212.0

89 HY

Reprinted from

COMPUTERS & GEOSCIENCES

Vol. 15, *No.* 3, *pp.* 255–267

HYDROLAB®: AN EXAMPLE OF A NEW GENERATION OF COMPACT EXPERT SYSTEMS

PATRICE POYET[1] and MICHEL DETAY[2]

[1]ILOG S.A., 2 Avenue Galiéni, B.P. 85, 94253 Gentilly Cédex and Université de Nice, Laboratoire de Geologie et Géochimie, Parc Valrose, 06034 Nice Cédex and [2]GEOLAB, Sophia Antipolis, B.P. 15, 06560 Valbonne Cédex and Institut Méditerranéen des Géosciences, Avenue Paul Arène, 83300 Draguignan, France

LIBRARY
INTERNATIONAL REFERENCE CENTRE
FOR COMMUNITY WATER SUPPLY AND
SANITATION (IRC)

PERGAMON PRESS PLC
OXFORD · NEW YORK · BEIJING · FRANKFURT
SÃO PAULO · SYDNEY · TOKYO · TORONTO
1989

212·0–89 HY-6609

Computers & Geosciences Vol. 15, No. 3, pp. 255–267, 1989
Printed in Great Britain. All rights reserved

0098-3004/89 $3.00 + 0.00
Copyright © 1989 Pergamon Press plc

Rec ; 15n
6113

HYDROLAB®: AN EXAMPLE OF A NEW GENERATION OF COMPACT EXPERT SYSTEMS

PATRICE POYET[1] and MICHEL DETAY[2]

[1]ILOG S.A., 2 Avenue Galiéni, B.P. 85, 94253 Gentilly Cédex and Université de Nice, Laboratoire de Géologie et Géochimie, Parc Valrose, 06034 Nice Cédex and [2]GEOLAB, Sophia Antipolis, B.P. 15, 06560 Valbonne Cédex and Institut Méditerranéen des Géosciences, Avenue Paul Arène, 83300 Draguignan, France

(*Received* 28 *January* 1988; *revised* 22 *July* 1988)

Abstract—The system described in this paper is an expert system implemented on a PC. The aim was to develop efficient software, running on hardware available in the developing countries, with a high level of expertise in the field of village water-supply programs. The hardware constraints led us to develop a specialized and sophisticated software architecture in order to reach a high-performance level that included many of the useful features generally available for the user on mainframes with larger tools or with expert-system shells. The underlying mechanisms of HYDROLAB do not rely on fully generic schemes, but rather on well-suited solutions to application-dependent problems.

Key Words: Artificial intelligence, Expert systems, Personal computers, Hydrogeology.

INTRODUCTION

During the last 20 yr, African countries have been improving the water supplies for their rural populations. The search for water has become crucial, and in 1988 more than 80% of the rural population in the developing countries are yet without drinkable water. In the past 20 yr hydrogeologists have carried out numerous studies which have led to the definition of well-location techniques for village water-supply programs.

The primary importance of a drinkable water supply, the existence of a large amount of statistical data that could be applied to the location criteria, the repetitive nature of the steps to be carried out for a hydrogeological search and the African technicians' training needs, all led us to develop an expert system. HYDROLAB embodies the major techniques used by experts in the field and draws on a considerable wealth of practical knowledge gained from numerous drilling campaigns that were carried out in Africa, particularly in North Cameroon. HYDROLAB is the result of a close partnership between P. Poyet who developed and implemented the program architecture and M. Detay who brought hydrogeological expertise to the project.

We believe that the system should run on microcomputers, in order to be available and useful widely to geologists who are faced with the real problems of developing countries. For this reason, HYDROLAB was written in Turbo-Prolog™, a high-level language that includes some of the main characteristics of prolog as defined by Colmerauer (1977, 1983) and Clocksin and Mellish (1984), but that also allows an efficient complication on microcomputers. Some of the original Prolog interpretative mechanisms have been suppressed in this dialect because they were resource consuming. The global performance and the compactness of the generated code however are impressive. For more details on Prolog efficiency refer to Warren (1977) and Borland (1986).

The ability of Turbo-Prolog™ to permit a modular approach to large-system design, enabled us to write HYDROLAB in fourteen modules, representing a global package of about 5000 lines of Prolog source code. An executive is generated by the linker, and may be distributed as a standalone on a floppy disk, whereas the complete system, with the related databases is distributed on two floppy disks.

As far as the user is concerned, the system acts at the dialog level as a diagnostic-like expert system, asking questions, analyzing user answers, and building plans in order to schedule the appropriate set of actions and to explore constrained parts of a large search tree. Interaction with the system is achieved through a natural language module. We developed a wide set of self-explanation functionalities in order to use this system in computer-aided learning, mainly for African technicians, in the water-supply domain.

A control panel is associated with the inference engine, in order to display the internal activity of the system at a high symbolic level. The user is informed in this way of the current goal and subgoal pursued by the system and of the branching factor of the search tree. The panel also shows the current position and the depth in this tree, the current rule fired from the conflict set and the number of solutions encountered so far during the run.

The aim of the system is to build a model of the user's problem and to try to match it initially, using a fuzzy logic, with a set of typical known well patterns that have emerged from well-drilling campaigns in Africa and are stored by HYDROLAB in a database-like structure. When this reasoning step fails, the system builds a new plan and schedules it in order, first, to get more data related to information that is lacking, which could help to reach a solution. It then tries to rematch the currently described situation with real solutions and finally with generic ones.

For all situations, the system is able to make a diagnosis about the user's situation, to give advice in order to increase the user's average success, and to evaluate the hydrodynamic characteristics of the future well, according to the experience based on the statistical processing of 1080 boreholes in Africa (Detay, 1987; Bernardi and Detay, 1988).

THE GENERAL ARCHITECTURE OF HYDROLAB AND THE DOUBLE PASS MODEL OF CONTROL

The control structure of HYDROLAB is derived from previous research on expert-system shells that are suited for the development of large tools written in Le-Lisp® (Haren and others, 1985a, 1985b; Poyet and coworkers, 1986–1988; SMECI, 1988).

The main parts of HYDROLAB are a planner that collects tasks and a scheduler that activates them. Each task stores partial results in the Prolog main memory database independently of the work accomplished by other modules, in a blackboard-like manner (Hayes-Roth, 1984; Nii, 1986a, 1986b). The scheduler starts its work on an initial plan which is built according to a set of responses given by the user to some discriminant questions that are asked in the initialization phase.

Replies given to questions relative to the geographical well location (X, Y, Z), lead HYDROLAB to make some assumptions on the geological context according to structural data stored in a country-specific database, and to initialize a plan well suited to the relevant context.

This set of initial actions corresponding to the supposed geological context then is scheduled, and each task in the list is activated by a recursive scheduler. During the scheduling, the user may backtrack on each of these tasks, as they are stored in a stack, in order to modify some of the previously given parameters. Tasks are viewed by the scheduler as a list of actions to be accomplished, or as a set of predicates to be scheduled in a particular order. This list of tasks is constructed dynamically and may be modified in a reflexive way by the system as reasoning goes on.

