KNOWLEDGE-BASED SYSTEMS PROGRAMMING FOR KNOWLEDGE INTENSIVE TEACHING

Henri Achten, Jan Dijkstra, Robert Oxman and Thijs Bax

Faculty of Architecture, Building, and Planning
Eindhoven University of Technology - Eindhoven, The Netherlands

Abstract

Typological design implies extensive knowledge of building types in order to design a building belonging to a building type. It facilitates the design process, which can be considered as a sequence of decisions.
The paper gives an outline of a new approach in a course teaching typological knowledge through the medium of Knowledge-Based Systems programming. It demonstrates how Knowledge-Based Systems offer an appropriate structure for analysing the knowledge required to implement typological design. The class consists of third-year undergraduate students with no extensive previous programming experience. The implementation language is AutoLISP which operates in the AutoCAD- environment. The building type used in the course is the office building. In order to become acquainted with both building type and programming in AutoLISP, information and instructions have been gathered and prestructured, including a worked out analysis and AutoLISP code.
Office plans are generated through use of the Knowledge-Based System. They are encoded in the form of frames. At the end of the course the students will have learned the basics of Knowledge-Based Systems, have been introduced to programming these systems, have analysed and reflected upon the design process, and gained insight into a specific building type.

Introduction

This paper presents a new approach in a course titled Knowledge-Based Systems in Building. The *Introduction* section will elaborate on the didactic issues of the course. The next section – *Typological Knowledge* – will deal with the knowledge aspect of a specific building type. In

Knowledge-Based Systems, a brief discussion introduces some basic terminology. *Teaching Typological Knowledge Through Knowledge-Based Systems Programming* discusses the specific approach towards the application of typological knowledge and Knowledge-Based Systems in the course. A few *Results* are shown from the class. The paper ends with a *Conclusion.*

The course – *Knowledge-Based Systems in Building* – consists of lectures and programming class. The lectures cover basic principles of Knowledge-Based Systems. Subjects covered by the lectures are, among others, knowledge, knowledge acquisition, knowledge representation, knowledge bases, representing designs and design knowledge, and modelling reasoning in design. Some techniques used in expert systems and other Al technologies will also be studied. The theoretical material of the lectures will be the subject of the exercise. In the exercise, students are required to develop a Knowledge-Based System that applies knowledge from the field of office buildings. The programming will be done in AutoLISP in the AutoCAD environment. Most students do not have extensive programming experience. Therefore, the start of the exercise is devoted to an introduction in AutoLISP programming.

The course is part of a series of four courses which are compulsory for students who want to graduate in Building Information Technology. The other three courses of the series are: *Advanced Architectural Modelling in CAD-systems; Advanced Architectural Representation in Multi-Media;* and *Information Systems in Building.* Students following Knowledge-Based Systems in Building are required to have followed *Information Systems in Building* first. The course is given by a member of the Building Information Technology Group and a member of the Design Methods Group. Approaches in the course reflect ongoing research. Part of the theoretical lectures are given by guest-lecturers.

Next to Building Information Technology students, the course is open to any student in architecture. The course is usually attended by third-year undergraduate students. It aims to introduce students to matters of knowledge acquisition, knowledge representation in information systems, and knowledge of design models. At the end, the students should have learned enough to be able to participate in projects that include knowledge acquisition for information systems development.

Typological knowledge

Implementing a Knowledge-Based System in an architectural context requires an architectural subject. In previous editions of the course, the case of kitchen-design was used. In order to provide a new angle on the course, the focus of the case was changed into typological design. Typological design.requires extensive knowledge of building types. Architects draw from this knowledge in order to design a building belonging to a type. Even when the design task does not ask for a specific building type, architects utilise knowledge present in building types to inform their activities.

It is beyond the scope of the exercise to investigate all the intricacies of a building type. However, it is possible to choose sensibly a type that is not too much complicated. According to Heath (1984), the complexity of the design task – in relation to building types – can be differentiated into three classes of buildings: commodity, symbolic, and systems buildings (similar distinctions by other authors are routine, innovative, and creative design). Of these three, commodity buildings are well-constrained building types about which exist a high degree of consensus. This class of buildings is relatively easy to analyse and produce – for architects at least – which leads to this class to choose a building type from. Office buildings are usually considered commodity buildings; therefore, they will be subject for implementation in the Knowledge-Based System.

