Structuring a Learning Building Design System.

Per Christiansson

Lund Institute of Technology
Department of Structural Engineering,
Box 118, 221 00 Lund, Sweden.
== 1986.03.14 ==

KEYWORDS

Conceptual modelling, Learning domain, Integrated database, Artificial intelligence, Design system.

ABSTRACT

It is now vital to aim at 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. In the paper formulations are made of basic artifact skeletons outgoing from the properties we want to give integrated CAD systems and to those rules by which the growth of the systems are governed. System learning domains including conceptual modelling tools are presented aiming at supporting professional skill, creativity and integration between process actors. The basis for system implementation is frames, descriptive language (prolog) and relational databases with regard taken to future possibilities to parallel processing.

Structurant un systeme d´assimilation de design de construction.

Per Christiansson

Université des Technique et Sciences de Lund
Institut de Technique a Construction
B.P. 118, S-221 00 Lund, Suede.
== 1986.03.14 ==

MOTS-CLES

Modelage conceptif, Domains d´érudition, Base de données intégrée, Intelligence artificielle, Systeme de design.

SOMMAIRE

Il est a présent important de viser a une formation de modules de systeme d´ordinateur possédant une grand capacité a adapter leur comportement aux valeurs humaines fondamentaux et a un procédé de construction complexe et pas normalisé (pas uniforme), mais en même temps leur mettre des contraintes pour que nous ne finissions pas par une confusion de routines informatisées difficiles a pénétrer, controler et comprendre. Dans le text, des formations de charpentes de base sont fait au départ des qualités que nous désirons donner aux systemes de CAO intégrés et aux regles par lesquelles la croissance des systemes sera dirigée. Des domains d´érudition comprenant des outils de modelage conceptif, sont présentés visant a soutenir de la technique professionelle, de la création et de l´integration entre des acteurs de procédé. Les bases pour mettre en oeuvre un tel systeme sont des plans (´frames´), de la langue descriptive (Prolog) et des bases de données relationaux eu égard aux possibilités futur de traitement parallele.

1. INTRODUCTION

It is now vital to aim at 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.

We are now in the position to do wise formulations on how to use new technology to achieve increased quality on work content and produced results. We will also be able to create flexible and adaptive building process environment and project descriptions, including interaction with manual routines, as well as achieve increased productivity.

2. POTENTIALS AND LIMITS OF KBS

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).

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 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. KNOWLEDGE REPRESENTATIONS AND INFERENCE MECHANISMS

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.

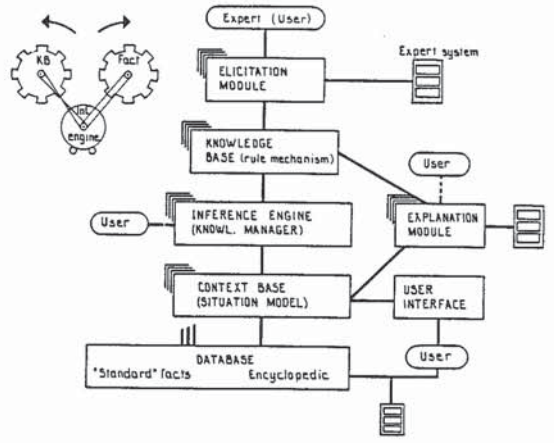

Fig 1. Conceptual model of an expert system.

Expert systems, see Fig. 1, 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.

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).

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 three are dealt with):

(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.

In our resarch we are using Prolog, see Ref. (2), to construct small inference engines (with control strategies), knowledge bases and context models.

Use of frame representations, see Refs. (3) and (4), facilitates expectation-driven processing. Our ongoing research also comprises a look at the possibilities for frame (object) definitions for a part of the building design process using the POPLOG system (from System Designers Scientific and University of Sussex). In this connection a very important issue is that of inheritance and succesive refinement of situation models with allowans for partial backtracking in a design sequence.

4. A CONCEPTUAL MULTILEVEL BUILDING PROCESS MODEL

The design process can shortly be described as a guided search and hypothesis formulation through a solution space to find satisfactory solutions, see Fig. 2. The computerized design system must be able to support this activity. See also Ref. (5) regarding planning systems in design and Ref. (6) regarding preliminary design of high rise buildings.