The simplest possible code for such a scheduler could be (Fig. 1):

```
scheduler([]) :- ! .
scheduler([Task | Other_tasks]) :-
     task(Task),
     scheduler(Other_tasks), ! .
```

Figure 1. Simplest sort of scheduler.

Another simplified version of such a task scheduler, enabling each task to modify the list of tasks can be written as (Fig. 2):

The questions asked by the system represent the visible part of the activated tasks. They tend to be focused on a subpart of the overall problem as they correspond to subgoals previously identified and pushed into the list of tasks by the planner.

Each task scheduled is processed by a generic and compact code: the task predicate which is built up of three clauses described in the following way (Fig. 3):

When the initial plan is accomplished, the system activates the inference module and tries matching the user data kept in the blackboard with the known solutions stored in a Prolog disk database (these are termed analogical references), and loaded in main memory by a "consult" system call. This is done using fuzzy logic (Arrivet, 1987). If this step fails for all the available boreholes databases, the system switches back to the control module and tries to evaluate more accurately the user situation in terms of water supply and water-bin, in order to identify lacking or complementary data that could help to reach a known analogical solution or even, later on, a generic one.

The associated tasks are identified and again pushed by the planner into the list of tasks, and the control is passed for the second time to the scheduler. A possible way to write such a task collector is in a simplified way (Fig. 4):

```
scheduler :-
     tasks_on_blackboard([]),!.  ; nothing to schedule, then slash.
scheduler :-
     tasks_on_blackboard([First_current_task,_]),
                                 ; a global fact
     task(First_current_task),   ; but each task can modify the task list
     pop_the_first_task,         ; pop if done
     adjust_task_structure_if_necessary,
                                 ; what should be done next ?
     scheduler.                  ; each task leaves pending choices and the
                                 ; recursive call pushes them into the PROLOG
                                 ; stack.
Comments appear as in LISP:      ; This is a comment.
```

Figure 2. Simplified code for dynamic scheduler.

```
task(Task_filter) :-           ; descending clause of the search tree,
    get_panel(Task_filter,Goal,Context,Rule,Probability),
                               ; obtains some variables according to the task filter,
    get_message(Task_filter,Message),
                               ; obtains the message associated with the task filter,
    update_control_panel(Goal,Context,Rule,Probability),
                               ; updates the control panel of the interface module,
    optional_message_printing(Task_filter),
                               ; special messages if needed,
    change_activity(activate,toggle),
                               ; the backtrack is now activated
    nl_processing(Message,Response,Task_filter)
                               ; obtains and analyses user reply. If a backtrack is
                               ; required, asserts the destination then cuts and fails.
    change_activity(inactivate,toggle),
                               ; no backtrack available from this point,
    task_entry(Task_filter,Response).
                               ; effective entry for the task.

task(Task_filter) :-           ; first possible clause for backtrack,
    backtrack(Symbol),         ; a backtrack control fact was asserted in the
                               ; blackboard and a destination has to be reached,
    symbol <> Task_filter,     ; the destination not yet has been reached,
    clear(Task_filter),        ; undo previous actions,
    !,                         ; no pending choices should be left,
    fail.                      ; fails, in order to pop the tasks stack.

task(Task_filter) :-           ; second possible clause for backtrack which can be
                               ; used either when: symbol = Task_filter (destination
                               ; reached) or in a step-by-step backtrack.
    not(generate_backtrack(Task_filter)),
                               ; the current task did not generate the backtrack,
    !,                         ; no pending choices should be left,
    clear(Task_filter),        ; undo previous actions,
    retract_backtrack,         ; retract the control fact used to explore the tasks
                               ; stack,
    task(Task_filter).         ; recursive entry.
```

Figure 3. Simplified code for three clauses of generic task predicate.

At the end of this cycle, the blackboard is supposed to be complete and the control is transferred definitively to the inference module, which always is able to conclude, even if the only solution located is a generic one. This will be explained later.

We termed this type of model *the double pass model of control*. Usually, knowledge-based systems use a corpus of knowledge to try to deduce some new facts using a forward-chaining strategy or to prove some goals according to a backward inference model. Generally both strategies are used simultaneously (Clayton, 1985; Poyet and De La Cruz, 1987; Poyet, Haren, and De La Cruz, 1987). In powerful control structures, when the reasoning is stopped in some context, the system is able to react and selects a task (viewed as a set of rules) from among a set of actions

```
task_collector([Datum_tested | Q], [Datum_tested | Tail]) :-
     is_datum_to_be_pushed_in_task_list(Datum_tested),
     !,
     task_collector(Q,Tail).
task_collector(Task_list,[_ | Tail]) :-
     task_collector(Task_list,Tail).
task_collector([],[]).

     ; The task_collector predicate is called with an adapted filter in order to
     ; recognize the plausible list of tasks to be tested for scheduling by:

evaluate_possible_lacking_data(L_water_supply_tasks,
     Some_filter) :-
     get_candidates_according_to_filter(Some_filter,
     Candidates),
     task_collector( L_water_supply_tasks,Candidates).

     ; The appropriate list of task is then activated. According to the first simplified
     ; scheduler this could be written as:

schedule(L_water_supply_tasks).
```

Figure 4. Simplified code for task collector.

