PROPERTIES OF FUTURE KNOWLEDGE BASED SYSTEMS. THE INTERACTIVE CONSULTATION SYSTEM EXAMPLE.

Computer Aided Architectural Design -
Developments in Education and Practice.
Singapore, May 28-30, 1986

Per Christiansson, Ph.D.
Lund Institute of Technology
Department of Structural Engineering,
Box 118, 221 00 Lund, Sweden.
== 1986.03.21 ==

ABSTRACT

An introduction to knowledge based systems is presented to point out possibilities and limitations of the new software and hardware technology now beginning to be available. A pilot study on the use of an expert system shell, (the ES/P Advisor), is shortly discussed. A part of the Swedish concrete building code was implemented in the expert system shell to exemplify the use of an interactive consultation system. Ideas on how compact video-discs can be used in this type of systems are also put forward.

1. INTRODUCTION.

It is now vital to aim at the formulation of computer system modules that possess a high ability to adapt their behaviour to fundamental human values and a complex and unstandardized (not uniform) building process but at the same time put constraints on them so that we don't end up with a confusion of computerized routines hard to access, control and understand.

The potentials of new software and hardware technology should be beneficially utilized as we formulate concepts and properties for the next generation of building design systems in parallel with the inclusion of existing Cad-systems, databases, calculation programs, evaluation programs etc. It will for example soon be possible to reach (reason with) an extensive amount of encyclopedic information stored on a video compact disc with the ability for crossreferences and use of (moving) pictures. The amount of information on a couple of discs is of the same order as that many people have in their homes or in the office (on arm lentgh distance).

2. ONLY A NEW SOFTWARE GENERATION?

It is now a fact that our society is entering a new age, the information age. Information can be represented with a higher degree of accessibility and "knowledge" than we are used to from books and conventional databases. We now begin to formulate computerized systems that behave more intelligent than we are used to. Information must henceforth to a higher degree than before be regarded and treated as a valuable resource.

What then is an intelligent system? According to professor Edward Feigenbaum, Stanford University, in a TV-interview it may be a system that

- pocess a broad spectrum of behaviours
- supports fine tuned reasoning in problemsolving
- has learning abilities
- understands natural language
- is flexible (can jump between different subjects, problem domains)

  etc.

So called (I)KBS, (Intelligent) Knowledge Based Systems, demand high hardware capacity (millions of syslogics per second, symbolic inferences per second). If a KBS would be able to perform common sence reasoning it would require access to more than 10 million facts/relations (Feigenbaum see above). The next generation of computers that can perform this task is not yet available.

Also in introducing KBS there will be potential risks of deskilling and an evolution towards uniform and unflexible procedures due to building in fundamental human values in the computerized systems. It is not enough to be aware of the risks but also a necessity that we (architects and engineers) actively take part in the design of the new systems.

It is important that the "anarchistic" period, involving formulation and test of KBS, must continue and that the derived results are tested and evaluated against each other. This is especially true as we can expect a dramatic growth in the complexity of software-hardware solutions during the next decades. The emergence of cross fertilization between the fields of building design, computer science, psychology etc. must be supported and encouraged.

3. CAAD TODAY AND TOMORROW.

The design process can shortly be described as a guided search (analysis) and hypothesis formulation through a solution space to find (synthesize) satisfactory solutions, see Fig. 1. The computerized design system must be able to support this activity.

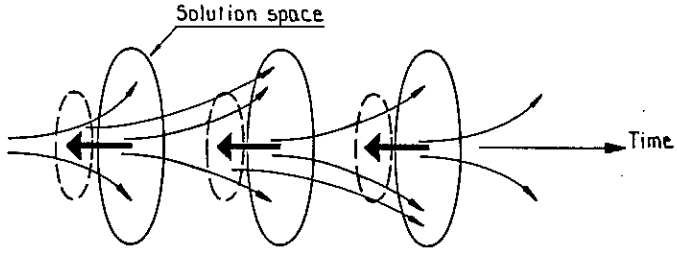

Fig. 1 Design is an explorative process.