The students have to become acquainted with the office building in a short space of time. Therefore, material concerning office buildings has been gathered for them beforehand from a literature survey (see the list included in References). Statements concerning major decisions in the design of office buildings are extracted. The statements are selected on their application to medium-height office buildings. They are presented in two ways: as a series of statements from each source (figure 1 a), and ordered on the basis of subject category (figure 1b). In the first way, students are able to extract knowledge from each source separately. In the second way, they are presented with the material ordered coherently per subject (e.g., building, organisation, spaces, structural system, circulation, HVAC, etc.). The references range from a period of 1973 to 1992, and also differ in their context (European and American). This diversity of sources is necessary since no single source covers all aspects of the office building. Different sources, therefore, add missing information. The material is ordered in a chronological fashion, which presents the most recent information first.

Neufert Bauentwurfslehre

1. Raumgruppen: Bürobereich: Zellenbüros 1-3 Personen mit AZUBI Arbeitsplätzen. Gruppenbüros bis 20 Personen mit AZUBI-Arbeitsplätzen. Großräume bis 200 Personen auf einer Ebene. Kombibüros mit Einzelarbeitsplätzen und gemeinschaftlich nutzbaren Gruppenbereichen.

2. Durch Bildschirmarbeitsplätze (n. Gottschalk 1990 ca. 20% Lit.s.BAP) und damit Computerterminals und Zusatsgeräte, steigt der Flächendbedarf für den Büroarbeitsplatz zunächst additiv um ca. 2-3 m2 auf ca. 15-18 m2 an.

3. 75% der täglichen Arbeit findet am "Engeren und Erweiterten Arbeitsplatz" statt... Notwendige Arbeitskontakte, sowie kollektiv genutzte Einrichtungen sind von Bedeutung. Daher die Forderung einer Nutzungsmischung aus Einzel- und Gruppenräumen, "Persönliche" und "Kollektive" Arbeitsplätze.

4. Nach Rosenauer bei 90% aller Burobauten in USA Hauptachse O/W, da tief eindringende Morgen- und Abendsonne stört. Südsonne ist durch Sonnenblenden leicht abzuschirmen. Nach Joedicke Hauptachse S/N Lage, damit Durchsonnung aller Räume gewährleistet ist. Nordräume nur bei flurfreien Anlagen vertretbar.

5. Einbündige Anlagen unwirtschaftlich, nur bei tiefen Büroräumen vertretbar.

Building

Neufert 4. Orientation E/W in USA, S/N in Europe

Neufert 6. Daylight useful as far as 7 m depth in room

Neufert 7C. Daylight possible through end-facades, reclining facade and light-coves

Neufert 9J. Daylight depth T = 1,5 Hf (height window)

Neufert 14. High-rise = building with highest floor higher than 22 m above ground-level

Integration 1. Office buildings typically have a Total Surface / Rental Surface ratio of 1.35

Offices 2. Four classes of depth in wings: shallow (4-5 m), semi-shallow (6-10 m), deep (11-19 m), and very deep (20-m)

Update 1. Trend towards shallow small offices

Update 3A. Half-shallow offices optimal depth (14-17 m)

Time 13. Area within 7,62-9,14 m of the facade best rentable, therefore offices usually are slabs, 18,29-21,34 m

Fig. 1a (left) - A sample of statements from Neufert (1992);
Fig. 1b (right) - A sample of statements ordered to subject category.

The material forms part of the knowledge base from which students implement the Knowledge-Based System. Before the analysis of the design process is discussed, some characteristics of Knowledge-Based Systems will be briefly introduced.

Knowledge-based systems

Knowledge-Based Systems attempt to capture and render operable human knowledge about some domain. They are typically implemented in computer systems. The goal is to assist users in executing tasks that usually have a problem-solving character. Through making knowledge comprehensible both to machine and human, Knowledge-Based Systems can provide understanding in these processes. The major characteristic of these systems, in contrast with conventional programs, lies in the separation between knowledge and reasoning. That is, the system is equipped with a general-purpose reasoning facility (inference engine) beforehand, which is capable of reasoning with pieces of knowledge it either has or acquires.

