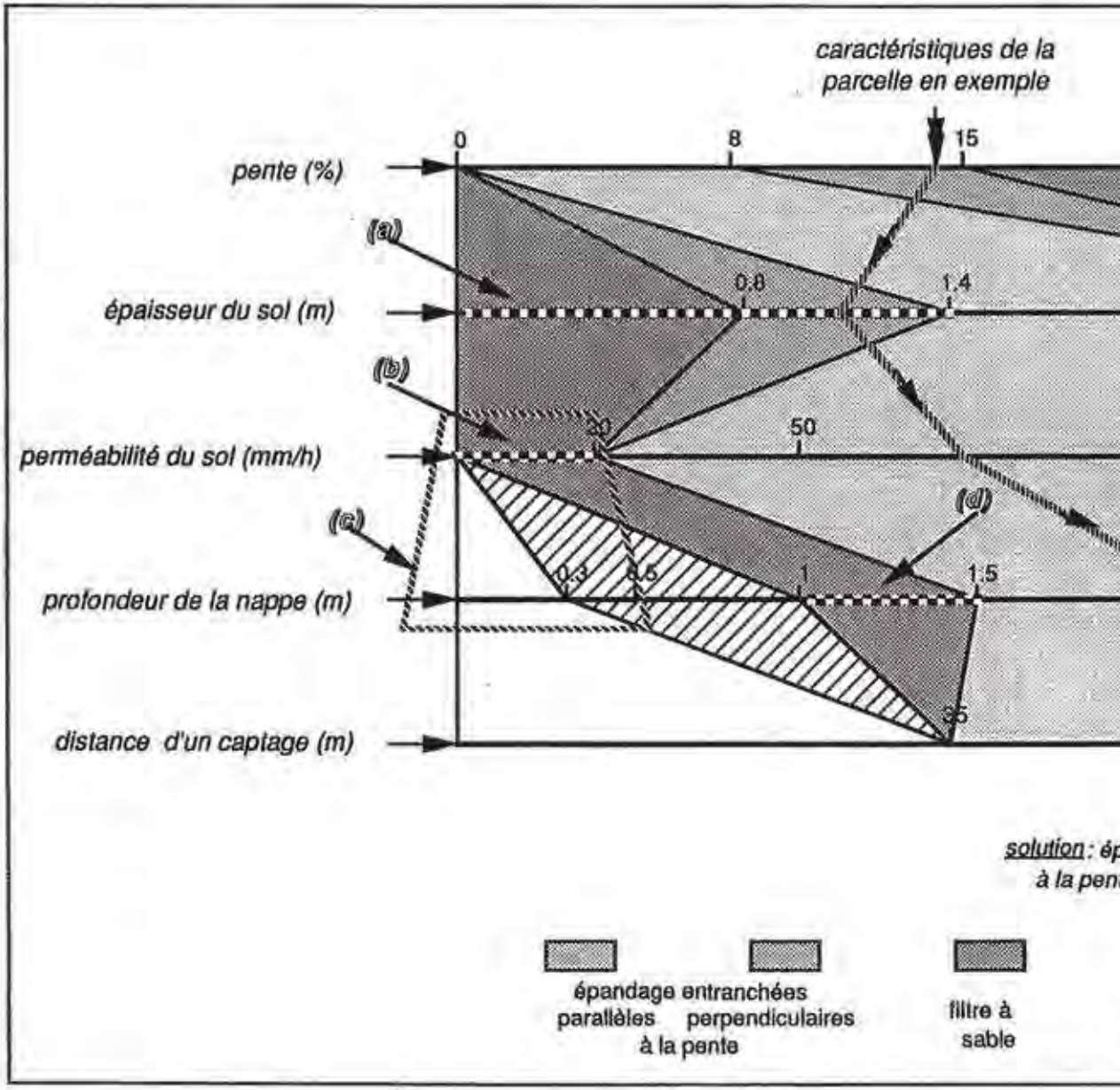


figure 3-2 : abaque Moïse
(aide au choix d'un assainissement individuel)

sélection d'une filière:

Les caractéristiques de la parcelle où sera installé un système d'assainissement individuel sont entrées sur l'abaque sous forme d'un graphe (voir exemple). Les zones que traverse le graphe indique les filières possibles.

La filière retenue correspond à la zone traversée de plus forte contrainte. Les filières sont, par ordre croissant de contrainte :

épandage en tranchées parallèles à la pente,
épandage en tranchées perpendiculaires à la pente,
filtre à sable,
tertre d'infiltration,
filtre bactérien percolateur,
zone blanche --> assainissement individuel impossible.

Tout dispositif peut être remplacé par un dispositif de contrainte supérieure.

Certaines zones particulières sont définies sur un ou plusieurs axes :

(a)
Si la filière retenue est
épandage en tranchées
filtre à sable
et que le graphe traverse la zone *(a)*
alors le sous-sol de la parcelle doit être perméable.

Si le sous-sol n'est pas perméable, on s'orientera vers une autre filière.

(b) Quelle que soit la filière retenue,
si le graphe traverse la zone *(b)*
alors le sous-sol de la parcelle doit être perméable *ou* les effluents doivent pouvoir être évacués par un écoulement superficiel (fossé, ruisseau ...).

Si ce n'est pas le cas, alors la filière retenue sera le filtre bactérien percolateur.

(c)
Si un segment du graphe (celui qui rejoint la perméabilité du sol à la profondeur de la nappe) est totalement inclus dans la zone *(c)*, alors c'est la filière "filtre bactérien percolateur" qui doit être retenue.

(d)
Si la filière retenue est le filtre à sable
et que le graphe traverse la zone *(d)*
alors le filtre à sable devra être à écoulement horizontal.

exemple :
Une parcelle a les caractéristiques suivantes :
pente du terrain : 14%,
épaisseur du sol : 1.15m,
perméabilité du sol : 100mm/h
profondeur de la nappe : 2m,
pas de captage à proximité.

La filière retenue sera "épandage en tranchées perpendiculaires à la pente" si le sous-sol est perméable; dans le cas contraire, on s'orientera vers un tertre.

tableau 3-1 : interprétation de l'abaque Moïse

Le lecteur pourra se rendre compte, en consultant ce schéma, de la difficulté de son interprétation. Cet aspect va concourir à renforcer l'approche informatique que nous avons adoptée.

On peut faire plusieurs remarques à propos de ce schéma :

- l'introduction des particularités se fait par l'isolement de zones spéciales. Il est alors précisé quelle attitude adopter si on traverse la zone en question. Cette façon de faire est très lourde mais elle n'est pas contournable; elle rend compte de l'interférence entre les paramètres et de l'apparition de combinaisons qui impliquent de sortir du cas général.
- il y a une "compatibilité ascendante" dans la mise en œuvre des différentes filières : là où on peut mettre un épandage en tranchées, on peut mettre en place un filtre à sable, un tertre ou un filtre bactérien; ces quatre dispositifs étant classés de façon à ce que leurs contraintes de mise en œuvre diminuent avec le numéro d'ordre du classement. Cette propriété est largement utilisée dans l'abaque.
- Ce type de schéma est incapable d'intégrer facilement de nouveaux dispositifs ou de nouvelles contraintes.

2.4 LA PROGRAMMATION

Nous avons trouvé dans la bibliographie deux auteurs ayant tenté de résoudre par la programmation le problème du choix d'un assainissement individuel. Ce sont les travaux de GRAILLOT [GRAILLOT 1986] et ROUMAGNAC [ROUMAGNAC 1987] .

D. GRAILLOT a utilisé un générateur de systèmes experts du commerce: Mac-Expert. Son moteur (d'ordre 0) autorise les deux modes de consultation 'déduction' et 'vérification', et une possibilité d'explorer le cheminement emprunté par le système parmi le graphe de règles. Sa principale faille réside dans l'interface "homme-machine" réalisée à partir d'un lexique limitant la souplesse d'utilisation du logiciel.

A. ROUMAGNAC a développé un moteur capable d'explorer une modélisation de la connaissance par un réseau sémantique. Ce logiciel, écrit en Pascal, tente de reproduire la scission que l'on trouve naturellement dans un langage comme Prolog.

L'auteur met en évidence l'intérêt d'une individualisation, dans une base indépendante, de la connaissance, notamment pour la maintenance. On peut regretter que cette base soit conçue à partir d'un codage numérique des nœuds et des relations du réseau sémantique. Cet aspect nuit à la lisibilité de la base de connaissances.

On notera, dans ces deux exemples, que la base de connaissances, sous forme d'un graphe arborescent, et bien qu'isolée du moteur, est difficilement maintenable - d'un point de vue conceptuel -. En effet, à défaut d'être un informaticien pour manipuler ces bases, il faut être bien informé de leur conception pour intervenir sur un nœud ou une règle afin d'apprécier avec justesse les conséquences d'une modification sur la cohérence de la base de règles.

Le problème de la maintenance s'en trouve transféré vers la logique de la base de règles, cette logique étant fortement imprégnée par la vision et l'esprit de rigueur du concepteur (tout comme un programme informatique classique est plus ou moins maintenable selon la façon dont il est écrit). Très concrètement, vouloir par exemple introduire un nouveau paramètre, pour peu que celui-ci intervienne au niveau de plusieurs nœuds, peut amener à refondre complètement le graphe. A terme, la personne la plus apte, et peut-être la seule, à pouvoir intervenir sur la base de règles est sa conceptrice.

Nous verrons que nous avons tenté de remédier à cet inconvénient en proposant un formalisme de règles très redondant, mais plus accessible.

2.5 CONCLUSION

Parmi les différentes représentations des connaissances en assainissement individuel évoquées, la technique de la programmation semble être la seule capable de rendre compte de la disparité des solutions possibles. Il est remarquable que les deux programmes cités soient réalisés à partir de la technique des systèmes experts.

En effet, les deux conditions de mise en œuvre d'un système expert sont remplies dans le cas du choix d'un assainissement individuel :

- il n'y a pas d'algorithme;
- il faut pouvoir maintenir la base de connaissances.

Aussi, nous avons conservé cette technique, en tâchant d'en orienter les caractéristiques vers une grande maniabilité.

3. FONCTIONNEMENT DE MOISE

3.1 METHODOLOGIE

3.1.1 FORMALISATION DES REGLES

Rappelons ici que le but de la méthodologie développée est de fournir au spécialiste un outil lui permettant de modéliser sa connaissance en favorisant la clarté des règles, notamment en vue de leur maintenance.

Ces règles, en assainissement individuel, peuvent se ramener à une série de combinaisons de valeurs de paramètres, ces valeurs pouvant être de nature qualitative ou quantitative. La difficulté consiste finalement à bien explorer toutes les combinaisons et à proposer dans chaque cas une solution circonstanciée, c'est-à-dire accompagnée d'un commentaire, de conseils, de réserves qui permettent, comme un expert, de nuancer la solution retenue en fonction du contexte.

En effet, à l'heure actuelle, il est illusoire de vouloir prendre en compte *tous* les facteurs que peut recenser un spécialiste. C'est par exemple le cas de l'encombrement de la parcelle à assainir: cet encombrement peut avoir de multiples origines (terrain de jeu, arbres, A.E.P....) et vouloir qualifier ou quantifier un paramètre 'encombrement de la parcelle' par rapport aux caractéristiques d'un assainissement à installer (surface, forme, profondeur...) reste à l'heure actuelle utopique, beaucoup de paramètres n'étant pas maîtrisés.

Il y a deux façons d'écrire les combinaisons de paramètres (figure 3-3) :

- un graphe "arborescent", à la manière d'A. ROUMAGNAC et D. GRAILLOT;
- un graphe "en haies" qui décrit explicitement les combinaisons.

Dans un graphe "en haies", chaque feuille représente un paramètre associé à une ou plusieurs valeurs. Une série de feuilles constitue une branche du graphe et illustre une combinaison de valeurs de paramètres aboutissant à une solution : c'est une règle. Un

figure 3-3 : comparaison des structures de formalisation de la connaissance

groupe de règles convergeant vers une même solution, donc une description de différents contextes pour lesquels une même solution s'impose, forme un "arbuste".

Cette structure laisse à jour de nombreuses redondances puisqu'une même condition sur un paramètre s'écrit autant de fois qu'elle apparaît au sein d'une règle, c'est-à-dire qu'une feuille identique peut être présente sur plusieurs branches.

En contrepartie, la lisibilité est grande puisque chaque règle est indépendante des autres. La structure se compulse linéairement, branche par branche, et il n'y a pas besoin de s'imprégner de la logique d'une structure arborescente, avec tous ses embranchements, pour pouvoir intervenir. De plus, chaque arbuste peut se transcrire sur un tableau à double entrée (en ligne les valeurs de paramètres et en colonne, les combinaisons de ces valeurs; nous en verrons des exemples plus loin). Ainsi, une haie peut se traduire par une série de tableaux, facilitant la mise au point des règles, les échanges d'information et leur lecture.

Le graphe "arborescent" est une solution plus compacte puisqu'il est en quelque sorte une factorisation du graphe en haie. En effet, il tente de regrouper en un nombre de nœuds minimum les enchaînements qui apparaissent plusieurs fois dans la structure. A terme, cela est plus concis, mais altère la lecture du graphe, ainsi que les possibilités d'intervention rapide.

Ce graphe "en haies" que nous avons retenu s'apparente au "raisonnement par étude de cas" ("case-based reasonning") [CARBONNEL, MICHALSKI, MITCHELL 1986] utilisé en Intelligence Artificielle, notamment dans des applications en droit et en médecine.

On trouvera des exemples de bases de règles ainsi structurées dans la suite du texte.

3.1.2 STRATEGIES D'EXPLORATION D'UNE HAIE

3.1.2.1 RECHERCHE D'UNE SOLUTION

La recherche d'une solution est la façon "naturelle" d'exploiter les règles de la base de connaissances. Cette recherche s'applique particulièrement dans le cas de personnes non qualifiées dans le domaine de l'assainissement, qui vont attendre de Moïse qu'il lui fournisse un dispositif adapté au contexte communiqué.

La recherche d'une solution revient à explorer l'ensemble des arbustes. Pour chacun, on scrute chaque branche.

En ce qui concerne l'assainissement individuel, nous avons déjà fait remarquer qu'il existe une hiérarchie dans la sophistication des dispositifs qui va d'ailleurs de pair avec le coût des installations. On a donc intérêt dans ce cas à classer les solutions par ordre de sophistication/coût croissant et à *interrompre l'exploration de la haie dès qu'une solution est obtenue* (tout en laissant éventuellement la possibilité de continuer la recherche vers une autre solution).

Cette stratégie se justifie d'autant mieux lorsque l'on a à faire à des utilisateurs non initiés à l'assainissement, pour lesquels une liste de toutes les solutions trouvées ne résoudrait en rien leur problème de choix.

On verra que, dans d'autres applications, c'est la stratégie consistant à rechercher et à soumettre toutes les solutions possibles qui a prévalu.

3.1.2.2 VERIFICATION D'UNE SOLUTION

La vérification d'une solution est un mode de consultation qui concerne des personnes initiées à l'assainissement individuel qui, à partir d'une hypothèse sur un dispositif à installer, peuvent vérifier si celui-ci correspond au contexte de la parcelle à assainir.

Cette vérification d'une solution consiste à ne considérer qu'un arbuste. Il suffit qu'une des branches corresponde au contexte communiqué par l'utilisateur pour que la solution soit vérifiée.

On verra cependant, dans un exemple de règles de choix, que l'introduction du mode vérification implique certaines contraintes dans la façon d'écrire les règles.

3.2 ORGANISATION DU LOGICIEL

Une fois la structure précédente adoptée, il faut écrire un programme permettant :

- d'initialiser le problème (caractéristiques des paramètres ...);
- d'écrire des règles sous forme de haie;
- d'explorer une haie selon la stratégie choisie;-

- d'assurer l'interface avec l'utilisateur pour lui permettre de soumettre son problème.

C'est l'objet de ce logiciel, Moïse, dont nous développons les caractéristiques ci-après.

3.2.1 STRUCTURE ET CHOIX TECHNIQUES

Nous allons retrouver la même structure que dans le logiciel précédent, avec la répartition des tâches suivantes :

- la base de règles, qui contient les règles de choix d'un assainissement individuel et l'initialisation des paramètres;
- le moteur, qui assure la stratégie d'exploration des règles et l'interface avec l'utilisateur (aide, question sur les paramètres ...);
- la base de faits, qui stocke les réponses de l'utilisateur et les déductions intermédiaires.

Le langage retenu est toujours Turbo-Prolog (Annexe A1). Outre les qualités intrinsèques au langage, ce choix permet une homogénéïté avec le simulateur de projet Promise dont Moïse constitue une procédure.

Par ailleurs, en vue du couplage de Moïse avec d'autres logiciels (serveur Minitel), Turbo-Prolog offre la possibilité d'être compilé, et d'accepter des procédures écrites en différents langages.

Le choix "machine" reste également le micro-ordinateur pour des raisons de coût et de diffusion.

3.2.2 LA BASE DE REGLES

3.2.2.1 LES PARAMETRES

Chaque paramètre est décrit indépendamment (par un prédicat), ce qui permet d'en considérer un nombre théoriquement illimité.

Cette description prend en compte, outre le nom du paramètre, un certain nombre de caractéristiques (voir en annexe A3 la description précise du prédicat).
Les principales sont :

• les qualificatifs associables aux paramètres et éventuellement, les bornes numériques correspondantes.

(ex : le paramètre "*pente du terrain*" peut prendre pour valeur :
très faible (< 2 %)
faible (de 2 à 8 %)
moyenne (de 8 à 15 %)
forte (de 15 à 25 %)
très forte (> 25 %)).

• l'unité du paramètre en question.

• le verbe d'accompagnement qui permet de construire "automatiquement" des phrases par une syntaxe :

paramètre - verbe - qualificatif
(ex : la pente du site - est - faible).

Ce verbe permet également de fabriquer des questions avec l'enchaînement :

comment - verbe - paramètre ?
(ex : comment - est - la pente du site ?) ou bien
est-ce que - paramètre - verbe - qualificatif ?
(ex : est-ce que - la pente du site - est - faible ?).

Ce formalisme de faits est relativement rigide, mais s'avère en l'occurrence suffisant. On pourrait éventuellement le compléter en faisant apparaître une valeur numérique, quand celle-ci est fournie.

3.2.2.2 LES REGLES

Chaque règle décrit une combinaison de valeurs des paramètres précédents, aboutissant à une filière. Le formalisme en est le suivant :

SI le paramètre (p_1) a pour valeur (v_{11} **ou** v_{12} ... **ou** v_{1n})
et le paramètre (p_2) a pour valeur (v_{21} **ou** v_{22} ... **ou** v_{2m})
...
et le paramètre (p_i) a pour valeur (v_{i1} **ou** v_{i2} ... **ou** v_{ij})
ALORS
la solution est le dispositif (d) accompagné des commentaires (com).

Il y a autant de règles que de contextes différents décrits chacun par un commentaire associé; il peut donc y avoir plusieurs règles pour un même dispositif.

Le nombre maximum de règles est théoriquement illimité.

On peut pour des raisons logiques ou physiques (mémoire centrale limitée) regrouper des conditions sur les valeurs de paramètres, qui apparaissent de façon répétitive sur une haie, sous une même notion. On crée alors une "bouture" qui vient remplacer une feuille. Ceci nous rapproche de la structure nœudale d'un graphe arborescent tout en conservant chaque arbuste de la haie : on altère peu la lisibilité de la base de règles.

3.2.2.3 REMARQUES

- Contrairement au logiciel précédent Promise, la base de règles est ici compilée avec le moteur. Cette version a été conservée car ce fut la première développée.

Cependant, une version avec la base de règles non compilée a été testée avec succès, autorisant éventuellement une maintenance dynamique.

- Rappelons que, dans le but de satisfaire la stratégie choisie de ne proposer que la solution "minimale" au problème posé, la construction de la base de règles doit être "ordonnée".

3.2.3 LA BASE DE FAITS

Elle enregistre les réponses des utilisateurs et, quand il y en a, les déductions intermédiaires du moteur.

Cette base de faits gère deux types principaux de prédicats qui ont le formalisme suivant :

- *oui* (paramètre, verbe, qualificatif)
- *non* (paramètre, verbe, qualificatif)

ex :

- *oui* (la pente du site, est, faible)
- *non* (le substratum, est, plutôt perméable)

On remarquera que la syntaxe adoptée rend lisible, "en français", les éléments de la base de faits.

3.2.4 LE MOTEUR

Nous ne détaillerons pas dans ce paragraphe le fonctionnement intime du moteur. Nous nous contenterons d'en indiquer la structure générale. Le lecteur trouvera en annexe A3 le mode opératoire du logiciel (action que provoque chaque touche au niveau de chaque menu) et des spécificités techniques approfondissant les caractéristiques du moteur.

Nous distinguons 4 menus principaux dont l'enchaînement est représenté sur la figure 3-4.

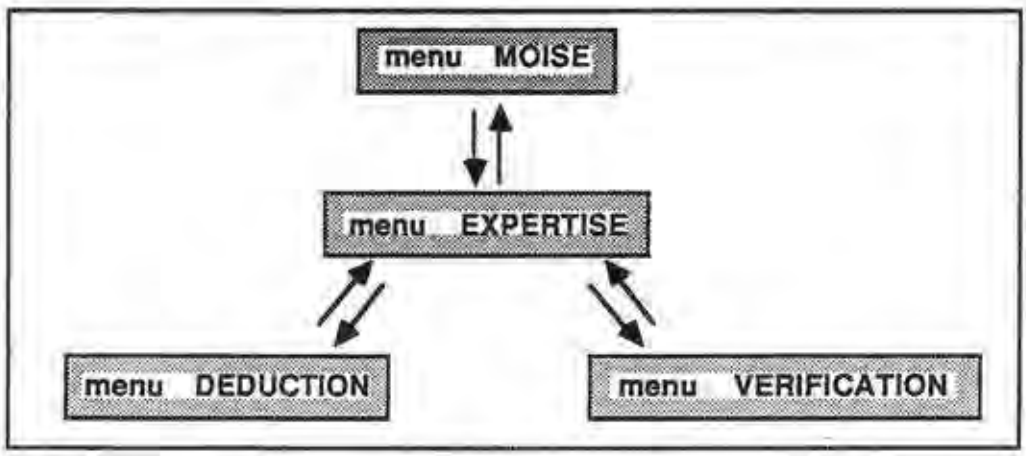


figure 3-4 : enchaînement des principaux menus de Moïse

C'est dans le premier menu, menu "Moïse" (figure 3-5) que se situe l'interface entre le système d'exploitation et *les bases de faits* qui peuvent être stockées.

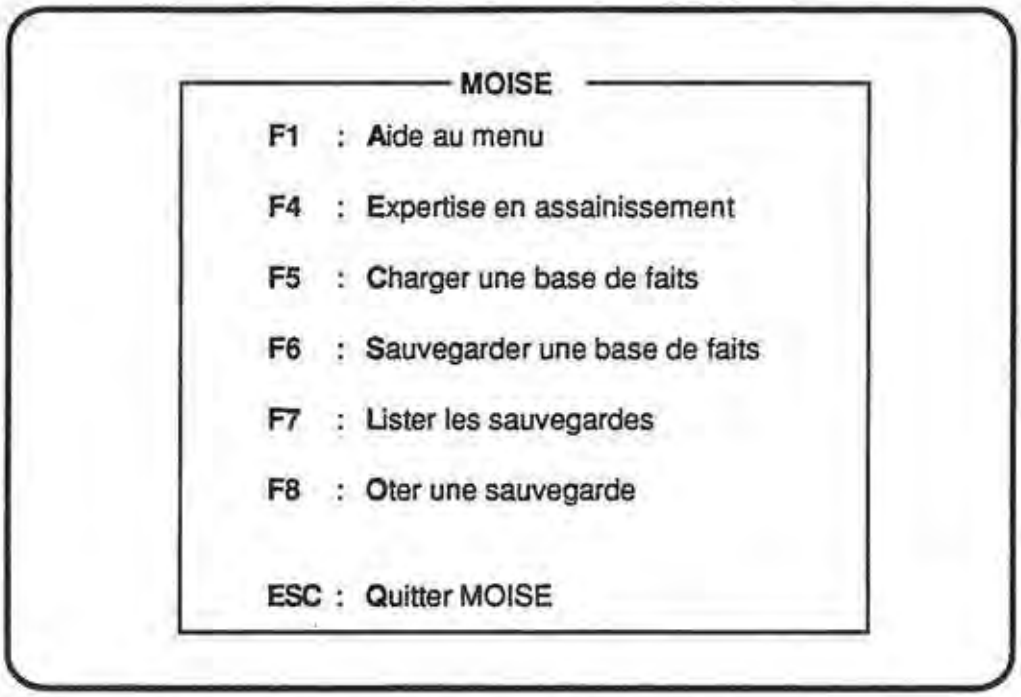


figure 3-5 : premier écran; menu Moïse

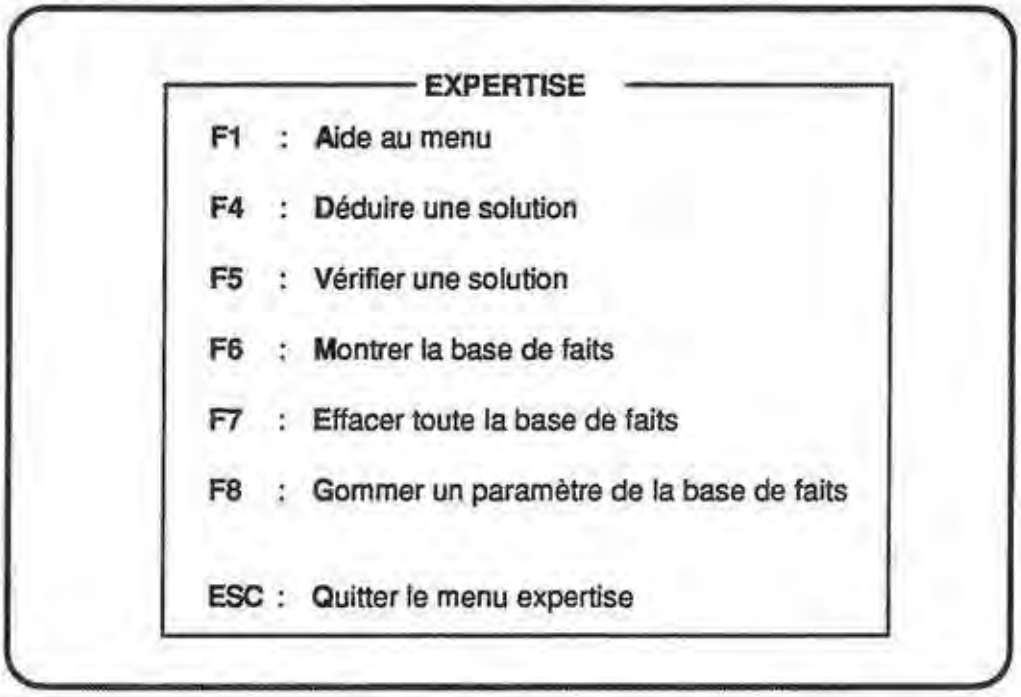


figure 3-6 : deuxième écran ; menu Expertise

En effet, une base de faits est un fichier dynamique dont le contenu peut à tout moment être enregistré sur un support physique (disque ou disquette).

Cette interface gère un certain nombre de fonctions :

- *charger* une base de faits enregistrée;
- *sauvegarder* une base de faits (sur support physique);
- *lister* toutes les sauvegardes réalisées;
- *ôter* une sauvegarde (du support physique).

Le deuxième menu (menu "Expertise") se charge des fonctions de manipulation *d'une base de faits* (figure 3-6) :

- *visualiser* la base de faits (plus exactement, les valeurs des paramètres telles que l'utilisateur les a saisies; on évite de faire apparaître toutes les déductions intermédiaires car cela rendrait l'affichage souvent trop volumineux, donc illisible);
- *gommer* la ou les occurences d'un paramètre;
- *effacer* totalement la base de faits pour une autre session.

(on retrouve ces 3 options dans les 2 derniers menus)

La deuxième fonction est extrêmement intéressante car elle permet à l'utilisateur de tester la sensibilité de la solution trouvée par rapport à un paramètre. Ainsi, en faisant varier la valeur de ce paramètre, et si les filières déduites sont alors différentes, l'utilisateur saura qu'il doit affiner la connaissance du paramètre en question. Dans le cas contraire, le paramètre n'est pas significatif et une approximation est largement suffisante.

Deux modes d'expertise sont proposés (figures 3-7 et 3-8) :

- le mode *déduction* , qui permet de rechercher une solution. Nous rappelons que, selon la stratégie choisie, le moteur affiche un résultat dès qu'il a trouvé une solution. On peut, en ne quittant pas le mode déduction, chercher un autre dispositif que celui déjà trouvé, en relançant une session sans modifier la base de faits.

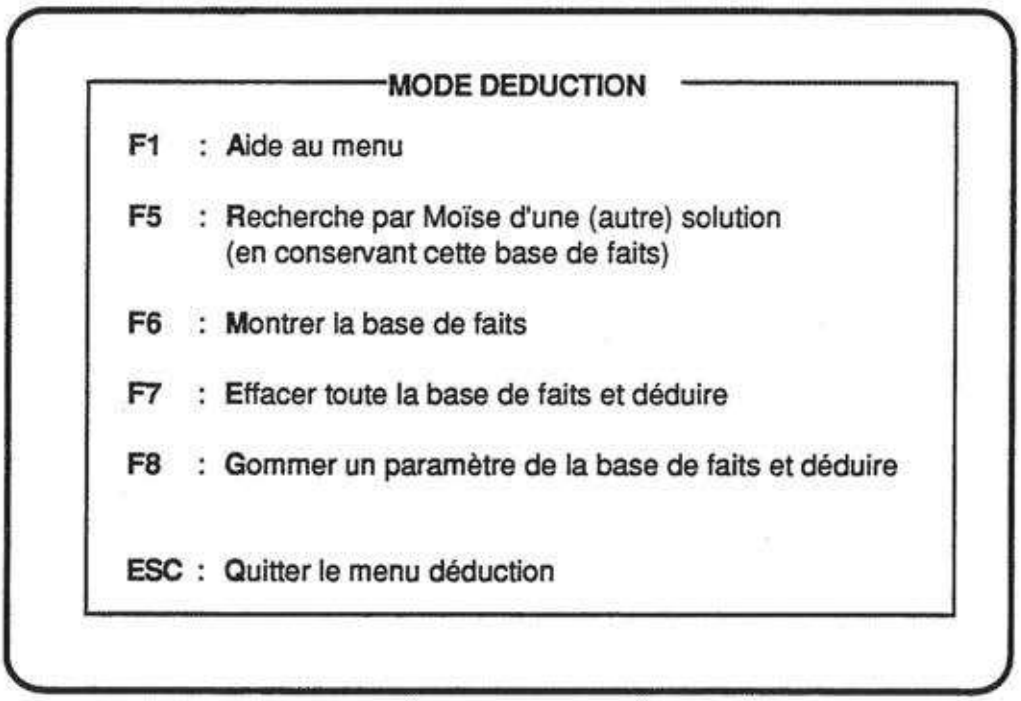


figure 3-7 : menu déduction

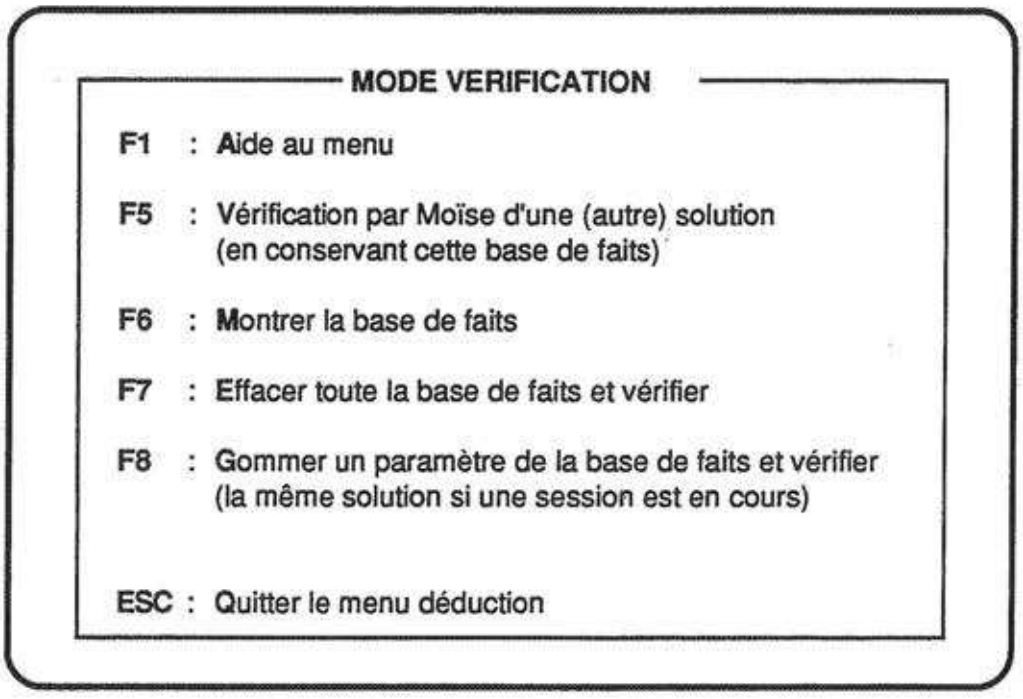


figure 3-8 : menu vérification

• le mode *vérification* , qui propose d'abord, parmi l'ensemble des filières que le système connaît, de choisir la filière à vérifier. Cette vérification se fait donc en n'explorant qu'un arbuste de la formalisation en haies.