We could pose some questions to ourselves to get a feeling for how computer resources could be usefully introduced in the design work.

What tools are available today?

Support detailed design (Cad).
Drawing systems (Cad).
Sketch (not much).
Visualization (on way).
Spread sheet for cost calculation (yes, not integrated).
Word processor for descriptions (yes, few "parametric" text tools).
Integration through layer technique (yes, numbering agreements lacking).

What we do not have.

Integrated software.
Model descriptions (agreed on).
Information transfer between models (some point to point translators and low level standard).

What we want to have. Do we know what? Find out. Emulate. Test.

Integrated software (Cad-KBS-Databases).
Adapted information access (views, documentation).
Model connections (dis-integrated models, level representations).
Flexible design descriptions.
Designer connections/coordination (multidiciplinary Cad).
Design-construction connection (data transfer).
Sketch design tools.
Symbol libraries and catalogues.
Software integration and maintenance tools.
Full control over design tools.
Good output hardware (laser).
Integrated Digital Services Network.
Cheap equipment.
Project management tools.
Graphic interfaces and improved man-machine interface.

What we do not want to have.

Automated architect. Formalized tacit (silent) knowledge.
Legal problems, copywright, fuzzy information responsibilities.
Expensive equipment.
Hardware problems.

Can we get systems that

we can reason with, ask questions and get explanations from?
that can capture experience (personal to project-process)?

are adaptive and learns?

The rest of my paper will cast light on some of the posed questions and hopefully give rise to ideas about the properties we want to give the future design systems.

4. THE KNOWLEDGE BASED SYSTEM.

What is a knowledge representation?

"In AI, a representation of knowledge is a combination of data structures and interpretive procedures that, if used in the right way in a program will lead to "knowledgeable" behaviour.", see Ref. (1). Knowledge may be a representation of objects and their relations in various problem and time domains. Knowledge can encapsulate information about physical objects (rooms, windows, layouts, etc.), events, methods and knowledge about knowledge (meta knowledge). Knowledge based systems have the qualities to support acquisition (elicitation), retrieval and reasoning about knowledge.

Expert systems, see Fig. 2, are the first knowledge based systems used in practice. The technology referred to as the 5:th generation (Japan) or super symbolic computers (USA) is slowly changing our conception of how to program and design computer hardware and software. As a starting point the structure of expert systems will shortly be commented on as familiarity with this group of knowledge based systems brings about new concepts, possibilities and ideas.