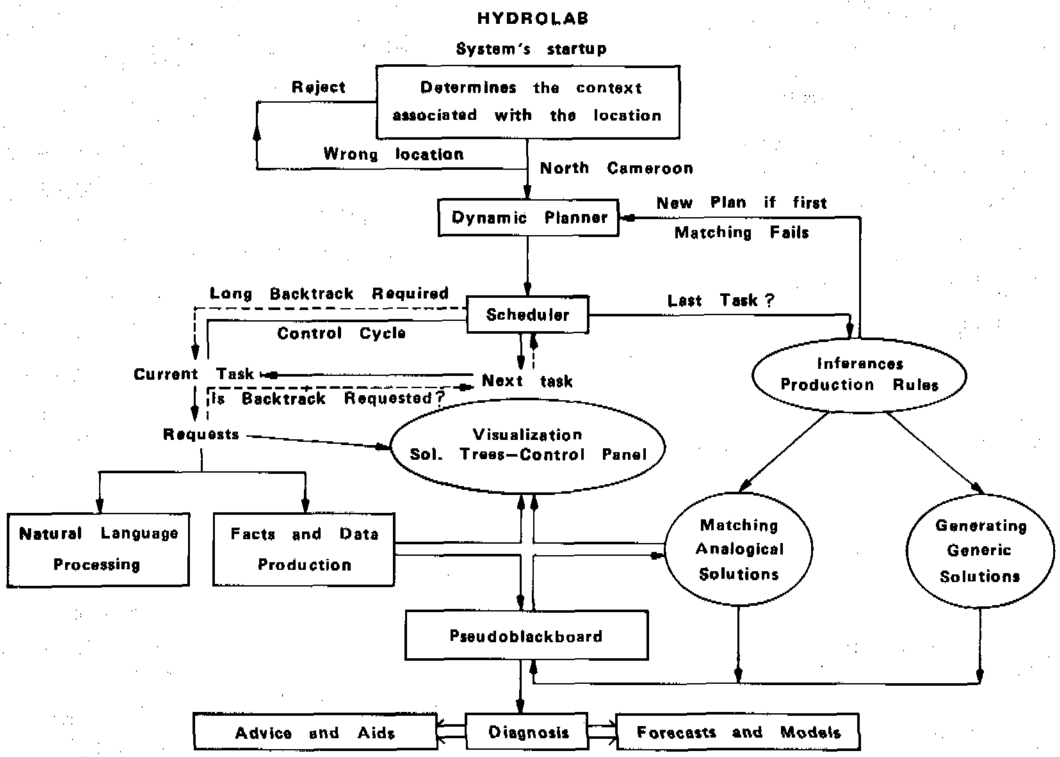

HYDROLAB
System's startup
Determines the context associated with the location
Reject
Wrong location
North Cameroon
Dynamic Planner
New Plan if first Matching Fails
Long Backtrack Required
Scheduler
Control Cycle
Last Task ?
Current Task
Next task
Is Backtrack Requested?
Inferences Production Rules
Requests
Visualization Sol. Trees—Control Panel
Natural Language Processing
Facts and Data Production
Matching Analogical Solutions
Generating Generic Solutions
Pseudoblackboard
Advice and Aids
Diagnosis
Forecasts and Models

Figure 5. Simplified HYDROLAB system flowchart.

considered to be useful in such a situation (Poyet, 1987). If no task selection can be done and no other rule can be fired in the current context, the reasoning process stops and aborts.

According to the model of control presented here, once the first pass has been accomplished in response to the first dynamic planning phase and if the problem-solving process fails, the system is able to reason about what it should know or acquire from the present set of data stored in the blackboard in order to reach a solution. The system knows how to react, according to the current and incomplete description of the problem, so as to identify new goals and to generate a new plan dynamically. HYDROLAB includes a reflexive and explicit description of its solution space which is used by a specialized knowledge source. This knowledge source is dedicated to identifying crucial but lacking data in the current blackboard to construct new plans that are suited to recognizing existing solutions or generating original ones.

Briefly, we can say that the first pass tries to collect some appropriate data according to a supposed geological context and to recognize an analogical solution. When this step fails, the system uses a reflexive description of its solution space in order to identify the culprit-lacking data which could be useful during the matching process or necessary for building up generic solutions that rely on criteria of a high symbolic level. Then the second pass is scheduled for catching the culprits and for reaching a solution. In the worst situations, this double-pass model of control leads to generic solutions.

Figure 5 illustrates of the double-pass model of control. Once the context has been determined by the system, rejecting problems located outside North Cameroon, a plan is generated by the dynamic planner and then scheduled. Each task becomes the current system task, and in the situation of the normal control cycle, generates facts and data which are asserted in the Prolog main memory database (here termed the pseudoblackboard) according to the results of the natural language analysis. When a backtrack is requested during the control cycle and detected thanks to the natural language module, its processing is done according to Figure 3 and is represented by the dotted lines in Figure 5. The visualization module controls the display of the system's internal activity. When the last task is reached, the matching is done between the blackboard data and the analogical references. If this step fails, a new plan is generated thanks to the system's reactivity and transmitted again to the scheduler. This second control phase leads to the two concurrent models of solutions described in the relevant section.

The graphic representation of the general architecture of the system illustrates the major components involved and the logical links that exist between them (Fig. 5). We previously described the top modules of the diagram which are responsible for the apparent behavior of the software, including the planner, the scheduler and the associated control strategy; we will focus now on the deductive capabilities of HYDROLAB which rely on knowledge representation, inference policies, pattern matching and the two concurent models of solutions. We finally will look in some detail at the user interface.

KNOWLEDGE REPRESENTATION

An interesting feature of HYDROLAB is its ability to reason on a set of static databases which can be updated through a simple text editor. No modification needs to be done within the expert system code in order to take into account the new information stored in the databases. The databases that relate to the characteristics of the wells, store the modelization of objects and of relations existing between them, through the declaration of Prolog structures that are recorded as facts and loaded by a "consult" system call. The structure of Prolog objects is declared in the executive, and cannot be changed at the user level; only the amount of data manipulated by the system may be updated. The inference mechanisms are able to handle in a generic way an arbitrary number of well or drilling descriptions. The aim is to match as a first approach and using a fuzzy logic, the user data with at least one of the known well patterns stored in the databases. If this step fails, as was noticed during the control description, the system activates a new plan, and is able to draw inferences for generic solutions.

A large part of the knowledge embedded in the system relies on a static description of the geological and hydrodynamic characteristics of a large number of wells and boreholes that we carried out in Africa. This knowledge describes, for each database record, the observed characteristics of the drillhole, and the success encountered. One of the main objectives of the HYDROLAB architecture conception process was to mix, *in an efficient manner, an object oriented representation* of the declarative knowledge stored in various Prolog databases, with a powerful control structure based on task scheduling. The reader can refer to Albert (1985), for an introduction to the Prolog implementation of objects.

INFERENTIAL KNOWLEDGE

The inferential knowledge of HYDROLAB relies on the usual Prolog features. We did not develop a specific-rule language which could be expanded and interpreted at run-time as in (SMECI, 1988) or compiled as for a RETE match (Forgy, 1982) or for an inference net (Konolidge, 1979). If the user needs to modify the content of the inferential strategies, the rule modules have to be recompiled by the Turbo-Prolog™ compiler. Taking hardware constrains into account, the flexibility induced by a rule language would not have justified the code overhead inherent in the rule compiler. The loss of flexibility is obvious but from a semantic point of view, the Prolog rules of HYDROLAB can express, using specialized predicates, the same type of expressions as other sophisticated first-order rule languages; it should be remem-

```
rule_predicate_name(some_filter) :-
      condition1(attributei,...,fuzzy-rangei),
      condition2(attributej,...,fuzzy-rangej),
      ...
      conditionn(attributek,...,fuzzy-rangek),
      associated_action.
```

Figure 6. Pseudocode for generic rule predicate.

bered that HYDROLAB is able to run on machines of 512 kbyte of main memory. A HYDROLAB rule where rule_predicate_name is *the name* of a clause that represents a rule is similar to a set of Prolog rules and uses specialized predicates to verify the preconditions of the premises (Fig. 6). The predicates condition$_1$ (attribute$_i$, ..., fuzzy-range$_i$), ..., condition$_n$ (attribute$_k$, ..., fuzzy-range$_k$) are the names of specialized predicates which verify the precondition i for the attribute$_j$ according to the fuzzy range "fuzzy-range$_j$" which is the deviation for the attribute$_j$ to be used during the matching process; associated action is the conclusion, the effects of which are to produce either control or deductive reasoning actions. From an inferential viewpoint, the main associated action of a rule is to update the dynamic Prolog database used by HYDROLAB as a blackboard. The valid actions are to assert or retract dynamically Prolog structures or to modify existing ones.

