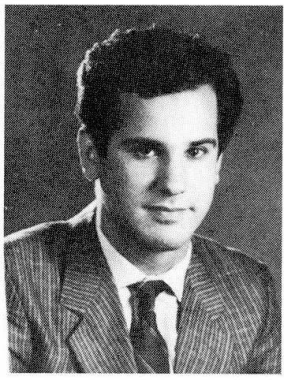
Knowledge-based design of steel portal frames for agriculture buildings

Knowledge-Based Design of Steel Portal Frames for Agricultural Buildings

Cadres métalliques pour bâtiments agricoles dimensionnés à l'aide d'une base de connaissance

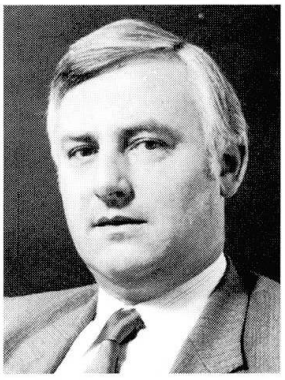
Wissensbasierende Projektierung von Stahlrahmen für landwirtschaftliche Gebäude

João BENTO
Civil Engineer
Technical Univ. of Lisbon
Lisbon, Portugal



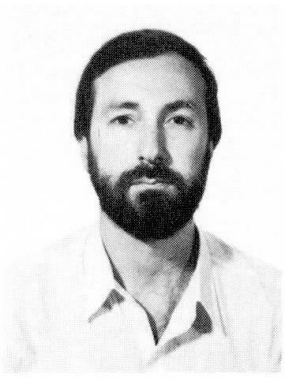
João Bento received his Engineering degree 1983 and MSc in Structural Engineering, 1987 at the Technical University of Lisbon were he is Lecturer. His research interests are steel/ composite structures and, more recently, expert systems for structural design. He is also Visiting Lecturer in the Expert Systems Lab Imperial College.

Bruno FEIJÓ
Aeronautics Engineer
PUC/RJ
Rio de Janeiro, Brazil



Bruno Fejió received his Engineering degree at ITA, São Paulo, 1975 and his PhD at Imperial College, 1988. For many years he worked with computers in engineering design. He is now Lecturer in Computer Science in PUC/RJ. He is also Visiting Lecturer in the Expert Systems Lab. Imperial College.

Patrick J. DOWLING
Prof. of Civil Engineering
Imperial College
London, UK



Patrick Dowling is a member of Britain's National Academy of Engineering, the Fellowship of Engineering. He is Chairman of the Drafting Panel of Eurocode 3 and Editor of the Journal of Construction Steel Research and Director of the Consulting firm Chapman & Dowling Associates Ltd which drafted the Brazilian Steel Bridge Code.

SUMMARY

A simple conceptual model of the structural design process has been proposed by Feijó (1988). This paper deals with the improvement and implementation of that model. A first implementation, however, demands a domain of application that is both simple and specific. These criteria are certainly met in the design of steel portal frames for agricultural buildings, especially through the rather heuristic style of design which has evolved. A new routine for knowledge acquisition in design problems is also proposed and some aspects of the implementation of the model are discussed.

RESUME

Les bases d'un modèle conceptuel d'une procédure de dimensionnement des structures ont été proposées par Feijó (1988). Cet article concerne le développement et la mise en oeuvre de ce modèle. Une première mise en ouvre requiert un domaine d'application à la fois simple et spécifique. Tel est le cas du calcul des cadres métalliques pour bâtiments agricoles, grâce surtout aux caractéristiques heuristiques de ces projets. Une nouvelle méthode pour l'aquisition de la connaissance est proposée et quelques aspects plus intéressants de la mise en oeuvre du modèle sont discutés.

ZUSAMMENFASSUNG

Feijó (1988) hat ein einfaches Begriffsmodell für die Berechnung von Strukturen vorgeschlagen. Dieser Artikel behandelt die Entwicklung und Verwirklichung jenes Modells.
Diese erfordert jedoch ein gleichzeitig einfaches wie auch spezifisches Anwendungsgebiet.
Die von Grund aus empirischen Stahlrahmen für landwirtschaftliche Gebäude entsprechen sicherlich diesen Kriterien. Eine neue Methode für dem Wissensgewinn bei der Berechnung wird vorgeschlagen und einige Aspekte der Verwirklichung des Modells werden kommentiert.

1. INTRODUCTION

This paper concerns the development of a model of the design process drawn on an idea proposed by Feijó [1]. The domain of steel portal frames for agricultural buildings is adopted because it is both simple and specific.