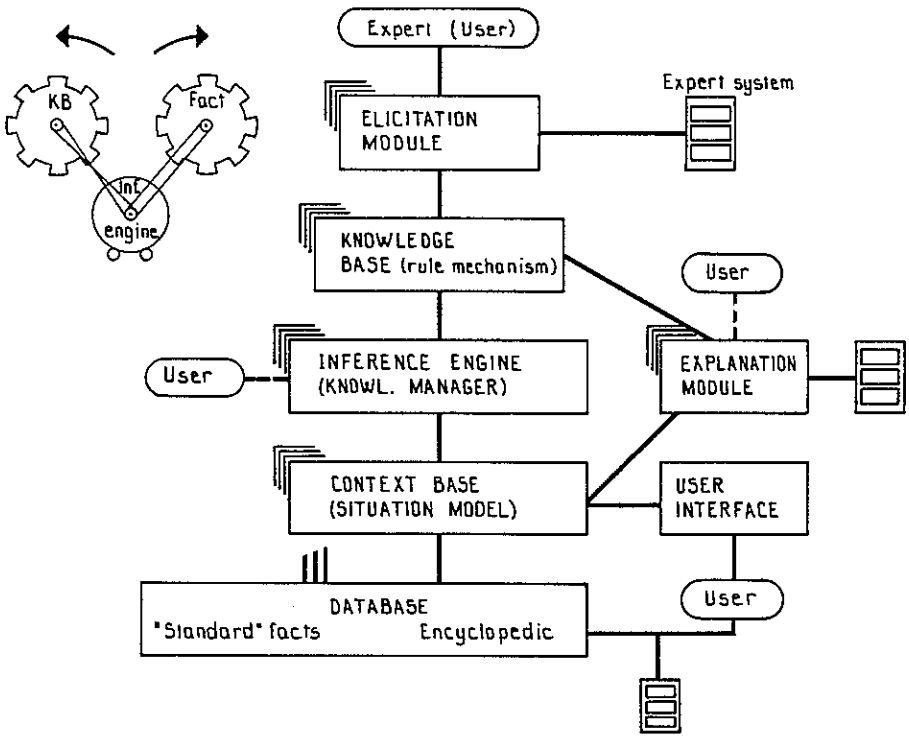

Fig. 2 Conceptual model of an expert system.

Expertsystems may be used as tools for: planning (sheduling), risk analysis (cost, time), design (find a satisfactory/"optimal" solution), interpretation (evaluation), diagnosis (also in emergency situations), configuration (of applications, tools, systems), monitoring (of activities, processes), prediction (forecasting), specification, controlling processes, context (state) descriptions (experience capturing) etc.

Since several years expert systems are built and rebuilt. This work has often been accomplished with tremendous efforts because of the very difficult task of knowledge elicitation. (Until now only a few systems are made exclusively for the construction industry, for example see Ref. (2) regarding preliminary design of high rise buildings). Other examples on development of knowledge based design systems can be found in Refs. (3), (4) and (5).

The choice of representation of knowledge and inference mechanism (reasoning strategies) is strongly dependent on the problem domain (application) under consideration.

The prevailing classes of knowledge representations are (in this paper only the first is dealt with, see Ref. (6)):

(1) Logic (predicate logic)
(2) Frames (object oriented representations)
(3) Productions (condition/action pairs, if <> <> then <>)
(4) Semantic nets
(5) Procedural
(6) Analogical

Of course no sharp borders exist between the representations and in reality a mix will be used. Logic has its advantages in that a clean and consistent representation may be achieved for small closed worlds (though to avoid explosion effects the pure logic often has to be abandoned). Use of frames puts higher demands on definition of syntax and semantics but allows on the other hand for valuable features as procedural attachement and inheritance among objects. The production representation form is often used for advisory and diagnosis systems incorporating uncertainty handling.

Obviously many questions arise and special considerations have to be taken as knowledge based systems are developed. Many of the questions are equally relevant in a non KBS development environment:

(a) Which are the limits of my problem domain?
(b) Which are the links between the knowledge domains in question?
(c) Wanted reasoning mechanism and appropriate knowledge representation?
(d) Which are the limits of regarded time domain?
(e) Is introduction of uncertain reasoning necessary?

(f) How complete must (may) the knowledge be represented?
(g) Which existing "systems" do I want to link in? Databases (encyclopedic, situation/context), procedures.
(h) What are my goals and hypotheses in the problem solving process?
(i) Do I want to add new knowledge to the system, if so how is already stored knowledge affected? Will the system behave more efficiently?
(j) Is the knowledge mainly application or system specific?
(h) How can system performance be validated?

5. THE KNOWLEDGE ENGINEER.

Knowledge engineers, see Fig. 3, are needed to act as links between applications and computerized systems. The knowledge engineer should (a now rare person) have substantial computer science background and be familiar with AI techniques and how to apply them. He/she should also be familiar with the building design work and be able to communicate with users and experts. Finally this rare person should be creative, intuitive as well as able to structural thinking. The knowledge engineer may be looked upon as a traditional systems analyst but hopefully in the future also as a person who designs appropriate tools for conceptual modelling and knowledge elicitation. Tools that should be directly accesible to the user.