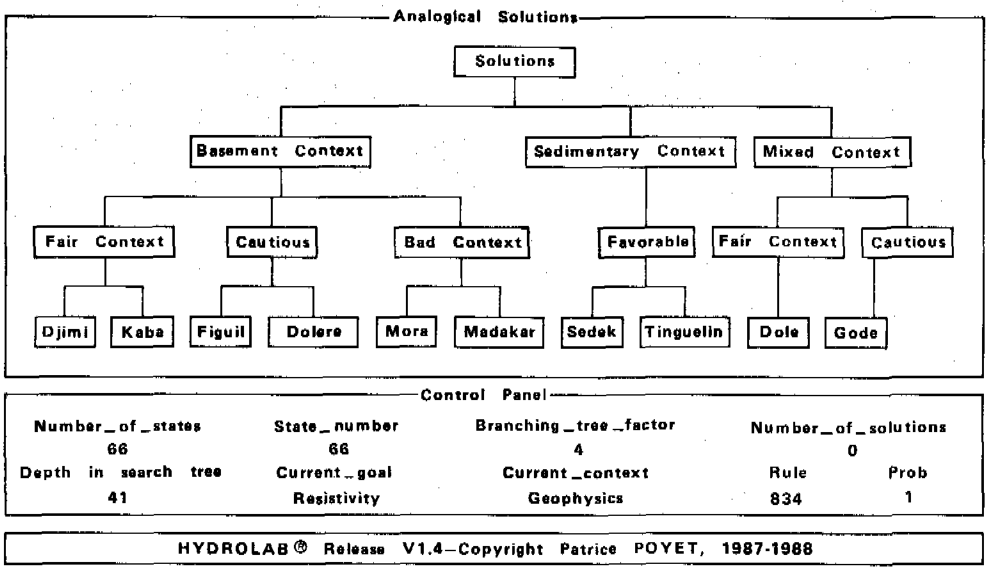
FUZZY LOGIC FOR AN EFFICIENT MATCHING

When the first phase of the task scheduling process ends, facts relevant to the user situation are all asserted and stored in the blackboard. As was noticed in the control section, the system may ask for complementary data later on, if the primary inference step fails. The next phase therefore is to pass the control for the first time to the inference module. For each solution known in the many available databases, the system tries to match the user data for each parameter with the currently considered solution. The matching is done for each parameter with specialized predicates that give a measure, according to a slot-dependent metric, of the distance between the user-given value and the recorded value for the supposed solution. When this processing is done for each attribute, the system is able to deduce a matching level which is a measure of the similarity between the plausible solution and the user data; the system also is able to compute a consistency measure which is significant of the number of channels used during the matching process to compute the previous similarity.

Expression (1) can be used to match the user data with a model and computes a global percentage of similarity:

$$\text{distance(user data, model)} = \min \left(100., \sum_{i=1}^{n} \frac{100.|\text{slot}_{1i} - \text{slot}_{2i}|}{n.\text{slot}_{1i}} \right) \quad (1)$$

where slot$_{1i}$ is the ith slot of the object 1 (a compound object which contains the user_data), slot$_{2i}$ is the ith



```
Number_of_states      State_number     Branching_tree_factor    Number_of_solutions
       66                  66                    4                       0
Depth in search tree  Current_goal     Current_context           Rule        Prob
       41              Resistivity       Geophysics               834          1
```

HYDROLAB® Release V1.4—Copyright Patrice POYET, 1987-1988

Figure 7. Example of analogical solutions tree, displayed during matching phase.

slot of the object 2 (the compound object taken as a model), and *n* is the number of slots used during the matching. In fact a similar calculation is processed slot by slot according to the appropriate metrics, before a sum can be done.

Predicates used to compute related distances, attribute-by-attribute, are context-dependent and have to be called with fuzzy ranges to take into account the diversity of the encountered metrics for the different attributes. A variant of this method was implemented first in a Naval expert simulator (Poyet and De La Cruz, 1987; Poyet, Haren, and De La Cruz, 1987), and is described fully in Arrivet (1987). We implemented an efficient subset of the complete method for HYDROLAB.

When the system is able to recognize at least a solution during this phase, the inference process is stopped, because an analogical solution corresponding to a real reference has been determined, and the control is transferred to the solutions' printing module. If not, the planner arranges a new plan which is activated by the scheduler as described in the control section. At the end of this second control cycle, the system rematches the blackboard's data with the references as a second attempt to identify some analogical solution; the system then elaborates a generic solution.

During the matching process, solution trees are displayed by the system (Fig. 7); the highlighted reverse-video items representing the current situation and the control panel window enabling the user to follow the reasoning. The trees are built up using the standard semigraphic IBM characters and do not require any enhanced graphic capabilities from the hardware. Scrolling functions are provided to visualize the large trees encountered in operational applications.

DIFFERENT MODES OF REASONING LEAD TO MANY TYPES OF SOLUTION

As was explained briefly, when the system is able to recognize a solution during the first inference cycle, we say that a real or *analogical* solution has been determined as the user data has been matched successfully with a known pattern stored in an object database. In this situation, we suppose that no other inference processing is necessary and the system stops reasoning. If not, we make an accurate analysis of the user's problem in order to schedule a new plan, which has to be suited perfectly to the peculiarities of the user situation. When all the complementary tasks have been scheduled, the blackboard is supposed to be as complete as possible, and the second phase of the inference process may start. The system looks first for analogical solutions then for *generic* ones. In all situations the system is able to elaborate a generic solution. A generic solution is a model conceived by HYDROLAB which does not rely on information obtained from a database. The model is based on abstract concepts such as the water-resource supplies and the characteristics of the water-bin, concepts of a high semantic level which have been elaborated from the primary user data. According to these concepts, the system always is able to draw an opinion from the current user situation, and to give related advice. Different types of aids in decision making are provided by the system, concerning the recommended location for the future drillhole, the estimated risk when drilling evaluated as a percentage of the chances of obtaining a positive drilling, as well as the foreseeable yield and specific yield which are predictable statistically in the given environment according to the developed models (Detay, 1987; Bernardi and Detay, 1988).

A REASONED PRINTING OF THE RESULTS

When a solution has been recognized, and the inference process stops, the system takes charge of the printing of the results. The analogical solutions recognized are supposed to match the user data, with a high degree of confidence, but according to the fuzzy logic used, a few attributes may be remote from the database model. For the latter, and according to an adaptable confidence threshold, no printing is carried out that could confuse the user. The best way to do this, is to use the same specialized predicates as those described in the fuzzy matching section, in order to verify the plausibility of each attribute on which the diagnosis is based before printing. This reasoning activity, considered by the system as normal tasks, is displayed by the control panel module, through the interface module side effect. This strategy leads to an intelligent printing, which visualizes only the reliable attributes, according to the same fuzzy logic used for their initial evaluation.

