An Expert System: a versatile tool for the real-time control of a solar building

J. NICOLAS, P. ANDRE
Fondation Universitaire Luxembourgeoise (F.U.L.), Arlon, Belgium

The F.U.L. building in Arlon, Belgium, includes a hybrid solar-assisted heat pump system linked to an underground storage facility [4]. The system was designed in 1977 and building was finished in 1982. Air conditioning of two buildings was foreseen with a total heat load of 900 GJ per year (*Figure 1*).

The main features of the heating system are :
- a total area of 382 m^2 of integrated solar collectors,
- a 500 m^3 water storage tank buried in the soil,
- an earth-storage volume with both horizontal and vertical exchangers,
- two heat pumps,
- a gas boiler as a backup.

The whole plant is supervised by an automatic control system ensuring regulation, management, work optimization and scientific monitoring. The control code, in PL.I language, is modular and has been designed so that it could be improved in the light of experience.

However, the heating system is a complex one, with various energy sources and many possibilities of heat distribution throughout the buildings. During the 8 years of operation therefore, the control strategy has been modified many times, based on the experience of the maintenance technician on the one hand, and for research purposes on the other.

Each software modification requires a rather long and cumbersome procedure: complete stop of the heating system, search for the subroutine and for the program line to be modified, compilation, linking and run test. In order to test a more flexible solution to those modifications, experimental research is now being conducted into an expert-system-based control system which could subsequently be made compatible with other heating plants [5], [6]. The F.U.L. building provides an ideal testing ground for the design and debugging of this control tool.

Suitability of an expert system

In general, when a computer is needed for the management of complex heating systems, many control strategies are tested. To implement such algorithms, standard procedural and sequential codes are often developed. In most cases, not only the algorithm but also the core of the software are specific and must be fundamentally recast for another building.

A computer tool like an expert system requires efficient hardware and sufficient memory resources. It is certainly not suited for dwellings with a single energy boiler and a simple distribution system, but it becomes profitable for a complex heating plant with, for example different energy sources, many distribution systems and a variable occupancy schedule.

This expert system (*Figure 2*) allows the input of management rules to be carried out in natural language (French for our implementation), with terms currently used by the maintenance staff. This may solve the problem of performing the process control, while taking economic optimization into account: for example, energy prices and their day-to-day trends can be easily incorporated into the rules.

The possibility of using heuristic knowledge in production rules allows the engineer to incorporate subjective assessments in

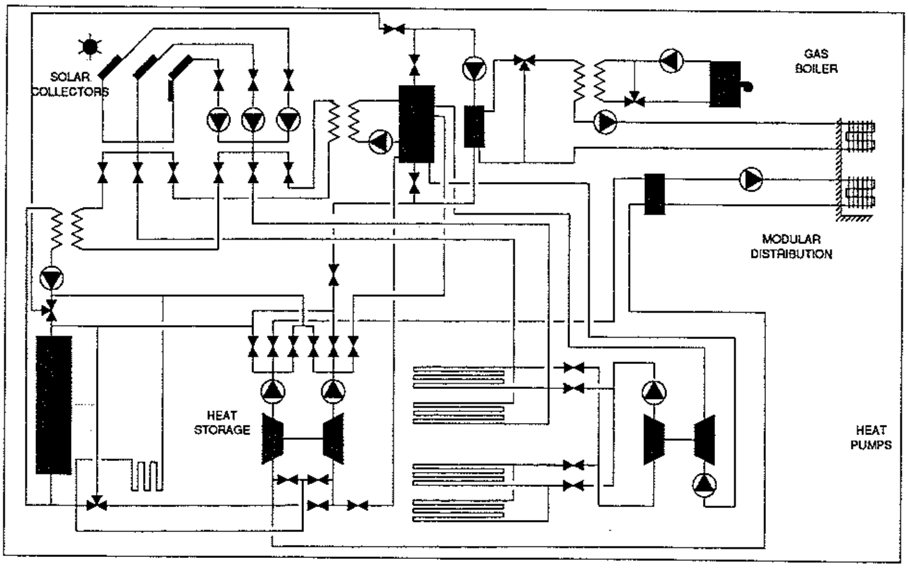

Fig. 1 : Simplified schematic representation of the heating plant.