Knowledge can be defined as anything someone knows about a certain subject, or as a reasoning model through which data are used in order to obtain new data or a new reasoning model. Knowledge is distinguished in several ways, one of which is between heuristic and deep knowledge. Heuristic knowledge concerns the ability to solve complex problems, usually based on experience. Deep knowledge is explicit, and is usually only acquired through study. Deep knowledge often is derived from sources. Another distinction often made is between procedural and declarative knowledge. Procedural knowledge concerns the ’how’ of knowledge, declarative knowledge concerns the ’what’ of knowledge.

A Knowledge-Based Design System stores and uses knowledge of a specific domain. Part of this knowledge refers to relevant concepts, facts, and objects. The list of concepts can be compared with the declaration of variables in a conventional program. This declarative knowledge in a design system is static. It can be expanded with dynamic information; procedural knowledge.

A specific form of representing knowledge is by frames. A frame is a structured description that consists of a number of slots that can be filled-in. The slots represent properties of the specific object being represented. The content of a slot can be a value, or an index or reference or an instruction to find a value for that specific aspect of the object. The instruction to find a value typically is in the form of a program which runs when a specific situation occurs. The frame technique is a flexible approach to represent knowledge because the content of a frame is not fixed and depends on the use it is put to. An empty defined frame (that is, a frame with slots that are not filled-in) is usually referred to as a class frame, and a filled-in frame is usually referred to as an instance frame.

Teaching typological knowledge through knowledge-based systems programming

The frame is the basis of the Knowledge-Based System in the exercise. It represents the general description of the office building type. The slots and the order in which they are evaluated constitute the procedural knowledge of the office building type. The values of the slots constitute the declarative knowledge of a particular office plan generated through use of the system. Through analysing the design process of the office building, students construct the structure of the frame. By applying this structure to a specific case, they gain insight in the feedback-loops and recursions that occur when designing an instance of the office building type.

The Knowledge-Based System is going to be implemented in AutoLISP. The choice for AutoLISP is made for several reasons: the large amount of experience with both AutoLISP and AutoCAD; the reliable graphic control/output in AutoCAD; and the aim to teach

Knowledge-Based Systems programming as well as their use. Most students do not have extensive programming experience. Therefore, they are first provided with a tutorial ’AutoLISP’ programming. Next, they are shown how a design description can be represented in a frame-like notation, and how an instance can be realised. Through a worked-out example it is demonstrated how a Lispfile is constructed both from the analysis of the type as actually programmed. Subsequently, the students will work out a specific subtype.

Building types provide the knowledge required to fill the knowledge base of the Knowledge-Based System. The declarative aspects have been dealt with above. Instantiation of a building from a type is generally conceived of as a process of reduction. In this view, an abstract type holds all generalised knowledge necessary to generate a building by specifying its general aspects (Gero 1990). Although the relation between an abstracted type and a typical instance makes logical sense, it is altogether not clear how a complex entity such as a building design can be derived from a process of reduction only (without making the type-object unmanageably large). There is no direct mapping from the context-free abstract type to the site and program specific building.

For the analysis in the exercise, therefore, a stepwise approach to using typological knowledge is considered. Briefly put, this approach is based on two notions. The first is that instantiating a design occurs through a design process, and therefore in a sequence of steps in which decisions are taken. The second notion is that architects make extensive use of graphic representations to represent the state of the design object. Combined, these notions result in considering design as a process that can be described through a sequence of well-defined graphic representations with specific knowledge- contents concerning the design object. These specific representations are called "generic representations".

