Architectural pre-processor for engineering expert systems

Autor(en): **Schmitt, Gerhard**

Objekttyp: **Article**

Zeitschrift: **IABSE reports = Rapports AIPC = IVBH Berichte**

Band (Jahr): **58 (1989)**

PDF erstellt am: **11.09.2024**

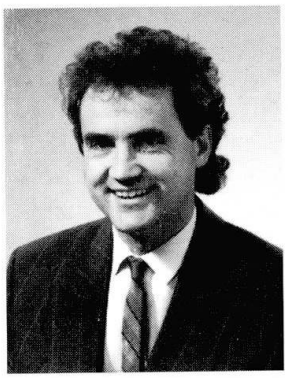
Persistenter Link: https://doi.org/10.5169/seals-44916

Architectural Pre-Processor for Engineering Expert Systems

Architecture d'un pre-processeur pour les systèmes experts d'ingénieurs

Präprozessor für architektonische Expertensysteme

Gerhard SCHMITT
Professor Dr. Ing.
Swiss Fed. Inst. of Technology
Zürich, Switzerland



Gerhard Schmitt, born in 1953, received his Architecture degree from the University of California at Berkeley and his PhD from the Technical University of Münich. Since 1984 Professor at Carnegie Mellon University, USA, he now holds the chair for CAAD. at ETH..

SUMMARY
In this paper, we present three issues important for the development of knowledge-based Architecture and Engineering design environments: first, intelligent interaction and feedback mechanisms between engineering and architectural design and their computational treatment, introducing classes of design and classes of tools. Secondly, the critical presentation of ARCHPLAN, a working prototype within IBDE, the Integrate Building Design Environment. Thirdly, thoughts on the role of criticism between programs and their implementation.

RESUME
Dans cet article nous présentons trois domaines importants pour le développement d'environnements de conception à base de connaissances en architecture et en ingénierie: Premièrement, et les méchanismes intélligents d'interaction et d'information du retour (feedback) ainsi que leur traitement interne par ordinateur, en introdusant un concept de classes de conception et d'outils. Deuxièmement, la présentation critique d'ARCHPLAN, un prototype fonctionnant déjà dans l'environnement de conception IBDE (Integrated Building Design Environment - environnement intégré de conception des bâtiments). Troisièmement, des réflexions sur le rôle de la critique entre les programmes et leur implementation.

ZUSAMMENFASSUNG
In diesem Artikel stellen wir drei Gebiete vor, die für Entwicklung von wissensbasierten Architektur- und Ingenieur-Planungsumgebungen von Wichtigkeit sind. Erstens, intelligente Interaktions- und Rückkoppelungsmechanismen, sowie deren computerinterne Behandlung, wobei das Konzept von Entwurfsklassen und zugehörigen Entwurfswerkzeugen vorgestellt wird. Zweitens die kritische Präsentation von ARCHPLAN, einem bereits funktionierenden Prototyp innerhalb von IBDE (Integrierte Gebäude-Entwurfs-Umgebung). Drittens, Gedanken und Rolle von Kritik zwischen Programmen sowie deren Implementation.

Abstract

High quality design under time constraints in architecture and the building industry is becoming one of the critical requirements in the development of new products. In their respective domains, knowledge based systems in diagnostics, maintenance, and construction have reached a high degree of sophistication, whereas less examples exist in the area of design and the integration of interdisciplinary knowledge. One of the problems is the loss of vital, qualitative information in the process of transferring an architectural design to further engineering synthesis and analysis. While only in the fewest cases the structural engineer is expected to deliver the architectural design or the architect is expected to complete the structural analysis, the two activities are closely related by sensitive interdependences.
The purpose of this paper is to present a number of intelligent interaction and feedback approaches between civil engineering and architecture and to suggest their computational treatment within an environment of knowledge-based systems. It is philosophical in the sense that it proposes and critically describes attempts rather than solutions.
The paper is divided into three parts. Part One addresses feasible design processes and their representations. Part Two describes an Integrated Building Design Environment and the architectural preprocessor necessary to begin the engineering design as a prototype system. Part Three presents thoughts on criticism mechanisms between knowledge-based processes.