A propos des deux derniers menus, on trouvera des précisions sur leur fonctionnement dans l'exemple de session du paragraphe 4.2.

L'organigramme général de Moïse est illustré par la figure 3-9 sur laquelle apparaissent tous les cheminements possibles du logiciel.

saisie et gestion des paramètres

En cheminant à travers les règles, le moteur doit, au fur et à mesure que les conditions sur la valeur des paramètres apparaissent, interroger l'utilisateur. Ces interrogations ne doivent intervenir qu'une seule fois pour chaque paramètre. Aussi, à l'issue d'une réponse, le moteur génère, quand cela lui est possible, des déductions lui permettant de pouvoir répondre par avance à toutes conditions concernant un paramètre déjà rencontré.

ex :

si l'utilisateur déclare :

"la pente du site est faible"

le moteur déduira que tous les autres qualificatifs possibles du paramètre "la pente du site" sont faux; ainsi, les faits suivants seront stockés :

oui(la pente du site, est, faible)
non(la pente du site, est, très faible)
non(la pente du site, est, moyenne)
...

recherche d'un échec

Dans le cas où aucune solution n'est trouvée, le système tente d'en comprendre les raisons grâce à un jeu de règles annexes intégrées à la base de règles.

Ces règles annexes s'écrivent strictement de la même façon que les premières et c'est donc le même mécanisme qui va les explorer. Elles rendent compte de certaines combinaisons de valeurs de paramètres qui font que l'assainissement individuel est inadapté. On trouvera plus loin des exemples de ce type de règles.

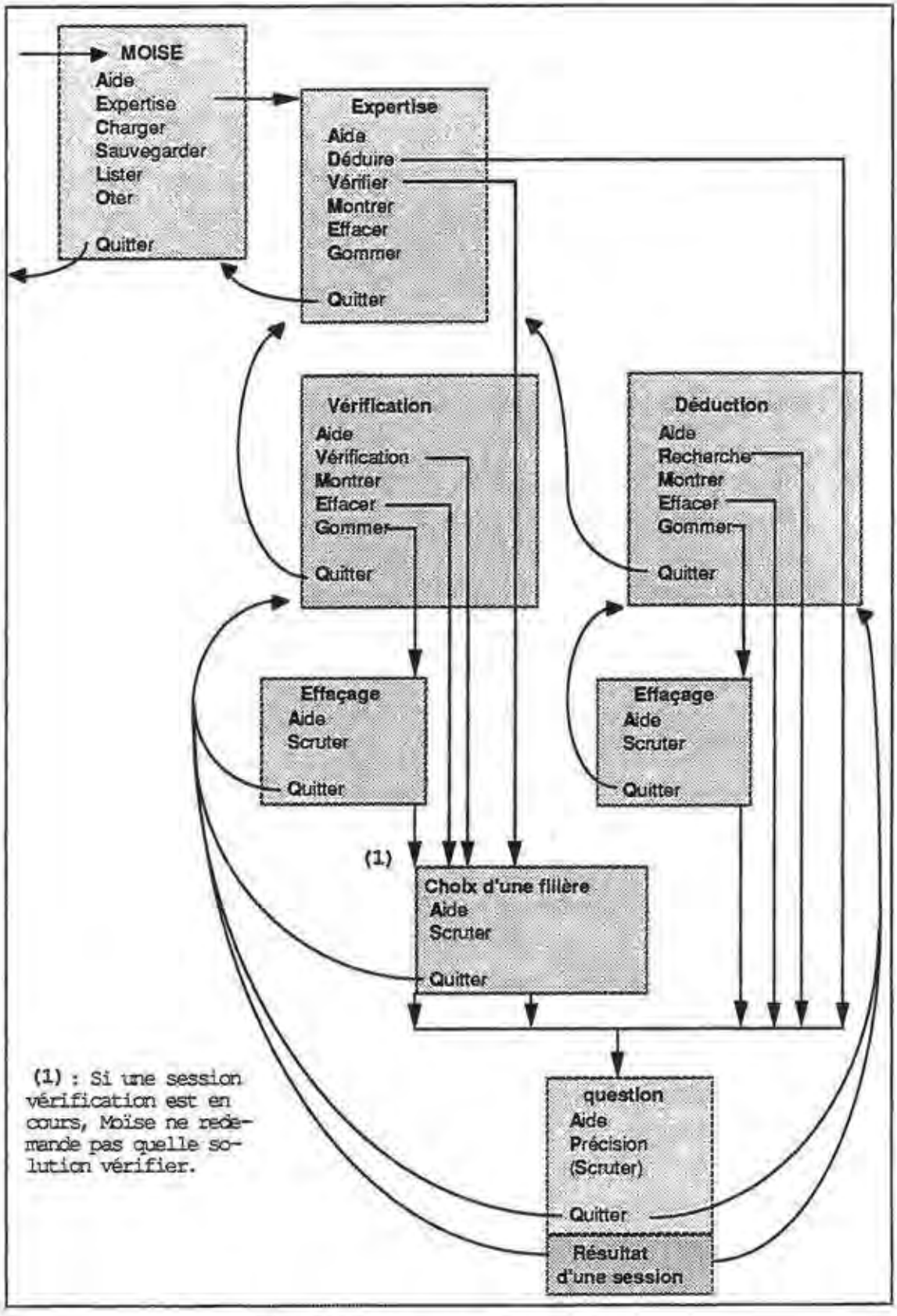


figure 3-9 : Organigramme général de Moïse

On verra plus loin comment des modules complémentaires peuvent être intégrés : contrôle de cohérence des faits, dimensionnement des dispositifs ...

3.3 CONCLUSION

Nous venons de jeter les bases d'une méthodologie permettant de programmer les solutions d'un problème où un grand nombre de combinaisons apparaît.

La technique retenue - structure en 'haies' - introduit de nombreuses redondances dans la formulation des règles. Cette redondance permet une lisibilité maximale avec l'intention de favoriser la maintenance du logiciel.

Nous allons tester ce principe et vérifier son caractère opérationnel, vis-à-vis de la lisibilité et de la maintenance, sur différents exemples.

Il est évident que cette manière d'écrire des règles est très éloignée de la façon dont un expert raisonne. Sans vouloir entrer dans ce débat, il apparaît acquis qu'un expert ne passe pas en revue, de façon systématique, toutes les combinaisons possibles d'un problème combinatoire pour aboutir à une solution.

Dans notre cas, cette modélisation de la connaissance présente l'avantage d'être facilement reproductible et accessible.

4. MISE EN ŒUVRE DE MOISE

4.1 EXEMPLE DE BASE DE REGLES

Afin d'illustrer les propos précédents, nous développons dans ce paragraphe une base de règles relative au choix d'un assainissement individuel.

Nous rappelons qu'il n'est pas question ici de prétendre proposer une base de règles universelle, mais plutôt de montrer comment construire les règles, et observer si leur lisibilité autorise une maintenance aisée.

La connaissance modélisée dans cette base est pour beaucoup issue de la bibliographie, et pour une moindre part, de discussions avec des spécialistes de la D.D.A.S.S (Direction Départementale de l'Action Sanitaire et Sociale) de la Loire et de l'A.F.B. (Agence Financière de Bassin) Loire-Bretagne.

En fait, dès les premiers essais, il est apparu que cette manière de faire était très profitable aux échanges de point de vue : on possède un noyau de règles à partir duquel on peut proposer des modifications, voire une refonte plus profonde.

Cette base a également été testée sur des cas réels pour lesquels, mis à part les fluctuations régionales et l'avis divergent des spécialistes, elle a donné des résultats jugés suffisants dans le cadre d'une démonstration des capacités du logiciel (sur Minitel par exemple) (on trouvera en annexe A4 la représentation informatique complète de la base de règles en question).

4.1.1 LES DIFFERENTES SOLUTIONS

L'exemple développé réunit 14 solutions possibles (figure 3-10), prenant simultanément en compte le système d'épuration (tranchées, filtre à sable, tertre, filtre bactérien percolateur) et le dispositif d'évacuation (par percolation, rejet superficiel ou puits d'infiltration).

4.1.2 LES PARAMETRES

Neuf paramètres entrent en jeu. Ils décrivent le contexte géo-morphologique et pédologique du site d'implantation du dispositif d'assainissement (tableau 3-3). Certains paramètres sont d'ordre strictement descriptif, d'autres sont des qualificatifs rattachés à des bornes numériques.

Il faut remarquer que l'on n'échappe pas ici au problème des limites de classes de part et d'autre desquelles on pourra avoir deux systèmes d'assainissement différents (pour une variation minime d'un paramètre). On améliore simplement un peu l'approche de ce problème en tâchant de mettre en avant les qualificatifs de classes plutôt que des valeurs numériques. Nous pensons en effet qu'un spécialiste, dans ses raisonnements, retient bien plus un ordre de grandeur de la valeur d'un paramètre (ce que nous ramenons à un qualificatif) qu'une valeur numérique précise.

Deux remarques renforcent cette approche qualitative :

- les erreurs de mesure relativisent parfois substantiellement la valeur mesurée, justifiant la prise en compte d'un ordre de grandeur;
- le manque d'outillage des utilisateurs, surtout quand on s'adresse à des personnes non initiées, nuit à une estimation précise des valeurs numériques.

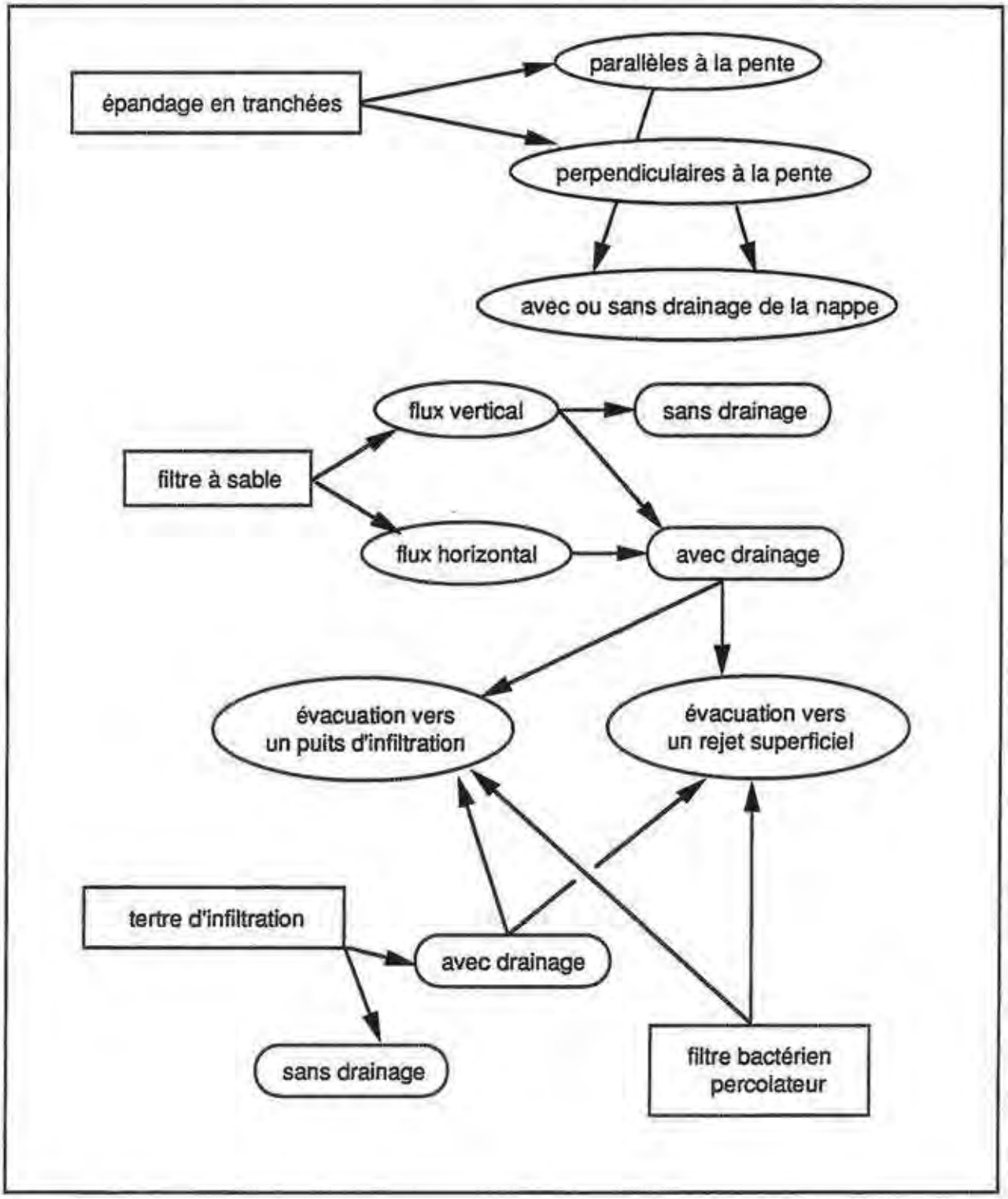


figure 3-10 : les différentes techniques possibles en assainissement individuel

nom	verbe	qualificatifs	bornes num.	unités
la pente du site	est	très faible faible moyenne forte très forte	< 2 2 à 8 8 à 15 15 à 25 > 25	pourcentage (%)
l'épaisseur du sol	est	très faible faible moyenne forte	< 0.8 0.8 à 1.4 1.4 à 2 > 2	mètres (m)
la perméabilité du sol	est	très faible faible moyenne forte très forte	< 10 10 à 20 20 à 50 50 à 500 > 500	millimètres par heure (mm/h)
l'existence d'une nappe	est	constatée pas constatée	- -	-
la profondeur de la nappe	est	rédhibitoire très faible faible moyenne forte très forte	< 0.3 0.3 à 0.5 0.5 à 1 1 à 1.5 1.5 à 2 > 2	mètres (m)
le sous-sol	est	plutôt perméable plutôt imperméable	- -	-
le rejet superficiel	est	possible impossible	- -	-
un captage d'eau dans les environs	est	constatée pas constatée	- -	-
la distance du captage d'eau	est	trop faible satisfaisante	< 35 > 35	mètres (m)

tableau 3-3 : liste et caractéristiques des paramètres

4.1.3 LES REGLES

Comme nous le signalions précédemment, les règles peuvent être représentées dans des tableaux à double entrée, ce qui facilite leur mise au point et leur maintenance.

Le symbolisme de ces tableaux (dont nous reproduisons un exemple dans le tableau 3-4, l'ensemble de ceux-ci se trouvant en annexe A4) est le suivant :

- une croix signifie que la condition doit être remplie; si la valeur du paramètre est inconnue, Moïse interroge l'utilisateur.
- des hachures signifient que le paramètre est ignoré dans l'exploration.
- 'oui' : la condition doit être remplie; 'non' : la condition doit être fausse; le logiciel n'interroge pas l'utilisateur dans ces deux derniers cas; si la valeur est inconnue, la condition est considérée fausse. Ce type de sysmbolisme n'apparaît que dans les règles annexes où l'exploration des règles est passive, c'est-à-dire sans interpellation de l'utilisateur.

Ainsi, la colonne 1-A du tableau 3-4 se lit :

SI *la pente du site* est faible **ou** très faible
et *l'épaisseur du sol* est forte **ou** moyenne
et *la perméabilité du sol* est moyenne **ou** forte
et il n'y a pas de *nappe*

ALORS la solution est "*épandage en tranchées parallèles à la pente*";

le commentaire associé pourra ici mettre en évidence qu'il s'agit d'un contexte très favorable à ce dispositif tout en mettant en garde au niveau des précautions élémentaires de mise en place des tranchées.

Cette base de règles rassemble 68 règles aboutissant à 14 solutions, et 9 règles expliquant un échec. Nous n'avons pas reproduit ici l'ensemble des commentaires associés à chaque combinaison. Ceux-ci reprennent le contexte décrit, dans lequel l'utilisateur retrouve ce qu'il vient de déclarer au système, et proposent des avis destinés à attirer l'attention sur certains cas particuliers de mise en œuvre, ou

EPANDAGE EN TRANCHEES PARALELLES à la PENTE								
			1A	1B	1C	1D	1E	1 F
pente	très forte	> 25%						
	forte	15 - 25%						
	moyenne	8 - 15%						
	faible	2 - 8%	X	X	X	X	X	X
	très faible	< 2%	X	X	X	X	X	X
épaiss. sol	forte	> 2m	X	X	X	X	X	X
	moyenne	1.4 - 2m	X	X	X	X	X	X
	faible	0.8 - 1.4m		X		X		X
	très faible	< 0.8m						
perméa. sol	très faible	< 10mm/h						
	faible	10 - 20mm/h						
	moyenne	20 - 50mm/h						
	forte	50 - 500mm/h	X	X	X	X	X	X
	très forte	> 500mm/h	X	X	X	X	X	X
présence nappe	oui				X	X	X	X
	non		X	X				
prof. nappe	très forte	> 2m	////	////	X	X	X	X
	forte	1.5 - 2m	////	////	X	X	X	X
	moyenne	1 - 1.5m	////	////				
	faible	0.5 - 1m	////	////				
	très faible	0.3 - 0.5m	////	////				
	rédhibitoire	< 0.3m	////	////				
rejet superficiel	possible		////	////	////	////	////	////
	impossible		////	////	////	////	////	////
sous-sol	plutôt perméable		////	X	////	X	////	X
	plutôt imperméable		////	////	////	////	////	////
existence d'un captage	oui		////	////			X	X
	non		////	////	X	X		
distance du captage	satisfaisante	> 35m	////	////	////	////	X	X
	trop faible	<35m	////	////	////	////		

tableau 3-4 : règles de choix d'un épandage en tranchées

suggérant de bien s'assurer de la valeur de certains paramètres (on peut envisager de générer automatiquement de genre de commentaires à partir de l'analyse des réponses; cet aspect pourrait constituer un sous-système expert).

Il existe une certaine analogie entre la formalisation des règles telle que nous la présentons, sous forme de tableaux, et le stockage de l'information dans une base de données. En réalité, Moïse se distingue par plusieurs aspects d'une base de données ou plus généralement d'un S.G.B.D. (Système Gestionnaire d'une Base de Données) :

- Moïse gère une base de faits (en plus de la base de règles), ce qui est caractéristique de la technique "systèmes experts". On ne rencontre pas dans un S.G.B.D. la notion de déductions mémorisées par le système afin d'avancer vers une solution.
- On a dans Moïse la possibilité de regrouper, sous une même notion, un contexte illustré par une séquence de valeurs de paramètres. Cette "factorisation" des règles permet éventuellement de sortir du schéma strict de la "haie" (et donc des tableaux), en "bouturant", ce qui permet une analogie avec une représentation en arbre.
- Moïse propose un enchaînement de questions non figées, dépendant des réponses de l'utilisateur et guidé par les règles.
- Enfin il existe dans Moïse différentes stratégies d'exploration d'un tableau : mode déduction (avec recherche d'une ou plusieurs solutions simultanément) ou mode vérification.

Dans tous les cas, il est admis que la représentation sous forme de tableaux est largement utilisée dans le monde scientifique et professionnel, et c'est ce qui a fait une partie du succès des tableurs et des S.G.B.D. Aussi, il nous semble que le fait de la réutiliser dans Moïse ne peut être qu'un atout; le fait de l'intégrer à la technique "système expert" en fait un "plus".

4.2 EXEMPLE DE SESSION

Une fois le logiciel activé, l'écran de la figure 3-5 est affiché (voir paragraphe 3.2.4 et annexe A3 sur le mode opératoire de Moïse).

L'option "Expertise" permet d'accéder aux 2 modes de fonctionnement du logiciel : déduction et vérification (figure 3-11).

4.2.1 MODE DEDUCTION

Selon la stratégie retenue, le moteur explore chaque branche de la haie en regardant si les conditions sur chaque paramètre sont remplies; si la valeur du paramètre est inconnue, Moïse questionne et enregistre la réponse.

Dans l'exemple choisi (figure 3-10), si l'utilisateur répond que la pente du site est faible, le logiciel stockera :

oui(la pente du site, est, faible)
non(la pente du site, est, très faible)
non(la pente du site, est, moyenne)
non(la pente du site, est, forte)
non(la pente du site, est, très forte)

Dès qu'une condition n'est pas remplie, le moteur passe à la scrutation d'une autre branche.

A chaque niveau, il est proposé :

- une aide au menu;
- une explication sur la nature du paramètre et sur les moyens de l'appréhender (figure 3-11); le nombre de pages d'explication est limité à deux pour des raisons ergonomiques (la figure 3-12 n'en représente que la première). En outre, ces pages sont une véritable agrégation d'un certain nombre de notions qui pourraient chacune être développée dans une règle. Le concepteur peut à ce niveau ajuster "en profondeur" la technicité des règles, notamment en fonction du type d'utilisateur.
- une annulation pour revenir au niveau supérieur de l'arborescence des menus.

Peu à peu, la saisie des paramètres se poursuivant, dans un ordre non pré-déterminé puisque dicté par l'activation des règles, le logiciel parvient :

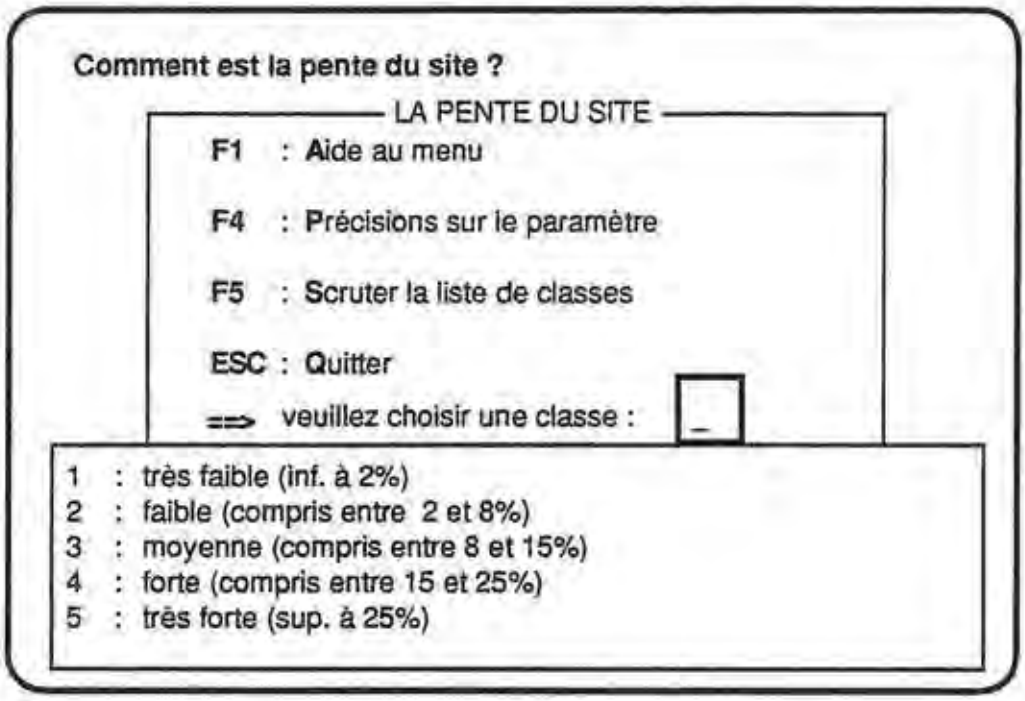


figure 3-11 : exemple de question sur un paramètre

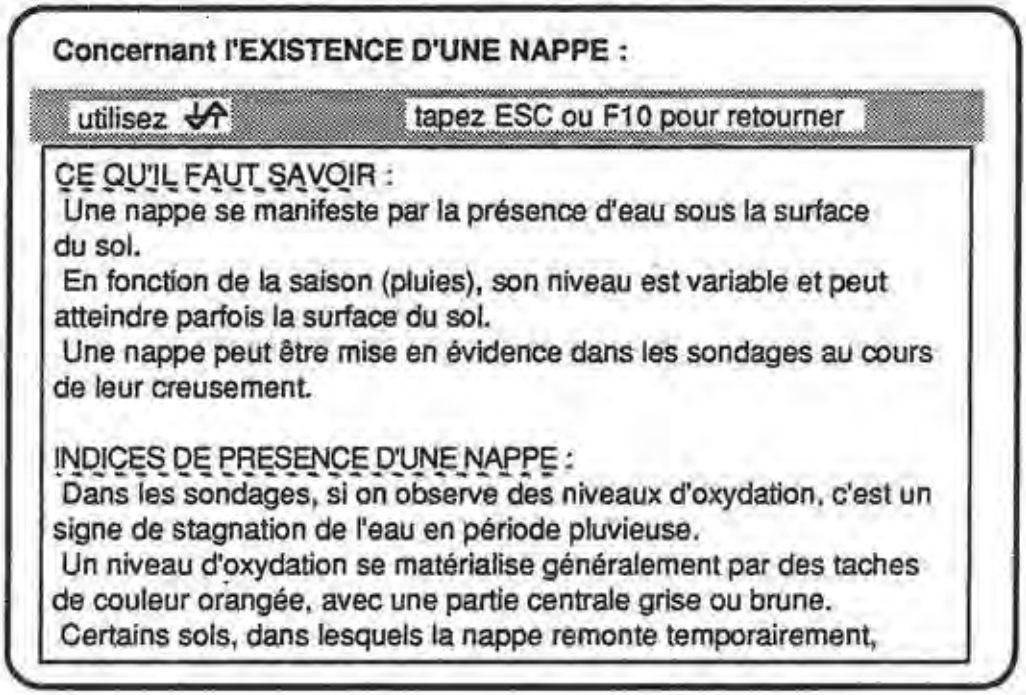


figure 3-12 : exemple de précisions sur le paramètre

- à une solution, accompagnée d'un commentaire (figure 3-13);
- à l'explication d'un échec (figure 3-14);
- au constat d'une base de règles déficiente, dans le cas où l'utilisateur a proposé une combinaison de valeurs de paramètres non répertoriée.

Ce dernier cas est particulièrement intéressant puisque son analyse permettra de compléter la base de règles en y incorporant cette combinaison absente.

Quand une solution est trouvée, le moteur peut, à la demande, poursuivre la déduction vers une autre solution, et afficher au fur et à mesure toutes les solutions possibles.

4.2.2 MODE VERIFICATION

L'instauration de l'option "vérification" entraîne l'écriture des règles selon un mode "*extensif*", par opposition au mode "*restrictif*".

Le mode "restrictif" suppose que les règles s'excluent mutuellement, c'est-à-dire que chaque combinaison complète menant à une solution n'apparaît qu'une seule fois dans la base de règles. Cette manière de faire, plus proche d'un graphe arborescent, est plus économe en volume de code.

Le mode "extensif" décrit pour chaque solution, toutes les combinaisons pour y parvenir. En assainissement individuel, dans la mesure où on peut mettre un tertre là où on met un épandage en tranchées, les combinaisons conduisant à un épandage en tranchées se retrouveront dans les règles menant à un tertre. De cette façon, le mode vérification restera cohérent et vérifiera cette compatibilité évoquée plus haut.

Un autre avantage du mode "extensif" est qu'il entraîne une meilleure indépendance des règles puisqu'on ne s'occupe que d'une solution à la fois, en cherchant à écrire toutes les combinaisons qui y mènent; il est alors également plus facile de vérifier que toutes les combinaisons sont concernées.

Le mode "vérification" débute par l'affichage de toutes les solutions possibles parmi lesquelles l'utilisateur choisit celle qu'il veut vérifier (figure 3-15) (la touche F5 permet de faire défiler dans la fenêtre inférieure, si elle est trop petite, l'ensemble des solutions).

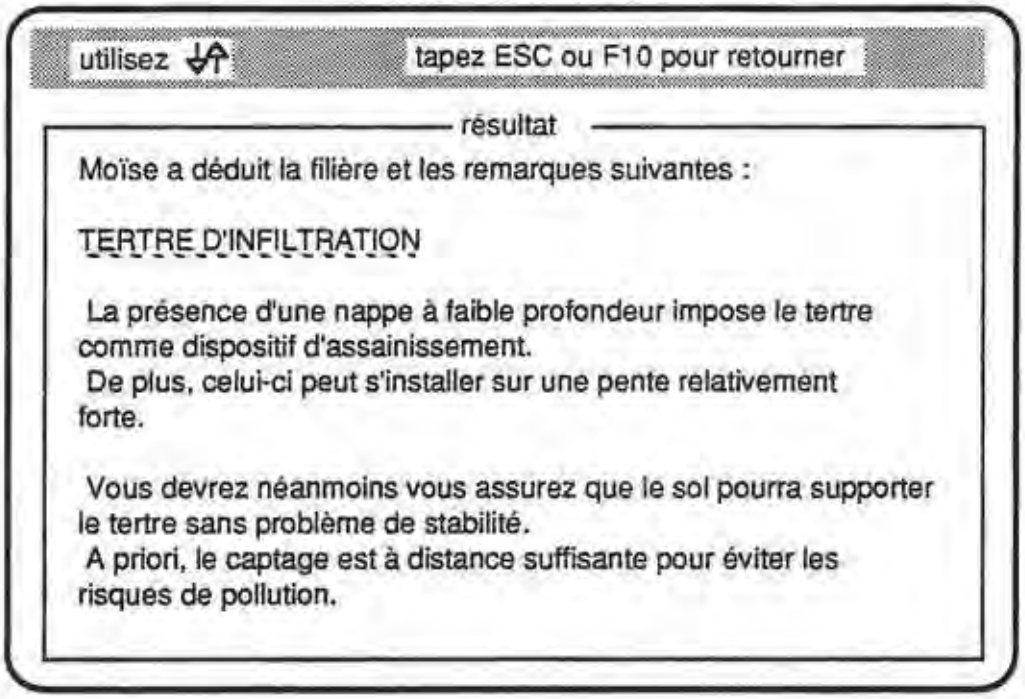


figure 3-13 : exemple de solution avec son commentaire

figure 3-14 : exemple d'échec dans la déduction

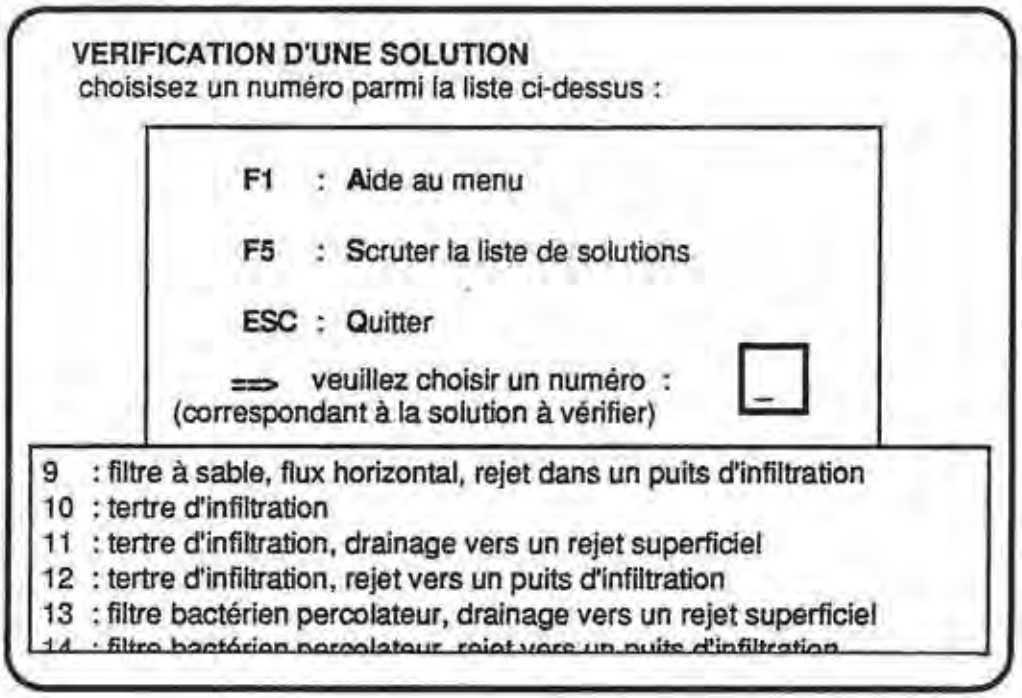


figure 3-15 : liste des solutions possibles

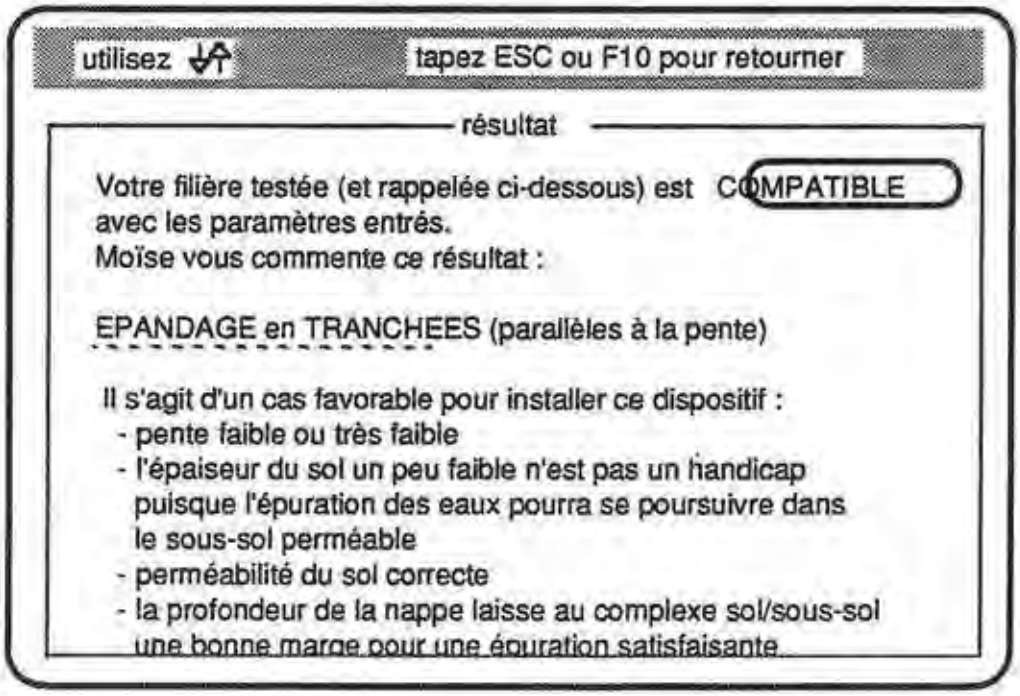


figure 3-16 : exemple de succès du mode "vérification"

Une fois le dispositif choisi, le moteur explore, comme avec le mode "déduction", toutes les branches d'un seul arbuste de la haie. Le mode "vérification" est donc en principe plus rapide que le mode "déduction" puisqu'il concerne moins de règles.