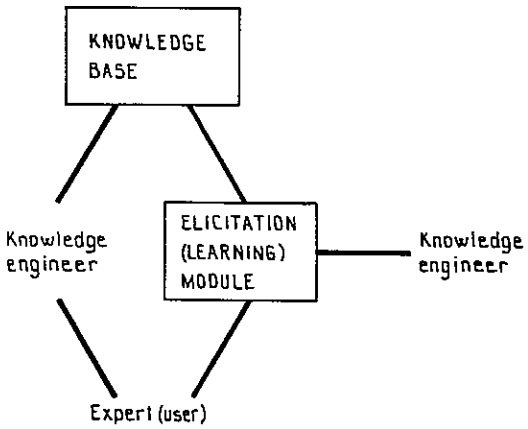

Fig. 3 The role of the knowledge engineer.

The knowledge elicitation work is exemplified by A. Hart in Ref. (7). Where among other things the difficulty for the experts to distinguish between facts and beliefs is mentioned as well as the experts difficulty to be aware of the knowledge he posesses and how he uses it.

6. EXAMPLE ON AN INTERACTIVE CONSULTATION SYSTEM.

In order to test the feasibilty of an interactive consultation system a part of the Swedish concrete building code was implemented in the ES/P Advisor expert system shell from Expert Systems International in Oxford, see Ref. (8).

One conclusion of the pilot study was that it is quite easy to transfer a building code into a practical system of this kind (a "text animation system"). At the end of this chapter possible extensions to the knowledge representation is given as well as ideas on how video-disc technology could be beneficially introduced to the system.

Chapters 7.3.2 and 3.9.5 of the Swedish Regulations for concrete structures (BBK79, Band 2 and 1) were chosen for the pilot study. These chapters contains requirements on concrete quality with regard to the influence of surrounding milieu. The requirements are expressed in terms of required concrete quality class, impermeability, increased percentage of air and manufacturing class. The surrounding milieu also puts requirements on width of cover of reinforcement (due to reinforcement corrosion protection).

The aim with a consultation is to check wether or not there is compliance with the code during design (evaluation).

After some "knowledge engineering" work the following controlling parameters were defined, namely moisture milieu, salt content, temperature variation, water pressure and surface inclination. These parameters constitutes the building application context (situation model) under consideration when they are given values (instantiated) during consultation. They may also be given new values by means of the CHANGE command (see below).

The ES/P Advisor shell can shortly be described as a "text animation" system. The knowledge is represented as paragraphs of text grouped into sections. Conditions in the form of logical propositions control the output of paragraphs and references to sections. The conditions are built up of clauses containing parameters. Parameters are given values by rules in the parameter definitions or by the user (or a combination of both).

Four types of parameters can be defined:

facts	(true or false)
numbers	
category	(one of a number of prescribed options, strings)
phrase	(a string)

The parameter moister-milieu has the following definition in the system:

```
moister-milieu:  'Surrounding moister milieu'
   category
```

```
explanation
~The moister milieu may be either very damp or dry~&
~A very damp milieu may cause a very or moderate concrete~&
~aggressive milieu~
options
damp - ~Very damp milieu~
dry  - ~Not very damp (or dry) milieu~
askable
~In what milieu do the concrete reside?~.
```

The application knowledge is encoded in a formal language called, KRL, Knowledge Representation Language. This code is compiled by the system into a knowledge base written in Prolog which in its turn later is accessed by the Consultation Shell (inference engine and user interface). (Only deterministic knowledge). It is also possible for the user to embed Prolog clauses in the KRL code to augment the inference mechanism and provide communication to non-standard I/O devices. Prolog clauses can either be placed in paragraphs or within rules in parameter definitions.

The search in the knowledge base is done in section after section in the order they are referenced. Controlling conditions to paragraphs and section references are evaluated and if true the paragraph is output or section referenced. Paragraphs may be output unconditionally. Parameter values may be referenced in paragraphs (the value inserted into the text). The WHY command will trace the current line of reasoning.