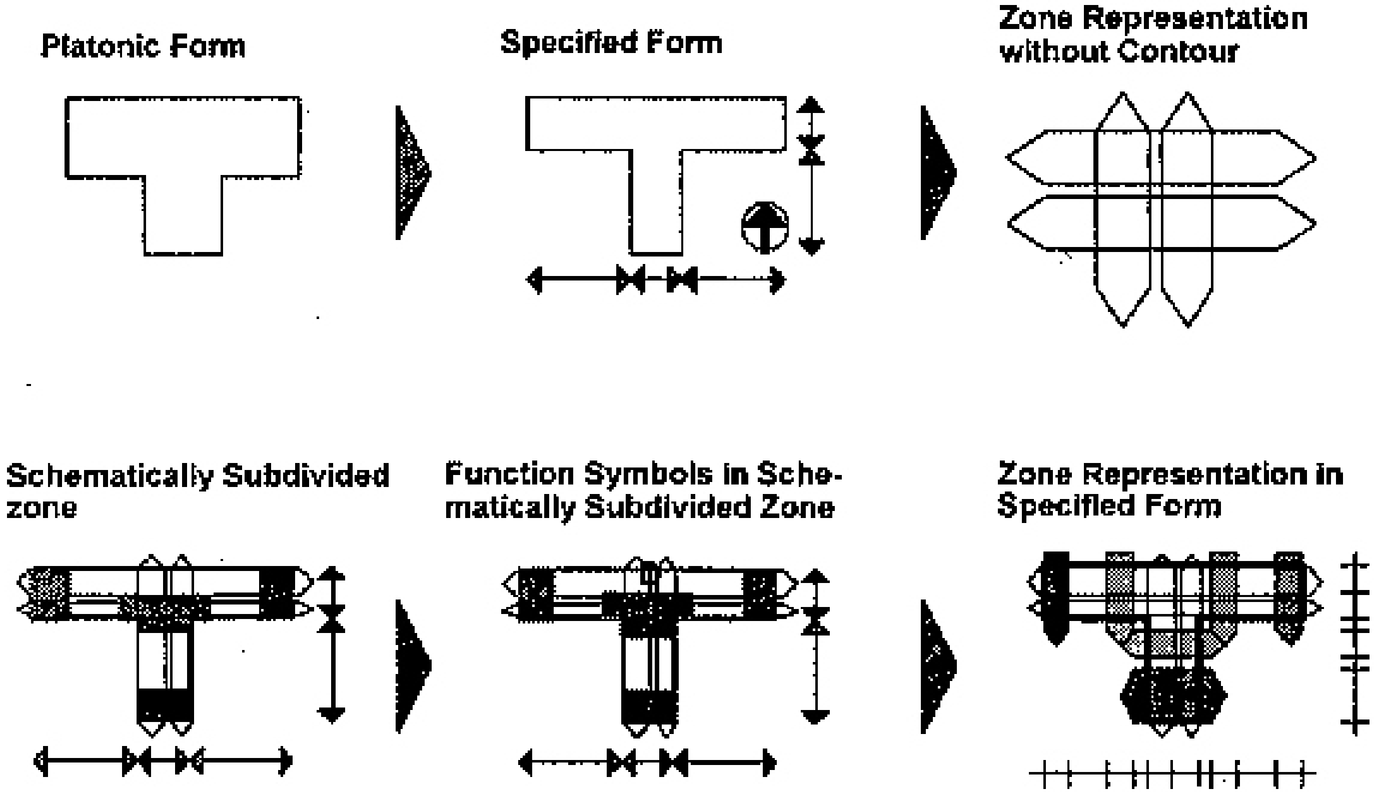


Fig. 2 - A sequence of generic representations of the office building

The students are provided with a small sequence of generic for a specific subtype of the office type: the T-Shape (figure 2). In each step of the sequence, the graphic representation encodes more aspects of the design object. This is done on the basis of the knowledge base provided. Together with a simplified program of demands and a site, it is possible to both denote the pieces of knowledge required to make decisions, as well as to denote the reasoning mechanisms necessary to come to these decisions. In the example for the students, the knowledge and the reasoning mechanisms are elaborated.

The sequence develops a design through the first stages of conceptual design. At the end of the last step, the design will be dimensioned, oriented, and positioned. The organisation will be defined through a zoning structure, and specific spaces will be located in the plan. Since the T-Shape is already worked out, students are required to analyse a different subtype of the office building. In the exercise, these are formed by the class of simple shapes: I, T, L, H, +, and . These shapes can be based on an orthogonal grid, which reduces their geometric complexity, and makes them therefore easier to implement. The steps of the sequence form the structure of the frame representation in the Knowledge-Based System.

The students are also provided with a Lisp-code of a Knowledge-Based System for the T-Shape which shows how the frame representation is constructed. The frame representation common to all subtypes of the office building is stored separately as a template-file in "office.frm". This template-file contains the sequence of generic representations in the form of slots. Each step in the process of generating an office plan is called through a subroutine in Lisp. These subroutines have to be programmed by the students. As in the example-code below, the new programs for each subtype (L, H, +, etc.) are substituted into the template-file (e.g., the L-Shape steps at the place of the dots "...."). In this way, the system can extend gradually its command over various subtypes.