En conclusion, la solution peut être :

- vérifiée, et accompagnée d'un commentaire sur le contexte (le même qu'en mode "déduction" (figure 3-16);
- non vérifiée; dans ce cas, il n'y a pas d'explication sur la cause de l'échec car son identification entraînerait la mise au point d'un grand nombre de règles, ce qui serait incompatible avec le souci de clarté et de maintenance aisée du logiciel. En contrepartie, Moïse offre la possibilité, par ses passages souples entre fonctionnalités, de basculer en mode "déduction" sans toucher à la base de règles et donc de proposer un dispositif compatible avec le contexte, s'il existe.

4.3 BASES DE REGLES COMPLEMENTAIRES

Ce paragraphe décrit des bases de règles proposant d'autres services que le choix d'un dispositif d'assainissement et qui peuvent être soit couplées avec ce premier service (dimensionnement, cohérence de la base de faits...), soit proposées comme une autre option dans le cadre plus général d'un logiciel (station de travail) relatif à l'assainissement individuel.

Ce qui est intéressant, c'est que ces bases de règles complémentaires sont conçues sur le même modèle "en haies" que précédemment. C'est dire que les modifications à apporter au moteur seront minimes et que l'on va conserver les mêmes avantages : lisibilité, simplicité de la maintenance et évolution possible vers une maintenance dynamique.

4.3.1 DIMENSIONNEMENT D'UN DISPOSITIF

Le dimensionnement d'un épandage en tranchées, filtre à sable, tertre ou filtre bactérien percolateur est proportionnel au nombre de chambres de l'habitation dont il faut assainir les eaux usées (certains auteurs remplacent le nombre de chambres par le nombre d'habitants; encore une fois, notre propos n'est pas de prendre parti dans ce

débat; nous verrons que la structure proposée permet de prendre en compte les deux approches).

Mis à part le filtre bactérien percolateur, les 3 autres dispositifs peuvent aussi dépendre des caractéristiques du sol en place, ce qui est surtout vrai pour l'épandage en tranchées.

Nous avons retenu l'équation de dimensionnement suivante :

$$\begin{cases} Dim = Surface\_minimale + \alpha\ (\ Nombre\_chambres - \beta) \\ Dim = Surface\_minimale \text{ si } Dim < Surface\_minimale \end{cases}$$

(dans le cas du filtre bactérien percolateur, il faut remplacer la notion de surface par celle de volume).

Trois paramètres (Surface_minimale, α, β) qui sont au moins fonction des caractéristiques du terrain en place et du dispositif à installer, vont donc pondérer le dimensionnement du dispositif.

Dans ce cas, le moteur devra aller sélectionner les bonnes valeurs des paramètres (Surface_minimale, α, β) en fonction du contexte géo-physique et du dispositif choisi. Or toute ces données sont directement accessibles : le contexte est dans la base de faits, le dispositif est lui une déduction qu'il suffit également de stocker dans la base de faits. Aussi, les modifications à apporter au moteur seront minimes puisqu'il s'agira de rajouter, suite à la déduction, un prédicat qui explore la base de faits, en déduise les paramètres adéquats en fonction de la filière et du contexte et calcule ensuite le dimensionnement du dispositif proposé.

La mise au point des paramètres (Surface_minimale, α, β) va se faire sur des tableaux du même type que ceux utilisés pour le choix d'un assainissement individuel, et qui utilisent le même symbolisme. Nous présentons dans les tableaux suivants (3-5) des exemples de paramètres pour dimensionner un épandage en tranchées (pour les autres dispositifs, certains auteurs admettent des paramètres indépendants des caractéristiques du sol, d'autres peuvent prendre en compte la distance d'un captage par exemple, en prenant des marges de sécurité si ce captage est proche; dans tous les cas, on voit qu'on peut intégrer toutes les dépendances et notions désirées).

Le prédicat déduisant la valeur des paramètres est pratiquement identique à celui explorant d'une façon générale une "haie".

PARAMETRES de DIMENSIONNEMENT							
			1	2	3	4	5	6
	très faible	< 10mm/h						
	faible	10 - 20mm/h						
perméa.	moyenne	20 - 50mm/h	X		X		X	
sol	forte	50 - 500mm/h		X		X		X
	très forte	> 500mm/h						
présence	oui		X	X	X	X		
nappe	non						X	X
	très forte	> 2m			X	X	////	////
	forte	1.5 - 2m	X	X			////	////
prof.	moyenne	1 - 1.5m					////	////
nappe	faible	0.5 - 1m					////	////
	très faible	0.3 - 0.5m					////	////
	rédhibitoire	< 0.3m					////	////
Epandage en tranchées parallèles à la pente								
		Smin	120	65	80	45	80	45
		alpha	40	20	25	15	25	15
		beta	3	3	3	3	3	3
Epandage en tranchées perpendiculaires à la pente								
		Smin	140	80	90	50	90	50
		alpha	45	25	30	15	30	15
		beta	3	3	3	3	3	3

tableau 3-5 : dimensionnement d'un épandage en tranchées

4.3.2 CONTROLE DE LA COHERENCE DE LA BASE DE FAITS

Il s'agit, dans ce module, de contrôler la cohérence des valeurs et des assertions qui sont communiquées par l'utilisateur au système expert.

Il faut d'une part éliminer ainsi des erreurs dues au manque de qualification ou de compréhension de l'utilisateur, mais d'autre part, mettre aussi en évidence les apparitions de combinaisons de deux ou plusieurs valeurs qui, sans être forcément incompatibles, sont d'occurence suffisamment rare.

Dans tous les cas, l'activation de ces règles de contrôle de cohérence doit plutôt être considérée comme une alerte, que comme un refus par le système de la prise en compte d'une valeur.

C'est dans cet esprit que nous avons testé les règles reportées dans le tableau 3-6 et commentées dans le tableau 3-7.

Là encore, nous utilisons une structure "en haies" qui nous permet de réutiliser le moteur précédent avec des modifications mineures : il suffit d'adjoindre un prédicat testant les règles de cohérence avant l'enregistrement dans la base de faits, avec possibilité de renoncer à cet enregistrement.

4.3.3 MODULE DE MAINTENANCE D'UNE INSTALLATION

Dans les deux exemples précédents, on a vu qu'il fallait effectuer de légères modifications au moteur pour intégrer ces fonctionnalités supplémentaires, moyennant quoi, c'est toujours la même base de faits qui est utilisée et qui permet la communication entre les fonctions : cette base sera tour à tour alimentée ou utilisée par une même fonction .

Dans le cas du module de maintenance, le moteur devra ou non être adapté selon la stratégie que l'on veut développer.

Mettons nous dans le cas d'un public non initié, c'est-à-dire un particulier qui a un problème avec son assainissement et qui désire en identifier la cause (on verra plus loin que cette hypothèse est réaliste dans la mesure où ce particulier peut accéder au logiciel grâce à une version "Minitel").

REGLES de COHERENCE des FAITS								
			1	2	3	4	5	6	7
	très forte	> 25%		////	////	////	////	////	
	forte	15 - 25%		////	////	////	////	////	X
pente	moyenne	8 - 15%		////	////	////	////	////	
	faible	2 - 8%		////	////	////	////	////	
	très faible	< 2%	X	////	////	////	////	////	
	forte	> 2m	////	X	X	////	////	////	X
épaiss.	moyenne	1.4 - 2m	////		X	////	////	////	X
sol	faible	0.8 - 1.4m	////		X	////	////	////	
	très faible	< 0.8m	////		X	////	////	////	
	très faible	< 10mm/h	////	X	X	////	////	X	////
	faible	10 - 20mm/h	////	X	X	////	////	X	////
perméa.	moyenne	20 - 50mm/h	////			////	////		////
sol	forte	50 - 500mm/h	////			////	////		////
	très forte	> 500mm/h	////			////	////		////
présence	oui		////	X	X		////	////	////
nappe	non		////			X	////	////	////
rejet	possible		X	////	////	////	////	////	////
superficiel	impossible			////	////	////	////	////	////
sous-sol	plutôt perméable		////	////		////		////	////
	plutôt imperméable		////	////	X	////	X	////	////
existence	oui		////	////	////	X	X	X	////
d'un captage	non		////	////	////				////

tableau 3-6 : règles de cohérence de la base de faits

1 : le rejet superficiel est possible alors que la pente du site est très faible.

Ces 2 faits ne sont pas incompatibles mais demandent à être vérifiés. Ils peuvent également mettre en évidence de futurs problèmes face aux conditions d'écoulement dans l'évacuation des eaux drainées.

2 et 3 : il s'agit dans ces deux cas de relever le fait qu'il y a une nappe alors que le sol (ou le sous-sol) est imperméable.

On peut en profiter pour poser la question des conditions météorologiques précédent l'observation de la nappe : s'il pleuvait, ne s'agit-il pas en fait d'eau de pluie retenue par des niveaux imperméables plus ou moins superficiels que l'utilisateur prend pour une nappe ?

4, 5 et 6 : l'existence d'un captage d'eau souterraine dans les environs se combine avec soit une absence de nappe, soit un sol profond imperméable.

Il faut donc bien vérifier que le captage est suffisamment éloigné ou que le niveau de la nappe dans le puits est assez bas pour ne pas tenir compte de l'existence de l'eau souterraine.

7 : une pente forte est-elle compatible avec une épaisseur du sol moyenne ou forte ?

En fait, il s'agit ici de bien s'assurer que l'utilisateur ne fait pas de confusion sur ce que l'on appelle un sol, qu'il pourrait confondre avec d'autres structures. En effet, généralement, l'épaisseur d'un sol diminue avec la pente de terrain, à cause de l'érosion.

tableau 3-7 : commentaires sur les règles de cohérence de la base de faits.

Il est évident qu'il peut y avoir plusieurs raisons au mauvais fonctionnement du dispositif.

Aussi, le moteur tel qu'il est écrit pour le choix d'un assainissement individuel est inadapté dans la mesure où il s'arrête dès qu'il a trouvé une solution. Ou bien, il faut écrire les règles de telle façon qu'elles explorent toutes les combinaisons de pannes possibles (qui est un concept différent de celui évoqué précédemment qui était d'écrire toutes les combinaisons de paramètres menant aux solutions; ici, il faut combiner les solutions en plus des paramètres). Les règles doivent alors être classées en présentant d'abord les plus complexes, et donc en premier, celle qui introduit la combinaison de toutes les origines possibles d'un mauvais fonctionnement.
Ceci est une première façon permettant de ne pas toucher au moteur. Elle a l'inconvénient d'être contraignante pour les règles qui perdent de leur indépendance vis à vis des différents types de pannes possibles.

Une autre stratégie va consister a écrire des règles indépendantes (chacune identifiant une cause de dysfonctionnement) et de demander à l'utilisateur de chercher toutes les solutions possibles (voir en Annexe A3 la fonction F5 du menu "déduction"). Là encore, on ne touche pas au moteur mais on ne peut pas être certain que l'utilisateur ira jusqu'au bout de toutes les solutions.

Dans ce cas, il apparaît que la meilleure approche consiste à modifier le moteur pour qu'il fasse apparaître toutes les origines de pannes d'un coup (en accord avec les réponses de l'utilisateur), les règles étant écrites de façon indépendante (on verra que c'est également la solution retenue dans le cas d'une application visant à informatiser les guides de choix en instrumentation d'analyse chimique).

La base de règles que nous présentons en exemple à la suite [tableaux 3-8 et 3-9] tente de régler *les problèmes d'odeur* qui peuvent apparaître lors du fonctionnement d'une installation. Nous limitons l'analyse du phénomène au niveau des fonctions de collecte et de pré-traitement, en n'indiquant dans cet exemple que les cas les plus fréquents (d'après les documents manuscrits des spécialistes de l'agence de bassin Loire-Bretagne). Il s'agit simplement d'illustrer les propos précédents.

On remarquera ici que le mode "vérification" est inutile au regard des règles développées. En effet, ce mode de fonctionnement est crédible si on a un nombre conséquent de classes de paramètres engendrant de multiples combinaisons. Or, dans ce

paramètre	qualificatif	1	2	3	4	5	6	7	8
odeurs	plutôt à l'intérieur	x	x	x	x				
	plutôt à l'extérieur	x		x	x	x	x	x	x
présence d'un siphon à l'entrée de la fosse septique	oui			x	/ /	/ /	/ /	/ /	/ /
	non	x	x		/ /	/ /	/ /	/ /	/ /
ventilation de la fosse septique	absente	x		x				/ /	/ /
	indépendante		x		x	x	x	/ /	/ /
	reliée au réseau de collecte		x	x	x		x	/ /	/ /
diamètre de la ventilation	>100mm	/ /	/ /	/ /		/ /	/ /	/ /	/ /
	<100mm	/ /	/ /	/ /	x	/ /	/ /	/ /	/ /
hauteur de la ventilation	>4m	/ /	/ /	/ /	/ /		/ /	/ /	/ /
	<4m	/ /	/ /	/ /	/ /	x	/ /	/ /	/ /
la ventilation dépasse du toit de l'habitation	oui	/ /	/ /	/ /	/ /	/ /		/ /	/ /
	non	/ /	/ /	/ /	/ /	/ /	x	/ /	/ /
joint au niveau de la fosse septique	absent	/ /	/ /	/ /	/ /	/ /	/ /	x	/ /
	en béton	/ /	/ /	/ /	/ /	/ /	/ /	x	/ /
	en plâtre	/ /	/ /	/ /	/ /	/ /	/ /		/ /
joint au niveau du préfiltre	absent	/ /	/ /	/ /	/ /	/ /	/ /	/ /	x
	en béton	/ /	/ /	/ /	/ /	/ /	/ /	/ /	x
	en plâtre	/ /	/ /	/ /	/ /	/ /	/ /	/ /	

tableau 3-8 : règles de maintenance d'un assainissement individuel pour des problèmes d'odeurs au niveau des fonctions de collecte et de prétraitement.

1 : mettre un siphon à l'entrée de la fosse septique et une ventilation indépendante.
2 : mettre un siphon à la fosse septique et s'assurer de la qualité de la ventilation.
3 : mettre une ventilation indépendante à la fosse septique.
4 : la règlementation prévoit un diamètre minimum de 100mm pour la ventilation de la fosse septique.
5 : la règlementation prévoit une hauteur de colonne de ventilation de 4m au minimum.
6 : des micro-courants d'air rabattent probablement les mauvaises odeurs sortant de la colonne de ventilation; il est préférable que celle-ci dépasse le faîte du toit de l'habitation et qu'elle soit placée de façon adéquate par rapport aux vents dominants.
7 : les joints de la fosse septique doivent être en plâtre.
8 : les joints du préfiltre doivent être en plâtre.

tableau 3-9 : commentaires sommaires associés aux règles du tableau ci-dessus

cas, les valeurs des paramètres décrivent en fait souvent l'origine directe d'un dysfonctionnement se traduisant par une odeur.

Ainsi, vouloir vérifier si l'odeur provient d'un défaut du joint du préfiltre revient à poser la question sur l'existence ou la nature du joint préfiltre (plâtre ou autre); tout ceci étant très explicite.

4.3.4 CONCLUSION

La structure que nous obtenons est intéressante, car très modulaire : le moteur peut traiter différentes bases de règles, soit dans un enchaînement automatisé (choix d'un dispositif parallèlement au contrôle de cohérence de la base de faits, puis dimensionnement), soit à la demande de l'utilisateur (maintenance).

Il y a indépendance entre les modules, et à l'intérieur de ceux-ci, on retrouve la modularité au niveau de chaque "solution", constituée de règles indépendantes.

Les liens entre les bases de règles sont assurés par la base de faits dont les constituants générés doivent pouvoir être interprétés si nécessaire par chaque module.

Cette structure permet d'envisager la mise au point d'outils opérationnels.

5 EVOLUTION VERS UN OUTIL OPERATIONNEL

On observe que la dernière base de règles est peu "experte", qu'elle présente un caractère trivial. Cet aspect tient aux utilisateurs visés : ici, des particuliers possédant en général peu de connaissances en assainissement. Aussi, la base de règles est d'un niveau d'expertise bas, faisant surtout appel à des règles "de bon sens". Cet aspect concourt à une prise de conscience des utilisateurs peu motivés par l'entretien et la mise en conformité de leur installation. La première action à entreprendre auprès d'eux recouvre une action d'information et de formation.

On notera quand même que les problèmes de maintenance ne se réduisent pas à ceux des mauvaises odeurs au niveau de la collecte et du prétraitement, et que cette base de règles peut donc évoluer jusqu'à devenir suffisamment exhaustive pour être opérationnelle et traiter d'autres problèmes de nuisances : résurgence, colmatage, réhabilitation d'anciens systèmes ...

Cependant, il est clair qu'en réfléchissant à ce qu'il faut mettre dans cet outil, il faut aussi se préoccuper parallèlement de son mode de diffusion.

C'est un problème qui se ramène souvent à un type d'ordinateur, un système d'exploitation... , ce qui est vrai tant qu'on s'intéresse à des personnes équipées du point de vue informatique.

Dans notre cas, une catégorie des utilisateurs concernés (particuliers, maçons, entrepreneurs...), n'ont pas à priori d'équipement informatique, et n'envisagent pas d'en acquérir dans le cadre d'une aide en assainissement individuel.

Les circuits habituels de distribution de logiciels sont donc ici inadaptés pour atteindre "Monsieur tout-le-monde" qui doit mettre en place un assainissement pour son habitation et l'entretenir.

Ainsi, l'évolution vers un outil opérationnel va emprunter deux voies différentes, selon le type des utilisateurs, ceux-ci étant de plus différenciés par leur niveau d'équipement informatique :

- les concepteurs qualifiés en assainissement, les bureaux d'études ou les services de l'administration de l'état pour lesquels l'adaptation de l'outil sera :
  - matériel : proposer un support suffisamment répandu;
  - logiciel : fournir un outil malléable, du moins en ce qui concerne la formalisation des connaissances à y inclure.
- les installateurs peu ou pas qualifiés en assainissement (maçons, petits entrepreneurs) et les particuliers pour lesquels nous avons développé une interface télématique, rendant le logiciel accessible depuis un Minitel, via le réseau téléphonique.

En partant de la conception d'un outil - Moïse -, on voit se dessiner deux schémas de valorisation, tant technique que fonctionnel, compte tenu des contraintes à la fois sociales, professionnelles et informatiques des utilisateurs potentiels.

Nous allons, dans le paragraphe suivant, développer plus particulièrement l'aspect télématique.

Il apparaît un intérêt technique (l'accès d'un large public à un logiciel performant) et scientifique de la télématique qui peut permettre la mise en place d'une base

d'expérimentation. Cette base autorise un retour d'informations rapide, interactif, concourant à un ajustement du système et une maintenance de la connaissance. En s'adressant à des professionnels - comme nous l'avons fait dans les tests de la version télématique de Moïse - on peut par exemple tenter une homogénéïsation des règles en confrontant les critères de chaque spécialiste.

5.1 EVOLUTION VERS UN SERVICE TELEMATIQUE

5.1.1 PRINCIPE

Nous avons la chance de posséder en France un réseau téléphonique suffisamment fiable pour assurer un transport d'information numérisée.

Envoyée par un ordinateur, cette information peut-être reçue sur un autre ordinateur, ou sur un terminal quelconque; pour peu que ce dernier soit équipé d'un clavier, il peut y avoir circulation d'information dans l'autre sens : c'est le Minitel.

Distribué gratuitement par les services de l'état à plusieurs millions d'exemplaires, tout abonné au service téléphonique peut posséder un Minitel. Celui-ci se connectant sur la prise téléphonique, tout Minitel peut communiquer avec n'importe quel ordinateur connecté au réseau téléphonique.

Un certain nombre de services télématiques ont donc vu le jour, assurant des fonctions plus ou moins appréciées. Celles-ci sont souvent de conception informatique peu élaborée puisque se contentant dans leur majorité d'être des suites de menus gérant des pages d'écran. Plus récemment, des professionnels se sont proposé de réaliser des applications faisant appel à des programmes performants, notamment des systèmes experts (tableau 3-10).

C'est dans cette catégorie que pourrait s'inscrire Moïse.

5.1.2 ASPECT TECHNIQUE [BEAUNE, GRAILLOT, CRES 1989]

Une des principales contraintes que nous voulions respecter dans la mise en œuvre d'une version télématique concerne le matériel où est implanté le logiciel.

Notre souci a été de montrer qu'il était possible de réaliser cette version sans investissement lourd, uniquement à partir d'un matériel répandu (type micro-ordinateur IBM), et d'acquisition de lociciels d'un coût raisonnable pour assurer la

SECTEUR	NOM	ORIGINE	FONCTION
Medecine	DIABETO SESAM-DIABETE SAM-ASTHME anonyme SM-champignons MEX DIAGSES/ABSES	Univ. Toulouse Hôtel-Dieu, CHU Pitié Hôpital Necker MC-TEL Univ. Toulouse et P.M. Curie, Canal 4 Mémoires d'experts, Serveur Scalaire Univ. Lille,groupe SES	Diagnostic du diabète Diagnostic du diabète Evolution de l'asthme Medecine générale Identification de champignons Médecine d'urgence Diagnostic et thérapeu-thie en infectiologie
Industrie	MULTIDIAG SEXSY "Réparation autos" XSEL CORNELIUS	Cap Gemini Sogeti Service S.A.(Philips) Infogrammes DEC Merlin Gerin	Maintenance Diag. de compact-discs Aide à l'automobiliste Aide à la vente Maintenance onduleurs
Agriculture	SEPV (17 syst. exp.) IVRAIE COUNSELLOR	INRA Acta ISI (G-B)	diag. pathologie végétale diag. grandes cultures Gestion des récoltes
Administra-tion	SYMA SAOR CARL Créations	Cognitech,SPA(Armée) Cognitech,IGIRS NDV	Assistance à la liquidation des pensions Aide dossiers retraites Aide création entreprise
Banque de données	"Pages jaunes" "Petites annonces"	ERLI Le Monde	Annuaires professionnels Recherche d'emploi
Traduction	MITRAD	Gachot	Traduction multi-langues
Loisirs	BAO COCKTAIL	Nouvel Obs., Ackia Infogrammes,Play Boy	enseignement du bridge Composition de cocktails

[Source : Etude Cognitech 1988 d'après France Telecom/ lettre de TELETEL n°16]

tableau 3-10 : Exemples d'applications d'Intelligence Artificielle sur réseaux

connexion avec le R.T.C. (Réseau Téléphonique Commuté), via un modem (modulateur-démodulateur) ordinaire du commerce.

Cette première version n'assure qu'un accès monovoie (1 utilisateur à la fois), mais l'extension vers un accès multivoie est possible.

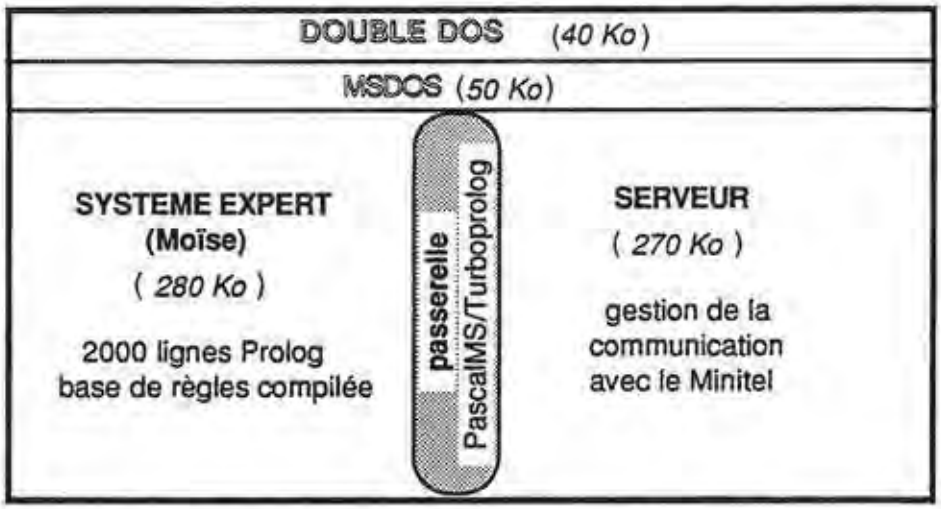
L'écriture d'un serveur télématique (gestion des pages vidéotex, des boîtes à lettres pour la fonction messagerie, des codes utilisateurs, des temps de connexion ...) est un travail long et délicat. Aussi, nous avons préféré nous tourner vers un logiciel du commerce capable d'assurer un service télématique et suffisamment ouvert pour nous permettre de gérer une base de connaissances. Ce serveur (Hostel de la société GOTO Informatique) acceptant un interfaçage avec des langages évolués (Pascal, C ...), nous avons étendu cette possibilité jusqu'à Turboprolog en développant une passerelle PascalMS/Turboprolog.

L'implantation simultanée des deux tâches (serveur télématique et système expert) se fait grâce à un sur-système d'exploitation (DoubleDOS) qui, placé au dessus de MSDOS, met à notre disposition 2 tâches MSDOS.

Compte tenu des contraintes mémoires des différents logiciels, il reste environ 300 Ko (kilo-octet) pour le système expert. Cet inconvénient peut être éventuellement levé en utilisant d'autres sur-systèmes d'exploitation (comme PCMOS 386 pour compatible IBM 386 qui autorise environ 580 Ko par tâche). De plus, on retrouve ici un des intérêts d'avoir des bases de règles non compilées, puisque celles-ci n'occupent pas en permanence de la place en mémoire centrale et ne sont chargées que lorsque le besoin s'en fait sentir.

La figure 3-17 illustre l'organisation générale de la version télématique de Moïse.

En ce qui concerne la connexion d'un micro-ordinateur sur le R.T.C., d'autres possibilités très simples existent, qui permettent notamment de commander le micro-ordinateur à partir d'un Minitel. Malheureusement, ces solutions escamotent complètement l'aspect "serveur", et n'autorisent pas une utilisation tournée vers un large public.



organisation des logiciels en version télématique

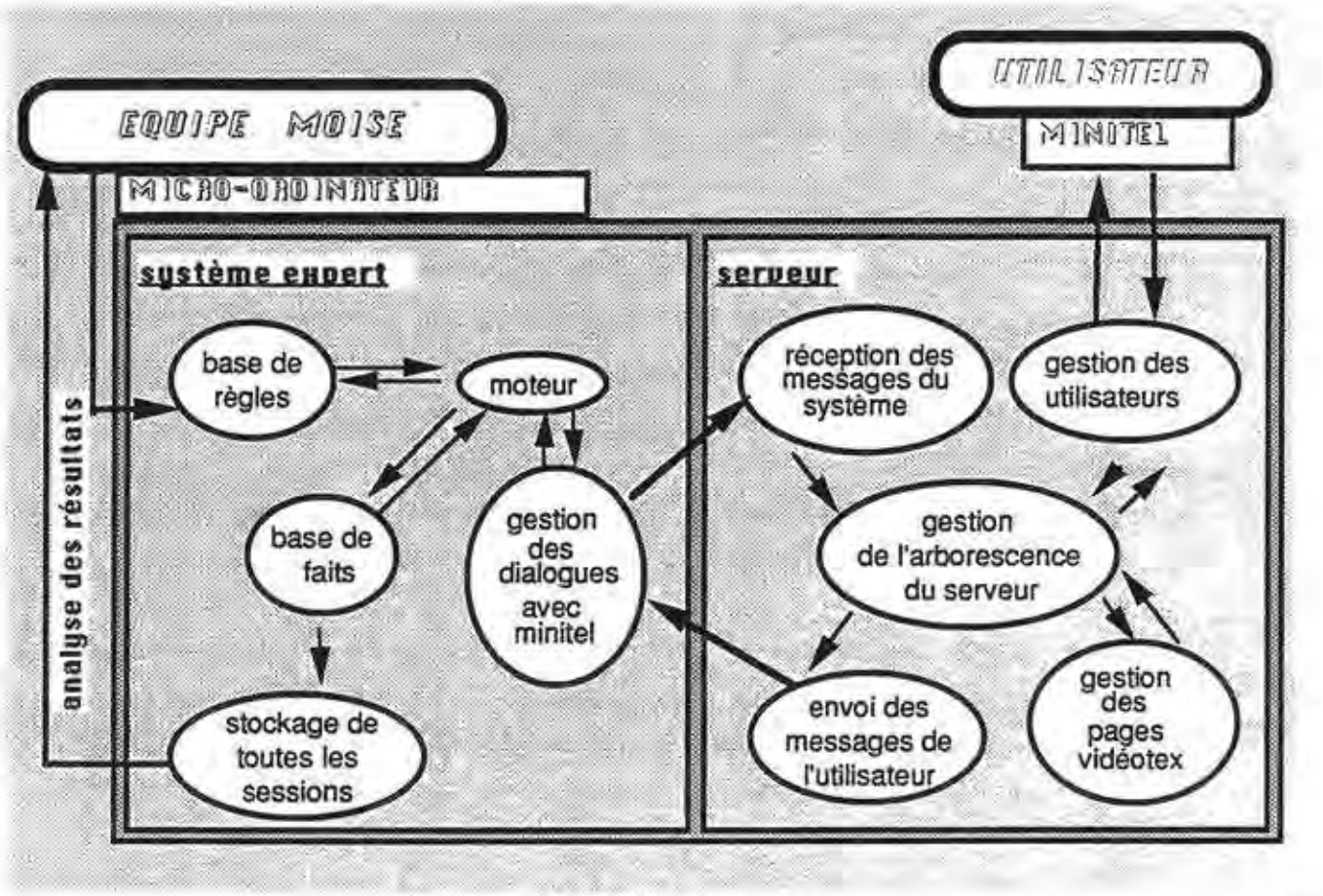


schéma du fonctionnement

figure 3-17 : Organisation générale de la version télématique de Moïse

5.1.3 EXEMPLE D'UNE SESSION SUR MINITEL

L'application a été mise en service pendant plusieurs semaines à l'usage de quelques spécialistes de la télématique ou de l'assainissement; d'autres connexions correspondent à une information de la part d'utilisateurs ou de concepteurs potentiels de systèmes télématiques analogues (50 personnes en tout environ, voir annexe A5).