First, a definition of design is presented followed by the assumption that most of the design activity is computation. Secondly, the model SAE of the design process is presented. Thirdly, some aspects of the implementation of this model are discussed. Finally, some results of the investigation into the domain of application and of the knowledge elicitation process are presented.

2. A DEFINITION OF DESIGN

The definition of design presented in this paper is an adaptation of the ideas proposed by Vinod Goel and Peter Pirolli [2]. Design is then defined by

D(Space, Inv, Struct)

where Space is a design problem space (2.1), Inv is a set of invariants of design (2.2) and Struct is a structure of the design problem space (2.3).

2.1 Design problem space

Research in design methodology has revealed that designers create a central case (a prototypical case) C_0 in the early stages of the process. Moreover, the designer is faithful to the central case during the whole process. In this context, design is an evolutionary process that refines the central case towards the artifact specification S(E,A,R), i.e.

$$I \rightarrow C_0 \rightarrow C_1 \rightarrow \cdots \rightarrow S(E,A,R)$$

where I = input specifications, E = entities, A = attributes of the entities and R = relationships between entities.

The sequence of states $\{C_i\}$ represents one of the many routes through a design problem space. This draws on the concept of plans of Newell and Simon [3], who pioneered the computational modelling of mental processes.

The states C_i are unpredictable but motivated variations of the central case C_0. This is equivalent to the radial category with prototype effects proposed by Goel and Pirolli [2]. Also, this evolutionary aspect of the process can be found in other models of design [4].

The sequence of states C_i may go into the design problem space, get lost and never emerge with the artifact specification. Constraints are needed to keep the search (for a possible route) to a manageable size and to help navigation in the problem space. In other words, design should be constraints driven.

The design problem space is supposed to have a set of design characteristics which explain movements in the problem space and support the evolution of the states C_i. A simplified version of this set, adapted from Goel and Pirolli [2], is as follows:

d1: Extensive Problem Structuring (i.e. finding missing information). The following steps are used by designers: Studying the design briefly; Soliciting information and clarification; Applying constraints: legislative (e.g. design codes), technical, pragmatic (e.g. money); Applying personal knowledge; Negotiating constraints (to fit personal ideas);

d2: Extensive Performance Modelling (simulation also included);

d3: Personalized Evaluation Functions (stopping rules also included);

d4: General Evaluation (i.e. a generated or focused component is continuously evaluated in three contexts: local context; current context of the complete artifact; future context - i.e. projecting the complete artifact in its final stage);

d5: Dynamic Commitments (i.e. making, negotiating and propagating commitments);

d6: Solution Decomposition (moreover, modules are not isolated, but interconnected at a function level);

d7: Abstraction Hierarchies (i.e. working with modules and entities at various levels of detail. Descending too soon or not descending at all this hierarchy is a common mistake of novice designers.);

d8: Use of Symbol Systems (e.g. bubble diagrams, rough sketches and natural language);

Design characteristics represent cognitive needs which should be satisfied by any CAD system. In this context, hypertext tools might be offered to help problem structuring (d1), object oriented languages might help the implementation of d4, d5, d6 and d7, and intelligent user interface techniques might help the use of symbol systems (d8).

2.2 Invariants of design

Goel and Pirolli [2] propose the differentiation between design problem solving from non design problem solving by identifying major invariants in the design process. The authors adopt the following list of invariants:

inv1: Large and Complex problems;

inv2: Input as goals and intentions (statements) and output as an artifact specification S(E,A,R);

inv3: Temporal separation of design phase and delivery of the artifact (usually long);

inv4: Delayed or limited feedback from the world (in the sense that, after delivery, reactions from the world can hardly guide the designer in the current project – only in the next similar project);

inv5: Independent functioning of the artifact, i.e. the designer cannot be there to explain its significance or perform its function (this is the case of a bridge designer and the project of a bridge);

inv6: Costs (penalties or benefits) associated with every action;

inv7: No right/wrong answers, only better/worse ones;

inv8: Many degrees of freedom in design problem statements (i.e. many open questions).

In this context, music composition is a better example of design activity than spontaneous conversation. In the later, there is no separation between design phase and delivery (they are simultaneous), the problem solver actually constructs the artifact rather than specifying it, there is no delay feedback and, finally, costs are not associated with actions.