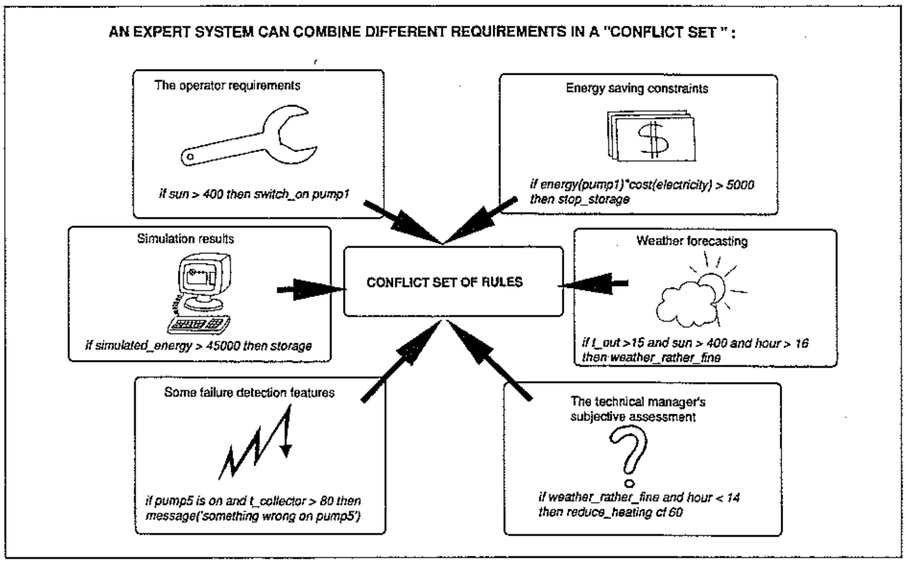

Fig. 2 : The conflict set in the expert system.

its management. The experience of the maintenance technician may also be taken into account. For example, he knows that if Pump 1 is switched on in the morning and if the weather is fair, it is likely that the temperature in Tank 2 will be sufficient to supply energy during the night. This reasoning may easily be added to the other rules in the database, which are sometimes more objective, thereby creating a "conflict set" from which the inference engine will extract the most probable solution.

The rules may be written almost at random, in bulk, within the limits of the syntax and any established precedences. New operating strategies are thus very easily tested, without changing the core of the computer code. For example, it is possible to decide in the morning to add a one-day rule, taking into account any unexpected event.

Finally, failure diagnosis, forecast, simulation, and so forth, may be considered simply by adding rules to the knowledge base.

State-of-the-art

Many computer codes are now being developed for use in the field of building design and management as energy optimisation tools. Some of these are expert systems to guide the user in the thermal design of the building [2], [3]. This approach is more suitable for two reasons. Firstly, the main options of the design may easily be changed, simply by rewriting some rules. Secondly, the expert system is able to give advice on insulation, window type and area, etc. on the basis of technical and heuristic knowledge of the problem, i.e. to search for an optimal design using simple if-then rules supplied by the user.

Very few commercial expert systems, however, exist for real time process control and still less for the management of a heating plant. Among all the applications of artificial intelligence in control and real-time operations, collected in abstract reviews and in updated bibliography [7], many deal with robotics but discuss expert systems and knowledge-based control far less. Moreover, most works relate to aeronautics whilst those relating to energy field are focused on nuclear power plants. We therefore chose to write our own expert system shell, for our specific purpose.

Restrictions imposed on the system

A real-time expert system, required to reach a decision and operate on real actuators such as pumps and valves, must be subject to some restrictions:

- In order to avoid the repetition of rules for each actuator or each sensor, general rules with variables must be drawn up. For example, instead of writing for each valve v_i :

  if opening v_i then wait-for limit-stop v_i

  it would be better to write the general rule :

  if opening V then wait-for limit-stop V.
  for any valve V.

  This constraint imposes at least the choice of first order predicate logic instead of a simple propositional logic (or zero-order logic) since the latter deals only with the simplest notions of truth and falsehood, including the processing of "and", "or", "not" operators. Predicate logic broadens propositional logic by adding the concept of objects, or items and the ability to unify a variable, such as V, to a given object, say v_i.