1. Part One: Feasible Design Processes and Representations in Architecture and Civil Engineering

Effective communication and exchange of information between disciplines often encounters a language and representation barrier. The use of different models to describe and reason about the world complicates matters and in many cases prevents designers from obtaining crucial feedback from other disciplines. There are, at the moment, three possibilities to escape this dilemma:

- Reliance on known algorithms and rules. An extensive body of knowledge exists in both the engineering and architecture literature describing solutions to known design problems. A practical example is the book "Entwurfslehre" by Neufert [1] which represents a significant collection of design knowledge and has been used and updated for the last 35 years. Similar books exist in civil engineering [2]. Books of this kind generally do not, however, help in solving new or unexpected design problems.
- Principled representation. The successful search for a generalized and principled representation that holds true for two or more disciplines could solve many of the communication and feedback problems between architecture and engineering. To solve similar representation problems is the central goal of qualitative physics [3], where models based on principled representations are extensively used [4]. Qualitative physics formalizes first-principles knowledge about physical phenomena. The resulting library of *domain models* provides knowledge for instantiating qualitative causal models of a physical system, such

as a steam plant, a mechanism, a structural system, or a building. It provides a uniform representation of a large class of phenomena, possibly covering several domains.

- Reasoning based on cases. The traditional knowledge representation paradigm of Artificial Intelligence is that of general production rules. Applied to design, it suffers from two shortcomings: (a) it is difficult to construct a coherent set of rules for representing an extensive body of knowledge, and (b) it is not clear how to formulate the knowledge needed to produce a complete design in the form of general rules. Furthermore, there is strong psychological evidence that people do not reason from general rules alone, but often refer to the memory of previously solved similar problems, as shown by Schank [5] and Akin [6]. This observation has led to the paradigm of case-based reasoning (CBR), which also helps to eliminate some of the deficiencies of rule-based systems.

While principled representation and case-based reasoning for design are the subject of on-going research, the first approach is computationally straight-forward and applied in some integrated system projects underway in Stanford [7], Carnegie Mellon [8], and Liege [9] The ARCHPLAN preprocessor described in Part Two is also based on this model and concentrates on sections of the architecture and civil engineering domains to explore the use of one representation and one model to facilitate the exchange of crucial information that goes beyond syntactic specifications. Levels on which common representations and models are desirable are:

- Low level representation. This includes syntactic representations of objects and functions to describe the geometry and purpose of designed artifacts. As differences in describing geometry do exist, standards that allow the transfer of descriptions between applications without information loss are of particular importance. For product modeling, one of the standards proposed is the STEP (STandard for the Exchange of Product model data) model [10].
- Intermediate level representation. This level mainly describes larger knowledge entities and may combine syntactic and functional descriptions in one representation. Commonly used are procedures, rules, frames, or objects, of which frames have emerged as the most flexible.
- High level representation. This includes models of expression for semantics or structures of design knowledge such as chunks and prototypes.

Beside these representations, compatible models of the design process itself in the respective disciplines are needed to go beyond sequential and thus cumbersome design simulations. Proposed by Gero, Maher, and Zhang [11], three different types of design processes in both architecture and engineering are proposed: routine, innovative, and creative design. Although such a simplification is not always acceptable and will not completely describe real-world design, it is useful to divide the otherwise too extensive field of design into manageable parts. The following is a brief description of the three types of design.

1.1. Routine Design

Routine design is a goal-directed activity, characterized by prototype refinement or instantiation of designs from a catalogue of parameterized examples. Beginning with a given prototype of, for example, a piece of furniture or equipment, the designer adjusts a number of parameters to the specifications of the design program. The parameters, typically geometric properties or materials, are normally well understood and are manipulated either in the designer's memory or with advanced modeling systems. The functional requirements of the design are known and the semantics or the teleology, (the purpose of each element) of the design are not changed but accepted from previous examples. Routine design relies heavily on instantiation of designs from a catalogue of parameterized examples which are considered relevant for the design problem at hand. Without doubt, routine design is a good preparation for innovative and creative design and its importance must therefore not be underestimated. The refinement of a standard floor plan is a good example. It could be claimed that most great architects started their career with routine design [12]. The example described later in the paper refers to this type of design.