Ce test a permis de vérifier la robustesse informatique du logiciel, et a montré que cette idée de l'association "système expert/Minitel" était appréciée et apte à faire avancer le bon emploi de la technique de l'assainissement individuel.

Le service télématique commence par une page "SOMMAIRE" qui contient 4 options :

- 1 - informations
- 2 - messagerie
- 3 - expertise en assainissement
- 4 - déconnexion

L'option '4' (déconnexion) permet de sortir du logiciel; celui-ci se remet alors en attente d'un autre appel téléphonique. Cette option correspond à une "sortie propre". Cependant, la touche "déconnexion" du Minitel permet à l'utilisateur d'interrompre la communication téléphonique à n'importe quel moment du déroulement du programme. Ce dernier doit donc être capable, à tout instant, de se mettre en attente d'une nouvelle connexion.

Sur la version micro-ordinateur, cette possibilité n'est pas prévue. Seule la touche "ESC" permet de remonter dans l'arborescence des menus jusqu'au retour au système d'exploitation.

Cet exemple illustre que le moteur assurant la communication télématique est différent de celui de la version micro-ordinateur. Il y a bien d'autres différences qui prennent en compte la spécificité Minitel, et qui font des 2 moteurs, qui assurent globalement les mêmes fonctions, deux programmes totalement différents.

Les options '1', '2' et '4' sont en fait des fonctions inhérentes au service télématique et concernent essentiellement une arborescence d'écrans pré-programmés.

Après identification de l'utilisateur, l'option '3' permet l'accès au système expert pour le choix d'un assainissement individuel et déclenche l'ensemble de la technique décrite précédemment.

On retrouve les deux types de consultation prévus (figure 3-18A) :

- rechercher une solution (mode déduction);
- vérifier une solution (mode vérification).

Toutes les options de gestion de la base de faits, à commencer par son stockage, sont éliminées pour ne pas compliquer la démarche de l'utilisateur dont le profil-type est "une personne non initiée".

Supposons que l'option '1' (de la figure 3-18A : mode déduction) ait été choisie, la session débute par une série de questions concernant les caractéristiques de la parcelle.

La démarche est ici la même sur les deux logiciels, au niveau de l'exploration des règles et de la mémorisation des réponses dans la base de faits. La version Minitel décompose une question en plusieurs étapes afin de mieux guider : si le paramètre peut être quantifié, le logiciel demande si l'utilisateur possède une valeur numérique (figure 3-18B) :

- si oui, quelle est cette valeur, qui sera traitée et intégrée dans une des classes d'appartenance du paramètre (figure 3-18C);
- si non, le système propose les différentes classes associées au paramètre étudié (figure 3-18D).

A chaque écran, l'abandon provisoire (touche "RETOUR") permet à l'utilisateur d'accéder à la messagerie pour poser une question ou faire une remarque, puis de revenir là où il en était dans la session d'expertise. Cette possibilité est intéressante quand on a à faire à des spécialistes qui peuvent aider à améliorer le service.

Pour chaque paramètre, une aide à la réponse informe l'utilisateur sur la nature du paramètre et le guide éventuellement vers les services administratifs ou les entreprises pouvant lui offrir le renseignement demandé. Ces aides sont communes aux deux logiciels.

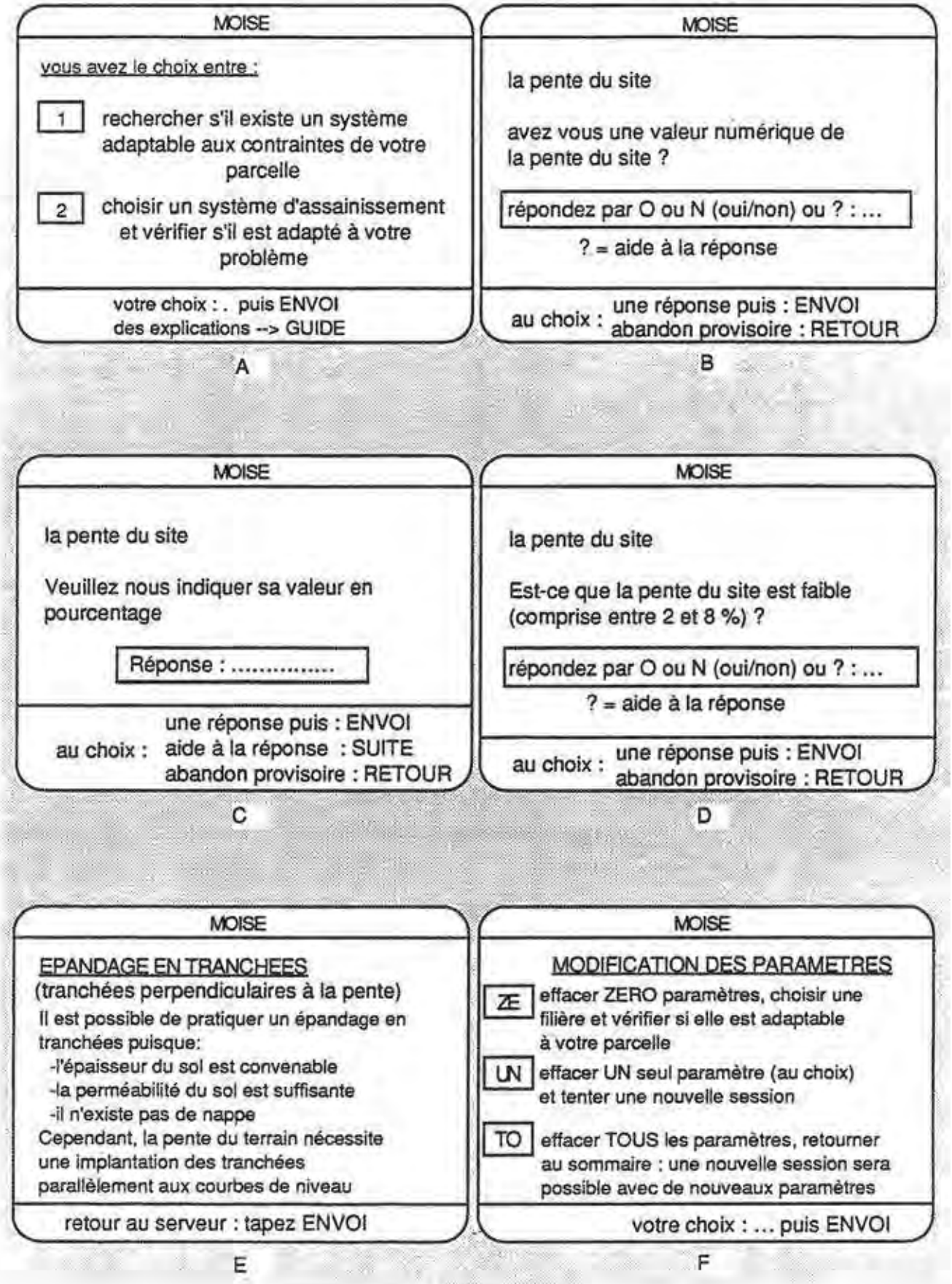


figure 3-18 : exemples d'écrans Minitel

Une fois les paramètres introduits, le système fournit, quand cela est possible, une solution et justifie le choix de la filière par un commentaire explicatif (figure 3-18E), suivi d'un schéma indiquant le principe de fonctionnement du dispositif à installer. Le schéma a pour but de tester les possibilités graphiques du Minitel, et de minimiser les textes explicatifs sur la nature d'un dispositif, souvent rébarbatifs (figure 3-19).

Les règles annexes d'analyse de l'échec de l'assainissement individuel sont également activées si nécessaire.

Enfin, 3 possibilités sont ensuite offertes (figure 3-18F) :

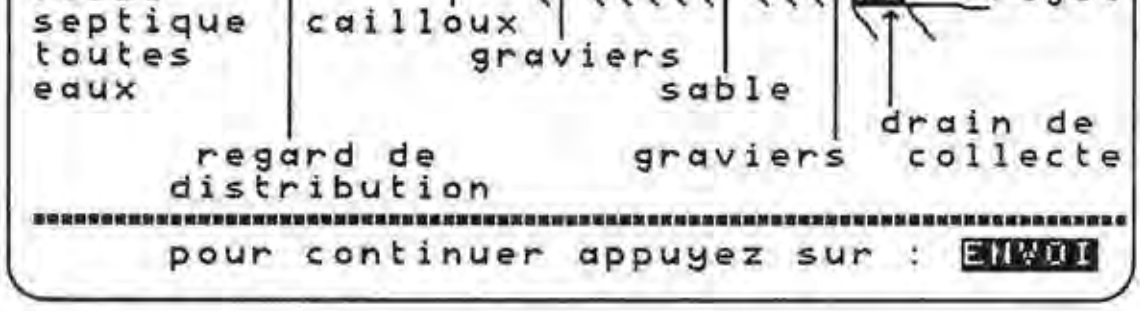
- l'option 'UN' permettant de tester la sensibilité d'une filière par modification d'un seul paramètre;
- l'option 'ZE' qui permet de passer en mode vérification sans affecter la base de faits, et de vérifier une filière au choix;
- l'option 'TO' qui efface tout et renvoie au "SOMMAIRE" au début de l'application.

La version Minitel est donc moins riche, et la démarche est plus décomposée. Un nombre moindre de réponses possible est proposé à chaque question afin de ne pas perdre l'attention de l'utilisateur. Ces facteurs jouent vers une simplification dans le déroulement du logiciel, eu égard au profil-type de l'utilisateur visé : une personne non initiée à l'assainissement, ni, souvent, à la manipulation d'un ordinateur.

Il faut noter que la base de règles utilisée par la version micro-ordinateur et la version télématique est strictement la même : c'est le même fichier.

5.1.4 SCHEMA D'UN OUTIL REGROUPANT LES ELEMENTS PRECEDENTS

Nous venons de montrer la faisabilité, à partir de moyens techniques limités, de l'adaptation d'une version télématique de Moïse, en utilisant la structure modulaire évoquée précedemment (un moteur qui traite différentes bases de règles), mais cette fois ci, avec deux moteurs dont un adapté au Minitel, qui utilisent la même base de règles.



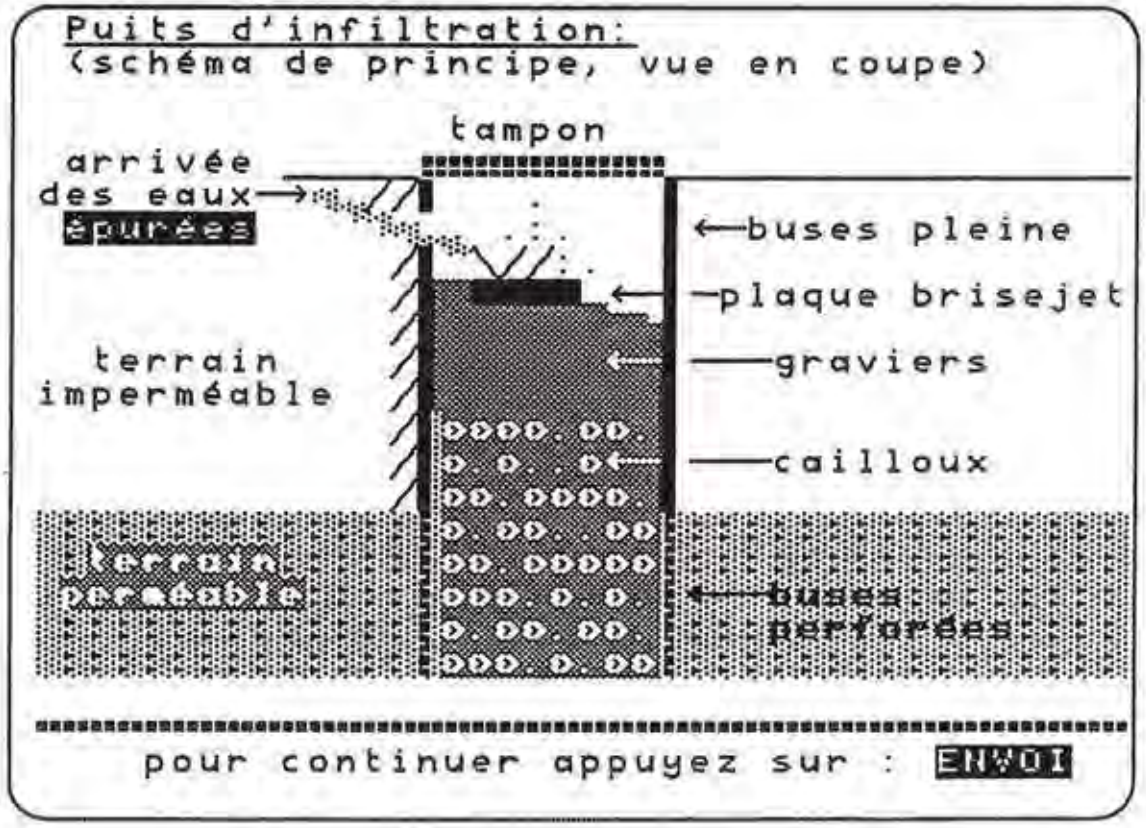


figure 3-19 : exemples de shémas obtenus sur Minitel

Cet aspect est très intéressant car il permet d'envisager une mise au point des bases de règles des systèmes experts sur micro-ordinateurs, plus rapides qu'à partir d'un Minitel, avant de les connecter sur un réseau télématique.

Nous pouvons maintenant mieux visualiser ce que pourrait être un service télématique utilisant les différentes bases de règles développées plus haut (figure 3-20).

Ce service est dédié à l'assainissement individuel, et s'adresse en priorité aux personnes non qualifiées. Cette restriction à un domaine de l'assainissement en fait peut-être un outil peu ambitieux, mais elle a l'avantage de n'impliquer que des éléments réalistes et réalisables, et dont la faisabilité est démontrée.

Exploité par exemple par une administration compétente (D.D.A.S.S., agence de bassin ...), les connaissances de ce système peuvent être maintenues à jour en temps réel, puisqu'il suffit de modifier la base de règles pour *qu'instantanément*, tout utilisateur en bénéficie.

Selon les besoins et les moyens du service gestionnaire, celui-ci peut réutiliser les sessions à des fins de statistiques, de suivi de dossiers (à condition de prévoir un module de saisie des coordonnées de l'utilisateur), il peut générer automatiquement des formulaires , envoyer un spécialiste sur place ...

5.2 EXTRAPOLATION DE CES TRAVAUX VERS D'AUTRES DOMAINES QUE L'ASSAINISSEMENT

La formalisation de la connaissance telle que nous venons de la développer, ainsi que son association avec une version télématique, peut aider à résoudre un certain nombre de problèmes caractérisés par :

- un choix rendu difficile par l'existence d'un nombre plus ou moins grand de paramètres. Ce choix est complexe s'il doit être fait parmi un grand nombre de possibilités, et s'il est généré par une exploration, pouvant être longue et délicate, des multiples combinaisons entre les valeurs des paramètres;
- une difficulté de diffusion auprès d'un large public, face à des besoins de mise à jour régulière, rapide et aisée.

L'Ecole des Mines de Saint ETIENNE s'est vu confier à plusieurs reprises par l'EXERA (association des exploitants d'équipements de mesure, de régulation et

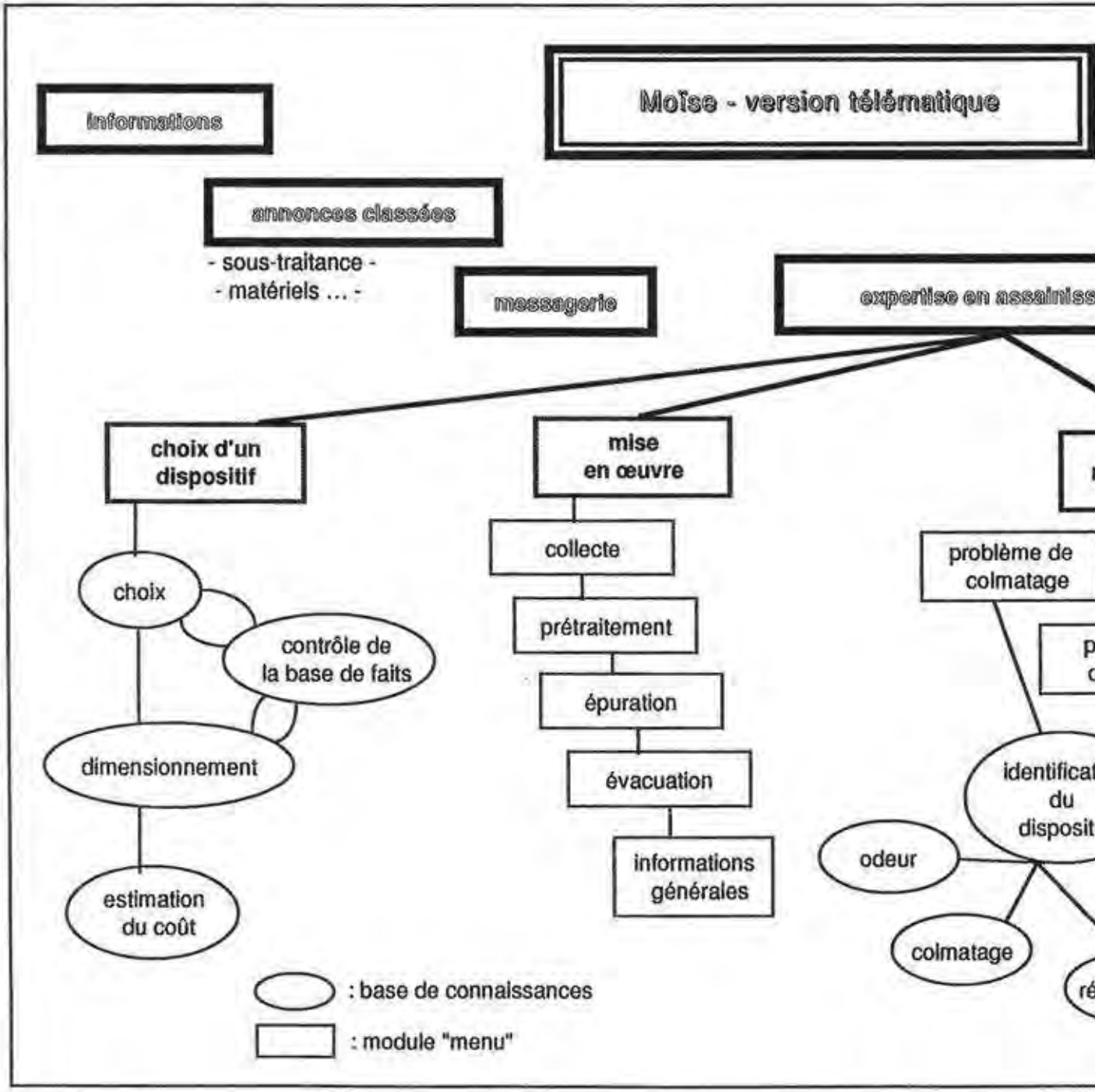


figure 3-20 : chéma général d'une version télématique de Moïse

d'automatisme) des études visant à proposer des "guides de choix" d'instrumentation chimique, pour des analyses d'eau ou d'autres types d'investigations.

Ces guides de choix, publiés sur de petits fascicules, concernant chacun un type d'appareils (par exemple "les analyseurs et détecteurs de CO") ont pour objet :

- d'expliquer les principes des mesures;
- d'exposer les avantages et inconvénients des procédés;
- de décrire les principales caractéristiques des appareils sur le marché et de guider l'utilisateur dans son choix.

C'est bien au niveau du dernier point que nous pouvons apporter notre contribution puisque l'utilisateur est confronté à un problème de choix, et le concepteur du guide est obligé de figer sa description des appareils à la date de parution du fascicule alors que les constructeurs proposent de nouvelles options ou de nouvelles machines régulièrement.

Nous avons donc proposé de tester le moteur de Moïse pour aider au choix, et de l'appliquer au Minitel, pour une diffusion élargie et pour une mise à jour en temps réel.

La méthodologie a été testée sur les "intégrateurs et logiciels d'intégration pour chromatographes". Le rôle d'un intégrateur en chromatographie est d'échantillonner le signal en provenance d'un chromatographe, de stocker les données, et, à l'aide de logiciels, de calculer certaines caractéristiques du chromatogramme (aires des pics, temps de rétention, calcul de compositions ...), avant de les éditer sur un bulletin d'analyse, ou de les visualiser sur un écran vidéo.

Il y a sur le marché français une quarantaine d'appareils en vente, -représentant une quinzaine de constructeurs-, dont la différenciation peut se faire à partir d'une douzaine de paramètres.

Dès les premiers entretiens avec les spécialistes du domaine, il est apparu que Moïse devrait satisfaire aux contraintes suivantes :

- proposer, en mode déduction, tous les appareils compatibles avec le contexte décrit par l'utilisateur. En effet, il ne faut surtout pas favoriser un constructeur par rapport à un autre en ordonnant la liste des appareils. Cette stratégie

d'exploration d'une 'haie' a déjà été vue dans le cadre de la maintenance de l'application précédente.

• accepter que l'utilisateur ne réponde pas à une question, c'est-à-dire que celui-ci déclare qu'il n'attache pas d'importance au paramètre en question. Le problème a été résolu en proposant systématiquement une réponse "peu importe" pour la valeur d'un paramètre, sachant que le moteur doit alors ignorer ce paramètre dans l'exploration de la base de règles.

• accepter également que des paramètres ne soient pas renseignés par les constructeurs. En effet, chaque constructeur préfère parfois avancer certains critères pour décrire ses produits, quitte à en éclipser d'autres. Cet aspect sera mis en évidence en associant à chaque appareil autant d'astérisques qu'il y a de paramètres non renseignés. Ensuite, ces lacunes devront être décrites explicitement dans le commentaire relatif aux caractéristiques de l'appareil.

En outre, l'utilisateur continue de bénéficier de l'aide en ligne sur la nature de chaque paramètre et du mode vérification, qui se révèle ici très pertinent dans la mesure où l'utilisateur peut avoir une idée préconçue de l'appareil qu'il veut acquérir et vérifier s'il est bien adapté à son cas.

Deux jours de travail ont été nécessaires pour réaliser ces adaptations, sous forme d'une maquette, permettant de proposer une démonstration avec deux objectifs :

• visualiser l'intérêt d'une telle démarche et sa faisabilité aux yeux des responsables de l' EXERA .

• développer un cahier des charges, à la lumière de la démonstration, pour une éventuelle application opérationnelle, notamment en fonction des contraintes informatiques de l'EXERA.

L'annexe A6 reprend le rapport fournit aux responsables de l'EXERA.

En outre, en cas de mise en service, le développement d'un éditeur de règles s'avère utile afin de pouvoir facilement mettre à jour la base de règles qui contient l'ensemble des appareils et leurs caractéristiques.

Conclusion

" un système expert est un programme ordinaire
écrit dans un style particulier "

J. FELDMAN (d'après [LEVINE, POMEROL 1989])

Nous avions commencé ce mémoire par une provocation; le terminerions-nous par une autre?

En fait, il serait assurément plus juste de remplacer les termes "système expert" par "*système à base de connaissances*" (le KBS : "Knowledge based system" des anglophones) sachant qu'une des possibilités de ces derniers est de pouvoir mimer le raisonnement d'un expert.

C'est au travers de deux applications que nous avons essayé de montrer le développement d'un système à base de connaissances et tout l'intérêt pour l'hydrologie :

• Promise est un simulateur de projet développé ici dans le cadre d'une action pédagogique qu'il ne faut à notre avis surtout pas négliger en sciences de l'eau, pluridisciplinaire par essence. L'hydrologue y évolue dans un espace multicritère et la simulation y est probablement la meilleure méthode de formation. Nous avons essayé de rendre ce simulateur "intelligent" en introduisant une gestion des liens entre les différentes étapes du projet, permettant la mise en place des fonctions de contrôle et "d'extraction de connaissances". Les évènements générés sont dépendants du passé du projet et c'est le concepteur de la base de connaissances, à qui il est fourni un formalisme de règles de simulation, qui va créer les scénarios d'évolution du projet en introduisant des heuristiques : c'est par tous ces aspects que le simulateur peut être affublé du qualificatif d'intelligent.

• Moïse est un outil de diagnostic pour le choix d'un dispositif d'assainissement individuel. A ce titre, c'est un des outils activable par Promise. Face à un grand nombre de valeurs de paramètres, conduisant à des combinaisons multiples, nous

montrons qu'un formalisme de règles adapté permet de simplifier le problème et fournit une représentation, sous forme de tableaux, très accessible. Cette méthode peut être reprise dans divers aspects de l'assainissement individuel comme le dimensionnement des dispositifs, leur maintenance, ou le contrôle de cohérence des assertions qui sont communiquées par l'utilisateur.

Pour l'aspect opérationnel, nous avons développé une interface télématique de l'outil, autorisant une communication rapide, interactive, soit avec des personnes non qualifiées en assainissement individuel, pour leur fournir une aide, et éventuellement pour provoquer une prise de conscience des contraintes engendrées par cette technique, soit avec des spécialistes, de qui on peut attendre un ajustement et/ou une maintenance des règles utilisées. Ce dernier point fait l'objet de travaux menés à l'Ecole des Mines de Saint Etienne, qui visent notamment à coupler l'utilisation des techniques d'apprentissage et de la télématique [BEAUNE, GRAILLOT, CRES 1989]. Ici, "la boucle est bouclée" puisque le Minitel sert à la fois à diffuser la connaissance, sous forme de conseils envers des non initiés, et à la concentrer en permettant la concaténation de l'expérience de plusieurs spécialistes.

Une extrapolation de ces techniques (système à base de connaissances et télématique) est proposée et testée dans un domaine voisin (choix d'intrumentation chimique pour des analyses d'eau par exemple) où un besoin a été émis par des industriels.

Ces deux applications sont écrites en Prolog, choix judicieux puisqu'un tel langage contient intrinsèquement les notions de règles, de faits, et les "instructions" de gestion de la base de connaissances permettant de séparer la "machinerie informatique" des connaissances proprement dites. De plus, ce langage, dit de quatrième génération, offre le traitement des listes, des symboles permettant l'introduction et la gestion des variables qualitatives, et une programmation de type déclarative. Fiabilité et confort de développement des logiciels sont grandement accrus, notamment pour des applications non structurées ou non normalisées ("not programmed" pour reprendre la terminologie anglaise), c'est-à-dire dont on ne possède pas d'algorithme de résolution.

De l'association de la technique des systèmes à base de connaissances et de Prolog, il découle trois conséquences importantes sur le plan pratique pour les applications :

• *la modularité,* qui s'établit à plusieurs niveaux. Dans Promise, une base de connaissances peut être créée pour différents type de projets du domaine de l'eau (AEP, assainissement, irrigation ...), chacune d'entre elles pouvant être scindée en une phase pré-étude, étude, travaux ... Dans Moïse, des modules différents concernent plusieurs aspects de l'assainissement individuel. Chaque règle, décrivant un contexte, est indépendante des autres.

• *la maintenance,* qui est une conséquence de la modularité notamment grâce à l'indépendance que peut acquérir chaque module et chaque règle.

• *la lisibilité,* qui couronne les deux aspects précédents en ouvrant la voie de la formalisation de problèmes complexes à des non-informaticiens, mais spécialistes d'un autre domaine (systèmes experts).

Enfin, pour conclure, il apparaît que le développement des applications tel que nous venons de l'exposer est une étape profitable et indispensable pour concevoir des logiciels plus généraux qui intègrent une panoplie d'outils, à l'image des S.I.A.D. (Système Interactif d'Aide à la Décision). Dans cette optique, Promise est un exemple de système interactif, et Moïse un apport à l'aide à la décision.

Bibliographie

BIBLIOGRAPHIE

BEAUNE P., GRAILLOT D., CRES F.-N. (1989)

Moïse : un système d'apprentissage accessible par Minitel.
Note interne. Ecole des Mines de Saint-Etienne. Janvier 1989.

BLACHERE A., GRAILLOT D. (1987)

Utilisation d'un système expert pour le choix d'un dispositif d'assainissement.
XXII^{ème} congrès AIRH. Lausanne.

BOUVERESSE J. (1985)

Les machines sont-elles intelligentes?
La Recherche n° 170. Octobre 1985. P 1126-1127.

CARBONNEL J., MICHALSKI R., MITCHELL J. (1986)

Machine learning. An artificial intelligence approach.
Vol II. MORGAN KAUFMAN.

CARSALADE J.P. (1988)

Systèmes experts : l'état des applications.
1ère journée 3G - systèmes experts et réseaux appliqués.
15 juin 1988. Université Lumière LYON 2. LYON.

CHAPMAN K.P., MANESERO A. (1988)

An intelligent knowledge-based system for construction project management.
Les systèmes experts et leurs applications. Huitièmes journées.
AVIGNON. 30 mai-3 juin 1988. p505-514.

COSTE C., LOUDET M. (1980)

Guide de l'assainissement en milieu rural et urbain.
Editions du MONITEUR. 415 p.

COUSIN F., BOURBIGOT M.-M. (1989)

Détection de la toxicité des rejets industriels à l'aide de systèmes experts.
1er symposium sur l'intelligence artificielle appliquée à l'environnement.
1er février 1989. La COURLY. LYON.

C.P.G.F. (1988) [Compagnie de Prospection Géophysique Française]
Assainissement de la vallée de la VAREZE.
Rapport d'étude n° 3192.

CRES F.-N. & al. (1988)
Development of a computerized tool for communication between users, experts and decision makers in a water project framework.
International workshop on water awareness in societal planing and decision making STOCKHOLM. 27 juin-1er juillet 1988. p141-144.

CRES F.-N., GRAILLOT D., BEAUNE P. (1988)
Intelligence artificielle et télématique pour l'adaptation des techniques d'assainissement en milieu rural.
Communication au Salon Hydroplan 88. MARSEILLE. p180-185.

CURIEN P., MENEGAUX J.-M. (1986)
Télécommunications sur IBM PC.
EDI-TESTS. Collection micro-informatique. 150p.

DAVOINE Ph., GRAILLOT D. (1984)
MISE : Modèle intégré en stratégie de l'eau.
Industrie Minérale, les techniques. Décembre 1984. p 816-822.

DAVOINE Ph., GRAILLOT D. (1986)
Enseignement assisté par ordinateur : MISE.
Communication aux journées internationales de l'eau organisées par l'ISTED. 9 au 11 juin 1986. MARSEILLE.

DAVOINE Ph.(1988)
Simulation de projets d'alimentation en eau : MISE. Présentation générale.
Note interne. Ecole des Mines de Saint-Etienne.

DELAHAYE J.P. (1989)
Une extension spectaculaire du théorème de Gödel : l'équation de Chaitin.
AFCET/INTERFACES n° 75. Janvier 1989. p 12-16.

DENOEUX T. (1988)
Systèmes experts et génie urbain: quelques enseignements tirés des premières réalisations en France. Rapport d'étude. CERGRENE. 72p.

DETAY M., POYET P. (1988)

Hydroexpert. Un système expert en hydrogéologie de terrain.
Salon Hydroplan. 17 - 20 mai 1988. MARSEILLE.

DREYFUS H.L. (1984)

Intelligence artificielle. Mythes et limites.
Editions FLAMMARION. 443p.

FARRENY H. (1985)

Les systèmes experts. Principes et exemples.
Editions CEPADUES. 254p.

FARRENY H., GHALLAB M. (1988)

Eléments d'intelligence artificielle.
Editions HERMES. Traité des nouvelles technologies. Série intelligence artificielle.
367p.

GARANCHER J. (1986)

L'assainissement autonome individuel et collectif.
Editions du Moniteur. 158 p.