Figs. 4-6 shows dialogue examples from a consultation:

```
I-------------------------------------------------------------I
IGOAL: moister-milieu          SECTION: demand-concrete-milieu I
I-------------------------------------------------------------I
I In what milieu do the concrete reside?                      I
I                                                             I
I  (1) - Very damp milieu                                     I
I  (2) - Not very damp (or dry) milieu                        I
I                                                             I
I-------------------------------------------------------------I
I Enter the number of relevant entry : 1                      I
I-------------------------------------------------------------I
```

Fig. 4 Screen appearance during ES/P Advisor consultation.

If WHY instead of 1 is entered, see Fig. 4, the screen according to Fig. 5 will appear.

```
I-------------------------------------------------------------I
I GOAL: moister-milieu          SECTION: demand-concrete-milieu I
I-------------------------------------------------------------I
I  am asking this because I wish to establish the state of     I
I  Surrounding moister milieu  (moister-milieu)                I
I                                                              I
I which is required in order to evaluate the expression:       I
I   "very-concrete-aggressive"                                 I
I                                                              I
I which establishes the state of                               I
I   definition of surrounding milieu (milieu)                  I
I                                                              I
I which is a necessary pre-condition for the display of the    I
I following paragraph                                          I
I                                                              I
I                                                              I
I     VERY CONCRETE AGGRESSIVE MILIEU                          I
I                                                              I
I Concrete quality                                Manufacturing I
I ------------------------------------------------class        I
I Quality  Imperme-  Increased perc. Water-cement              I
I class    ability   of air          ratio (max)               I
I ---------------------------------------------------------    I
I                                                              I
I  K40     Required  Required          0.50          I         I
I                                                              I
I-------------------------------------------------------------I
```

Fig. 5 WHY command, trace of the current line of reasoning.

Fig. 5 shows how the current goal under investigation will result in output of a relevant code requirement (in a form familiar to the user).

The next question is about salt content in the surrounding milieu, see Fig. 6.

```
I-------------------------------------------------------------I
I GOAL: salt-content            SECTION: demand-concrete-milieu I
I-------------------------------------------------------------I
I Assess the salt content of surounding milieu. Is it          I
I                                                              I
I  (1) - High                                                  I
I  (2) - Average                                               I
I  (3) - Small                                                 I
I  (4) - No presence of salt                                   I
I-------------------------------------------------------------I
I Enter the number of relevant entry : 1                       I
I-------------------------------------------------------------I
```

Fig. 6 Continue to establish context by parameter instantiation.

Etcetera!!

A consultation can be SAVEd. Questions can be EXPLAINEd. Parameter descriptions can be SHOWn. STATUS of beliefs (parameter values plus the short description within "" of the first parameter definition line) can be reported. The consultation session can be LOGged.

During consultation a parameter value may be CHANGEd by the user. The shell will backtrack and repeat the chain of reasoning. If the user so wishes the revised advices will be displayed.

The system was run on a IBM/PC under MS-DOS. In all 5 sections, 29 paragraphes and 11 parameters were defined (15000 bytes source code) covering approximately 3% of the concrete code. It was coded in 8 man days which implies at least 1-2 man years to code the total regulation.

During the implementation work inconsistency was discovered which could have been avoided if the code writers had access to (were "forced" to use) a computerized code building system, see also Ref. (9) and (10).

In the sketched system the user (designer, etc.) has to find the appropriate knowledge base if the code is fragmentized into many independent bases (due to lack of computer capacity to handle one big knowledge base). Connections between the bases (regulation context) could in this case be established via external files (storing/reading Prolog clauses). These files could be built up during consultation or as a result of an introductory consultation with a regulation index knowledge base. In the latter case the index knowledge base encapsulates meta knowledge about the sub bases (knowledge about the regulation content).

The regulation index knowledge base could of cource also be expanded (or rather supplemented in a separate base) with generic knowledge about which regulation knowledge bases that are relevant for different application domains (requirements on impermeability for this type of structural element etc.) This is not an easy task because it requires agreement on definition of building "grammars" and a potent tool set to keep updating loosely connected knowledge bases.