2.3 Structure of design problem space

The structure of the design space is a relation that assigns invariants to the characteristics of the design problem space, i.e.

$$\begin{aligned} \text{inv8} &\rightarrow \text{d1} \\ (\text{inv3, inv4, inv5, inv6}) &\rightarrow \text{d2} \\ \text{inv7} &\rightarrow \text{d3} \\ \text{inv3} &\rightarrow \text{d4} \\ (\text{inv3, inv2}) &\rightarrow \text{d5} \\ \text{inv1} &\rightarrow \text{d6} \\ (\text{inv1, inv2}) &\rightarrow \text{d7} \\ \text{inv1} &\rightarrow \text{d8} \end{aligned}$$

The association of characteristics with particular movements in the problem space and the interconnection between characteristics are not considered part of the definition of structure. These are specific aspects of a model of the design process.

3. DESIGN AS COMPUTATION

Design is a cognitive process involving deduction, induction, experience, creation and intuition. These are complex ways of reasoning that are difficult to treat with a mathematical formalism. However, the authors believe that some problems of cognition can be formulated with some precision in terms of the mathematical theory of computability [5]. On the other hand, they recognize the limitations of this approach because there are aspects of mental life that cannot be

represented mathematically. In this particular, the reader is addressed to the classic works of the critics of artificial intelligence, such as Hubert Dreyfus [6], John Searle [7] and Terry Winograd [8].

There are various computational methods for considering the ways of reasoning mentioned above. Formal proofs in first-order logic capture an important aspect of deduction. Also some forms of induction (where induction is a systematic way of reasoning that increases the given information) can be easily adopted, e.g.

High_rise_building(a)
Steel_framework(a)
High_rise_building(b)
Steel_framework(b)
...

might induce

For all x, High_rise_building(x) ⇒ Steel_framework(x)

Other forms of induction are also possible [9][10]. Furthermore, learning techniques can handle some aspects of design experience. However, very few methods can be proposed for creation (e.g. analogy across different domains). Creation and intuition are still in the area of competence of human designers.

Artificial reasoning is associated with the problem of knowledge representation. As far as design is concerned, every characteristic of the problem space is associated with one or more types of knowledge (knowledge about design codes, personal knowledge, knowledge about evaluation, knowledge about abstractions, and others).

A taxonomy of personal knowledge is proposed by Goel and Pirolli [2]. They identify three kinds of personal knowledge: procedural knowledge (which cannot be made explicit - i.e. it is not open to introspection); abstract conceptual knowledge (principles, laws and heuristics which are situation specific knowledge; knowledge of patterns (forms or patterns directly evoked by a designer). Furthermore, Goel and Pirolli [2] claim that personal knowledge is organized in two levels: a general level and a domain specific level. The second represents a more organized and complete knowledge than the first and it is built on top of it. The authors believe that most of these types of knowledge can be satisfactorily represented by first-order logic and frames.

4. THE MODEL SAE

In the model SAE, a state C_i is represented by a set of propositions in first order logic, such as:

Tension_member(b1)
There exists x, supports(x,b1).

The evolution of the states C_i is carried out by a recursive process of three subprocesses *Synthesis - Analysis - Evaluation* which are themselves recursive processes. At least one process S-A-E is required to move from one state to another. Furthermore, these processes are organized in levels of abstraction. The central case is generated by a higher level process and detailing is a low level process. In the case of structural engineering design, the following processes can be identified: Higher level processes (e.g., conception); Preliminary design; Structural analysis; Detailing. The model SAE is shown pictorially in Figure 1.

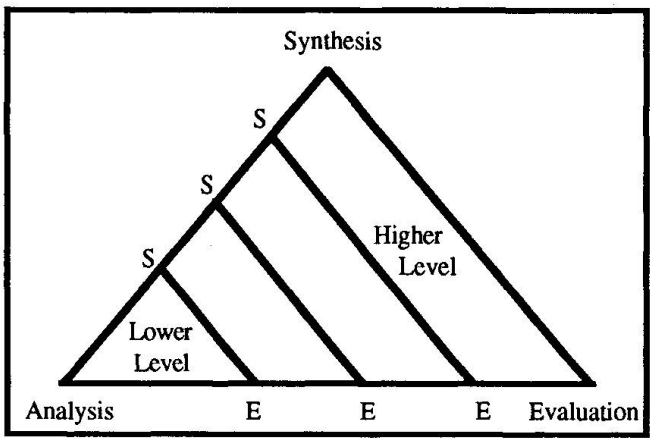


Figure 1

Design in the model SAE is recursively specified by the following sentences in first-order logic:

design (X, end).
design (X, Level)
if **synthesis** (X, Level) and
analysis (X, Level) and
evaluation (X, Level, New-Level) and
design (X, New-Level).