GONDRAN M., LALEUF J.-C. (1988)

Intelligence et systèmes experts.
AFCET/INTERFACES. n°64. Février 1988. p15-21.

GRAILLOT D. (1983)

MISE : Modèle intégré en stratégie de l'eau, outil pédagogique et d'aide à la décision.
Thèse 3ème cycle.
Université des Sciences et Techniques du Languedoc. MONTPELLIER.

GRAILLOT D. (1986)

Faisabilité d'un système d'ingénierie pour la réalisation de projets d'aménagement en eau à partir du modèle de simulation MISE.
Thèse d'état.
Université des Sciences et Techniques du Languedoc. MONTPELLIER.

HAGERMAN D.A., RHUDY R.L., WIGINTON J.C. (1988)

Evolution of a knowledge-based system for project consulting.
Les systèmes experts et leurs applications. Huitièmes journées.
AVIGNON. 30 mai-3 juin 1988. p485-504.

HEBENSTREIT J. (1988)

Simulation et pédagogie : une rencontre du troisième type.
AFCET/INTERFACES n° 65. Mars 1988. p 14-18.

HOFSTADTER D. (1985)

GODEL, ESCHER, BACH : les brins d'une guirlande éternelle.
INTEREDITIONS. 883p.

JOURNAL OFFICIEL

J.O. du 9 avril 1982. Arrêté du 3 mars 1982. Titre premier, article 3.

J.O. du 21 septembre 1984. Affaires sociales. Circulaire du 20 août 1984.
Assainissement autonome des bâtiments d'habitation.
Texte officiel n° 84-40.

LAPORTE J., DELPORT D. (1987)

Turbo-Prolog, construisez des applications.
Editions EYROLLES. 247p.

LARDERA S. (1989)

Un modèle dynamique pour le pilotage de projets.
AFCET/INTERFACES n° 79. Mai 1989. p 9-15.

LA RECHERCHE (1988)

Les nouveaux ordinateurs.
Numéro spécial 204. Novembre 1988.

LAURIERE J.-L. (1987)

Intelligence artificielle; résolution de problèmes par l'homme et la machine.
Editions EYROLLES. 473p.

LEVINE P., POMEROL J.C. (1989)

Systèmes interactifs d'aide à la décision et systèmes experts.
Edition HERMES. Traité des nouvelles technologies.
Série décision assistée par ordinateur. 335 p.

MAZAS P. (1986)

Dans quelles circonstances l'expert peut-il modifier lui-même sa base de connaissances ?.
CIIAM 86. p141-152.

MERING C., BLAMONT D., GANASCIA J.-G., MONJANEL F. (1988)

CIME : une application des systèmes experts à la télédétection.
Les systèmes experts et leurs applications. Huitièmes journées.
AVIGNON. 30 mai-3 juin 1988. p427-448.

MINISTERE DE L'ENVIRONNEMENT (1981)

Cahiers techniques de la direction de la prévention des pollutions.
Assainissement individuel n° 5. 71 p.

MINISTERE DE L'INTERIEUR ET DE LA DECENTRALISATION (1982)

Assainissement en milieu rural. Supplément au n° 22 de "Démocratie locale".
Direction Générale des Collectivités Locales. 48 p.

MINISTERE DE L'URBANISME, DU LOGEMENT ET DES TRANSPORTS (1985)

Assainissement autonome : éléments pour un bilan technico-économique.
Service Technique de l'Urbanisme. 109p.

MIRET P. (1987)

SIAD et systèmes experts : outils d'apprentissage des individus et facteurs d'évolution de l'organisation.
AFCET/INTERFACES. n°57. Juillet 1987. p10-13.

PEREZ J.-Cl. (1988)

De nouvelles voies vers l'intelligence artificielle, pluri-disciplinarité,auto-organisation, réseaux neuronaux.
Editions MASSON. 247p.

PUJOLLE G. (1985)

La télématique. Réseaux et applications
Editions EYROLLES. 157p.

RICARD B., MONNIER B., MOREL J. (1986)

Système expert d'aide au diagnostic des défauts d'un groupe turbo-alternateur.
CIIAM 86. MARSEILLE. 1 - 5 décembre 1986. p 233-259.

RETOUR D. (1984)

Les systèmes experts aux Etats-Unis.
Université des Sciences Sociales de GRENOBLE.
Institut d'Etudes Commerciales de GRENOBLE.

RICH E. (1987)

Intelligence artificielle.
Editions MASSON. 440p.

ROUMAGNAC A. (1987)

Assistance à la conception de l'assainissement autonome collectif par l'intelligence artificielle: ACACIA.
Diplôme d'études approfondies. Université des Sciences et Techniques du Languedoc. MONTPELLIER. 68 p.

ROUHART J. (1986)

L'épuration des eaux usées domestiques.
Edition CEBEDOC. 54 p.

ROUSSET M.-C. (1986)

Sur la validité et la cohérence des bases de connaissances.
CIIAM 86. p723-739.

ROY B., BOUYSSOU D. (1988)

Aide à la décision.
AFCET/INTERFACES n° 65. Mars 1988. p 4-13.

SIBONY J., BEUTLER E., LEGRAND P. (1989)

Rôle de systèmes experts dans l'optimisation du fonctionnement de la station d'épuration de MEYZIEU.
1er symposium sur l'intelligence artificielle appliquée à l'environnement. 1er février 1989. La COURLY. LYON.

SIMON H.A. (1981)

The sciences of the artificial.
MIT press. Cambridge.

SMITH K. A.. (1987)

Building small expert system.

One day workshop. University of Minnesota. Duluth. 18p.

S.N.P.E.A.I. (1983) [chambre syndicale nationale des entreprises et industries de l'hygiène publique]

Assainissement autonome individuel et privé.

Imprimerie Cadet. BLERE (37). 28 p.

STARFIELD A.M., BUTALA K.M., ENGLAND M.M., SMITH K.A. (1983)

Mastering engineering concepts by building an expert system.

Engineering education. November 1983. p104-107.

TREILLET M. (1987)

Le livre de turbo-Prolog.

Editions P.S.I. 185p.

VERONIS J., WURBEL N. (1989)

Une interface en langage naturel pour un système expert d'enseignement de la géométrie..

Les systèmes experts et leurs applications. Neuvièmes journées.

AVIGNON. 29 mai-2 juin 1988. p117-131.

VOIGNIER P. (1989)

Notice d'utilisation du logiciel EAUSER.

Rapport n°8. Saunier Eau et Environnement/Ecole des Mines de Saint-Etienne.

WATERMAN D.A. (1986)

A guide to expert systems.

Addison-Wesley; Reading.

ZHENG-YANG LIU (1988)

Diagnosing and modeling in intelligent teaching systems.

Les systèmes experts et leurs applications. Huitièmes journées.

AVIGNON. 30 mai-3 juin 1988. p87-106.

Annexe 1

introduction à

Turbo-Prolog

TURBO-PROLOG™

(Borland International)

Prolog est un langage né à l'université de MARSEILLE au début des années 1970 et dont le créateur est A. COLMERAUER. L'évolution de cette première version, notamment à l'université d'EDIMBOURG sous l'impulsion de R. KOWALSKI , a donné à Prolog une syntaxe standard (celle de CLOCKSIN / MELLISH) , reprise globalement par Turbo-Prolog.

OBJETS ET RELATIONS

Turbo-prolog considère qu'il a toujours affaire à des objets :

- Des nombres,
- Des constantes , si le premier caractère est une minuscule,
- Des variables , si le premier caractère est une majuscule,

appelés arguments , et qu'il existe entre ces objets des relations, appelées prédicats. Le nombre d'arguments dépendant d'un prédicat s'appelle l'arité.

Par exemple, entre les objets (Ville, Numéro, Heure) , on pourra introduire la relation (train) , reliant deux villes , avec pour arguments le numéro de cette liaison, l'heure de départ, et l'heure d'arrivée .
Ceci peut s'écrire :

```
train(lyon, quimper, 6848, 22h27, 9h56) .
```

Il convient de remarquer que cette notation peut prendre de multiples formes selon le problème que l'on traite, et selon la vision du concepteur du programme.

Ainsi,on pourrait imaginer une relation prenant en compte le moyen de déplacement:

```
moyen_dep(train, lyon, quimper, 6848, 22h27, 9h56)
```

ou bien une relation "train au départ de lyon" :

train_lyon(quimper, 6848, 22h27, 9h56)

ou bien encore une relation "train au départ de lyon avant 23h", l'heure d'arrivée étant ignorée :

train_lyon_av_23h(quimper, 6848) .

Ici, le symbole " _ " sert à favoriser la lecture d'un nom d'objet et remplace "l'espace" qui est considéré comme un séparateur. On verra que le symbole " _ " a une toute autre signification quand *il remplace un nom d'argument.*

FAITS ET REGLES

Toutes les instructions en Turbo-prolog se limitent à des faits et à des règles, qui s'écrivent selon les clauses de HORN :

Conclusion si Conditions(s)

Un fait (ou assertion) est toujours vrai. C'est une clause de HORN sans condition. C'est le cas des exemples précédents, qui s'écriront :

(1) train(lyon, quimper, 6848, 22h27, 9h56).
train(lyon, nantes, 6804, 9h14, 15h54).
train(nantes, quimper, 87773, 16h02, 19h49).

où le point a pour objet de terminer la clause.

Une règle est une clause complète, assortie d'une ou de plusieurs conditions réunies par les opérateurs logiques "et" et "ou". Syntaxiquement, la virgule est synonyme de "et",le point-virgule de "ou", et le symbole ":-" de "si". L'application d'une règle a pour effet un résultat : "vrai" ou "faux".
Par exemple, si l'on veut, au départ de Lyon, arriver à Quimper avant une heure fixée, le train à prendre sera déterminé par les conditions :

1 / Il existe un train entre Lyon et Quimper

2 / Ce train arrive avant l'heure fixée

ce qui donnera la règle :

```
(2)     arrivée_avant(Heure_fixée, H_départ, H_arrivée) :-
                    train(lyon, quimper, _, H_départ, H_arrivée),
                    H_arrivée ≤ H_fixée.
```

Le symbole "_" permet à Turbo-prolog d'ignorer l'argument ainsi désigné.
Les variables sont, soit liées (ou instanciées par le processus d'unification), c'est-à-dire identifiées à une constante , soit libres (ou non instanciées).

Le principe de programmation consiste à assigner un but à Turbo-prolog, qui nous dira s'il l'a atteint (vrai) ou non (faux) avec les règles et faits dont il dispose.

Si on fixe le but suivant :

```
arrivée_avant(10h, X, Y).
```

à Turbo-prolog , avec les faits (1) et la règle (2) , il répondra :

X = 22h27 , Y = 9h56

par contre, dans les mêmes conditions , le but :

```
arrivée_avant(8h, X, Y).
```

échouera.

Une autre façon d'arriver à Quimper sera d'accepter une correspondance (et une seule dans cet exemple). On pourra ainsi choisir un train par la règle suivante :

1/ Il existe un train t1 entre Lyon et une gare X

2 / Il existe un train t2 entre la gare X et Quimper

3 / Le train t1 arrive en X avant que t2 n'en parte

4 / Le train t2 arrive à Quimper avant l'heure fixée

Ce qui se codera :

```
arrivée_avant(H_fixée, H_départ, H_arrivée) :-
        train(lyon, X, _, H_départ, H_arrivée_X),
        train(X, quimper, _, H_départ_X, H_arrivée),
        H_arrivée_X < H_départ_X,
        H_arrivée < H_fixée.
```

Appliqué à la base de faits (1) , le but :

arrivée_avant(20h, X, Y).

donnera la réponse :

X = 9h14 , Y = 19h49 .

LISTES ET RECURSIVITE

La liste est une structure connue dans des langages plus classiques : Nous citerons par exemple le Pascal, avec les chaînes, pour lesquelles il faut malheureusement gérer le début de chaîne, les pointeurs, la fin de chaîne...En Turbo-prolog, cette gestion est assurée par le compilateur, et une liste se représente simplement par des crochets :

soit [lyon, quimper, nantes] une liste de villes .

Le nombre théorique d'éléments d'une liste n'est pas limité. Chaque élément d'une liste peut lui-même être une liste. Une seule fonction permet d'intervenir sur une liste : elle se note "|" (ascii 124) , et isole un ou plusieurs éléments du début de la liste .

Par exemple :

[X | Fin] conduira au résultat suivant : X = lyon ,
Fin = [quimper, nantes].

[X, Y, Z | Fin] conduira au résultat suivant : X = lyon,
Y = quimper,
Z = nantes,
Fin = [] (liste vide).

[X, Y, Z, T] échouera .

L'association des listes et de la récursivité (propriété, pour un prédicat, de se retrouver à la fois sous forme de condition et de conclusion dans une même règle) fournit un grand confort de programmation.

En effet, l'affichage à l'écran de tous les éléments d'une liste se codera :

```
(4)     ecrire_liste([X | Q]) :-     on sépare le premier élément
              ecrire(X),             on écrit le premier élément
              ecrire_liste(Q).       on continue avec la suite de la liste
```

Dans ce cas, le prédicat finira par échouer après avoir écrit les éléments de la liste, faute de pouvoir séparer une liste vide en deux parties.

LA REMONTEE ("backtracking")

Par défaut, le compilateur Turbo-prolog va rechercher toutes les solutions possibles pour atteindre le but qui lui est assigné. C'est le principe de la remontée, qui va procurer au langage sa faculté d'inférence.

Ainsi, en reprenant la base de faits (1) et les règles (2) et (3), le lancement du but:

```
arrivée_avant(20h, X, Y).
```

donnera deux réponses puisque le compilateur aura trouvé deux façons d'atteindre ce but.

En reprenant la règle (4), on va associer récursivité et remontée, de façon à ce que l'écriture d'une liste se termine par un succès et non plus par un échec :

```
ecrire_liste([]).                  toujours vrai
ecrire_liste([X | Q]) :-
              ecrire(X),
              ecrire_liste(Q).
```

Tant que la liste n'est pas vide, le but réussit avec la 2ème partie de la règle. Lorsque la liste est entièrement affichée, le but est satisfait avec la première partie (écrire une liste vide).

LA PROGRAMMATION DECLARATIVE

Pour conserver notre exemple de train, supposons que l'on veuille calculer d'éventuelles réductions sur le prix d'un billet.

En langage classique procédural, ceci se traduirait par l'algorithme :

" Soustraire la date de naissance du voyageur de la date du jour.
Si cette opération donne un résultat inférieur à 18 ans,
alors réduire le prix du billet de 30%,
sinon ... ".

En langage Prolog, on écrira :

" Le prix pour les mineurs est égal au tarif normal diminué de 30%.
Les mineurs sont des personnes agées de moins de 18 ans.
L'age d'une personne se calcule en soustrayant sa date de naissance à la date du jour ... " .

Ainsi, plutôt que de donner des ordres à la machine, le principe, avec un langage tel que Prolog, consiste à énoncer des faits (les horaires de train par exemple) et à définir des règles (comment aller d'une gare à une autre pour arriver avant une heure fixée).

C'est le principe de la programmation déclarative.

LA STRUCTURE TRI-PARTITE D'UN PROGRAMME PROLOG

On distingue trois parties dans un programme Turbo-prolog :

- Le moteur (ou programme)
- La base de faits
- La base de connaissances (qui peut contenir règles et faits).

Le moteur contient les règles du type de celles que nous avons écrites, c'est-à-dire d'une part celles qui permettent de gérer la communication avec l'utilisateur, et d'autre part celles qui formalisent la connaissance d'un certain domaine, comme par exemple la façon de joindre deux villes par un transport en commun avec des contraintes d'horaire.

Certains auteurs décrivent le moteur comme le principe d'inférence permettant l'exploration d'un arbre, c'est ce que nous appelons le compilateur; il s'agit simplement d'une question de définition.

La base de faits recueille les paramètres et résultats intermédiaires d'une session. En effet, chaque règle du moteur est strictement indépendante des autres : On ne peut communiquer le résultat d'une règle à une autre règle. Il n'y a pas de "variable globale'. Le rôle de la base de faits est d'assurer cette fonction de mise en commun et de mémorisation de la connaissance pendant le déroulement des recherches effectuées par le moteur.

La base de connaissances contient les règles et faits concernant le domaine à explorer: horaires, prix des billets, réductions possibles sur les billets d'avion, de train ou de car...

CARACTERISTIQUES DE TURBO-PROLOG

Turbo-prolog possède l'un des compilateurs Prolog pour compatible PC les plus rapides du marché. Ceci est notamment obtenu par un "typage" des variables.

Un programme Turbo-prolog peut être compilé, ce qui rend l'exécution d'autant plus rapide que l'interprétation des faits et règles est déjà traduite en langage directement compréhensible par la machine. Cette compilation peut, de surcroît, être modulaire, ce qui autorise une grande souplesse de mise au point : chaque module est édité, compilé, testé séparément,puis n'est intégré à l'ensemble qu'une fois parfaitement au point.

L'environnement de développement et de mise au point offre un éditeur performant (multi-fenêtrage, menus déroulants...) et un mode trace permettant une pseudo-interprétation du code;

l'environnement d'exécution offre des fonctions :

- de gestion des fichiers
- de gestion des fenêtres
- d'appel à l'éditeur
- d'interfaçage avec le système
- d'appel aux fonctions mathématiques courantes
- de gestion du graphisme
- ...

Nous terminerons ce bref exposé par une citation de M. VAN CANEGHEM, professeur à l'université d'AIX-MARSEILLE II [dans Génie Logiciel et Systèmes Experts ; Mars 89]: " *Le super Prolog devrait exister dans quelques années et avec un peu de chance il fonctionnera sur un ordinateur à architecture parallèle. Que deviendront alors les autres langages de programmation ?* ".

Annexe 2

notice technique de Promise

logiciel Promise

- Simulateur de projet -

1. ENVIRONNEMENT INFORMATIQUE

machine	:	IBM PC et compatibles sous MS-DOS
mémoire	:	120 Ko environ en RAM (.EXE du moteur)
		+ base de règles (taille variable) en RAM
		+ base de faits (taille variable) en RAM
		+ fichiers sur disque (nombre variable selon base de règles)
langage	:	Turbo-Prolog 1.0

2. LES FICHIERS

2.1 POUR LA COMPILATION

il s'agit de la compilation du moteur dont on ne s'étendra pas sur le fonctionnement.
Voici les fichiers intervenants pour fabriquer le .EXE du simulateur :

MISE.pro	(2Ko)	:	module principal
DECLAR.pro	(1Ko)	:	déclarations "domains" et "database"
TOUCHE.pro	(4Ko)	:	gestion des touches du clavier
DEC_PRED.pro	(2Ko)	:	déclarations "predicates"
PRED.pro	(50Ko)	:	prédicats du moteur
NBFENETRE.pro	(15Ko)	:	gestion des fenêtres

2.2 POUR LE FONCTIONNEMENT

2.2.1 FICHIERS .PRO

Ce sont la base de faits (**BDF**.pro) et la base de règles.
Doivent également être présents les fichiers :

- **REMARK**.pro enregistrant les remarques de l'utilisateur
- **HISTOIRE**.pro enregistrant l'historique pour les besoins de l'utilisateur (touche F7:histo du simulateur)

2.2.2 FICHIERS .INF

Ce sont les fichiers "informations" associés à chaque objectif possible. Dans le cas où on ne désire pas fournir d'informations (objectif parfaitement explicite ou autre raison), le fichier "**zz.inf**" contient le texte : "pas d'information sur cet objectif". C'est ce même fichier qui est affiché quand on demande une information sans avoir décrit complètement un chemin de l'arbre décisionnel.

2.2.3 FICHIERS .REA

Ils sont associés à chaque réalisation, que celle-ci soit un affichage, l'appel d'une procédure ou une question.

2.2.4 FICHIERS .QUE

Ils contiennent les questions que l'on veut poser à l'utilisateur.

2.2.5 FICHIERS .EXE, .COM OU .BAT DES PROCEDURES

L'appel d'une procédure se traduit par l'activation d'un .EXE, .COM ou .BAT. Indépendamment du simulateur, chaque procédure peut faire appel à un certain nombre de fichiers.

2.2.6 ACTIVATION DU SIMULATEUR

Le simulateur est activée par l'ordre "**PROMISE**" qui correspond à un fichier ".BAT".

PROMISE.BAT =
```
ECHO OFF
CLS
MISE
ECHO OFF
PROCED
```

MISE est le moteur du simulateur. PROCED est un fichier ".BAT" *engendré par le simulateur* quand il faut enclencher une procédure (voir paragraphe 4.3.2).

Au démarrage du simulateur, la base de règles doit simplement contenir l'initialisation des coûts du projet, soit *"projet (0, 0, 0)"*.

2.2.7 GESTION DES EXTENSIONS DE FICHIERS

C'est le simulateur qui se charge de gérer toutes les extensions de fichiers. En aucun cas, ces extensions ne doivent apparaître dans la base de règles.

3. LA BASE DE FAITS

Cinq prédicats sont gérés par la base de faits (prédicats déclarés en "database") [L'adjonction de nouveaux prédicats en base de faits (pour l'échange d'informations avec des procédures par exemple) doit entraîner une nouvelle compilation du module principal "mise.pro" du simulateur] :

lis	(pour "liste") est le prédicat décrivant l'arbre décisionnel
desc	(pour "description") est le prédicat d'écriture des règles de décision
fait	enregistre les objectifs passés
fenêtre	sert à gérer les fenêtres du simulateur
projet	enregistre les différents coûts du projet

Lors d'une sauvegarde de la base de faits, seuls les prédicats *"fait"* et *"projet"* sont sauvegardés puisque ce sont les seuls à concerner l'état d'avancement d'un projet.

4. LA BASE DE REGLES

4.1 DEUX TYPES DE FICHIERS

Deux fichiers, représentant exactement la même chose (*les mêmes règles*) sont utilisés pour la base de règles :

- un fichier ***de travail*** pour l'utilisateur

- un fichier ***d'exploitation*** pour le simulateur

Le passage du fichier de travail au fichier d'exploitation est pris en charge par le programme "STB.pro" qui assure également les fonctions de contrôle syntaxique et orthographique (voir paragraphe 5).

Le fichier d'exploitation est un fichier contenant des faits (les règles) directement incorporables dans la base de faits, c'est-à-dire sans blancs, avec un seul fait par ligne et une syntaxe contrôlée. Ces contraintes, qui doivent être impérativement respectées, rendent ce fichier particulièrement difficile à traiter directement par l'utilisateur.

C'est pourquoi ce dernier officie sur un fichier de travail où il peut à loisir incorporer des commentaires et aménager la présentation de façon à améliorer ses interventions.

4.2 L'ARBRE DECISIONNEL

Ce sont les prédicats

liste	dans le fichier *de travail*
lis	dans le fichier *d'exploitation*

qui décrivent l'arbre décisionnel. Dans les deux cas, les arguments sont absolument identiques. (voir la syntaxe paragraphe 3.3.1.2 et 4.4.1, 2ème partie du mémoire).

4.3 LA BASE DE REGLES

Ce sont les prédicats

description	dans le fichier *de travail*
desc	dans le fichier *d'exploitation*

qui décrivent les règles de la simulation. Dans les deux cas, les arguments sont absolument identiques. (voir la syntaxe paragraphe 3.3.1.3 et 4.4.2, 2ème partie du mémoire).

une règle générique s'écrit :

	1ère partie
description(	nom du prédicat
[obj_i],	du type [A_1, O_1, $P_{i\,1}$,... , P_{n1}]
nom_1,	nom du fichier DOS d'information
Cd, Ci, Cf,	les 3 coûts prévus
	2ème partie
[fonction ([	fonction=*l_et_influ* par exemple
l_ch([obj_k]), ch(r_k),	si (obj_k) a eu (r_k) comme réalisation
l_ch([obj_m]), ch(r_m),	et / ou selon la fonction
...	...
l_ch([obj_n]), ch(r_n),	...
p(a), p(b)]),	pondération appliquée sur le tirage
...],	nouvelle fonction
	3ème partie
[p(b_1), p(b_2),	entre les bornes b_1 et b_2
p(∂d), p(∂i), p(∂f),	influence sur les coûts
ch(**nom_2**),	nom du fichier DOS de la réalisation
ch(r_j),	qualificatif de la réalisation
...]	nouvelle bornes de l'axe
).	fin de la règle

On va décrire dans les lignes qui suivent les fichiers, associés à une règle, qui doivent être présents dans le répertoire du simulateur.

4.3.1 DANS TOUS LES CAS

Au moins deux fichiers par règle sont indispensables :

- le fichier d'informations ("zz" à la rigueur) avec **.INF** comme extension;
- le fichier de réalisation avec **.REA** comme extension. Dans le cas d'une activation de procédure, le fichier avec l'extension ".rea" peut être une introduction annonçant la procédure qui va être enclenchée (avertissement ou autre).

exemple :

• à Nom_1= "INFOS" dans la base de règles doit correspondre un fichier "INFOS.INF" dans le répertoire courant.

• à Nom_2 = "REALISA1" dans la base de règles doit correspondre un fichier "REALISA1.REA" dans le répertoire courant.

4.3.2 APPEL DE PROCEDURE

Pour activer une procédure, il suffit de faire commencer le nom de la réalisation par la lettre "**p**" suivie du nom de la procédure. A ce nom, doit également correspondre un fichier *.REA* dans le répertoire courant.

exemple :

• Nom_2 = "phydro" provoquera

1) l'affichage du fichier "HYDRO.REA"

2) l'activation de la procédure "HYDRO.EXE", "HYDRO.BAT" ou "HYDRO.COM"

principe :

le simulateur reconnaissant l'appel d'une procédure par la lettre "p" au début du nom du fichier, va engendrer un fichier "PROCED.BAT" ainsi composé :

PROCED.BAT =	ECHO OFF	
	CLS	
	HYDRO	(le nom de la procédure)
	ECHO OFF	
	CLS	
	PROMISE	

A la fin de la procédure "HYDRO", le simulateur reprendra la main.

D'autre part :

-Si il existe un fichier "PROCED.BAT" au début d'une simulation, celui-ci est éliminé.

- Si "HYDRO" est un fichier ".BAT" ou fait appel à d'autres fichiers".BAT", ceux-ci devront être organisés de façon à rendre la main à PROMISE à la fin de leur exécution.

4.3.3 POSER UNE QUESTION

Pour activer une question, il suffit de faire commencer le nom de la réalisation par la lettre "**q**" suivie d'un nom de fichier. A ce nom, doit correspondre dans le répertoire courant un fichier *.REA* qui sera affiché à la suite de la question, et un fichier "**.QUE**" qui contient la question.

Les réponses de l'utilisateur sont enregistrées dans le fichier "**REMARK.PRO**".

exemple :

• Nom_2 = "QCOMMENT" provoquera

1) l'affichage de la question contenue dans le fichier "COMMENT.QUE" et l'ouverture d'une fenêtre pour que l'utilisateur puisse répondre;

2) l'affichage du fichier "COMMENT.REA" après la réponse de l'utilisateur.

remarque : on peut combiner une question suivie d'un appel de procédure : pour cela, il suffit que le nom du fichier commence par les lettres "**qp**".

exemple :

• Nom_2 = "QPCOMB" provoquera

1) l'affichage de la question contenue dans le fichier "COMB.QUE" et l'ouverture d'une fenêtre pour que l'utilisateur puisse répondre;

2) l'affichage du fichier "COMB.REA" après la réponse de l'utilisateur.

3) l'activation de la procédure "COMB.EXE", "COMB.BAT" ou "COMB.COM"

4.3.4 LE CONTROLE DES QUALIFICATIFS DE REALISATION

Un prédicat suplémentaire est écrit dans le fichier de travail servant à contrôler l'orthographe des qualificatifs de réalisation.

Ce prédicat "qualif" a pour argument une liste de mots, chacun étant un qualificatif utilisé plus loin dans une règle.

Ce prédicat n'est pas reproduit dans le fichier d'exploitation qui sera utilisé par le simulateur.

5 . LE PROGRAMME STB

5.1 OBJET

Il est utilisé pour passer du fichier de travail au fichier d'exploitation. Durant cette phase, plusieurs vérifications sont effectuées afin de délivrer au simulateur un fichier d'exploitation "sain".

Sauf en cas d'erreurs de syntaxe, pour lesquelles une interruption en cours de compilation du programme STB se produit, le fichier d'exploitation est toujours créé, même si des erreurs dans la conception de la base de règles sont détectées; ces dernières s'affichant à l'écran, c'est au concepteur d'en tenir compte en modifiant le fichier de travail. Si ces modifications ne sont pas effectuées, le concepteur s'expose à des erreurs d'exécution ou des fonctionnements anormaux du simulateur.

5.2 LISTE DES CONTROLES EFFECTUES

[En ce qui concerne les erreurs de syntaxe, on se repportera au manuel de Turbo-Prolog]

Le contrôle s'effectue en considérant les prédicats "description" les uns après les autres. A chaque prédicat, l'ensemble des contrôles suivant est effectué. Quand une erreur est détectée, l'intitulé de l'objectif décrit est toujours rappelé afin que le correcteur puisse rapidement retrouver l'endroit de l'erreur dans la base de règles.

Voici l'ensemble des contrôles effectués :

a) l'objectif décrit dans le prédicat "description" doit correspondre à un chemin possible de l'arbre décisionnel. Ce contrôle est intéressant notamment en ce qui concerne l'orthographe de l'intitulé d'un objectif. En effet, une orthographe différente (entre celle d'un objectif décrit dans l'arbre décisionnel et celle dans le prédicat "description") entraîne une impossibilité pour le simulateur de faire l'identification entre les chemins possibles de l'arbre décisionnel et la règle correspondante.

b) l'existence du fichier d'information "Nom_1.inf".

c) dans les fonctions l_et_obli, l_et_influ, l_ou_influ, STB vérifie :

• que les objectifs passés correspondent à un chemin possible de l'arbre décisionnel (comme la première vérification).

• que le qualificatif utilisé figure dans la liste du prédicat "qualif". Les négations sont traitées automatiquement, c'est-à-dire qu'il est inutile de faire figurer dans "qualif" le préfixe "*n_*" pour indiquer la non réalisation d'un objectif.

• que les poids sont conformes à ce qu'on peut attendre :

>= 15000 pour une fonction de type "obli"

< 15000 pour une fonction de type "influ".

d) dans le prédicat décrivant l'axe des réalisations, STB vérifie :

• qu'au niveau de chaque réalisation, le fichier correspondant *.REA* existe .

• que si cette réalisation est un appel de procédure, cette procédure *.BAT*, *.COM* ou *.EXE* existe.

• que si cette réalisation est associée à un question, le fichier correspondant *.QUE* existe.

• que le qualificatif utilisé figure dans la liste du prédicat "qualif".

• que l'influence sur les coûts est inférieure au coût prévu. Il est évident dans ce cas qu'il n'y a pas incompatibilité et qu'il s'agit de mettre peut-être en évidence une anomalie.

e) l'existence du fichier *REMARK.PRO* où sont stockées les réponses aux questions et les remarques de l'utilisateur.

f) l'existence du fichier *HISTOIRE.PRO* où est stockée l'histoire du projet.

Annexe 3

notice technique de Moïse

Logiciel Moïse

- Système Expert en assainissement autonome -

1 MODE OPERATOIRE

1.1 DESCRIPTION DES MENUS

Nous distinguons 2 types de menus :

- les menus "*choix*" qui proposent différentes options de fonctionnement;
- les menus "*questions*" qui interrogent l'utilisateur sur
  - la valeur d'un paramètre,
  - le choix d'une solution à vérifier (mode vérification),
  - l'identité d'un paramètre à effacer.

L'accès aux différentes options disponibles sur les menus peut se faire

- soit par la touche fonction indiquée,
- soit par la lettre en surbrillance pour cette option.