- Like all expert systems devoted to planning or design, a real-time expert system devoted to process control must use a forward-chaining inference engine. Indeed, it is not possible in our case to use a goal oriented search. This chaining mode, called backward chaining, and used in diagnostics, sets hypothesis and searches in the database for known facts which would enable this hypothesis to be confirmed.

In process control, it is impossible to set a hypothesis and the inference engine must progress from known facts towards a "terminal action", i.e. a fact which is not used in another rule premise, for example a pump switching on. At this stage, the system stops the consultation mechanism for this inference and once more starts to explore the database for a new one.

- These so-called "terminal actions", or "actions" in general, are special facts, added to the database, but leading in addition to actual actions, having a real effect on the environment. The inference engine must be able to distinguish between such actions (for example : opening a valve, switching on a burner, writing a message on the screen ...) and simple new facts (such as : heating_mode_active or energy = 1 000). Besides, when the system backtracks and if the chosen path through rules leads to a dead-end, actions already carried out (e.g. : opening a valve) must be cancelled by the back actions (closing the valve).
- If heuristics are allowed in order to take into account the maintenance technician's assessment, uncertainty must be handled by means of certainty factors for rules and facts.
- Finally, one of the most important features of such an expert system is the ability to handle time and to wait for events.

Basic options

In the field of artificial intelligence, most codes are written by computer specialists. In our case, we thought that it was interesting to explore the heating engineers' approach. We therefore chose to write a new shell, for this specific purpose, starting with a suitable computer and language. Although this approach might appear a little unorthodox to a knowledge engineer experienced in artificial intelligence, it might bring some original ideas to light.

Among the set of requirements described above, many of them could already be fulfilled by choosing PROLOG as the development language. Prolog fits into the first-order logic, needed for our problem. Its intrinsic mechanism corresponds exactly to the process of an and/or graph path, typical of that used by an inference engine. Prolog belongs to the object-oriented family of languages, in that the problem is described in terms of facts and relations between facts. It allows an easy implementation of hierarchical knowledge representation with an object-attribute-value scheme, suited to a heating plant layout (e.g. : object = pipe, attributes = temperature, flowrate, ..., values = 20° C, 10 m^3/H, ...).

Finally, backtracking is included as a standard feature of Prolog : if a dead-end is reached, the language backtracks itself, or re-explores the current rule occurence (e.g. a variable assignment in the premise part), discards it, and searches for an alternative one. The system will continue backtracking and supplying new solutions until there are none left. This feature is obviously useful when designing an inference engine.

Quintus Prolog [8], from Quintus Computer System Inc (USA) and distributed by Artificial Intelligence Ltd (United Kingdom) has been chosen, especially for its powerful development environment and its ability to interface with foreign languages, such as C, Pascal or Fortran. The machine is an Apollo-3000 workstation, with 4 Mbytes core-memory and a 70 Mbytes hard disk. For our particular application, the interface with the process is achieved through the serial line and is driven by a Pascal code.

The graphical part is written in GMR (Graphics Metafile Resource), a resource from Apollo.

All these system-dependent features, however, are well separated from the core of the expert system and, on the whole, we endeavour to ensure that the software is portable :

- Quintus Prolog runs on standard hardware and operating systems, either mainframe, technical workstations or 386 personal computers;
- UNIX operating system was chosen for its portability;
- input and output data flows transit in a "mail box" file, in a standard format, shared by the expert system and the process interface;
- foreign routines (graphics, window management on the screen, time handling, ...) are collected in separate libraries and are called by the Prolog code.

System organization

The expert system is organized in three levels (*Figure 3*) :

- An inner core of pure software, including the inference engine itself and its own codes (drivers, interfaces, ...), inaccessible to the final user.
- An intermediary shell, including typical functions of heat engineering, such as power computation from a temperature difference and a flow rate, energy integration, estimation of performance coefficient, exchanger efficiency computation, ... This shell also includes some economic functions, such as the calculation of energy cost on the basis of fuel prices (with a possible discount during night periods) and consumption data.