1.2. Innovative Design

According to Faltings [13], this type of design could be described as prototype combination, which makes it a prime example for case-based reasoning. According to Gero, innovative design is achievable with prototype modification [11]. In both cases, the designer has a general idea of the desired object and the design process is, as in routine design, a goal-directed activity. However, the design process cannot be completed with routine design because the functional description or the object properties are not achievable utilizing a given prototype. Therefore, the combination of two or more prototypes which each have some of the desired properties is necessary. An example would be the development of intelligent office buildings for which some new information infrastructure needs are still unknown. Case-based reasoning and explanation-based learning systems are of particular interest in innovative design because they may selectively capture desired qualities of existing buildings and avoid their shortcomings. Once these qualities have been discovered, an existing prototype may indeed be modified to incorporate innovations.

1.3. Creative Design

Creative design is the development of new solutions that may only be partially defined at the outset. Both functional requirements and the object's properties are not completely known. It is possible that a unique solution may be found to a problem in which case the result would be an archetype. In most cases, prototype creation is necessary, which later can be combined and modified (innovative design) and instantiated (routine design). An example is the invention of a new machine, such as the personal computer. Once sufficiently understood and formalized, creative design becomes part of the mainstream and can be proceduralized. In some cases, architectural language aspects of once creative design, such as the Barcelona Pavilion or Richard Meier's residences, can be proceduralized or implemented in shape grammars [12].

1.4. Levels of Design and Levels of Representation

Until recently, commercial CAD programs typically allowed manipulation of design at the level of geometry or routine design: once a design is completed with traditional means and input into the computer, manipulations are performed on the geometric model. Macros containing simple checking rules for stair angles or distance calculations could be seen as intermediate level representations of routine design knowledge. And parameterized prototypes of furniture and stairs that interactively generate more complex, but well defined building elements, are examples of higher level representations in routine design. The existence of this complete set of representation and manipulation tools, developed to support day-to-day design activities, is one reason for the popularity of computers in routine design. In order to build as powerful tools to support innovative and creative design, the following table might serve as a preliminary attempt to relate types of design and levels of representation:

Level of Representation	**Routine Design**	**Innovative Design**	**Creative Design**
Low	Geometry	Syntax	Semantics
Intermediate	Rules, Frames	Rules, Frames	Prototypes
High	Prototypes	Object Semantics	Structures

2. Part Two: The IBDE Prototype

Based on the overview of design processes and representations in Part One, this section describes the Integrated Building Design Environment (IBDE) which eventually should have the capability to perform routine and innovative design at a level of competence and completeness approaching that of a human designer and out-performing a designer in terms of consistency and time required to complete a design.
IBDE integrates 7 independent, knowledge-based computer programs. Their declarative representation of knowledge permits rapid development and modification. The prototype integrated environment serves as a test bed for examining the following issues:

- Integration of multiple disciplines, such as architecture, engineering, and HVAC design. A positive result is a more realistic simulation of the final building in the design phase and the elimination of coordination problems that usually occur during construction and can lead to costly changes.
- Discipline specific multiple views of a building design. This issue is important because it allows complete and consistent observation of the design and prevents the discovery of conflicts at too late a stage.
- Improvement of communication between disciplines. Although more a side effect of the project, it has developed into one of its most positive aspects. An integrated design environment helps eliminate most common misunderstandings about engineering and architectural design.

integrated design environment helps eliminate most common misunderstandings about engineering and architectural design.

- Automation of sub-processes to various degrees. Wherever possible, and whenever enough knowledge is available for parameterization, processes may be automated. Examples are layout of elevator core areas and an HVAC design sub-system.
- Hardware and software independence. Traditionally hardware dependent applications can be used in a distributed environment. Parallel execution of individual processes is possible.