INTERFACE OVERVIEW THE NATURAL LANGUAGE PROCESSING MODULE

A specialized module manages the user interactions, and traps all the answers given by the user to the questions asked by the tasks. Each task activated by the scheduler, communicates with the user through the available entry points in this module; this guarantees the homogeneity of the interaction between the system and user.

This module is nested within the natural language module which is in charge of the lexical and syntaxical analysis of the input sentences. When a concept or a desired action is recognized in the input stream, the associated functions are activated as in a compiler code generation phase. Any of the following actions may be valid at any time: backtracking to the previous task, backtracking to a user-determined destination in the tasks stack, an action that is termed goal-directed backtracking (Poyet, 1987); explaining the reason behind the current goal or why something is being done currently, obtaining symbolic or numerical data with magnitude order checking; asserting or retracting facts in the blackboard; or switching to an on-line editor that displays all the documentation stored in a disk database to help the user.

All the user requests are expressed in a natural language form, and are interpreted by the system as one of the interactions described previously. For example, we can give the following sentences, and their associated meanings to the system:

The user sentence: ⟨*There are cretaceous limestones near the village*⟩

is interpreted by the system as: a new fact has to be asserted by the system into the blackboard. The new fact is a Prolog structure dynamically constructed by the system and recorded in the main memory Prolog database.

The user sentence: ⟨*I'd like to come back on the rainfall measures*⟩

is interpreted by the system as: a goal directed action has to be accomplished by the control structure, in order to pop control blocks from the stack and to backtrack to a task previously achieved.

The user answer ⟨*why*⟩: to the system request: enter the depth of the traditional wells (where a numerical value is expected), is interpreted by the system as: a user request has been made to the "teaching module" (*why* may be entered at any moment during the dialog). The current goal and a variable related to the user model are carried on by the tasks and are used by the teaching module to determine the appropriate explanation messages.

Finally, the user sentence: ⟨*I need somebody's help*⟩ (or any sentence having the same meaning) is interpreted by the system as: the user needs to access the on line documentation!

The system is able to help the user in two ways. Each time a task queries the interface module services, a short explanation of the type of answer expected by the system is displayed to the user in an on-line help window. However the user also may call explicitly the on-line editor by a help query at any time, and in this way access the complete detailed documentation stored by the system in a database. The database is loaded in the heap area of the main memory, which may be reallocated for other purposes. This is an economical memory management strategy. The basic vocabulary used by HYDROLAB for the natural language processing is stored in an ASCII database, and may be modified or updated if necessary. The description and representation of the grammar used by HYDROLAB is internal to the system and cannot be modified or accessed at the user level.

THE CONTROL PANEL MODULE

When a task makes a request to the interface module, a main side effect is to update a control panel that displays the current state of the expert system (as was done by the update_control panel call of Fig. 3). The panel shows the depth and the branching factor of the search tree, the current goal, and subgoal pursued by the system, the current rule fired from the conflict set and the number of solutions so far encountered. When the control is passed to the inference module, the user may ask for a step-by-step reasoning and discover each rule or action applied, and the level of success encountered by the system when attempting to schedule it.

Each task activated by the scheduler, and each rule fired by the task, updates a global Prolog structure of type panel stored in the blackboard. The current state of this structure is displayed in the panel window; it is achieved by means of a specialized part of the interface module. When the reasoning process is running, the user can visualize the activity of the system as the inference module queries the services of this subpart of the Interface Module.

AN OUTLOOK ON THE HYDROLAB SCREEN

The HYDROLAB screen is composed of three areas. The first, an interactive window is devoted to the dialog with the user; the second is used to switch from a help window to a reasoning window where the system displays the heading of the current rule fired, and the third area is the control panel described in the previous section. The on-line documentation is accessed by a full-screen editor which can be popped at any time over an active screen. The results are displayed through other windows with specialized templates. Figure 8 is an English reconstruction of the startup screen, showing the answer given by the system to a teaching request.

A working session with HYDROLAB appeals to the user (apart from the system's ultimate objectives), because of its ability to explain its own reasoning, understand a natural language dialog, explain its approach, and to visualize its internal state at any given time.

HYDROLAB-WHAT FOR?

HYDROLAB is a practical tool for the following reasons:

— a specialist who needs support on fieldwork when facing an unusual problem that requires a large amount of comparative data, and adapted inference mechanisms to match the observed data with the references stored in many databases. In this situation the system acts as in a fuzzy automatic-classification system, except that HYDROLAB is able to elaborate a generic solution from concepts of a high symbolic level and to work on a set of fully integrated database of Prolog objects with the expert system.

— a specialist who wishes to forecast some of the hydrodynamic properties of the future drillhole according to a large set of heuristic and numerical data. The numerical data manipulated by HYDROLAB may be the result of a statistical processing of the primary data. HYDROLAB includes some of the statistical results obtained by Detay (1987), many of them being related closely to heuristic criteria (as, for example, the foreseeable delivery of a drillhole equipped with a pump).

— a student who is learning the strategies of village water-supply programs. The system is able to explain the steps followed during the reasoning, and can be used to teach the hydrogeological step.

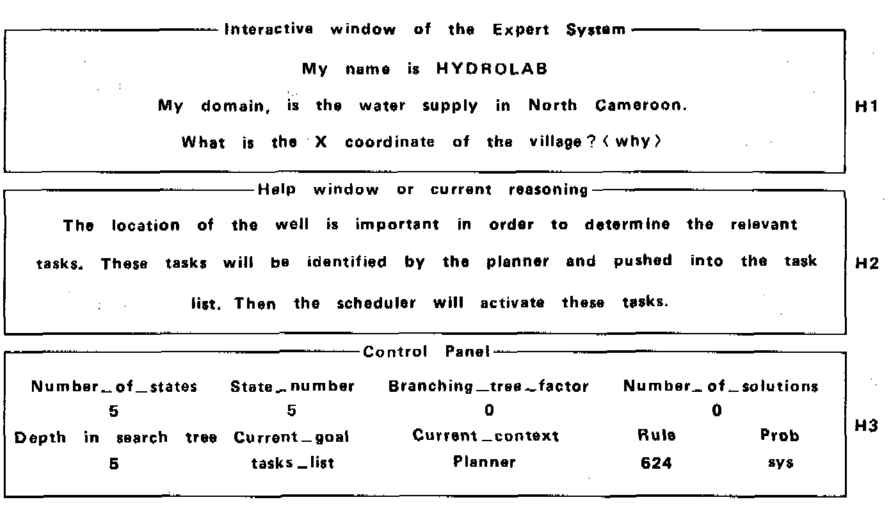
— any decision-maker who needs advice in order to increase his success rate.

The system is being used currently by GEOLAB in order to increase the percentage of productive drillholes in North Cameroon (Detay and others, 1986a, 1986b). Complete databases have been developed in this context and will be extended to other countries in order to handle a larger set of applications. North Cameroon is itself a large area ($155{,}000\,km^2$), and includes many different structural and geological contexts.