Finally some advantages of the described approach for designing interactive consultation systems will be commented on. During encoding a better understanding of the problem domain will be achieved. By using text chunks and perhaps pictures later a step on the way towards "intelligent databases" may be taken (making books, catalogues etc. more "intelligent").

As the compact video-disc now is available to a reasonable price (also chep player equipment) it ought to be possible to store the "paragraphs" (text, pictures) on the video-disc. A digital video-disc could also store high level knowledge bases and the expert system shell itself and thus be totally self contained. (During development stages only the "paragraphs" should be stored on the video-discs. Making discs is a costly procedure). By taking

advantage of the developments in the computer graphic fields pictures can be produced and refined and stored in a compressed form. Information stored on video-discs is handled and transferred in a very effective way. A video compact disc can approximately contain 550 Mega bytes.

7. THE FUTURE SYSTEM

There will in the time to come be an increasing interest in making knowledge acquisition tools more effective. As a prerequisite we should try to define basic knowledge structures, see also Ref. (11), which allow us to add new knowledge to the computerized systems thereby sometimes relieving us from the impossible task of designing "complete" ready-to-use systems. In other words we should aim at designing systems that are taught rather than programmed. The systems should provide the users with tools to better control and understand the process of design. The user should concequently have access to an effective toolbox with guidance for effective tool selections.

Emphasize should be on tools to
(a) create (build, compose) a model of the problem domain (application) under consideration,
(b) to handle existing tools and to introduce new tools to the system,
(c) support integration of process actors and system modules,
(e) establish and refine situation models (contexts),

Fig. 7, from Ref. (12), shows how the conceptual building process model could be implemented in a computerized environment. The basic underlying knowledge structure (general building process knowledge) must initially be designed and may subsequently be modified. During modification stored knowledge should be transferable to modified structures.

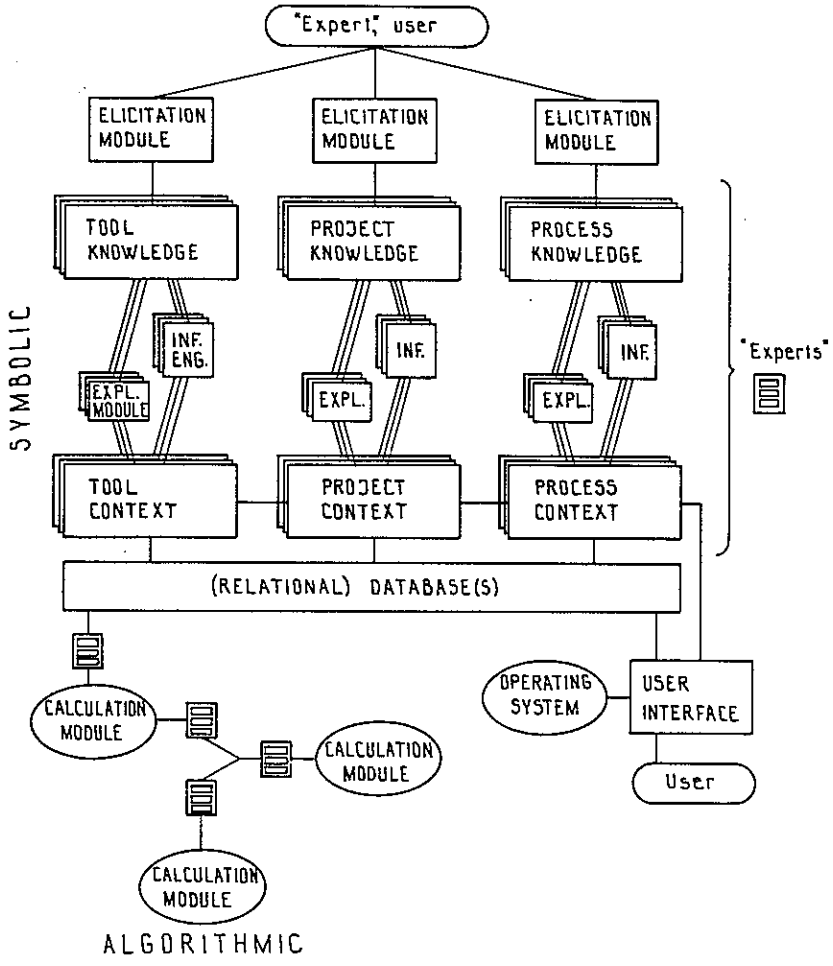

Fig. 7 Implementation of a building design system.