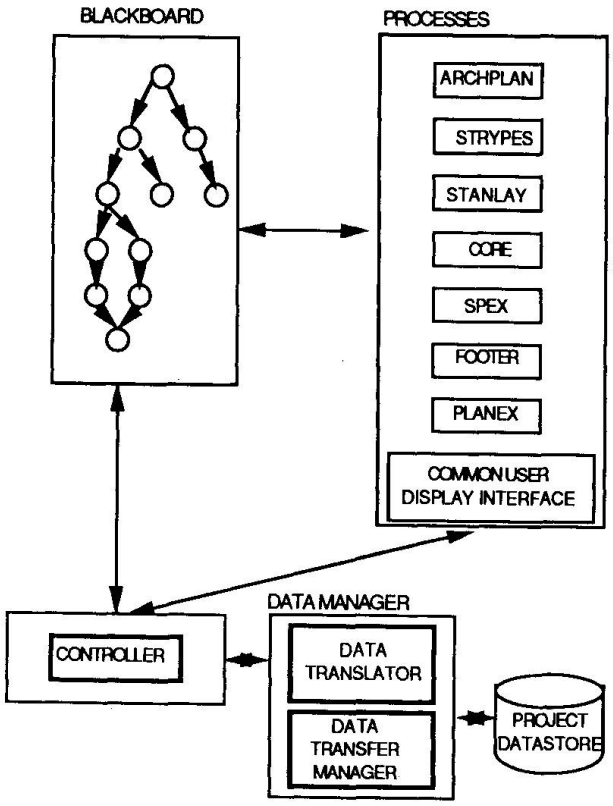


Figure 1: Schematic view of the IBDE architecture.

While the systems produces a complete and consistent design within reasonable time (two to three hours) and describes the building to a high level of detail, the system could be improved. One of the most important necessities is the implementation of critiques that act between processes. The blackboard, which is now used for posting status messages, needs to take on a more significant role. The individual processes need refinement as well: as of now, IBDE can only simulate rectangular office buildings with interior cores. Finally, development work is necessary on the common display interface and on the individual program interfaces to provide the designer with a more friendly discipline specific and unified view of the project.

2.1. ARCHPLAN - The Architectural Preprocessor

ARCHPLAN - ARCHitectural PLANning expert system - is the first of the seven knowledge-based processes and provides necessary input for IBDE to proceed. ARCHPLAN assists in the development of the conceptual design of high-rise office buildings. Input describes the site, the client's program, budget, and geometric constraints. The output produces three-dimensional functional, circulation, cost and massing information. ARCHPLAN uses prototype refinement to develop individual solutions from a generic prototype and may therefore be seen as an example for a routine design program. The user can freely move between four modules which are the site, cost and massing module (SCM), function module, circulation module, and structure selection module. Knowledge is stored in algebraic form and as heuristic rules. The program is implemented in common LISP with object-oriented extensions [14].