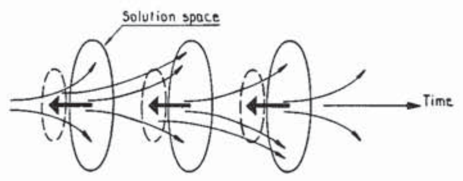

Fig 2. Design is an explorative process.

As an initial approach in the conceptual modelling procedure the building project knowledge is separated from the building process knowledge according to Fig. 3, from Ref. (7). The building project model (connected to the "functional project model" node in Fig. 3) contains knowledge about the model of the product, the final product and its documentation. The nodes and their connections reflect the flow, storage and manipulation of information (and material) and mechanisms to control these activities.

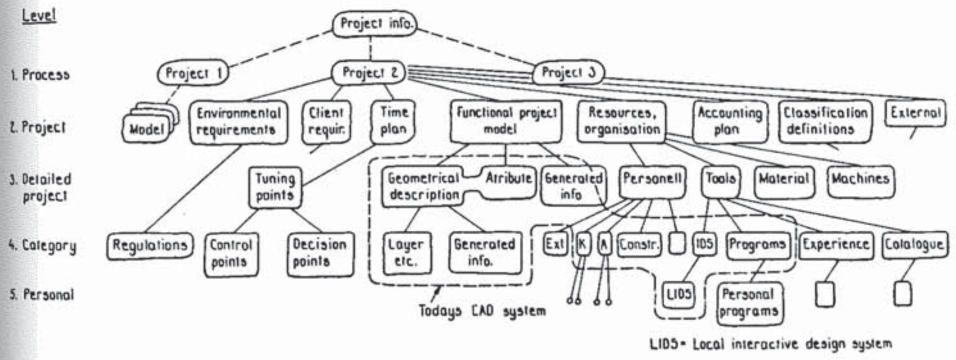

Fig 3. Building process information and control structures.

Fig. 4, Ref. (8), is now superimposed Fig. 3 to introduce the time concept and the dynamic relation between the real world and its implementation in the computerized system.

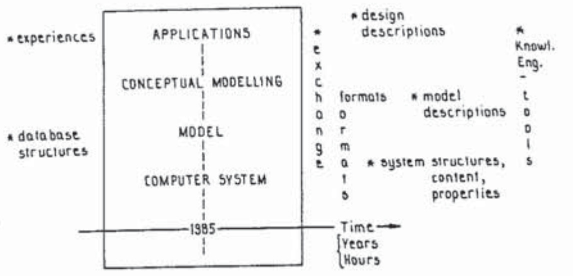

Fig 4. From application to implementation in different time domains.

I believe that a conceptual total model as the one sketched is necessary to better understand where "local" improvements may be made.

5. LEARNING DOMAINS

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 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. 5 shows how the conceptual building process model could be implemented in a computerized environment.

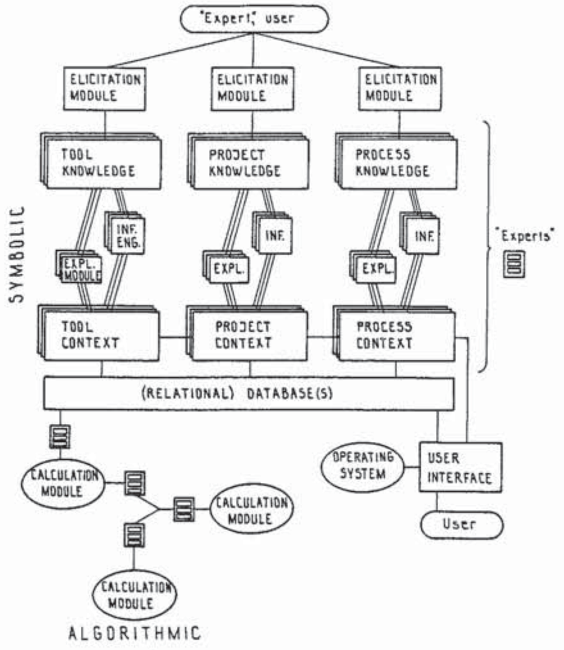

Fig 5. Implementation of a building design system.

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.

Automated machine learning is not dealt with in this paper, see Ref. (9), though interesting areas are to be found in connection with adaptive man-machine interface and interfaces between system modules.

6. THE KNOWLEDGE ENGINEER

Knowledge engineers, see Fig. 6, are needed to act as links between applications and computerized systems. The knowledge engineer may thus 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.