A subprocess of synthesis, analysis or evaluation represents a specific form of knowledge (i.e. how to synthesize, to analyze or to evaluate). In turn, this knowledge can evoke other subsets of knowledge associated with this type of

subprocess. Furthermore, that specific form of knowledge can call a new design process for a new entity or module (at a specific level of abstraction).

Synthesis (the "so what ?" process in design) is the crucial problem because it involves creation. The degree of artificial intelligence of a CAD system is proportional to the degree of independence that each process of synthesis has from human intervention. Interactive computer graphics and intelligent user interface techniques can be understood as the link between the incomplete artificial synthesis (null in conventional CAD systems) and the external human synthesis (Fig. 2). The peculiarities of synthesis activities will be focused further on when the problems involved in knowledge elicitation in design problems will be discussed.

5. PRINCIPLES FOR AN ARCHITECTURE OF THE MODEL SAE

The proposed architecture starts adopting the framework of multi-expert systems. This framework consists of a set of Artificial Design Assistants (each assistant corresponding to a specific domain of expertise) and a strategy of cooperation. The assistants are related with subprocesses, specific sub-domains and levels in the model SAE. These assistants are supposed to provide design scripts, advices and answers to the user or another assistant. The objective of the cooperation is to reach an artifact specification S(E,A,R) with no inconsistency. In the present work, the strategy of cooperation is not presented.

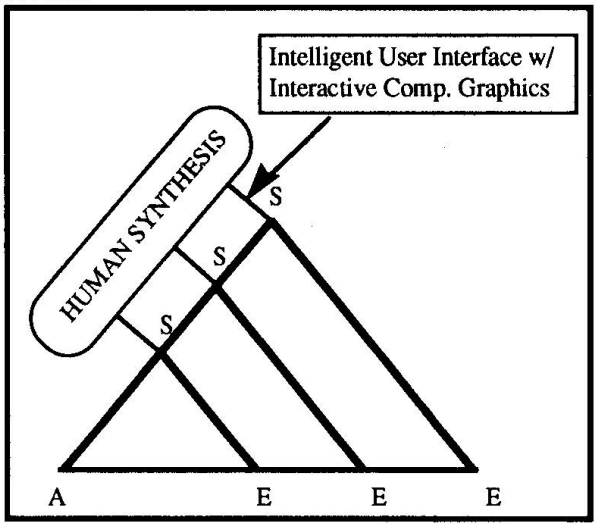


Figure 2

The proposed architecture is integrated with a set of conventional CAEngineering systems, such as CADraughting and Finite Element packages, as illustrated in Figure 3. In this particular, an effective interface between first-order logic and procedural programming should be developed [11].

The Database of Design Facts (DDF) represents the current state C_i and is classified into three areas as follows: Design Requirements (these facts represent a design request which includes hard constraints and basic assumptions); Design Options (these facts are soft constraints which represent a design script); Data (these facts are temporary data retrieved from the conventional database or generated by a subprocess). Synthesis processes may change the classification of any particular design fact.

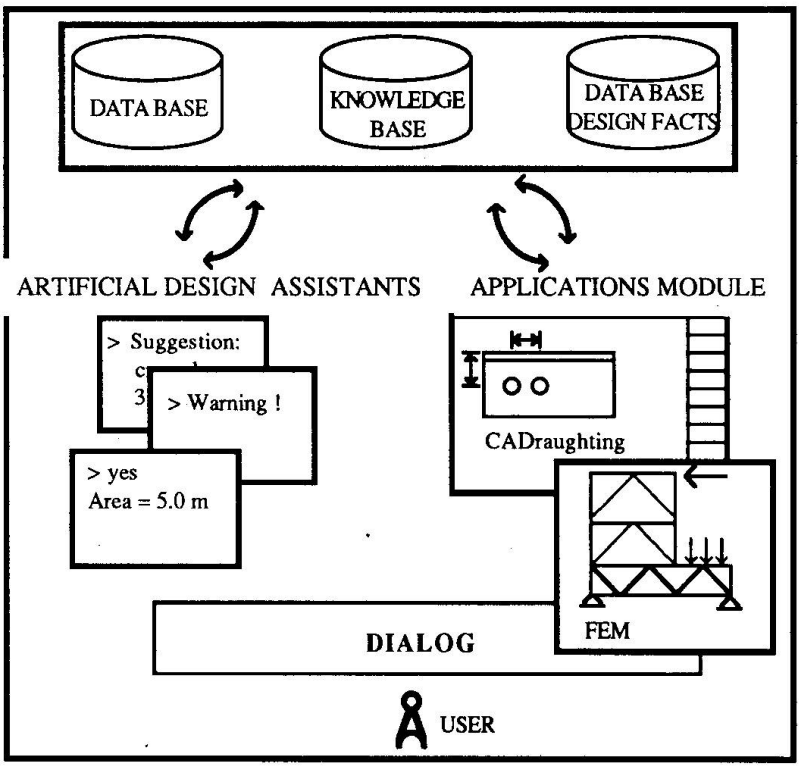


Figure 3

5.1 The role of facts

The evolution of the recursive process in the model SAE is closely aligned to the changes in the Database of Design Facts (DDF). Analysis processes can be triggered by changes in the DDF provided by synthesis processes. Evaluation processes can be triggered by new design facts provided by analysis processes (moreover, if any analysis process fails, the reason for failure will also be present in DDF).