EXAMPLE OF HYDROLAB USER SESSION

Here is included a simplified example of a session in order to give the reader an idea on the implementation of the system's hydrogeological approach. A user session normally lasts for 20 min and the number of questions asked ranges from 75 to 200 according to the complexity of the problem to be solved.

To make this session understandable, and referring to the notations adopted for the windowing system (H1, H2, H3) for Figure 8, we use the following



```
---------------- Interactive window of the Expert System ----------------
                         My name is HYDROLAB
            My domain, is the water supply in North Cameroon.          H1
             What is the X coordinate of the village ? ( why )

------------------- Help window or current reasoning --------------------
   The location of the well is important in order to determine the relevant
 tasks. These tasks will be identified by the planner and pushed into the task   H2
              list. Then the scheduler will activate these tasks.

------------------------------ Control Panel ----------------------------
 Number_of_states   State_number   Branching_tree_factor   Number_of_solutions
        5                5                  0                       0
 Depth in search tree  Current_goal   Current_context      Rule        Prob     H3
        5              tasks_list        Planner           624          sys
```

Figure 8. English reconstruction of HYDROLAB screen at startup time. Different parts of windowing system devoted to specific logical functions are referred to as H1, H2, H3. This notation will be reused in section describing "Example of HYDROLAB user session".

symbols:

H.: means HYDROLAB and corresponds to
H1—the questions asked by the system (H1 subwindow),
H2d—the deductions made by the system (H2 subwindow),
H2H—the Computer Assisted Learning answers given by the system (H2 subwindow),
H2h—the help in entering the answers (H2 subwindow),
H3—the state of the inference engine thanks to the control panel (H3 subwindow),
HH—the on-line help function, where HH refers to the full page display (full screen editor).

U.: means User and corresponds to the user's answers made using the H1 subwindow. They can be of several types:
(D)—numerical data,
(Y/N)—answer yes or no,
(NL)—answers in natural language,
(W)—questions asked to the system as: Why and CAL function,
(H)—help requested,
(B)—backtrack: for example if the user has made a mistake and would like to go back to the question where the error is located.

In all situations, the possible answer is put into brackets, followed by the selected answer in the given example of the session.

C.: indicates comments designed to either explain the system's approach or the ins and outs of the questions asked, and so on. *The following font will be used.*

As the H3 subwindow contains nine subassemblies (Fig. 8), we decided not to detail its contents, in order not to make this example considerably heavy. We have not presented the Help function either which includes several dozens of screens. The backtrack function has not been presented either; it is possible to go back at any time on any type of question asked previously.

C. *At the beginning of the session, the system tries first to define the general environmental conditions: geographical, geological, hydrological, climatic, morphological, and so on.*

H.H1: What is the longitude of the village (*X* coordinate)?
H2h: I expect a numeric value in degrees and minutes
U. (D/W/H): 12°23′
H.H1: What is the latitude of the village (*Y* coordinate)?
H2h: I expect a numeric value in degrees and minutes
U. (D/W/H): 10°40′
H.H1: What is the altitude of the village (*Z* coordinate)?
H2h: I expect a numeric value in meters
U.(D/W/H): 450.
H.H1: Did you perform photo-interpretation?
H2h: Enter your answers in natural language
U.(Y/N/W/H): Yes
H.H1: Can you see faults on the aerial views?
U. (Y/N/W/H): Yes
H.H1: Main fault trend?
H2h: The answer must be entered in relation to the North on the basis of 360°
U. (D/W/H): 47
H.H1: Other fault trend?
U. (Y/N): Yes
H.H1: Main fault trend?
H2h: The answer must be entered in relation to the North on the basis of 360°
U. (D?W/H): 123
H.H1: Other fault trend?
U. (Y/N): No
H.H1: Describe the fault state and quantify the fault network.
H2h: Enter your answers in natural language
U. (NL/W/H/B): there are several faults over kilometers in the village area
H.H2d: Fault state valid and stored
H2d: Fault density valid and stored
H.H1: Considering the important fracturing, do you know if there are any nodal points near the village?
U. (Y/N/W/H/B): No
H.H1: Do you have precise indications on the lithology?
U. (Y/N/W/H/B): Yes
H.H1: Describe the lithological blocks observed, one after the other, with their characteristics.
H2h: Enter your answers in natural language
U. (NL/W/H/B): after analysis of the aerial views, the village is situated in a crystalline block. It has to be noted on the geological map the presence of granitic rocks with alkali feldspar dating back from the Cambrian era
H.H2d: name of rock valid
H2d: age of rock valid
H2d: valid texture characteristics
H2d: as far as I can see, you are in the "basement context"
H.H1: Do you know the pluviometry of the area concerned?
U. (Y/N/W/H/B): Why?
H.H2h: The pluviometric data correlated with the size of the watershed, the field and geomorphological data, give an idea of the portion of infiltrated water and thus of the aquiferous possibilities in the region.
H.H1: Do you know the pluviometry of the area concerned?
U. (Y/N/W/H/B): Yes
H.H1: What is the average pluviometry in the area?
H2h: I expect a numeric value in mm
U. (D/W/H): 900
H.H1: What is the geomorphological context?
H2h: Enter your answers in natural language
U.(NL/W/H/B): the village is at the bottom of a massif
H.H2d: You are in a mountainous region

H.H1: What is the size of the watershed upstream from the village?
H2h: I expect a numeric value in km^2
U. (D/W/H): 12
H.H1: Is there a brook in the area concerned?
U. (Y/N/W/H/B): Yes
H.H1: For how many months per year does it flow?
H2h: I expect a numeric value in months
U. (D/W/H): 8
H.H1: How far is it from the village?
H2h: I expect a numeric value in km
U. (D/W/H): 0.4
H.H1: According to you what is its hydrogeological importance?
H2h: Enter your answers in natural language
U.(NL/W/H/B): the brook is near the village, it is certainly of major importance
H.H2d: Consequently, I consider the brook as playing an important part
H.H1: Considering the importance of the brook, do you have its detailed characteristics?

C. *If you answered yes, the system would have tried to know the degree of inclination of the stream, the size of its channel, the aspect of the bed, and so on. All this information will enable an estimation of the possible supply of the underground water flow from the underflow sheet of the brook.*

H.H1: Can you give the difference of altitude between the stream bed and the village?

C. *The system understood that the village is in the basement zone. It also knows that the decompressed and fractured zone of the basement is generally 30 m deep. It tries to know the difference of altitude between the village and the brook. This parameter will enable it to estimate the possible supply of the decompressed zone of the basement under the village.*

U. (Y/N/W/H/B): Yes
H.H1: What is the difference in level?
H2h: I expect a numeric value in m
U. (D/W/H): 5
H.H1: During the dry season, do the villagers dig catch basins in the brook?