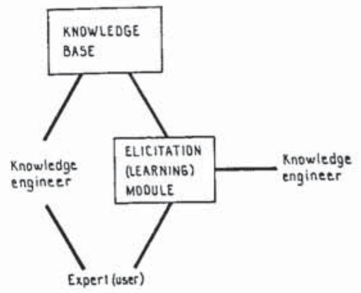

Fig 6. The role of the knowledge engineer.

Below "learning by example" is illustrated by use of the EXPERT-EASE system (from Intelligent Terminals Ltd, Glasgow). The knowledge elicitation work is further exemplified by A. Hart in Ref. (10). 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.

7. EXAMPLE ON RULE INDUCTION

The results from rule induction sessions could be used to create a base structure for an advisory or configurational expert system. During induction it is easier for the expert to describe a set of examples, which are successively refined and added to, than to try to describe a forward reasoning process from a general problem domain description to an advice (goal). A decision tree is produced which may be translated to a consistent set of production rules.

During an iterative process creating examples (giving new or defined values to defined main attributes, see TABLE I) the resultant rule set is refined. The system is especially useful when the "expert" do not know how he reaches a particular decision, but only what input data that decision requires. (Uncertainties are not handled in this particular system). A bad choice of attributes will produce poor results. Attribute must perhaps be changed or deleted as work progresses. The result is very sensitive to the given examples especially lower down in the decision tree.

The appearance, during a rule induction session, of the attribute and example screens and the produced decision tree is illustrated in TABLE I. The generated advisory system may be intermediately run during its building up.

TABLE I. Attribute, example and rule screens during a rule induction session using Expert-Ease.

	logical foundation	logical labor/mtrl	integer bridge-length	logical material	(Advice) bridge-type
1	shallow	high		concrete	ARCH
2	deep	low		steel	BOX-BEAM
				aluminium	BOX-GIRDER
				stone	SUSPENSION
					UNSUITABLE

	logical foundation	logical labor/mtrl	integer bridge-length	logical material	(Advice) bridge_type
1.	shallow	low	250	concrete	ARCH
2.	shallow	high	250	steel	ARCH
3.	deep	low	250	steel	SUSPENSION
4.	*	*	600	steel	SUSPENSION
5.	*	*	600	stone	UNSUITABLE

Rule screen:

```
material
   concrete : ARCH
      steel : bridge-length
             < 425 : foundation
                 shallow : ARCH
                    deep : SUSPENSION
             >=425 : SUSPENSION
  aluminium : null
      stone : UNSUITABLE
```

8. FUTURE SYSTEMS

Researchers are now investigating 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 Ref. (11).

Work is underway concerning implementation of functional and descriptive languages on parallel processor machines. The HOPE language implemented on INMOS Transputers is shortly commented on in Ref. (12). In Ref. (13) a distributed logic programming language is described, Delta-prolog, which to some extent admits parallel processing. An architecture, based on the so called MIMD concept, Multiple Instruction Multiple Data stream in a parallel processor environment, is described in Ref. (14).

Ongoing work to design parallel computers can be found in Ref. (15), (11) and in the proceedings of the International Conference on Fifth generation Computer Systems, see Ref. (13), where not only the Japanese efforts is described.

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

3. 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).

4. 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).

5. 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.

6. 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).

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

8. P. Christiansson, "Integrated Systems. Results of the W78 Survey", (Lund Institute of Technology, Department of Structural Engineering, 1984), p. 12. (Presented at the CIB W78 Integrated Cad Symposium, Rotterda

9. "Learning and Inductive Inference", in The Handbook of Artificial Intelligence, edited by P. R. Cohen, E. A. Feigenbaum (Pitman Books Limited, London, 1982), 3rd ed., vol. 3, Chapter XIV, pp. 323-511.

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

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

12. D. Pountain, "A look at the ALICE hardware and the HOPE language", BYTE, May, pp. 385-395 (1985).

13. 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.

14. J. Gabriel, T. Lindholm, E. L. Lusk, R. A. Overbeek, "Logic Programming on the HEP", in Parallel MIMD Computation: The Hep Supercomputer and its Applications, edited by J. S. Kowalik (MIT Press, Cambridge, 1985), pp. 181-202.

15. P. Walker, "The Transputer", BYTE, May, pp. 219-235 (1985).