Synthesis processes can be triggered by facts which evaluation processes identify as causing deviations.

This evolution can be exemplified by the following simple example, in which structural analysis is assumed to be the current design level:

> – Under a sub-process of analysis, a steel portal frame is beeing analyzed by means of the use of a FE package; the output of the FE analysis will generate new design facts, e.g. displacements; evaluation will then be triggered by the arrival of these new design facts; during evaluation the displacements may be found to have exceeded the displacement limits defined in the knowledge base; this occurrence will then trigger a synthesis process to allow the system to make a decision about the deviation identified; assuming the synthesis process provided a solution for this problem by redefining the depth of the haunches in the beams of the portal frame, then a change in the DDF would occur (e.g., *hauch_depth(frame, 1.5)* would become *hauch_depth(frame, 2.3)*); this would trigger a new sub-process of analysis...

In the present model facts are, as described, of great importance in driving the evolution during the process of design, towards the delivery of the design itself. This draws the authors' attention for the way of dealing with facts at implementation (6.2) and at knowledge acquisition (7.3).

6. IMPLEMENTATION OF THE MODEL SAE

6.1 Domain of application

Extracting, structuring and formally representing experts personal knowledge is extremely difficult. Furthermore, the definition which is being presented for the representation of the structure of the design problem space is formal and theoretical.

It is mainly due to this reasons that a specific and simple domain of application had to be selected for a first implementation of the model SAE. Steel portal frames for agricultural buildings, being simple but peculiar structures which are designed rather heuristically, seemed to be an attractive solution meeting those requirements.

6.2 Programming requirements

From a programming point of view, the more relevant aspects under consideration in the implementation of the proposed model are related to the need for an integrated use of several cooperating expert systems (Artificial Design Assistants – ADAs), conventional procedural applications (Applications Module – AM) and databases.

First order logic and frames are the formalisms upon which each of the ADAs is being built. It is, hence, a logic programming style which is being used to build them. It has been stated that the strategy for cooperation between the ADAs will not be presented in this article.

Concerning the procedural applications involved in the system, the main problem being tackled is their effective integration with the logic systems (the ADAs) both in terms of its mutual invocation and in terms of data interchange. A number of effective localized solutions for these problems have been proposed and already implemented [11], and the need for the development of a global logic $\leftrightarrow$ procedural programming interface was identified.

As concerns the presence of databases providing support to the applications module (CAD, FEM, etc.) and the ADAs, special attention has to be taken to the possible ability for logic systems to handle data – namely ground clauses (facts) – stored in secondary memory (e.g. databases) in the line of the contributions from Cotta [12]. This may be found to be of great importance for the improvement of the efficiency of the overall system; namely, in terms of response time for inferences involving different expert systems and procedural programs as well as in terms of the possible physical size of the knowledge bases.

7. KNOWLEDGE ACQUISITION

In spite of the simplicity and specificity of the chosen domain, the current project had a bottleneck in its knowledge elicitation component associated with the implementation of the Automatic Design Assistants (the expert systems). This is a statement common to most expert systems development reports. In fact, in the current situation, the traditional scarce availability of knowledge engineers is worsened by the relatively low number of available experts involved in the design of agricultural buildings. Furthermore, a pure design problem is being considered as opposed to typical and well covered non design problems such as diagnosis ones.

The knowledge elicitation methods more widely used may be separated into two main groups:

- psychological techniques [13];
- induction [14].

The first group comprises some of the most popular and better tested methods, such as:

- interviews;
- protocol analysis;
- multidimensional scaling (e.g., Kelly grids);
- concept sorting.

Although some of this methods evolved into very well structured routines, and even into computer programs to be directly used by the expert(s) (such is the case of Kelly grids, for instances) some very pragmatic authors as Parsay and Chignell [15] claim that "each knowledge acquisition method developed in the late 1970s and early 1980s has been a variant on the general theme of *talking to the expert*".