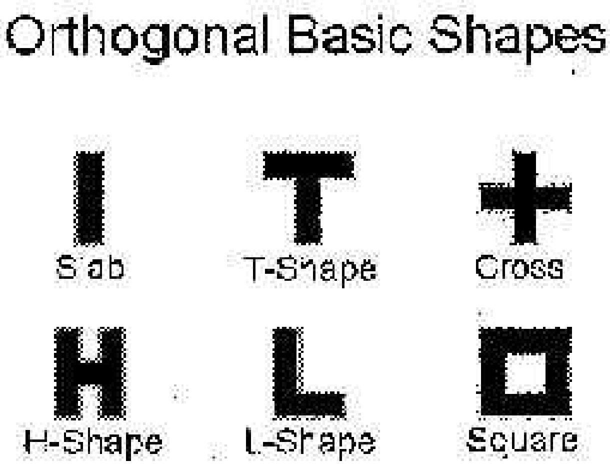
```
‘(
(Office: (myoffice)
(Is_a: (officetype))
(Arca: (def_area))
(No_stories: (def_no stories))
(Height_ storey: (def_height_story))
)
("TShape"
(Orientation: (def_T_orient))
(Dimensions: (def_T_dim))
(Insertpoint: (def_insert))
)
("LShape"
….
)
)
```

The frame as it is used, is a rather limited knowledge-representation. Much reasoning is dealt with in the Lisp-code that reads the template-file and makes an instance of an office building. In the exercise this means that students implement the instructions that are mentioned in a slot in a particular sublist of the template-file. For example, each office building has its own specific set of dimensions which are made and stored in a list in the slot named ’Dimensions’. The actual design in can be stored as a drawing file, but also as an instance frame. In this way, a case base of office designs can be made which is more compact than a similar set of drawing files. Regenerating a design from the case base follows from interpreting the slots of the instance frame.

Results

The students were required to have finished an analysis of the specific subtype they were dealing with, and a Knowledge-Based System that implements typological knowledge of the office building. The analysis formed the basis of programming the Lisp routines that were to control

the system; therefore, it had to take place first. During the period of the analysis, which would be in the first half of the trimester (nine weeks), students would be following lectures and learning programming in Lisp. In that period there is not much point in programming, which does not disturb the analysis, since it can be done by hand. The objective of the implementation, next to leading to a Knowledge-Based System, is also to show graphically the sequence of decisions taken in the process of generating office plans.
AutoCAD provides the graphic environment necessary for showing the visual aspects. Implementation of the interface is programmed at a rather low level, through showing slides instead of dialogue boxes. The implementation work showed here gives an example of how the specific approach is worked out.



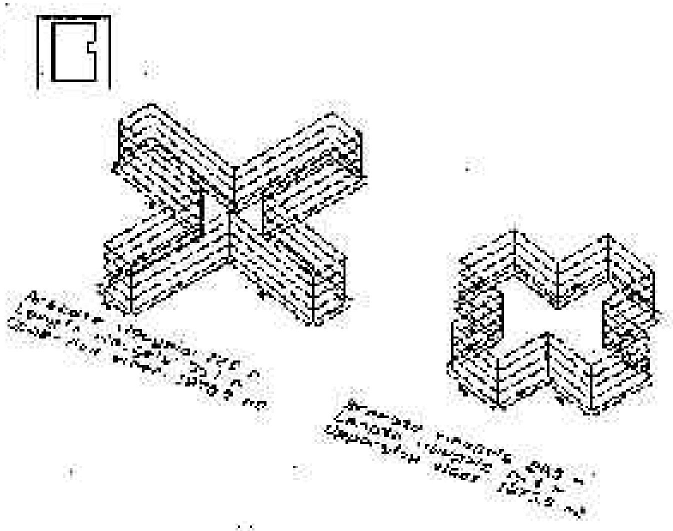
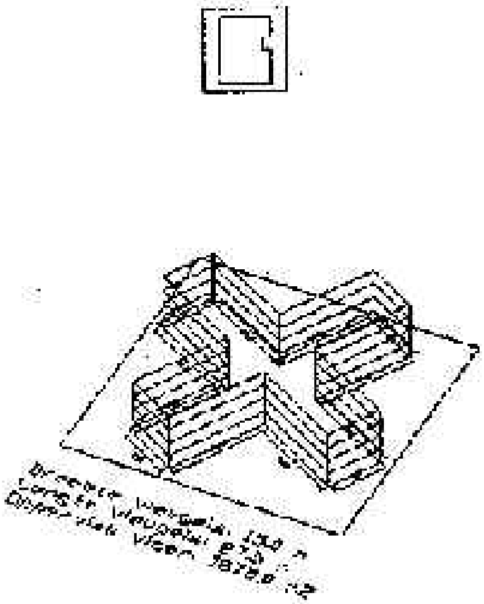