1.2 LES MENUS "CHOIX"

Ils sont au nombre de 4;
leur enchaînement est représenté sur la figure 1 :

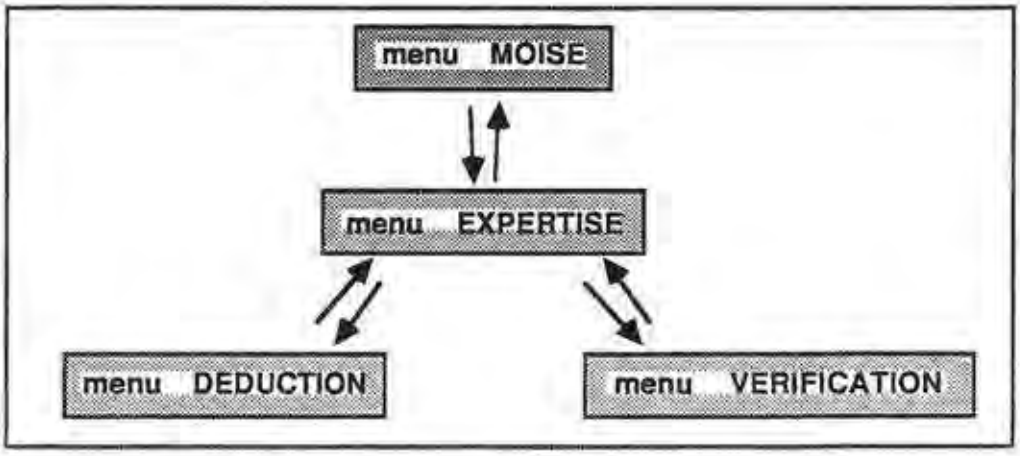


fig. 1 : enchaînement des menus de Moïse

1.2.1 LE MENU "MOISE"

Il gère la maintenance des fichiers sauvegardés dans le répertoire courant. Ces fichiers stockent les données concernant une parcelle.

Outre les touches :

F1 - A - a : aide au menu.
F4 - E - e : activation du menu Expertise.
ESC - Q - q : retour au système d'exploitation.

ce menu propose :

F5 - C - c : charge une base de faits auparavant sauvegardée. La base de faits en cours est détruite. Le chargement d'une base vide ou non lisible entraîne l'affichage du message : "votre fichier de sauvegarde n'est pas correct".

F6 - S - s : sauvegarde une base de faits dans un fichier du répertoire courant. Un nom de fichier est demandé. Si ce nom existe déjà, une confirmation est demandée pour l'effaçage de l'ancien fichier.

F7 - L - l : liste de toutes les sauvegardes effectuées.

F8 - O - o : efface (ôte) un fichier de sauvegarde du répertoire courant.

1.2.2 LE MENU EXPERTISE

Ce menu concerne l'activation du système expert;
Outre les touches :

F1 - A - a : aide au menu.
ESC - Q - q : retour au menu Moïse.

l'activation des 2 modes de fonctionnement s'obtient par :

F4 - D - d : moteur en mode déduction.
F5 - V - v : moteur en mode vérification.

A partir de ce niveau de menu, on peut manipuler la base de faits en mémoire dynamique :

F6 - M - m : affiche la base de faits.
F7 - E - e : efface toute la base de faits.
F8 - G - g : efface (gomme) un seul fait de la base. Le paramètre à effacer figure parmi ceux que Moïse a déjà acquis et dont il propose la liste.

Remarque :

Les menus "déduction" (F4 - D - d) et "vérification" (F5 -V - v) ne sont pas affichés tout de suite. Quand on frappe au clavier l'une des touches correspondantes, on active la recherche ou la vérification d'une solution et Moïse pose des questions ou affiche un résultat.

Cependant, l'avancement dans l'arbre des menus a été effectué selon la figure 2. (vous pouvez le vérifier en tapant " Escape - Q - q " à n'importe quel moment : l'affichage des menus correspondants se fera).

1.2.3 LE MENU DEDUCTION

Le moteur va chercher une solution.
Les touches (F1 - A - a) et (Esc - Q - q) sont connues :

F1 - A - a : aide au menu.
ESC - Q - q : retour au menu Expertise.

De plus, on peut activer :

F5 - R - r : permet la recherche d'une ou plusieurs solutions : tant qu'on reste dans ce menu déduction et qu'on ne modifie pas la base de faits (touches F7 - E - e et F8 - G - g), Moïse

recherche toutes les solutions au problème au fur et à mesure qu'on active " F5 - R - r " (en continuant de poser ou non des questions).

F 6 - M - m : affiche la base de faits.

F 7 - E - e : efface toute la base de faits et active le mode déduction.

F 8 - G - g : efface (gomme) un seul fait de la base et active le mode déduction. Le paramètre a effacer figure parmi ceux que Moïse a déjà acquis et dont il propose la liste.

1.2.4 LE MENU VERIFICATION

Le moteur va chercher à vérifier une solution.
Nous ne revenons pas sur les touches :

F 1 - A - a : aide au menu.

ESC - Q - q : retour au menu Expertise.

Nous retrouvons la même structure qu'au paragraphe précédent :

F 5 - R - r : permet la vérification d'une solution à choisir parmi la liste que propose Moïse.

F 6 - M - m : affiche la base de faits.

F 7 - E - e : efface toute la base de faits et active le mode vérification.

F 8 - G - g : efface (gomme) un seul fait de la base et active le mode vérification. Si une session est en cours (c'est-à-dire que Moïse connaît déjà une solution à vérifier), cette même solution est vérifiée; sinon, Moïse proposera, parmi les solutions possibles, de choisir celle à vérifier.

1.3 LES MENUS "QUESTION"

Ce sont des menus qui permettent au moteur d'interroger l'utilisateur.
Les questions peuvent concerner la valeur d'un paramètre avec 2 types de réponses possibles :

- réponse qualitative,
- réponse quantitative,

ou bien, lorsque l'on veut effacer un paramètre, le choix du paramètre à effacer, ou enfin en mode vérification, le choix de la solution à vérifier. Tous ces menus sont détaillés ci-après.

1.3.1 LE MENU "QUESTION" QUALITATIVE

Le logiciel demande à l'utilisateur de lui fournir une réponse qualitative sur la valeur d'un paramètre. Les réponses possibles figurent dans la fenêtre inférieure. Parfois, lorsque cela est possible, Moïse spécifie les bornes numériques qui précisent et délimitent une classe qualitative.

On retrouve les touches :

F1 - A - a : aide au menu.

ESC - Q - q : retour au menu supérieur (déduction ou vérification).

De plus, on peut activer 2 fonctions :

F4 - P - p : affiche un fichier d'aide précisant la nature du paramètre en question et les moyens de cerner sa valeur.

F5 - S - s : dans le cas où la fenêtre inférieure (où s'affichent les différentes valeurs possibles du paramètre) s'avère trop petite, les touches (F5 - S - s) permettent de la rendre active et de faire défiler l'affichage.

1.3.2 LE MENU "QUESTION" QUANTITATIVE

La réponse à fournir au logiciel est un nombre à écrire dans la fenêtre active. Les 3 fonctions activables ont déjà été rencontrées :

F1 - A - a : aide au menu.

F4 - P - p : affiche un fichier d'aide précisant la nature du paramètre en question et les moyens de cerner sa valeur.

ESC - Q - q : retour au menu supérieur (déduction ou vérification).

1.3.3 LE MENU "EFFACAGE"

Moïse propose, parmi la liste de paramètres qu'il connaît (c'est-à-dire pour lequel vous lui avez fourni une valeur, soit par un menu "question", soit par le chargement d'une base de faits au menu Moïse), de choisir le paramètre qu'il effacera.

3 fonctions sont activables :

F1 - A - a : aide au menu.

F5 - S - s : dans le cas où la fenêtre inférieure (où s'affichent les différentes valeurs possibles du paramètre) s'avère trop petite, les touches (F5 - S - s) permettent de la rendre active et de faire défiler l'affichage.

ESC - Q - q : retour au menu supérieur (déduction ou vérification).

1.3.4 LE MENU "QUESTION" DU CHOIX D'UNE SOLUTION A VERIFIER

En chaînage arrière, Moïse doit savoir quelle solution il doit tenter de vérifier; c'est l'objet de ce menu. Il y a les touches :

F1 - A - a : aide au menu.

F5 - S - s : dans le cas où la fenêtre inférieure (où s'affichent les différentes valeurs possibles du paramètre) s'avère trop petite, les touches (F5 - S - s) permettent de la rendre active et de faire défiler l'affichage.

ESC - Q - q : retour au menu supérieur vérification.

Remarque :

Dans tous ces menus, lorsqu'une réponse incorrecte est fournie, c'est-à-dire :

- choix en dehors des numéros que propose Moïse pour les menus question qualitative, effaçage et choix d'une solution à vérifier.

• entrée d'une chaîne non numérique pour le menu question quantitative, le logiciel refuse la réponse en effaçant la fenêtre où la réponse se fait et en emettant un "bip".

2 ASPECTS TECHNIQUES

2.1 ENVIRONNEMENT INFORMATIQUE

machine	:	IBM PC et compatibles sous MS-DOS.
mémoire	:	200 Ko environ en RAM (moïse.exe), + 1 Ko environ par fichier d'aide (.aid) sur disque, + 1 Ko environ par règles (fichier .fil) sur disque.
langage	:	Turbo Prolog 1.0.

2.2 LES FICHIERS TRAITES

2.2.1 LES FICHIERS *.pro*

Il s'agit des fichiers Prolog nécessaires à la compilation. Mis à part le fichier REGLES.PRO, les autres constituent le moteur du programme et ne seront pas détaillés dans cette notice (on trouvera dans les listages de ces fichiers les commentaires sur le fonctionnement de chaque prédicat).

Voici tous les fichiers qui doivent être présents pour compiler Moïse.exe :

MOISE.pro	(3 Ko)	:	module principal,
MOIDEC.pro	(1 Ko)	:	déclarations "domains" et "database",
TOUCHE.pro	(4 Ko)	:	gestion des touches du clavier,
MOIDECPR.pro	(3.5 Ko)	:	déclarations "predicates",
UTILIT.pro	(23 Ko)	:	prédicats dits "utilitaires",
MENU.pro	(10 Ko)	:	prédicats de gestion des menus,
EXPLOIT.pro	(28 Ko)	:	prédicats d'exploitation des menus,
REGLES.pro	(10 Ko)	:	règles spécifiques à l'assainissement individuel.

2.2.2 LES FICHIERS *.aid*

Il s'agit de tous les fichiers d'aide proposés au cours de l'exécution de Moïse :

• aide à chaque menu (9 actuellement, 1 Ko environ sur disque par fichier)

MOISE.aid	:	menu moïse,
EXPERT.aid	:	menu expertise,
DEDUIRE.aid	:	menu déduire,
VERIFIER.aid	:	menu vérifier,
GOMMER.aid	:	menu effaçage,
L_VERIF.aid	:	menu choix d'une solution à vérifier,
Q_C_ABS.aid	:	menu question quantitative,
Q_C_OU_F.aid	:	menu question qualitative avec bornes numériques,
Q_QUE_F.aid	:	menu question qualitative sans bornes numériques.

• aide pour l'identification des paramètres. Les noms de ces fichiers apparaissent dans la base de règles (voir paragraphe 2.3.3).

2.2.3 LES FICHIERS *.fil*

Ce sont les fichiers directement liés aux règles : à chaque règles est associé un fichier commentaire. Il y en a autant que de règles (voir paragraphe 2.3.4).

2.3 DESCRIPTION DU FICHIER REGLES.PRO

Ce fichier constitue la base de connaissances. Il décrit :

- les solutions possibles,
- les paramètres,
- les noms des fichiers d'aide associés aux paramètres,
- les règles de choix des solutions.

2.3.1 LES SOLUTIONS POSSIBLES

Il s'agit d'une liste de 'string' (chaîne de caractères) qui est l'unique argument du prédicat "***tech_ass***". Ce prédicat s'écrit :

```
tech_ass ( [ "solution 1",
             "solution 2",
             .....
             "solution n"] ).
```

"solution i" donne l'intitulé de la solution n°i.

Le numéro d'ordre des solutions servira ensuite à les identifier (cela évite d'avoir à réécrire systématiquement les intitulés complets des solutions).

Il n'y a pas de limite théorique au nombre de solutions.

exemple :

```
tech_ass ( [ "épandage en tranchées parallèles à la pente",
             "épandage en tranchées perpendiculaires à la pente",
             "filtre à sable"] ).
```

2.3.2 LES PARAMETRES

Leurs descriptions apparaissent dans le prédicat "***param***". Tous les renseignements concernant les paramètres sont dans ce prédicat (à l'exeption du nom du fichier d'aide du paramètre). Il y a autant de déclarations *"param"* que de paramètres.

Les arguments de "param" sont :

param(Tp, Nc, Lcl, Genre, Nom, Verbe, Ulit, Uabr, Lbo), avec

Tp : type du paramètre;

Tp = "chif_ou_four" : les valeurs qualitatives que peut prendre le paramètre sont associées à des bornes numériques. Ces bornes apparaissent à l'écran lorsque Moïse propose de choisir une classe pour le paramètre.

Tp = "que_four" : il n'y a pas de bornes numériques associées aux classes du paramètre (ex: le sous-sol est perméable).

Tp = "chif_absolu" : il s'agit du même type de paramètre que pour Tp = "chif_ou_four". Cependant, Moïse n'affiche pas les bornes numériques, ni les classes possibles lorsqu'il interroge l'utilisateur sur ce paramètre. Il s'agit de ne pas influencer les réponses. L'utilisateur doit alors obligatoirement entrer une valeur numérique pour continuer.

Nc : nombre de classes;

Nc = "multi_cl" : il y a plus de 2 classes pour décrire les valeurs qualitatives possibles d'un paramètre.

Nc = "bi_cl" : il n'y a que 2 classes pour décrire les valeurs qualitatives possibles d'un paramètre.

Lcl : liste de classes décrivant les valeurs possibles

(exemple : [classe1, classe2, ..., classen])

Il n'y a pas de limite au nombre de classes.

Genre : genre du paramètre;

Genre = "e" : genre féminin.

Genre = "" : genre masculin.

Nom : intitulé du paramètre.

ex : "la pente du site".

Verbe : verbe associé au paramètre;

Moïse compose ses phrases par la sémantique suivante : sujet - verbe - complément (ex : la pente du site - est - faible). Le choix du verbe est donc primordial pour une bonne compréhension de l'utilisateur.

Ulit : unité associée au paramètre (en toutes lettres);

ex : Ulit = "pourcentage " pour "la pente du site";

Ulit = "" si il n'y a pas d'unité.

Uabr : unité en mode abrégé;

ex : Uabr = "%" pour "la pente du site";

Uabr = "" si il n'y a pas d'unité.

Lbo : liste des bornes associées au descriptif des classes possibles Lcl;

ainsi, si il y a *n* valeurs possibles du paramètres dans *Lcl*, il y aura *n-1* bornes déclarées dans la liste *Lbo*;

Lbo = [] (liste vide) s'il n'y a pas d'association valeur-borne.

Remarque :

Dans le moteur de cette version 1.0, tous les arguments associés à un paramètre ne sont pas utilisés. Cependant, ils ont été conservés afin de garder la compatibilité de ce fichier de règles avec la version "minitel" de Moïse.

Ainsi, toute amélioration de la base de connaissances est instantanément répercutée sur toutes les versions de Moïse.

Exemple :

param (chif_ou_four, multi_cl, ["très faible", "faible", "moyenne", "forte", "très forte"], "e", "la pente du site", est, pourcentage, %, [2, 8, 15, 25]).

2.3.3 LES FICHIERS D'AIDE ASSOCIES AUX PARAMETRES

Ils sont déclarés dans le prédicat "**param-serveur**" qui, à partir de l'intitulé du paramètre retrouve le nom du fichier d'aide. "*param-serveur*" s'écrit :

param-serveur (Nom, Fichier), avec

Nom : intitulé du paramètre;

il doit être rigoureusement le même que dans le prédicat précédent "*param*".

Fichier : nom du fichier associé;

le moteur y adjoindra automatiquement l'extension ".*aid*".

Exemple :

param-serveur ("la pente du site", "PENTESID").

2.3.4 LES REGLES DE DECISION

Elles sont formées de 2 prédicats essentiels :

- prédicat "***la_solution_est***" qui, s'il réussit, permet à Moïse de conseiller une filière,
- prédicat "***la_non_solution_est_expliquée***" (placé de façon séquentielle après le prédicat précédent "la_solution_est") qui, s'il réussit, permet à Moïse d'expliquer pourquoi il ne conseille pas l'assainissement individuel.

Le premier prédicat s'écrit :

la_solution_est (Num, Fichier), avec

Num : numéro de la solution (voir prédicat "*tech_ass*");

Fichier : nom du fichier associé;
il s'agit du fichier qui commente la solution retenue.
Le moteur ajoute automatiquement l'extension "*.fil*".

exemple :
la_solution_est (1, COM1A).

Le deuxième prédicat s'écrit :

la_non_solution_est_expliquée (Fichier), avec

Fichier : nom d'un fichier;
Dans ce fichier, on trouvera une explication pour laquelle Moïse ne conseille pas l'assainissement individuel. Il y a autant de fichiers de ce type que de combinaisons identifiées d'échec de l'assainissement individuel.
Le moteur ajoute automatiquement l'extension "*.fil*".

exemple :
la_non_solution_est_expliquée (COM70A).

Ces 2 prédicats sont dépendants d'un troisième prédicat "***cond***" qui conclue vrai ou faux selon la valeur d'un paramètre. Ce prédicat s'écrit :

cond (Par, Lvaleur), avec

Par : intitulé de paramètre (voir le prédicat "*param*");

Lvaleur : liste de valeurs;

Ces valeurs doivent figurer dans la description du paramètre en question (argument *Lcl* du prédicat "*param*").

exemple :

cond ("la pente du site", [faible , "très faible"])

s'interprète comme : *si* "la pente du site" est "faible" ***ou*** "très faible"

exemple écriture d'une règle :

```
la_solution_est ( 1, COM1A ) :-
        cond ( P1, [ V1, V2] ),
        cond ( P2, [ V3 ] ),
        ....,
        cond ( Pi, [ Vj, j=k,l ] ).      s'interprète comme :
```

La solution est la *numéro 1* (associée au fichier commentaire COM1A)

si le paramètre P1 vérifie V1 ***ou*** V2 ***et***

le paramètre P2 vérifie V3 ***et***

... ***et***

le paramètre Pi vérifie Vk ***ou*** ... ***ou*** Vl.

On décrit ainsi toutes les combinaisons débouchant sur une solution par une série de prédicats "*la_solution_est*".

Les associations entre les prédicats "*la_non_solution_est_expliquée*" et "*cond*" sont identiques au cas précédent.

2.4 COMMENT INTERVENIR SUR LE FICHIER REGLES.PRO

2.4.1 PREAMBULE

Ce chapitre décrit comment on peut modifier la base de règles. Si des modifications entraînent des interventions sur d'autres fichiers, ces interventions sont signalées dans le texte.

Dans tous les cas, il faudra compiler à nouveau le logiciel afin que les modifications soient intégrées à la nouvelle version. Nous indiquons ci-après le protocole à suivre :

1) charger Turbo-Prolog,
2) faire les modifications partout où cela est nécessaire,
3) charger "moïse.pro",
4) dans le menu Option (O), choisir l'option "Exe file" (E),
5) lancer la compilation (C),
6) quitter Turbo-Prolog (Q).

2.4.2 STRUCTURATION DE LA CONNAISSANCE

La structuration de la connaissance peut être un choix, ou une nécessité.

C'est éventuellement un choix du concepteur de l'expertise qui préfère regrouper une ou des combinaisons de paramètres sous une même notion. Cela peut aussi être une nécessité face aux capacités mémoire d'une machine afin de diminuer la taille du fichier de règles. En effet, le code de ce fichier peut devenir rapidement important puisqu'on répète à chaque prédicat "*cond*" le nom du paramètre et les valeurs qu'il peut prendre. Cependant, la lisibilité des règles s'en trouve optimale.

Comment, en pratique, structurer le fichier de règles ? Tout simplement en regroupant des prédicats "*cond*".

L'exemple qui suit est volontairement simpliste afin d'être bien explicite.

exemple :

```
la_solution_est ( 1, COM1A) :-
        cond ( P1, [ V1, V2] ),
        cond ( P2, [ V3 ] ),
        cond ( Pi, [ Vj, j=k,l ] ).

la_solution_est ( 2, COM2A) :-
        cond ( P3, [ V1, V2] ),
        cond ( P4, [ V3 ] ),
        cond ( Pm, [ Vn, n=o,p ] ).
```

Le groupe `cond ( P1, [ V1, V2] ),`
`cond ( P2, [ V3 ] ),`
apparaît à 2 reprises. On peut donc les regrouper sous un même nom (appelons ce regroupement "*groupe*") qui peut être une notion correspondant à une réalité physique.

Les règles deviennent :

```
la_solution_est ( 1, COM1A) :-
        groupe,
        cond ( Pi, [ Vj, j=k,l ] ).

la_solution_est ( 2, COM2A) :-
        groupe,
        cond ( Pm, [ Vn, n=o,p ] ).

groupe :-
        cond ( P1, [ V1, V2] ),
        cond ( P2, [ V3 ] ).
```

Le prédicat "*groupe*" devra alors être déclaré dans le fichier "MOIDECPR.PRO" qui récapitule pour le compilateur tous les noms de prédicats utilisés.

2.4.3 INTERVENTION SUR LES INTITULES DE SOLUTION

2.4.3.1 MODIFIER UN INTITULE DE SOLUTION

Les solutions étant déclarées dans le prédicat "*tech_ass*" (paragraphe 2.3.1), il suffit de modifier l'intitulé correspondant dans la liste, argument de "*tech_ass*".

On prendra cependant garde au fait que tous les fichiers commentaires (extension *.fil*) associés aux solutions répètent cet intitulé. Il faudra donc aussi intervenir sur ces fichiers.

Une autre recommandation est, dans la mesure du possible, de limiter à 70 caractères la longueur du nouvel intitulé. Ainsi, il pourra s'écrire sur une seule ligne, ce qui est appréciable pour un bon esthétisme des affichages au cours de l'éxécution du logiciel.

2.4.3.2 AJOUTER UN INTITULE DE SOLUTION

Il suffit d'ajouter le nom de cette nouvelle solution dans la liste du prédicat "*tech_ass*". Les recommandations du paragraphe précédent restent valables. De plus, nous rapellons que l'ordre des solutions dans le prédicat "*tech_ass*" est important et en rapport direct avec le numéro indiqué dans le prédicat "*la_solution_est*".

2.4.4 INTERVENTION SUR LA DESCRIPTION D'UN PARAMETRE

2.4.4.1 MODIFIER LA DESCRIPTION D'UN PARAMETRE

Dans tout ce paragraphe, il s'agit d'interventions sur le prédicat "*param*", dont nous rappelons ici la structure :

param(Tp, Nc, Lcl, Genre, Nom, Verbe, Ulit, Uabr, Lbo).

- le type du paramètre Tp

3 types sont actuellement possibles : chif_ou_four, que_four, chif_absolu (voir paragraphe 2.3.2). Le changement du type doit être compatible avec les autres

arguments de "param"; c'est-à-dire par exemple que les types précédents chif_ou_four, chif_absolu sont liés à l'existence de bornes numériques "Lbo" délimitant les valeurs de classes contenues dans "Lcl".

L'introduction d'un nouveau type (autre que les 3 précédents) doit se traduire par l'écriture en Turbo-Prolog d'un nouveau prédicat "question" du fichier "exploit.pro" pour prendre en compte ce nouveau type de question.

• le nombre de classes Nc
ce paramètre n'est pas utilisé dans cette version de Moïse.

• les valeurs de classes Lcl
la modification d'un intitulé de valeur devra se répercuter dans les règles de décisions utilisant la valeur de classe en question (pour le paramètre étudié; voir paragraphe 2.3.4). Le changement du nombre de classes (ajout ou retrait) devra -éventuellement - être combiné avec un changement du nombre de bornes numériques associées aux valeurs. Les règles de décision devront aussi être revues en accord avec la modification.

• le genre Genre
ce paramètre n'est pas utilisé dans cette version de Moïse.

• l'intitulé d'un paramètre Nom
il suffit de changer cet intitulé. Ce changement devra aussi affecter les règles de décision faisant appel à ce paramètre (voir paragraphe 2.3.4, prédicat "cond").

• le verbe Verbe
on prendra simplement garde à ce que l'enchaînement Nom - Verbe - Classe conserve son intelligibilité.

• l'unité litérale Ulit
il suffit de changer cette unité (en accord avec l'unité en abrégé).

• l'unité abrégée Uabr
il suffit de changer cette abréviation (en accord avec l'unité littérale).

• la liste des bornes Lbo

le changement d'une valeur numérique devra rester en accord avec la terminologie des classes possibles d'un paramètre, contenues dans "Lcl". Le changement du nombre de bornes (ajout ou retrait) devra se faire en harmonie avec un changement du nombre de valeurs possibles "Lcl". Les règles de décisions devront aussi être revues en accord avec la modification.

2.4.4.2 AJOUTER UNE DESCRIPTION DE PARAMETRE

Cela se fait simplement en rajoutant une description parmi l'ensemble des prédicats "param". Ce nouveau paramètre interviendra, en toute logique, parmi les règles de décision.

2.4.4.3 ENLEVER UNE DESCRIPTION DE PARAMETRE

Il suffit de retirer de la liste des prédicats "param" la description devenue inutile. Toutes les règles de décisions faisant intervenir ce paramètre devront être corrigées.

2.4.5 INTERVENTION SUR LES REGLES

2.4.5.1 MODIFICATION D'UNE REGLE

Cette intervention se fait soit sur le prédicat "*la_solution_est*", soit sur le prédicat "*la_non_solution_est_expliquée*".

Ces changements devront faire référence à des paramètres existants. Il n'existe pas actuellement de programme de contrôle (éditeurs de règles) pouvant aider aux modifications. Aussi, il est recommandé de bien vérifier les changements que l'on introduit.

2.4.5.2 AJOUTER UNE REGLE

La difficulté issue de l'ajout d'une règle est différente selon le cas :

• s'il s'agit de compléter la façon d'arriver à une solution, c'est-à-dire de rajouter une combinaison de paramètres menant à un solution déjà répertoriée, on prendra bien

garde de classer correctement cette nouvelle règle dans la base. Il faudra également introduire un fichier ".fil" commentant cette nouvelle combinaison.

• s'il s'agit d'introduire une nouvelle solution, pouvant correspondre à un nouveau système d'épandage par exemple, il faudra tout d'abord que ce nouveau dispositif apparaisse dans le prédicat "tech_ass" récapitulant toutes les solutions. Puis, la nouvelle règle devra être classée parmi l'ensemble des règles en n'oubliant pas, comme précédemment, le fichier d'accompagnement ".fic".

2.4.5.2 ENLEVER UNE REGLE

Pour toute intervention de cet ordre, il suffit de retirer de la base de règles, la règle correspondante.

Annexe 4

Exemple de règles de choix d'un assainissement individuel

EPANDAGE EN TRANCHEES PARALELLES à la PENTE								
			1A	1B	1C	1D	1E	1 F
	très forte	> 25%						
	forte	15 - 25%						
pente	moyenne	8 - 15%						
	faible	2 - 8%	X	X	X	X	X	X
	très faible	< 2%	X	X	X	X	X	X
	forte	> 2m	X	X	X	X	X	X
épaiss.	moyenne	1.4 - 2m	X	X	X	X	X	X
sol	faible	0.8 - 1.4m		X		X		X
	très faible	< 0.8m						
	très faible	< 10mm/h						
	faible	10 - 20mm/h						
perméa.	moyenne	20 - 50mm/h						
sol	forte	50 - 500mm/h	X	X	X	X	X	X
	très forte	> 500mm/h	X	X	X	X	X	X
présence	oui				X	X	X	X
nappe	non		X	X				
	très forte	> 2m	////	////	X	X	X	X
	forte	1.5 - 2m	////	////	X	X	X	X
prof.	moyenne	1 - 1.5m	////	////				
nappe	faible	0.5 - 1m	////	////				
	très faible	0.3 - 0.5m	////	////				
	rédhibitoire	< 0.3m	////	////				
rejet	possible		////	////	////	////	////	////
superficiel	impossible		////	////	////	////	////	////
sous-sol	plutôt perméable		////	X	////	X	////	X
	plutôt imperméable		////	////	////	////	////	////
existence	oui		////	////			X	X
d'un captage	non		////	////	X	X		
distance du	satisfaisante	> 35m	////	////	////	////	X	X
captage	trop faible	<35m	////	////	////	////		

EPANDAGE EN TRANCHEES PERPENDICULAIRES à la PENTE		2A	2B	2C	2D	2E	2 F
	très forte > 25%						
	forte 15 - 25%						
pente	moyenne 8 - 15%	X	X	X	X	X	X
	faible 2 - 8%	X	X	X	X	X	X
	très faible < 2%	X	X	X	X	X	X
	forte > 2m	X	X	X	X	X	X
épaiss.	moyenne 1.4 - 2m	X	X	X	X	X	X
sol	faible 0.8 - 1.4m		X		X		X
	très faible < 0.8m						
	très faible < 10mm/h						
	faible 10 - 20mm/h						
perméa.	moyenne 20 - 50mm/h						
sol	forte 50 - 500mm/h	X	X	X	X	X	X
	très forte > 500mm/h	X	X	X	X	X	X
présence	oui			X	X	X	X
nappe	non	X	X				
	très forte > 2m	////	////	X	X	X	X
	forte 1.5 - 2m	////	////	X	X	X	X
prof.	moyenne 1 - 1.5m	////	////				
nappe	faible 0.5 - 1m	////	////				
	très faible 0.3 - 0.5m	////	////				
	rédhibitoire < 0.3m	////	////				
		////	////	////	////	////	////
rejet	possible	////	////	////	////	////	////
superficiel	impossible						
sous-sol	plutôt perméable	////	X	////	X	////	X
	plutôt imperméable	////	////	////	////	////	////
existence	oui	////	////			X	X
d'un captage	non	////	////	X	X		
distance du	satisfaisante > 35m	////	////	////	////	X	X
captage	trop faible <35m	////	////	////	////		

EPANDAGE EN TRANCHEES								
PARALLELES à la PENTE								
DRAINAGE de la NAPPE								
		3A	3B	3C	3D			
	très forte > 25%							
	forte 15 - 25%							
pente	moyenne 8 - 15%							
	faible 2 - 8%	X	X	X	X			
	très faible < 2%	X	X	X	X			
	forte > 2m	X	X	X	X			
épaiss.	moyenne 1.4 - 2m	X	X	X	X			
sol	faible 0.8 - 1.4m			X	X			
	très faible < 0.8m							
	très faible < 10mm/h							
	faible 10 - 20mm/h							
perméa.	moyenne 20 - 50mm/h	X	X	X	X			
sol	forte 50 - 500mm/h	X	X	X	X			
	très forte > 500mm/h							
présence	oui	X	X	X	X			
nappe	non							
	très forte > 2m	X	X	X	X			
	forte 1.5 - 2m	X	X	X	X			
prof.	moyenne 1 - 1.5m	X	X	X	X			
nappe	faible 0.5 - 1m							
	très faible 0.3 - 0.5m							
	rédhibitoire < 0.3m							
rejet	possible	X	X	X	X			
superficiel	impossible							
sous-sol	plutôt perméable	////	////	X	X			
	plutôt imperméable	////	////					
existence	oui		X		X			
d'un captage	non	X		X				
distance du	satisfaisante > 35m	////	X	////	X			
captage	trop faible <35m	////		////				

EPANDAGE EN TRANCHEES								
PERPENDICULAIRES à la PENTE								
DRAINAGE de la NAPPE								
		4A	4B	4C	4D			
	très forte > 25%							
	forte 15 - 25%							
pente	moyenne 8 - 15%	X	X	X	X			
	faible 2 - 8%	X	X	X	X			
	très faible < 2%	X	X	X	X			
	forte > 2m	X	X	X	X			
épaiss.	moyenne 1.4 - 2m	X	X	X	X			
sol	faible 0.8 - 1.4m			X	X			
	très faible < 0.8m							
	très faible < 10mm/h							
	faible 10 - 20mm/h							
perméa.	moyenne 20 - 50mm/h	X	X	X	X			
sol	forte 50 - 500mm/h	X	X	X	X			
	très forte > 500mm/h							
présence	oui	X	X	X	X			
nappe	non							
	très forte > 2m	X	X	X	X			
	forte 1.5 - 2m	X	X	X	X			
prof.	moyenne 1 - 1.5m	X	X	X	X			
nappe	faible 0.5 - 1m							
	très faible 0.3 - 0.5m							
	rédhibitoire < 0.3m							
rejet	possible	X	X	X	X			
superficiel	impossible							
sous-sol	plutôt perméable	////	////	X	X			
	plutôt imperméable	////	////					
existence	oui		X		X			
d'un captage	non	X		X				
distance du	satisfaisante > 35m	////	X	////	X			
captage	trop faible <35m	////		////				