Induction is, the authors believe, reasonably inadequate as a major knowledge acquisition technique in a structural design problem, although the system should contain some rule induction capabilities (see 3.). Rule induction defenders seem to agree [14]. This idea can be partially supported by some of the adopted invariants of design (2.1) – inv1, inv2 and inv8.

7.2 A new kind of knowledge engineers

In the psychological techniques group, interviewing and protocol analysis seem to form the core of the so called "team approach". In this situation a team formed of knowledge engineers and domain experts work together to produce (and validate) a knowledge base (or the expert system itself). Under such a working environment a number of successful and well tested routines and methodologies have been used to elicit and organize experts' knowledge mostly in diagnosis problems. Some outstanding products emerged of such an approach. This is the case of PROSPECTOR a geology consultant for hard rock mineral exploration which has already predicted the occurrence of very important molybdenum ore deposits, Duda et al. [16] and XCON (formerly R1), an assistant for computer systems configuration which actually replaced humans in its domain of expertise, McDermott [17].

However, acquiring knowledge in a design project needs a special approach as synthesis plays a major role in design problems (5.1), the knowledge involved in synthesis activities being extremely hard to elicit. The problem with knowledge about synthesis is that experts are very often unaware of the structure of their own knowledge. As the decisions they made seem so obvious to them, they are not able to isolate and identify something close to "design rules" which can be programmed in a knowledge based system. It is the role of the knowledge engineering team and methodology to make the acquisition of that knowledge possible.

Due to the mentioned difficulties, the authors believe that, especially in design problems, the "team approach" should evolve into a situation in which pure domain experts need to be slightly educated on expert systems technology and pure knowledge engineers (traditionally, computer scientists) should give place to domain-almost-experts with a strong background of cognitive science, design theory, knowledge engineering and expert systems technology. This is the situation of the current project, in which structural engineers deeply involved in expert systems research, have been interviewing and observing structural engineering experts involved in the design of steel portal frames for agricultural buildings. This leads to a better efficiency in terms of elicitation of the experts' personal knowledge giving deeper access to abstract conceptual knowledge and knowledge of patterns (3.).

7.3 A knowledge elicitation methodology driven by facts

The knowledge elicitation methodology developed and used in the current project was originally based in the work by Grover [18], who first proposed the establishment of a Knowledge Acquisition Documentation Series. Such formalized documentation, by providing strong guidance and self monitoring during the all knowledge acquisition process, would be a first preventive step against the usual intuitive interpretation of psychological techniques such as interviewing or protocol analysis.

Grover's proposal for a knowledge acquisition (KA) cycle identifies three separate stages, as pictorially described in Figure 3 (reference [18] has a complete description of this cycle):

- Domain definition;
- Fundamental knowledge formulation;
- Basal knowledge consolidation.;

Considering the specific case of knowledge acquisition in a structural design context, a modified version of Grover's KA cycle is presented (Figure 5).

7.3.1 Domain definition

Being the knowledge acquisition team formed as described above (7.2), most of the Problem Definition items of Grover's routine become either unnecessary or of less importance, namely those concerning bibliography and glossary.

On the other hand, keeping in mind that a design system is being built it is of vital importance, during this stage, for the expert(s) to become briefly acquainted with the model of design being implemented.

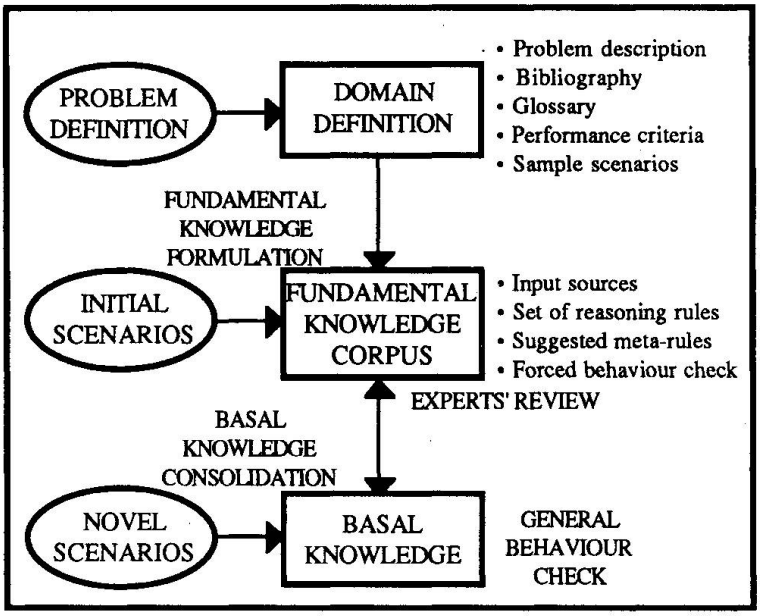


Figure 4

Also at this level of the knowledge acquisition process, the expert(s) should provide a list of questions to be answered before each designs starts, as discussed below (7.3.2).