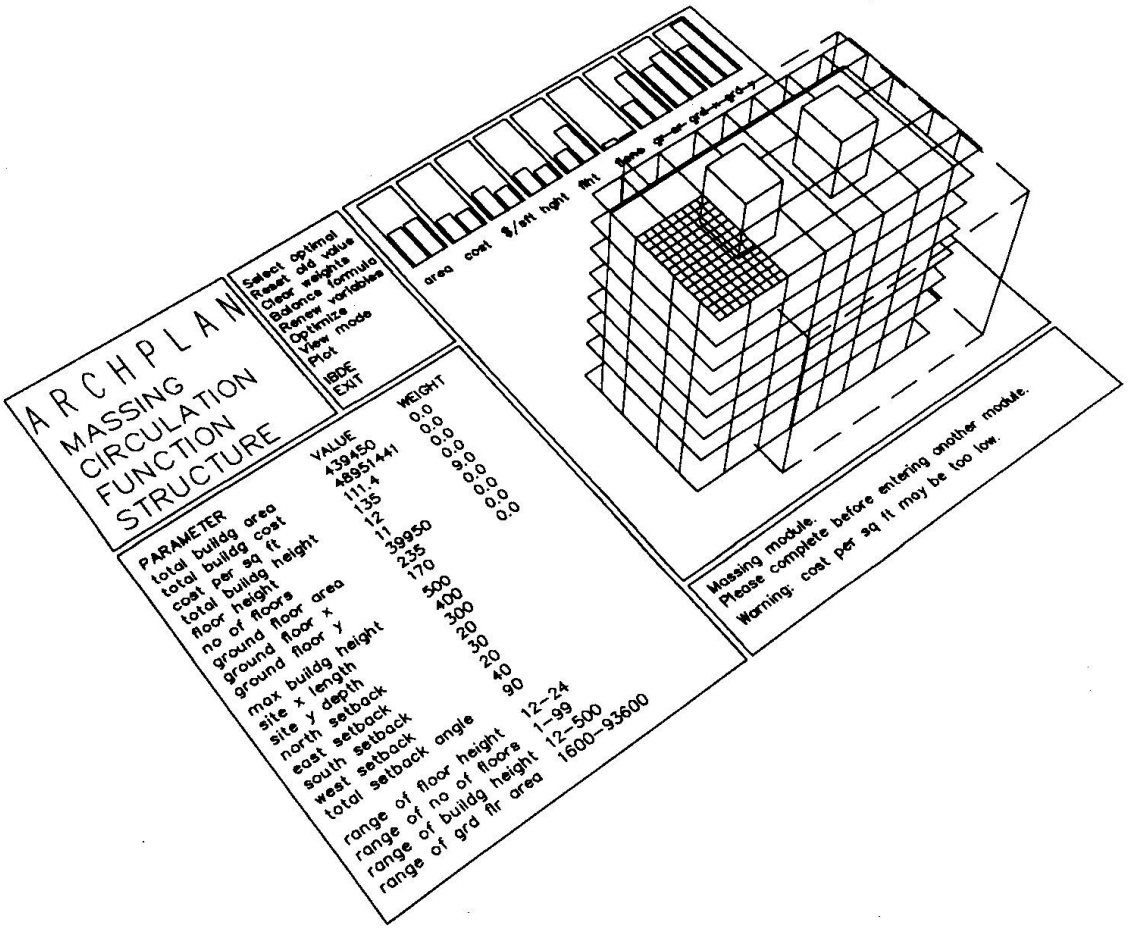


Figure 2: ARCHPLAN - designer's view of the extended site, cost, and massing (SCM) module

ARCHPLAN interaction begins with the *site, cost and massing module* (SCM). The designer finds a set of default values in an interactive window which can be modified at will. After a building site is determined, preliminary design begins with the development of a massing model that will accomodate a given budget and a range of

other parameters listed in Figure 2. Cost, site and massing options are inter-dependent concerns. Site characteristics are considered to be facts and therefore fixed, whereas building requirements are more flexible. The user describes the degree of commitment to a certain requirement, such as floor-to-floor height or ground floor area, by entering a certainty factor. The SCM module also contains simple optimization options: minimum cost, maximum daylighting, or a combination of the two. Cost data are based on the Means catalogue, a prominent summary of building cost data in the United States. The calculated cost is a total per square-foot number and includes interior and exterior construction as well as finishings. Design factors are represented as objects. Relations and constraints between factors are expressed within those objects [15].
Critical issues are the restriction to rectangular building shapes and the fact that relations between individual knowledge objects can be changed only by modifying the source code. User-defined additions of new considerations and non-monotonic reasoning for the design phase are planned for the next release.