Fig. 3 (left) - The menu for choosing between subtypes
Fig. 4 (right) - The range of sizes depending upon wing-depth and length (work by B. De Haan (2) and P. Bremer (3)

When started, the system begins to fill in the slots of the class frame. It requires of the user to give site dimensions, required total area of the program, and the subtype (H, T, L, +, etc., figure 3) In the system, the number of stories of the building is dependent of the site, site conditions, and the program. It is able to rule out improbable combinations (e.g., very small surface areas and high buildings, or too few stories which would cause the building to extend over the boundaries of the site) from which the user can choose. Storey height and orientation are next required.
On the basis of various classes of wing depth distinguished in the literature, the user can choose among various options. Choosing, for example, the medium-deep class (12- 20 m) wing depth, results in a range of options for the length of the wings. The system displays the minimum and maximum lengths of the wings (figure 4). Through choosing an ultimate wing depth, the dimensions are set. All decisions are provisional; later in the sequence it may appear that other dimensions will do better. Feedback loops need to be identified and introduced in the appropriate places

to assume new dimensions.



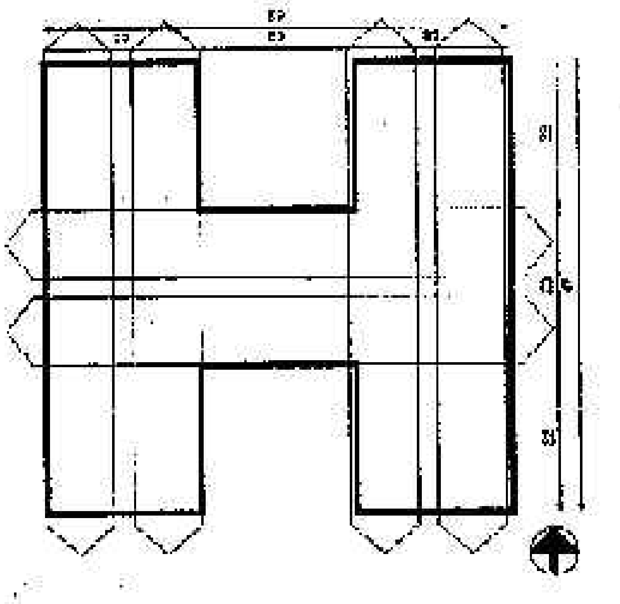


Fig. 5 (left) - The dimensioned building located in the site
Fig. 6 (right) Zoning structure in building layout
(Work by P.Bremen (5) and B.De Haan (6))

The icon in the upper left corner of figure 4 represents the specific generic representation the system is working out. In this way, the designer keeps track of his position in the design process.
When the shape has been defined, it has to be located in the site. After this, the exterior form of the building has been defined, as well as its orientation and place in the site (figure 5). The number of stories and storey-height are known. Choosing a zoning type (single, double, triple, and mixed zoning) is part of the next step. Several students got as far as implementing the choice between several zoning types and implementing it in the building layout (step four in the sequence of figure 2). An example of showing the zoning in the layout is provided in figure 6.

Conclusions

The prestructuring of the analysis through generic representations proved helpful in understanding the sequence leading to a specific design. It