As opposed to Grover's suggestion, the authors believe that performance criteria should not be explicitly discussed with the expert(s) at this early stage, to avoid unreasonable expectations.

7.3.2 Informal knowledge formulation

As mentioned before, some of the experts' personal knowledge can hardly, or not at all, be made explicit. Most experts would not identify a design rule they use even if confronted with a situation where they would use it. On the other hand, facts, as design requirements or constraints, are extremely easy to extract, i.e. the expert(s) can easily identify which questions need to be answered so that they may start or proceed with a particular design. The list of possible answers to those questions can be formalized as a set of first order logic ground clauses representing facts, e.g:

Question	Answer	Fact
• Which code to use in the design ? (BS5502, Eurocode3, ...)	the code to use is BS5502	code to use(BS5502);
• Are there dominant openings ? Where ? (sides, roof)	the roof has dominant openings	has(dominant_openings, roof);
• Who is going to build the foundations ? (contractor, farmer,...)	the farmer will built the foundations	builder(farmer, foundations)

For the informal knowledge formulation, the expert(s) can be confronted with the list of facts elicited during Domain Definition. It becomes then easier for the expert(s) to explicate which rules use these facts. This behaviour corroborates the organization of personal knowledge (3.) proposed by Goel and Pirolli [2].

A practical example of such a result is described:

– During initial interviews between a knowledge engineer and an expert in the design of agricultural buildings, the expert identified a number of questions he needed to place to his clients, in order to commence the design. Later on, at the Informal Knowledge Formulation phase, two of such questions seemed to be of no use for the design:

- Is there a soil survey for this building site ?
- Who is going to build the foundations, the farmer or a contractor ?

The answer for this questions could have produced the following facts (among others):

A soil survey was made for this site → exists(soil_survey);
The farmer will build the foundations → builder(farmer, foundations);
A contractor will build the foundations → builder(contractor, foundations);

Confronted with these possible facts, the designer immediately provided his use of that information, and a new design rule could actually be written:

– If there is no soil survey for the building site and the farmer will build the foundations himself, then the safety factor for the portals' design should be the highest possible in the set of design codes being used

or, design safety factor(highest) if don't know exists(soil_survey) and
builder(farmer, foundations)

Following Grover's proposal [18], a validation of this informal knowledge base should be attempted by producing, for specific scenarios, the behaviour expected by the expert. Only then, the elaboration of a formal knowledge base should proceed.

7.3.3 Knowledge base formulation

Starting from a validated natural language knowledge base, this phase should aim at establishing a formal one, assuming the authors' commitment to first order logic and frames as the knowledge representation formalisms to use (as discussed in paragraph 3.).

At this stage, adequacy of the elicited knowledge as regarding the model SAE is to be enforced. Major attention should also be devoted to the adequacy of the current knowledge base with external procedural programs and cooperating expert systems [11].

An iterative reorganization of the informal and formal knowledge bases should finally take place, as corroboration and validation activities suggest. This should include validation by external experts and end users of the system.

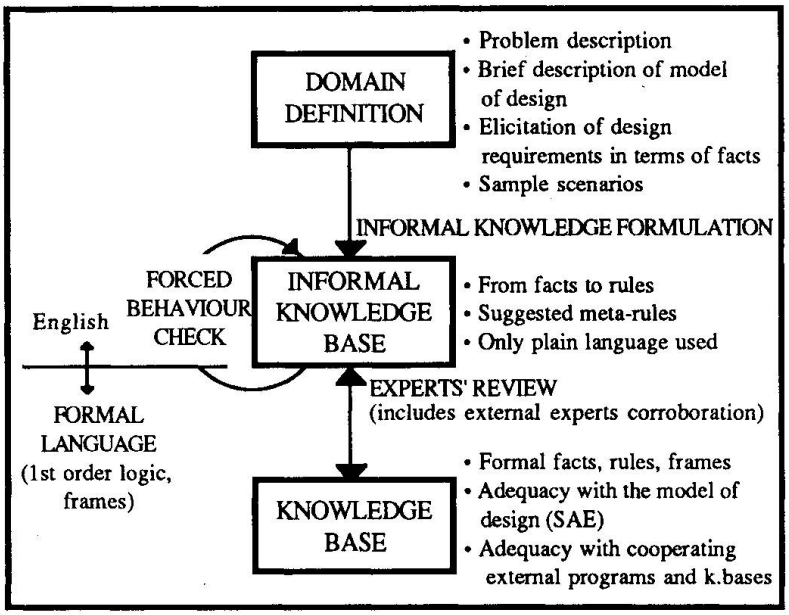


Figure 5

8. CONCLUSION

As fundamental pieces of the implementation of an intelligent computer aided (structural) design system (ICAD), a formal definition of design and a computable model of the design process were presented and briefly described in this work. First order logic and frames were identified as privileged knowledge representation formalisms.