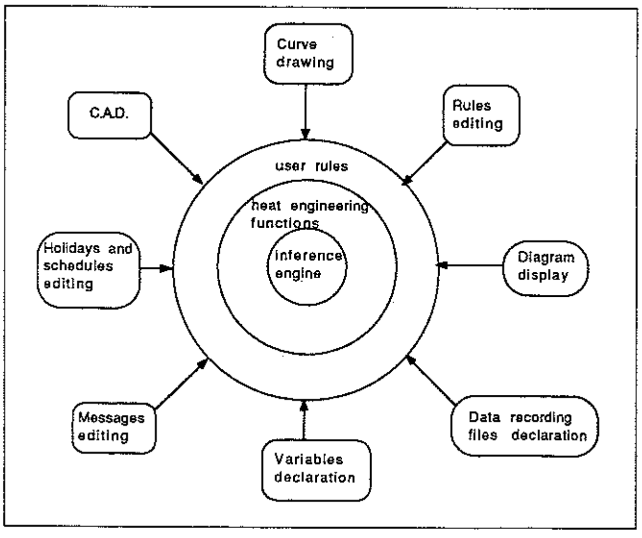

Fig. 3 : The expert system organisation.

All these functions may be called up by the user in the rules.

- An outer shell, devoted to the specific application. It includes the user rules written in natural language and it is the only part of the system concerning the user, the rest being transparent.

Operating principles

Managing time

One of the most important features of an expert system for real-time control is obviously the ability to handle time and to wait for events. Time is managed through a *"lifetime"* concept.

A lifetime, expressed in seconds, may be attributed to every item handled by the expert system (variable, mode of operation, recording file, ...). The user may write a rule that tests whether this item is "still alive", for example, in order to trigger a mode of operation for a given period. A typical application of this lifetime concept, in a set of rules for failure detection, would be the comparison of the valve-opening lifetime and the waiting time before it reaches its limit stop.

The system also tests "death" of variables, files through rules at the end of each consultation process and tests "death" of facts whenever those facts are used in rule premises. But the rules themselves may use the reserved words "birth", "death" and "lifetime" in their body.

The system presents a special behaviour when the words "wait_for(event)" (where "event" is any possible event) occur in the conclusion part of a rule. This case is considered each time an event must be waited for, in the real-time process, before continuing. At this level, the sequence freezes the rule and will continue the depth-first search mechanism only when the event occurs. In the meantime, another inference sequence may begin.

A rule-oriented system

The rules are written in French, but for convenience, everything has been translated into English in this paper. In order to guarantee the versatility of the tool, much of the system operation is left to user rules, at the expense of simplicity. For example, data recording is managed by a rule like this :

if file1 is dead then record file1.

where file1 is the name of the recording file,
or by the general rule

if FILENAME is dead then
record FILENAME
for every filename.

Also the cancellation of some actions when the inference engine backtracks, as mentioned above, is provided by the rules rather than by the system itself. For example, if a valve has been opened and if a terminal action (say a pump switching on) fails due to some defect (say a thermal trip) then the back action (valve closing) when this defect occurs, must have been provided for in the user rules.

Application procedure

The application procedure for a final user is divided into several stages :

1. Formulation of the control rules for the particular plant and data gathering.
2. Editing of the variables of the heating plant - includes the evaluation of each attribute, such as lifetime and extreme values.
3. Drawing of the heating plant diagram - includes the linking of each segment with temperature and flowrate symbols.
4. Definition of the data acquisition files, with the name of the variables to record and the recording frequency.
5. Rule checking.
6. Definition of the initialisation file which is called when the consultation mechanism starts and which is the compilation of all the actions that the system must perform to reset the heating plant (closing all valves, switching off all pumps, ...).

7. Starting the run.
All these stages are selected in a menu displayed permanently on the screen.

Implementation

Managing the solar building

The number of rules that must be written to perform at least the equivalent control of the old system on our solar building is about 150. However, the power and the versatility of such an expert system create many new possibilities. A detailed description or list of the rules already written for the F.U.L. building application, without having a detailed description of the heating plant, is however not feasible here.

Application for environmental control

The Fondation Universitaire Luxembourgeoise specialises in environmental monitoring and control, a field in which such an expert system may be applied whenever complex control is considered. In our case, it is planned to use the expert system as a tool for the management of a waste water treatment plant : the core of the software and the general ideas remain unchanged, only the library in the indermediate shell must be filled out with specific functions in order to allow the manager of the waste-water treatment plant to write his own rules.