also supported a modular approach in developing the subtypes of the office building. The static structure of the frame was fairly easily defined on the basis of the analysis.
Students reported that they had gained insight into the office building type with which most of them were not very familiar beforehand. Some information in the prestructured material provided for, was found redundant, and some inconsistencies between sources were detected.
Basically, students followed the worked out example of the analysis and re-used the knowledge provided there.
During the course, it appeared that AutoLISP is a rather poor environment for developing Knowledge-Based Systems. Students were required to spend a large amount of time writing relatively unimportant code to support the several routines necessary for implementing the system. In combination with the lack of programming experience this proved to be a bottleneck. There are two ways of easing the programming-task relative to the exercise. First, by placing the exercise in a more work-intensive context like a project instead of a course.
In that way students are able to spend more time working on the implementation. Second, by providing a frame notation that is formulated in more natural language terms. This can result in a less limited frame-representation. The exercise can then concentrate on implementing knowledge in a frame representation technique. The emphasis will be less on programming in AutoLISP. An example of a general syntax for such a frame is provided:

```
‘(
(Define_General_Concepts
(SiteWidth
(IS: Parcel_Width)
(PROMPT:"Give width of the site: ")
(VALUE: RANGE: 0 1000)
(ASK: YES)
)
(SiteLength
(IS: Parcel_Length)
(PROMPT:"Give length of the site: ")
(VALUE: RANGE: 0 1000)
(ASK: YES)
)
(SiteSurface
(IS:Parcel Area)
(VALUE: CALC: (* SiteWidth SiteLength))
)
(OfficeSurface
(IS: Office Area)
3 "3.40 m"
4 "3.70 m"
5 "4.20 m"
DEFAULT: 3)
(ASK:YES)
)
(No Stories
(IS:Office_No_of_Stories)
(VALUE:FROM:Floorarea)
)
(Floorarea
(IS: Office_Floor_Area)
(VALUE: WHILE
DEP_OF:OfficeSurface
COND: (< No_Stories 8)
(> Floorarea 800))
(ASK: NOT)
)
)
("T"
(Orientation
```

```
(PROMPT: "Give required floor area: ")
(VALUE: RANGE: 0 10000) (ASK: YES)
)
(BasicForm
(IS:Office_Type)
(PROMPT: "What platonic shape ")
(VALUE:CASE:T"T-Shape"
H "H-Shape"
L"L_Shape"
K "K-Shape"
R "Square"
DEFAULT:T)
(SHOWSLIDE: "ORTHOTYP")
(ASK: YES)
)
(StoreyHeight
(IS: Office_Height_Of_Stories)
(PROMIPT: "Give the storey height: ")
(VALUE: CASE: 1 "3.00 m"
2 "3.10 m"
(IS:Office_Orientation)
(PROMPT: "Give orientation: ")
(VALUE:CASE:LL RR LU RD)
(SHOWSLIDE: "TSLIDE")
(ASK: YES)
)
(Dimensions
(IS:Office_Dimensions)
(VALUE: DIM_LIST:
DIM_COND: (= A E C)
(= B D))
(SHOWSLlDE: "SLlDE")
(ASK: YES)
)
(InsertionPoint
(IS: Office_InsPnt)
(VALUE: BECALC)
)
)
)
```

References

For the sake of completeness, the sources used in the course for the knowledge base of office buildings are included. After that, the references of the paper are stated.

Bailey, S. 1990. *Offices – A briefing and design guide.* London: Butterworth Architecture.

Bovill, C. 1991. *Architectural Design – Integration of Structural and Environmental Systems.* New York: Van Nostrand Reinhold.

Chiara, J. De, Callender, J.H. (eds). 1980. *Time Saver Standards for Building Types.* MacGraw-Hill.

Duffy, F., Cave, C., Williams, B., Worthington, J. 1981. "Buildings Update - Offices Part I." *Architect's Journal*, 1 1 Nov:951-964

Hoke jr, J.R. (ed). 1988. *Architect's Room Design Data Handbook.* John Wiley & Sons.

Neufert, E., Neufert, P. [bearb.] 1992. *JVeufert Bauentwurfslehre.* Braunschweig/Wiesbaden: Friedr. Vieweg &. Sohn Verlagsgesellschaft mbH.

Peters, P. 1973. *Entwurf und Planung – Verwaltungsbauten.* MOnchen: Verlag Georg D.W. Callwey.

Stichting Bouwresearch 1985. *Schachten als kern voor hoge gebouwen?* Publ.nr. 120.

References of the paper:

Coyne, R.D., Rosenman, M.A., Radford, A.D., Balanchandran, M., Gero, J.S. 1990. *Knowledge-Based Design Systems.* Addison-Wesley

Gero, J.S. 1990. "Design Prototypes: A Knowledge Representation Schema for Design." Al Magazine, Winter

Heath, T. 1984. *Method in Architecture*, John Wiley & Sons Ltd.

Schmitt, G. 1993. *Architectura el Machina – Computer Aided Architectural Design und Virtuelle Architektur*, Vieweg