Some major implementation issues were raised and discussed, namely the need for selecting a specific domain of application to pursue implementation and development of the model of the design process.

Some typical knowledge elicitation problems were identified and a structured routine used and enhanced during this project was described.

The ideas and methods presented in this article are being tested by their current application to an ICAD system for assisting in the design of steel portal frames for agricultural buildings.

ACKNOWLEDGEMENTS

The authors would like to thank the partial financial support provided by the Steel Construction Institute (UK) and the Calouste Gulbenkian Foundation (Portugal), in the final stages of this project.

Special thanks are due to Mr. Gordon Rose, Head of Structural Engineering at the Ministry of Agriculture, Fishery and Foods (UK).

REFERENCES

1. FEIJÓ, B., Fundamental steps towards an intelligent CAD system in structural steel, PhD Thesis, Expert Systems Laboratory, Dept. Civil Eng., Imperial College of Science, Technology and Medicine, London, 1988.

2. GOEL, V. and PIROLLI, P., Design within information-processing theory: the design problem space, AI Magazine, spring 89, 1989, pp. 19-36.

3. NEWELL, A. and SIMON, H.A., Human Problem Solving, Prentice-Hall, 1972.

4. VETH, B., An integrated data description language for coding design knowledge, in ten Hagen, P.J.W. and Tomiyama, T. (eds), Intelligent CAD Systems I, Spring Verlag, 1988, pp. 295-313.

5. JOHNSON-LAIRD, P., The Computer and the Mind: an Introduction to Cognitive Science, Fontana Press, 1988.

6. DREYFUS, H.L., What Computers Can't Do: The Limits of Artificial Intelligence, (revised edition), Harper Colophon Books, 1979.

7. SEARLE, J.R., Minds, brains and programs, Behavioral and Brain Sciences, 3, 1980, pp. 417-424.

8. WINOGRAD, T. and FLORES, F., Understanding Computers and Cognition: a New Foundation for Design, Ablex, 1986.

9. OKI, A. and FEIJO, B., Explanation based learning and civil engineering expert systems, Technical Report ESL, Expert Systems Laboratory, Dept. Civil Eng., Imperial College of Science, Technology and Medicine, London, (to appear).

10. DOYLE, J., A truth maintenance system, Artificial Intelligence, 12, 1979, pp. 231-272.

11. BENTO, J. and FEIJÓ, B., On the interaction between procedural and declarative programming styles for expert systems in structural engineering design, Technical Report ESL, Expert Systems Laboratory, Dept. Civil Eng., Imperial College of Science, Technology and Medicine, London, (to appear).

12. COTTA, J.C., DB-Prolog: A portable system based on simple and appropriate solutions, LNEC Thesis for Informatics Specialist, LNEC Proc. 71/13/7492, National Laboratory of Civil Engineering, Lisboa, 1986.

13. GAMMACK, J.G and YOUNG, R.M, Psychological Techniques for Eliciting Expert Knowledge, Research & Development in Expert Systems, Cambridge University Press, 1985, pp. 105-112.

14. BLOOMFIELD, B.P., Capturing Expertise by Rule Induction, The Knowledge Engineering Review, Vol.1 n. 4, 1986.

15. PARSAY, K and CHIGNELL, M., Expert Systems for Experts, John Wiley & Sons, Inc., New York, 1988.

16. DUDA, R.O., GASCHNIG, J., HART, P.E., KONOLIGE, K., REBOH, R. BARRETT, P. and SLOCUM, J., Development of the PROSPECTOR consultation system for mineral exploration, Final Report, SRI Projects 5821 and 6415, SRI International, Inc., Menlo Park, California, 1978.

17. McDERMOTT, J., R1: The formative years, AI Magazine, Vol.2, pp. 21-29, 1981.

18. GROVER, M.D., A Pragmatic Knowledge Acquisition Methodology, Proceedings of the 8th International Joint Conference on Artificial Intelligence (IJCAI), pp. 436-438, 1983.