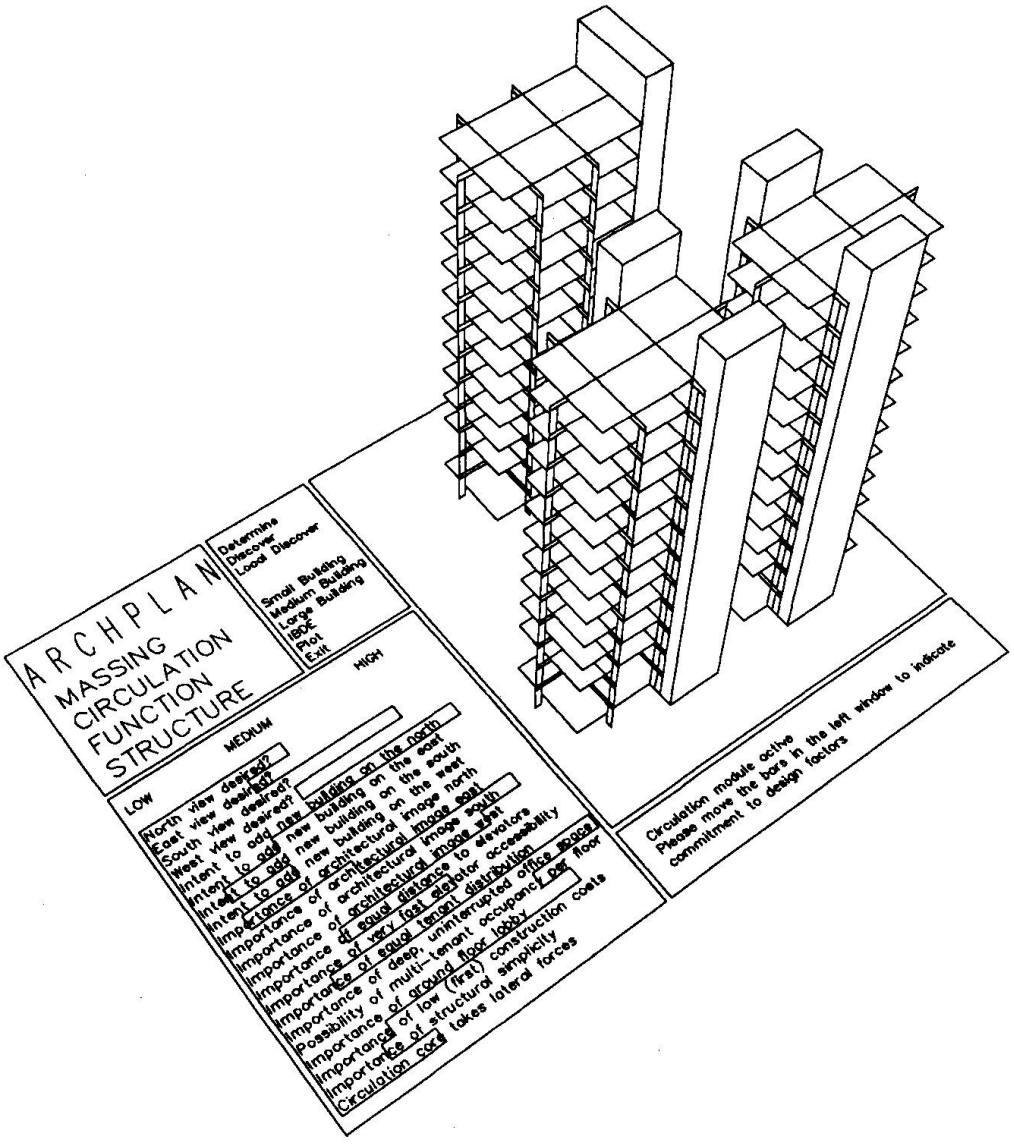


Figure 3: ARCHPLAN - designer's view of the extended circulation module

The *function module* assists in the vertical and horizontal distribution of building functions within the basic massing volume. Examples of building functions are office, retail, atrium, mechanical and parking space. Each function has particular requirements and affects the layout, appearance, and cost of the building. ARCHPLAN proposes a three-dimensional layout scheme which is displayed in solid or wire frame representation. Functional decisions are made and reflected locally, unless the constants in the global building description object are violated. In this case, the program backtracks and control is passed back to the SCM module where the designer can choose either to automatically adjust the design description to the information received from the function module or to implement changes manually. Critical issues are the shallowness of the knowledge used to determine three-dimensional functional layout. Although a number of existing buildings were used as examples, we did not discover all reasons behind a particular layout. The function module can therefore not explain its decision but executes them algorithmically.

The *circulation module* addresses the problem of moving occupants and equipment from floor to floor and within floors and guaranteeing the safe evacuation of the occupants in emergencies. Circulation also has a major impact on the internal functioning and on the architectural expression of a high-rise building. The two extreme cases for the placement of vertical circulation are the internal (service and elevator core in the centre of the building) or the external solution (service and elevator cores attached to the outside of the building). ARCHPLAN concentrates on creating vertical circulation proposals based on variations of these two prototypes. The user manipulates the relative importance of each factor leading to the final design proposal by sliding graphical bars, a more user-friendly but less exact interaction than typing in weighting factors as in the site, cost, and massing module. Figure 3 shows a view of the user interface for the circulation module.
Critical in this module is again the quality of knowledge that leads to a particular placement and size of vertical elements. Future versions must include user-definable criteria and their spatial consequences. The module does not provide for the decrease in the number of elevators in the upper levels. This calculation is handled by CORE, the elevator layout program developed by Flemming [16].

The *structure module* presents the user with a choice of eight structural systems, some of which may apply for the proposed design. The program compares the state of the design object with the characteristics of each structural type and decides which type is not applicable and available for the present design. Adjustments to the structural grid are still possible at this point. As these decisions are more competently handled in STRYPES and STANLAY, the structure module is available only when ARCHPLAN is used in stand-alone mode.

The program has gone through several revisions and is presently ported to a SUN 4 environment. In the absence of an established paradigm for architectural *design* programs, documentation of the experimental LISP source code is sometimes sketchy. ARCHPLAN is interesting for mainly two reasons:

- It makes architectural design knowledge explicit as it implements decision-making design processes in various representations. Thus, critique may be voiced openly and representation and knowledge manipulation methods for design can be improved.