FILTRE à SABLE à FLUX VERTICAL								
			5A	5B	5C	5D	5E	5 F	
	très forte	> 25%							
	forte	15 - 25%	X	X	X	X	X	X	
pente	moyenne	8 - 15%	X	X	X	X	X	X	
	faible	2 - 8%	X	X	X	X	X	X	
	très faible	< 2%	X	X	X	X	X	X	
	forte	> 2m	X	X			X		
épaiss.	moyenne	1.4 - 2m	X	X			X		
sol	faible	0.8 - 1.4m			X	X		X	
	très faible	< 0.8m			X	X		X	
	très faible	< 10mm/h							
	faible	10 - 20mm/h							
perméa.	moyenne	20 - 50mm/h	X	X	X	X	X	X	
sol	forte	50 - 500mm/h	X	X	X	X	X	X	
	très forte	> 500mm/h	X	X	X	X	X	X	
présence	oui			X		X	X	X	
nappe	non		X		X				
	très forte	> 2m	////	X	////	X	X	X	
	forte	1.5 - 2m	////	X	////	X	X	X	
prof.	moyenne	1 - 1.5m	////		////				
nappe	faible	0.5 - 1m	////		////				
	très faible	0.3 - 0.5m	////		////				
	rédhibitoire	< 0.3m	////		////				
rejet	possible		////	////	////	////	////	////	
superficiel	impossible		////	////	////	////	////	////	
sous-sol	plutôt perméable		////	////	X	X	////	X	
	plutôt imperméable		////	////			////		
existence	oui		////		////		X	X	
d'un captage	non		////	X	////	X			
distance du	satisfaisante	> 35m	////	////	////	////	X	X	
captage	trop faible	<35m	////	////	////	////			

FILTRE à SABLE à FLUX VERTICAL								
DRAINAGE des EFFLUENTS vers un								
REJET SUPERFICIEL								
			6A	6B	6C	6D	6E	6 F
	très forte	> 25%						
	forte	15 - 25%	X	X	X	X	X	X
pente	moyenne	8 - 15%	X	X	X	X	X	X
	faible	2 - 8%	X	X	X	X	X	X
	très faible	< 2%	X	X	X	X	X	X
	forte	> 2m	X	X				X
épaiss.	moyenne	1.4 - 2m	X	X				X
sol	faible	0.8 - 1.4m			X	X	X	
	très faible	< 0.8m			X	X	X	
	très faible	< 10mm/h	X	X	X	X	X	X
	faible	10 - 20mm/h	X	X	X	X	X	X
perméa.	moyenne	20 - 50mm/h	X	X	X	X	X	X
sol	forte	50 - 500mm/h	X	X	X	X	X	X
	très forte	> 500mm/h	X	X	X	X	X	X
présence	oui			X		X	X	X
nappe	non		X		X			
	très forte	> 2m	////	X	////	X	X	X
	forte	1.5 - 2m	////	X	////	X	X	X
prof.	moyenne	1 - 1.5m	////		////			
nappe	faible	0.5 - 1m	////		////			
	très faible	0.3 - 0.5m	////		////			
	rédhibitoire	< 0.3m	////		////			
rejet	possible		X	X	X	X	X	X
superficiel	impossible							
sous-sol	plutôt perméable		////	////	////	////	////	////
	plutôt imperméable		////	////	////	////	////	////
existence	oui		////		////		X	X
d'un captage	non		////	X	////	X		
distance du	satisfaisante	> 35m	////	////	////	////	X	X
captage	trop faible	<35m	////	////	////	////		

FILTRE à SABLE à FLUX VERTICAL								
DRAINAGE des EFFLUENTS vers un								
PUITS D'INFILTRATION								
			7A	7B	7C	7D	7E	7 F
	très forte	> 25%						
	forte	15 - 25%	X	X	X	X	X	X
pente	moyenne	8 - 15%	X	X	X	X	X	X
	faible	2 - 8%	X	X	X	X	X	X
	très faible	< 2%	X	X	X	X	X	X
	forte	> 2m	X	X				X
épaiss.	moyenne	1.4 - 2m	X	X				X
sol	faible	0.8 - 1.4m			X	X	X	
	très faible	< 0.8m			X	X	X	
	très faible	< 10mm/h	X	X	X	X	X	X
	faible	10 - 20mm/h	X	X	X	X	X	X
perméa.	moyenne	20 - 50mm/h	X	X	X	X	X	X
sol	forte	50 - 500mm/h	X	X	X	X	X	X
	très forte	> 500mm/h	X	X	X	X	X	X
présence	oui			X		X	X	X
nappe	non		X		X			
	très forte	> 2m	////	X	////	X	X	X
	forte	1.5 - 2m	////	X	////	X	X	X
prof.	moyenne	1 - 1.5m	////		////			
nappe	faible	0.5 - 1m	////		////			
	très faible	0.3 - 0.5m	////		////			
	rédhibitoire	< 0.3m	////		////			
rejet	possible		////	////	////	////	////	////
superficiel	impossible		////	////	////	////	////	////
sous-sol	plutôt perméable		X	X	X	X	X	X
	plutôt imperméable							
existence	oui		////		////		X	X
d'un captage	non		////	X	////	X		
distance du	satisfaisante	> 35m	////	////	////	////	X	X
captage	trop faible	<35m	////	////	////	////		

FILTRE à SABLE à FLUX HORIZONTAL							
DRAINAGE des EFFLUENTS vers un							
REJET SUPERFICIEL							
		8A	8B	8C	8D	8E	8 F
	très forte > 25%						
	forte 15 - 25%	X	X	X	X	X	X
pente	moyenne 8 - 15%	X	X	X	X	X	X
	faible 2 - 8%	X	X	X	X	X	X
	très faible < 2%	X	X	X	X	X	X
	forte > 2m	X	X				X
épaiss.	moyenne 1.4 - 2m	X	X				X
sol	faible 0.8 - 1.4m			X	X	X	
	très faible < 0.8m			X	X	X	
	très faible < 10mm/h	X	X	X	X	X	X
	faible 10 - 20mm/h	X	X	X	X	X	X
perméa.	moyenne 20 - 50mm/h	X	X	X	X	X	X
sol	forte 50 - 500mm/h	X	X	X	X	X	X
	très forte > 500mm/h	X	X	X	X	X	X
présence	oui		X		X	X	X
nappe	non	X		X			
	très forte > 2m	////	X	////	X	X	X
	forte 1.5 - 2m	////	X	////	X	X	X
prof.	moyenne 1 - 1.5m	////	X	////	X	X	X
nappe	faible 0.5 - 1m	////		////			
	très faible 0.3 - 0.5m	////		////			
	rédhibitoire < 0.3m	////		////			
rejet	possible	X	X	X	X	X	X
superficiel	impossible						
sous-sol	plutôt perméable	////	////	////	////	////	////
	plutôt imperméable	////	////	////	////	////	////
existence	oui	////		////		X	X
d'un captage	non	////	X	////	X		
distance du	satisfaisante > 35m	////	////	////	////	X	X
captage	trop faible <35m	////	////	////	////		

FILTRE à SABLE à FLUX HORIZONTAL								
DRAINAGE des EFFLUENTS vers un								
PUITS D'INFILTRATION								
			9A	9B	9C	9D	9E	9 F	
	très forte	> 25%							
	forte	15 - 25%	X	X	X	X	X	X	
pente	moyenne	8 - 15%	X	X	X	X	X	X	
	faible	2 - 8%	X	X	X	X	X	X	
	très faible	< 2%	X	X	X	X	X	X	
	forte	> 2m	X	X				X	
épaiss.	moyenne	1.4 - 2m	X	X				X	
sol	faible	0.8 - 1.4m			X	X	X		
	très faible	< 0.8m			X	X	X		
	très faible	< 10mm/h	X	X	X	X	X	X	
	faible	10 - 20mm/h	X	X	X	X	X	X	
perméa.	moyenne	20 - 50mm/h	X	X	X	X	X	X	
sol	forte	50 - 500mm/h	X	X	X	X	X	X	
	très forte	> 500mm/h	X	X	X	X	X	X	
présence	oui			X		X	X	X	
nappe	non		X		X				
	très forte	> 2m	////	X	////	X	X	X	
	forte	1.5 - 2m	////	X	////	X	X	X	
prof.	moyenne	1 - 1.5m	////	X	////	X	X	X	
nappe	faible	0.5 - 1m	////		////				
	très faible	0.3 - 0.5m	////		////				
	rédhibitoire	< 0.3m	////		////				
rejet	possible		////	////	////	////	////	////	
superficiel	impossible		////	////	////	////	////	////	
sous-sol	plutôt perméable		X	X	X	X	X	X	
	plutôt imperméable								
existence	oui		////		////		X	X	
d'un captage	non		////	X	////	X			
distance du	satisfaisante	> 35m	////	////	////	////	X	X	
captage	trop faible	<35m	////	////	////	////			

TERTRE D'INFILTRATION								
			10A	10B	10C	10D	10E	10 F
	très forte	> 25%						
	forte	15 - 25%	X	X	X	X	X	X
pente	moyenne	8 - 15%	X	X	X	X	X	X
	faible	2 - 8%	X	X	X	X	X	X
	très faible	< 2%	X	X	X	X	X	X
	forte	> 2m	X	X	X	X	X	X
épaiss.	moyenne	1.4 - 2m	X	X	X	X	X	X
sol	faible	0.8 - 1.4m	X	X	X	X	X	X
	très faible	< 0.8m		X		X		X
	très faible	< 10mm/h						
	faible	10 - 20mm/h						
perméa.	moyenne	20 - 50mm/h	X	X	X	X	X	X
sol	forte	50 - 500mm/h	X	X	X	X	X	X
	très forte	> 500mm/h	X	X	X	X	X	X
présence	oui				X	X	X	X
nappe	non		X	X				
	très forte	> 2m	////	////	X	X	X	X
	forte	1.5 - 2m	////	////	X	X	X	X
prof.	moyenne	1 - 1.5m	////	////	X	X	X	X
nappe	faible	0.5 - 1m	////	////	X	X	X	X
	très faible	0.3 - 0.5m	////	////	X	X	X	X
	rédhibitoire	< 0.3m	////	////				
rejet	possible		////	////	////	////	////	////
superficiel	impossible		////	////	////	////	////	////
sous-sol	plutôt perméable		////	X	////	X	////	X
	plutôt imperméable		////		////		////	
existence	oui		////	////			X	X
d'un captage	non		////	////	X	X		
distance du	satisfaisante	> 35m	////	////	////	////	X	X
captage	trop faible	<35m	////	////	////	////		

TERTRE D'INFILTRATION								
DRAINAGE des EFFLUENTS vers un								
REJET SUPERFICIEL								
			11A	11B	11C			
	très forte	> 25%						
	forte	15 - 25%	X	X	X			
pente	moyenne	8 - 15%	X	X	X			
	faible	2 - 8%	X	X	X			
	très faible	< 2%	X	X	X			
	forte	> 2m	X	X	X			
épaiss.	moyenne	1.4 - 2m	X	X	X			
sol	faible	0.8 - 1.4m	X	X	X			
	très faible	< 0.8m	X	X	X			
	très faible	< 10mm/h	X	X	X			
	faible	10 - 20mm/h	X	X	X			
perméa.	moyenne	20 - 50mm/h	X	X	X			
sol	forte	50 - 500mm/h	X	X	X			
	très forte	> 500mm/h	X	X	X			
présence	oui			X	X			
nappe	non		X					
	très forte	> 2m	////	X	X			
	forte	1.5 - 2m	////	X	X			
prof.	moyenne	1 - 1.5m	////	X	X			
nappe	faible	0.5 - 1m	////	X	X			
	très faible	0.3 - 0.5m	////					
	rédhibitoire	< 0.3m	////					
rejet	possible		X	X	X			
superficiel	impossible							
sous-sol	plutôt perméable		////	////	////			
	plutôt imperméable		////	////	////			
existence	oui		////		X			
d'un captage	non		////	X				
distance du	satisfaisante	> 35m	////	////	X			
captage	trop faible	<35m	////	////				

TERTRE D'INFILTRATION								
DRAINAGE des EFFLUENTS vers un								
PUITS D'INFILTRATION								
		12A	12B	12C				
	très forte > 25%							
	forte 15 - 25%	X	X	X				
pente	moyenne 8 - 15%	X	X	X				
	faible 2 - 8%	X	X	X				
	très faible < 2%	X	X	X				
	forte > 2m	X	X	X				
épaiss.	moyenne 1.4 - 2m	X	X	X				
sol	faible 0.8 - 1.4m	X	X	X				
	très faible < 0.8m	X	X	X				
	très faible < 10mm/h	X	X	X				
	faible 10 - 20mm/h	X	X	X				
perméa.	moyenne 20 - 50mm/h	X	X	X				
sol	forte 50 - 500mm/h	X	X	X				
	très forte > 500mm/h	X	X	X				
présence	oui		X	X				
nappe	non	X						
	très forte > 2m	////	X	X				
	forte 1.5 - 2m	////	X	X				
prof.	moyenne 1 - 1.5m	////	X	X				
nappe	faible 0.5 - 1m	////	X	X				
	très faible 0.3 - 0.5m	////						
	rédhibitoire < 0.3m	////						
rejet	possible	////	////	////				
superficiel	impossible	////	////	////				
sous-sol	plutôt perméable	X	X	X				
	plutôt imperméable							
existence	oui	////		X				
d'un captage	non	////	X					
distance du	satisfaisante > 35m	////	////	X				
captage	trop faible <35m	////	////					

FILTRE BACTERIEN PERCOLATEUR								
DRAINAGE des EFFLUENTS vers un								
REJET SUPERFICIEL								
		13A	13B	13C				
	très forte > 25%							
	forte 15 - 25%	X	X	X				
pente	moyenne 8 - 15%	X	X	X				
	faible 2 - 8%	X	X	X				
	très faible < 2%	X	X	X				
	forte > 2m	X	X	X				
épaiss.	moyenne 1.4 - 2m	X	X	X				
sol	faible 0.8 - 1.4m	X	X	X				
	très faible < 0.8m	X	X	X				
	très faible < 10mm/h	X	X	X				
	faible 10 - 20mm/h	X	X	X				
perméa.	moyenne 20 - 50mm/h	X	X	X				
sol	forte 50 - 500mm/h	X	X	X				
	très forte > 500mm/h	X	X	X				
présence	oui		X	X				
nappe	non	X						
	très forte > 2m	////	X	X				
	forte 1.5 - 2m	////	X	X				
prof.	moyenne 1 - 1.5m	////	X	X				
nappe	faible 0.5 - 1m	////	X	X				
	très faible 0.3 - 0.5m	////	X	X				
	rédhibitoire < 0.3m	////						
rejet	possible	X	X	X				
superficiel	impossible							
sous-sol	plutôt perméable	////	////	////				
	plutôt imperméable	////	////	////				
existence	oui	////		X				
d'un captage	non	////	X					
distance du	satisfaisante > 35m	////	////	X				
captage	trop faible <35m	////	////					

FILTRE BACTERIEN PERCOLATEUR								
DRAINAGE des EFFLUENTS vers un								
PUITS D'INFILTRATION								
			14A	14B	14C			
	très forte	> 25%						
	forte	15 - 25%	X	X	X			
pente	moyenne	8 - 15%	X	X	X			
	faible	2 - 8%	X	X	X			
	très faible	< 2%	X	X	X			
	forte	> 2m	X	X	X			
épaiss.	moyenne	1.4 - 2m	X	X	X			
sol	faible	0.8 - 1.4m	X	X	X			
	très faible	< 0.8m	X	X	X			
	très faible	< 10mm/h	X	X	X			
	faible	10 - 20mm/h	X	X	X			
perméa.	moyenne	20 - 50mm/h	X	X	X			
sol	forte	50 - 500mm/h	X	X	X			
	très forte	> 500mm/h	X	X	X			
présence	oui			X	X			
nappe	non		X					
	très forte	> 2m	////	X	X			
	forte	1.5 - 2m	////	X	X			
prof.	moyenne	1 - 1.5m	////	X	X			
nappe	faible	0.5 - 1m	////	X	X			
	très faible	0.3 - 0.5m	////					
	rédhibitoire	< 0.3m	////					
rejet	possible		////	////	////			
superficiel	impossible		////	////	////			
sous-sol	plutôt perméable		X	X	X			
	plutôt imperméable							
existence	oui		////		X			
d'un captage	non		////	X				
distance du	satisfaisante	> 35m	////	////	X			
captage	trop faible	<35m	////	////				

```
/*****************************************************************************/
/*  VERSION PC 1.0                                                           */
/*  -------------                  FICHIER : REGLES.PRO                      */
/*                                                                           */
/*****************************************************************************/

/*****************************************************************************/
/*  REGLES RELATIVES A L'ASSAINISSEMENT INDIVIDUEL                           */
/*                                                                           */
/*****************************************************************************/

clauses

/*****************************************************************************/
/* TECH_ASS(L:liste_string)                                                  */
/*                                                                           */
/* liste de toutes les solutions possibles.                                  */
/* l'ordre est important puisqu'il est ensuite utilisé pour identifier les   */
/* solutions par leur numéro.                                                */
/*****************************************************************************/
        tech_ass([
/*  1*/  "épandage en tranchées parallèles à la pente",
/*  2*/  "épandage en tranchées perpendiculaires à la pente",
/*  3*/  "épandage parallèle à la pente,avec drainage de la nappe",
/*  4*/  "épandage perpendiculaire à la pente, avec drainage de la nappe",
/*  5*/  "filtre à sable, flux vertical",
/*  6*/  "filtre à sable, flux vertical, rejet vers un rejet superficiel",
/*  7*/  "filtre à sable, flux vertical, rejet vers un puits d'infiltration",
/*  8*/  "filtre à sable, flux horizontal, rejet vers un rejet superficiel",
/*  9*/  "filtre à sable, flux horizontal, rejet vers un puits d'infiltration",
/* 10*/  "tertre d'infiltration",
/* 11*/  "tertre d'infiltration, rejet vers un rejet superficiel",
/* 12*/ "tertre d'infiltration, rejet vers un puits d'infiltration",
/* 13*/  "filtre bactérien percolateur, rejet vers un rejet superficiel",
/* 14*/  "filtre bactérien percolateur, rejet vers un puits d'infiltration"
                                                                        ]).
```

```
/*****************************************************************************/
/* PARAM(Tp:string, Tc:string, Lcl:liste_string, G:string, P:string,         */
/*                          V:string, Ul:string, Ua:string, Lbo:liste_real)  */
/*                                                                           */
/* description des paramètres.                                               */
/*****************************************************************************/
        param(chif_ou_four,multi_cl,
        ["très faible","faible","moyenne","forte","très forte"],
        "e","la pente du site",est,"pourcentage","%",[2,8,15,25]).
        param(chif_ou_four,multi_cl,
        ["très faible","faible","moyenne","forte"],
        "e","l'épaisseur du sol",est,"mètres","m",[0.8,1.4,2]).
        param(chif_ou_four,multi_cl,
        ["très faible","faible","moyenne","forte","très forte"],
        "e","la perméabilité du sol",est,"millimètres/heure","mm/h",
        [10,20,50,500]).
        param(chif_ou_four,multi_cl,
        ["rédhibitoire","très faible","faible","moyenne","forte","très
        forte"], "e","la profondeur de la nappe",est,"mètres","m",
        [0.3,0.5,1,1.5,2]).
        param(que_four,bi_cl,
        ["constatée sur le site","pas constatée sur le site"],
        "e","l'existence d'une nappe",est,"","",[]).
        param(que_four,bi_cl,
        ["plutôt perméable","plutôt imperméable"],
        "","le sous-sol",est,"","",[]).
        param(que_four,bi_cl,
        ["possible","impossible"],
        "","le rejet superficiel",est,"","",[]).
        param(que_four,bi_cl,
        ["constatée dans les environs", "pas constatée dans les environs"],
        "e","l'existence d'un captage d'eau",est,"","",[]).
        param(chif_absolu,bi_cl,
        ["trop faible", "satisfaisante"],
        "e","la distance du captage d'eau",est,"mètres","m",[35]).
```

```
/*************************************************************************/
/* MOT_CLE_SERVEUR(P:string, Mc:string)                                  */
/*                                                                       */
/* association entre le nom d'un paramètre et le nom du fichier d'aide qui */
/* lui correspond.                                                       */
/*************************************************************************/
        mot_clé_serveur("la pente du site","PENTESID").
        mot_clé_serveur("l'épaisseur du sol","EPAISSOD").
        mot_clé_serveur("la perméabilité du sol","PERMESOD").
        mot_clé_serveur("la profondeur de la nappe","PROFONAD").
        mot_clé_serveur("l'existence d'une nappe","EXISTNAD").
        mot_clé_serveur("le sous-sol","SUBSTRAD").
        mot_clé_serveur("le rejet superficiel","REJETSUD").
        mot_clé_serveur("l'existence d'un captage d'eau","CAPTEAUD").
        mot_clé_serveur("la distance du captage d'eau","DISTCAPD").


/*************************************************************************/
/* LA_SOLUTION_EST(I:integer, Fic:string)                                */
/*                                                                       */
/* règles définissant les conditions de mise en oeuvre d'une filière.    */
/*************************************************************************/
/*1a */  la_solution_est(1,"COM1A   ") :-
        tra1,
        not(cond("l'existence d'une nappe",["constatée sur le site"])).

/*1b */  la_solution_est(1,"COM1B   ") :-
        tra2,
        not(cond("l'existence d'une nappe",["constatée sur le site"])),
        cond("le sous-sol",["plutôt perméable"]).

/*1c */  la_solution_est(1,"COM1C   ") :-
        tra1,
        nap1,
```

```
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*1d */  la_solution_est(1,"COM1D   ") :-
        tra2,
        nap1,
        cond("le sous-sol",["plutôt perméable"]),
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*1e */  la_solution_est(1,"COM1E   ") :-
        tra1,
        nap1,
        pol.

/*1f */  la_solution_est(1,"COM1F   ") :-
        tra2,
        nap1,
        cond("le sous-sol",["plutôt perméable"]),
        pol.

/*2a */  la_solution_est(2,"COM2A   ") :-
        tra3,
        not(cond("l'existence d'une nappe",["constatée sur le site"])).

/*2b */  la_solution_est(2,"COM2B   ") :-
        tra4,
        not(cond("l'existence d'une nappe",["constatée sur le site"])),
        cond("le sous-sol",["plutôt perméable"]).

/*2c */  la_solution_est(2,"COM2C   ") :-
        tra3,
        nap1,
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).
```

```
/*2d */  la_solution_est(2,"COM2D   ") :-
         tra4,
         nap1,
         cond("le sous-sol",["plutôt perméable"]),
         not(cond("l'existence d'un captage d'eau",["constatée dans les
         environs"])).

/*2e */  la_solution_est(2,"COM2E   ") :-
         tra3,
         nap1,
         pol.

/*2f */  la_solution_est(2,"COM2F   ") :-
         tra4,
         nap1,
         cond("le sous-sol",["plutôt perméable"]),
         pol.

/*3a */  la_solution_est(3,"COM3A   ") :-
         tra1,
         cond("l'existence d'une nappe",["constatée sur le site"]),
         cond("la profondeur de la nappe",["moyenne","forte","très forte"]),
         cond("le rejet superficiel",["possible"]),
         not(cond("l'existence d'un captage d'eau",["constatée dans les
         environs"])).

/*3b */  la_solution_est(3,"COM3B   ") :-
         tra1,
         cond("l'existence d'une nappe",["constatée sur le site"]),
         cond("la profondeur de la nappe",["moyenne","forte","très forte"]),
         cond("le rejet superficiel",["possible"]),
         pol.

/*3c */  la_solution_est(3,"COM3C   ") :-
         tra2,
         cond("l'existence d'une nappe",["constatée sur le site"]),
```

```
        cond("la profondeur de la nappe",["moyenne","forte","très forte"]),
        cond("le rejet superficiel",["possible"]),
        cond("le sous-sol",["plutôt perméable"]),

        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*3d */  la_solution_est(3,"COM3D   ") :-
        tra2,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["moyenne","forte","très forte"]),
        cond("le rejet superficiel",["possible"]),
        cond("le sous-sol",["plutôt perméable"]),
        pol.

/*4a */  la_solution_est(4,"COM4A   ") :-
        tra3,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["moyenne","forte","très forte"]),
        cond("le rejet superficiel",["possible"]),

        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*4b */  la_solution_est(4,"COM4B   ") :-
        tra3,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["moyenne","forte","très forte"]),
        cond("le rejet superficiel",["possible"]),
        pol.

/*4c */  la_solution_est(4,"COM4C   ") :-
        tra4,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["moyenne","forte","très forte"]),
        cond("le rejet superficiel",["possible"]),
```

```
        cond("le sous-sol",["plutôt perméable"]),
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*4d */  la_solution_est(4,"COM4D   ") :-
        tra4,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["moyenne","forte","très forte"]),
        cond("le rejet superficiel",["possible"]),
        cond("le sous-sol",["plutôt perméable"]),
        pol.

/*5a */  la_solution_est(5,"COM5A   ") :-
        fas1,
        cond("la perméabilité du sol",["moyenne","forte","très forte"]),
        not(cond("l'existence d'une nappe",["constatée sur le site"])).

/*5b */  la_solution_est(5,"COM5B   ") :-
        fas1,
        cond("la perméabilité du sol",["moyenne","forte","très forte"]),
        nap1,
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).


/*5c */  la_solution_est(5,"COM5C   ") :-
        fas2,
        cond("la perméabilité du sol",["moyenne","forte","très forte"]),
        not(cond("l'existence d'une nappe",["constatée sur le site"])),
        cond("le sous-sol",["plutôt perméable"]).

/*5d */  la_solution_est(5,"COM5D   ") :-
        fas2,
        cond("la perméabilité du sol",["moyenne","forte","très forte"]),
        nap1,
        cond("le sous-sol",["plutôt perméable"]),
```

```
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*5e */  la_solution_est(5,"COM5E   ") :-
        fas1,
        cond("la perméabilité du sol",["moyenne","forte","très forte"]),
        nap1,
        pol.

/*5f */  la_solution_est(5,"COM5F   ") :-
        fas2,
        cond("la perméabilité du sol",["moyenne","forte","très forte"]),
        nap1,
        cond("le sous-sol",["plutôt perméable"]),
        pol.

/*6a */  la_solution_est(6,"COM6A   ") :-
        fas3,
        not(cond("l'existence d'une nappe",["constatée sur le site"])),
        cond("le rejet superficiel",["possible"]).

/*6b */  la_solution_est(6,"COM6B   ") :-
        fas3,
        nap1,
        cond("le rejet superficiel",["possible"]),
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*6c */  la_solution_est(6,"COM6C   ") :-
        fas4,
        not(cond("l'existence d'une nappe",["constatée sur le site"])),
        cond("le rejet superficiel",["possible"]).

/*6d */  la_solution_est(6,"COM6D   ") :-
        fas4,
        nap1,
```

```
        cond("le rejet superficiel",["possible"]),
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*6e */  la_solution_est(6,"COM6E    ") :-
        fas4,
        napl,
        cond("le rejet superficiel",["possible"]),
        pol.

/*6f */  la_solution_est(6,"COM6F    ") :-
        fas3,
        napl,
        cond("le rejet superficiel",["possible"]),
        pol.

/*7a */  la_solution_est(7,"COM7A    ") :-
        fas3,
        not(cond("l'existence d'une nappe",["constatée sur le site"])),
        cond("le sous-sol",["plutôt perméable"]).

/*7b */  la_solution_est(7,"COM7B    ") :-
        fas3,
        napl,
        cond("le sous-sol",["plutôt perméable"]),
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*7c */  la_solution_est(7,"COM7C    ") :-
        fas4,
        not(cond("l'existence d'une nappe",["constatée sur le site"])),
        cond("le sous-sol",["plutôt perméable"]).

/*7d */  la_solution_est(7,"COM7D    ") :-
        fas4,
        napl,
```

```
        cond("le sous-sol",["plutôt perméable"]),
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*7e */  la_solution_est(7,"COM7E   ") :-
        fas4,
        napl,
        cond("le sous-sol",["plutôt perméable"]),
        pol.

/*7f */  la_solution_est(7,"COM7F   ") :-
        fas3,
        napl,
        cond("le sous-sol",["plutôt perméable"]),
        pol.

/*8a */  la_solution_est(8,"COM8A   ") :-
        fas3,
        not(cond("l'existence d'une nappe",["constatée sur le site"])),
        cond("le rejet superficiel",["possible"]).

/*8b */  la_solution_est(8,"COM8B   ") :-
        fas3,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["moyenne","forte","très forte"]),
        cond("le rejet superficiel",["possible"]),
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*8c */  la_solution_est(8,"COM8C   ") :-
        fas4,
        not(cond("l'existence d'une nappe",["constatée sur le site"])),
        cond("le rejet superficiel",["possible"]).

/*8d */  la_solution_est(8,"COM8D   ") :-
        fas4,
```

```
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["moyenne","forte","très forte"]),
        cond("le rejet superficiel",["possible"]),
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*8e */ la_solution_est(8,"COM8E   ") :-
        fas4,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["moyenne","forte","très forte"]),
        cond("le rejet superficiel",["possible"]),
        pol.

/*8f */ la_solution_est(8,"COM8F   ") :-
        fas3,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["moyenne","forte","très forte"]),
        cond("le rejet superficiel",["possible"]),
        pol.

/*9a */ la_solution_est(9,"COM9A   ") :-
        fas3,
        not(cond("l'existence d'une nappe",["constatée sur le site"])),
        cond("le sous-sol",["plutôt perméable"]).

/*9b */ la_solution_est(9,"COM9B   ") :-
        fas3,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["moyenne","forte","très forte"]),
        cond("le sous-sol",["plutôt perméable"]),
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*9c */ la_solution_est(9,"COM9C   ") :-
        fas4,
        not(cond("l'existence d'une nappe",["constatée sur le site"])),
```

```
        cond("le sous-sol",["plutôt perméable"]).

/*9d */  la_solution_est(9,"COM9D   ") :-
        fas4,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["moyenne","forte","très forte"]),
        cond("le sous-sol",["plutôt perméable"]),
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*9e */  la_solution_est(9,"COM9E   ") :-
        fas4,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["moyenne","forte","très forte"]),
        cond("le sous-sol",["plutôt perméable"]),
        pol.

/*9f */  la_solution_est(9,"COM9F   ") :-
        fas3,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["moyenne","forte","très forte"]),
        cond("le sous-sol",["plutôt perméable"]),
        pol.

/*10a*/  la_solution_est(10,"COM10A  ") :-
        cond("la pente du site",["très faible","faible","moyenne","forte"]),
        cond("l'épaisseur du sol",["faible","moyenne","forte"]),
        cond("la perméabilité du sol",["moyenne","forte","très forte"]),
        not(cond("l'existence d'une nappe",["constatée sur le site"])).

/*10b*/  la_solution_est(10,"COM10B  ") :-
        cond("la pente du site",["très faible","faible","moyenne","forte"]),
        cond("l'épaisseur du sol",["très faible","faible","moyenne","forte"]),
        cond("la perméabilité du sol",["moyenne","forte","très forte"]),
        not(cond("l'existence d'une nappe",["constatée sur le site"])),
        cond("le sous-sol",["plutôt perméable"]).
```

```
/*10c*/  la_solution_est(10,"COM10C  ") :-
         cond("la pente du site",["très faible","faible","moyenne","forte"]),
         cond("l'épaisseur du sol",["faible","moyenne","forte"]),
         cond("la perméabilité du sol",["moyenne","forte","très forte"]),
         cond("l'existence d'une nappe",["constatée sur le site"]),
         cond("la profondeur de la nappe",["très faible","faible","moyenne",
          "forte","très forte"]),
         not(cond("l'existence d'un captage d'eau",["constatée dans les
         environs"])).

/*10d*/  la_solution_est(10,"COM10D  ") :-
         cond("la pente du site",["très faible","faible","moyenne","forte"]),
         cond("l'épaisseur du sol",["très faible","faible","moyenne","forte"]),
         cond("la perméabilité du sol",["moyenne","forte","très forte"]),
         cond("l'existence d'une nappe",["constatée sur le site"]),

         cond("la profondeur de la nappe",["très faible","faible","moyenne",
         "forte","très forte"]),
         cond("le sous-sol",["plutôt perméable"]),
         not(cond("l'existence d'un captage d'eau",["constatée dans les
         environs"])).

/*10e*/  la_solution_est(10,"COM10E  ") :-
         cond("la pente du site",["très faible","faible","moyenne","forte"]),
         cond("l'épaisseur du sol",["faible","moyenne","forte"]),
         cond("la perméabilité du sol",["moyenne","forte","très forte"]),
         cond("l'existence d'une nappe",["constatée sur le site"]),

         cond("la profondeur de la nappe",["très faible","faible","moyenne",
         "forte","très forte"]),
         pol.

/*10f*/  la_solution_est(10,"COM10F  ") :-
         cond("la pente du site",["très faible","faible","moyenne","forte"]),
         cond("l'épaisseur du sol",["très faible","faible","moyenne","forte"]),
```

```
        cond("la perméabilité du sol",["moyenne","forte","très forte"]),
        cond("l'existence d'une nappe",["constatée sur le site"]),

        cond("la profondeur de la nappe",["très faible","faible","moyenne",
        "forte","très forte"]),
        cond("le sous-sol",["plutôt perméable"]),
        pol.

/*11a*/ la_solution_est(11,"COM11A  ") :-
        ter1,
        not(cond("l'existence d'une nappe",["constatée sur le site"])),
        cond("le rejet superficiel",["possible"]).

/*11b*/ la_solution_est(11,"COM11B  ") :-
        ter1,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["faible","moyenne","forte",
        "très forte"]),
        cond("le rejet superficiel",["possible"]),
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*11c*/ la_solution_est(11,"COM11C  ") :-
        ter1,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["faible","moyenne","forte",
        "très forte"]),
        cond("le rejet superficiel",["possible"]),
        pol.

/*12a*/ la_solution_est(12,"COM12A  ") :-
        ter1,
        not(cond("l'existence d'une nappe",["constatée sur le site"])),
        cond("le sous-sol",["plutôt perméable"]).

/*12b*/ la_solution_est(12,"COM12B  ") :-
```

```
        ter1,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["faible","moyenne","forte",
        "très forte"]),
        cond("le sous-sol",["plutôt perméable"]),
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*12c*/ la_solution_est(12,"COM12C  ") :-
        ter1,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["faible","moyenne","forte",
        "très forte"]),
        cond("le sous-sol",["plutôt perméable"]),
        pol.

/*13a*/ la_solution_est(13,"COM13A  ") :-
        ter1,
        not(cond("l'existence d'une nappe",["constatée sur le site"])),
        cond("le rejet superficiel",["possible"]).

/*13b*/ la_solution_est(13,"COM13B  ") :-
        ter1,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["très faible","faible","moyenne",
        "forte","très forte"]),
        cond("le rejet superficiel",["possible"]),
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*13c*/ la_solution_est(13,"COM13C  ") :-
        ter1,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["très faible","faible","moyenne",
        "forte","très forte"]),
        cond("le rejet superficiel",["possible"]),
```

```
        pol.

/*14a*/ la_solution_est(14,"COM14A  ") :-
        ter1,
        not(cond("l'existence d'une nappe",["constatée sur le site"])),
        cond("le sous-sol",["plutôt perméable"]).

/*14b*/ la_solution_est(14,"COM14B  ") :-
        ter1,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["faible","moyenne","forte",
        "très forte"]),
        cond("le sous-sol",["plutôt perméable"]),
        not(cond("l'existence d'un captage d'eau",["constatée dans les
        environs"])).

/*14c*/ la_solution_est(14,"COM14C  ") :-
        ter1,
        cond("l'existence d'une nappe",["constatée sur le site"]),
        cond("la profondeur de la nappe",["faible","moyenne","forte",
        "très forte"]),
        cond("le sous-sol",["plutôt perméable"]),
        pol.

/*70a*/ la_non_solution_est_expliquée ("COM70A  ") :-
        not(cond("la pente du site",["très faible","faible","moyenne",
        "forte"])).

/*70b*/ la_non_solution_est_expliquée ("COM70B  ") :-
        not(cond("l'épaisseur du sol",["très faible","faible","moyenne",
        "forte"])).

/*70c*/ la_non_solution_est_expliquée ("COM70C  ") :-
        not(cond("la perméabilité du sol",["très faible","faible","moyenne",
        "forte","très forte"])).
```

```
/*70d*/  la_non_solution_est_expliquée ("COM70D  ") :-
         oui("l'existence d'une nappe",est,"constatée sur le site"),
         not(cond("la profondeur de la nappe",["très faible","faible",
         "moyenne","forte","très forte"])).

/*70e*/  la_non_solution_est_expliquée ("COM70E  ") :-
         oui("l'existence d'une nappe",est,"constatée sur le site"),
         oui("l'existence d'un captage d'eau",est,"constatée dans les
         environs"),
         non("la distance du captage d'eau",est,"satisfaisante").

/*70f*/  la_non_solution_est_expliquée ("COM70F  ") :-
         per1,
         oui("le sous-sol",est,"plutôt imperméable"),
         non("le rejet superficiel",est,"possible").

/*70g*/  la_non_solution_est_expliquée ("COM70G  ") :-
         per1,
         oui("l'existence d'une nappe",est,"constatée sur le site"),
         oui("la profondeur de la nappe",est,"très faible"),
         non("le sous-sol",est,"plutôt imperméable").

/*70h*/  la_non_solution_est_expliquée ("COM70H  ") :-
         per1,
         oui("l'existence d'une nappe",est,"constatée sur le site"),
         oui("la profondeur de la nappe",est,"très faible").

/*70i*/ la_non_solution_est_expliquée ("COM70I  ") :-
         oui("l'épaisseur du sol",est,"très faible"),
         oui("le sous-sol",est,"plutôt imperméable"),
         non("le rejet superficiel",est,"possible").

         per1 :-
         oui("la perméabilité du sol",est,"très faible");
         oui("la perméabilité du sol",est,"faible").
```

```
tra1 :-
cond("la pente du site",["très faible","faible"]),
cond("l'épaisseur du sol",["moyenne","forte"]),
cond("la perméabilité du sol",["moyenne","forte"]).

tra2 :-
cond("la pente du site",["très faible","faible"]),
cond("l'épaisseur du sol",["faible","moyenne","forte"]),
cond("la perméabilité du sol",["moyenne","forte"]).

tra3 :-
cond("la pente du site",["très faible","faible","moyenne"]),
cond("l'épaisseur du sol",["moyenne","forte"]),
cond("la perméabilité du sol",["moyenne","forte"]).

tra4 :-
cond("la pente du site",["très faible","faible","moyenne"]),
cond("l'épaisseur du sol",["faible","moyenne","forte"]),
cond("la perméabilité du sol",["moyenne","forte"]).

fas1 :-
cond("la pente du site",["très faible","faible","moyenne","forte"]),
cond("l'épaisseur du sol",["moyenne","forte"]).

fas2 :-
cond("la pente du site",["très faible","faible","moyenne","forte"]),
cond("l'épaisseur du sol",["très faible","faible"]).

fas3 :-
fas1,
cond("la perméabilité du sol",["très faible","faible","moyenne",
"forte","très forte"]).

fas4 :-
fas2,
cond("la perméabilité du sol",["très faible","faible","moyenne",
```

```
"forte","très forte"]).

ter1 :-
cond("la pente du site",["très faible","faible","moyenne","forte"]),
cond("l'épaisseur du sol",["très faible","faible","moyenne","forte"]),
cond("la perméabilité du sol",["très faible","faible","moyenne",
"forte","très forte"]).

pol :-
cond("l'existence d'un captage d'eau",["constatée dans les
environs"]),
cond("la distance du captage d'eau",["satisfaisante"]).

nap1 :-
cond("l'existence d'une nappe",["constatée sur le site"]),
cond("la profondeur de la nappe",["forte","très forte"]).
```