Of great interest is the ongoing research about the possibilities to implement multiple concurrent processes in new computer architectures. Different approaches are tested on how to best exploit computer architectures which admit many processes to be active in parallel, see for example Refs. (13) and (14).

8. CONCLUSIONS

The emerging software/hardware technology opens up the possibilities for us to do wise and innovative formulations on how to introduce computer resources in the design process. We should then be able to achieve increased quality on work content and produced results. It should also be possible to create flexible and adaptive building process environments and project descriptions and include interaction with manual routines, as well as achieve increased productivity.

REFERENCES

1. "Representation of Knowledge", in The Handbook of Artificial Intelligence, edited by A. Barr, E. A. Feigenbaum (Pitman Books Limited, London, 1982), 3rd ed., Volume 1, Chapter III, p. 143.

2. M. L. Maher, S. J. Fenves, "HI-RISE: An Expert System for the Preliminary Structural Design of High Rise Buildings", (Department of Civil Engineering, Carnegie Mellon University, Pittsburgh, 1984) pp. 1-11. (Presented at IFIP, WG5.2 Working Conference on Knowledge Engineering in Computer-Aided Design, Budapest, 1984).

3. A. Bijl, P. J. Szalapaj, "Knowing where to draw the line", (EDCAAD, Department of Architecture, University of Edinburgh, 1984), pp. 1-19. (Presented at IFIP, WG5.2 Working Conference on Knowledge Engineering in Computer-Aided Design, Budapest, 1984).

4. J. Gero, "An Overview of Knowledge Engineering and its Relevance to Caad", in Proceedings of CAAD futures, (Technical University of Delft, 1985), pp. 15-18.

5. D. R. Rehak, H. C. Howard, "Interfacing Expert Systems with Design Databases in Integrated CAD Systems", Computer Aided Design, 17, (9), pp. 443-454, (1985).

6. W. F. Clocksin, C. S. Mellish, "Programming in Prolog", (Springer-Verlag, Berlin Heidelberg New York, 1981), pp. 1-279.

7. A. Hart, "Knowledge elicitation: issues and methods", Computer Aided Design, 17, (9), pp. 455-462, (1985).

8. "ES/P Advisor. User Guide and Reference Manual", Issue 1 (Expert Systems International Ltd, Oxford, 1984), pp 1-202.

9. F. I. Stahl, R. N. Wright, S. J. Fenves, "Expressing standards for computer-aided design", Computer Aided Design, 15, (6), pp. 329-334, (1983).

10. M. A. Rosenman, J. S. Gero, "Design codes as expert systems", Computer Aided Design, 17, (9), pp. 399-409, (1985).

11. P. Christiansson, "Integrated Computer Aided Design. Present and Future Data Structures", CIB Proceedings Publication 78, CIB W78 (CIB, Rotterdam, 1984), pp. 20-25.

12. P. Christiansson, "Structuring a Learning Building Design System", (Department of Structural Engineering, Lund Institute of Technology, 1986), pp 1-9.

13. V. Kumar, "Thoughts on Parallel Processing", BYTE, May, pp. 174-175 (1985).

14. L. M. Pereira, R. Nasr, "Delta-Prolog: A Distributed Logic Programming Language", in Fifth Generation Computer Systems 1984. Proceedings of the International Conference on Fifth Computer Systems 1984, edited by Institute for New Generation Computer Technology (ICOT) (North Holland, Amsterdam, 1984), pp. 283-291.