- It addresses important user interface issues. The common display user interface employs direct graphical object manipulation techniques to give immediate feedback. Architectural decisions are visualized rapidly and facilitate the commencement of design from either very little or very detailed client information.

3. Part Three: On Criticism Between Processes

The introduction of criticism mechanisms between programs is based on the assumption that (a) one program alone will not be able to provide a solution to a complex problem, such as the design of a building, and that (b) a sequential execution of related knowledge-based programs will render the design process too cumbersome and does not allow for the exploration of enough alternatives.

Common representation of design objects and design processes is one enabling concept for the exchange of and the reaction to design criticism. It is assumed, as is the case in the IBDE project, that a number of programs in a knowledge-based environment (named controller, A, B, . . . in the following) are to interact, exchange information and criticism, in order to improve the final design. Design criticism may be grouped into several levels:

- The first level (on/off) is that program B, using input from program A, cannot start. Program B posts the message "Unable to commence". It is the responsibility of program A, another program, or the controller to react to the message and provide new input so that the computation can begin. This is a low level operation comparable to the range checking in data base input operations.
- The second level (impasse) is that program B, using partial input from program A, encounters an impasse in its reasoning process that it cannot resolve with the knowledge it has access to. Program B posts the message "Impasse". It is the responsibility of program A, another program, or the controller to react to the message and provide new input so that the computation can proceed.
- The third level (quality) is that program B reaches a solution that is acceptable, but has severe, definable drawbacks. Program B posts the message "Improve Quality (in a specific area)". It is the responsibility of the preceding program(s) or of the controller to provide new input.

The formulation of and the reaction to criticism becomes more difficult from the first through the third level. In the IBDE project, these levels could map to the following scenarios:

- First level. STRYPES, the structural system configurer (in the above example, program B) needs not yet existing input from ARCHPLAN (in the above example, program A). STRYPES posts the message "Unable to commence - respect constraints in structural grid - longest span less or equal to 35 ft - shortest span greater or eqal to 25 ft" on the blackboard, the controller initiates the execution of ARCHPLAN; it re-starts STRYPES once ARCHPLAN has completed a configuration.

- Second level. CORE, the space planner for the service core, attempts to fit the necessary elevator banks and service areas into the space defined by ARCHPLAN. CORE's knowledge base is unable to fit the required functions into the given area. Rather than continuing and creating an inconsistency or unilaterally updating the project data store, CORE posts the message "Impasse - core size must be greater or equal to 550 sqft - smallest side longer than 10 ft". The controller notifies ARCHPLAN which then should re-execute its circulation module. Once CORE's critique is satisfied, the impact on the other processes of changing the size and possibly the location of the elevator core in ARCHPLAN must be checked: if necessary, parts of STRYPES, STANLAY, CORE, SPEX, FOOTER, and PLANEX must re-execute as well.
- Third level. CORE has produced a layout for the elevator zone. A change in the city zoning regulations requires re-execution of the circulation and function module in ARCHPLAN. Although technically feasible, the orientation and individual layout of the core zone is no longer satisfactory for the new situation. This decision is either made manually after visual control of the core layout and the new situation, or by an additional design quality knowledge module in ARCHPLAN which regularly checks the architectural impact of solutions proposed by CORE and the other processes. ARCHPLAN posts the message "Improve Quality - re-configure core layout - core entrance should face east" on the blackboard. The controller restarts CORE, given the new information concerning orientation, and CORE proposes a new solution which is passed back to ARCHPLAN. As in the second level, impact on the other processes must be checked at this point. In re-executing processes, level one critiques have highest and level three critiques have lowest priority.

One major problem in this scenario is the circularity of critique, re-execution, and propagation of new results, possibly ending in endless loops of program execution. The settling of conflicts created by reactions to critique from other processes is not a mechanical, value-free activity, but involves judgement. This calls for high level decision knowledge in the controller or in the individual processes which should possibly allow for temporary inconsistencies in the building representation. We have not found a solution to this important problem yet. However, a system of this type would make any interdisciplinary cooperation of programs more realistic and possibly improve the results.