The expert system concept is well suited to environmental control. Indeed, it matches the features of such management exactly : vagueness and diversity of the available information, yet the need to take a rapid decision without ambiguity.

Conclusion

Evaluation of the system

Expert systems can encapsulate abstract and experimental knowledge, which is characteristic of this kind of complex control. Such an approach frees the engineer from the need to know the details of the software and of the process interface and allows him to concentrate on control. He also has the ability to make a rapid evaluation of all possible control algorithms. This would improve the efficiency of the controlled system and avoid time-consuming and tedious program-compilations and debuggings. Care should be taken however to write all the necessary rules effectively and to consider all possible situations. Such a complete set of rules, even for quite a simple plant, may become rapidly cumbersome and inextricable.

In order to use this tool with maximum efficiency, one must bear in mind important limitations of the expert system concept. Firstly, it must be considered only as a tool which can improve the control flexibility by means of a user-friendly rules editor. But the performance will still depend on the algorithms used in the rules rather than on the tool itself. More specifically, an "optimal" solution is never achieved with an expert system, but the "less bad" solution is reached by a trial and error process. Secondly, the expert system must act essentially as a supervisor and a scheduler for all tasks, but it is not well suited for standard regulation, such as a local PID controller.

Possible developments

The research performed provides a basic framework for future developments in the field of smart process control. Possible developments are set out below :

- A complete hierarchical approach to knowledge representation is needed, in order to clarify the rule base. This will require consideration of the system in terms of subsystems and components and of their interrelationships. For such an approach, a more developed theory for handling uncertainty may be used. The expert system may, for example, incorporate the fuzzy logic reasoning methodology, already tested successfully in other knowledge-based expert systems [1].

- Simulation of the plant should play an important role in the choice of a control strategy. More work should be done to investigate whether the expert system can be interfaced with standard simulation packages, such as TRNSYS [9] in the heat engineering field.
- User interface is an important consideration : for a real-time expert system, no debug is allowed, and the graphical output of the variables' evolution with time remain the only link between the user and the system core. The user interface in the present implementation is rudimentary and should be improved.
- Auto-learning features, in the form of meta-rules, should also be interesting in order to allow the system to improve itself thanks to past experience.

References

1. Chang, T.C. and IBBS, C.W., "PRIORITY RANKING - A Fuzzy Expert System for Priority Decision Making in Building Construction Resource Scheduling", Building and Environment, 25, n°3, 253-267 (1990).
2. Doheny, J.G. and Monaghan, P.F., "IDABES : an expert system for the preliminary stages of conceptual design of building energy systems", Artifical intelligence in engineering, 2, n°2, 54-64 (1987).
3. Faist, A., Hagen, F. and Morel, N., "An expert system for passive and low energy building design.", ISES Solar World Congress 1989 - Kobe (Japan), Sept 4-8 (1989).
4. Nicolas, J., and J.P. Poncelet, "Solar-assisted heat pump system and in-ground energy storage in a school building", Solar Energy, 40, 117-125 (1988).
5. Nicolas, J., P. André, J.F. Rivez and V. Debbaut, "An expert system for the real-time control of a building heating plant", Proceedings 3th SSB conference, Liège, Belgium (1990).
6. Nicolas, J., P. André, J.F. Rivez and V. Debbaut, "Use of an expert system for the real time control of a solar building", Proceedings ISES'91 Congress, Denver, U.S.A., part I, pp 1726-1731 (1991).
7. N.T.I.S., "Published search in artificial intelligence in control and real-time operations", U.S. Department of Commerce, Springfield (U.S.A.) (1990).
8. QUINTUS PROLOG - USER MANUAL, Artificial Intelligence Limited - Watford (England) (1989).
9. TRNSYS, "TRNSYS, a transient simulation program", Solar Energy Laboratory - Univ. of Wisconsin, Madison U.S.A.-v.13.1-sep.90.

The authors

- Jacques NICOLAS, F.U.L. (Fondation Universitaire Luxembourgeoise), avenue de Longwy 185, B-6700 Arlon, Belgium.
- Philippe ANDRE, F.U.L., address as above.