C. *The system tries to estimate the possible supply of the basement decompressed zone in the dry season.*
U. (Y/N/W/H/B): Yes
H.H1: How deep are these catch basins?
H2h: I expect a numeric value in m
U. (D/W/H): 1.2
H.H1: Are they supplied during the entire dry season?
U. (Y/N/W/H/B): Yes

C. *The catch basins supplied in the dry season show the presence of a perennial underflow sheet. The latter supplies the decompressed zone of the underlying basement. Thus, the system has identified a determining hydrogeological element.*

H.H1: Did you carry out a field study?

C. *The system asks this question but cannot make any relevant recommendation if this field study has not been carried out. HYDROLAB is designed as a field system.*

U. (Y/N/W/H/B): Yes
H.H1: Do field studies give more information on the lithology?
U. (Y/N/W/H/B): Yes
H.H1: Describe the lithological blocks you identified
H2h: Enter your answers in natural language
U.(NL/W/H/B): there are gneiss outcrops intersected by quartz sills
H.H2d: Name of rock valid
H2d: I acknowledge the presence of metamorphic rocks in the given series
H.H1: Can you give the current water supply (wells, springs, catch basins, drillings)?
U. (Y/N/W/H/B): Yes
H.H1: Are there traditional wells?
U. (Y/N/W/H/B): Yes
H.H1: Describe the characteristics of the wells one after the other
H1: Traditional well 1, depth?
H2h: I expect a numeric value in m
U. (D/W/H): 17
H.H1: Is it supplied all year or sometimes dry?
H2h: Enter your answers in natural language
U. (NL/W/H/B): this well does not run dry in the dry season
H.H1: What is the static level of this well?
H2h: I expect a numeric value in m
U. (D/W/H): 16.65
H.H1: Are there other traditional wells?
U. (Y/N/W/H/B): No
H.H1: Are there modern wells?
U. (Y/N/W/H/B): No
H.H1: Are there drillings previous to the current campaign?
U. (Y/N/W/H/B): No
H.H1: Did you carry out a geophysical surveying?
U. (Y/N/W/H/B): Yes
H.H1: What is the depth of the infinite resistance level?
H2h: I expect a numeric value in m

C. *Considering we are talking about the basement, the system tries to estimate the thickness of the saprolite reservoir.*

U. (D/W/H): 19
H.H2d: Therefore, I consider this value as the geophysical thickness of the basement hanging wall.

H.H1: Are there local anomalies in the thickness of the hanging wall?

C. *The geophysical surveying makes it possible to draw a map of the basement hanging wall. One then can locate areas with higher alteration which can be correlated with fractured zones.*

U. (Y/N/W/H/B): Yes
H.H1: Can these anomalies be correlated with faults observed on the aerial views?
U. (Y/N/W/H/B): Yes
H.H1: Are these anomalies near the brook?
U. (Y/N/W/H/B): Yes
H.H1: What is the resistance of the hanging wall in the area concerned?
H2h: I expect a numeric value in ohm/m

C. *The value of the hanging wall resistance coupled with the thickness enables the calculation of the overall longitudinal conductance which is a way of estimating the aquiferous potentialities of the area.*

U. (D/W/H): 25
H.H1: Do you want to carry out a step-by-step reasoning?

C. *The step-by-step reasoning allows the user to visualize the inferences carried out by the system (subwindow system H3) and to understand the expert's reasoning.*

U. (Y/N/H): No

C. *The system then goes through the inference cycle. It analyses the data set described by the user. Therefore it uses its analogical database. There may be several situations:*
— either it locates a similar context in its analogical database, it then will make recommendations for the implementation of the drilling and determine the analogical type situations
— or it does not locate any similar situations to the described context in its analogical database. The system then will evaluate the missing data, plan its approach again, and ask the user a set of questions which may lead to the solution.

We have not presented the replanning phase as in all situations it is too long and extensive. After replanning, the system tries to locate an analogical solution then jumps to the generic solutions and finally makes all its recommendations in relation to implementation. It will use a certain number of mathematical functions to evaluate the nature of the risk, the percentage of chances to have a positive drilling, the average yield and the average specific yield predictable in the said context (Detay, 1987).

For the described session, a simplified report of the system's recommendations is:

H.Hd 310: The village is in a basement environment.
Hd 425: The size of the watershed considering the pluviometry seems sufficient.
Hd 565: The presence of a brook near the village must be considered.
Hd 365: The alteration thickness is satisfactory.
Hd 450: The nature and the importance of the fractures are favourable.
Hd 820: The water supply is indeed moderate.
Hd 850: The conductance values lead to predict a water logged bed in the hanging wall of the fractured zone.
Hd 235: The presence of a fractured zone must be considered.
Hd 655: Locate the drilling on the fractured zone, in declivous zone if possible and near the brook.
Hd 495: The drilling must also consider the decompressed zone of the crystalline basement.
Hd 960: Your percentage of success in relation to a statistical study on 1080 drillings in North Cameroon is estimated at 86%.
Hd 965: The average specific yield which is statistically predictable in relation to a statistical study on 1080 drillings in North Cameroon is estimated at $0.16\,m^3/h/m$.
Hd 970: The average statistical yield deduced in relation to a statistical study on 1080 drillings in North Cameroon and the conductance, is estimated at $2.9\,m^3/h$.

C. *The type examples of analogical solutions then are displayed. In the selected example:*

H. Type case Manawatchi—Département: Mayo Sava—Arrondissement: Mora.
H. Type case Doléré—Département: Bénoué—Arrondissement: Pitoa.

C. *The number of type examples is related to the importance of the analogical database, which can be undated with a simple text editor as the drilling works progress. Similarly for a generic solution, a message indicates that the solution is derived from an analysis by HYDROLAB of the described context only referring to rules without standard drillings.*

CONCLUSIONS

HYDROLAB is a practical system running on microcomputers that includes many features usually available only with expert-system shells on mainframes. Its architecture is modular and flexible and allows the substitution of packages of static knowledge, as the system is able to operate on various data belonging to different geographical areas or even to different countries. Because it has to run on a personal computer, the system is compact and does not include generic tools such as an object-oriented editor or a specific-rule language. We focused our efforts developing a powerful and reflexive control structure, on completely integrating domain-related databases into the expert-system inference mechanisms, and on building a natural language interface (including powerful explanation capabilities and backtracking strategies), and an efficient fuzzy matching mechanism.

The fuzzy reasoning achieved by HYDROLAB is improved by the dynamic task scheduling and by the double-pass model of control which is able to focus the system's attention on pertinent data. In the worst situations, where the matching fails for the many databases available, the system is able to elaborate an original solution based on abstract hydrogeological concepts; this is termed a generic solution, as it does not correspond to any observed real situation. Finally, the system includes the synthesis of a large amount of statistical processing for the North Cameroon area, and is able to forecast, according to numerical models and heuristical parameters, some of the drillhole characteristics, and to give an estimate of foreseeable successes.