4. Conclusions

In writing an architectural preprocessor for engineering expert systems similarities and differences between the two disciplines become apparent. The same would probably hold true if more than two areas were involved. The first approach was to use knowledge representations that were already tested in architecture and civil engineering and to explore if there was a smallest common denominator for representing architecture and engineering design. With the exception of the Tartan grid as representation of geometry and frames as containers of knowledge, almost everything was different. This lead to the present architecture of IBDE, in which a global data store is the common depository of building data which is used as needed by the individual processes, and the common user display interface which interactively displays the content of the global data store. The advantage of this

loosely coupled approach is a high degree of freedom in the processes; a drawback is the inevitable loss of high level, application specific information which may be crucial for the meaningful completion of other processes. The existence of the blackboard eases this situation somewhat and the introduction of critique mechanisms between processes is another means to achieve the appropriate balance between global and local information preference. It shows, however, even more the need for a general building description language which would be capable of operating on a design and construction model of growing complexity.

5. Acknowledgements

The IBDE project is supported by the National Science Foundation of the United States in the Engineering Design Research Center at Carnegie Mellon University in Pittsburgh. Professors Steven Fenves, Ulrich Flemming, Chris Hendrickson, and Mary-Lou Maher are the developers of IBDE and the individual knowledge-based processes shown in Figure 1. Special thanks to research assistants Chen-Cheng Chen, Chia Ming Chen, and Shen Guan Shih for their development work in ARCHPLAN.

6. References

1. NEUFERT, Ernst. *Bauentwurfslehre - Grundlagen, Normen, und Vorschriften.* Viehweg Verlag, Wiesbaden. 1980.
2. SCHLEICHER, F. *Taschenbuch für Bauingenieure - Tome I und II.* Springer Verlag, Berlin. 1955.
3. HAYES, Patrick. *The Naive Physics Manifesto.* ISCCO Working Paper. 1978.
4. FORBUS, Ken. *Qualitative Process Theory.* Artificial Intelligence, Volume 24, 1984.
5. SCHANK, R. *Dynamic Memory - A Theory of Reminding and Learning in Computers and People.* Cambridge University Press, 1982.
6. AKIN, Ömer. *Psychology of Architectural Design.* Pion, London. 1986.
7. STANFORD University. *CIFE - Center for Integrated Facility Engineering.* Project description summary. Terman Engineering Center, Stanford University, Stanford, CA. 1989.
8. FENVES, S. U. Flemming C. Hendrickson M. Maher and G. Schmitt. *An Integrated Software Environment for Building Design and Construction.* Proceedings of the 2nd Int. Symposium on CAD in Architecture and Civil Engineering, ARECDAO '89, Barcelona, Spain. April 1989.
9. DUPAGNE, A. *Models of Building Representation Referred to Designing and Evaluating Process.* Proceedings of the 1st. Convegno Nazionale, Consiglio Nazionale delle Ricerche, Progetto Finalizzato Edilizia, Sorrento, 1989.
10. GIELINGH, Ir Wim. *Computer Integrated Construction, a major STEP forward.* Proceedings of the 2nd International Symposium on CAD in Architecture and Civil Engineering, ARECDAO '89, Barcelona, Spain. April 1989.
11. GERO, John, Maher, Mary Lou and Zhang, Weiguang. *Chunking Structural Design Knowledge as Prototypes.* The Architectural Computing Unit. Department of Architectural Science, University of Sydney, Australia. January 1988.
12. SCHMITT, Gerhard. *Microcomputer Aided Design.* John Wiley & Sons, New York. 1988.
13. FALTINGS, Boi. *Qualitative Kinematics and Computer-Aided Design* in: Proceedings of the Second IFIP WG 5.2 Workshop on Intelligent CAD, Cambridge, UK. September 1988.
14. SCHMITT, Gerhard. "ARCHPLAN: An Architectural Front End to Engineering Expert Systems", in Rychener, Michael (ed.) *Expert Systems for Engineering Design.* Academic Press, New York. 1988.
15. GOSLING, James. *Algebraic Constraints.* PhD Thesis, Department of Computer Science, Carnegie Mellon University, Pittsburgh. 1983.
16. FLEMMING, Ulrich, Coyne, R., Glavin, T., and Rychener, M. "A Generative Expert System for the Design of Building Layouts - Version 2", in Gero, John (ed.) *Artificial Intelligence in Engineering: Design.* Elsevier (Computational Mechanics Publications), New York. 1988.