Annexe 5

Exemples de connexions télématiques à Moïse

Le tableau de la page suivante présente un échantillon d'organismes dont un ou plusieurs employés se sont connectés au serveur Moïse .

Il va de soi que les motivations de ceux-ci peuvent être très différentes selon le cas . Cependant, il semble que la majorité de ces personnes a d'abord été séduite par l'aspect communication via le minitel. En effet, il s'agit là d'une ouverture relativement nouvelle pour nombre d'organismes et c'est ce qui peut expliquer la grande variété des sociétés qui se sont connectées :

- administrations (STU,DDE,DDASS,CEMAGREF,ministères...)
- banques (crédit lyonnais,caisse d'épargne)
- télécom
- université
- sociétés distributrices d'eau (CGE,Lyonnaise des Eaux)
- organisme de formation (Fondation de l'eau) ...

On remarquera par ailleurs que nous n'avons pas contacté des organismes aussi divers, mais que nous nous sommes contentés d'aviser uniquement les personnes concernées par le problème de l'eau de cette possibilité de connection. Il s'avère que le "bouche à oreilles" a particulièrement bien fonctionné, ce qui fait que même des banques en sont venues à tester Moïse . Cet aspect prouve bien toute la puissance du minitel, issue de la facilité avec laquelle quiconque peut communiquer avec un ordinateur via le réseau téléphonique.

D'autres personnes se sont connectées à la fois par curiosité vis-à-vis de l'aspect minitel mais aussi et surtout pour tester les règles propres à l'assainissement individuel (Agence de bassin, DDASS, DDE, Université de Montpellier, Ministère des Affaires Sociales). De ce point de vue, les réactions ont été moins unanimement positives. Nous nous y attendions dans la mesure où la base de règles testée, certes solide, n'était pas suffisamment fine : Notre but n'était pas de la concevoir ainsi. Aussi, nous nous sommes heurtés aux différentes sensibilités, spécificités et savoir-faire des experts. Un certain nombre d'entre eux a toutefois été satisfait.

Cette expérience nous a cependant confortés dans notre approche visant à ne pas constituer une base de règles "universelle", mais plutôt à offrir à chaque spécialiste une méthode commune de formalisation de la connaissance, suffisamment simple à mettre en oeuvre.

Peut-être la confrontation de ces différentes bases, construites sur le même modèle, permettra-t-elle d'en tirer des enseignements et de progresser vers une homogénéisation des conclusions des différents experts ?

Organisme	nombre de connexions	temps de connexion
SRAE Rhône-Alpes	2	8'
Caisse d'Epargne de Lyon	2	9'
CEMAGREF Lyon	4	52'
département informatique EMSE	1	13'
Crédit Lyonnais	1	1'
DDE 93	3	1h10'
CERU (bureau d'études)	2	38'
Université Montpellier (hydrogéologie)	1	5'
ANVAR	1	40'
DDASS 42	5	1h04'
Fondation de l'Eau (Limoges)	8	2h30'
DOT (Télécom)	2	16'
GFI (société informatique)	1	8'
Ministère affaires sociales	2	1h02'
STU (Service Techn. de l'Urbanisme)	3	1h09'
Université Montpellier (hydrologie)	6	3h24'
DDE 29	1	8'
CGE (Compagnie Générale des Eaux)	3	20'
Lyonnaise des Eaux	5	45'
Medecins Sans Frontières	2	14'
Personnels de l'école des Mines	4	1h09'
DRASS 69	3	13'
ENSHG (Hydraulique Grenoble)	1	30'
Chatin Conseil SCI	1	12'
PRAUD (bureau d'études)	1	1'
Agence de bassin Loire-Bretagne	4	57'
DDAF 42	1	7'
Cité des Sciences et de l'Industrie	1	13'

Exemples de connexions "Minitel" à Moïse.

Annexe 6

Faisabilité télématique

des guides de choix en instrumentation chimique

FAISABILITE TELEMATIQUE DES GUIDES DE CHOIX EN INSTRUMENTATION CHIMIQUE

Philippe BEAUNE / François-Noël CRES
Ecole des Mines de St-Etienne
158 cours Fauriel
42023 Saint Etienne cedex 2
tél. : 77 42 01 36

PROBLEMATIQUE

L'EXERA a confié à plusieurs reprises à l'école des Mines de St-Etienne la confection de guides de choix en matière d'instrumentation chimique. Le résultat de ces études se présente sous forme de rapports, chacun d'entre eux concernant un type d'appareil réalisant une certaine mesure. L'objet de ces rapports est d'une part de faire le point sur les différentes technologies existantes et d'autre part de récapituler les critères à prendre en compte pour différencier les appareils. De plus, une liste aussi exhaustive que possible des appareils disponibles sur le marché français permet à l'utilisateur d'établir une sélection en fonction de ses propres contraintes.

L'intérêt des guides de choix n'est pas discutable, mais deux inconvénients subsistent :

- La **diffusion** sous forme de rapports empêche d'atteindre un large public mais surtout rend laborieuse la mise à jour puisqu'il faut rééditer à chaque fois une série de documents. Cet aspect est très important quand il s'agit de fournir aux adhérents de l'EXERA une information actualisée et tenant compte des avancées techniques des constructeurs.

- La **sélection**, par un utilisateur, d'un ou plusieurs appareils répondant à ses contraintes reste une opération délicate, parfois fastidieuse face à la multiplicité des critères à explorer et au grand nombre d'appareils disponibles. Comment peut-on, dans ces conditions, être certain d'avoir réalisé le bon choix ?

Notre équipe de recherche mène depuis plusieurs années à l'école des Mines de St-Etienne des travaux qui visent à exploiter les possibilités de la télématique. Etymologiquement, ce mot contient la solution aux deux inconvénients évoqués précédemment, puisque la télématique consiste à marier la communication à distance (résolution du problème de la diffusion) et l'informatique (résolution du problème d'aide à la décision). C'est pourquoi nous avons proposé à l'EXERA une démonstration de la faisabilité télématique des guides de choix, afin qu'elle puisse mieux apprécier l'opportunité d'une telle démarche.

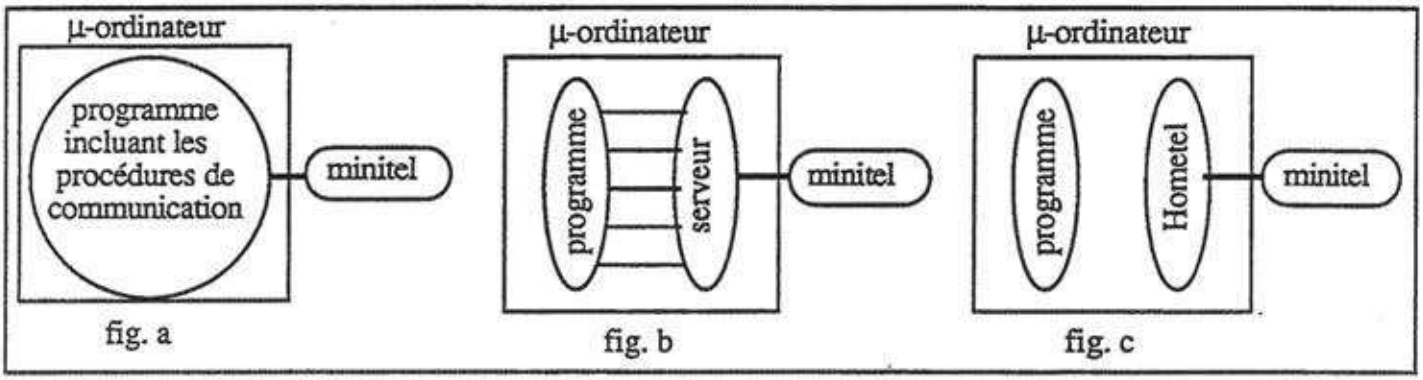
Octobre 1989

LA SOLUTION TELEMATIQUE

Cette solution comporte, comme nous venons de le voir, deux facettes : communication et aide à la décision. Nous décrivons succinctement dans les lignes qui suivent chacun de ces deux aspects, ainsi que le déroulement d'une démonstration-type du logiciel, telle celle qui a été réalisée au Centre de Recherche Elf-Solaise le 11 octobre 1989.

ASPECT COMMUNICATION

Il y a différentes façons de connecter une application informatique sur le réseau téléphonique (fig a,b,c).



La première consiste à écrire le programme d'aide à la décision et les procédures de dialogue avec le minitel dans la même unité informatique (fig. a). Cette solution est très coûteuse en temps car la mise au point de telles procédures est très longue, surtout si l'on veut assurer une bonne gestion télématique, c'est-à-dire une bonne représentation des fonctionnalités d'un serveur (gestion des utilisateurs, des mots de passe, des boîtes aux lettres, de l'accès multivoies,...).

Aussi, une deuxième façon de procéder consiste à acheter un serveur du commerce, qui contient donc déjà toute la gestion des fonctions "serveur", mais qui puisse aussi communiquer avec un programme écrit dans tel ou tel langage (c'est-à-dire que le concepteur du logiciel serveur prévoit et livre des sous-programmes à inclure dans l'application utilisatrice, et permettant l'échange d'informations avec le serveur)(fig.b). Cette technique est à notre avis la meilleure, et nous l'avons mise en œuvre dans une autre application que celle faisant l'objet de ce papier, avec le serveur HOSTEL et un programme écrit en prolog (pour lequel nous avons dû développer spécifiquement les prédicats assurant la communication, ce langage n'étant pas géré par le concepteur de HOSTEL).

Enfin, une troisième solution, extrêmement simple à mettre en œuvre, consiste à acquérir un logiciel du commerce permettant le pilotage à distance d'un micro-ordinateur par minitel (le clavier et l'écran du minitel s'utilisant alors à la place de ceux du micro-ordinateur ainsi équipé) (fig. c). Dans ce cas, il suffit de développer son application dans le langage de son choix de façon classique et d'incorporer le logiciel de communication au moment de l'exploitation. Cette solution est intéressante dans deux cas : lorsque l'on veut faire une démonstration d'un logiciel à distance et s'assurer de son impact médiatique, ou bien lorsque l'on veut limiter l'utilisation de ce logiciel à un échantillon d'utilisateurs pour lequel on n'a alors plus besoin d'assurer les fonctions "serveur". C'est cette troisième technique qui a été mise en œuvre lors de la démonstration du 11 octobre 1989 à Elf-Solaise. A titre indicatif, le logiciel de communication - HOMETEL- coûte environ 2500 F.

ASPECT AIDE A LA DECISION

Côté utilisateur, l'objectif est de favoriser **l'interactivité** d'une session. Côté concepteur, il convient d'optimiser les possibilités de **maintenance.**

Une technologie informatique récemment développée, les systèmes à base de connaissances (systèmes experts), permet d'atteindre ces objectifs. Rappelons brièvement qu'un système à base de connaissances (SBC) se subdivise en trois parties :

- les bases de connaissances :
  - la base de règles, qui contient les connaissances relatives à un domaine, et qui doit être accessible par un spécialiste de ce domaine, donc écrite hors de tout formalisme ou jargon informatique ;
  - la base de faits, qui mémorise les éléments d'une session d'expertise, et qui correspond donc à l'espace utilisateur ;
- le moteur, créé généralement par un informaticien, qui va assurer l'exploration des bases de connaissances et toutes les fonctions annexes du logiciel, éventuellement par exemple la communication avec un serveur.

Cette structuration poussée procure plusieurs avantages :

- l'entité "base de règles" est particulièrement intéressante dans le cas des guides de choix. En fait, on peut créer autant de bases de règles qu'il y a de types d'appareils, ces bases étant indépendantes. Le moteur pourra explorer telle ou telle base selon les besoins de l'utilisateur.

- la base de faits récapitule à tout moment les réponses qu'a communiquées l'utilisateur aux questions du logiciel. Cette base est un élément important de l'interactivité; l'utilisateur peut y accéder pour la visualiser, ou modifier une valeur dont il n'est pas certain.

- c'est dans le moteur que l'on va insuffler les propriétés du logiciel. Dans le cas des guides de choix, il nous a paru intéressant d'insister sur deux modes de fonctionnement :
  - le mode *déduction*, dans lequel le moteur va rechercher tous les appareils compatibles avec les contraintes de l'utilisateur ;
  - le mode *vérification*, dans lequel l'utilisateur pourra vérifier si tel appareil qu'il désire par exemple acquérir, est compatible avec l'environnement expérimental et les performances attendues.

Pour toutes ces raisons, la technique SBC retenue s'avère particulièrement efficace pour l'aide à la décision, voire la sensibilisation et la formation des utilisateurs.

EXEMPLE DE DEMONSTRATION

Ce paragraphe, qui ne peut pas remplacer une démonstration réelle, a pour but de mettre en évidence les principales étapes d'une session. La base de règles choisie en exemple concerne un guide de choix de chromatographe.

Nous distinguons deux menus principaux. Le premier, intitulé "sommaire" (fig. 1), contribue essentiellement à la gestion de l'espace utilisateur, donc des bases de faits. Un utilisateur peut y sauvegarder sa base de faits, pour éventuellement, à la lumière des sollicitations du logiciel, mieux s'informer sur ses contraintes expérimentales, et recharger plus tard cette même base de faits. Il appartiendra aux futurs gestionnaires du serveur de décider si de telles possibilités doivent être conservées, ou bien si des limitations doivent être introduites (une seule sauvegarde autorisée par utilisateur, options réservées à certaines personnes,...) , ceci afin de ne pas risquer un encombrement prématuré des mémoires physiques de l'ordinateur.

Ce premier menu permet aussi l'accès au menu "expertise" (fig. 2), clef de voûte de l'aspect aide à la décision. Ce menu, ainsi que les deux suivants qui ne sont pas détaillés ici, permet de réaliser les actions suivantes sur une base de faits :

- la visualiser (fig. 5)
- l'effacer
- supprimer une occurrence

- la compléter en activant les modes "déduction" ou "vérification".

Dans ce dernier cas, le système s'informe sur les contraintes de l'utilisateur en l'interrogeant (fig. 3). Chaque question concerne un paramètre pour lequel la réponse peut être "oui", "non", une valeur, ou "sans importance", c'est-à-dire que l'utilisateur demande au système de ne pas intégrer ce paramètre dans ses investigations.

A chaque interrogation, l'utilisateur peut demander des précisions sur la nature du paramètre en question (fig. 4); ces précisions peuvent concerner non seulement la définition du paramètre, mais aussi les valeurs les plus courantes, les valeurs limites pour la technologie actuelle (ceci afin de ne pas aboutir à un échec dans la sélection d'appareils dans la mesure où la valeur du paramètre serait aberrante).

Après une série de saisies illustrée par la fig. 5, le système propose en mode déduction tous les appareils compatibles avec le contexte communiqué (fig. 6). A chaque appareil est associée une série d'astérisques dont chacune signale un paramètre non renseigné par le constructeur. Dans ce cas, la sélection s'effectue à partir d'un nombre réduit de paramètres. L'utilisateur peut prendre connaissance des paramètres ignorés en sélectionnant le numéro d'ordre de l'appareil (fig. 7). Il accède alors à un fichier contenant la liste des critères manquants. On pourrait envisager de faire suivre cette liste par un commentaire - éventuellement rédigé par le constructeur - décrivant l'appareil.

En mode vérification, le fonctionnement du système est le même en ce qui concerne les interrogations sur la valeur des paramètres. Deux différences apparaissent par rapport au mode déduction :

- au début d'une session, le système demande à l'utilisateur quel appareil est à vérifier parmi la liste de tous les appareils contenus dans la base de règles (fig. 8).
- à la fin de la session, le système informe simplement l'utilisateur sur la compatibilité ou non de l'appareil choisi avec ses critères.

En cas de non compatibilité, il suffit à l'utilisateur de basculer en mode déduction sans modifier la base de faits pour obtenir la liste des appareils compatibles.

CONCLUSION

La formalisation télématique des guides de choix en instrumentation chimique permet de résoudre deux problèmes :

- celui de la **diffusion** et de la mise à jour des connaissances,
- celui de l'**aide à la décision.**

L'aspect communication doit maintenant être étudié en fonction des capacités informatiques actuelles ou projetées de l'EXERA, de ses ambitions et possibilités d'investissement. Le schéma de communication que nous préconisons (Fig. b) nous paraît le plus apte à intégrer une évolution vers d'autres applications.

L'aspect aide à la décision s'appuie sur la technique des systèmes à base de connaissances. Cette structuration du logiciel procure plusieurs avantages :

- l'utilisateur a son propre espace de travail (la base de faits), ce qui favorise l'interactivité;
- toutes les connaissances relatives à une gamme d'appareils sont regroupées dans une base de règles indépendante. Le formalisme des règles, que nous n'avons pas développé ici, permet une maintenance aisée qui ne nécessite pas de compétence informatique.

Le développement en langage prolog du moteur d'exploitation des bases de connaissances procure un grand confort de mise au point.

Il appartient maintenant à l'EXERA, à la lumière de la démonstration effectuée le 11 octobre 1989 et des arguments développés dans ce rapport, de définir les moyens qu'elle prévoit d'affecter à une version opérationnelle, et d'établir un cahier des charges visant à préciser les objectifs à atteindre.

L'école des Mines de Saint-Etienne se tient à la disposition de l'EXERA pour poursuivre ces travaux.

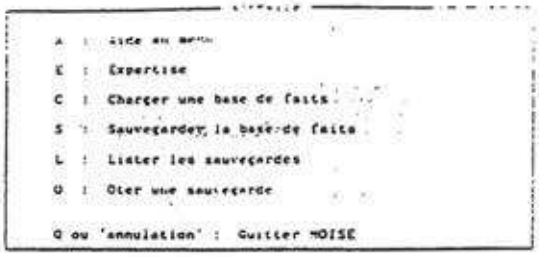


fig. 1

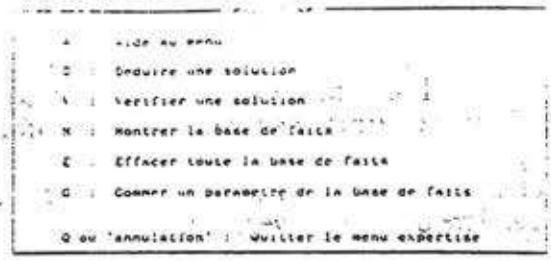


fig. 2

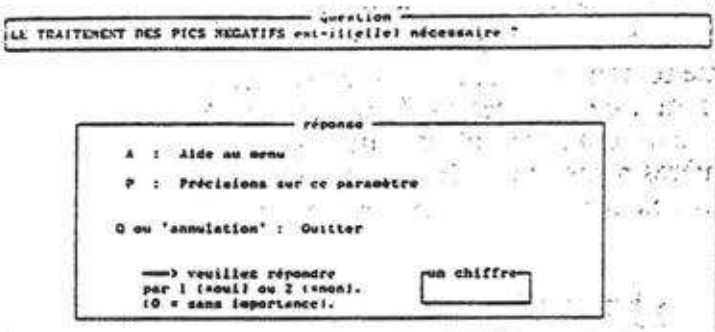


fig. 3

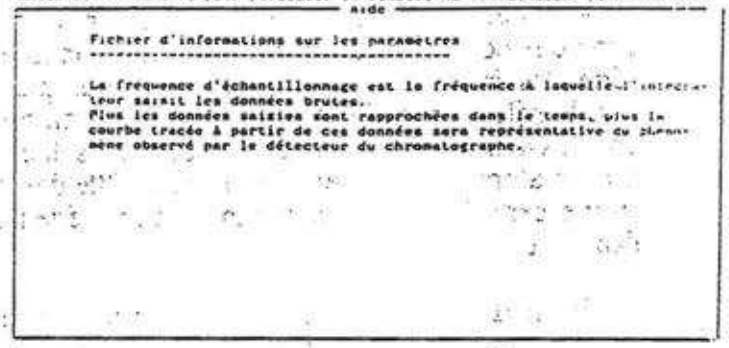


fig. 4

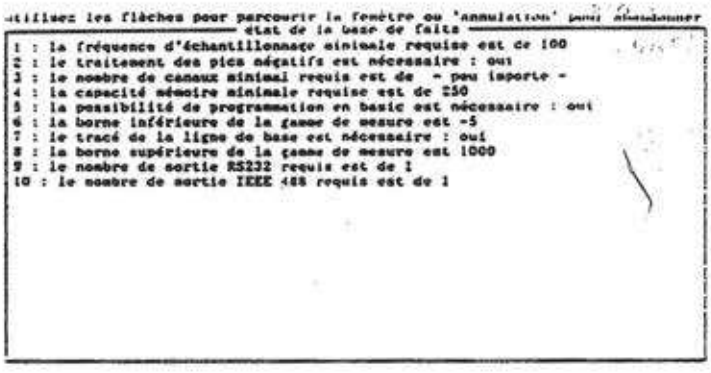


fig. 5

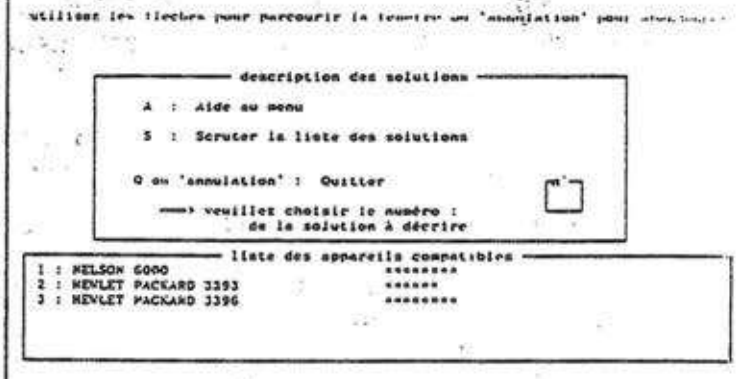


fig. 6

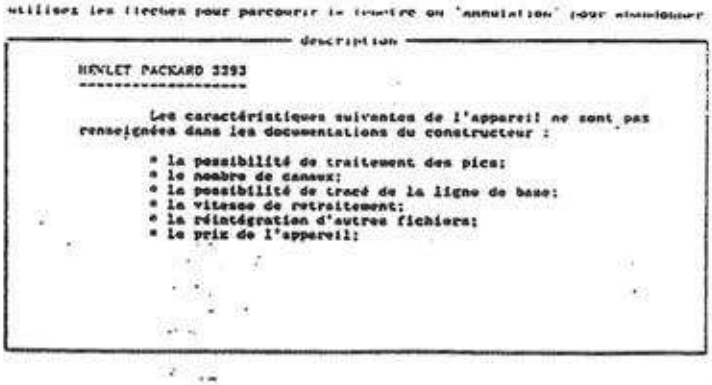


fig. 7

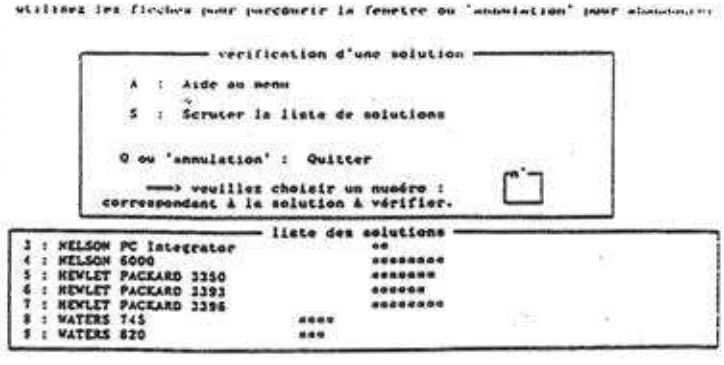


fig. 8

THESE de l'ECOLE NATIONALE SUPERIEURE des MINES de PARIS et
de l'ECOLE NATIONALE SUPERIEURE des MINES de SAINT-ETIENNE.
(1989; numéro d'ordre 35 HD)

AUTEUR : François-Noël CRES

TITRE :

Contribution des systèmes à bases de connaissances en sciences de l'eau;
Promise : un simulateur de projet;
Moïse : un système de diagnostic en assainissement autonome.

SPECIALITE : Hydrologie et Hydrogéologie Quantitatives

Résumé :

La formalisation de la connaissance sous forme de règles permet de modéliser des problèmes dont on ne possède pas d'algorithme de résolution. Ce sont les systèmes à bases de connaissances (SBC) qui vont permettre de séparer la "machinerie informatique" du domaine d'expertise. C'est à travers deux applications du domaine de l'eau que les SBC sont mis en œuvre. "Promise" est un simulateur de projet "intelligent" et interactif capable d'intégrer dans son comportement le passé du projet, d'activer n'importe quelle procédure externe, de poser des questions programmées...La mise en forme des règles de simulation est une étape vers une meilleure maîtrise des facteurs d'un projet d'assainissement. "Moïse" est un système de diagnostic en assainissement autonome dont on montre que le formalisme de règles peut être adapté à plusieurs aspects : choix d'un dispositif d'assainissement autonome, dimensionnement, maintenance, cohérence de la base de faits. Une version télématique de Moïse a été développée. L'association SBC-télématique est envisagée dans un autre domaine d'application (instrumentation chimique).

Mots clés :

Systèmes à bases de connaissances
Enseignement assisté par ordinateur
Systèmes experts
Simulation de projet
Projet d'assainissement
Assainissement autonome
Aide à la décision
Télématique