In the near future, the spread of cheap powerful computers, based on the generation of the Intel 80386 or Motorola 68020 chips, will reduce the challenging hardware limitations encountered in this project and will contribute to the success of this expert-system approach to the water-supply problem in developing countries. This will lead to a considerable reduction in the cost of studies, thanks to the training and the increased involvement of African experts in the development of their countries' water resources. Artificial Intelligence therefore should be able to help solve the water problem in Africa, especially within the scope of programs for village water supplies.

Acknowledgments—We are grateful to Pierre Haren who always has encouraged us in our research. We also would like to thank the SMECI team at INRIA Sophia Antipolis, as many of the ideas presented here are the result of close collaboration with them. Thanks also are due to Yves Emsellem who supported the project and to Pierre Leymarie and Richard Sinding-Larsen, who introduced us to the subject.

A free copy of HYDROLAB® can be obtained by university members and students. If interested, write to the authors and include two floppy disks.

REFERENCES

Albert, P., 1985, Prolog et les objets: Proc. Fifth Intern. Workshop on Expert Systems and their Applications, Avignon, France, p. 332–350.

Arrivet, L., 1987, Introduction du flou dans un système expert: Mémoire d'Ingénieur de l'Ecole Supérieure d'Informatique d'Electronique et d'Automatique, Paris, France, 44 p.

Bernardi, A., and Detay, M., 1988, Corrélations entre les paramètres géoélectriques et les caractéristiques hydrodynamiques des forages en zone de socle: Revue du BRGM Hydrogéologie, no. 4, 15 p.

Borland International, 1986, Turbo Prolog, owner's handbook: Borland, Scotts Valley, California, 221 p.

Clayton, B., 1985, ART™ reference manual: Inference Corporation, Los Angeles, California, v. 1, 138 p., v. 2, 158 p., v. 3, 102 p.

Clocksin, W. F., and Mellish, C. S., 1984, Programming in PROLOG: Springer, Berlin, 304 p.

Colmerauer, A., 1977, Programmation en logique du premier ordre: Actes des journées La compréhension, IRIA, France, p. 124–132.

Colmerauer, A., 1983, Prolog in ten figures: Proc. Eighth Intern. Joint Conf. on Artificial Intelligence, p. 487–499.

Detay, M., 1987, Identification analytique et probabiliste des paramètres numériques et non-numériques et modélisation de la connaissance en hydrogéologie sub-sahélienne: Thèse de Doctorat d'Etat ès Sciences, Université de Nice, France, 456 p.

Detay, M., Goulnik, Y., Casanova, R., Ballestracci, R., and Emsellem, Y., 1986a, HYDROLAB an expert system for groundwater exploration in Africa: Bull. IWRA—Intern. Conf. Water Resources Needs and Planning in Drought Prone Areas, Khartoum, Sudan, 6 p.

Detay, M., Boulnik, Y., Casanova, R., and Ballestracci, R., 1986b, HYDROLAB un système expert pour la recherche des eaux souterraines en Afrique: Actes du deuxième colloque Int. Eau Gestion des données et aide à la Décision, Montpellier, France, p. 94–99.

Forgy, C. L., 1982, A fast algorithm for the many pattern —many object pattern match problem: A. I. Jour. no. 19, p. 17–37.

Haren, P., Neveu, B., Corby, O., and Montalban, M., 1985a, MEPAR: Un Moteur d'inférences pour la conception en ingenierie: 5 ème Congrès Reconnaissance des Formes et Intelligence Artificielle, Grenoble, France, p. 1273–1280.

Haren, P., Neveu, B., Giacometti, J. P., Montalban, M., and Corby, O., 1985b, SMECI: Cooperating Expert Systems for Civil Engineering Design: SIGART Newsletter (April 1985), no. 92, p. 67–69.

Hayes-Roth, B., 1984, BB1: An architecture for blackboard systems that control, explain and learn about their own behavior: Heuristic Programming Project, Report no. HPP-84-16, Stanford Univ., 22 p.

Knolidge, K., 1979, An inference net compiler for the PROSPECTOR rule based consultation system: Proc. of Sixth Intern. Joint Conf. Artificial Intelligence, p. 487–489.

Nii, P., 1986a, Blackboard systems: the blackboard model of problem solving and the evolution of blackboard architectures: A.I. magazine (July 1986), p. 38–53.

Nii, P., 1986b, Blackboard system, blackboard application system, blackboard systems from a knowledge engineering perspective: A.I. magazine (August 1986), p. 82–106.

Poyet, P., 1986, Discrimination des anomalies géochimiques multi-élémentaires significatives: Thèse de Doctorat d'Etat ès Sciences, Université de Nice, France, 436 p.

Poyet, P., 1987, La structure de contrôle dans les systèmes experts de simulation: Proc. 6ème Congrès Reconnaissance des Formes et Intelligence Artificielle de l'Association Française de Cybernétique Economique et Technique, Antibes, France, p. 723–738.

Poyet, P., and De La Cruz, P., 1987, Simulation navale en environnement hostile: Proc. Ecole d'Autonme sur l'intelligence Artificielle, THOMSON et Ecole Normale Supérieure d'Ulm, Jouy en Josas, France, 10 p.

Poyet, P., Haren, P., and De La Cruz, P., 1987, Un système expert de simulation navale: Proc. 6ème Congrès Reconnaissance des Formes et Intelligence Artificielle de l'Association Française de Cybernétique Economique et Technique, Antibes, France, p. 587–592.

Poyet, P., 1988, Cours de Modélisation Assistée par Ordinateur: Les problèmes, les langages, les machines et les techniques de l'I.A.: Laboratoire de Géologie et Géochimie, Université de Nice, France, 78 p.

Poyet, P., and De La Cruz, P., 1988, Une nouvelle classe de simulateurs destinée aux aides tactiques et aux systèmes d'armes: Proc. Eighth Intern. Workshop on Expert Systems and their Applications, Specialized Conferences A.I and Defense, Avignon, France, p. 89–99.

Poyet, P., and Detay, M., 1988a, Aide à l'implantation d'ouvrages d'hydraulique villageoise: Proc. Eight Intern. Workshop on Expert Systems and their Applications, Avignon, France, p. 397–410.

Poyet, P., and Detay, M., 1988b, HYDROLAB Version V1.4, Un système expert d'aide à l'implantation de forages en hydraulique villageoise, Manuel d'introduction et de référence: INRIA Research Report no. 936, Institut National de Recherche en Informatique et en Automatique, Sophia Antipolis, France, 42 p.

SMECI, 1988, Le manuel de l'utilisateur SMECI V1.3: ILOG S.A., France, 112 p.

Stefik, M., Bobrow, D. G., Mittal, S., and Conway, L., 1983, Knowledge programming in LOOPS: A.I. Magazine (Fall 1983) p. 3–13.

Stefik, M. J., Bobrow, D. G., and Kahn, K. M., 1986, Integrating access oriented programming into a multiparadigm environment: IEEE Software (January 1986), p. 10–18.

Warren, 1977, Implementing PROLOG: D.A.I. Research Rept. Vols. 1 and 2, Univ. Edinburgh.